# Genesys Rules System Deployment Guide

## Creating the GRAT Application Object in Configuration Manager

4/24/2025

# Creating the GRAT Application Object in Configuration Manager

## Purpose

To create the `Application` objects in Configuration Manager that will link the GRAT with Configuration Server. The GRAT requires two applications in Configuration Server: a server application and a client application.

## Procedure

## 1. Import the GRAT application template for the server.

## Import the GRAT application template for the server

To import the application template that is to be used for the server application:

1. In Configuration Manager, navigate to the `Application Templates` folder.
2. Right-click the `Application Templates` folder, and select `Import Application Template`.
3. Browse to the `templates` folder of the installation CD, and select the appropriate template for your version of Management Framework.
   - For Management Framework 8.1.1, select `Genesys_Rules_Authoring_Server_811.apd`.
   - For Management Framework 8.1 and earlier, select `Genesys_Rules_Authoring_Generic_Server_811.apd`.

- Click `OK` to save the template.

## 2. Import the GRAT application template for the client.

## Import the GRAT Application Template for the client

To import the template that is to be used for the client application:

1. Right-click the `Application Templates` folder.
2. Select `Import Application Template`.
3. Browse to the `templates` folder of the installation CD.
4. Select `Genesys_Rules_Authoring_Generic_Client_810.apd`.
5. Click `OK` to save the template.

# 3. Configure the server application.

## Configure the GRAT Application

To configure the GRAT server application:

1. On the `Tenants` tab, add all tenants that should be visible in the GRAT interface.
   a. In the `Server Info` section, configure a default listening port.
   b. On the `Connections` tab, add a connection to the Rules Engine application.
   c. On the `Connections` tab, add a connection to the Database Access Point.
   d. On the `Options` tab, configure log options.

### log

| Description | Valid values | Default value | Takes effect |
|---|---|---|---|
| **all** | | | |
| Specifies the outputs to which an application sends all log events. The log output types must be | • `stdout`—Log events are sent to the Standard output (`stdout`).<br>• `stderr`—Log events are sent to the Standard error output (`stderr`).<br>• `network`—Log events are sent to Message Server, | stdout | After restart |

| Description | Valid values | Default value | Takes effect |
| --- | --- | --- | --- |
| separated by a comma when more than one output is configured. For example: all = stdout, logfile | which can reside anywhere on the network. Message Server stores the log events in the Log Database. Setting the `all` log level option to the network output enables an application to send log events of the Standard, Interaction, and Trace levels to Message Server. Debug-level log events are neither sent to Message Server nor stored in the Log Database.<br>• `memory`—Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.<br>• `[filename]`—Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory. | | |
| **expire** | | | |
| Determines how many log files will be kept on disk. If set, expire specifies the maximum | Any number | (blank) | After restart |

| Description | Valid values | Default value | Takes effect |
|---|---|---|---|
| number of log files kept on disk. | | | |
| **segment** | | | |
| Determines whether a log output written to file is split in multiple segments. If it is, segment specifies the maximum size of each segment file. | Any number that represents the log size in megabyte | (blank) | After restart |
| **standard** | | | |
| Specifies the outputs to which an application sends the log events of the Standard level. The log output types must be separated by a comma when more than one output is configured. For example: | • `stdout`—Log events are sent to the Standard output (`stdout`).<br>• `stderr`—Log events are sent to the Standard error output (`stderr`).<br>• `network`— Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.<br>• `memory`—Log events are sent to | stdout | After restart |

| Description | Valid values | Default value | Takes effect |
|---|---|---|---|
| standard = stderr, network | the memory output on the local disk. This is the safest output in terms of the application performance.<br>• [filename]—Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory. | | |
| **trace** (not in application template by default) | | | |
| Specifies the outputs to which an application sends the log events of the Trace level and higher (that is, log events of the Standard, Interaction, and Trace levels). The log outputs must be separated by a comma when more than one output is configured. For example: trace = stderr, network | • stdout—Log events are sent to the Standard output (stdout).<br>• stderr—Log events are sent to the Standard error output (stderr).<br>• network—Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.<br>• memory—Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.<br>• [filename]—Log events are stored in a file with the specified name. If a path is not | stdout | After restart |

