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.

Predictive Routing Deployment and Operations Guide

Deploy the URS Strategy Subroutines

12/14/2025

Contents

Deploy the URS Strategy Subroutines

Genesys Predictive Routing (GPR) provides subroutines components that are integrated with your Genesys Routing solution. The subroutines are placed within an existing strategy, where they add agent scoring and best-match functionality that enables you to fine-tune the routing of a specific interaction to the agent who can best handle it, based on the KPIs you want to optimize.

This topic explains how to use the pre-configured strategy subroutines with Interaction Routing Designer (IRD) and Universal Routing Server (URS). This topic includes the following information:

- List of subroutines and other files included in the IP, with brief explanations of what they do.
- Deployment procedures.
- See Routing Scenarios Using GPR for a discussion of how the subroutines handle agent-interaction matching in various scenarios.

For information on how to deploy the Composer Subroutines, see Composer, Orchestration Server (ORS), and URS.

Out-of-the-Box IRD/URS Strategy Subroutines

The IP for the URS Strategy Subroutines component contains the following strategy subroutine RBN files as well as the **object.kvlt** file and the **Predictive_Route_Data_Template.cfg** file. (The RBN files are listed in alphabetical, not logical, order.)

- **ActivatePredictiveRouting_v3**: Called from a routing strategy when the conditions you specify are met. This subroutine automatically calls additional subroutines that score agents, rank the scored agents, and evaluate whether there is an agent available to handle the interaction based on agent score. Referred to simply as *ActivatePredictiveRouting* in the remainder of this topic.
- **GetActionFilters**: Called from the ActivatePredictiveRouting subroutine to parse the Virtual Agent Group string representation. In release 9.0.015.00, the GetActionFilters subroutine was enhanced to identify the list of agents matching the target skill group along with the configured login status expression. This information is also reported in the action filters of the scoring request. This functionality is invoked only when the **use-action-filters** configuration option is set to `false`.
- **GetScoringAuthToken**: Called from the ActivatePredictiveRouting subroutine to authenticate with the AI Core Services REST API.
- **GPRIxnCleanup**: Starting in URS Strategy Subroutines 9.0.015.00, performs score log and UserEvent (KVP) distribution (previously done in the GPRIxnCompleted subroutine). This attached UserEvent data includes details required for Predictive Routing performance analysis, such as gpmStatus, which indicates whether, when the interaction was routed, there was an agent-surplus or an interaction-surplus situation. Also cancels the execution of the ScoreIdealAgent and isAgentScoreGood callback subroutines if an attempt to route an interaction with Predictive Routing times out. The strategy then tries to route the interaction using skill-based routing.
  - Starting in URS Strategy Subroutines 9.0.015.00, this subroutine requires you to configure certain parameters. See Configure GPRIxnCleanup for instructions.

  - This subroutine must be called from the default port of the last Routing block used for Predictive Routing. You must also set the **Clear targets** flag in that Routing block. Alternatively, you can configure the strategy to loop back to the same Routing block again after calling the GPRIxnCleanup subroutine.

- **GPRIxnCompleted**: Computes the values for the GPR KVPs based on the selected agent scores and updates the URS global map. In releases prior to URS Strategy Subroutines 9.0.015.00, also attaches Predictive Routing–specific keys to interaction user data (done in the GPRIxnCleanup subroutine in URS Strategy Subroutines 9.0.015.00 and later). This attached data includes details required for Predictive Routing performance analysis, such as gpmStatus, which indicates whether, when the interaction was routed, there was an agent-surplus or an interaction-surplus situation. Links to the interaction ID in Genesys Info Mart. (Named PRRIxnCompleted in earlier releases.)
- **GPRIxnSetup**: (Introduced in URS Strategy Subroutines 9.0.015.00) Creates the interaction global map in URS memory with the ConnectionID as the key name; adds the Genesys Info Mart KVPs and initializes them with the default values; sets the gpmMode parameter to `off` and gpmResult to 15 (Predictive Routing is turned off or not used for this interaction). See Configure GPRIxnSetup for complete configuration instructions and how to use this subroutine.
- **isAgentScoreGood**: URS calls this subroutine to suppress routing to an agent who is in ready state if this agent does not provide an acceptable match for the interaction. You can use this in conjunction with relaxation thresholds to target better-matched agents preferentially, expanding the pool of agents if the best-matched agents are unavailable.
- **ScoreIdealAgent**: Called by URS before routing an interaction to an Agent Group or Virtual Agent Group at the time URS invokes the SelectDN function. This subroutine sorts the agents within the target group according to their scores, in decreasing order. It can also rescale the agent score to the range accepted by URS, if necessary.
- **ScoreIdealAgentDefault**: Used either when the agent scores for the current interaction are unknown or when running GPR in dry-run mode.
- **SetIdealAndReadyCondition**: Called from ActivatePredictiveRouting subroutine. This is a wrapper subroutine around scheduling the calls to subroutines such as ScoreIdealAgent and isAgentScoreGood, which are executed outside the main interaction processing flow. Implements the GPR operation modes controlled by the **prr-mode** option.
- **SetIdealAndReadyConditionforOCS**: Required if you are using the Composer Workflow Subroutines.

