Web Services and Applications Deployment Guide

Caching

4/16/2025

Contents

Caching

In Genesys Web Services 8.6, caching is done in **GWS API Service** and **GWS Platform Service**, however, the caching mechanism is different for each service which is noted as follows:

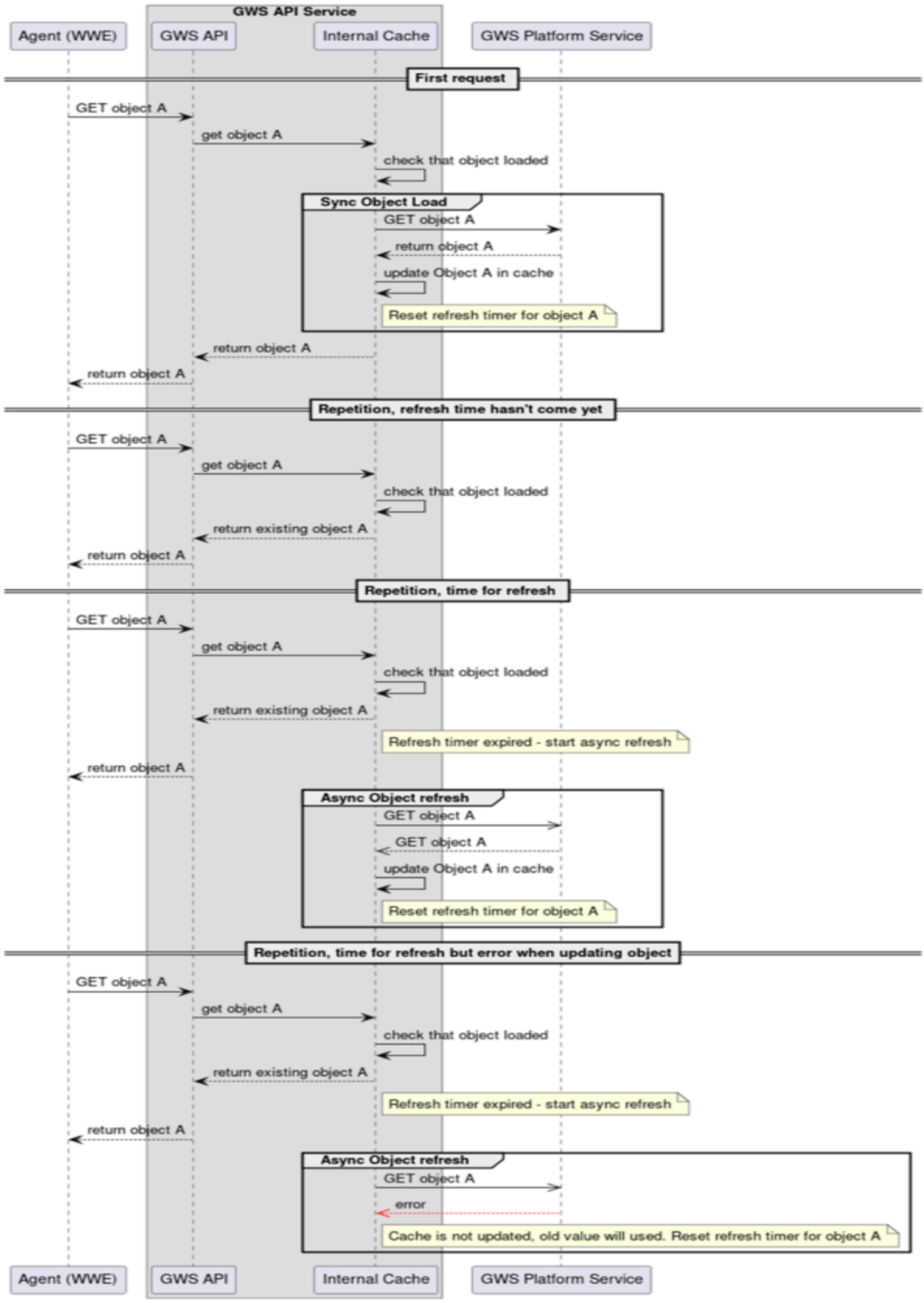
- **GWS API Service** implements its own small local cache.
- **GWS Platform Service** caches specific object types from Configuration database and keeps them in its own memory cache.

GWS API Caching

GWS locally caches a small subset of configuration data. The maximum size of the cached subset of data is limited and by default it is set to 1000 objects of same type (users, DNs, places, and so on). Adding additional objects will depend on the Heap size. Each object will be refreshed after the refresh timeout has lapsed (default is 5 min) in the background. Meanwhile, the object stored in cache will be returned immediately without waiting for the refresh completion. Since the object refresh is triggered by read requests, the *stale* objects will not be refreshed until accessed, which can significantly reduce the load on Configuration Database and Platform API.

For each object type, the refresh rate can be configured using the corresponding option in the **cachingSettings** section of the **application.yaml** file, for example, **placeTtl**, **usersTtl**, and so on. For full list of caching options, see Caching Settings.