| Description | Valid values | Default value | Takes effect |
|---|---|---|---|
| | specified, the file is created in the application's working directory. | | |
| **verbose** | | | |
| Determines whether a log output is created. If it is, specifies the minimum level of log events generated. The log events levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug. | • `all`—All log events (that is, log events of the Standard, Trace, Interaction, and Debug levels) are generated.<br>• `debug`—The same as all.<br>• `trace`—Log events of the Trace level and higher (that is, log events of the Standard, Interaction, and Trace levels) are generated, but log events of the Debug level are not generated.<br>• `interaction`—Log events of the Interaction level and higher (that is, log events of the Standard and Interaction levels) are generated, but log events of the Trace and Debug levels are not generated.<br>• `standard` Log events of the Standard level are generated, but log events of the Interaction, Trace, and Debug levels are not generated.<br>• `none`—No output is produced. | standard | After restart |

In addition to the standard logging options that you can configure, you can configure an option named `fileEncoding` in the logging section.

`fileEncoding` specifies the encoding to be used when creating the log file. For example, `UTF-8`. This value is optional. If you do not specify this option, the server's locale information will determine the log file encoding.

This option is available for both the GRE and the Genesys Rules Authoring Tool. Also, the `log4j.properties` file that is included in both components supports a similar option, `log4j.appender.runtime.Encoding`. The `log4j.properties` file is used for initial log configuration prior to the reading of the log configuration from the Configuration Server database.

5. In the `settings` section, the following options can be configured:

## Settings in GRAT

| Description | Valid values | Default value | Takes effect |
|---|---|---|---|
| **group-by-level** (group rules by business level) | | | |
| There are three levels of rules: global, department, and process.<br><br>With value `true`, rules are grouped by business level:<br><br>• All global rules belong to agenda group `level0`.<br>• Department rules belong to agenda group `level1`.<br>• Process rules belong to agenda group `level2`.<br><br>When a rule package is executed, | true/false | true | Immediately |

| | | | |
|---|---|---|---|
| `level0` rules are executed first. Updates from this first pass then influence the department (`level1`) rules which are executed in the second pass. Updates from this second pass then influence any process rules (`level2`), which are executed in a third pass.<br><br>**Note:** The GRE option `sequential-mode` must be false when `group-by-level` is set to `true`.<br><br>When `group-by-level` is set to `false`, all rules are executed in a single pass. Changes made by a rule do not influence which other rules are executed (unless a Drools “update” or “insert” command is used). | | | |

| | | | |
|---|---|---|---|
| *CEP functionality*<br>• Genesys Web Engagement's CEP functionality strips out the rule attribute that indicates which level a rule is associated with. So, the setting of the `group-by-level` has no influence on rule execution. | | | |
| **max-connections** | | | |
| Specifies the maximum number of different users that may be connected to the server. Multiple connections from the same user ID are only counted once. | Any positive integer | 99 | After GRAT restart |
| **session-timeout** | | | |
| Specifies the amount of time (in minutes) a client session can have no communication with the Rules Authoring Server before timing out. If no value is specified, the timeout (if any) | Any positive integer | 30 | Immediately |

| | | | |
|---|---|---|---|
| defined by the application server applies. If the value is less than or equal to `0`, the session will not time out. | | | |
| **session-timeout-alert-interval** | | | |
| The amount of time (in minutes), prior to an expected timeout, for a user to be warned of a pending timeout. If no value is specified, or if the value is less than or equal to `0`, the default warning period of 1 minute will be used. For example, if you set the value of this option to 3, the user will be warned 3 minutes prior to an expected timeout. This warning dialog box will prompt the user to extend the session. If the session is not | Any positive integer | 1 | Immediately |

| | | | |
|---|---|---|---|
| extended, the user will be logged out and the login dialog box will be displayed. Any unsaved changes that the user made during their session will be lost. | | | |
| **strict-mode** | | | |
| This option controls whether or not the rules authoring tool enables *strict* mode in the DROOLS rule compiler. Strict mode will cause the compiler to catch common mistakes when the rule author attempts to validate or save a rule. | true/false | true | Immediately |
| **verify-deployer-address** | | | |
| Indicates whether to verify the TCP address of the application deploying rules to be that of an associated Genesys Rules Engine. | true/false | true | Immediately |
| **display-n-template-versions** (new in 8.1.3) | | | |
| Specifies the maximum number of versions to | Minimum value 1 | 3 | Immediately |

