Working with the iWD Business Process in IRD

5/6/2025

Changes to IWDBP Strategies & Subroutines in 8.5.105

These topics describe changes made to the IWD Business Process for IRD in IWD release 8.5.105. Where strategies and subroutines are not referenced below, they remain the same as for release 8.5.104.

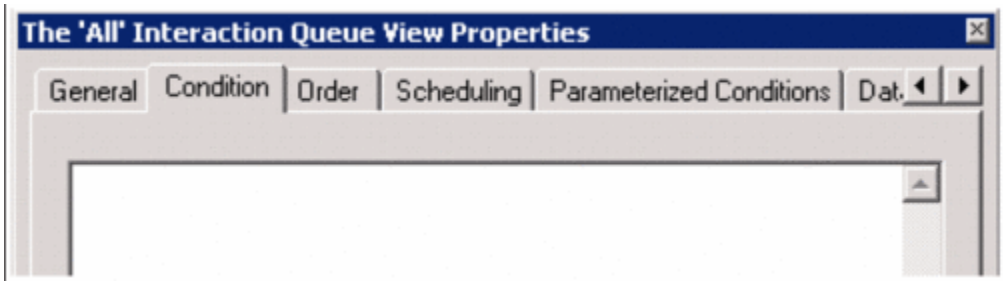
Classification Strategy

Classification Strategy

The purpose of this strategy is to invoke corresponding classification rules, analyze the result of the rules application and place the interaction into the appropriate queue, depending on the result.

This strategy processes interactions from the following queues:

- `iWD_New`—Interactions have to satisfy the following conditions:
  - There are no conditions here.
  - Interactions are taken in order they were submitted.

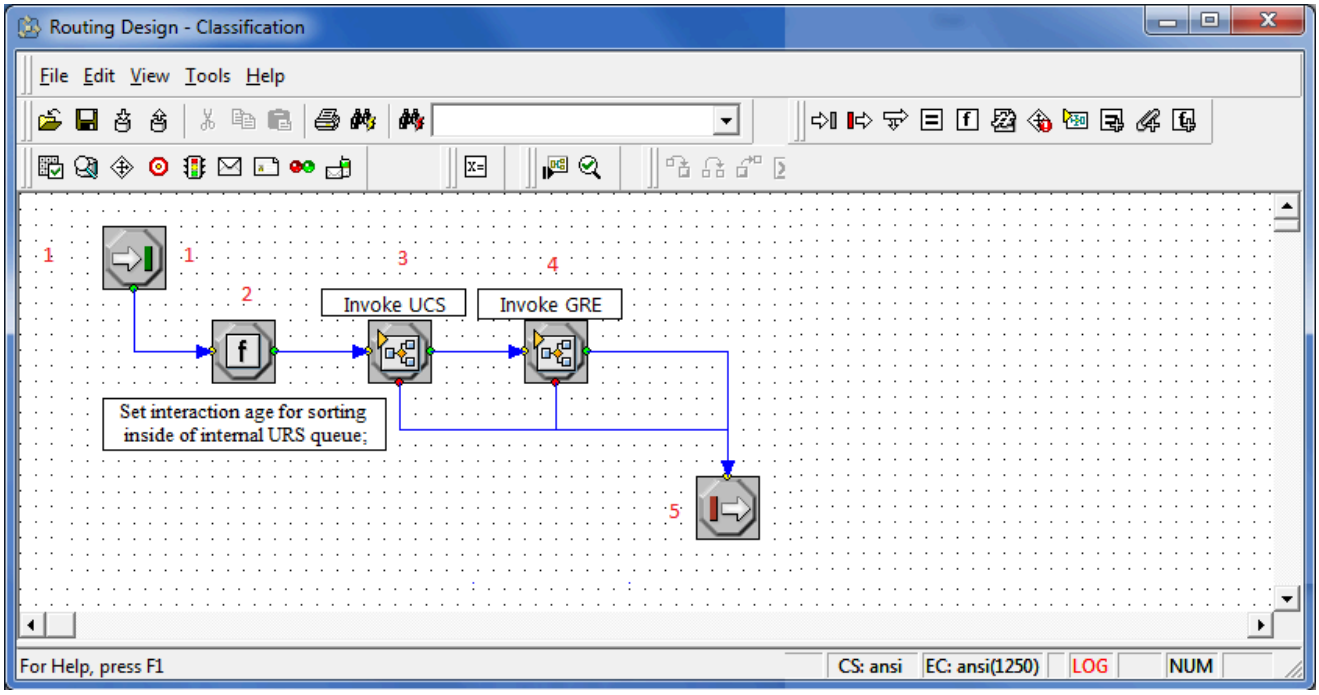


Changes in 8.5.105

Code has been refactored and removed to the InvokeGRE and InvokeUCS strategies in order to simplify them.

Flow Summary

Click to enlarge.



Flow Detail

1. Entry to the Classification strategy.
2. A command is sent to URS to use the interaction age while sorting interactions in internal queues.
3. The InvokeUCS subroutine is invoked to create a new interaction in the UCS database.
4. The InvokeGRE subroutine is invoked.
5. Exit Classification strategy.

