

Administering webMethods CloudStreams

Version 9.8

April 2015

This document applies to webMethods CloudStreams Version 9.8 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2013-2015 Software AG, Darmstadt, Germany and/or Software AG USA Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Table of Contents

About this Guide.....	9
Document Conventions.....	9
Online Information.....	10
Overview.....	11
About CloudStreams.....	12
The Components of CloudStreams.....	13
Outbound Architecture.....	14
Inbound Architecture.....	15
The CloudStreams Connectors.....	17
Custom Cloud Connectors.....	17
Configuring a CloudStreams Connector.....	17
Cloud Connections.....	18
Cloud Connector Services.....	18
Virtual Services and Connector Virtual Services.....	19
The Processing Steps of Virtual Services.....	19
The Processing Steps of Connector Virtual Services.....	20
Policies for Virtual Services and Connector Virtual Services.....	21
Policies for Virtual Services.....	21
Policies for Connector Virtual Services.....	22
The CloudStreams Analytics Dashboard.....	22
Getting Started with CloudStreams.....	23
CloudStreams Configuration Options.....	27
Overview.....	28
The Administration Options.....	28
Setting the General Options.....	28
Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts.....	31
Setting the Database Options for Publishing Performance Metrics and Events.....	33
The Intervals for Metric Publishing.....	34
The Virtual Services Option.....	36
Setting the STS Options.....	36
Setting the Service Fault Configuration Options.....	40
Setting the Consumers Options.....	42
Setting the OAuth Tokens Options.....	44
The Providers Options.....	47
The Streaming Options.....	48
Setting the Streaming Providers Options.....	48
Setting the Streaming Subscribers Options.....	50
Creating an ESB Service as a Streaming Subscriber Destination.....	52

ESB Services for the Client Type of Comet.....	52
ESB Services for the Client Type of HTTP.....	53
Streaming subscriber behavior in case of an error.....	53
Virtual Services.....	55
Overview.....	56
Define a CloudStreams Server Target.....	56
Create a CloudStreams Governance Project.....	56
Create the Virtual Services.....	56
The Processing Steps of a Virtual Service.....	57
The Processing Steps of a Connector Virtual Service.....	58
Create Policies for the Virtual Services.....	58
Policies for Connector Virtual Services.....	58
Defining and Managing a CloudStreams Server Target.....	59
Creating a CloudStreams Governance Project.....	60
Creating a New Virtual Service (SOAP).....	62
The Properties of a Virtual Service (SOAP).....	63
Managing a Virtual Service (SOAP).....	65
The "In Sequence" Step (SOAP).....	66
The "Entry Step" (SOAP).....	67
The "Transform" Step (Inbound, SOAP).....	68
The "Invoke IS Service" Step (Inbound, SOAP).....	69
The "Routing Rule" Step for HTTP or HTTPS (SOAP).....	71
The "Straight Through" Routing Rule Step (SOAP).....	71
The "Context-Based" Routing Rule Step (SOAP).....	73
The "Content-Based" Routing Rule Step (SOAP).....	77
The "Load Balancing" Routing Rule Step (SOAP).....	79
The "Out Sequence" Step (SOAP).....	82
The "Transform" Step (Outbound, SOAP).....	83
The "Invoke IS Service" Step (Outbound, SOAP).....	84
The "Error Sequence" Step (SOAP).....	85
The Fault Handler Variables.....	87
IS Service Constructs.....	89
Using the Security API in IS Services.....	89
pub.cloudstreams.security.ws.AddUserNameToken.....	89
pub.cloudstreams.security.ws.AddX509Token.....	93
pub.cloudstreams.security.ws.AddSamlSenderVouchesToken.....	94
pub.cloudstreams.security.ws.AddTimestamp.....	95
pub.cloudstreams.security.ws.AddWSAddressingHeaders.....	97
Using Context Variables in IS Services.....	100
The Predefined Context Variables.....	100
The API for Context Variables.....	105
Creating a New Connector Virtual Service (SOAP).....	118
The Properties of a Connector Virtual Service (SOAP).....	120
Managing a Connector Virtual Service (SOAP).....	122

The "In Sequence" Step (Connector Virtual Service, SOAP).....	123
The "Entry Step" (Connector Virtual Service, SOAP).....	123
The "Routing Rule" Step (Connector Virtual Service, SOAP).....	124
The "Invoke IS Service" Step (Inbound, Connector Virtual Service, SOAP).....	125
The "Out Sequence" Step (Connector Virtual Service, SOAP).....	125
The "Error Sequence" Step (Connector Virtual Service, SOAP).....	125
Creating a New Virtual Service (REST).....	126
The Properties of a Virtual Service (REST).....	128
Managing a Virtual Service (REST).....	130
The "In Sequence" Step (REST).....	131
The "Entry Step" (REST).....	132
The "Transform" Step (Inbound, REST).....	133
The "Invoke IS Service" Step (Inbound, REST).....	135
The "Routing Rule" Step (REST).....	136
The "Straight Through" Routing Rule Step (REST).....	136
The "Context-Based" Routing Rule Step (REST).....	139
The "Content-Based" Routing Step (REST).....	141
The "Load Balancing" Routing Rule Step (REST).....	144
The "Out Sequence" Step (REST).....	147
The "Transform" Step (Outbound, REST).....	147
The "Invoke IS Service" Step (Outbound, REST).....	148
The "Error Sequence" Step (REST).....	150
Creating a New Connector Virtual Service (REST).....	152
The Properties of a Connector Virtual Service (REST).....	154
Managing a Connector Virtual Service (REST).....	156
The "In Sequence" Step (Connector Virtual Service, REST).....	157
The "Entry Step" (Connector Virtual Service, REST).....	157
The "Routing Rule" Step (Connector Virtual Service, REST).....	158
The "Invoke IS Service" Step (Inbound, Connector Virtual Service, REST).....	159
The "Out Sequence" Step (Connector Virtual Service, REST).....	159
The "Error Sequence" Step (Connector Virtual Service, REST).....	159
Important Considerations for REST Virtual Services.....	160
HTTP Method/Content-Type Combinations.....	161
Changing the HTTP Method of a REST Virtual Service.....	162
Implications of Changing HTTP Methods.....	162
Changing HTTP Methods in Requests Dynamically Using a Context Variable.....	165
Sample XSLT Transformation for GET-to-POST or GET-to-PUT.....	165
Policies.....	167
Overview of Policies.....	168
Policies for Virtual Services.....	168
WS-SecurityPolicy 1.2 Actions.....	168
Monitoring Actions.....	169
Additional Actions.....	170
Policies for Connector Virtual Services.....	170

Creating Policies.....	171
Modifying Policies.....	172
The Policy Actions.....	173
Action Evaluation Order and Dependencies.....	173
Usage Cases for Identifying/Authenticating Consumers.....	175
Multiple Security Elements in Requests/Responses.....	176
The Policy Action Reference.....	177
Authorize User.....	177
Identify Consumer.....	178
Include Timestamps.....	179
Log Invocation.....	180
Monitor Service Performance.....	181
Monitor Service Level Agreement (SLA).....	184
Require Encryption.....	188
Require HTTP Basic Authentication.....	190
Require SAML Token.....	190
To Identify a Trusted STS to Integration Server.....	192
Require Signing.....	192
Require SSL.....	194
Require WSS Username Token.....	194
Require X.509 Token.....	195
Throttling Traffic Optimization.....	195
Validate Schema.....	198
Deploying Virtual Services and Connector Virtual Services.....	199
Overview of Deployment.....	200
Before You Deploy Virtual Services or Connector Virtual Services.....	200
Deploying All Services in a CloudStreams Governance Project.....	201
Deploying a Single Service.....	202
Undeploying Services.....	202
What Happens When You Deploy a Service?.....	202
Policy Validation.....	203
Policy Conflict Resolution Rules.....	203
What if You Need to Modify Deployed Services?.....	205
CloudStreams Analytics.....	207
Overview.....	208
The Run-Time Events.....	209
The Key Performance Indicator (KPI) Metrics.....	209
Installation and Configuration Tasks.....	210
Installing MashZone and Starting the MashZone Server.....	211
Copying the Required Resource Files for CloudStreams Analytics.....	211
Creating the Schema and Tables Required by CloudStreams Analytics.....	212
Creating a JDBC Connection Pool for CloudStreams.....	215
SQL Statement Execution and Connection Timeouts.....	217
Configuring the Data Feed.....	218

Importing the CloudStreams Analytics MashApp.....	218
Creating New Users for CloudStreams Analytics.....	219
Setting the Publishing Options for Performance Metrics and Events.....	219
Using the CloudStreams Analytics Dashboard.....	220
Cloud Connector Performance Summary View.....	220
Cloud Connector API Usage View.....	221
Cloud Connector Performance View.....	222
Cloud Connector Error View.....	224
Cloud Connector "Service Level Agreement" (SLA) View.....	225
Service View.....	226
Transaction View.....	228
Creating Custom Cloud Connectors.....	231
Overview of Creating Custom Cloud Connectors.....	232
Creating and Managing CloudStreams Provider Projects.....	233
Creating CloudStreams Provider Projects.....	233
Managing CloudStreams Provider Projects.....	234
Creating Custom SOAP or REST Cloud Connectors.....	234
Configuring Custom SOAP Cloud Connectors.....	238
Creating Run-Time Connections (SOAP).....	239
Configuring Cloud Connector Services (SOAP).....	249
Configuring Custom REST Cloud Connectors.....	255
Creating Run-Time Connections (REST).....	255
Configuring Cloud Connector Services (REST).....	265
Understanding REST Parameters.....	272
Parameter Types.....	272
URI_CONTEXT.....	272
QUERYSTRING_PARAM.....	273
CFG_PARAM.....	273
Parameter Data Types.....	273
String.....	273
Record.....	274
Parameter Formatters for REST Connector Services.....	274
Using the Default Parameter Formatter.....	275
Implementing a Custom Parameter Formatter.....	276
Handling JSON representations of REST resources.....	277
Support for multiple root JSON data format with anonymous collection.....	278
Viewing the Constraints Applied to Variables.....	279
Deploying Custom Cloud Connectors.....	280
Managing Custom Cloud Connectors.....	280
Managing the Custom Cloud Connector Packages.....	281
Admin Folder.....	283
Summary of Elements in this Folder.....	284
pub.cloudstreams.admin.connection:disableConnection.....	284
pub.cloudstreams.admin.connection:enableConnection.....	285

pub.cloudstreams.admin.connection:getConnectionStatistics.....	285
pub.cloudstreams.admin.connection:queryConnectionState.....	286
pub.cloudstreams.admin.service:update.....	286
Advanced Settings.....	289
Introduction.....	291
pg.backupFailedProxies.....	291
pg.CollectionPool.....	291
pg.CollectionWorkQueue.....	292
pg.debug.....	292
pg.delayedRefresher.....	292
pg.ehcache.config.....	292
pg.email.....	292
pg.endpoint.....	294
pg.failedProxies.....	294
pg.http.....	294
pg.IntervalPool.....	294
pg.jaxbFileStore.....	295
pg.jdbc.....	295
pg.keystore.....	295
pg.lb.....	296
pg.passman.....	296
pg.PgMenConfiguration.....	296
pg.PgMenSharedCacheManager.....	297
pg.PgMetricsFormatter.....	297
pg.policygateway.....	298
pg.proxyLoader.....	298
pg.rampartdeploymenthandler.....	298
pg.ReportingPool.....	298
pg.ReportingWorkQueue.....	299
pg.serviceReader.....	299
wst.rest.json.operation.multiroot.wrapPayload.....	299
wst.cloudConnectorService.....	299
wst.connfactory.....	300
wst.default.tenant.....	300
wst.product.....	300

About this Guide

webMethods CloudStreams provides a scalable approach for the development and governance of integration flows between "Software as a Service" (SaaS) applications and "on-premise" applications.

CloudStreams provides predefined, configurable CloudStreams Connectors that enable you to connect to popular SaaS cloud applications. Using the CloudStreams plug-ins and administrative options, you configure these Connectors to govern transactions, log payloads and monitor run-time performance. You can publish and view run-time performance metrics and events using the Software AG MashZone dashboards provided by CloudStreams.

Additionally, you can use the CloudStreams framework to create custom connectors to other SaaS applications.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies storage locations for services on webMethods Integration Server, using the convention <i>folder.subfolder:service</i> .
UPPERCASE	Identifies keyboard keys. Keys you must press simultaneously are joined with a plus sign (+).
<i>Italic</i>	Identifies variables for which you must supply values specific to your own situation or environment. Identifies new terms the first time they occur in the text.
Monospace font	Identifies text you must type or messages displayed by the system.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the symbol.

Convention	Description
[]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at <http://documentation.softwareag.com>. The site requires Empower credentials. If you do not have Empower credentials, you must use the TECHcommunity website.

Software AG Empower Product Support Website

You can find product information on the Software AG Empower Product Support website at <https://empower.softwareag.com>.

To submit feature/enhancement requests, get information about product availability, and download products, go to [Products](#).

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the [Knowledge Center](#).

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at <http://techcommunity.softwareag.com>. You can:

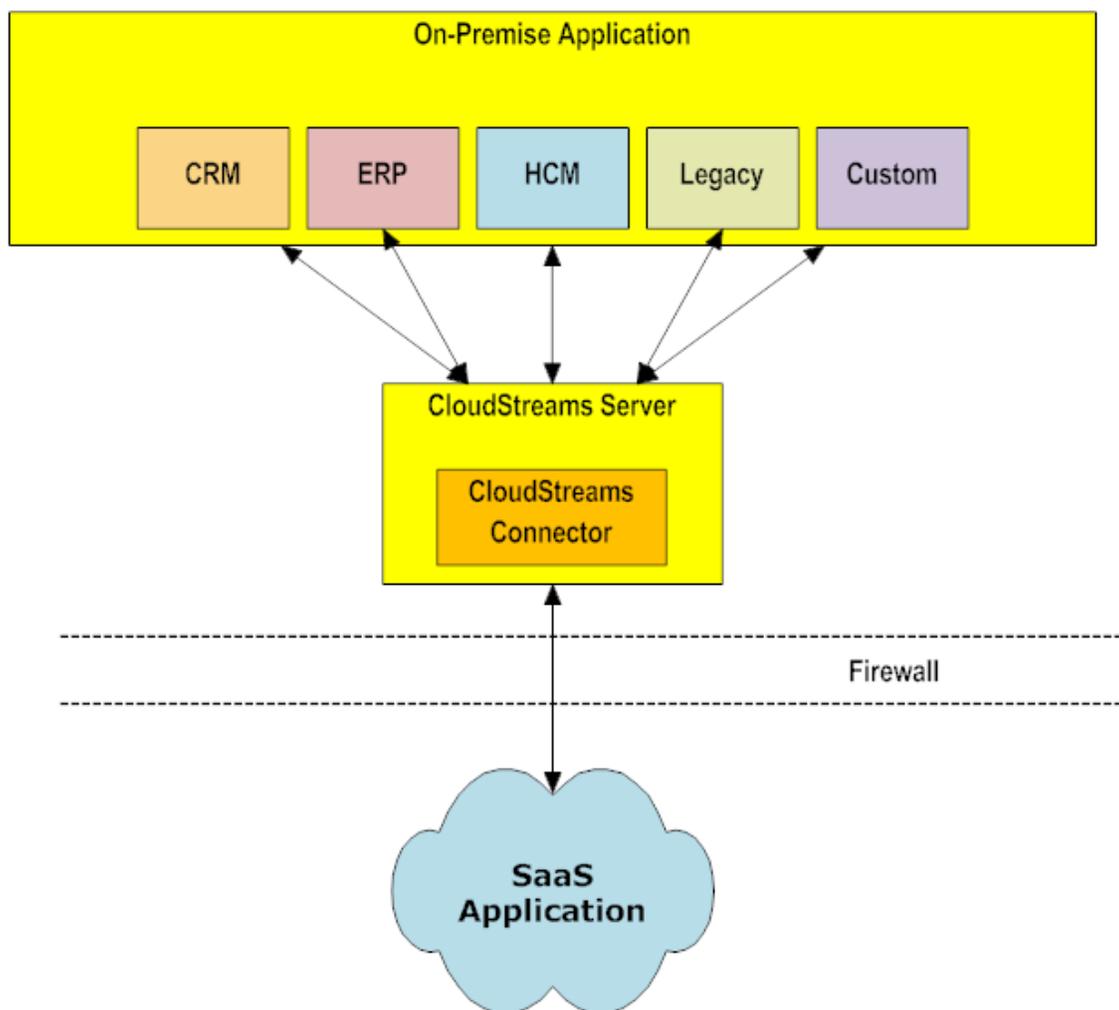
- Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

1 Overview

- About CloudStreams 12
- Getting Started with CloudStreams 23

About CloudStreams

webMethods CloudStreams provides a scalable approach for the development and governance of integration flows between "Software as a Service" (SaaS) applications and "on-premise" applications.



