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.

Genesys App Automation Platform Reference Guide

Genesys Intelligent Automation 3.5.1

1/12/2022

Table of Contents

Genesys App Automation Platform Reference Guide

Welcome to the Genesys App Automation Platform Reference Guide.

This document details useful reference information pertaining to your Genesys App Automation Platform (GAAP) deployment.

Click on a link below to learn more:

- Database schema information
- Call Processing

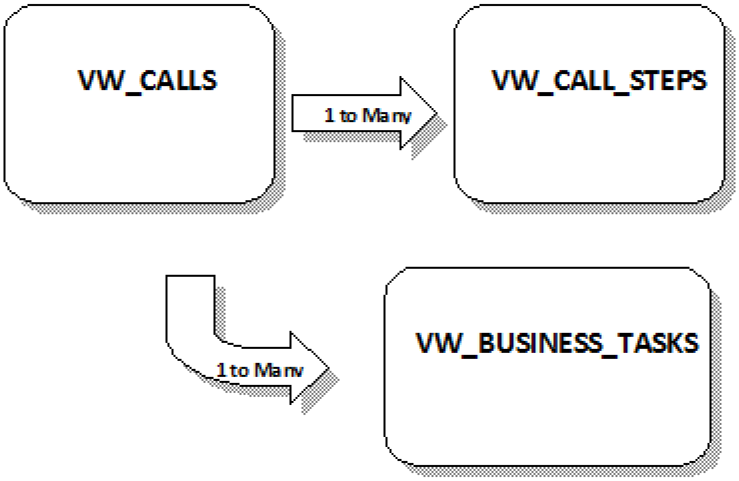
Database Views Schema

Call Reporting Database Views

The three main call reporting views within the reporting database are:

- VW_CALLS - Records each call made into the GAAP application.
- VW_CALL_STEPS - Records each step made by the caller.
- VW_BUSINESS_TASKS - Records which business tasks started and ended, along with details of each instance.

The diagram below describes the relationship between the three views:



These views are configured so they will not lock the database when they run. Genesys recommends that you always filter based on:

- **company_id**.
- **is_test_call**.
- **call_start_date**.

Refer to the Useful SQL Queries section to find queries that might be useful when working with the database

VW_CALLS

This view records each *call* made into the GAAP application.

Calls might be physical or not. For example, if a caller started a phone call to GAAP but was transferred out of GAAP to a routing strategy and then returned. In this case, two rows are recorded into this view.

Column	DataType	Length	Description	Example
call_id	bigint		Unique identifier for each call. This is an incrementing number.	10001
company_id	int		Link to GAAP company identifier. The company ID can be found in the GUI (look in **Administration > Companies**).	3
voice_platform_session_id	nvarchar	100	Media Platform Identifier for the call within GAAP (e.g. Genesys Voice Platform).	26A92695-9557-444A-A9B2-CCB4D71C1B69-1791
voice_platform_full_call_id	nvarchar	100	Unique identifier to link multiple GAAP interactions together within a single phone call.	10KMMSG8LL37TETGHTNQOF4KK80001NV
start_site_id	int		The module ID of the Inbound application from which the call started.	1
start_site_name	nvarchar	100	The name of the Inbound application from which the call started.	SIM Activation
call_start_time	datetime		The timestamp for when the call started.	2014-02-13 15:35:31.737
call_start_date	datetime		The date timestamp for when the call started (time is always **00.00.00.000**).	2014-02-13 00:00:00.000
call_start_hour	int		The hour of the day for when the call started, in 24-hour format.	15
call_end_time	datetime		The timestamp for when the call ended.	2014-02-13 15:36:02.640
call_end_date	datetime		The date timestamp for	2014-02-13 00:00:00.000

