# Composer Help

## Using eServices Blocks

12/13/2025

# Using eServices Blocks

## Contents

This page contains general information on working with eServices blocks.

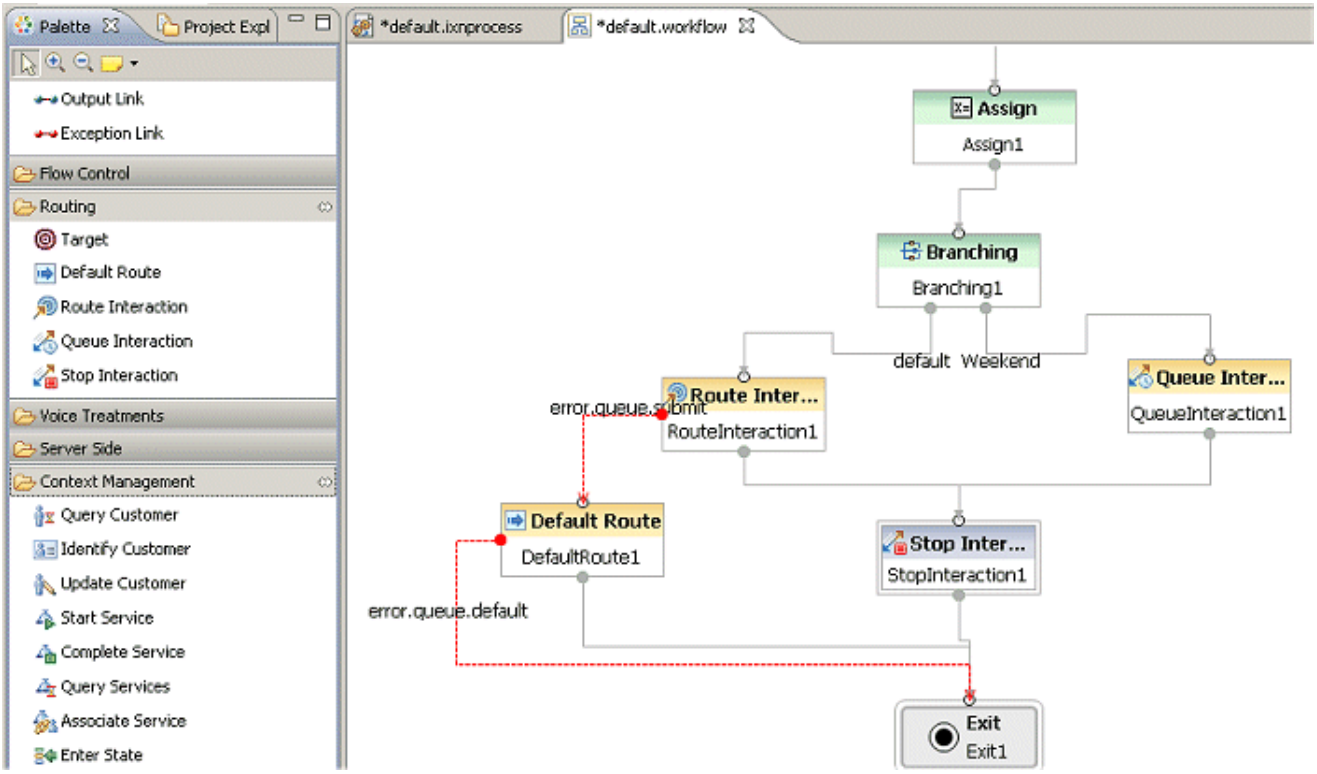
## Example Multimedia Workflow

In general, an interaction process diagram (IPD) for multimedia interactions works like this:

1. A media server (such as E-mail Server Java) directs Interaction Server to place an interaction into an inbound interaction queue.
2. Using the Views property, the interaction is then taken out of the queue and submitted to a routing workflow.
3. The workflow performs specialized processing and eventually routes the interaction to a target, but not necessarily the final target. For example, an e-mail interaction may be placed in an agent queue for construction of a response.
4. The target processes the interaction and places it into another queue where another workflow may process it. For example, a workflow may send an agent's draft e-mail response to a queue for Quality Assurance checking.
5. The cycle of going from queue/view/workflow continues until processing is stopped or the interaction reaches some final (usually an outbound) queue.

## Example Diagram

The figure below shows a sample multimedia workflow diagram in Composer Design perspective.