You use CloudStreams to:

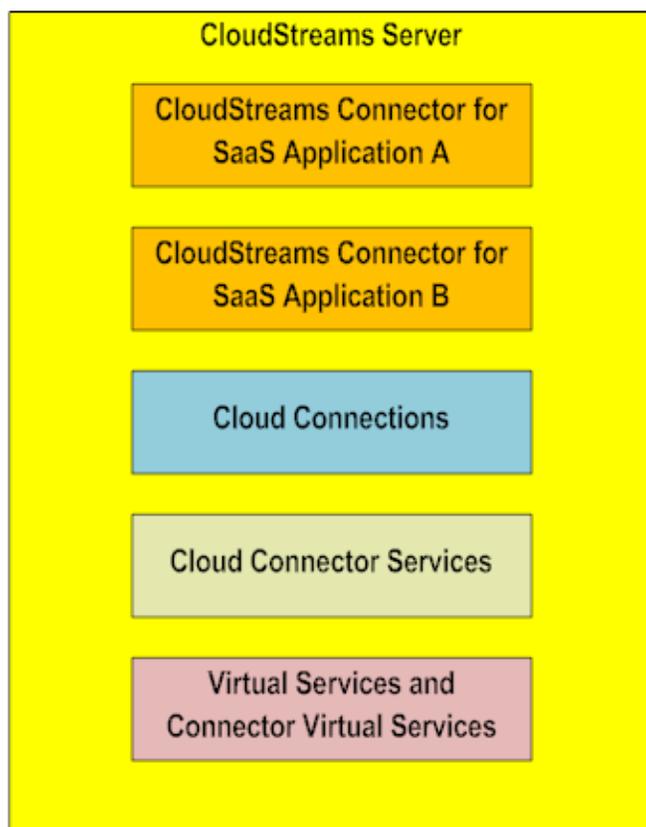
- Develop services for a CloudStreams connector.
- Manage the security of service requests.
- Perform user-defined processing instructions (optional).
- Log transactions.
- Monitor run-time performance and events.

Additionally, you can use the CloudStreams framework to create custom cloud connectors for other SaaS applications.

The Components of CloudStreams

CloudStreams consists of the following components:

- **The CloudStreams server**, which uses the same run-time framework as the one used by Integration Server. CloudStreams runs as a package on top of Integration Server. A CloudStreams server hosts the following components:

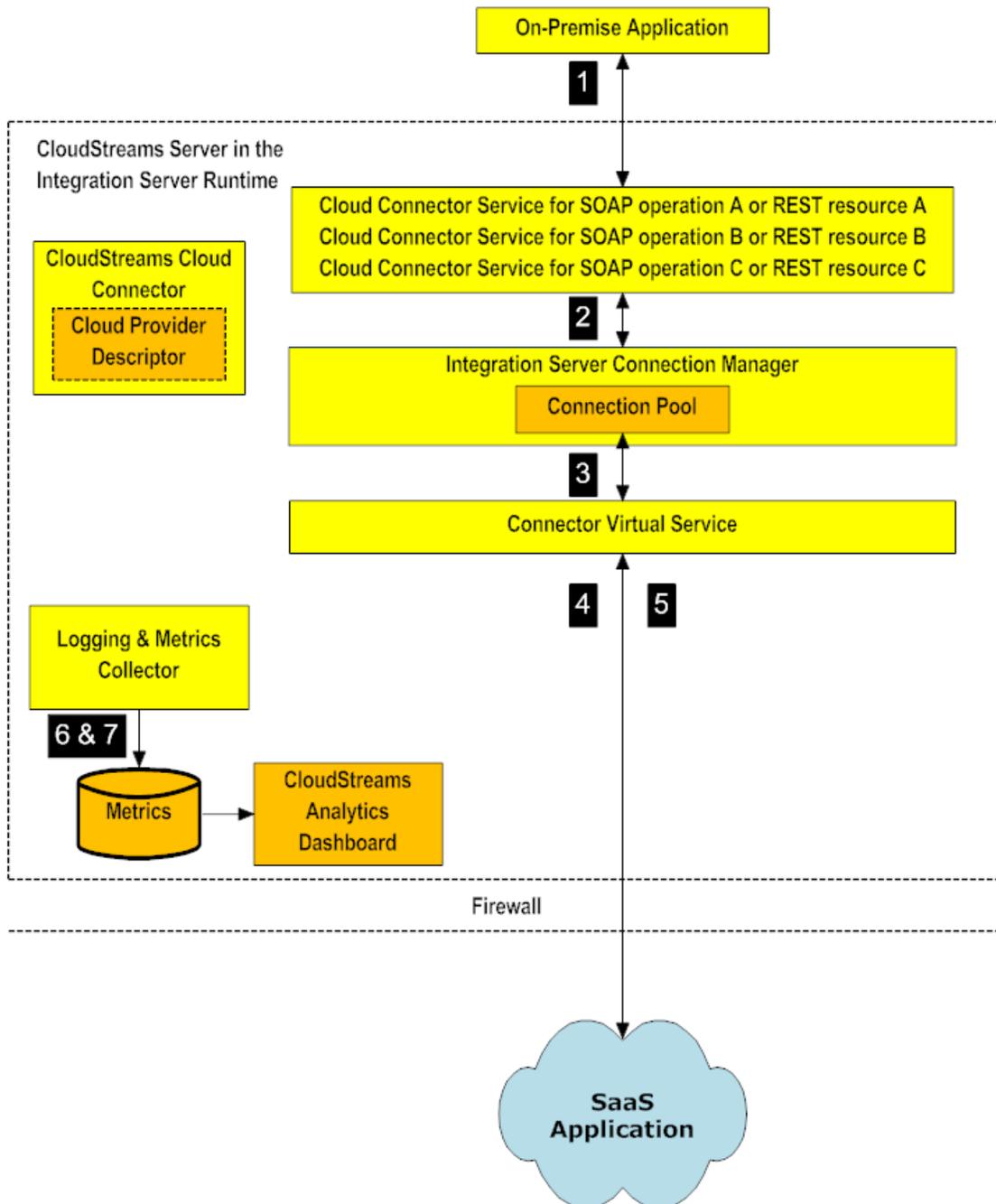


- **Plug-ins provided by Software AG Designer**, which you use as the design-time tool for CloudStreams:
 - You use the Service Development plug-in to create cloud connector services, which perform SOAP operations or access REST resources.
 - You use the CloudStreams Development plug-in to create virtual services, connector virtual services, and their policies. These services send the requests from the on-premise application to the native service in the SaaS application, and vice versa.

- **The CloudStreams Analytics dashboard**, which you use to view and analyze the run-time performance metrics and run-time events of your CloudStreams connectors, cloud connector services and consumers.

Outbound Architecture

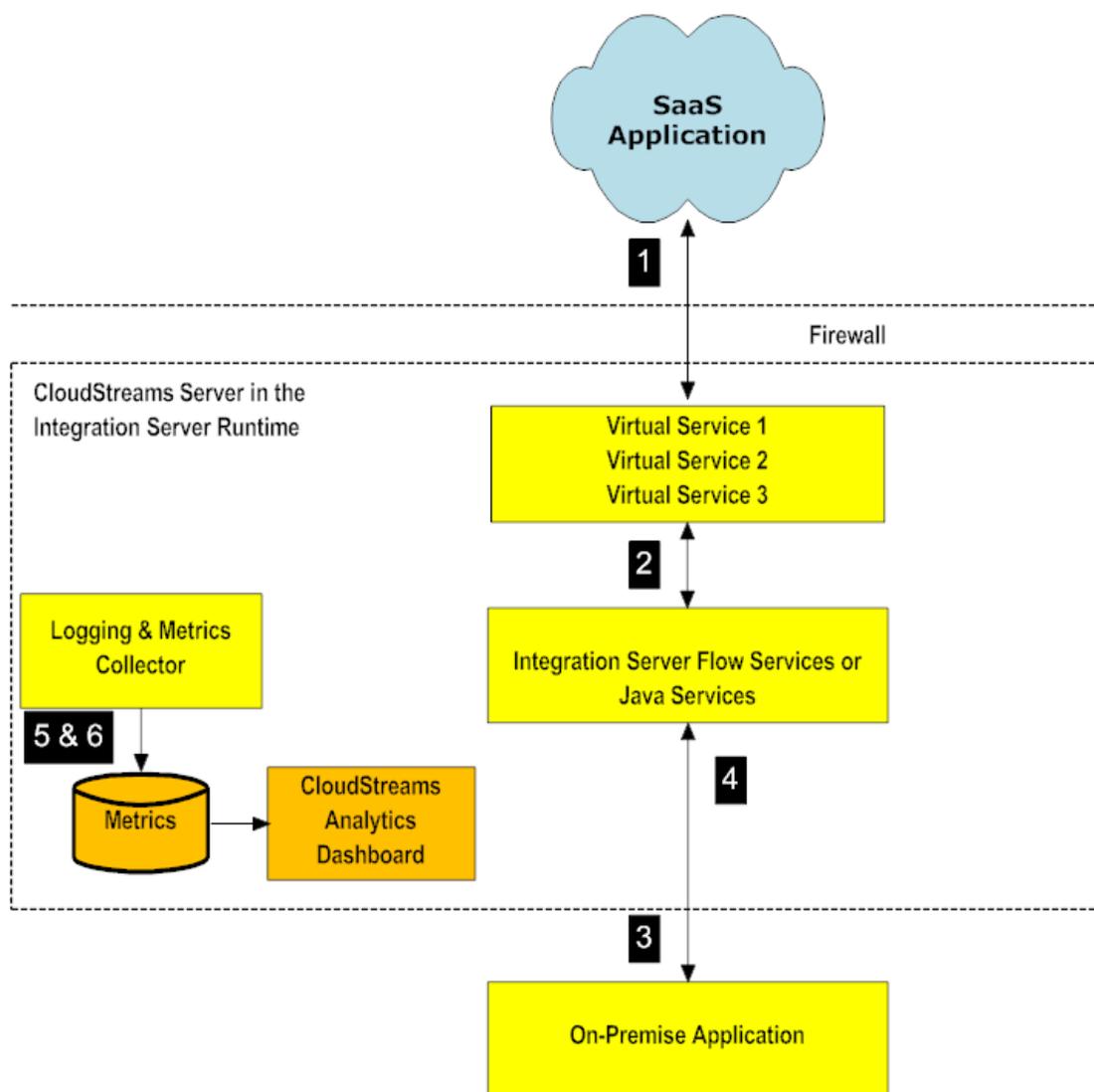
In the outbound run-time scenario, an on-premise application invokes a SaaS application through CloudStreams as follows:



Step	Description
1	The on-premise application invokes a cloud connector service that you defined to perform the desired SOAP operation (such as querying an account) or to access the desired REST resource.
2	The cloud connector service obtains a connection from the Integration Server connection pool.
3	The cloud connector service then invokes a connector virtual service.
4	The cloud connector service sends the request to the native service in the SaaS application.
5	The SaaS application processes the request for the native service and sends a response to the connector virtual service.
6	The CloudStreams server collects run-time metrics and events, and publishes them to the CloudStreams Analytics facility, which is powered by Software AG MashZone. You can use the CloudStreams Analytics dashboard to monitor the metrics.
7	The connector virtual service logs the request/response transaction and its payload.

Inbound Architecture

In the inbound run-time scenario, a SaaS application invokes an on-premise application through CloudStreams as follows:



Step	Description
1	A SaaS application invokes a virtual service that has been deployed to CloudStreams. The virtual service's policy performs security checks and any other user-defined processing before sending the request to the on-premise application.
2	The virtual service invokes a service endpoint (an Integration Server flow service or a Java service) in the on-premise application.
3	The endpoint service then invokes the on-premise application service(s).

Step	Description
4	The on-premise application processes the request against the native service and sends a response back to the SaaS application.
5	The CloudStreams server collects run-time metrics and events, and publishes them to the CloudStreams Analytics facility, which is powered by Software AG MashZone. You can use the CloudStreams Analytics dashboard to monitor the metrics.
6	The virtual service logs the request/response transaction and its payload.

The CloudStreams Connectors

CloudStreams provides pre-configured CloudStreams connectors. A *CloudStreams connector* contains SaaS provider-specific information that enables you to connect to a particular SaaS provider. You install a CloudStreams connector on Integration Server as an Integration Server package.

Each connector contains a *CloudStreams Connector Bundle*, which contains all the assets to enable the CloudStreams runtime to connect with the provider's back-end, and to perform operations on the back-end. One of these assets is the *Cloud Connector Descriptor*, which describes:

- A CloudStreams metadata handler (SOAP or REST), which creates the data model representing the connector for a particular provider.
- Meta information that enables you to create the cloud connector's connections.
- The connector's SOAP operations or REST resources.

Custom Cloud Connectors

You can use the CloudStreams Development plug-in to define cloud connectors for other SaaS applications. Then, you configure a custom connector the same way you configure a CloudStreams connector.

Configuring a CloudStreams Connector

You configure a CloudStreams connector by creating:

- A cloud connection to connect to the SaaS provider.
- Cloud connector services.
- Virtual services.
- Connector virtual services, if the default one does not suit your needs.

- Policies that provide governance capabilities for the virtual services and connector virtual services.

These tasks are summarized below.

Cloud Connections

A CloudStreams connector uses a *cloud connection* to connect to a SaaS application at run time.

Integration Server includes a connection management service that dynamically manages connections and connection pools based on the configuration settings you specify for the connection. A connection pool is a collection of connections with the same set of attributes. Integration Server maintains connection pools in memory. Connection pools improve performance by enabling cloud connector services to reuse open connections instead of opening new connections.

