



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.

Developer's Guide

Customizing the Chat Routing Strategy

4/12/2025

Customizing the Chat Routing Strategy

Contents

- **1 Customizing the Chat Routing Strategy**
 - 1.1 Main Interaction Workflow
 - 1.2 Routing to a Static Agent Group
 - 1.3 Sending Messages from the Chat Routing Strategy into the Chat Session

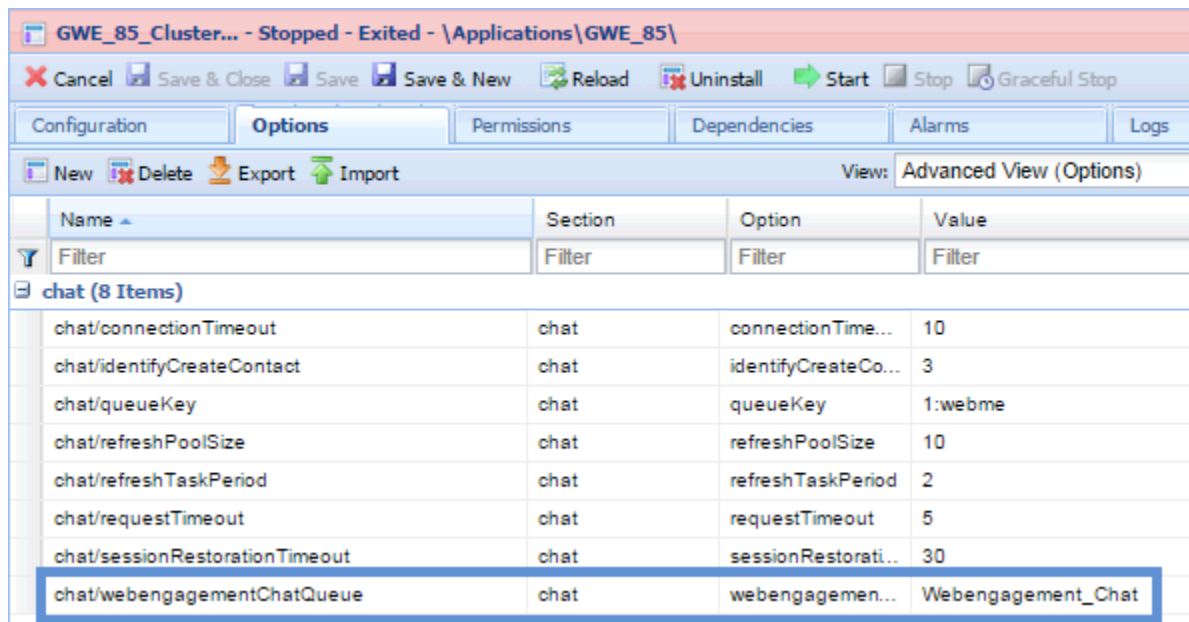
When you create your Web Engagement application, Genesys Web Engagement also creates default Engagement Logic and Chat Routing **SCXML strategies** in the **\apps\application_name\resources_composer-projects** folder. Orchestration Server (ORS) uses these strategies to decide whether and when to make a proactive offer and which channels to offer (chat or web callback).

The default Chat Routing strategy delivers chat interactions that are initiated in Genesys Web Engagement to a specific target. Although this strategy is included as part of the Web Engagement installation, it is possible to use your own existing strategy for routing. For example, a URS-based chat routing strategy; however, in this scenario you will need to adjust the Web Engagement solution to support the pacing algorithm functionality.

You can modify the Chat Routing SCXML by **importing the Composer project into Composer**. The project is located here: **\apps\application_name\resources_composer-projects\WebEngagement_ChatRouting**. Refer to the sections below for details about the Chat Routing strategy and how it can be modified.