| | | | |
|---|---|---|---|
| display for any published template. | | | |
| **deploy-response-timeout** (new in 8.1.3 - not in application template by default) | | | |
| Specifies the timeout (in seconds) applied to the deployment of a rule package. | Any positive integer | 300 | Immediately |
| **require-checkin-comment** (new in 8.1.3) | | | |
| Specifies whether users must add a check-in comment when committing changes to rules. These comments show up when viewing package history. If the value is set to false (default), users can save changes to rules without specifying a comment. | true/false | false | Immediately |
| **force-snapshot-on-deployment** (new in 8.1.3) | | | |
| Specifies whether users can deploy only a package snapshot. If the value is `true`, users can only deploy a package snapshot. If `false` (default), users can deploy either the LATEST package or a | true/false | false | Immediately |

| | | | |
|---|---|---|---|
| snapshot. | | | |
| **encoding** (not in application template by default) | | | |
| Activates Unicode support for the conversion of data between the local character set that is used by Configuration Manager and the UTF-8 encoding that is used by the Rules Authoring Server. By default, code page conversion is disabled. To activate this functionality, set this option to the name of a converter that can translate the local character set to UTF format. The converter that is suitable for a particular deployment can be found by using the ICU Converter Explorer. There is no default value for this option. For valid values, see the ICU Home > Converter Explorer pages (http://demo.icu-project.org/icu-bin/convexp). | | | After GRAT restart |
| **clear-repository-cache** (new in 8.1.4) | | | |
| The GRAT server builds and maintains a cache of the rules repository database (for example, index files, and so on), and stores this on the file system under `WEB-INF/classes/` | true/false | false | After GRAT (re-)start |

| | | | |
|---|---|---|---|
| `repository`. The cache improves performance when accessing frequently used rules, calendars, and so on. However, this cache must stay synchronized with the rules repository database.<br><br>Normally, if GRAT is restarted, it re-uses the existing cache, which is synchronized with the rules repository database. In this case, the `clear-repository-option` should be set to `false` (default).<br><br>However, if you are configuring a second GRAT for warm standby (see High Availability Support), this option should be set to `true` for both the primary and the standby instances of GRAT. Since | | | |

| | | | |
|---|---|---|---|
| either GRAT could be brought online in the event of a failure, this option forces GRAT always to rebuild the cache and re-synchronize it with the rules repository database. Setting this option to `true` can delay the startup of GRAT, since the cache must be rebuilt, but it ensures that it is properly synchronized with the rules repository database. | | | |

6. Give the application Read, Create, and Change permissions on the `Scripts` folder for each Tenant that you add. (One approach is to create a user called `GRAT_Application_Proxy` and add that user to the `SYSTEM` access group. Then, on the `Security` tab of the application, in the `Log On As` section, select `This account` and add the `GRAT_Application_Proxy` user. Make sure that the "System" access group has Read, Create, and Change permissions to the Scripts folder, and that you have applied these changes recursively.) The `Security` tab is available only in Genesys Administrator 8.1.0 and later. Therefore, if you are not using Genesys Administrator 8.1.0 or higher, you must perform this step through Genesys Configuration Manager.
7. Give the application Read permission for all roles, access groups and persons needed for GRAT.
8. Create the GRAT client application by first importing the `Genesys_Rules_Authoring_Generic_Client_810.apd` to create the application template. From the application template, create the GRAT client application. The name of this application was specified during the installation of the IP. You just need to create the application and save it. You are not required to fill in any of the configuration properties.

# 4. Configure the client application.

## Configure the Client Application

To configure the client application:

1. Right-click the `Applications` folder.
2. Select `New > Application`.
3. Select the `Genesys_Rules_Authoring_Generic_Client` template.
4. On the `General` tab, enter a name for the application, such as `Rules_Authoring_Client`.
5. Click Save.

### Next Steps

- Installing the GRAT Component