Prioritization Strategy

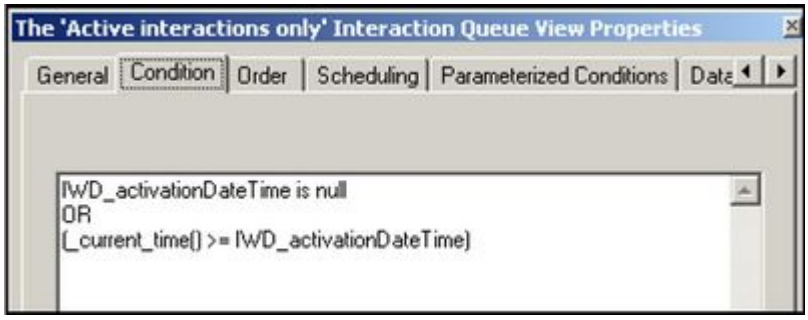
Prioritization Strategy

The purpose of this strategy is to invoke the corresponding prioritization rules, analyze the result of the rules application and place the interaction into the appropriate queue, depending on the result.

This strategy processes interactions from the following queues:

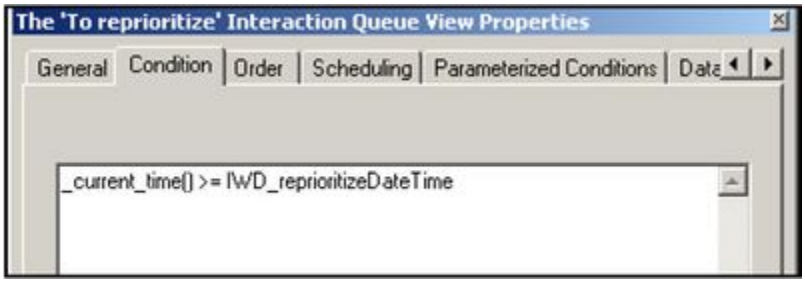
- `iWD_Captured`—Interactions have to satisfy the following conditions:

- Active interactions only (interactions which do not have the property `IWD_activationDateTime` set, or this property has a time stamp which is in the past.
- Interactions are taken in the order they were submitted.



Active Interactions only

- `iWD_Queued`—Interactions have to satisfy the following conditions:
  - Interactions that are subject for immediate reprioritization (interactions that have the property `IWD_reprioritizeDateTime` set to a time stamp which is in the past)
  - Interactions are taken in order of `IWD_reprioritizationDateTime` (oldest first).

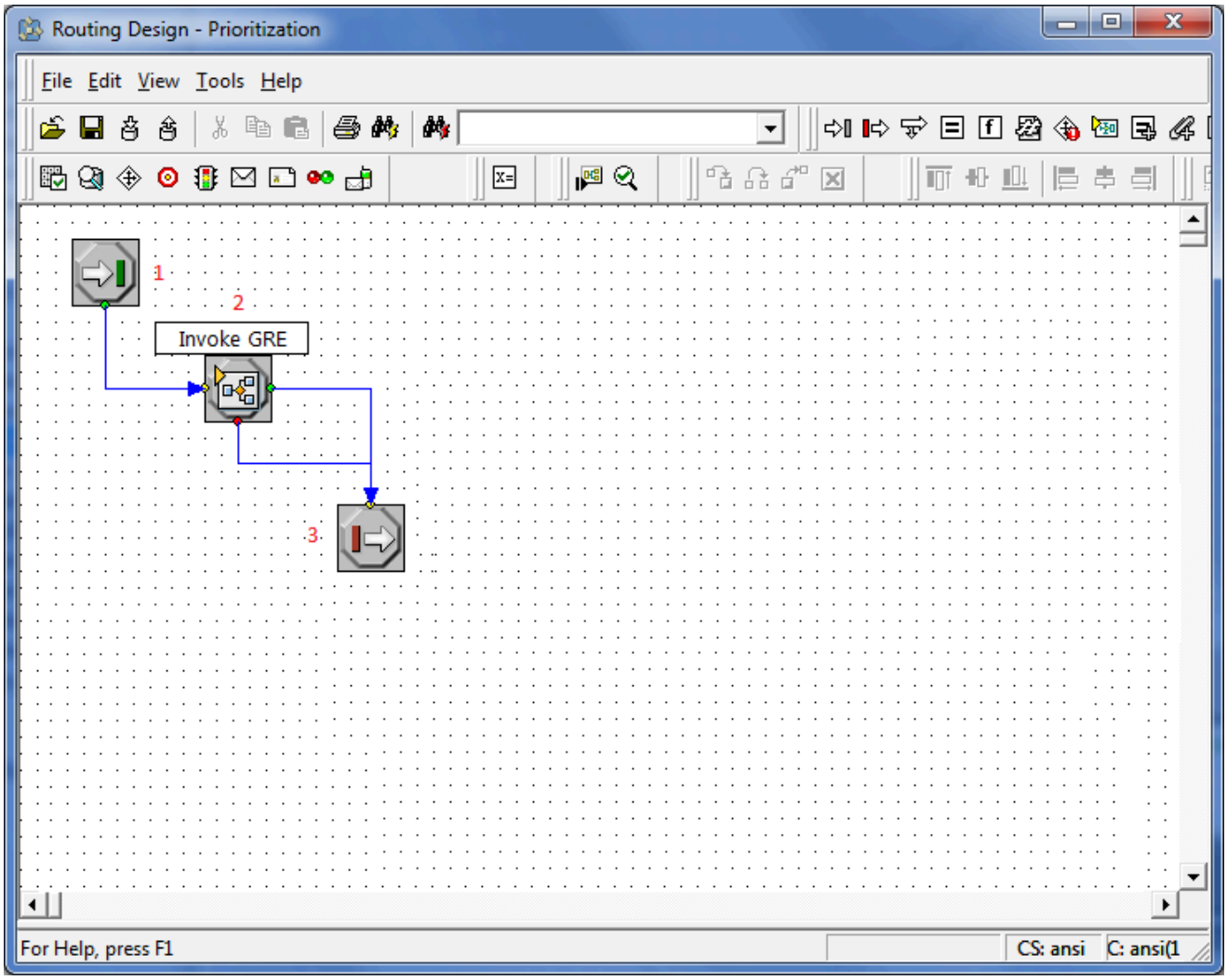


For reprioritization

Changes in 8.5.105

Code has been refactored and removed to the InvokeGRE and InvokeUCS strategies in order to simplify them.