Main Interaction Workflow

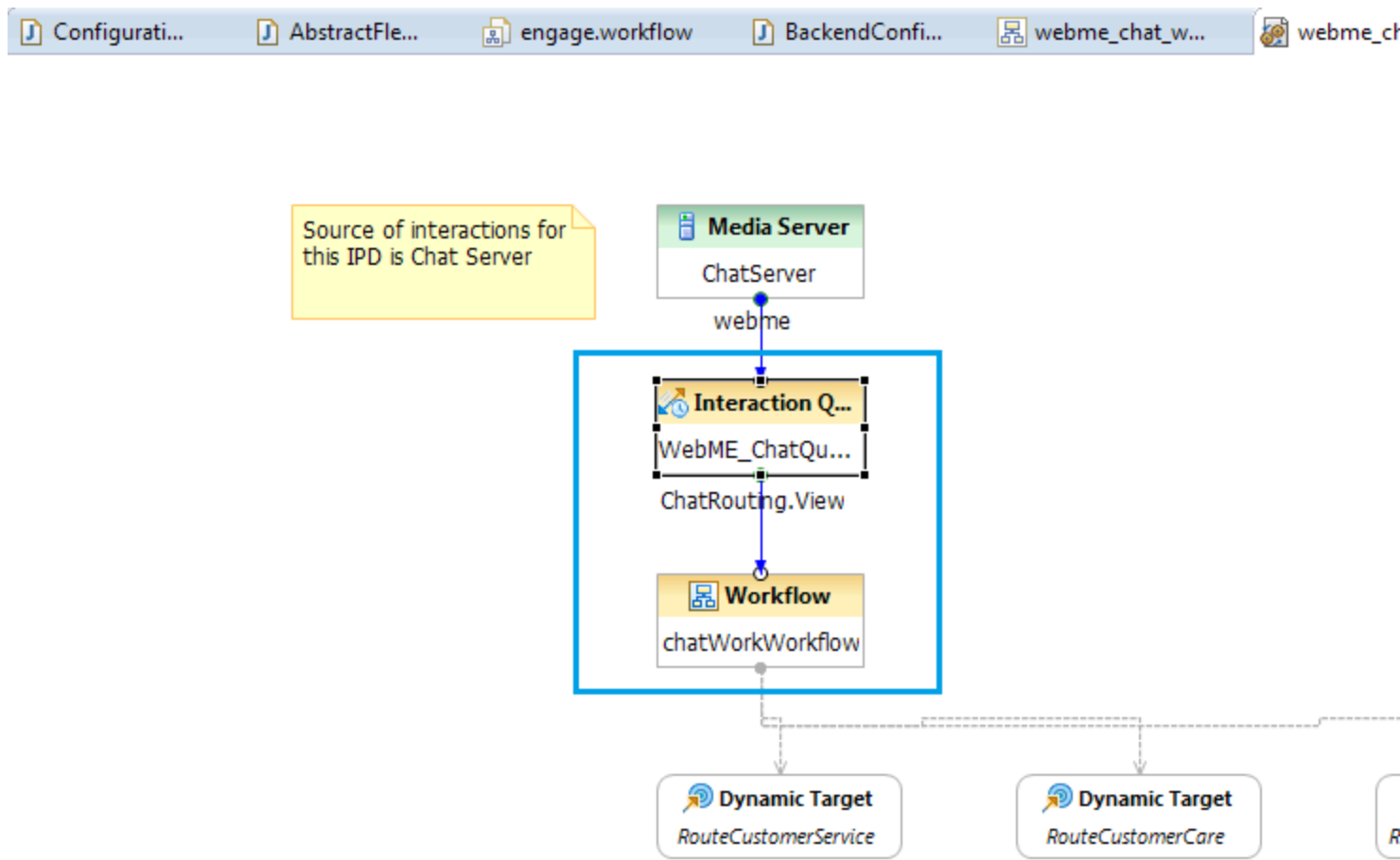
The default entry point to the GWE Chat Routing strategy is the Interaction Queue specified in the **webengagementChatQueue** option on the Web Engagement Cluster application.