Column	DataType	Length	Description	Example
			when the call ended (time is always **00.00.00.000**).	
call_end_hour	int		The hour of the day when the call ended, in 24-hour format.	15
call_end_site_id	int		The ID of the last module accessed on the call.	3
call_end_site_name	nvarchar	100	The name of the last module accessed on the call.	Transfer to RS – With Announcement
call_end_block_type	tinyint	1	Type of the last block accessed on the call. Possible values are: • **1** – Start. • **2** – End. • **3** – Script. • **4** – Message. • **5** – Menu. • **6** – Custom Question. • **7** – Phone Transfer. • **8** – URL Transfer. • **9** – Recording. • **10** – Interceptor.	2
call_end_block_name	nvarchar	100	The name of the last block or step accessed on the call.	End Call
call_end_result	nvarchar	100	The resulting outcome	**hangup**

Column	DataType	Length	Description	Example
			description. Possible values are: • **error**. • **success**. • **hangup**. • **system hangup**.	
has_recent_failure	bit	1	Internal flag to determine if an error or recognition failure occurred in the callflow step/ block. This flag can be reset, as controlled by VUI preferences, if the succeeding Menu or Custom Question step/block is successful.	0
is_test_call	bit	1	Flag to determine if call is for the test or production version of the module. Values can be **1** for Test or **0** for production.	1
call_duration	int		Call duration in seconds.	30
cli	nvarchar	45	Calling Line Identifier (the number that the caller is calling from).	3100
dnis	nvarchar	45	Dialed number that is associated to the GAAP application.	1234
cluster_id	int		The ID of the GAAP server cluster that handled the call. Cluster information can be	1

Column	DataType	Length	Description	Example
			found in the GAAP GUI (look in **Administration > Clusters**).	
cluster_name	nvarchar	100	The name of the GAAP server cluster that handled the call. Cluster information can be found in the GAAP GUI (look in **Administration > Clusters**).	Default Voice Cluster
cti_fields	nvarchar	1000	Computer Telephony Integration (CTI) attached data. Fields are pipe separated, with each field represented as a key-value pair separated by colon. Literal colons, pipes, or percent symbols within keys or values are represented as **%c**, **%p**, or **%%**, respectively. You can define the list of allowed CTI variables that are included in this field in the GAAP GUI (look in **Administration > Default Server Settings**).	Segment:Gold
last_menu_block_type	tinyint	1	The type of the last Menu or Custom Question block accessed on the call. Possible values are: • **5** – Menu • **6** – Custom Question	5

Column	DataType	Length	Description	Example
last_menu_block_name	nvarchar	100	The name of the last Menu or Custom Question block accessed on the call.	If the callflow visited the following blocks: Welcome Message > Proceed With Activation Menu >Process Request Script > Successful Message > End Call This value is **Proceed With Activation Menu**.
cli_type	smallint	1	Representation of whether the caller is using a landline or a mobile handset. Possible values are: • **0** - Unknown • **1** - Landline • **2** - Mobile The list of CLI mobile number prefixes that are defined in server settings is used to determine if the CLI is a mobile number.	1
server_id	int		The ID of the GAAP server that handled the call. Server information can be found in the GAAP GUI (look in **Administration > Servers**).	1
start_channel	int		The channel in which the call started. Possible values are: • **0** - Unknown. • **1** – Voice.	2

Column	DataType	Length	Description	Example
			• **2** – Web. • **3** – Facebook. • **4** – Web with Voice.	

VW_CALL_STEPS

Each row in this view details a single block within the callflow that the caller progressed through.

Column	DataType	Length	Notes	Example
id	int		Unique call step identifier within call. This is an incrementing number.	1
call_id	int		See VW_CALLS.call_id.	10001
call_start_site	int		See VW_CALLS.start_site_id.	1
call_start_date	date		See VW_CALLS.call_start_date.	2014-02-13
cli_type	smallint	1	See VW_CALLS.cli_type.	1
company_id	int		See VW_CALLS.company_id.	3
is_test_call	bit	1	See VW_CALLS.is_test_call.	1
site_id	int		The ID of the GAAP module where this block/step (in the callflow) belongs.	265
site_name	nvarchar	100	The name of the GAAP module where this block/step (in the callflow) belongs.	Call Initialization
is_submodule	bit	1	Flag to determine if the module is flagged as an Inbound application or just a module.	1
block_type	tinyint	1	Block type indicator. Possible values are: • **1** – Start • **2** – End • **3** – Script • **4** – Message	1