Other Files

- **objects.kvlt**: A text file containing objects required by the strategy subroutines, including the Print_Log_Message macro. You must import this file as well as the RBN file for the strategy subroutines to work correctly.
- **Predictive_Route_Data_Template.cfg**: Template for the Predictive_Route_DataCfg Transaction List object. Stores GPR-related configuration options and values.

Subroutines for Environments Using Dynamic Priority Routing

Release 9.0.014.04 and higher of the URS Strategy Subroutines provide the following subroutines that provide improved dynamic-priority interaction handling:

- **ActivatePredictiveRouting_v3** - This subroutine does the following:
  1. The main routing strategy should set the initial priority for the interaction. The Initialize Variables block takes the initial priority parameter from the skill data `List` object and saves it to the

parBasePriority local variable.

2. After a scoring request is completed and the agent scores are placed into the URS Global Map linked to the interaction, the subroutine executes priority increments. This happens once per interaction.
3. The subroutine verifies whether the **set-dynamic-priority** option is set to `true`. The following steps are executed only on the first routing attempt when the option is set to `true`.
4. The subroutine reads the remaining priority options configured on the Predictive_Route_DataCfg `List` object (**priority-increment**, **priority-init-interval**, and **priority-interval**).
5. The subroutine calls the IncrementPriorityEx function with the required arguments.

Subroutines for Environments Using Non-ASCII Encoding

Release 9.0.014.04 and higher of the URS Strategy Subroutines can support non-ASCII encoding (by default, all GPR components use UTF-8 encoding). The subroutines specified below include enhancements for non-ASCII environments:

- **ActivatePredictiveRouting_v3** - Converts non-ASCII characters to UTF-8 characters before sending scoring requests to the Predictive Routing scoring engine.
- **GPRIxnCompleted** - Converts non-ASCII characters to UTF-8 characters before sending scoring logs.
- **GetScoringAuthToken** - Replaces the StrFormat function with the SetStringKey function. This eliminates an issue with the *~s* character in the StrFormat function, which does not work correctly in a non-ASCII environment.

> **Important**
>
> URS 8.1.400.55 is the minimum required version for GPR-specific IRD subroutines to work in non-ASCII environments.

Installing the URS Strategy Subroutines

The following is a high-level overview of the steps required to deploy the URS Strategy Subroutines:

1. Configure URS to Support Predictive Routing.
2. Import the subroutines. For a list of the subroutines, with descriptions of their functionality, see IRD/URS Strategy Subroutines.
3. Define the entry points in your IRD strategy for the appropriate subroutines.
4. Set appropriate values for the strategy subroutine configuration options, which are located in the Predictive_Route_CfgData Transaction List object.
5. Configure the parameters for the subroutines used in your environment.
6. Test that the subroutines are correctly directing interactions to agents.

The following sections provide detailed instructions for setting up your subroutines.

Configuring URS to Support Genesys Predictive Routing

Perform the following steps to configure URS to work with GPR:

1. Create the **static_strategy** configuration option in the **[default]** section of the URS `Application` object and set its value to `empty`. You can set this option either on the level of individual routing points or on the tenant/URS level. Depending on where you set it, the option works slightly differently. This option takes effect immediately and does not require that you restart URS.
2. Configure the http log for URS to check scoring requests and responses. To do this, set the **verbose** option in the **[web]** section to 3.
3. Set the **run_verbose** option in the **[default]** section to `0`.
4. Set the **vqtime** option in the **[default]** section to `13:2048`.
5. Enable HTTPS support:
   No changes to the Subroutines components are required. However, HTTPS support on Unix-based operating systems requires that you install the Genesys Security Pack on the host running Universal Routing Server (URS). See Installing Genesys Security Pack in the *Genesys Security Deployment Guide* for more information.

