



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

# Callback User's Guide

[Enable Status Notifications](#)

5/1/2025

# Enable Status Notifications

## Modified in 8.5.211

Version	Update
8.5.105	The Callback service can now publish notifications to GMS that distributes these notifications to the target specified in the callback's service request, and consequently, to the subscribers of these notifications. The possible targets can be an ORS session of an existing GMS service (orscb notification type) or any URL (httpcb notification type).
8.5.107	<p>You can now receive two types of notifications: Callback SCXML and additional GMS Callback notifications.</p> <div> <p><b>Important</b></p> <p>By default, this feature is turned off for all callback services.</p> </div>
8.5.211	GMS can now send a notification reminder event before the callback is dialed.
8.5.232	<ul style="list-style-type: none"> <li>The <code>_cbe_on_dial_done</code> event is now sent for each dial request, not just one time.</li> <li>The <code>_cbe_on_service_exit</code> event is always sent at the end of the Callback strategy before the subscription is removed. Its parameters, such as the <code>c_last_dialed_number</code> parameter, are set in the different states of the strategy, according to the status of the Callback.</li> </ul>

To enable Callback Status Notifications (SCXML), you can either:

- **Enable** the Default Status Notifications (from SCXML).
- Create a Transaction Event object that **overwrites** the list of default notifications and assign it to your Callback Service. You can configure additional GMS Callback Status notifications by using the Transaction List entries which override the defaults. In that scenario, the notifications will only report the events specified in this Transaction List.
- Add notifications parameters to your Callback Services query.

Callback will send the notification events and provides two subscription modes to receive them:

- `subscribe_notify`—Callback subscribes for your application to the notifications.
- `notify`—Your application must subscribe to receive events.

## Enable Default Status Notifications in a Callback Service

Callback Later

Q \_status x Select All + Add New Delete Advanced Parameters Collapse All Refresh

Name	Value	Description
General (4)		
_status_notification_type	httpcb	Both httpcb and orscb can be configured to send out status notifications
_status_notification_target	http://<host>:1664/test	Name of the notification target that receives status notifications. If notification type is orscb then ORS session id is required. If notification type is httpcb then target is the url of the application.
_status_notification_provider		Name of the notification provider to be used for status notifications. If left blank default provider information is used.
_enable_status_notification	subscribe_notify	defaults to false. If set to subscribe_notify callback application will subscribe for status notification on behalf of the client; if set to to_notify it is assumed that client has already subscribed for status notifications. This setting will override setting in GMS events transaction list object.

To receive default callback status notifications (SCXML), open the Service Management User Interface and navigate to your Callback Service (in the **Configured Services** panel).

Enable **Advanced Parameters** and configure the following options in the **General** section:

- \_enable\_status\_notification= subscribe\_notify
- \_status\_notification\_type= httpcb (or orscb)
- \_status\_notification\_target = Target URL (or the ORS session id if \_status\_notification\_type = orscb)

You can add the following additional parameters to your Callback queries:

- \_status\_notification\_debug = false—Set to true to enable the debug mode for notification.
- \_status\_notification\_language = <language> where the language matches one of the supported languages used for **push notifications**.

The \_status\_notification\_debug option defines the URL where the notifications will be pushed using HTTP POST requests.

### Tip

The orscb notification type should be used for advanced ORS customization only.

## Enable Reminder Notifications

If you enabled default status notifications, you can also enable the Reminder Notifications in your Callback service.

- Configure `_enable_notification_reminder` to `true` and, by default, the system will send the `_cbe_on_callback_reminder` notification event 300 seconds before the dial time of the call.
- You can change the value of the `_notification_reminder_buffer` option to get the reminder notification earlier or later. The default value is 300 seconds.