Column	DataType	Length	Notes	Example
			- **5** – Menu - **6** – Custom Question - **7** – Phone Transfer - **8** – URL Transfer - **9** – Recording - **10** – Interceptor	
block_name	nvarchar	100	Name of the block or step in the callflow.	Start
block_detail	nvarchar	500	Internal field used to store additional information (if any) about the block.	For example, if this is a URL Transfer block, this field will show the module it will transfer to. (**Link to module : 5**)
start_time	datetime		Timestamp for when the callflow step/block was first visited.	2014-02-13 15:35:34.770
start_date	datetime		Date timestamp for when the callflow step/block was first visited (time is always **00.00.00.000**).	2014-02-13 00:00:00.000
start_hour	int		Hour of the day when the callflow step/block was first visited, in 24-hour format.	15
end_time	datetime		Timestamp for when the callflow step/block ended.	2014-02-13 15:35:34.780
end_date	datetime		Date timestamp for when the callflow step/block ended (time is always	2014-02-13 00:00:00.000

Column	DataType	Length	Notes	Example
			00.00.00.000).	
end_hour	int		Hour of the day for when the callflow step/block ended, in 24-hour format	15
duration	int		Duration, in seconds, spent within the step/block.	0
result	nvarchar	100	Resulting outcome description. Possible values are: • **error**. • **success**. • **hangup**. • **system hangup**.	Success
result_detail	nvarchar	100	Additional information relating to the result (for example, **transferred telephone number**).	tel://123456789
error_messages	nvarchar	500	Detailed error messaging (if any)	
wav_filename	nvarchar	200	Only applicable for a Recording block. This is the filename of the saved recording.	temprecording_123456.wav
is_recording_saved	bit	1	Only applies for a Recording block. This is the flag to determine if there is a wav file recording saved. Possible values are **0** if no recording is saved or **1** if a recording exists.	1

Column	DataType	Length	Notes	Example
recognition_type	int		Internal recognition type. Possible values are: • **0** – None • **1** – Menu • **2** – Custom • **3** – Defaults • **4** - Global	1
is_dtmf	bit	1	Flag that indicates if block/step is DTMF enabled. Value will be set to **1** if its DTMF enabled; otherwise, this value is **0**.	1
num_retries	tinyint		Count of no-match entries by caller in this callflow step/block. This field populates only if the caller leaves this block (i.e. doesn't hang up).	2
num_timeouts	tinyint		Count of no-input entries by caller in this callflow step/block. This field populates only if the caller leaves this block (i.e. doesn't hang up).	1
num_helps	tinyint		Number of times the *help* command was used in this callflow block/step. This field populates only if the caller leaves this block (i.e. doesn't hang up).	1
num_repeats	tinyint		Number of times the *repeat*	1

Column	DataType	Length	Notes	Example
			command was used in this callflow block/step. This field populates only if the caller leaves this block (i.e. doesn’t hang up).	
num_recovery_attempts	int		Number of times the callflow step/block was visited when the caller failed to be recognized and took the GAAP recovery route.	1
num_nbest	tinyint		Number of best possible matches (nbest) recognized in Automatic Speech Recognition (ASR). For an answer provided via DTMF, this will always have a value of **1**.	10
nbest_meaning_1	nvarchar	45	First highest match from the ASR against the SRGS grammar in context.	07712344401
nbest_rawanswer_1	nvarchar	100	First highest synonym match from the ASR against the SRGS grammar in context.	Oh seven seven one two three four four four oh one
nbest_confidence_1	int		Confidence scoring out of 1000 of first highest match against the SRGS grammar in context. For DTMF, this value is always **1000**.	700
nbest_slots_1	nvarchar	100	First highest slot content from the ASR against the SRGS grammar in context.	Type:Mobile|Number:12344401
nbest_recognition_type_1	int		Internal recognition type reference for the first highest match.	2