   - For instructions, and a general discussion of HTTPS and TLS support among Genesys components, see the *Genesys Security Deployment Guide*.
   - For HTTP/S configuration for URS, see the section "Web Service Connections Using HTTP Bridge" in the *Genesys 8.1 Universal Routing Deployment Guide*, the sections on Web Service Option (p. 687), IRD Web Service Object (p.771), and TLS (p. 782) in the Universal Routing Reference Manual, and HTTP Bridge Updates in the Supplement to the *Universal Routing Reference Manual*.
     - Only simple TLS is supported. For simple TLS, you need to configure only the URS **def_trusted_ca** option.

     If you are using a chain of trusted certificates (intermediate ca and root ca, in pem file format), you can concatenate them, as described in Configuring Multiple Trusted CAs in the *Genesys Security Deployment Guide*.

Strategy Subroutine Integration for URS/IRD

To use the strategy subroutines provided with Predictive Routing, perform the following steps:

1. Navigate to the Export/Import Bar in IRD and import the strategy subroutines from the URS Strategy Subroutines IP.
   For instructions, see the *Export/Import Bar* topic in the *Interaction Routing Designer Help*, which you can open from the Universal Routing landing page.
2. Open the strategy you plan to use with Predictive Routing and place the ActivatePredictiveRouting Call Subroutine object in the desired location. It must be inserted into the routing strategy before the call to a Routing block or a call to the SelectDN function. The graphic below shows the subroutine insertion points.

This subroutine requires the following three input parameters:

- `skill_target:` A STRING indicating the target selected by the URS for an interaction. This must be a virtual agent group.
- `skill_data:` A LIST, which must contain the following two keys:
  - `overflow_timeout:` A STRING defining a period of time in seconds during which URS tries to route the interaction to the current target.
  - `base_priority:` An INTEGER defining the priority value the interaction has before the call to the ActivatePredictiveRouting subroutine.
- `default_skill_data:` A LIST, which must contain the following keys:
  - `AgentScore:` Takes either `Y` or `N` as its value. To activate predictive routing, you must set this parameter to `Y`.
  - `predictor:` Contains the name of the section in the Predictive_Route_DataCfg `Transactions List` configuration object that defines predictor configuration.
  - `START_TS:` (Used in URS Strategy Subroutines 9.0.015.00 and later. See gpmWaitTime Configuration for details.) The UTC time indicating when an interaction arrived in the queue.
  - `prr-ixn-timestamp:` (Used in URS Strategy Subroutines 9.0.014.04 and earlier. See gpmWaitTime Configuration for details.) Contains a value defining the number of seconds since midnight when the interaction was queued.

3. The isAgentScoreGood subroutine checks for interactions that have been in queue for an unusually long time using the `varQueueTimeSec` parameter, which is set to `600` seconds by default.

> **Important**
>
> - If you are expecting conditions that might result in longer wait times, you might need to set a larger value for this variable. To change the value of this variable, contact Genesys Customer Care for assistance.
> - URS might experience high CPU loads if the timeout is extended beyond 600 seconds and you are using agent hold-out (that is, you have set the value for the **use-setreadycondition** option to `true`).

1. The GPRIxnCompleted subroutine (called PrrIxnCompleted in earlier releases) can be inserted either as a Custom Routing step in the Routing object or immediately in front of the object in your strategy that contains a call to the RouteCall function.
2. In URS Strategy Subroutines releases prior to 9.0.007.00, insert the PrrIxnCleanup subroutine after the green port exit from either the Routing object or the object that contains a call to the RouteCall function.
   NOTE: PrrIxnCleanup is not used in URS Strategy Subroutines release 9.0.007.00 and higher. This update is supported only in environments running URS version 8.1.400.37 and higher.
3. Set values for the configuration options in the Predictive_Route_DataCfg Transaction List Object. Among others, the GPR subroutines send the scoring requests to the URL configured in the **jop-scoring-url** option and score_log requests to the URL configured in the **jop-logging-url** option. The **orig-connid-key** options is required in all deployments to store the original

connection ID, used as a unique identifier, for each interaction.

Configure GPRIxnSetup

The GPRIxnSetup subroutine, introduced in URS Strategy Subroutines 9.0.015.00, does the following:

- Creates the interaction global map in URS memory with the ConnectionID (ConnID) as the key name.
- Adds all the Genesys Info Mart KVPs used in GPR and initializes them with the default values. All the KVP values initialized by the GPRIxnSetup are replaced with actual values when the remaining GPR subroutines are invoked. If the GPR subroutines are not invoked, the score log and the Genesys Info Mart database continue to store the default values set in the GPRIxnSetup subroutine.
- Sets the gpmMode parameter to `off` and gpmResult to 15 (Predictive Routing is turned off or not used for the current interaction). This establishes the initial condition in the contact center and establishes clearly when GPR actively begins to score agents and determine routing targets. As soon as interactions are routed using the GPR subroutines, these KVP values are updated to reflect actual contact center conditions.