You create and configure connections using Integration Server Administrator. The procedures for doing this appear in the documentation specific to your CloudStreams connector (for example, the *webMethods CloudStreams Provider for Salesforce.com Installation and User's Guide*).

Cloud Connector Services

A *cloud connector service* is an Integration Server service that CloudStreams uses to integrate an "on-premise" application with a SaaS application.

The number of cloud connector services you should create depends on the following:

- For a SOAP-based provider, you should create at least one cloud connector service for each desired operation defined in the provider's WSDL. For example, for a query operation you might create a cloud connector service to query accounts and another to query contacts.
- For a REST-based provider, you should create at least one cloud connector service for each desired REST resource on which the connector can operate.

When you configure a cloud connector service, you specify the following information:

- The cloud connection you created.
- A SOAP operation or a REST resource, which is defined in the cloud connector descriptor.
- A *connector virtual service*, which sends service requests from CloudStreams to the SaaS provider, handles the provider's responses, and logs the request/responses (discussed below).
- The business object associated with the operation (for SOAP-based providers).
- The headers to include in the service's input pipeline.

- The input/output signature that determines how the user interacts with the service.
- Optional parameters to include in the input/output signature, such as variables to be replaced at run time with a user's input.

At run time, the cloud connector service constructs and maps a SOAP or REST request from the service's input pipeline to an appropriate message builder. When the SaaS application provider sends a response, the cloud connector service processes and maps the response and populates the output pipeline.

You create and configure cloud connector services using the Service Development plug-in provided in Software AG Designer. The procedures for doing this appear in the documentation specific to your CloudStreams connector (for example, the *webMethods CloudStreams Provider for Salesforce.com Installation and User's Guide*).

Virtual Services and Connector Virtual Services

A *virtual service* is a service that runs on a CloudStreams server and acts as the consumer-facing proxy for a native service that runs in a SaaS application or in an on-premise application. Service requests go to a virtual service for processing rather than directly to the service provider. You should create a virtual service for each SaaS service you want to expose to consumers.

There are two types of virtual services: *virtual services* and *connector virtual services*.

- In the inbound processing scenario, when a SaaS application sends a service request, a virtual service's policy performs security checks and any other user-defined processing before sending the request to the on-premise application.
- In the outbound processing scenario, when an on-premise application sends a service request, a special kind of virtual service is used, called a connector virtual service. A *connector virtual service* handles the SaaS provider's responses and logs the requests/responses and their payloads.

CloudStreams provides a default connector virtual service for each metadata handler (one for the SOAP handler and one for the REST handler). You cannot modify these default services, which are located in the WmCloudStreams package. Alternatively, you can create additional connector virtual services with custom policies.

You create both types of virtual services using the CloudStreams Development plug-in provided in Designer, and then deploy them to a CloudStreams server.

Both types of virtual services can have user-defined *policies*, which provide governance capabilities for the services; policies are discussed later in this overview.

The Processing Steps of Virtual Services

When you create a virtual service, you can configure the following processing steps:

- The service's "**In Sequence**" step, which you configure to manipulate the request messages. This step can include the following sub-steps:

- The **"Entry Step"** (provided by default), which specifies the protocol (HTTP or HTTPS) of the requests that the virtual service will accept. For REST-based virtual services, you also specify the REST resource that the service will access and the HTTP methods (GET, POST, PUT and DELETE) that the virtual service should be allowed to perform on the REST resource.
- The **"Transform" step** (optional), which performs an XSLT message transformation on the request message before the virtual service submits it to the native service.
- The **"Invoke IS Service" step** (optional), which preprocesses the request message before the virtual service submits it to the native service.
- The **"Routing Rule" step** (provided by default), which specifies how the virtual service will route the requests to the native service endpoint(s). For HTTP or HTTPS requests, you have four routing choices:
 - "Straight Through" routing (to route requests directly to the native service endpoint).
 - "Context-Based" routing (to route specific types of messages to specific endpoints according to context-based routing rules).
 - "Content-Based" routing (to route specific types of messages to specific endpoints based on specific values that appear in the request message).
 - "Load Balancing" routing (to distribute requests among multiple endpoints).
- The service's **"Out Sequence" step**, which you configure to manipulate the response messages. This step can include the following sub-steps:
 - The **"Transform" step** (optional), which specifies how the response message from the native service provider is to be transformed before the virtual service returns it to the consuming application.
 - The **"Invoke IS Service" step** (optional), which preprocesses the request message before the virtual service returns it to the consuming application.
- The service's **"Error Sequence" step** (provided by default). CloudStreams returns a default fault response to the consuming application, which you can customize with context variables. This fault response is used for faults returned by the native service provider as well as faults returned by internal CloudStreams exceptions, such as policy violation errors, connection timeouts, etc. You can choose whether or not to send the native service provider's service fault content, or just send the message. In addition, you can invoke IS services to pre-process or post-process the error messages.

The Processing Steps of Connector Virtual Services

A connector virtual service can contain similar processing steps as a virtual service.

Policies for Virtual Services and Connector Virtual Services

Policies provide run-time governance capabilities for virtual services and connector virtual services.

Policies for Virtual Services

You should create policies that apply to one or more virtual services. A *policy* is a sequence of actions that is carried out by CloudStreams when a consumer requests a particular service through CloudStreams. A policy for a virtual service can include the following kinds of actions:

■ **WS-SecurityPolicy 1.2 actions:**

There are two kinds of WS-SecurityPolicy 1.2 actions:

- Authentication actions, to verify that the requests for virtual services contain a specified WS-SecurityPolicy element. You can authenticate consumers by their WSS X.509 certificates, WSS username tokens, or WSS SAML tokens.
- XML security actions, to provide confidentiality (through encryption) and integrity (through signatures) for request and response messages.

■ **Monitoring actions:**

- The "Monitor Service Performance" action, which monitors a user-specified set of run-time performance conditions for a virtual service, and sends alerts to a specified destination when these conditions are violated
- The "Monitor Service Level Agreement" action, which provides the same functionality as "Monitor Service Performance", but this action is different because it enables you to monitor a virtual service's run-time performance for particular consumers. You configure this action to define a *Service Level Agreement* (SLA), which is set of conditions that defines the level of performance that a specified consumer should expect from a service.
- The "Throttling Traffic Optimization" action, which limits the number of service invocations allowed during a specified time interval.

■ **Additional actions:**

- "Identify Consumer", which you use in conjunction with an authentication action. Alternatively, this action can be used alone to identify consumers only by host name or IP address.
- "Require HTTP Basic Authentication", which uses HTTP Basic authentication to verify the consumer's authentication credentials contained in the request's Authorization header against the Integration Server's user account. This action supports WS-SecurityPolicy 1.2.

- "Authorize User", which authorizes consumers against a list of users and/or a list of groups registered in the Integration Server. You use this action in conjunction with an authentication action.
- "Log Invocation", which logs request/response payloads.
- "Validate Schema", which validates all XML request and/or response messages against an XML schema referenced in the WSDL.

Policies for Connector Virtual Services

Each default connector virtual service has a default policy, which logs the requests/responses and their payloads to a database. You cannot modify the default connector virtual services or their policies. However, you can create additional connector virtual services with custom policies.

If you create a connector virtual service with a custom policy, you can include the actions from the "Monitoring" and "Additional" categories (except for "Validate Schema"); the "WS-SecurityPolicy 1.2" actions are not needed. For example, you might want to create a custom policy that monitors run-time performance, customizes how the service invocations are logged, or optimizes server traffic.

The CloudStreams Analytics Dashboard

CloudStreams can publish key performance indicator (KPI) metrics and run-time events to the CloudStreams Analytics dashboard. The dashboard enables you to view and analyze the KPI metrics and run-time events of your CloudStreams connectors, cloud connector services and consumers.

The dashboard is powered by a MashApp called CloudStreams Analytics, which is powered by Software AG MashZone. CloudStreams Analytics uses data feeds from Software AG MashZone and uses the Integration Server's JDBC connection pool to publish this data at regular intervals.

The dashboard displays the following analytic views:

View	Filtered by...
Cloud Connector Performance Summary (performance and availability)	N/A; shows all connectors.
Cloud Connector API Usage (total requests and data volume)	Provider and time period.
Cloud Connector Performance (performance trends)	Provider and time period.

View	Filtered by...
Cloud Connector Error (error trends by provider and consumer)	Provider and time period.
Cloud Connector Service Level Agreement (SLA) violations and alerts, shown by consumer and provider.	Provider and time period.
Service (performance)	Provider, connector and time period.
Transaction (event results of request transactions)	Provider, connector, service, event type and time period.

The types of run-time events that can be published are as follows:

Event Type	Description
Lifecycle	A Lifecycle event occurs each time the Integration Server is started/shutdown/restarted or the WmCloudStreams package is unloaded/reloaded in a running Integration Server.
Transaction	A Transaction event occurs each time a cloud connector service is invoked (successfully or unsuccessfully).
Error	An Error event occurs each time an invocation of a cloud connector service or an inbound virtual service results in an error.
Policy Violation	A Policy Violation event occurs each time an invocation of a cloud connector service violates a policy that was set for its associated virtual service.
Monitoring	CloudStreams publishes key performance indicator (KPI) metrics, such as the average response time, fault count and availability of all connectors and connector services.

Getting Started with CloudStreams

The high-level steps for configuring a cloud connector are as follows:

Step	For procedures, see...
<p>1. Using the Software AG Installer, install the webMethods CloudStreams Server and the Software AG MashZone. The Software AG MashZone is required for using CloudStreams Analytics.</p> <p>Also, download the CloudStreams Provider (such as the Salesforce.com Provider) from the Software AG TECHcommunity website.</p> <p>Note: From the 9.7 release, CloudStreams providers and provider user guides are available for download from the TECHcommunity website. The provider installation guide is available in the webMethods section of the Software AG Documentation website.</p>	<ul style="list-style-type: none"> ■ The document <i>Installing webMethods and Intelligent Business Operations Products</i>. ■ The documentation specific to your CloudStreams provider (for example, <i>webMethods CloudStreams Provider for Salesforce.com Installation and User's Guide</i>).
<p>2. Create a cloud connection and configure the session management parameters, using the Integration Server Administrator.</p>	<p>The documentation specific to your CloudStreams provider</p>
<p>3. Create a cloud connector service for each desired operation defined in the cloud provider's WSDL, or for each desired REST resource, using the Software AG Designer's Service Development plug-in.</p>	<p>The documentation specific to your CloudStreams provider</p>
<p>4. Set the CloudStreams configuration options to specify things such as the keystore, truststore and ports to use, the database to use for publishing performance metrics and events, the consumer applications that can access CloudStreams, and more.</p>	<p>"CloudStreams Configuration Options" on page 27</p>
<p>5. Define a <code>CloudStreamsserver target</code>, which specifies an instance of a CloudStreams server to which you will deploy your virtual services and any optional custom connector virtual services. You do this using Software AG Designer. You can define one or more server targets.</p>	<p>"Virtual Services" on page 55</p>

Step	For procedures, see...
<p>6. Create a CloudStreams Governance project in which you will create the virtual services, connector virtual services, and their policies. You create projects in your local file system, using Designer's CloudStreams Development plug-in. You can create one or more projects.</p>	<p>"Virtual Services" on page 55</p>
<p>7. In the CloudStreams Governance project, create virtual services and connector virtual services. You do this using Designer's CloudStreams Development plug-in. You will:</p> <ul style="list-style-type: none">■ Create virtual services to handle the inbound requests.■ Optionally override the default connector virtual services with custom ones to handle outbound requests. CloudStreams provides a default connector virtual service for each metadata handler (which you cannot modify), but you can create additional connector virtual services with custom policies, if desired.	<p>"Virtual Services" on page 55</p>
<p>8. In the CloudStreams Governance project, create policies for the virtual services, using Designer's CloudStreams Development plug-in.</p> <ul style="list-style-type: none">■ For the virtual services, you create policies that include WS-SecurityPolicy actions (and other actions) provided by CloudStreams.■ For the connector virtual services, CloudStreams provides a default policy that logs all outbound transactions. You cannot modify these policies, but you can create additional connector virtual services with custom policies that include the monitoring, logging, and schema validation actions.	<p>"Policies" on page 167</p>
<p>9. Deploy the virtual services and any custom connector virtual services to a package in a CloudStreams server target.</p>	<p>"Deploying Virtual Services and Connector Virtual Services" on page 199</p>

Step	For procedures, see...
10. Configure CloudStreams Analytics to monitor performance metrics and run-time events.	" CloudStreams Analytics" on page 207
Note: If you want to create a custom cloud connector, see "Creating Custom Cloud Connectors" on page 231 .	

2 CloudStreams Configuration Options

- Overview 28
- The Administration Options 28
- The Providers Options 47
- The Streaming Options 48

Overview

Set the CloudStreams configuration options to specify things such as the keystore, truststore and ports to use, the database to use for publishing run-time metrics and events, the consumer applications that are allowed to access CloudStreams, and more.

The Administration Options

In Integration Server Administrator, you can set the following options under **Solutions > CloudStreams > Administration**:

- "Setting the General Options" on page 28.
- "Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts" on page 31.
- "Setting the Database Options for Publishing Performance Metrics and Events" on page 33.
- "The Virtual Services Option" on page 36.
- "Setting the STS Options" on page 36.
- "Setting the Service Fault Configuration Options" on page 40.
- "Setting the Consumers Options" on page 42.
- "Setting the OAuth Tokens Options" on page 44.

Setting the General Options

Use these options to manage the inbound service requests sent by consumers to CloudStreams. To manage the *outbound* requests that CloudStreams sends to the cloud provider, set the options for your cloud connections in your webMethods CloudStreams cloud connector. For details, see the section *Creating Cloud Connections* in the documentation for your webMethods CloudStreams cloud connector.

To set the CloudStreams > Administration > General options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Administration > General**.
2. Click **Edit**, complete the following fields and click **Save**.