For other sample diagrams, see the Sample Applications topic. The default.workflow shown above works as follows:

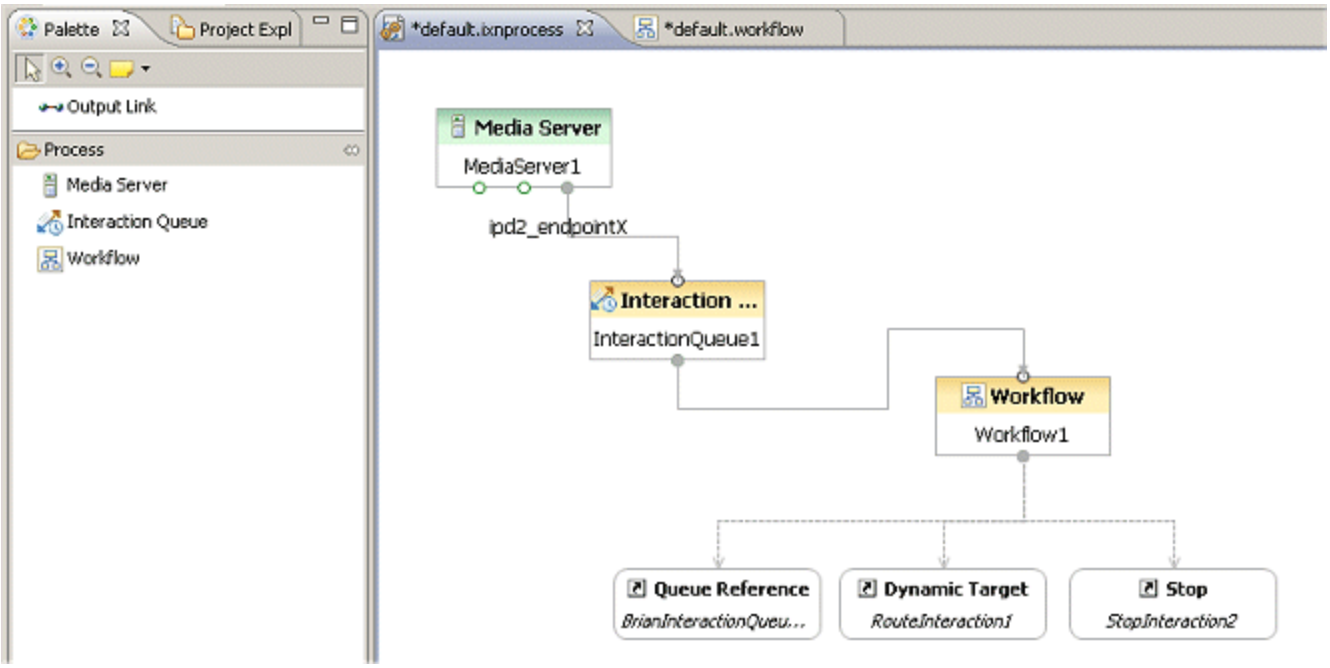
- The Entry block defines a variable called Today.
- The Assign block gives a value to the Today variable. In Expression Builder this is defined as:

```
data.Today==(_genesys.session.day.Saturday)||(data.Today==_genesys.session.day.Sunday)
```

- The Branching block Conditions property contains an expression used for segmenting interactions based on whether today is a week day or the weekend. The expression determines whether an interaction goes to a queue for the weekend crew or whether it is routed to a target.
- The Stop block notifies Interaction Server to stop processing and whether to notify Universal Contact Server about the interaction.

## Associated IPD

The figure below shows the IPD containing the Workflow block that points to the workflow diagram above. Three work-flow generated blocks are automatically generated in this example.




## eService Preparation

Genesys eService/Multimedia lets you process non-voice interactions in your contact center. At the center of the collection of components are:

- Interaction Server: Works with Universal Routing Server and Stat Server to process non-voice interactions by executing interaction process diagrams.
- Universal Contact Server (UCS): Works with its database to deliver customer contact history and information to the agent desktop.
- In addition, there are a number of other Multimedia servers that facilitate the handling of non-voice media, including E-mail Server Java, SMS Server, and Chat Server.

This help system assumes you have already installed and configured the eService/Multimedia components as described in the *eService/Multimedia 8.1 Deployment Guide*.

## Working with Returned Data

A few Composer Route blocks will return data back to the application:

- Email Response (Output Result)
- Create Email (Output Result)
- Create SMS (Output Result)

- Identify Customer
- Render Message (Result Property)
- Query Customer (Result Property)

Each qualifying block will expose a output result property (or equivalent) that will specify an application variable to store the results. These results will then be available in other blocks in the application for further processing. The format of returned data is usually JSON. Any post-processing work to be done on returned results can be done in the existing Assign block. It provides access to ECMAScript functions and supports writing simple or complex expressions to extract values out of JSON strings and arrays.