To use GPR, set the value of the URS **prestrategy** option to `GPRIxnSetup`.

- If you already specify a subroutine in the **prestrategy** option, you can copy the contents of the GPRIxnSetup subroutine into your existing subroutine.
- For a complete description of the **prestrategy** option, see the Universal Routing Reference Manual.

Configure GPRIxnCleanup

The GPRIxnCleanup subroutine must be called from the default port of the last Routing block used for Predictive Routing. Starting in URS Strategy Subroutines 9.0.015.00, this subroutine can now correctly identify abandoned interactions and interactions in which GPR was unable to route the interaction and add this information to the score log and the Genesys Info Mart gpmResult KVP. The KVP values used are the following:

- gpmResult = 13 - Interaction abandoned
- gpmResult = 14 - GPR routing was unsuccessful

To enable this functionality, configure the GPRIxnCleanup subroutine as follows:

1. When routing succeeds, call GPRIxnCleanup with the parameter `true` from the **default** port of the last Routing block used for Predictive Routing.
2. When routing fails, call GPRIxnCleanup with the parameter `false` from the **red** port of the last Routing block used for Predictive Routing.
3. When an interaction is abandoned, GPRIxnCleanup is called automatically with an empty parameter.

Configure Reporting on Both GPR and Non-GPR Interactions

There are a number of scenarios in which interactions routed using GPR and those routed using alternative methods (non-GPR) might target the same pool of agents:

- Non-GPR interactions might be sent to the same queue as GPR interactions.

- A strategy might send some interactions to GPR and to non-GPR routing, based on specified conditions.
- A different strategy might be routing interactions to the same target agents as the GPR interactions.

URS Strategy Subroutines 9.0.015.00 and later provides support for tracking interactions not routed using GPR. The following KVP values provide the necessary information:

- gpmMode = `off`
- gpmResult = 13 (abandoned) or 14 (routing attempt failed, non-GPR routing used)

> **Important**
>
> If routing fails for a non-GPR interaction, or it is abandoned, GPR does not record these status conditions. For GPR the following is recorded: gpmResult = 15, gpmMode = `off`, gpmMessage = `Predictive Routing is turned off or not used for this interaction`.

To configure your routing environment to handle both GPR and non-GPR routed interactions, perform the following steps:

1. Configure the gprIxnSetup subroutine, which is require to report correctly on blended GPR and non-GPR routing.
2. Call the following subroutines in all strategies routing interactions to agents also targeted from GPR. You can invoke them directly, without going through the ActivatePredictiveRouting_v3 subroutine.
   - GPRIxnCompleted should be called as a custom routing step in all Target Selection blocks or as a Function block before routing interactions to agents using a RouteCall block.
   - GPRIxnCleanup should be called after all Target Selection blocks or RouteCall blocks.

gpmWaitTime Configuration

In release 9.0.015.00 and later, the URS Strategy Subroutines use the value of the START_TS variable to calculate when the interaction arrived in the queue. START_TS is a mandatory variable passed in the default_skill_data List object and stored as a KVP for use in Genesys Info Mart reporting.

- gpmWaitTime is calculated based on the difference between: (the UTC time when the agent is selected) - (START_TS variable value).
- The START_TS variable is also used for measuring periods of time in A/B tests.
- In Genesys Info Mart reporting, the value of the START_TS variable is converted to the DateTimeKey function, used to populate the START_DATE_TIME_KEY column in the GPM_FACT table.

In release 9.0.014.04 and earlier, the subroutines use the value of the prr-ixn-timestamp variable, in the default_skill_data List object, which is passed to the ActivatePredictiveRouting_v3 subroutine. prr-ixn-timestamp defines the number of seconds since midnight when the interaction was queued.

This variable is then used for the calculation of the gpmWaitTime which denotes the actual wait time of the interaction in the queue before an agent is selected. This calculation uses the TimeStamp[] and TimeDifference[] functions, available in IRD/URS.

Turn Off Predictive Routing

You might choose to turn predictive routing off temporarily for A/B testing or troubleshooting. To do so, use one of the following methods:

- Set the **prr-mode** configuration option to `off`.
- Set AgentScore = `N` in the default_skill_data object parameter passed to the ActivatePredictiveRouting_v3 subroutine.