Option	Description
Target Name	The CloudStreams target name. This name is included in the events that are sent to the database for metrics reporting purposes.
Load Balancer URL (HTTP)	<p>The primary HTTP load balancer URL and port number to use.</p> <p>For the URL, you can specify either the IP address or host name of the load balancer with the port number, as follows:</p> <p><code>http://IP-address :portnumber</code></p> <p>or</p> <p><code>http://hostname :portnumber</code></p> <p>If specified, all virtual services hosted in CloudStreams will use this value.</p>
Load Balancer URL (HTTPS)	<p>The primary HTTPS load balancer URL and port number to use.</p> <p>For the URL, you can specify either the IP address or host name of the load balancer with the port number, as follows:</p> <p><code>https://IP-address :portnumber</code></p> <p>or</p> <p><code>https://hostname :portnumber</code></p> <p>If specified, all virtual services hosted in CloudStreams (and exposed on HTTPS) will use this value.</p>
IS Keystore Name	The Integration Server keystore that CloudStreams should use. This field lists all available Integration Server keystores. If there are no configured Integration Server keystores, the list will be empty. To configure Integration Server keystores, see the section <i>Securing Communications with the Server</i> in the document <i>webMethods Integration Server Administrator's Guide</i> .
Alias (signing)	The signing alias. This alias is the value that is used to sign the outgoing response from CloudStreams to the original consumer. It is auto-populated based on the keystore selected from the IS Keystore Name above. This field will list

Option	Description
IS Truststore Name	<p>all the aliases available in the chosen keystore. If there are no configured keystores, this field will be empty.</p> <p>Specify the same truststore that is specified in Integration Server (as described in the section <i>Securing Communications with the Server</i> in the document <i>webMethods Integration Server Administrator's Guide</i>).</p> <p>This truststore will be used in the following cases:</p> <ul style="list-style-type: none"> ■ If you select the Secure Connection option for a CloudStreams server as described in "Defining and Managing a CloudStreams Server Target" on page 59. The truststore you specify in the IS Truststore Name field must contain the required SSL certificates. ■ This truststore is also used by the CloudStreams SMTP e-mail server if you select the TLS Enabled option while "Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts" on page 31. The truststore you specify in the IS Truststore Name field must include a certificate in the e-mail server's certificate chain or the certificate authority of the e-mail server. ■ This truststore is also used when making a connection to a third-party OAuth 2.0-based service provider to refresh the access token (as described in "Setting the OAuth Tokens Options" on page 44).
HTTP Ports Configuration	<p>Select one or more HTTP ports on which CloudStreams and the deployed virtual services will be available.</p> <p>CloudStreams is always available on the primary HTTP port. The primary HTTP port is the port specified on the Integration Server's Security > Ports page. You can specify additional HTTP and HTTPS ports here. To configure ports for Integration Server, see <i>webMethods Integration Server Administrator's Guide</i>.</p> <div style="background-color: #f0f0f0; padding: 10px;"> <p>Important: you specify multiple ports, the port that is reported in the WSDL is the <i>non-primary port with the lowest number</i>. There is only one port reported in the WSDL retrieved through CloudStreams, but all the specified ports are usable.</p> <p>If you are using a webMethods Enterprise Gateway (EG) Registration port to process requests from external clients, the EG port will be displayed with the EG hostname prefixed to the port, in order to</p> </div>

Option	Description
	<p>distinguish it from the local internal server ports. The Enterprise Gateway Registration port must be specified for CloudStreams.</p>
HTTPS Ports Configuration	<p>Select one or more HTTPS ports on which CloudStreams and the deployed virtual services will be available.</p>
Connection Factory Wire Logging	<p>This option captures additional debug messages that show the request's HTTP content that is sent over the wire to the native provider. The messages also show the response content that was received. This can be useful when debugging connector service invocations, but there are two implications you should be aware of:</p> <ul style="list-style-type: none"> ■ The server.log will grow quickly as it captures this verbose content, and performance will be severely impacted. Use this option sparingly. ■ Since all "cleartext" HTTP content is captured (even the content sent over SSL), be careful to avoid security issues. In addition to all request and response message content, the transport headers are also included, including Authorization headers. <p>If you disable and then re-enable the Connection Factory Wire Logging option, in certain cases it may be necessary to disable and then re-enable any existing managed connection pools. To do this, in the Integration Server Administrator go to Solutions > CloudStreams > Providers, click your cloud provider's name, click the connector name, and click the Enabled column to disable/re-enable the connection pool.</p>

Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts

If you use the following policy actions to send e-mail messages, you must specify the SMTP mail server to be used by CloudStreams:

- A "Log Invocation" action that you configure to send request/response payloads as e-mail attachments.
- A Monitoring action that you configure to send monitoring alerts as e-mail messages ("Monitor Service Performance", "Monitor Service Level Agreement" or "Throttling Traffic Optimization").

To set the CloudStreams > Administration > Email options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Administration > Email**.
2. Click **Edit**, complete the following fields and click **Save**.

Option	Description
SMTP Host Name/IP Address	The address of the SMTP server to be used by CloudStreams for sending e-mail alerts.
Port	The port on which the SMTP server is listening.
User	The user name of the e-mail account used to log into your SMTP server.
Password	The password of the e-mail account used to log into your SMTP server.
From	<p>The e-mail address that will appear in the From field of the event e-mail.</p> <p>If this field is left blank, CloudStreams generates a default e-mail address composed of the configured target name and hostname of the Integration Server, as follows:</p> <p><i>Target-Name@hostname</i></p>
TLS Enabled	<p>Whether to use transport-layer security (TLS). Requirements:</p> <ul style="list-style-type: none"> ■ A keystore and a truststore must be configured in Integration Server (see the section <i>Securing Communications with the Server</i> in the document <i>webMethods Integration Server Administrator's Guide</i>). ■ An Integration Server keystore and truststore must be specified in the IS Keystore Name and IS Truststore Name fields in CloudStreams > Administration > General (see "Setting the General Options" on page 28). The truststore specified in IS Truststore Name must include a certificate in the e-mail server's certificate chain or the certificate authority of the e-mail server.
Test Recipient	To test your connection, enter in this field an e-mail address of the person who should receive the test e-mail, and click Test . If the test is successful, CloudStreams sends an e-mail to the address and displays a success message. Otherwise,

Option	Description
	CloudStreams displays an error message; reconfigure your e-mail settings and try again.

Setting the Database Options for Publishing Performance Metrics and Events

CloudStreams can publish data about key performance indicator (KPI) metrics and run-time events to the CloudStreams Analytics dashboard.

CloudStreams uses the Integration Server's JDBC connection pool to publish this data. The JDBC connection pool is specified in Integration Server Administrator's **Settings > JDBC Pools** page. You must define this connection pool for use by CloudStreams as described below.

To set the CloudStreams > Administration > Database options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Administration > Database**.
2. Click **Edit**, complete the following fields on the Database Configuration page, and click **Save**.

Option	Description
Connection Pool Alias	<p>The Integration Server connection pool alias that is defined in the Integration Server Administrator's Settings > JDBC Pools page, as described in "Creating a JDBC Connection Pool for CloudStreams" on page 215.</p> <p>If the Integration Server connection pool alias is not yet defined, click the link to define it.</p>
Database Publishing	<p>Click Enable to enable the database specified in the Connection Pool Alias to receive data.</p> <p>Note: By default, Database Publishing is disabled. If you want to use the CloudStreams Analytics feature (KPI metrics and run-time events), enable this option.</p>
Publish Performance Metrics	<p>Select this option to enable CloudStreams to publish key performance indicator (KPI) metrics to the database. For more information, see "The Key Performance Indicator (KPI) Metrics" on page 209.</p>

Option	Description
Publish Interval (minutes)	The interval at which CloudStreams should publish performance metrics to the database. Default: 60 minutes. For more information, see " The Intervals for Metric Publishing " on page 34, below.
Publish Events (Error, Lifecycle, Policy Violation)	Select the types of events to publish to the database: <ul style="list-style-type: none"> <li data-bbox="524 558 1349 659">■ Error: An Error event occurs each time an invocation of a cloud connector service or an inbound virtual service results in an error. <li data-bbox="524 680 1349 821">■ Lifecycle: A Lifecycle event occurs each time the Integration Server is started/shutdown/restarted or the WmCloudStreams package is unloaded/reloaded in a running Integration Server. <li data-bbox="524 842 1349 934">■ Policy Violation: A Policy Violation event occurs each time an invocation of a cloud connector service violates a policy that was set for its associated virtual service.

The Intervals for Metric Publishing

CloudStreams tracks performance metrics by intervals. The interval is a period of time you set in CloudStreams, during which metrics are collected for reporting to CloudStreams.

There are two cases in which you specify an interval when measuring performance:

- When you publish performance indicator (KPI) metrics and events to the CloudStreams Analytics dashboard. In this case, you set the interval in Integration Server Administrator, in **Solutions > CloudStreams > Administration > Database**.
- When monitoring a virtual service's run-time performance with the actions "Monitor Service Performance", "Monitor Service Level Agreement", or "Throttling Traffic Optimization". In this case, you set an interval in the action's "interval" parameter.

The rules of intervals are the same in both cases.

CloudStreams only tracks metrics for the current interval. At the end of the interval, CloudStreams aggregates the metrics and reports them to CloudStreams. CloudStreams resets its counters immediately for the next interval, even if the last interval's data has not yet been reported to the database (so there is no gap between intervals when metrics are not accumulated). CloudStreams does not calculate and aggregate metrics across intervals. If CloudStreams is shut down or the service is undeployed before the current interval expires, the performance data is discarded.

Examples of interval metric publishing

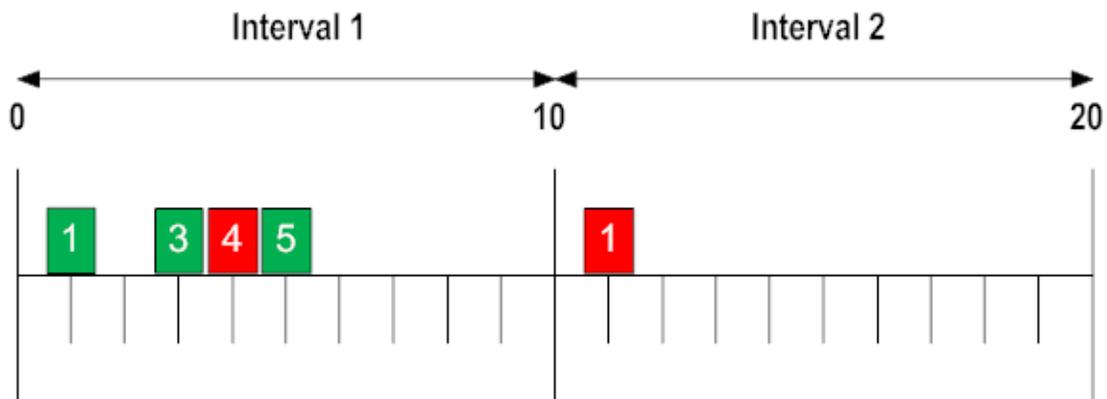
For example, suppose that the tracking interval is 10 minutes. One of the key performance indicator (KPI) metrics is "Availability", which reports the amount of time that a service was available during the current interval, shown as a percentage. The green boxes in the illustration below indicate successful requests and the red ones indicate unsuccessful requests.

A request is considered unsuccessful when a network fault occurs or when the native service is unavailable due to the following errors:

- `ConnectException`
- `MalformedURLException`
- `NoRouteToHostException`
- `ProtocolException`
- `SocketTimeoutException`
- `UnknownHostException`
- `UnknownServiceException`
- Web Service not available
- The service cannot be found

In the case of a normal application-level SOAP fault, CloudStreams considers the request to be successful. In the illustration below, in Interval 1 (0 - 10 minutes) a request failed at the 4 minute mark, followed by a successful request at the 5 minute mark. Therefore, CloudStreams considers the interval between 4 and 5 minutes to be service downtime (even though this may not be accurate). So in this case, for Interval 1 the availability is 9/10 (90%). In the case of Interval 2, only one request was sent, and it failed at the 1 minute mark. Therefore, CloudStreams considers the time between 1 minute to the end of the interval as service downtime. So the time between the start of Interval 2 (the 10 minute mark) to the failed request is service uptime (1 minute); the availability is 1/10 (10%) for Interval 2. At the end of the interval, CloudStreams resets the KPI metrics.

The "Availability" status of a service is transferred to the beginning of the next interval. When CloudStreams starts up, it is assumed that the services are available. If a service is unavailable at the end of interval 1, and no activity occurs for several intervals, the service's availability is 0 during the intervals when no activity occurred.



The Virtual Services Option

This option displays a list of:

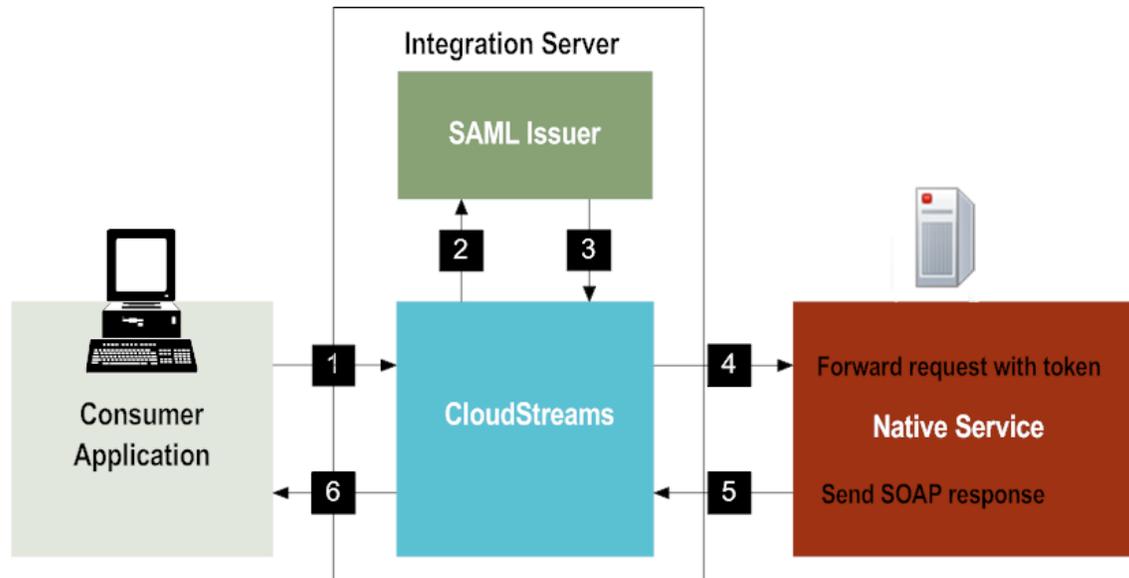
- The default connector virtual services (**WmCloudStreams.S SoapVS** and **WmCloudStreams.RestVS**) as well as any other user-defined connector virtual services in the WmCloudStreams package.
- All virtual services defined in the WmCloudStreams package that are currently deployed to a CloudStreams server target.

To view the list, select **Solutions > CloudStreams > Administration > Virtual Services** in Integration Server Administrator.

Setting the STS Options

CloudStreams can act as a Security Token Service (STS) client. The following illustration shows what happens at run time.

CloudStreams as an STS Client



Step	Description
1	The user's client sends a SOAP request with SAML authentication information to CloudStreams. Integration Server authenticates the incoming request.
2	<ul style="list-style-type: none"> ■ CloudStreams sends a WS-Trust RST to the STS to request a SAML v2 token. ■ CloudStreams sends the <code><OnBehalfOf></code> element that contains the authenticated user name to the STS.
3	The SAML Issuer sends the SAML v1/v2 assertion to CloudStreams.
4	<p>CloudStreams forwards the SOAP request (along with the SAML assertion) to the native service.</p> <p>CloudStreams also uses the IS keystore and signing alias you specified to sign the SAML token and the request body before sending the request to the native service.</p> <p>Also, if you have configured the predefined Java service <code>pub.cloudstreams.security.ws.AddSamlSenderVouchesToken</code> to add a timestamp in the outbound request, CloudStreams will sign the timestamp as well.</p>
5	The native service sends a SOAP response to CloudStreams.

Step	Description
6	CloudStreams sends the response to the user's client.

Use the following procedure to enable CloudStreams to act as a Security Token Service (STS) client.

To set the CloudStreams > Administration > STS options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Administration > STS**.
2. You can use Integration Server's default STS, **DefaultSTS**. To view its default parameters, click its name.
3. Alternatively, you can use a third-party STS that has been defined in the Integration Server (as described in the *Web Services Developer's Guide*, in the section *Securing Web Services Using Policies Based on WS-SecurityPolicy*). To use a third-party STS, click **Add new STS configuration** and complete the Add STS Configuration page as follows.