### Important

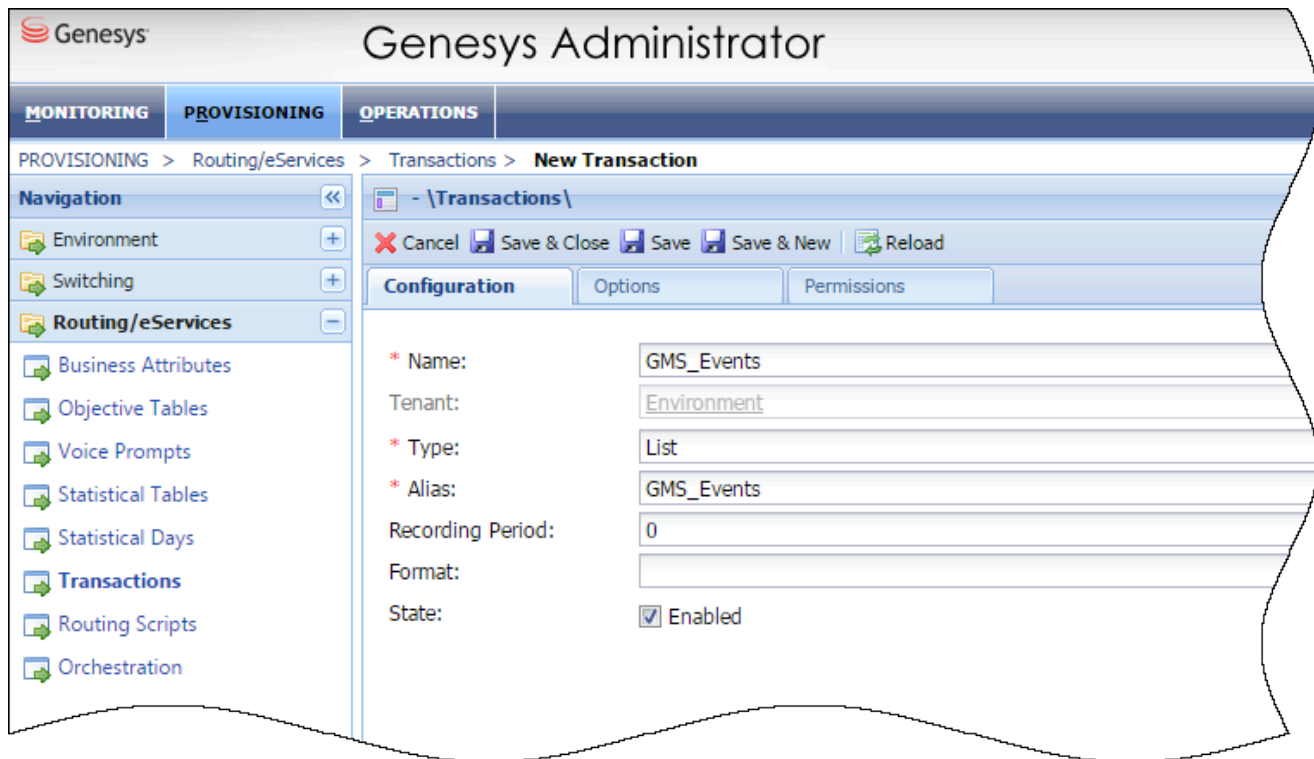
The time when the reminder is sent depends on the URS Estimated Wait Time (URS EWT) of the callback. You can get the URS EWT value by checking the **callback's position in queue (ewt)** using the callback API.

The Reminder feature periodically checks the EWT retrieved from URS for each call.

- If `URS EWT < _notification_reminder_buffer`, the Reminder feature sends the reminder event status notification.
- If not, depending on the EWT's value, the Reminder feature schedules the next check for the call:
  - Every 30 seconds if URS EWT is not defined,
  - Every 45 seconds if `URS EWT < 600` seconds,
  - Every 300 seconds if `600 seconds < URS EWT < 3600` seconds
  - Every 1800 seconds if `URS EWT > 3600` seconds

**Limitation:** The frequency of the Reminder periodical checks is not configurable.

## Overwrite Default Notifications with a Transaction List



Start by defining a Transaction List object that includes the notifications and the associated events triggering notifications.

Open Genesys Administrator. In PROVISIONING > Routing/eServices > Transactions, click **New** to create the GMS\_Events list.

In the **Options** tab, create a properties section with:

- `_enable_status_notification = subscribe_notify`
- `_status_notification_provider = <customerprovider>` or blank for default provider
- `_status_notification_type = httpcb`
- `_status_notification_target=<Target URL>`
- `_status_notification_debug= false`
- `_status_notification_language = <language>` where the language matches one of the supported languages used for **push notifications**.

Then, create a section for each subscribed event and define the data that your application needs to receive in the notification event.

- `notify_params`—The comma-separated list of callback parameters to retrieve. See the **reference** to get

the list of parameters that can be retrieved. Note that you can also retrieve some specific user data there in addition to callback parameters.

- `notify_custom`—(Optional) A JSON object of the custom attached data to send in the notification in addition to the callback parameters set in `notify_params`.

### Tip

- Either click **New** to add the following options or copy the source below to a `GMS_Events.cfg` file that you can import in your Transaction List.
- You do **not** have to include all the events listed below.
- The `notify_custom` parameter should suite your use case or can be removed if not needed.

In the XML sample below, `c_target` must match `c_target` as provided by URS.