Name	Section	Option	Value
chat/connectionTimeout	chat	connectionTime...	10
chat/identifyCreateContact	chat	identifyCreateCo...	3
chat/queueKey	chat	queueKey	1:webme
chat/refreshPoolSize	chat	refreshPoolSize	10
chat/refreshTaskPeriod	chat	refreshTaskPeriod	2
chat/requestTimeout	chat	requestTimeout	5
chat/sessionRestorationTimeout	chat	sessionRestorati...	30
chat/webengagementChatQueue	chat	webengagemen...	Webengagement_Chat

The Interaction Queue.

The interaction process pulls interactions from this queue and sends them through the chat workflow:



Markers Properties Servers Data Source Explorer Snippets Console Progress Search TestNG

Interaction Queue

Core	Property	Value
Appearance	Alias	
	Name	WebME_ChatQueue
	Annotation	
	Block Notes	
	Configuration Server	
	Object Name	Webengagement_Chat
	Queue	

The chat workflow

Important
 If you decide to change the value of **queueWebengagement**, make sure to also

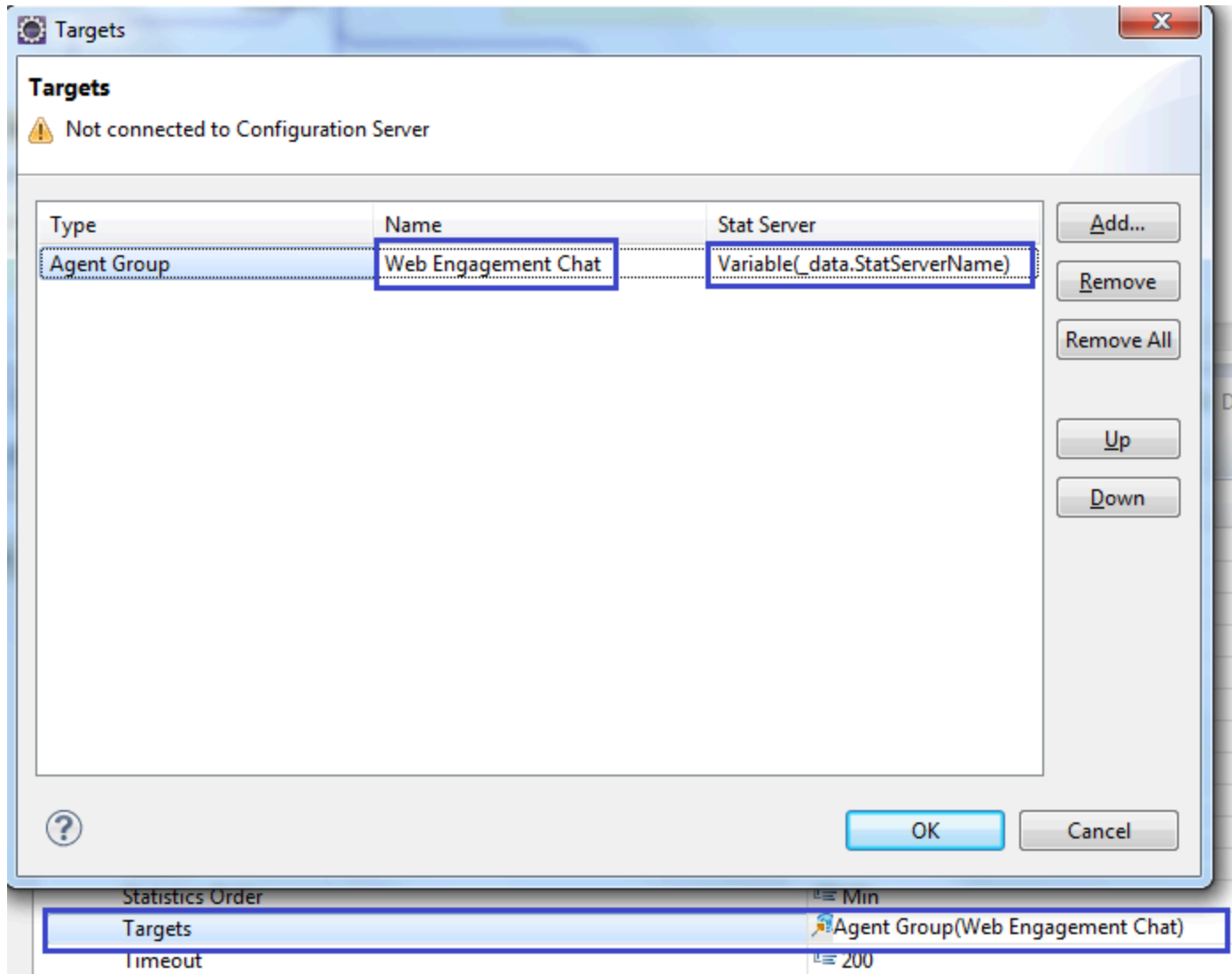
adjust the name of the queue in the Chat Routing strategy.