Flow Summary



Flow Detail

1. Entry to the Prioritization strategy.
2. The InvokeGRE subroutine is invoked.
3. Exit Prioritization strategy.

Distribution Strategy

Distribution Strategy

This strategy routes interactions to a requested Agent, requested Agent Group, requested Skill, or to the default iWD Agent Group, and can now process an `IWD_Segment` attribute from the segmentation settings.

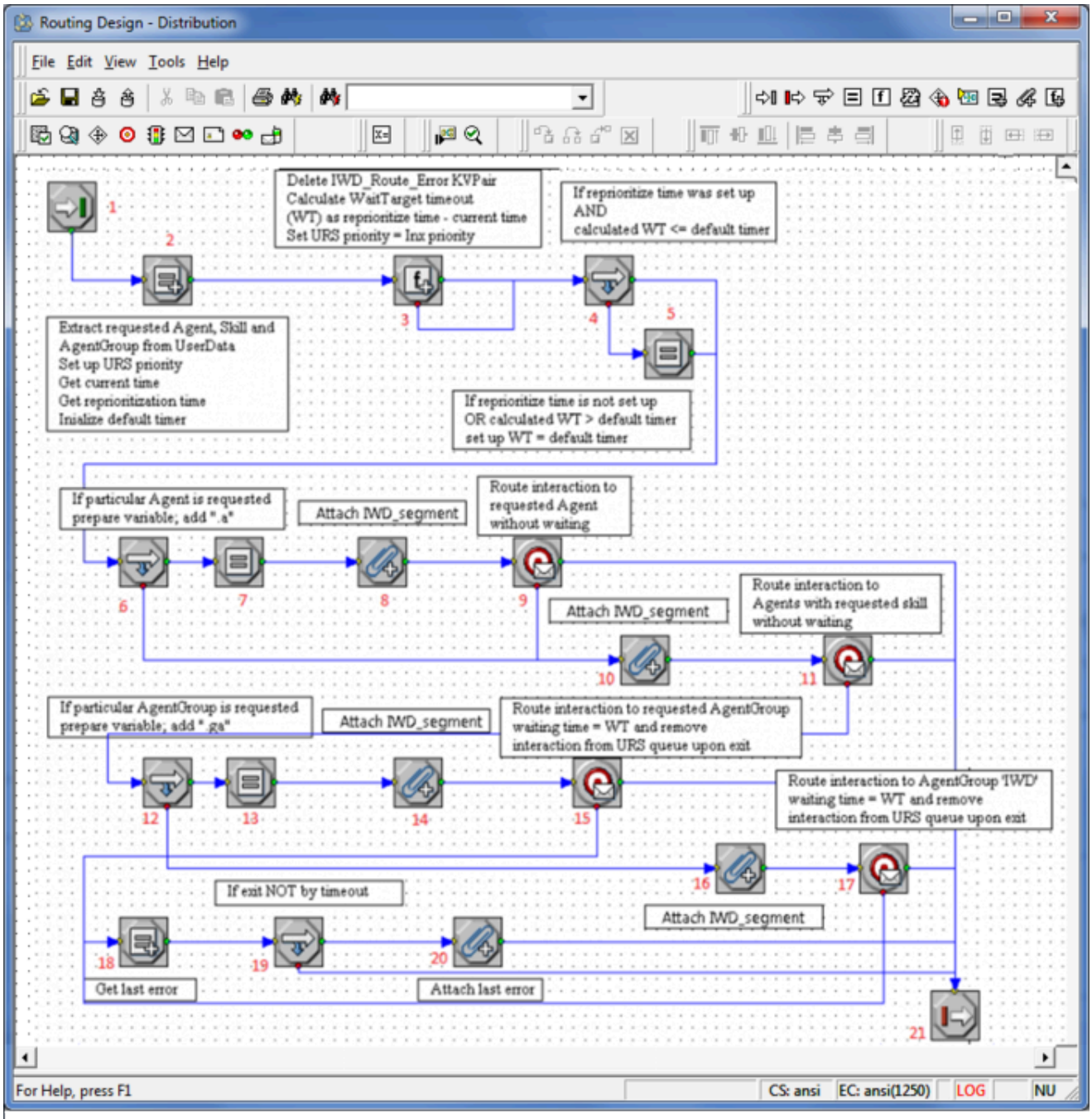
This strategy processes interactions from the following queues:

- `iWD_Queued`—Interactions have to satisfy the following conditions:
  - Interactions that are not subject for immediate reprioritization (interactions that do not have the property `IWD_reprioritizeDateTime` set, or that have this property set to a time stamp that is in the future).
  - Interactions are taken in order of priority (highest priority first)