## Enable Status Notifications

---

```
[properties]
_enable_status_notification = notify
_status_notification_provider =
_status_notification_type = httpcb
_status_notification_target =<your URL>

[_cbe_on_service_create]
notify_params = _service_id, _service_name, _customer_number, _urs_virtual_queue
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_virtual_ixn_create]
notify_params = _service_id, _service_name
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_target_found]
notify_params = _service_id, _service_name, c_target, _urs_virtual_queue
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_dial_init]
notify_params = _service_id, _service_name, _customer_number, c_dialed_number
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_dial_done]
notify_params = _service_id, _service_name, _customer_number, c_dialed_number, c_call_result, c_call_num_attempt
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_connect_treatment_start]
notify_params = _service_id, _service_name, _customer_number, _vq_for_outbound_calls, c_dialed_number
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_customer_queued]
notify_params = _service_id, _service_name, _customer_number, _vq_for_outbound_calls, c_dialed_number
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_route_to_agent]
notify_params = _service_id, _service_name, _customer_number, _urs_virtual_queue, c_agent_id,
c_agent_extension
notify_custom = {"name1":"value1", "name2": "value2"}

[_cbe_on_service_exit]
notify_params = _service_id, _service_name, _customer_number, c_last_dialed_number, c_termination_type
notify_custom = {"name1":"value1", "name2": "value2"}
```

---

## Enable Status Notifications

---

```
[_cbe_on_callback_scheduled]  
notify_params=_customer_number,_phone_number,_desired_time  
notify_custom={"state":"scheduled"}
```

```
[_cbe_on_callback_rescheduled]  
notify_params=_customer_number,_phone_number,_desired_time  
notify_custom={"state":"rescheduled"}
```

## Enable Status Notifications

The screenshot shows the Genesys Administrator interface. The top navigation bar includes 'MONITORING', 'PROVISIONING', and 'OPERATIONS'. The 'PROVISIONING' tab is active, and the 'Routing/eServices' sub-tab is selected. The main content area displays the 'GMS\_Events' configuration page. A handwritten note 'Import your configuration file' with an arrow points to the 'Import' button in the 'Options' menu. The 'Import' button is circled. Below the menu, there is a table with columns: Name, Section, Option, and Value. The table lists various events and their configurations.

Name	Section	Option	Value
cbe_on_service_create/notify_params	cbe_on_service_create	notify_params	__service_id, __service_name, __customer_number, __urs_v...
cbe_on_service_create/notify_custom	cbe_on_service_create	notify_custom	("name1": "value1", "name2": "value2")
<b>cbe_on_service_exit (2 Items)</b>			
cbe_on_service_exit/notify_params	cbe_on_service_exit	notify_params	__service_id, __service_name, __customer_number, __c_las...
cbe_on_service_exit/notify_custom	cbe_on_service_exit	notify_custom	("name1": "value1", "name2": "value2")
<b>cbe_on_target_found (2 Items)</b>			
cbe_on_target_found/notify_params	cbe_on_target_found	notify_params	__service_id, __service_name, __c_target, __urs_virtual_queue
cbe_on_target_found/notify_custom	cbe_on_target_found	notify_custom	("name1": "value1", "name2": "value2")
<b>cbe_on_virtual_kkn_create (2 Items)</b>			
cbe_on_virtual_kkn_create/notify_params	cbe_on_virtual_kkn_create	notify_params	__service_id, __service_name
cbe_on_virtual_kkn_create/notify_custom	cbe_on_virtual_kkn_create	notify_custom	("name1": "value1", "name2": "value2")
<b>properties (3 Items)</b>			
properties/status_notification_provider	properties	status_notification_provider	default
properties/status_notification_type	properties	status_notification_type	httpcb
properties/enable_status_notifications	properties	enable_status_notifications	true

Page 1 of 1

Displaying objects 1 - 21 of 21

6/17/2016

## Add the Event Transaction List to the Callback Service

Callback Now

Search Table General + Add New Delete **Advanced Parameters**

Name	Value	Description
_customer_number		<b>Request Parameter</b> - Customer's phone number. Can be used to match the call with service data when call direction is set to USERORIGINATED. Also used to match call direction is USERTERMINATED to establish connection with the call.
_service	callback	
_type	ors	
General (26)		
_attach_udata	single_json	Specifies the format in which the user data should be attached to the call prior to routing to agent. Select data_id to attach only the storage data (GMS_UserData). Select single_json will attach all user data as one json object (key: GMS_UserData). Select separate_keys to attach each user data as a separate key. Name of the key will be the same as the user data key.
_business_hours_service		Specifies a configured office-hours service. Request Desired Time is verified against the defined regular and specific calendar hours.
_call_direction	USERTERMINATED	When value is USERORIGINATED, this implies the user (device) will initiate call to be connected to the agent. If USERTERMINATED is specified, the enterprise will initiate the call to the specified target.
<input checked="" type="checkbox"/> _callback_events_list	GMS_Events	Name of the transaction list object which has the customized version of the events list. If value is set to empty string then default hardcoded set is used.