The default Chat Routing strategy is straightforward and includes the following highlights in the workflow:

1. Obtain information from the User Data of the chat interaction that is being routed. See the **AssignCategory** block in the Chat Routing Strategy for details.
2. Send messages to the chat session from the routing strategy. See [Sending Messages from the Chat Routing Strategy into the Chat Session](#) for details.
3. Branch the workflow based on categories obtained from the chat interaction User Data. See the **BranchingByCategory** block for details.
4. Route to skill-based Virtual Groups. See the **RouteCustomerServer** and **RouteCustomerCare** blocks for details.
5. Route to a static Agent Group. See [Routing to a Static Agent Group](#) for details.

Routing to a Static Agent Group

When you plan to route an interaction to a static Agent Group, you should specify the name of this group and the name of the Stat Server in the Target property of the **RouteInteractionDefault** block.



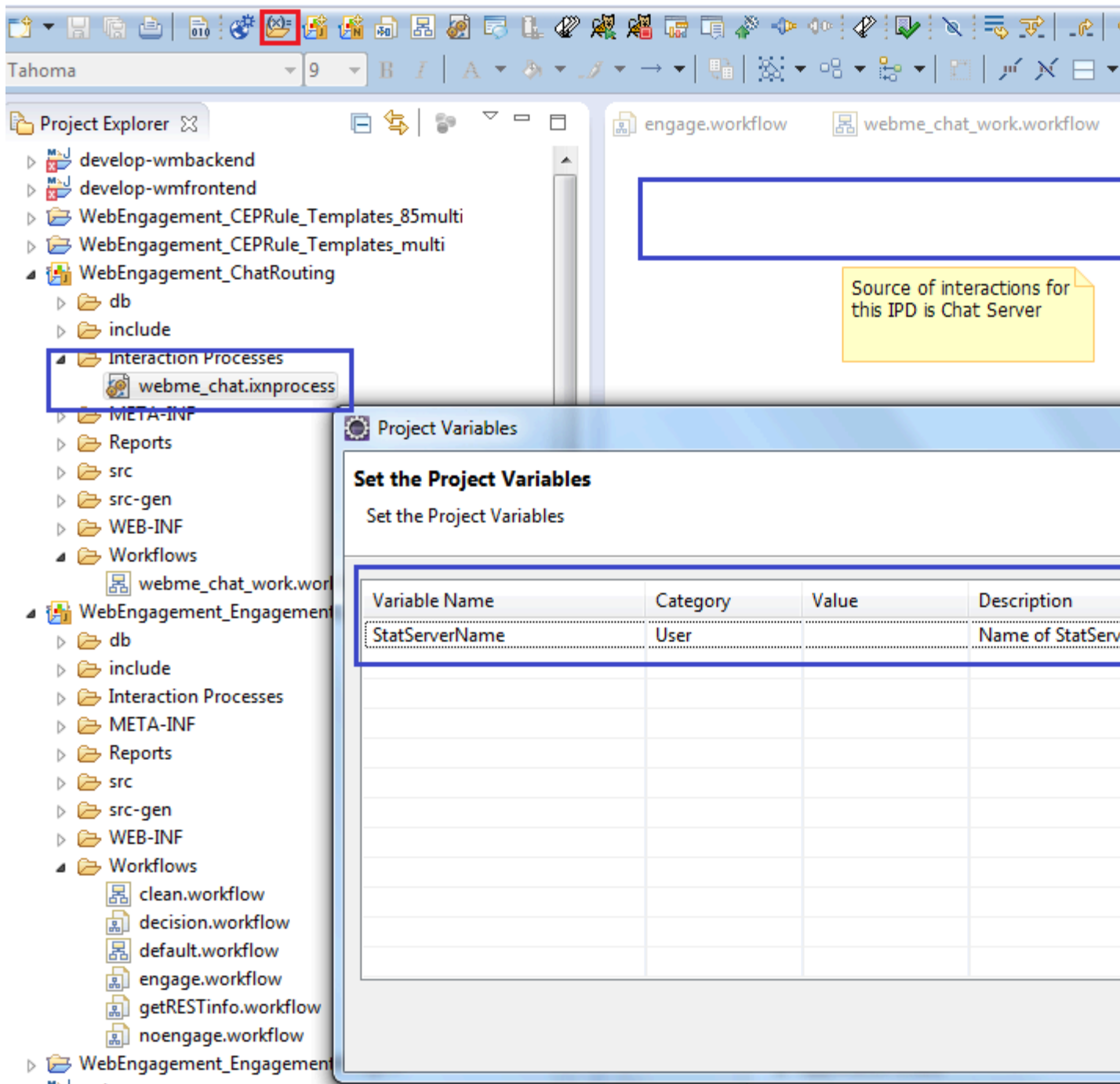
The Target property of the **RouteInteractionDefault** block.