Changes in 8.5.105

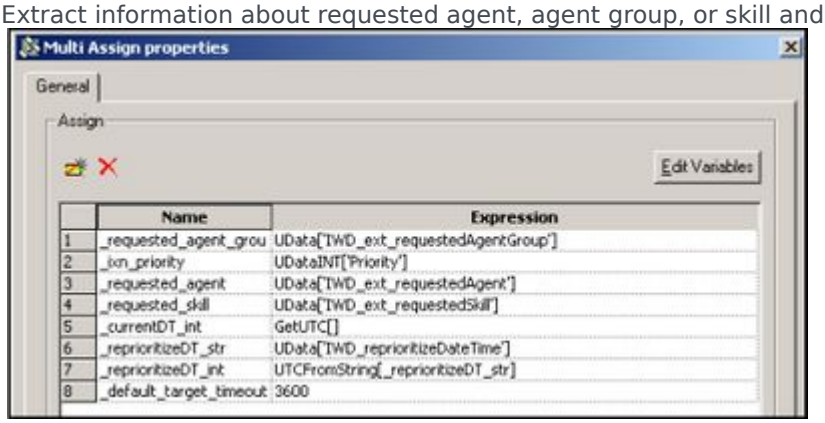
Segmentation settings have been added to the **ToDistribute** View in the Distribution strategy, enabling it to make a call to the configured segments then add an **IWD_Segment** attribute to the interaction data. See **the 8.5.105 diagram here** and the **feature description here**.

Flow Summary



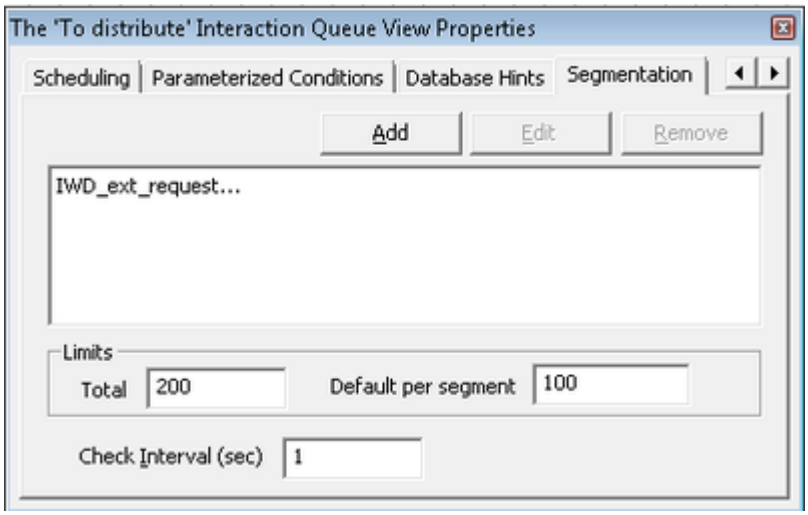
Flow Detail

1. Entry to the Distribution strategy.
2. Extract information about requested agent, agent group, or skill and initialize internal variables.



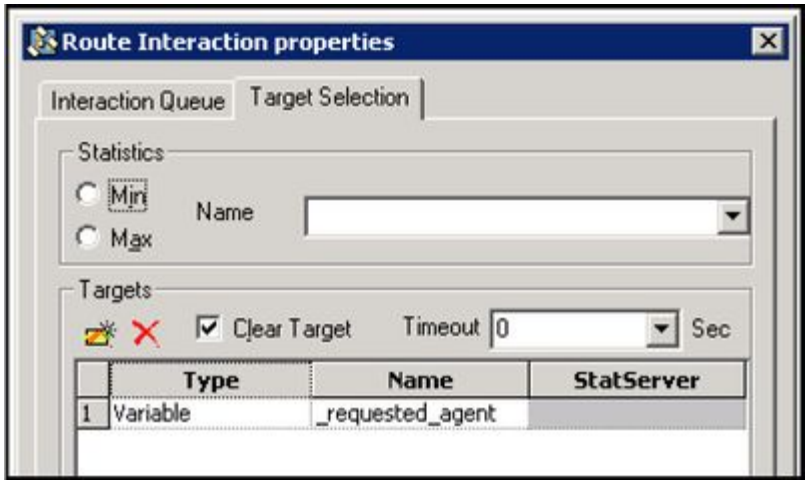
Multi-Assign - Requested Agent and Skill

3. A calculation is done to determine the timeout—how long the interaction should wait for a target to become available.
4. If the reprioritize time was set up and the calculated timeout is less than or equal to the default timeout (`1 hour`, see Step 1), then the timeout remains as it is.
5. If the reprioritize time was not set, or the calculated timeout is greater than the default timeout, then the waiting timeout is set to the default (`1 hour`).
6. Analysis is done to determine whether an agent was requested.
7. If an agent was requested, the URS variable is prepared (`.a` is added).
8. Update the `IWD_segment` attribute to '`_requested_agent`'.