*Make sure to display Advanced Parameters*

Edit the **Advanced Parameters** in the **General** section of your Callback Service.

Set the `_callback_events_list` to the name of the Transaction List created above, `GMS_Events` in our example.

### Important

If you set other status notification parameters (`_status_notification_type`, `_status_notification_target`, `_status_notification_provider`) in your Callback service configuration or in your REST queries, they override the values set in the Transaction List object.

## Callback Status Notifications Events

Callback notifications consist of a JSON object which contains:

- **deviceId**—The custom id provided at subscription time by the subscriber.
- **message**—The notification message as defined in the Callback Events Transaction List.
- **timestamp**—The timestamp for this notification.
- **\_service\_id**—The ID of the service which sent the notification.
- **\_service\_name**—The name of the service which sent the notification.

The Notification events can include some additional attributes detailed in this table. Check the [Notification Event reference](#) to get the list of attributes available for a given notification.

Optional attributes	Description	Example
c_target	A selected target that specifies the agent/queue resource that will process this request.	"c_target": { "agent": "KSippola", "dn": "7001", "id": "Customer_Service", "place": "SIP_Server_Placel", "resource": "7001", "return": "target", "stat_value": "0", "switch": "SIP_Switch", "type": "GA", "vq": "SIP_VQ_SIP_Switch" }
c_agent_id	Equals to c_target.id.	"c_agent_id": "Customer_Service"
c_agent_extension	The agent's DN target (equals to c_target.dn).	"c_agent_extension": "7001"
c_dialed_number	The customer number.	"c_dialed_number": "5115"
c_call_result	Indicates the _genesys.ixn.callState state. Possible values are listed in the <a href="#">IxnlntfObjectModel</a> page.	"c_call_result": "0"
c_call_num_attempt	The number of outbound call dialing attempts.	"c_call_num_attempt": "1"
c_termination_type	The termination type, also known as Disposition value. It equals to the _CB_DISPOSITION value.	"c_termination_type": "COMPLETED.AGENT"

The following JSON code is an event sample where the notify\_custom parameter was configured to {"name1": "value1", "name2": "value2"}.

```
{
  "event_id": "_cbe_on_service_create",
```

```
"timestamp": "1467575991",
"_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
"_service_name": "samples_dev",
"_callback_state": "QUEUED",
"_customer_number": "5115",
"_urs_virtual_queue": "SIP_VQ_SIP_Switch",
"name1": "value1",
"name2": "value2"
}
{
  "event_id": "_cbe_on_virtual_ixn_create",
  "timestamp": "1467575992",
  "_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
  "_callback_state": "QUEUED",
  "name1": "value1",
  "name2": "value2"
}
{
  "event_id": "_cbe_on_dial_init",
  "timestamp": "1467575992",
  "_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
  "_service_name": "samples_dev",
  "_callback_state": "QUEUED",
  "_customer_number": "5115",
  "c_dialed_number": "5115",
  "name1": "value1",
  "name2": "value2"
}
{
  "event_id": "_cbe_on_dial_done",
  "timestamp": "1467576012",
  "_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
  "_service_name": "samples_dev",
  "_callback_state": "QUEUED",
  "_customer_number": "5115",
  "c_dialed_number": "5115",
  "c_call_result": 0,
  "c_call_num_attempt": 1,
  "name1": "value1",
  "name2": "value2"
}
{
  "event_id": "_cbe_on_connect_treatment_start",
  "timestamp": "1467576012",
  "_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
  "_service_name": "samples_dev",
  "_callback_state": "QUEUED",
  "_vq_for_outbound_calls": "VQ_GMS_REP_SIP_Switch",
  "c_dialed_number": "5115",
  "name1": "value1",
  "name2": "value2"
}
{
  "event_id": "_cbe_on_customer_queued",
  "timestamp": "1467576016",
  "_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
  "_service_name": "samples_dev",
  "_callback_state": "QUEUED",
  "_vq_for_outbound_calls": "VQ_GMS_REP_SIP_Switch",
  "c_dialed_number": "5115",
  "name1": "value1",
  "name2": "value2"
}
```

```
{
  "event_id": "_cbe_on_target_found",
  "timestamp": "1467576016",
  "_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
  "_service_name": "samples_dev",
  "_callback_state": "QUEUED",
  "_urs_virtual_queue": "SIP_VQ_SIP_Switch",
  "c_target": {
    "agent": "KSippola",
    "dn": "7001",
    "id": "Customer_Service",
    "place": "SIP_Server_Place1",
    "resource": "7001",
    "return": "target",
    "stat_value": "0",
    "switch": "SIP_Switch",
    "type": "GA",
    "vq": "SIP_VQ_SIP_Switch"
  },
  "name1": "value1",
  "name2": "value2"
}
{
  "event_id": "_cbe_on_service_exit",
  "timestamp": "1467576291",
  "_service_id": "445-20e740d3-8458-43d6-834d-3713c3385bac",
  "_service_name": "samples_dev",
  "_callback_state": "QUEUED",
  "c_termination_type": "COMPLETED.AGENT_CONNECTED",
  "name1": "value1",
  "name2": "value2"
}
```