GWS API Caching Sequence diagram


GWS API Service
Agent (WWE)
GWS API
Internal Cache
GWS Platform Service
First request
GET object A
get object A
check that object loaded
Sync Object Load
GET object A
return object A
update Object A in cache
Reset refresh timer for object A
return object A
return object A
Repetition, refresh time hasn't come yet
GET object A
get object A
check that object loaded
return existing object A
return object A
Repetition, time for refresh
GET object A
get object A
check that object loaded
return existing object A
Refresh timer expired - start async refresh
return object A
Async Object refresh
GET object A
GET object A
update Object A in cache
Reset refresh timer for object A
Repetition, time for refresh but error when updating object
GET object A
get object A
check that object loaded
return existing object A
Refresh timer expired - start async refresh
return object A
Async Object refresh
GET object A
error
Cache is not updated, old value will used. Reset refresh timer for object A
Agent (WWE)
GWS API
Internal Cache
GWS Platform Service

> **Important**
>
> As the contact center objects are cached at two levels, under normal circumstances, any object that is modified in Configuration Server will take a maximum of (API cache timeout + 1 second) to be reflected. If caching is disrupted for any unknown reason, then the Platform API's cache refresh will self-heal and restore the caching process.

> **Important**
>
> On top of caching, GWS API utilizes ElasticSearch to fulfill API requests that require searching or filtering operations. API Configuration Indexing nodes add new objects or update existing during re-indexation procedure that happens on regular basis by schedule that is 15 minutes by default (indexVerificationInterval configuration parameter). Hence, if new object has been added to ConfigServer or something has been updated, those changes will be visible by GWS API only when next re-indexation procedure is completed. That behavior is different in compare with 8.5 where changes are applied immediately in primary region where API SyncNode is deployed. Non-primary regions in 8.5 deployments behave the same way as 8.6 now.

GWS Platform API Caching

GWS Platform Service serves 'read API requests' using an internal in-memory cache or through Configuration Server. Note that not all object types are supported for caching. For the list of supported datatypes for which data is read directly from Configuration Server database, refer here.

Unsupported types and write operations (create/update/delete) are still handled through Configuration Server. All reads from database are done in subsequent manner through a single database connection by background processes. The background processes that support these requests are:

- A process for periodical full cache reevaluation that runs every 15 minutes (can be configured).
- A process for ongoing cache update procedure that runs every second picking up all changes in the database since the previous run, hence keeping the cache up-to-date. GWS Platform Service will retain and use existing cached information for the entire duration of any connectivity issues with the Configuration Database.

Supported object types for handling via in-memory cache are:

- CFGAccessGroup
- CFGActionCode
- CFGAgentGroup
- CFGAgentLogin

- CFGApplication
- CFGCallingList
- CFGCampaign
- CFGCampaignGroup
- CFGDN
- CFGDNGroup
- CFGEnumerator
- CFGEnumeratorValue
- CFGField
- CFGFilter
- CFGFolder
- CFGFormat
- CFGHost
- CFGPerson
- CFGPlace
- CFGPlaceGroup
- CFGRole
- CFGScript
- CFGService
- CFGSkill
- CFGSwitch
- CFGTableAccess
- CFGTenant
- CFGTransaction
- CFGTreatment

GWS Platform Caching Sequence diagram


GWS Platform Service
GWS API Service
Platform API
ConfigServer
Config DB
Service Initialization
Service instance started
Read configuration
Open connections to Database
Read all data from Tenants table
Compose Tenat objects
Read all data from <ObjectType> table
Compose <ObjectType> objects
Read ACL (permissions) data from table
Compose ACL objects
All data has been read from Database
Initialize dataset
Calculate VAGs
Calculate Permissions
Create initial data snapshot and write into cache
In-memory cache
Start backround cache update procedure
Schedule async cache refresh task
Open connection with ConfigServer
Service instance is ready to serve API requests
Update Cache
Check Database for changes (read HistoryLog)
no changes
Schedule next check task
Check Database for changes
There are changes
Recalculate updated objects
Recalculate affected VAGs
Update changed objects in cache
Schedule next check task
Refresh Cache
Read the latest record from HistoryLog
Read Tenant data
Compose Tenat objects
Read <ObjectType> data
Compose <ObjectType> objects
Read ACL (permissions) data
Compose ACL objects
All data has been read from Database
Initialize dataset
Calculate VAGs
Calculate Permissions
Stop cache update procedure
Read the latest record from HistoryLog
Recalculate updated objects
Recalculate affected VAGs
Create a new data snapshot and write into the cache
Data in the cache was swapped with a new one
Start backround cache update procedure
Schedule next async refresh task
Handle API Requests
API request to read objects
Apply request filters
Filter out objects by their permissions
API request to update object
Update object in Database
Write Delta object into HistoryLog
GWS API Service
Platform API
In-memory cache
ConfigServer
Config DB

Frequently Asked Questions

GWS API Caching

Do I have control over the cache refresh timer?

Yes, you do have control over refresh timeouts, you can configure them using Caching Settings.

Will I know how old the data is in the response? If the refresh fails, will I be notified in the next request that the data is outdated?

No, we don't provide any info on how old the data is since it might be composed from multiple data sources (caches) with different ages.

GWS Platform API Caching

What happens when a read API request is made and the configuration synchronization process fails more than once?

GWS Platform Service isn't ready until cache is populated once. After that, all read requests for supported data types are handled only via in-memory cache which is updated by background processes. If the update procedure fails then a full refresh is scheduled immediately (with delay in action). Also, cache is refreshed on a regular basis, every 15 mins.