## Mandatory User Data for UCS Blocks

When working with the Update Contact and Render Message blocks (which map to Universal Contact Server services), certain properties must exist in the interaction User Data. These properties are:

- Update Customer Block `ContactId`
- Render Message Block `ContactId` (if some contact-related Field Codes (as described in the *eServices 8.1 User's Guide*) are used in the text to render); `InteractionId` (if some interaction-related Field Codes are used in the text to render); `OwnerEmployeeId` (if some agent-related Field Codes are used in the text to render).

As is the case with IRD, these properties are not set in the blocks themselves. Instead, the properties are assumed to be put in the interaction's User Data by some other block earlier in the workflow, such as the Identify Customer block or Create Interaction block with the Update User Data property set to true. In case no other block does this, the User Data block may be used for this purpose.

> **Important**
>
> If those properties are not available, an explicit UCS error message (missing parameter) shows in the Orchestration Server log.

## Working with Child interactions

The Composer Create Email, Email Forward, and Create SMS blocks create new interactions as part of block execution and these new interactions can be controlled using queue, `detach` and `associate` attributes. Optionally, these interactions can be queued to start a new workflow strategy. For details please refer to the comparison table below.

### Detach Delay for New Interaction

While detaching a new interaction, in the case of an ACK or Response, it might take a few seconds to receive the new interaction. During this time, the application "Detach" loop logic could exceed the

Orchestration Server `scxml/max-state-entry-count` value. As a workaround in this case, set the ORS `scxml/max-state-entry-count` value higher than 150. 200 seems to be a optimal value.

## Associate New Interaction

This property, introduced in Composer 8.1.420.14 for the Chat Transcript, Create Email, Create SMS, Email Forward, and Email Response blocks, supports the Orchestration Server <ixn:createmessage> tag `associate` attribute. This property requires Orchestration Server version 8.1.400.45+.

## <ixn:createmessage> Attributes Behavior Comparisons

| queue | associate | detach | runtime behavior | comments |
|---|---|---|---|---|
| Queue specified | true | true | Session will get `interaction.added` and `interaction.present` events for the new interaction. | New interaction detached from the session and does not get pulled automatically from the queue. Use a Queue Interaction block to manually queue the interaction. |
| Queue specified | true | false | Session will get `interaction.added` and interaction.present events for the new interaction. | ORS pulls the new interaction from the queue and starts new session. |
| Queue specified | false | true | Session will not get `interaction.added` and `interaction.present` events for the new interaction. | 1. ORS pulls the new interaction from the queue and starts new session.<br>2. Detach operation is not possible in this case since the interaction is not associated with the session. Error message: No interaction with specified ID' |
| Queue specified | false | false | Session will not get `interaction.added` and `interaction.present` events for the new interaction. | ORS pulls the new interaction from the queue and starts new session. |
| No value | true/false | false | Session will not get `interaction.added` | Recommended approach by Platform |

| queue | associate | detach | runtime behavior | comments |
|---|---|---|---|---|
| | | | and `interaction.present` events<br><br>for the new interaction. Platform automatically handles/sents the ixn using internal queue. | |
| No value | true/false | true | Session will not get `interaction.added` and `interaction.present` events<br><br>for the new interaction. | Detach attempt will result in error. "Error message: No interaction with specified ID" |

**Note:** If the "queue'" attribute is not specified the associate attribute does not have any effect at all. Use the `associate` attribute to handle the interaction.

## Detach - Handle interaction.deleted for the New Interaction

If you are using a Composer versions prior to 8.1.400.35, adding the `interaction.deleted` event in the Entry block would add a "condition-less" transition. This might handle the `interaction.deleted` event from the new interaction and ends/takes the error path for the workflow. Also, the automatically added `interaction.deleted` handler in the block, while setting Detach property to `true`, would ignore this new `interaction.deleted` event due to the `interactionId` condition check for the original interaction.

As a workaround in this case, add the following changes to the diagram.

In the block:

1. Create a User variable to collect the new interaction's `interactionID`.
2. Assign this new variable in the Output Result property to collect the result.
3. Open the Exceptions property in the block, and add the `interaction.deleted` event.
4. In the Configure Exceptions dialog, uncheck the Target box to make this a "target-less" transition.
5. Add the condition to check for the new interaction. (`_event.data.interactionid == varChildIxn`).

In the Entry Block:

Make sure the `interaction.deleted` event handler has the following condition set to handle the `interaction.deleted` event only if the original interaction was deleted and not as a result of the Detach operation:

```
_event.data.interactionid == system.InteractionID && (!_event.data.resultof ||
_event.data.resultof == 'deletion')
```