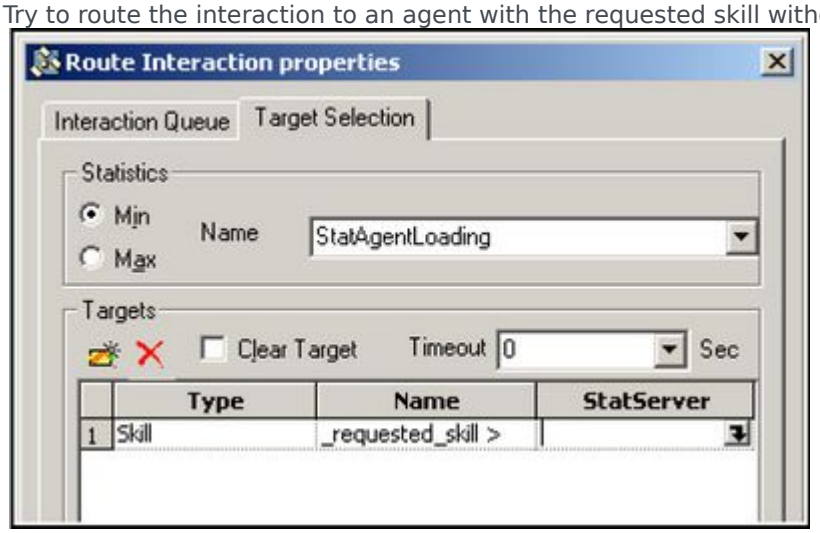
Segmentation view

9. Try to route the interaction to the requested agent without waiting.



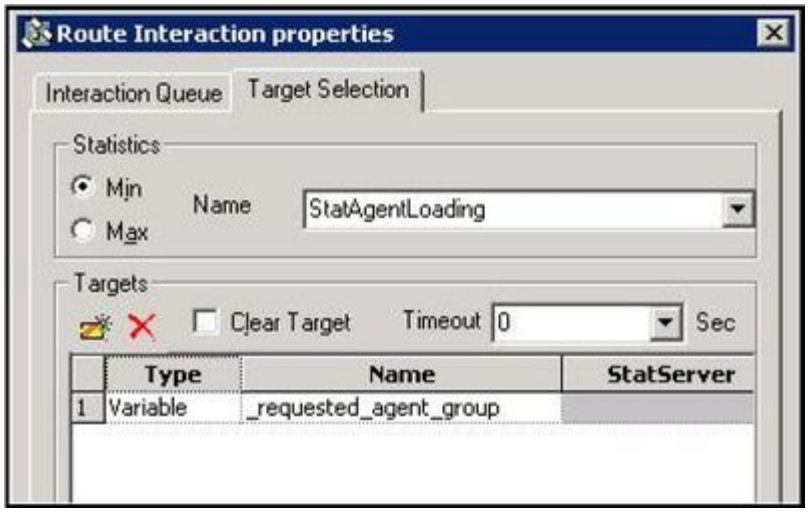
Route to agent

10. Update the `IWD_segment` attribute to '`_requested_skill`'.
11. Try to route the interaction to an agent with the requested skill without waiting.



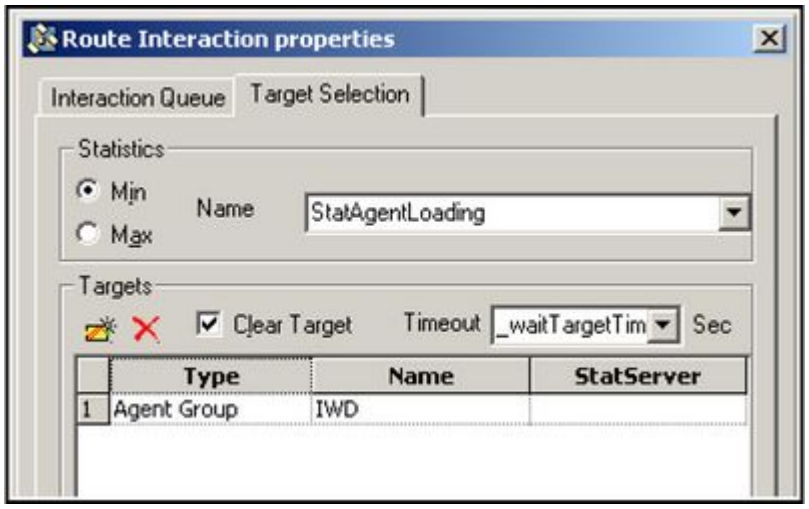
Route to Skill

12. Analysis is done to determine whether an Agent Group was requested.
13. If an Agent Group was requested, the URS variable is prepared (`.ga` is added).
14. Update the `IWD_segment` attribute to '`_requested_agent_group`'.
15. Try to route the interaction to the requested Agent Group with wait time set to `_waitTargetTimeout`.



Route to Requested Agent Group

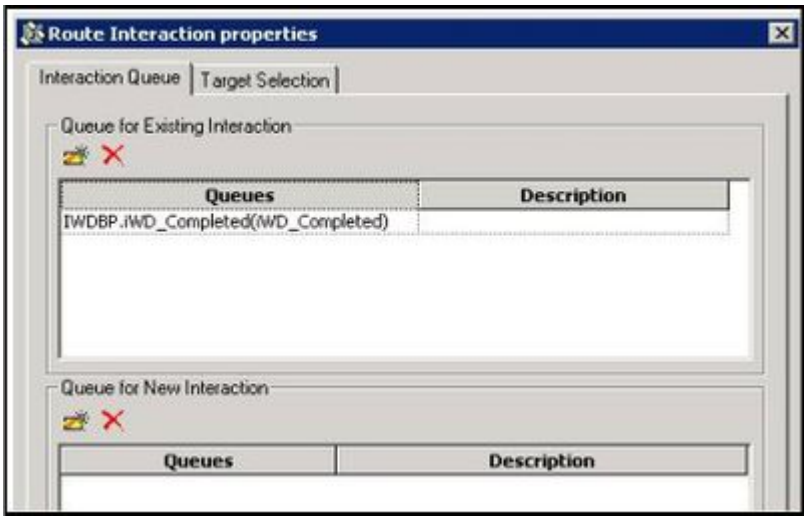
16. Update `IWD_segment` attribute to '`_default`'.
17. Try to route the interaction to the iWD agent group with a a wait time to `_waitTargetTimeout`.