Option	Description
Name	A unique name for the STS being configured. If this value is changed after creating an STS, the previous STS configuration will be deleted and replaced with the new one.
Endpoint	The STS endpoint to which the WS-Trust request will be sent by CloudStreams to obtain the SAML token.
Token Type	The type of token that CloudStreams should request from the STS. Value can be SAML_11 or SAML_20.
	Note: Currently the default Integration Server STS only supports issuing SAML 2.0 Sender-vouches tokens.
WS-Trust Version	The version of WS-Trust that CloudStreams should use to send the RST to the SAML Issuer. Value can be VERSION_05_02 or VERSION_05_12.
Time-To-Live (TTL)	Indicates the time-to-live value in seconds that will be specified in the RST. If not specified, the default is 300 seconds (5 minutes).

Option	Description
Keystore	Select a keystore that has been configured in Integration Server. To configure a keystore in Integration Server, see the section <i>Securing Communications with the Server</i> in the document <i>webMethods Integration Server Administrator's Guide</i> .
Sign Request?/Signing Alias	Select Sign Request? if the STS requires a signed request, and specify the signing alias in the Signing Alias field.
Encrypt Request?/Encryption Alias	Select Encrypt Request? if the STS requires an encrypted request, and specify the encryption alias in the Encryption Alias field.
HTTP Basic Authorization Username/Password	If the STS requires authentication, enter the HTTP Basic Authentication username and password.
WS-Security Username Token/Username, Password, and Password Type	The WS-Security username token to send to the STS. Also specify the username's password and password type (None, Text or Digest).

- If you selected Integration Server's default STS (**DefaultSTS**), edit the default STS's configuration file to specify the keystore and alias so that the STS can sign the SAML assertion that it is issuing.

The configuration file is:

```
IntegrationServer_directory \instances\instance_name \config\security\saml
\esb_sts.xml
```

The contents of the file are shown below. Use the comments as a guide to configure this file for your system.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- This configuration file is used to configure the IntegrationServer token
issuer that generates the SAML Sender Vouches token for CloudStreams outbound
requests -->
<IDataXMLCoder version="1.0">
<record javaclass="com.wm.data.ISMemDataImpl">
<!-- IssuerName - will be used as the IssuerName for each SAML token issued by
this Service; the default value is ESB_STS -->
<value name="IssuerName">ESB_STS</value>
<!-- IssuerKeystoreAlias - specify an Integration Server Keystore Alias that
contains the private keys that can be used to sign the generated SAML
Assertion -->
<value name="IssuerKeystoreAlias">STS</value>
  <!-- IssuerKeyAlias - the name of the key alias within the
IssuerKeystoreAlias that points to the private key files -->
<value name="IssuerKeyAlias">sts</value>
  <!-- TimeToLiveSeconds - how long in seconds the generated token should be
valid? the default is 300 seconds (i.e. 5 minutes) from the time of token
```

```

creation -->
<number name="TimeToLiveSeconds" type="java.lang.Integer">300</number>
</record>
</IDataXMLCoder>

```

- Configure the desired virtual services so they can use the STS. To do this, write an IS service that includes the predefined Java service `pub.cloudstreams.security.ws.AddSamlSenderVouchesTokenand` and invoke this IS service in the "In Sequence" step of the virtual service. This service must reference an STS that is identified in CloudStreams. For details about the `AddSamlSenderVouchesTokenand` service, see ["Using the Security API in IS Services" on page 89](#).

Setting the Service Fault Configuration Options

Note: The service fault configuration options are intended only for virtual services, and not for Connector Virtual Services. For information about error handling for Connector Virtual Services, see ["The "Error Sequence" Step \(Connector Virtual Service, SOAP\)" on page 125](#) or ["The "Error Sequence" Step \(Connector Virtual Service, REST\)" on page 159](#).

You can use these options to configure global error responses for all virtual services.

Alternatively, you can configure error responses for virtual services individually, in the services' "Error Sequence" step, as described in ["The "Error Sequence" Step \(Connector Virtual Service, SOAP\)" on page 125](#) or ["The "Error Sequence" Step \(Connector Virtual Service, REST\)" on page 159](#).

The precedence of the error messaging instructions is as follows:

- If you create an "Error Sequence" step for a virtual service, the error messaging step takes precedence over any settings on the Service Fault Configuration page.
- If you do not create an "Error Sequence" step for a virtual service, the settings on the Service Fault Configuration page take precedence.

To set the CloudStreams > Administration > Service Fault Configuration options

- In Integration Server Administrator, select **Solutions > CloudStreams > Administration > Service Fault Configuration**.
- Click **Edit**, complete the following fields and click **Save**.

Option	Description
Default Fault Response	<p>When you select this option, CloudStreams returns the following fault response to the consuming application:</p> <pre> CloudStreams encountered an error:\$ERROR_MESSAGE while executing operation:\$OPERATION service:\$SERVICE at time:\$TIME on date:\$DATE. The client ip was: </pre>

Option	Description
	<p data-bbox="428 310 1192 373">\$CLIENT_IP. The current user:\$USER. The consumer application:\$CONSUMER_APPLICATION".</p> <p data-bbox="428 394 1214 426">This fault response is returned in both of the following cases:</p> <ul data-bbox="428 447 1292 961" style="list-style-type: none"> <li data-bbox="428 447 1292 846">■ When a fault is returned by the native service provider. In this case, the \$ERROR_MESSAGE variable in the fault response will contain the message produced by the provider's exception that caused the error. This is equivalent to the <code>getMessage</code> call on the Java Exception. This maps to the <code>faultString</code> element for SOAP 1.1 or the <code>Reason</code> element for SOAP 1.2 catch. For REST service calls, the message is returned inside an <code></Exception></code> tag. CloudStreams discards the native service provider's fault and does not return this content to the web service caller since it could be considered a security issue, especially if the native provider is returning a stack trace with its response. <li data-bbox="428 867 1292 961">■ When a fault is returned by internal CloudStreams exceptions (policy violation errors, cloud connection errors and cloud connector service errors). In this case, the \$ERROR_MESSAGE variable will contain the error message generated by CloudStreams. <p data-bbox="428 1077 1208 1171">The default fault response contains predefined fault handler variables (\$ERROR_MESSAGE, \$OPERATION, etc.), which are described in "The Fault Handler Variables" on page 87.</p> <p data-bbox="428 1192 1263 1287">You can customize the default fault response using the following substitution variables, where CloudStreams replaces the variable reference with the real content at run time:</p> <ul data-bbox="428 1308 1292 1507" style="list-style-type: none"> <li data-bbox="428 1308 1292 1381">■ The predefined context variables listed in "The Predefined Context Variables" on page 100. <li data-bbox="428 1402 1292 1507">■ Custom context variables that you declare using the CloudStreams API (see "The API for Context Variables" on page 105). <div data-bbox="428 1528 1318 1789" style="background-color: #f0f0f0; padding: 10px;"> <p data-bbox="444 1539 1292 1770">Note: If you want to reference a custom context variable that you have already defined in a context-based routing rule (as opposed to one you have declared using CloudStreams API for context variables), then you must add the prefix <code>\$mx</code> to the variable name in order to reference the variable. For example, if you defined the variable <code>TAXID</code>, you would reference it as <code>\$mx:TAXID</code>.</p> </div>

Option	Description
Send Native Provider Fault	<p>When you select this option, CloudStreams sends the native service provider's fault content, if available. CloudStreams will send whatever content it received from the native service provider.</p> <p>If you select this option, the Default Fault Response is ignored when a fault is returned by the native service provider. (Faults returned by internal CloudStreams exceptions will still be handled by the Default Fault Response option.)</p>

Note: Unlike with SOAP specifications, there is no agreed upon format to suggest an error condition for REST services (that is, there is no element nested in a `</soap:Fault>` element nested in a `</soap:Body>`). CloudStreams assumes that REST services will follow HTTP conventions and return responses with return codes in the 200-299 range when service calls are successful. This is the only way CloudStreams can determine that a native provider's response should be interpreted as a failure.

Setting the Consumers Options

Use this option to define the consumer applications that should be allowed to access CloudStreams.

On this page, specify the precise values for the consumer identifier(s) that you specified for the Identify Consumer action in the policies of your virtual services.

To set the CloudStreams > Administration > Consumers options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Administration > Consumers**.
2. Click **Add New Consumer**, complete the following fields and click **Save**.

Option	Description
Name	Assign a name for the consumer application.
Description	Enter a description for the consumer application.
IP Addresses	<p>Use this field when the Identify Consumer action is configured to identify consumer applications by IP address. Enter one or more IP addresses (or ranges of addresses) to identify consumers. IPv4 and IPv6 addresses are supported.</p> <p>If entering an IPv4 address, enter a 4-byte IP address. Note that:</p>

Option	Description
	<ul style="list-style-type: none"> <li data-bbox="524 323 1349 428">■ To specify an individual IP address, type the address in the From field. The Identify Consumer action will identify only those requests that originate from the specified IP address. <li data-bbox="524 449 1349 653">■ To specify a range of IP addresses, type the lowest IP address in the From field and the highest IP address in the To field. For example, the values 192.168.0.0 and 192.168.0.10 indicate that requests originating from any IP address that lies between the specified range will be identified by CloudStreams. <p data-bbox="524 674 1349 743">If entering an IPv6 address, use the 128-bit IPv6 format. For example: 1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0.</p> <p data-bbox="524 764 1349 833">To specify additional IP addresses, use the plus button to add more rows.</p>
Identification Token	<p data-bbox="524 873 1349 1010">Use this field when the Identify Consumer action is configured to identify consumer applications by an identification token. Specify one or more of the following tokens. To specify additional tokens, use the plus button to add more rows.</p> <ul style="list-style-type: none"> <li data-bbox="524 1031 1349 1167">■ Host Name: To identify consumers based on a specified host name, type the host name (for example, <code>pcmachine.ab.com</code>) in the Name field. CloudStreams will identify only those requests that originate from the specified host name. <li data-bbox="524 1188 1349 1461">■ HTTP Authorization Token : To authorize consumers based on the user name transmitted in an HTTP authorization user token, type the user name (for example, <code>testuser123</code>) in the Name field. The application asset will identify only the requests that contain the specified user name encoded and passed in the HTTP user token in the HTTP Authorization header. Authentication is handled by LDAP or another external authentication mechanism. <li data-bbox="524 1482 1349 1755">■ WS-Security Authentication Token: To authenticate consumers based on the user name transmitted in the SOAP or XML message header (HTTP body), type the user name (for example, <code>userwss</code>) in the Name field. CloudStreams will identify only the requests that contain the specified user name passed in the SOAP or XML message header. Authentication is handled by LDAP or another external authentication mechanism. <li data-bbox="524 1776 1349 1904">■ A custom identification token (XPath): To identify consumers based on the result of applying an XPath expression on the SOAP or XML message or request, enter the XPath expression in the Name field. For example,

Option	Description
	<p>typing <code>//*[local-name()='Envelope']/*[local-name()='Body']/*[local-name()='echoInt']/*[local-name()='echoIntInput='][.='2']</code> in the Name field will identify the requests that contain the XPath and the consumers.</p> <ul style="list-style-type: none"> ■ User ID: For connector virtual services only. Identifies consumers based on a list of users who are registered in the Integration Server on which CloudStreams is running.
X.509 Certificate	Use this field when the Identify Consumer action is configured to identify consumer applications by an X.509 v3 consumer certificate. Specify the path of a certificate file on the server.

Setting the OAuth Tokens Options

If you will be setting the **Authentication Type** field to **OAuth** on the Streaming Providers page (as described in ["Setting the Streaming Providers Options" on page 48](#)), you need to set the options here, on the OAuth Access Tokens page.

If OAuth will be used as an authentication mechanism, you must create an alias containing OAuth details. For example, when a Streaming Provider is created that will use OAuth, the provider details include an OAuth alias. When OAuth is chosen on the Streaming Providers page, an alias must be selected from a drop-down list of OAuth Token aliases that are currently defined. At run time, the OAuth alias is used to lookup parameters, such as Access Token, that are used to establish the connection with the provider.

OAuth is a scheme to limit the need for user credentials when operating on behalf of a resource owner. Instead of using the owner's credentials at run time, an Access Token is used. Prior to run time, the owner and user engage in a "dance" with a Resource Server and an Authorization Server to obtain an Access Token that will grant the user limited access to resources controlled by the owner. The precise steps in the dance vary with OAuth version, provider (Salesforce versus Twitter) and OAuth grant scheme ("two-legged dance" versus "three-legged dance"). Given this variation and the complexity of this process, CloudStreams assumes that the proper OAuth fields are obtained, and stores the tokens and other properties for use at run time.

Some of the information that is stored is considered to be sensitive and is stored securely, just as a password would be.

Currently, two OAuth versions are supported, v1.0a and v2.0. The parameters that are stored with an alias are different in the two versions, though there is also significant overlap.

To set the CloudStreams > Administration > OAuth Tokens options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Administration > OAuth Tokens**.
2. Click **Add New Token**, complete the following fields, and click **Save**.

Option	Description
Name	A unique string to identify a collection of alias token parameters.
Provider	Identifies the OAuth provider (such as Salesforce or Twitter).
OAuth Version	Select the OAuth version, v1.0a or v2.0.
Consumer ID	The 'Consumer Key' issued by the Service Provider and used by the consumer to identify itself to the Service Provider. In OAuth 2.0, this is a 'client identifier' issued to the client to identify itself to the authorization server. Also referred to as the "Client ID".