## Reference for Notification Events

Event Name	List of attributes specific to this event	When this event is triggered
<b>_cbe_on_service_create</b>	_customer_number _urs_virtual_queue	As soon as the callback service (ORS session) is started.
<b>_cbe_on_virtual_ixn_create</b>		When the virtual interaction is successfully created in URS.
<b>_cbe_on_target_found</b>	_urs_virtual_queue c_target c_agent_id c_agent_extension	When the callback has found the target and URS reports the target to ORS.
<b>_cbe_on_dial_init</b>	_customer_number c_dialed_number	When the dialing to the customer is started. Note: This behavior applies to both standard and preview callback.
<b>_cbe_on_dial_done</b>	_customer_number c_dialed_number c_call_result c_call_num_attempt	When the dialing result is known. Starting in 8.5.232, the _cbe_on_dial_done event is now sent for each dial request, not just one time.  <b>Important</b> Only <code>_genesys.ixn.callState</code> types related to dial tone will be mapped with the <code>c_call_result</code> attribute of the <code>_cbe_on_dial_done</code> event. Refer to the <code>IxnIntfObjectModel</code> to get the list of call states.
<b>_cbe_on_connect_treatment_start</b>	_vq_for_outbound_calls c_dialed_number	When the greeting treatment is started right after the successful CPD.
<b>_cbe_on_customer_queued</b>	_vq_for_outbound_calls c_dialed_number	In User Terminated scenarios, as soon as the onconnect treatment is over, the virtual interaction becomes routable and the customer is placed into a queue to wait for an agent.

Event Name	List of attributes specific to this event	When this event is triggered
<b>_cbe_on_route_to_agent</b>	_urs_virtual_queue c_agent_id c_agent_extension	When the call is transferred from Routing Point to the agent.
<b>_cbe_on_service_exit</b>	c_last_dialed_number _customer_number c_termination_type	In all exit scenarios. Starting in 8.5.232, the _cbe_on_service_exit event is always sent at the end of the Callback strategy before the subscription is removed. Its parameters, such as the c_last_dialed_number parameter, are set in the different states of the strategy, according to the status of the Callback.
<b>_cbe_on_callback_scheduled</b>	_desired_time _customer_number _v_queue	When a callback in SCHEDULE status is created.
<b>_cbe_on_callback_rescheduled</b>	_desired_time _customer_number _v_queue	When a callback in SCHEDULE status is re-scheduled.
<b>_cbe_on_callback_cancelled</b>	_desired_time _customer_number _v_queue	When the callback is canceled.
<b>_cbe_on_callback_status_updated</b>	_desired_time _customer_number _v_queue	When the _callback_state field is updated by a REST query. This can be due to ORS updates.

Event Name	List of attributes specific to this event	When this event is triggered
	_urs_virtual_queue	
<b>_cbe_on_callback_reminder</b> Added in 8.5.211		By default, you receive the reminder event 300 seconds before the dial time of the call. Configure <code>_enable_notification_reminder</code> to true to enable this event and change the value of <code>_notification_reminder_buffer</code> to get the reminder earlier or later.
<b>_cbe_on_callback_submitted</b>	_desired_time _v_queue _urs_virtual_queue	When the callback is submitted for ORS execution.
<b>_cbe_on_callback_resubmitted</b>		When the callback is re-submitted for ORS execution.
<b>_cbe_on_callback_submit_failed</b>		When submit for execution fails.
<b>_cbe_on_callback_processing_failed</b>	_desired_time _customer_number _v_queue	When the callback processing fails.
<b>_cbe_on_callback_queued</b>	_customer_number _v_queue _v_queue_for_outbound_calls	When the callback is successfully submitted and its state changed to QUEUED.