Route to Agent Group

18. Get the last error.
19. Verification is done as to why the target was not found.
20. An error code is attached in case of any error other than a timeout. If more than one target is available, URS uses the `StatAgentsLoading` statistic to select the Agent who has the minimum load (this applies to routing to Skills and routing to Groups only; routing to a requested Agent does not use statistics). For more information about this statistic, see the Universal Routing 8.1 Reference Manual. The Route Interaction object also has an `Interaction Queue` tab. (This applies to all three Route Interaction

objects in this strategy.)



Route Interaction Properties—Interaction Queue

The optional `Interaction Queue` tab enables you to specify two types of queues:

- Queues for existing interactions (the queue in which the interaction should be placed after the agent is done working with it).
- Queues for new interactions (the queue in which new interactions created by the agent should be placed).

A Description (optional) appears as a hint on the agent desktop as to where to place the interaction.

- Exit from the Distribution strategy.

Invoke GRE Strategy

Invoke GRE Strategy

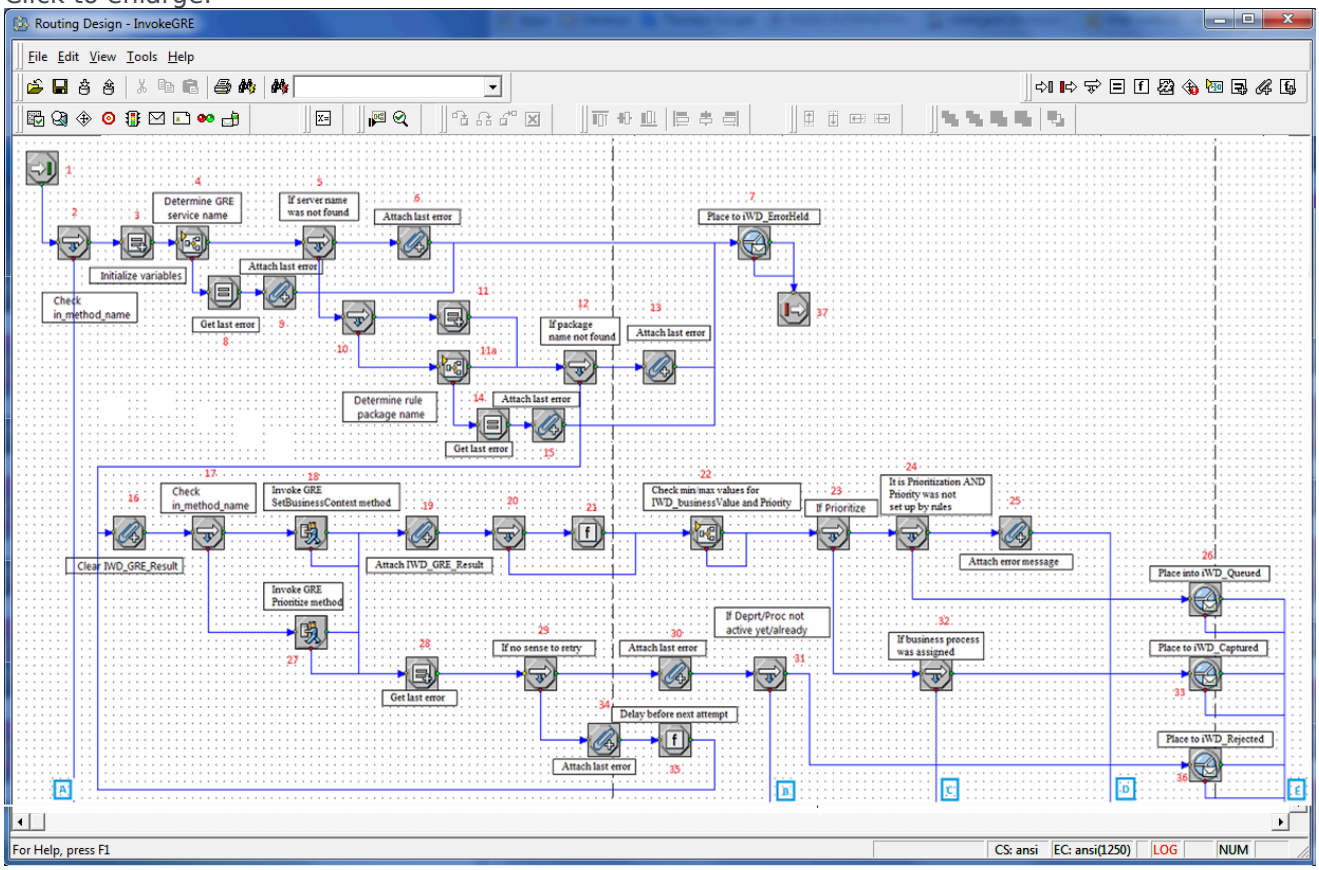
Changes in 8.5.105

Code has been removed from Classification and Prioritization strategies to the InvokeGRE strategy to simply the overall business process.