Consumer Secret	A secret used by the Consumer to establish ownership of the 'Consumer Key'. In OAuth 2.0, this is a secret matching to the 'client identifier'. Also referred to as the "Client Secret".
Access Token	<p>The token used for authentication, instead of other credentials. This token is issued by the Authorization Server. The process for obtaining a token varies with version, vendor and OAuth grant type.</p> <p>For OAuth 2.0, CloudStreams will use the keystore and truststore specified in the CloudStreams > Administration > General tab when making a connection to the third-party OAuth 2.0-based service provider to refresh the access token. The refresh URL is almost always SSL-based, so the truststore specified in CloudStreams will be used during the SSL handshake to verify the provider certificate. If the provider requires two-way SSL authentication, then the keystore specified in CloudStreams will be used to provide the client certificates. For details about the keystore and truststore specified in CloudStreams, see "Setting the General Options" on page 28.</p> <p>If a truststore is <i>not</i> specified in the CloudStreams > Administration > General tab, CloudStreams will use the default truststore for the jvm (cacerts). This might be a problem if you have defined the default truststore to only contain specific certificates, because if that alias is defined, the well-known</p>

Option	Description
	certificates will not be included when defining the trust. In this case, make certain the truststore alias keystore includes the certs necessary to trust the URLs defined in this field too.
Access Token Secret	Issued with the v1.0a Access Token only. A secret used by the Consumer to establish ownership of a given 'Access Token'. Secrets are used to sign the requests.
Instance URL	This is an optional field and is used to specify a runtime host, if applicable. This may be required in some back ends like Salesforce.
Refresh Access Token	<p>Option to refresh the 'Access Token'. OAuth 2.0 access tokens typically have a very short lifetime. When an access token expires, the OAuth profile does not automatically refresh the expired access token. Select this option if you want an expired access token to be refreshed automatically. If you select this option, you must also specify Refresh Token and Refresh URL.</p> <p>The access token is refreshed whenever the session expires. Session expiration is handled according to the setting of the Session Management property in your connection. Note that if Session Management is set to "none", then you must manually modify the access token in the OAuth alias. (The Refresh Access Token option will not be applicable in this case.)</p> <p>If you want to refresh the 'Access Token' automatically, set Session Management to either 'fixed' or 'idle'. The Timeout value should be based on the backend settings.</p>
Refresh Token	<p>Issued with the OAuth v2.0 access token only. Required only if Refresh Access Token is enabled.</p> <p>A token used by the client to obtain a new access token without having to involve the resource owner.</p>
Refresh URL	<p>Available only if Refresh Access Token is selected.</p> <p>The provider specific URL to refresh an 'Access Token'. This is required when 'Refresh Access Token' is enabled.</p>
Refresh URL Request	<p>Options for sending the parameters, for example, refresh_token, grant_type, client_id, and client_secret required by an access token refresh (HTTP POST) request.</p> <p>The available options are:</p>

Option	Description
	<ul style="list-style-type: none"> ■ URL Query String - The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the URL of the POST request. Example: <pre>www.examplebackend.com/o/oauth2/token?grant_type=refresh_token&client_id=842428530070-pubfebfqkkgj6t54m4ns6&client_secret=4adQT95cAtUxWINbDxGP9SJ4&refresh_token=1%2Fn072P4BXpuNObjCLUtiZTc4fMH6YersmxBIv8QN3bhw</pre> ■ Body Query String - The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the body of the POST request. Example: <pre>POST /o/oauth2/token HTTP/1.1 Host: accounts.backend.com Content-length: 163 content-type: application/x-www-form-urlencoded client_secret4adQT95cAtUxWINbDxGP9SJ4&grant_type=refresh_token&refresh_token=1%2Fn072P4BXpuNObjCLUtiZTc4fMH6YersmxBIv8QN3bhw&client_id=407408718192</pre> ■ Custom ESB Service - Use this option if the back end requires some refresh requests in a custom format, for example, requests which need more parameters than the ones specified by OAuth v2.0, or the back end uses some custom way of organizing parameters, or expects some other HTTP method request (other than POST). If you select this option, you must specify the name of your custom service in the 'Service Name' field.
Service Name	<p>User implemented service for refreshing the 'Access Token'. This is required when the Refresh URL Request is specified as Custom ESB Service. This service should strictly conform to the following specification:</p> <pre>-wm.cloudstreams.service.common.lookup.specs: oauthTokenRefreshServiceSpec</pre>

The Providers Options

Use these options to:

- Enable/disable/delete/configure CloudStreams cloud connectors.

- Create/enable/disable/delete/view/copy managed cloud connections.

For details, see the documentation specific to your webMethods CloudStreams cloud connector (for example, *webMethods CloudStreams Provider for Salesforce.com Installation and User's Guide*).

The Streaming Options

In Integration Server Administrator, you can set the following options under **Solutions > CloudStreams > Streaming**:

- ["Setting the Streaming Providers Options" on page 48](#)
- ["Setting the Streaming Subscribers Options" on page 50](#)

Setting the Streaming Providers Options

To set the **CloudStreams > Streaming > Providers** options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Streaming > Providers**.
2. Click **Add New Provider**, complete the following fields and click **Save**.

Option	Description
Provider Name	Specify a unique name for this streaming service provider. This is a required field and is used by the Subscriber configuration to refer to it.
Client Type	<p>The type of client that will be used by CloudStreams to provide the streaming functionality. There are two types of clients:</p> <ul style="list-style-type: none"> ■ Comet: This is a client implementation that uses the Bayeux HTTP protocol (using the CometD library) to communicate with the streaming service. This client should be used if the streaming provider explicitly indicates that they support the Bayeux protocol for their streaming functionality. For example, Salesforce uses this mechanism for their streaming functionality and hence Salesforce integrations must use this client type. ■ HTTP: An HTTP client type is used for HTTP-based streaming service providers that use a long-lived HTTP connection to send data periodically to the client connected to them. For example, Twitter uses this mechanism to send updates

Option	Description
	made to their streaming API endpoints (Public, User and Site streams).
API Version	Optional. The version of the streaming API supported by the provider service.
Streaming API Endpoint	Points to the endpoint of the streaming provider. For example, when configuring a Salesforce streaming endpoint, you must specify the URL that corresponds to the streaming API, such as: https://na9-api.salesforce.com/cometd/25.0 Do not specify the actual subscription channel/topic name here as part of the URL because that is subscriber configuration-specific.
Notify Meta Channel Events	Optional. Indicates whether any meta-channel (connect, disconnect, handshake, etc.) events must be notified to the subscriber. If you choose this option, then the events are delivered to the destination configured in the subscriber (see "Setting the Streaming Subscribers Options" on page 50).
Enable Debug	Optional. Indicates whether any connection-related debugging messages must be logged to the server log.
Connection Timeout (ms)	Specify the streaming connection timeout value, in milliseconds. If unspecified, this value defaults to a system default.
Read Timeout (ms)	Specify the read timeout value after a connection has been established with the provider. This indicates the maximum time before which a data packet must be read from the endpoint before a timeout occurs.
Truststore Alias	Specify the IS truststore alias to use if the endpoint is SSL-based and the provider certificate must be validated as part of the SSL handshake. By default, if no alias is configured, then all server certificates are trusted during SSL handshake. The alias must be configured in Integration Server Administrator. To configure Integration Server truststores, see the section <i>Securing Communications with the Server</i> in the document <i>webMethods Integration Server Administrator's Guide</i> .
Keystore Alias	Specify the IS keystore alias to use if the endpoint is SSL-based and the CloudStreams certificate must be provided to the

Option	Description
	server endpoint as part of the SSL handshake. By default, if no alias is configured, then no client certificate is sent during SSL handshake. The alias must be configured in Integration Server Administrator. To configure Integration Server keystores, see the section <i>Securing Communications with the Server</i> in the document <i>webMethods Integration Server Administrator's Guide</i> .
Validate Certificate	If set to true, the client will validate the provider's certificate for the SSL handshake. Default: False.
Authentication Type	<p data-bbox="524 665 1279 800">Specifies the authentication information that must be sent to the streaming provider when a subscription to a topic or channel is defined. Choose one of the following ways to provide the authentication information:</p> <ul data-bbox="524 825 1360 1680" style="list-style-type: none"> <li data-bbox="524 825 1360 926">■ Basic: Basic refers to HTTP Basic Authentication. This option can be used if the provider requires or supports HTTP Basic authentication using a username and password. <li data-bbox="524 947 1360 1152">■ OAuth 1.0a and OAuth 2.0: Select an OAuth alias that was defined on the OAuth Access Tokens page (see "Setting the OAuth Tokens Options" on page 44). At run time, CloudStreams will use the OAuth configuration information to retrieve the access token to be sent to the provider in the HTTP request. <li data-bbox="524 1173 1360 1680">■ ESB Callback: This option can be used when the above two means of authentication are not sufficient. For example, if the provider does not support Basic authentication or OAuth tokens, but instead requires another form of authentication information as part of the <code>Authorization</code> header, then you can provide another form of authentication in a user-defined ESB service. Specify the service name in this field in the following form: <code>folder1.folder2:serviceName</code>. The only requirement for this ESB service is that it should specify the value for the <code>Authorization</code> header in the <code>authorization.header</code> pipeline variable. CloudStreams will use the value present in this pipeline variable to send it as part of the <code>Authorization</code> header value. You should <i>not</i> include the header name (<code>Authorization</code>) in the value set in this variable.

Setting the Streaming Subscribers Options

CloudStreams supports the following kinds of streaming client implementations:

- Client implementations based on the Comet library, which works with streaming services using the Bayeux protocol (for example, Salesforce).
- An HTTP client type, which is used for HTTP-based streaming service providers that use a long-lived HTTP connection to send data periodically to the client connected to them. For example, Twitter uses this mechanism to send updates made to their streaming API endpoints (Public, User and Site streams).

To set the CloudStreams > Streaming > Subscribers options

1. In Integration Server Administrator, select **Solutions > CloudStreams > Streaming > Subscribers**.
2. Click **Add New Subscriber**, complete the following fields and click **Save**.

Option	Description
Enable Subscriber	Enables or disables the subscriber configuration. If the subscriber configuration is disabled, then no streaming notifications will be received by CloudStreams until it is enabled again.
Subscriber Name	A name that uniquely identifies a subscriber.
Provider	One of the streaming providers previously configured in "Setting the Streaming Providers Options" on page 48 . The provider names are listed in a drop-down field, and each subscriber must choose the provider configuration to use in order for the streaming subscription to work.
Channel Endpoint	<p>The topic/channel name or the endpoint address for the subscription request, depending on the streaming provider's client type, as follows:</p> <ul style="list-style-type: none"> ■ If the streaming provider's client type is Comet, specify the topic or channel name for the subscription request. This is usually of the form <code>/channel_name</code>. For example, for Salesforce it can something like <code>/topic/InvoiceUpdate</code>. You must not specify the complete subscription endpoint address (of the form <code>http://server:port/...</code>), because that information is part of the provider configuration information. The channel endpoint must refer to the actual channel name only. This endpoint will be appended to the provider endpoint before the actual streaming connection is made to the provider. ■ If the streaming provider's client type is HTTP, specify the endpoint address without the actual stream/channel name included. For example, Twitter's Public Stream address is:

Option	Description
	<code>https://stream.twitter.com/1.1/channelEndpoint</code>
Content Type	The HTTP Content-Type header value. Not required if the Streaming Providers Client Type field is set to Comet .
HTTP Method	The HTTP method to use for the streaming request made to the service provider. Not required if the Streaming Providers Client Type field is set to Comet .
Request Body	The optional request entity or message contents to send to the service endpoint. This is only applicable for the POST, PUT, OPTIONS and PATCH HTTP methods; the other HTTP methods do not require a message body. Not required if the Streaming Providers Client Type field is set to Comet .
HTTP Headers	Additional HTTP headers (name-value pairs) to be sent to the service endpoint (optional). Not required if the Streaming Providers Client Type field is set to Comet .
Destination Type	<p>The type of destination to which the streaming subscription response will be delivered when a notification is received from the provider:</p> <ul style="list-style-type: none"> ■ ESB Service: An IS service that will be invoked when a streaming notification is received. The service name must be of the fully qualified form <code>folder1.folder2:serviceName</code>. For more information, see "Creating an ESB Service as a Streaming Subscriber Destination" on page 52. ■ Journal Log: The IS journal log to which the streaming subscription response contents will be logged when a notification is received from the provider.

Creating an ESB Service as a Streaming Subscriber Destination

When you select the **ESB Service** option for the **Destination Type** field for a streaming subscriber, you specify an IS service that will be invoked when a streaming notification is received. Following is guidance on how to write such an IS service.

ESB Services for the Client Type of Comet

The response is present under an IData variable in the pipeline, with a key whose value is the same as the name of the subscriber. For example, if the subscriber is named `MySalesforceTopic`, then there will be a key in the pipeline with the same name.

The IData structure is:

```
[subscriber_name]
  message
    channel.name (if present)
    channel.endpoint
    data.json
    comet.msg
```

This IData variable contains the actual subscription response contents within the message pipeline variable, which contains the following pipeline variables:

- `channel.name`: The name of the channel over which the subscription response arrived.
- `channel.endpoint`: The endpoint value of the channel.
- `data.json`: The streaming response contents in JSON format (note that this is only applicable if the provider client type is Comet).
- `comet.msg`: The actual CometD message instance value. The type of the value is: `org.cometd.bayeux.Message`.

ESB Services for the Client Type of HTTP

The response is present under an IData variable in the pipeline, with a key whose value is the same as the name of the subscriber. For example, if the subscriber is named `MyTwitter`, then there will be a key in the pipeline with the same name.

The IData structure is:

```
[subscriber_name]
  message
    response.data
    channel.endpoint
```

This IData variable contains the actual subscription response contents within the message pipeline variable, which contains the following pipeline variables:

- `response.data`: A String value representing the HTTP response contents.
- `channel.endpoint`: The value corresponds to the subscriber destination from which the response was received.

For HTTP streaming providers, if an ESB service is configured to consume incoming notification messages, the service may get triggered frequently as the back end may be sending periodic control messages to ensure connectivity. In such cases, the ESB service which consumes the notifications should be designed to ignore such control messages. The contents of the periodic connectivity messages and their frequency may vary from back end to back end. For more information, refer the back end specific streaming documentation, which provides information on how streaming notifications and control messages function for a given back end.

Streaming subscriber behavior in case of an error

If a streaming subscriber runs into an error, it logs an error message and using the underlying CometD client, the streaming subscriber automatically retries the connection.

If connectivity is successfully established, a fresh subscription request is sent whenever a meta/handshake event occurs. Streaming notification messages are received as before.

This behavior is observed in the case where the streaming provider **Authentication Type** is selected as **ESB Callback**.

3 Virtual Services

■ Overview	56
■ Defining and Managing a CloudStreams Server Target	59
■ Creating a CloudStreams Governance Project	60
■ Creating a New Virtual Service (SOAP)	62
■ Creating a New Connector Virtual Service (SOAP)	118
■ Creating a New Virtual Service (REST)	126
■ Creating a New Connector Virtual Service (REST)	152
■ Important Considerations for REST Virtual Services	160

Overview

Following is a summary of the four high-level steps for creating virtual services:

1. ["Define a CloudStreams Server Target" on page 56.](#)
2. ["Create a CloudStreams Governance Project" on page 56.](#)
3. ["Create the Virtual Services" on page 56.](#)
4. ["Create Policies for the Virtual Services" on page 58.](#)

Define a CloudStreams Server Target

A CloudStreams *server target* specifies a CloudStreams server to which you will deploy virtual services and connector virtual services. You define CloudStreams server targets using Software AG Designer. You can create one or more server targets.

Create a CloudStreams Governance Project

You need to create a CloudStreams Governance project in which you will create the virtual services and their policies, plus any custom connector virtual services and their policies. You create projects in your local file system, using the CloudStreams Development plug-in provided by Designer. You can create one or more projects.

Create the Virtual Services

A *virtual service* runs on a CloudStreams server and acts as the consumer-facing proxy for a native service that runs in a SaaS application or in an on-premise application. Service requests sent from a service consumer go to a virtual service hosted on the CloudStreams server for processing rather than directly to the service provider. You should create a virtual service for each SaaS service you want to expose to consumers.

There are two types of virtual services: *virtual services* and *connector virtual services*.

- In the inbound processing scenario, when a SaaS application sends a service request, a virtual service performs security checks and other user-defined processing before sending the request to the on-premise application.
- In the outbound processing scenario, when an on-premise application sends a service request, a special kind of virtual service is used, called a connector virtual service. A *connector virtual service* handles the SaaS provider's responses and logs the request/responses and their payloads.

CloudStreams provides a default connector virtual service for each metadata handler (one for the SOAP handler and one for the REST handler). You cannot modify these default services, which are located in the WmCloudStreams package. Alternatively, you can create additional connector virtual services with custom policies.

You create both types of virtual services using the CloudStreams Development plug-in provided in Designer, and then you deploy them to a CloudStreams server.

Both types of virtual services can have *policies*, which provide governance capabilities for the services; policies are discussed later in this overview.

The Processing Steps of a Virtual Service

When you create a virtual service, you can configure the following processing steps for the service:

- The **"In Sequence" step**, which you configure to manipulate the request messages. This step can include the following sub-steps:
 - The **"Entry Step"** (required), which specifies the protocol (HTTP or HTTPS) of the requests that the virtual service will accept, and for REST services it also specifies the HTTP methods that the virtual service should be allowed to perform on a REST resource.
 - The **"Transform" step** (optional), which performs an XSLT message transformation on the request message before the virtual service submits it to the native service.
 - The **"Invoke IS Service" step** (optional), which pre-processes the request message before the virtual service submits it to the native service.
 - The **"Routing Rule" step** (required), which specifies how the virtual service will route the requests to the native service endpoint. There are four ways to route HTTP or HTTPS requests:
 - "Straight Through" routing (to route requests directly to the native service endpoint).
 - "Context-Based" routing (to route specific types of messages to specific endpoints according to context-based routing rules).
 - "Content-Based" routing (to route specific types of messages to specific endpoints based on specific values that appear in the request message).
 - "Load Balancing" routing (to distribute requests among multiple endpoints).
- The service's **"Out Sequence" step** (optional), which you configure to manipulate the response messages. This step can include the following sub-steps:
 - The **"Transform" step** (optional), which specifies how the response message from the native service provider is to be transformed before the virtual service returns it to the consuming application.
 - The **"Invoke IS Service" step** (optional), which pre-processes the request message before the virtual service returns it to the consuming application.
- The service's **"Error Sequence" step** (required). CloudStreams returns a default fault response to the consuming application, which you can customize with context variables. This fault response is used for faults returned by the native service provider as well as faults returned by internal CloudStreams exceptions (policy

violation errors, cloud connection errors and cloud connector service errors). In addition, you can:

- Choose whether or not to send the native service provider's service fault content, or just send the response message.
- Invoke IS services to pre-process or post-process the error messages.

The Processing Steps of a Connector Virtual Service

A connector virtual service can contain similar processing steps as a virtual service.

Create Policies for the Virtual Services

You should create policies that apply to one or more virtual services. A *policy* is a sequence of actions that is carried out by CloudStreams when a consumer requests a particular service through CloudStreams. You create policies using the same editor you use to create virtual services.

A policy for a virtual service can include the following kinds of actions:

- **WS-SecurityPolicy 1.2 actions**, which include authentication actions and XML security actions for the inbound requests.
- **Monitoring actions**, to monitor run-time performance conditions for virtual services, and to optimize server traffic (and to send alerts when the conditions are violated).
- **Additional actions**, to identify/authenticate consumers, to validate request/response messages against an XML schema, and to log request/response payloads.

For complete details, see the chapter ["Policies" on page 167](#).

Policies for Connector Virtual Services

Each default connector virtual service has a default policy, which logs all request/response payloads to a database. You cannot modify the default connector virtual service or its policy. Alternatively, you can create additional connector virtual services with custom policies.

If you create a connector virtual service with a custom policy, you can only include the actions in the "Monitoring" or "Additional" action categories; you cannot include the "WS-SecurityPolicy 1.2" actions (except the "Require SSL" action). For example, you might want to create a custom policy that monitors run-time performance, customizes how the service invocations are logged, validates response messages against an XML schema, or optimizes server traffic.

The procedures to perform the above tasks (except for creating policies) appear below, in the following sections:

- ["Defining and Managing a CloudStreams Server Target" on page 59](#).
- ["Creating a CloudStreams Governance Project" on page 60](#).

- "Creating a New Virtual Service (SOAP)" on page 62.
- "Creating a New Connector Virtual Service (SOAP)" on page 118.
- "Creating a New Virtual Service (REST)" on page 126.
- "Creating a New Connector Virtual Service (REST)" on page 152.

Defining and Managing a CloudStreams Server Target

A CloudStreams server target specifies a CloudStreams server to which you will deploy virtual services and connector virtual services. You define CloudStreams server targets using Software AG Designer. You can create one or more server targets.

To define and manage a server target

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. Select **Window > Preferences > Software AG > CloudStreams Servers** from the menu.
3. In the CloudStreams Servers window, click **Add**.
4. Complete the fields in the Add CloudStreams Server dialog box as follows and click **OK**.

Option	Description
Name	A name for the new target. Target names can contain alphanumeric characters and underscores (_) and hyphens (-).
Host	The server's host name (for example, localhost).
User	Optional. The Integration Server user who is permitted to deploy assets to this target. By default, only a member of the Integration Server's Administrator group is permitted to deploy assets to this target.
Port	The server's port number.
Password	Optional. The password of the Integration Server user who is permitted to deploy assets to this target. By default, the password of this user is <code>manage</code> .
Secure Connection	Indicates whether to open an HTTP session or an HTTPS session on the selected server.

Option	Description
	<ul style="list-style-type: none"> ■ If you want to open an HTTPS session on the selected server using the Secure Socket Layer (SSL), select this option.
	<p>Note: If you select this option, it is critical that you also specify the IS Truststore Name option for CloudStreams (in Integration Server Administrator, go to Solutions > CloudStreams > Administration > General). For details, see "Setting the General Options" on page 28.</p>
	<ul style="list-style-type: none"> ■ If you want to open an HTTP session on the selected server, clear this check box.
5.	<p>In the CloudStreams Servers window, test the server connection by clicking the name of the server in the list and clicking the Test button. If the connection is not active and valid, activate the deployment endpoint and modify the user credentials as required.</p>
6.	<p>To edit any parameter, click the Edit button, type your changes and click OK.</p>
7.	<p>To delete the server from the list, click the Remove button.</p>
8.	<p>To export target instances from the CloudStreams to an archive file on the file system, click the Export button.</p>
9.	<p>To import target instances from the archive file into the same CloudStreams or to another instance of the server, click the Import button.</p>

Creating a CloudStreams Governance Project

You need to create a CloudStreams Governance project in which you will create your virtual services, connector virtual services, and their policies. You create projects in your local file system, using the CloudStreams Development plug-in provided by Designer. You can create one or more projects. Each project will contain the folders in which you will create virtual services, connector virtual services, and their policies.

Note: Instead of creating a new CloudStreams Governance project, you can import an existing one. To do this, from the Software AG Designer menu click **File > Import > Software AG > CloudStreams Governance Project** and complete the dialog box that appears.

To create a CloudStreams Governance project

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.

The CloudStreams Governance view on the left side of the page lists all existing CloudStreams Governance projects.

2. Right-click an existing CloudStreams Governance project and click **New > CloudStreams Governance Project** (or click **File > New > CloudStreams Governance Project** from the menu).
3. In the New CloudStreams Governance Project wizard, complete the following fields and click **Finish**.

Option	Description
Name	<p>A project name that is a valid resource name on your operating system. The name must not be null, cannot be an empty string, and the following invalid OS resource characters are not allowed:</p> <ul style="list-style-type: none"> ■ \\ (double backward slashes) ■ / (forward slash) ■ : (colon) ■ * (asterisk) ■ ? (question mark) ■ " (double quote) ■ < (Less Than symbol) ■ > (Greater Than symbol) ■ (vertical bar)
Use Default Location	<p>This option is selected by default. The default location is the Workspace root.</p> <p>For example, if you are using C:\Workspaces \My_Workspace.. the default location would be C:\Workspaces\workspace-Eclipse Application, which is the Workspace root.</p> <p>If you want to specify a different location, clear the Use Default Location check box, click Browse and select a location.</p>
Description	Optional. A description of the project.

The new project is added to any existing projects in the CloudStreams Governance view, and it includes the default folders **Virtual Services**, **Connector Virtual Services** and **Policies**. You can create virtual services, connector virtual services and policies as described in the following sections.

4. To delete a project:
 - a. Undeploy all services in the project by right-clicking the project name and clicking **Undeploy**.

- b. Right-click the project name and click **Delete**.
5. To export a project, right-click the project name and click **Export** and complete the dialog box that appears.

Creating a New Virtual Service (SOAP)

To create a new virtual service (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, right-click the CloudStreams Governance project and click **New > Virtual Service** (or expand the project, right-click the **Virtual Services** folder and click **New Virtual Service**).
3. In the New Virtual Service wizard, complete the following fields and click **Finish**.

Option	Description
CloudStreams Governance Project	Click Browse and select a CloudStreams Governance project in which to create the virtual service.
Name	Assign a name for the service. Unlike native services, the names of virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the virtual service name, then the names of the services themselves must not contain characters that are invalid in virtual service names. <div style="background-color: #f0f0f0; padding: 5px;"> <p>Note: If you want to change the service name after it has been created, right-click the service name in the Virtual Services folder and select Rename.</p> </div>
Version	The version is always set to 1.0.
Type	Select SOAP .
WSDL	The WSDL that the virtual service uses. Select either File (and click Browse to select a WSDL) or select URL (and enter the URL of the WSDL).

Option	Description
	<p>Note: If you need to add a WSDL to the service later, you can leave the WSDL field blank and add the WSDL later (see "Managing a Virtual Service (SOAP)" on page 65).</p>
Description	Optional. A description for the virtual service. This description appears when a user displays a list of virtual services in the user interface.

The new virtual service is added to the CloudStreams Governance project, in the **Virtual Services** folder.

The Properties of a Virtual Service (SOAP)

To view and modify the properties of a virtual service (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name. (If the Properties view is not already open, click **Window > Show View > Other > General > Properties**.)
3. The **General** page in the Properties view displays the following properties:

Option	Description
Name	(Read-only field.) The service name. You can change the name of an undeployed service by right-clicking the name in the Virtual Services folder and clicking Rename .
Service Type	(Read-only field.) SOAP .
Created/Last Modified	(Read-only field.) The service's creation/modification timestamps.
Target Namespace	(Read-only field.) This value is derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
Namespaces	Click the button next to this field to view the virtual service's available namespaces (such as <code>wSDL</code> , <code>xsd</code> , <code>soap</code> , etc.).
Version	The version is always set to 1.0.

Option	Description
WSDL URL	Click this URL to display the contents of the service's abstract WSDL. If a WSDL file was not added, this will be empty. You can override the WSDL by attaching a new one; see "Managing a Virtual Service (SOAP)" on page 65 .
Description	You can change the service description.

4. View additional properties in the **Advanced** page.

Option	Description
Name	(Read-only field.) The service name. You can change the name of an undeployed service by right-clicking the name in the Virtual Services folder and clicking Rename .
Type	(Read-only field.) SOAP .
Target Namespace	(Read-only field.) The value derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
WSDL URL	Click this URL to display the contents of the service's abstract WSDL. If a WSDL file was not added, this will be empty. You can override the WSDL by attaching a new one; see "Managing a Virtual Service (SOAP)" on page 65 .
Namespaces	Click the button next to this field to view the virtual service's available namespaces (such as <code>wSDL</code> , <code>tns</code> , <code>xsd</code> , <code>soap</code> , etc.).
Version	The version is always set to 1.0.
Description	(Read-only field.) The service description.
VSD	Click the button next to this field to view the service's "virtual service definition" (VSD). This button is disabled until you deploy the virtual service to a CloudStreams server. When you deploy the virtual service to a CloudStreams server, CloudStreams generates an XML document called a <i>virtual service definition (VSD)</i> . The VSD defines the virtual service for CloudStreams, and contains all the resources required to deploy the virtual service to a CloudStreams server, including the policy that applies to the service. You cannot edit the VSD.

Option	Description
Applicable Policies	<p>Note: If multiple policies apply to the service, CloudStreams combines all those policies into a single policy known as the <i>effective policy</i>. The effective policy is a simple UNION of the run-time actions specified in all policies that apply to a service. To create the effective policy, CloudStreams evaluates the combined list of actions from all policies, using a set of internal rules known as Policy Conflict Resolution rules. For details, see "Policy Conflict Resolution Rules" on page 203.</p>
Endpoint	Click the button next to this field to view the endpoint to which the virtual service is deployed, if applicable.
Deployed Status	(Read-only field.) Indicates whether the service is Deployed , Undeployed or Not Deployed (which is the initial status before you deploy the service). To deploy or undeploy a service, see " Deploying Virtual Services and Connector Virtual Services " on page 199.
Resource	Not applicable to SOAP services.

Managing a Virtual Service (SOAP)

You can rename, delete, deploy or undeploy a virtual service, attach a WSDL to it, and change the list of policies that apply to the virtual service.

To manage a SOAP virtual service

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project.
3. Right-click the virtual service name and choose one of the following tasks from the context menu:

Option	Description
Rename	Use this option to assign a new name to the service. The service must be undeployed before you can rename it

Option	Description
	<p>(check the Deployed Status field in the Advanced pageUse of the service's Properties view).</p> <p>Unlike native services, the names of virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the virtual service name, then the names of the services themselves must not contain characters that are invalid in virtual service names.</p>
Delete	Deletes the virtual service from CloudStreams.
Attach WSDL	<p>Use this option to attach a WSDL to an existing virtual service. The new attached WSDL will override the old WSDL. In the dialog box that appears, choose one of the following options and click OK:</p> <ul style="list-style-type: none"> ■ URL: Specify the URL of the WSDL to attach. ■ File: Click Browse to select a WSDL from your local file system. <div style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p>Note: If the new WSDL contains referenced XSDs or WSDL, they will be copied if they are resolved by CloudStreams. But if the referenced XSDs or WSDL cannot be resolved, a dialog box will prompt you whether to import the unresolved XSDs or WSDL.</p> </div>