Column	DataType	Length	Notes	Example
			Possible values are: - **0** – None - **1** – Menu - **2** – Custom - **3** – Defaults - **4** - Global	
nbest_meaning_2	nvarchar	45	Second highest match from the ASR against the SRGS grammar in context.	07712344501
nbest_rawanswer_2	nvarchar	100	Second highest synonym match from the ASR against the SRGS grammar in context.	Zero seven seven one two three four four five oh one
nbest_confidence_2	int		Confidence scoring out of 1000 of second highest match against the SRGS grammar in context. For DTMF, this value is always **1000**.	10
nbest_slots_2	nvarchar	100	Second highest slot content from the ASR against the SRGS grammar in context.	Type:Mobile|Number:12344501
nbest_recognition_type_2	int		Internal recognition type reference for the second highest match. Possible values are: - **0** – None	2

Column	DataType	Length	Notes	Example
			- **1** – Menu - **2** – Custom - **3** – Defaults - **4** - Global	
nbest_meaning_3	nvarchar	45	Third highest match from the ASR against the SRGS grammar in context.	07712345401
nbest_rawanswer_3	nvarchar	100	Third highest synonym match from the ASR against the SRGS grammar in context.	Zero seven seven one two three four five four zero one
nbest_confidence_3	int		Confidence scoring out of 1000 of third highest match against the SRGS grammar in context. For DTMF, this value is always **1000**.	10
nbest_slots_3	nvarchar	100	Third highest slot content from the ASR against the SRGS grammar in context.	Type:Mobile|Number:12345401
nbest_recognition_type_3	int		Internal recognition type reference for the third highest match. Possible values are: - **0** – None - **1** – Menu - **2** – Custom	2

Column	DataType	Length	Notes	Example
			- **3** – Defaults - **4** - Global	
output_node_name	nvarchar	100	Name of the path that leads to this callflow step/block.	success
sequence_in_call	int		Given the list of callflow steps/blocks that were visited within the call, this is the position this step/block was visited within the sequence.	3
sequence_in_site	int		Given the list of callflow steps/blocks that were visited within the module, this is the position this step/ block was visited within the sequence.	1
persona_name	nvarchar	100	The name of the persona active during the current call step. An empty string is saved if using the default persona.	French
channel	int		The channel that is being used by the caller during this call step. Possible values are: - **0** – Unknown - **1** – Voice - **2** – Web - **3** – Facebook	2

Column	DataType	Length	Notes	Example
			- **4** – Web with Voice	

VW_BUSINESS_TASKS

Each row in this view details the business task that was processed (started, ended, and so on) within the callflow that the caller progressed through.

Column	DataType	Length	Notes	Example
id	bigint		Unique business task identifier within the call. This is an incrementing number.	10008
call_id	int		See VW_CALLS.call_id.	10001
voice_platform_session_id	nvarchar	100	See VW_CALLS.voice_platform_session_id.	26A92695-9557-444A-A9B2-CCB4D71C1B69-1791
voice_platform_full_call_id	nvarchar	100	See VW_CALLS.voice_platform_full_call_id.	10KMMSG8LL37TETGHTNQOF4KK80001NV
company_id	int		See VW_CALLS.company_id.	3
is_test_call	bit	1	See VW_CALLS.is_test_call.	1
start_site_id	int		The ID of the module where this business task belongs.	10
start_site_name	nvarchar	100	The name of the module where this business task belongs.	Payment By Full Balance
start_time	datetime		Timestamp for when this business task started.	2014-02-13 15:36:31.367
start_date	datetime		Date timestamp for when this business task started (time is always **00.00.00.000**).	2014-02-13 00:00:00.000
start_hour	int		Hour of the day for when this business task started, in 24-hour format.	15
end_time	datetime		Timestamp for when this business task ended.	2014-02-13 15:36:50.367
end_date	datetime		Date timestamp for when this business task ended (time is always **00.00.00.000**).	2014-02-13 00:00:00.000