Flow Summary

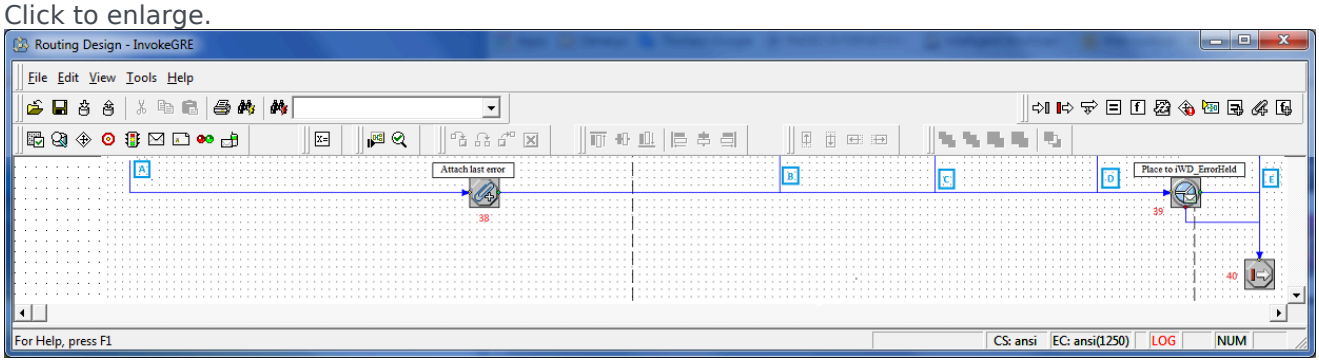
Part 1

Click to enlarge.



Part 2

Click to enlarge.



Flow Detail

1. Entry to InvokeGRE strategy.
2. Check if in_method_name is set to `SetBusinessContext` or `Prioritize`.
3. A variable is initialized—`_delay_ms` specifies the delay (in milliseconds) between attempts to invoke rules.
4. The DetermineESPServerName subroutine is invoked to determine the name of the Genesys Rules Engine Application. The subroutine uses the List Object list `GREServerList`.
5. If the subroutine was successful, a check is done to ensure the existence of the ESP server name that was returned by the subroutine. If the ESP server name was found, the flow goes to 10. The DetermineRulePackageName subroutine is invoked to determine the name of the rule package that the Genesys Rules Engine will be invoking to evaluate the classification rules.
6. If the ESP server name was not found, this error is attached to user data as a key-value pair with the key `IWD_GRE_Determination_Error`.
7. The interaction is placed in the `iWD_ErrorHeld` queue.
8. If the subroutine fails an error is extracted.
9. This error is attached to user data as a key-value pair with the key `IWD_GRE_Determination_Error`.
10. Check if `in_custom_package_name` was published to this subroutine. If it is set then `in_custom_package_name` will be run. Otherwise package name needs to be found in `Iwd_Package_List`.
11. Assign `in_custom_package_name` to `_gre_package_name` and set `_return_code` to `0`.

    11a. The DetermineRulePackageName subroutine is invoked to determine the name of the rule package that the Genesys Rules Engine will be invoking to evaluate the classification rules. If the rule package name was found, the flow goes to Step 16.

12. If the rule package name was found, the flow goes to Step 16.
13. If the rule package name was not found, this error is attached to user data as a key-value pair with the key `IWD_Rule_Package_Determination_Error`.
14. If the subroutine fails an error is extracted.
15. This error is attached to user data as a key-value pair with the key `IWD_Rule_Package_Determination_Error`.
16. Delete `IWD_reprioritizeDateTime` from attached data.
17. Check if `in_method_name` = `SetBusinessContext`.
    - If `in_method_name` is set to `SetBusinessContext` then the process calls classification rules in GRE.
    - If `in_method_name` is set to Prioritize then the process calls prioritization rules in GRE.
18. An ESP request is sent to the Genesys Rules Engine to evaluate the classification rules.
19. The ESP result is attached to user data as a key-value pair with the key `IWD_GRE_Result`.
20. Check if `IWD_reprioritizeDateTime` was changed by rules.

21. Delete `IWD_reprioritizeDateTime` from interactin if it was not changed by rules.
22. Invoke CheckBusinessValueAndPriority subroutine to verify if `IWD_businessValue` and `Priority` have correct values.
23. Check if `in_method_name` = `Prioritize`.
24. Check if `Priority` was changed by rules in `Prioritization`.
25. Attach `IWD_Prioritization_Error` to interaction with message `Priority is not set up by rules`.
26. The interaction is placed in the `iWD_Queued` queue.
27. An ESP request is sent to the Genesys Rules Engine to evaluate the prioritization rules.
28. The last Interaction Server-related error is extracted from a variable.
29. A check is done to see if the error code is related to the ESP server communication.
30. The last error is attached to user data as a key-value pair with the key `IWD_GRE_Error`.
31. Check if Department/Process is active.
32. Check if business process was assigned.
33. The interaction is placed in the `iWD_Captured` queue.
34. The last error is attached to user data as a key-value pair with the key `IWD_GRE_Error`. If not, the value of the `_counter` variable is incremented by 1.
35. A delay is introduced, based on the value of the `_delay_ms` variable. The flow goes back to 18 to retry the connection to the ESP server. The result from the ESP call to the Genesys Rules Engine is attached to the interaction as user data, with the key `IWD_GRE_Result`. This key-value pair will have the following format:

    ```
    return:ok| NumberOfRulesApplied:<number of applied rules>|
    RulesApplied:<rule 1 id> <rule1 name>, <rule2 id> <rule2
    name>, ...
    ```

    The following example shows what the result might look like:

    ```
    AttributeUserData [list, size (unpacked)=168] = 'ESP_Result'
    [str] =
    "return:ok|NumberOfRulesApplied:12|
    RulesApplied:McrSlt1GlbClsf1
    McrSlt1GlbClassification1, McrSlt1GlbClsf2
    McrSlt1GlbClassification2"
    ```

36. The interaction is placed in the `iWD_Captured` queue.
37. Exit InvokeGRE subroutine.
38. The last error is attached to user data as a key-value pair with the key `IWD_GRE_Error` when `in_method_name` is not set to `SetBusinessContext` or `Prioritize`.
39. The interaction is placed in the `iWD_ErrorHeld` queue.
40. Exit InvokeGRE subroutine.

FindListObjectItem Subroutine

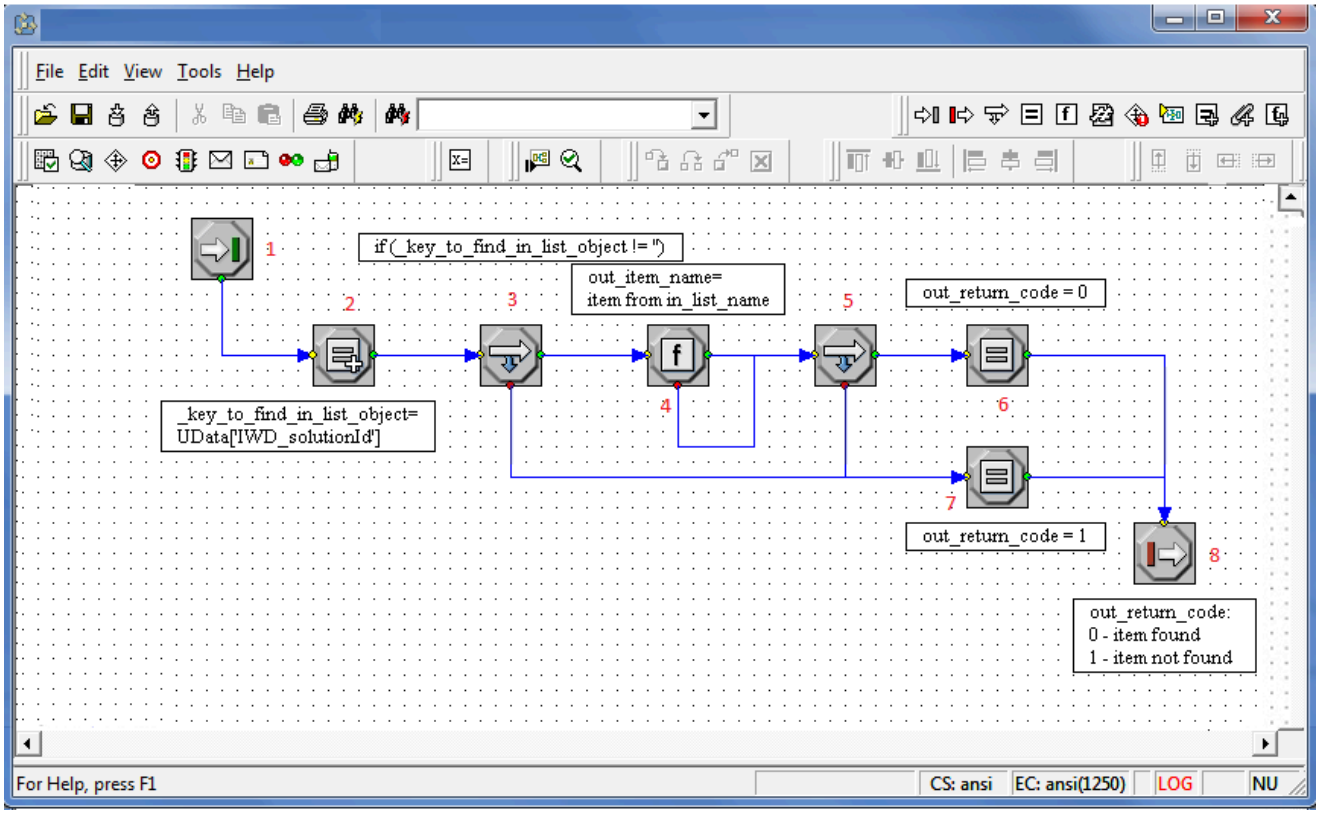
FindListObjectItem Subroutine

This subroutine amalgamates two subroutines that were previously separate in release 8.5.104:

- DetermineESPServerName
- DetermineRulePackageName

Flow Summary

Click to enlarge.



Flow Detail

1. Entry to FindListObjectItem subroutine.
2. Initialize variables:
   - `_key_to_find_in_list_object`—Assign task IWD_SolutionId.
   - `out_item_name`—Set its value to empty string.

3. Check if `_key_to_find_in_list_object` is not empty.
4. Search `_key_to_find_in_list_object` in `in_list_name`. Result will be assigned to `out_item_name`.
5. Check if `out_esp_name` is empty.
6. Set `out_return_code` to `0`.
7. Set `out_return_code` to 1.
8. Exit FindListObjectItem subroutine