



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 Rules System Deployment Guide

High Availability Support

12/19/2025

High Availability Support

Contents

- **1 High Availability Support**
 - **1.1 GRE**
 - **1.2 GRAT**

GRE

The Genesys Rules Engine (GRE) can be set up in a cluster in order to provide a highly available configuration. GRE is considered a critical path application because the execution of rules depends upon at least one node in the system being available. Since GRE is stateless, each rule execution request can be dispatched to any node in the cluster, and should a node fail, another node could execute the request.

The load balancer can be set up to dispatch requests to each GRE node at random, or in a round-robin fashion. There is no need to configure "session stickiness" as there are no sessions to maintain between rule execution requests. The load balancer should only route rule evaluation requests to a node that returns an HTTP 200/ SYSTEM_STATUS_OK, as described in GRAT Status below.

GRE Status

GRE has a `status.jsp` URL that can be used for a health check. The following statuses are available via `/genesys-rules-engine/status.jsp`.

Status	Response Text/Meaning
HTTP 503	<ul style="list-style-type: none">• SYSTEM_STATUS_CONFIG_SERVER_NOT_CONNECTED—Configuration Server is not connected (same as pre-8.5.2 response)• SYSTEM_STATUS_ENGINE_NOT_INITIALIZE D—Engine is not initialized• SYSTEM_STATUS_CLUSTER_SYNCING—Engine syncing with Cluster
HTTP 200	<ul style="list-style-type: none">• SYSTEM_STATUS_OK—Ready to take rule execution requests (same as pre-8.5.2 response)

GRAT

Unlike GRE, only one Genesys Rules Authoring Tool (GRAT) instance can be connected to a particular rules repository database at a time. GRAT is not considered a critical path application because it only handles the creation, editing and deployment of rules. If GRAT should fail, rule execution continues uninterrupted. Only rule editing becomes unavailable.

GRAT can be set up in a cold standby configuration. A standby GRAT can be installed as a mirror image on a separate machine and be configured to use the same configuration management application, same HTTP ports, and so on. Should the primary GRAT fail (hardware failure, network), the standby GRAT could be brought online quickly to restore service. Both the primary and standby GRATs can be connected to the same repository database; however, they should not be connected

simultaneously. The rule author would have to log in again and resume their activity.

When configuring a standby GRAT, use option `clear-repository-cache=true` for both the primary and backup GRAT instances. Setting this option to `true` can delay the startup of GRAT, since the cache must be rebuilt each time, but it ensures that it is properly synchronized with the rules repository database.

GRAT Status

GRAT has a `status.jsp` URL that can be used for a health check. GRAT's `status.jsp` always returns HTTP 200. The response text must be queried to determine if the GRAT server is up or down.

Status	Response Text/Meaning
HTTP 200	<ul style="list-style-type: none">• <code>SYSTEM_STATUS_OK</code>—GRAT server is up and running• <code>SYSTEM_STATUS_CONFIG_SERVER_NOT_CONNECTED</code>—GRAT server is not connected to Configuration server• <code>SYSTEM_STATUS_DB_INITIALIZING</code>—GRAT server is currently initializing local cache from repository Database. This can take several minutes for a large repository.• <code>SYSTEM_STATUS_DB_NOT_CONNECTED</code>—GRAT Server cannot connect to the repository database. Check the database status and/or check the database credentials that are specified in the DAP on the GRAT application object.• <code>SYSTEM_STATUS_UNKNOWN</code>—GRAT server is down. Check logs for more details.