Column	DataType	Length	Notes	Example
end_hour	int		Hour of the day for when this business task ended, in 24-hour format	15
duration	int		Duration, in seconds, between when the time business task started and ended.	19
name	nvarchar	100	Name of the business task (for example, **postal address lookup**).	Payment
outcome_category	tinyint	1	Outcome category identifier. Possible values are: • **1** – success • **2** – failure • **3** – CPFL (customer perceived failure) • **4** – Unknown • **5** - Hangup	3
outcome_description	nvarchar	100	Outcome category description.	Invalid balance
details	nvarchar	100	Additional information regarding the business task outcome.	Balance = null
call_start_site	int		See VW_CALLS.start_site_id.	1
call_start_date	date		See VW_CALLS.call_start_date.	2014-02-13

Useful SQL Queries

This page describes SQL queries that might be useful when working with the database.

Get complete call details for a given call ID

```
SELECT
calls.*, call_steps.*
FROM VW_CALLS calls
INNER JOIN VW_CALL_STEPS call_steps
ON calls.call_id = call_steps.call_id
AND calls.call_id = xxx
```

Get complete business task details for a given call ID

```
SELECT
calls.*, business_tasks.*
FROM VW_CALLS calls
INNER JOIN VW_BUSINESS_TASKS business_tasks
ON calls.call_id = business_tasks.call_id
AND calls.call_id = xxx
```

Get complete call details for a company, filtered by call date and module version

```
SELECT
calls.*, call_steps.*
FROM
VW_CALLS calls,
VW_CALL_STEPS call_steps
WHERE
        calls.company_id = call_steps.company_id
        AND calls.company_id = 1
        AND calls.is_test_call = call_steps.is_test_call
        AND calls.is_test_call = 0
        AND calls.call_start_date = '2017-01-01'
```

Call Processing

This page describes the logic applied at the end of calls. It also describes associated backlog processing, if necessary.

End-of-call logic

When a call ends, GAAP writes the call records to the configured database. A call ends under the following conditions:

- GAAP receives a hang-up event from MCP.
- There are no more blocks to process in the callflow.
- The call reaches the **End Call** block (can be configured either to disconnect or return to strategy).
- The call reaches a **Transfer** block.
- Session timeout due to no MCP requests during the configured timeout period.

  The **<session-timeout>** parameter value, found in the application **web.xml** file, defines this timeout period in seconds. In the example below, the session timeout period is set at 30 seconds:

```
<session-config>
 <session-timeout>30</session-timeout>
 </session-config>
```

Important

The **<session-timeout>** parameter resets to the default value each time you upgrade GAAP. Therefore, you must update this value after each upgrade if you do not want to use the default value.

Backlog processing

Important

Backlog processing of calls only occurs if the **Backlog.Processor.Enabled** setting is set to **true** and the hive-off process is not running. Backlog processing is paused during the hive-off process.

When a call ends, GAAP attempts to write the call to the database. If this process fails, GAAP sends the call to backlog processing.

The table below, **Handled Database Write Errors**, describes reasons for why the write might fail. If the write fails for another reason that is not described in this table, the call is unrecoverable and its data is lost.

Handled Database Write Errors

Database timeout
Connection pool exhaustion
The SQL Exception from the JDBC driver matches one of the following: • 08 – Connection Error • 66 – Driver Error • HY – Operation Cancelled • S0001 – SQL Login Failed

Configuring backlog processing

You can configure the settings below to periodically check the the backlog folder size:

- **Backlog.DiskSpaceMonitor.CheckIntervalMillis** - Specify, in milliseconds, how often to check the size of the backlog folder.
- **Backlog.DiskSpaceMonitor.WarningUsedMB** - Specify, in megabytes, the size that the backlog folder must exceed before a WARNING-level SMTP alert is sent.
- **Backlog.DiskSpaceMonitor.MaxUsedMB** - Specify, in megabytes, the size that the backlog folder must exceed before a MAJOR-level SMTP alert is sent. This is the size that, once reached, data is lost.