In the image above, the Stat Server name is specified through the `Variable(_data.StatServerName)` variable. You can define this variable, or others like it, in Composer and Genesys Administrator.

Specifying Variables in Composer

Start

1. Double click the interaction process - in this case, `webme_chat.ixnprocess`.
2. Make sure that there are no elements selected in the opened interaction process.
3. Access the interaction process variables by clicking "Access Project Variables", marked with a red square in the image below:



Access the project variables

In the image above, the StatServerName variable is used in the default Chat Routing strategy.

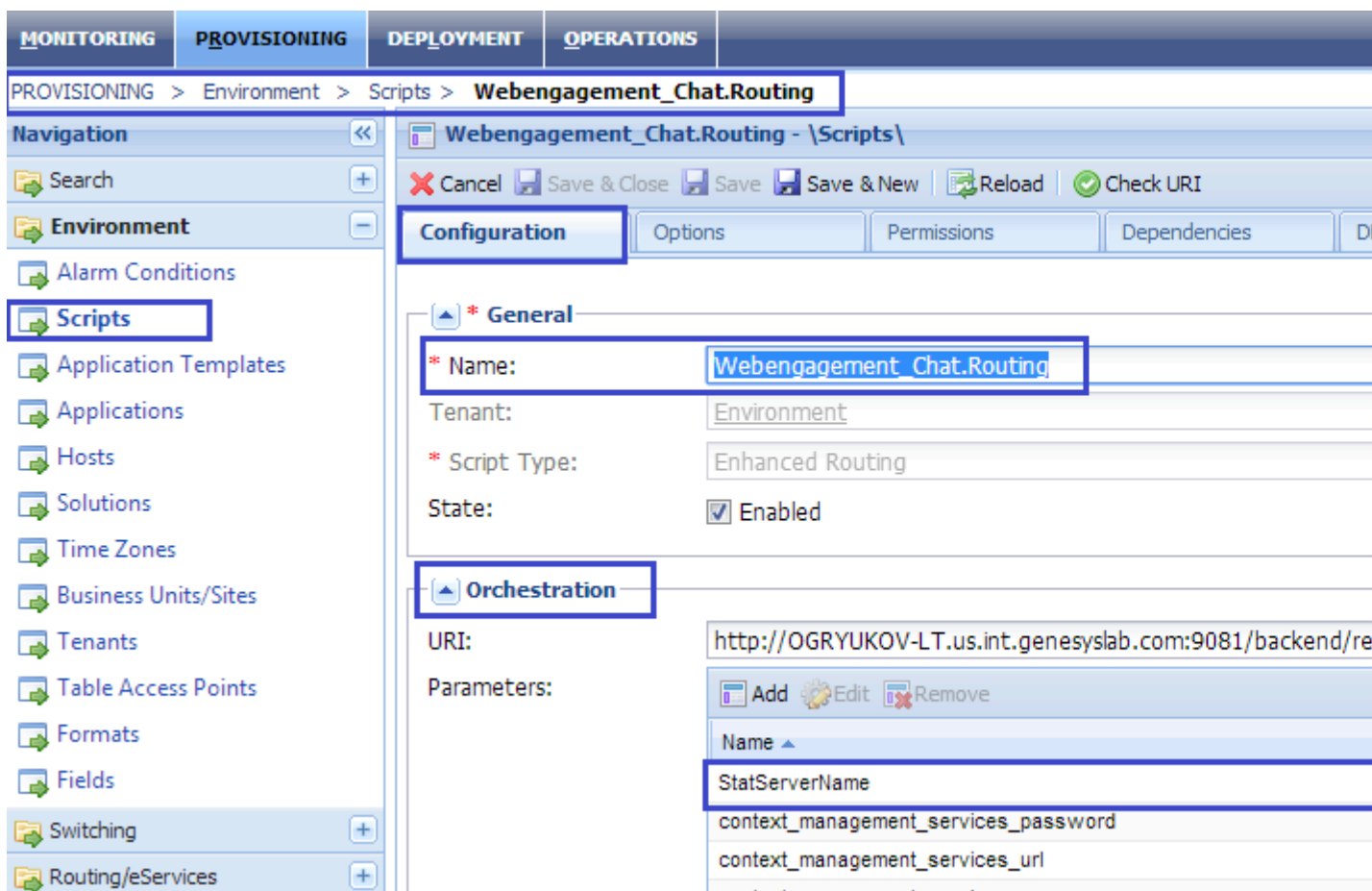
End

Specifying Variables in Genesys Administrator

The StatServerName parameter is set automatically by the **Provisioning Tool** when you install Genesys Web Engagement, but it can be changed manually.

Start

1. Navigate to Provisioning > Environment > Scripts and find the script with the entry-point Interaction Queue. In this case, the script is Webengagement_Chat.Routing.
2. In the Configuration tab, open the Orchestration section.
3. Now you can see a list of parameters that are passed into the Chat Routing strategy, including StatServerName.



The StatServerName parameter.

End

Sending Messages from the Chat Routing Strategy into the Chat

Session

There are times when you might need to send messages into the chat session directly from the routing strategy. For example, this could be additional information messages, advertising messages, and so on.

The default Chat Routing strategy contains an **External Service** block that provides this functionality:

The screenshot shows a workflow editor with three tabs: `engage.workflow`, `webme_chat_work.workflow`, and `webme_chat.ixnprocess`. The workflow diagram includes:

- Entry** block (Entry1): A black circle icon. Callout: "Entry block is used to began an application, and to define and initialize system(predefined) and user(custom) variables".
- Assign** block (AssignCategory): A green box with an 'X=' icon. Callout: "Assign block is used to assign a category to a variable 'categories', depending on user data parsing".
- External Service** block (SendMsgToChat...): A dashed box with a blue icon. Callout: "It is possible to send message into chat session prior it is routed to an agent. Enable this block to turn on demonstration of this feature".

Below the workflow is a detailed view of the **External Service** block's properties:

Model	Property	Value
Appearance	Alias	
	Name	SendMsgToChatSession
	Annotation	
	Block Notes	
	Exceptions	
	Exceptions	
	External Service Details	
	Application	Chat Server()
	Method Name	Message
	Method Parameters	MessageText= 'You can specify post
Service Name	Chat	
Service Timeout	10	

The External Services block lets you send message from the routing strategy.

Important

The **External Service** block is disabled by default.