As of the 3.5.100.04 release, you must configure the following settings to specify which call-records errors are sent to the backlog:

- **Backlog.TreatAllFailuresAsBackloggable** - If `true`, all possible errors are sent to the backlog. If `false`, you can specify which errors are backlogged by configuring the following settings:
  - **Backlog.BackLoggableErrorsList.ExactMatch** - Specify an error code that must be matched. For example, `S0001`.
  - **Backlog.BackLoggableErrorsList.StartsWith** - Specify a comma-separated list of prefixes for accepted error codes. For example, `08,66,HY`.

Important

If you set **Backlog.TreatAllFailuresAsBackloggable** to `true`, GAAP sends all records to the backlog, even if these records cannot be backlogged. This movement consumes bandwidth until the records are manually removed from the backlog directory on the disk.

Sending backlog items to the database

If processing continues, the call data is serialized to XML, saved in an XML file on disk, and added to the internal file backlog list. You can use the setting **Backlog.Processor.MinItemAgeBeforeProcessingMillis** to specify, in milliseconds, how long GAAP must wait before trying to re-insert an item from the backlog back into the database.

Multiple backlog processor threads process any items that are added to the backlog. The **Backlog.Processor.ThreadCount.CallHistory** server setting defines the number of threads to use. Consequently, this value is also the maximum number of concurrent database writes from the backlog, regardless of database pool availability.

> **Important**
>
> The backlog shares the connection pool with the main application. Therefore, pool exhaustion might result if the **Backlog.Processor.ThreadCount.CallHistory** server setting is set too high.

Scenario

The following provides an example of how backlog processing functions.

Consider the following scenario:

- Maximum Pool Size = **100**
    - Main Application Connection Usage = **95**
    - Backlog Thread Count = **10**

The total number of items is **105**, which is five more than the maximum pool size. The excess items are sent to the backlog.

> **Important**
>
> GAAP does not guarantee whether excess items come from the backlog or the main application. Therefore, you must provide some headroom on your maximum pool size configuration. You must also account for the number of VUI servers and gauge whether the database server can cope with the number of total connections configured per server.

Once items are in the backlog, each backlog processing thread tries to grab an item from the backlog and re-insert it into the database. If no backlog items exist, the processing threads sleep for 20 seconds. This sleep value is not configurable.

Once a thread finds an item to process:

1. It attempts to move the file into the processing folder.

2. It de-serializes the file contents back into call data and deletes the XML file.

> Important
>
> If an error occurs in one of the first two steps, the file moves permanently to the *failed* folder and is no longer accessible to the backlog processing threads. This is the only scenario by which a file is moved to the *failed* folder and abandoned. In other words, the *failed* folder is not used for calls that cannot be re-saved. In that case, a backlog thread continues to put the call into the work queue without limit on failed attempts (unless the error does not match one of the handled errors referenced above).

3. It attempts to re-save the call data. One of the following occurs:
    - If the re-save is successful, the thread moves onto the next item in backlog.
    - If the save fails but it is a handled error, the thread adds a new backlog item for the call data. The thread that tried to process the item sleeps for 60 seconds (this value is non-configurable). The new item is not processed by another thread until the value of **Backlog.Processor.MinItemAgeBeforeProcessingMillis** has passed.
    - If the save fails and it is an unhandled error (not part of the list of handled errors, call data is lost.

Notes

- There is no limit on the the number of times a backlog thread can fail to process an item. Each time it fails, the item is added to the backlog again.
- The same code executes each time a backlog thread attempts to re-save an item. Therefore, a thread in the reporting connection pool is used each time an attempt is made.
- The only server setting that you can change at runtime is **Backlog.Processor.Enabled**. All other settings require a restart of the server.