Deploy	Deploys the virtual service to a CloudStreams server target. For more information, see " Deploying Virtual Services and Connector Virtual Services " on page 199.
Undeploy	Undeploys the virtual service from a CloudStreams server target.

4. To change the list of policies that apply to the virtual service, see "[Modifying Policies](#)" on page 172.

The "In Sequence" Step (SOAP)

You configure the service's "In Sequence" step to manipulate the request messages. This step can include the following sub-steps:

- The "Entry Step" (required), which specifies the protocol (HTTP or HTTPS) and SOAP format (1.1 or 1.2) of the requests that the virtual service will accept.

- The "Transform" step (optional), which performs an XSLT message transformation on the request message before it is submitted to the native service.
- The "Invoke IS Service" step (optional), which pre-processes the request message before it is submitted to the native service.
- The "Routing Rule" step (required), which specifies how the virtual service will route the service requests to the native service endpoint. There are four ways to route HTTP or HTTPS requests:
 - "Straight Through" routing (to route requests directly to the native service endpoint).
 - "Context-Based" routing (to route specific types of messages to specific endpoints according to context-based routing rules).
 - "Content-Based" routing (to route specific types of messages to specific endpoints based on specific values that appear in the request message).
 - "Load Balancing" routing (to distribute requests among multiple endpoints).

The "Entry Step" (SOAP)

The Entry Step (required) specifies the protocol (HTTP or HTTPS) and SOAP format (1.1 or 1.2) of the requests that the virtual service will accept. This step is required.

This step allows you to bridge protocols between the consuming application and the native service. For example, suppose you have a native service that is exposed over HTTPS and a consuming application that submits SOAP requests over HTTP. In this situation, you can configure the virtual service's Entry Step to accept HTTP requests and configure its Routing Rule step to route the request to the Web service using HTTPS.

To configure the "Entry Step" (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.

The virtual service editor displays the three main processing steps: **In Sequence**, **Out Sequence** and **Error Sequence**.

3. In the virtual service editor, expand **In Sequence**.

By default, the **In Sequence** step contains the sub-steps **Entry Step** and **Routing Rule**. (You can add the optional **Transform** and **Invoke IS Service** steps, as described later.)

4. Click **Entry Step** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read only.) Entry Step.
Protocol	The protocol (HTTP or HTTPS) over which the virtual service will accept requests. To specify HTTPS, select both HTTP and SSL .
Format	The SOAP format (SOAP 1.1 or SOAP 1.2) of the requests that the virtual service will accept.

The "Transform" Step (Inbound, SOAP)

The optional "Transform" step specifies how the SOAP request message is to be transformed before it is submitted to the native service.

No message transformation is required as long as the request message structure matches the request message structure that is required by the operation associated with the `soapAction`. However, in some cases a virtual service might need to transform SOAP messages.

For example, you might need to accommodate differences between the message content that a consuming application is capable of submitting and the message content that a native service expects. For example, if the consuming application submits an order record using a slightly different structure than the structure expected by the native service, you can use the Transform step to transform the record submitted by the consuming application to the structure required by the Web service.

In this case, you would need to create two Transform steps:

- One in the "In Sequence" step, to transform the request messages into the format required by the native service, before CloudStreams sends the requests to the native services. To do this, you pass the message to an XSLT transformation file. (Additionally in this case, the transformation is required if the virtual service has a schema validation policy that validates the requests.)
- One in the "Out Sequence" step, to transform the native service's response messages into the format required by the consumer applications, before CloudStreams returns the responses to the consumer applications.

To add the "Transform" step (inbound, SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.

- In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.

- Right-click **In Sequence** and select **Transform**.

The Transform step is added under the Entry Step. You cannot change the order of the steps.

- Click **Transform** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only.) Transform .
XSLT File	<p>The XSLT file to transform the request message before it is submitted to the native service. Click Browse to select a file from your file system and click Save.</p> <p>The XSL file uploaded by the user should not contain the XML declaration in it (<code><?xml version="1.0" encoding="UTF-8"></code>). This is because when the virtual service is deployed to CloudStreams, CloudStreams embeds the XSL file in the virtual service definition (VSD), and since the VSD itself is in XML format, there cannot be an XML declaration line in the middle of it. This can lead to unexpected deployment issues which can be avoided by making sure the XSL file does not contain the declaration line.</p>

Note: If you make changes to the XSLT file in the future, you must re-deploy the virtual service.

- To create an additional Transform step, right-click **In Sequence** and select **Transform** again.
- To delete a Transform step, right-click **Transform** and click **Delete Step**.

The "Invoke IS Service" Step (Inbound, SOAP)

This optional step invokes a user-defined Integration Server flow service. You can invoke an IS service in:

- The "In Sequence" step, to pre-process the request message before it is submitted to the native service.
- The "Out Sequence" step, to pre-process the response message from the native service before it is returned to the consuming application.

The IS service you invoke must be running on the same Integration Server as CloudStreams. It can call out a C++ or Java or .NET function. It can also call other IS services to manipulate the SOAP message.

You can use the following constructs in an IS service:

- Predefined or custom context variables. For more information, see ["Using Context Variables in IS Services" on page 100](#).
- The Security API provided by CloudStreams (for SOAP-based services only). For more information, see ["Using the Security API in IS Services" on page 89](#).

You can create multiple "Invoke IS Service" steps.

To add the "Invoke IS Service" step (inbound, SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
3. Right-click **In Sequence** and click **Invoke IS Service**.

The "Invoke IS Service" step is added under the Entry Step (and under a Transform step, if one exists).

4. Click **Invoke IS Service** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only.) Invoke IS Service .
Service	The IS service to pre-process the request message before it is submitted to the native service.

5. When you define the IS service, the **Pipeline In** section of the flow should have the following input variables:
 - `proxy.name`: This is the name of the virtual service.
 - `SOAPEnvelope`: This is of the Java type `org.apache.axiom.soap.SOAPEnvelope`.
 - `MessageContext`: CloudStreams will automatically place a `MessageContext` variable into the pipeline before executing the IS service call. `MessageContext` is of the Java type `org.apache.axis2.context.MessageContext`.

Integration Server users can use the `Axis2 MessageContext` object to manipulate the incoming SOAP request. The Integration Server provides built-in services

(the `pub.soap.*` services) to work with the `MessageContext` object to get/set/modify the SOAP body, header, properties, etc. Integration Server users should use these services to extract the information they need from the `MessageContext` to build the necessary business logic. Users do not need to understand Axis2 or Axiom (the XML object model based on StAX) to work with the SOAP request, because if they are familiar with the Integration Server `pub.soap.*` services, they can accomplish most of the tasks. For more information about these related Integration Server services, see the *webMethods Integration Server Built-In Services Reference*.

6. To create an additional "Invoke IS Service" step, right-click **In Sequence** and select **IS Service** from the context menu.
7. To delete an "Invoke IS Service" step, right-click **Invoke IS Service** and click **Delete**.

The "Routing Rule" Step for HTTP or HTTPS (SOAP)

The Routing Rule step is required. You can choose one of the following protocols in your Routing Rule step for routing the requests:

- "Straight Through" routing (to route requests directly to the native service endpoint).
- "Context-Based" routing (to route specific types of messages to specific endpoints according to context-based routing rules).
- "Content-Based" routing (to route specific types of messages to specific endpoints based on specific values that appear in the request message).
- "Load Balancing" routing (to distribute requests among multiple endpoints).

The "Straight Through" Routing Rule Step (SOAP)

When you select the "Straight Through" routing protocol, the virtual service will route the requests directly to the native service endpoint you specify. You may specify how to authenticate requests (as with all routing protocols).

To configure the "Routing Rule" step for "Straight Through" routing (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
3. Expand **In Sequence**.
4. Click **Routing Rule** and complete the following fields in the **General** page in the Properties view.

<u>Option</u>	<u>Description</u>
Name	You can optionally change the step name to any other name. There are no naming restrictions.

Option	Description
Type	(Read-only field.) Routing Rule.
Protocol	(Read-only field.) HTTP.
Routing Type	Select Straight Through.
Default To	<p data-bbox="524 556 1299 588">Specify the URL of the service to which to route the request.</p> <p data-bbox="524 611 1218 674">Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows:</p> <ul style="list-style-type: none"> <li data-bbox="524 697 1373 1056"> <p data-bbox="524 697 1354 728">■ SOAP Optimization Method: Select one of the following options:</p> <ul style="list-style-type: none"> <li data-bbox="570 751 850 783">■ None: The default. <li data-bbox="570 806 1325 932">■ MTOM: Indicates that CloudStreams expects to receive a request with a Message Transmission Optimization Mechanism (MTOM) attachment, and will forward the attachment to the native service. <li data-bbox="570 955 1373 1056">■ SwA: Indicates that CloudStreams expects to receive a "SOAP with Attachment" (SwA) request, and will forward the attachment to the native service.

Note: Bridging between SwA and MTOM is not supported. If a consumer sends an SwA request, CloudStreams can only forward SwA to the native provider. The same is true for MTOM, and applies to responses received from the native provider. That is, an SwA or MTOM response received by CloudStreams from a native provider will be forwarded to the caller using the same format it received.

Note: When sending SOAP requests that do not contain a MTOM or SWA attachment to a virtual service for a native provider endpoint that returns an MTOM or SWA response, the request `Accept` header must be set to 'multipart/related' (or the virtual service's Request Processing Step should include an IS service callout that sets the `BUILDER_TYPE` context variable to 'multipart/related'). This is necessary so CloudStreams knows how to parse the response properly.

- **Connection Timeout:** The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified

Option	Description
Use credentials from incoming request	<p>(or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server.</p> <ul style="list-style-type: none"> ■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. ■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank. <ul style="list-style-type: none"> ■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication. ■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication. <p>Default. Authenticates requests based on the credentials specified in the HTTP header. CloudStreams passes the <code>Authorization</code> header present in the original client request to the native service.</p>
Use specific credentials	Authenticates requests according to the values you specify in the User and Password fields, and optionally the Domain field.
Invoke service anonymously	Does not authenticate requests.
Use existing HTTP Headers	Use the HTTP headers that are contained in the requests.
Customize HTTP Headers	Use the HTTP headers that you specify in the Name and Value columns on this page. If you need to specify multiple headers, use the plus button to add rows.

The "Context-Based" Routing Rule Step (SOAP)

If you have a native service that is hosted at two or more endpoints, you can use the Context-Based routing protocol to route specific types of messages to specific endpoints according to the context-based routing rules you create.

A routing rule specifies where the requests should be routed, and the criteria by which they should be routed there. For example, requests can be routed according to certain

consumers, certain dates/times, or according to requests that exceed/fall below a specified metric (Total Count, Success Count, Fault Count, etc.). You can create one or more rules. For example, you might use this capability to route requests from certain high-priority consumers to endpoints on a fast machine.

To configure the "Routing Rule" step for "Context-Based" routing (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.
3. Expand **In Sequence**.
4. Click **Routing Rule** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) HTTP .
Routing Type	Select Context-Based .
Rule Name	Assign a name to the rule. Then, click the icon next to this field and choose one of the following variables in the Configure Routing Rule dialog box that appears: Time , IP Address (IPv4 or IPv6 format), Date , Consumer , Predefined Context Variable or Custom Context Variable (see " Using Context Variables in IS Services " on page 100). Then specify a value and operator appropriate for your chosen variable. If you need to specify multiple variables, use the plus button at the end of the row to add rows.

Note: If you select the value **Custom Context Variable**, you must write an IS service to get/set the custom context variable, and then invoke that service in an **Invoke IS Service** step. CloudStreams provides an API to get/set custom context variables. For more information, see "[The API for Context Variables](#)" on page 105. CloudStreams automatically declares any custom context variables you have specified; there is no need for you to declare them.

Option	Description
Route To	<p>Specify the URL of the native service to route the request to, if the rule criteria are met.</p> <p>Click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows:</p> <ul style="list-style-type: none"> ■ Optimization Method: Select one of the following options: <ul style="list-style-type: none"> ■ None: The default. ■ MTOM: Indicates that CloudStreams expects to receive a request with a Message Transmission Optimization Mechanism (MTOM) attachment, and will forward the attachment to the native service. ■ SwA: Indicates that CloudStreams expects to receive a "SOAP with Attachment" (SwA) request, and will forward the attachment to the native service. <div style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <p>Note: Bridging between SwA and MTOM is not supported. If a consumer sends an SwA request, CloudStreams can only forward SwA to the native provider. The same is true for MTOM, and applies to responses received from the native provider. That is, an SwA or MTOM response received by CloudStreams from a native provider will be forwarded to the caller using the same format it received.</p> <p>Note: When sending SOAP requests that do not contain a MTOM or SWA attachment to a virtual service for a native provider endpoint that returns an MTOM or SWA response, the request <code>Accept</code> header must be set to <code>multipart/related</code> (or the virtual service's "In Sequence" step should include an IS service callout that sets the <code>BUILDER_TYPE</code> context variable to <code>multipart/related</code>). This is necessary so CloudStreams knows how to parse the response properly.</p> </div> <ul style="list-style-type: none"> ■ Connection Timeout: The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified (or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server.

Option	Description
	<ul style="list-style-type: none"> <li data-bbox="524 323 1341 428">■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. <li data-bbox="524 470 1341 638">■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank. <ul style="list-style-type: none"> <li data-bbox="570 659 1341 722">■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication. <li data-bbox="570 743 1341 884">■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication.
Default To	Enter a native service endpoint to route the request to in case all routing rules evaluate to False. Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box, as described for the Route To field above.
Use credentials from incoming request	Default. Authenticates requests based on the credentials specified in the HTTP header. CloudStreams passes the <code>Authorization</code> header present in the original client request to the native service.
Use specific credentials	Authenticates requests according to the values you specify in the User , Password and Domain fields.
Invoke service anonymously	Does not authenticate requests.
Use existing HTTP Headers	Use the HTTP headers that are contained in the requests.
Customize HTTP Headers	Use the HTTP headers that you specify in the Name and Value columns on this page. If you need to specify multiple headers, use the plus button to add rows.

The "Content-Based" Routing Rule Step (SOAP)

If you have a native service that is hosted at two or more endpoints, you can use the Content-Based routing protocol to route specific types of messages to specific endpoints based on specific values that appear in the request message.

You might use this capability, for example, to determine which operation the consuming application has requested, and route requests for complex operations to an endpoint on a fast machine.

The requests are routed according to the content-based routing rules you create. That is, they are routed based on the successful evaluation of one or more XPath expressions that are constructed utilizing the content of the request payload. For example, a routing rule might allow requests for half of the methods of a particular service to be routed to Service A, and the remaining methods to be routed to Service B.

To configure the "Routing Rule" step for "Content-Based" routing (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.
3. Expand **In Sequence**.
4. Click **Routing Rule** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) HTTP .
Routing Type	Select Content-Based .
Rule Name	Assign a name to the rule.
XPath	Create an XPath expression as follows: <ol style="list-style-type: none"> a. Click the icon next to this field to display the XPath Editor. b. In the XPath Editor that appears, the Namespace tab displays all predefined namespaces in the WSDL. If you want to add custom namespaces, click Add Custom Namespace/prefix, specify a name and value for the namespace, and click OK.

Option	Description
	<p>To add additional rows, use the plus button at the end of the row to add them.</p> <p>c. In the XPath Editor's Nodes tab, expand the namespace's node, choose the method you want for the XPath expression, and click OK.</p> <p>d. In the XPath Editor's Evaluator tab, evaluate the XPath expression by specifying an argument in the XPath Expression field, and clicking Evaluate.</p> <p>The true/false result of the evaluation is displayed in the Result field.</p> <p>To specify additional XPath expressions, use the plus button at the end of the row to add them.</p>
Route To	<p>Specify where to route the request if the rule criteria are met. Specify either the URL of a native service or a connection pool name.</p> <p>Click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows:</p> <ul style="list-style-type: none"> ■ Optimization Method: Select one of the following options: <ul style="list-style-type: none"> ■ None: The default. ■ MTOM: Indicates that CloudStreams expects to receive a request with a Message Transmission Optimization Mechanism (MTOM) attachment, and will forward the attachment to the native service. ■ SwA: Indicates that CloudStreams expects to receive a "SOAP with Attachment" (SwA) request, and will forward the attachment to the native service. <p>Note: Bridging between SwA and MTOM is not supported. If a consumer sends an SwA request, CloudStreams can only forward SwA to the native provider. The same is true for MTOM, and applies to responses received from the native provider. That is, an SwA or MTOM response received by CloudStreams from a native provider will be forwarded to the caller using the same format it received.</p> <p>Note: When sending SOAP requests that do not contain a MTOM or SWA attachment to a virtual service for a native provider endpoint that returns an</p>

Option	Description
	<p data-bbox="706 306 1369 548">MTOM or SWA response, the request <code>Accept</code> header must be set to 'multipart/related' (or the virtual service's Request Processing Step should include an IS service callout that sets the <code>BUILDER_TYPE</code> context variable to 'multipart/related'). This is necessary so CloudStreams knows how to parse the response properly.</p> <ul style="list-style-type: none"> <li data-bbox="524 590 1369 726">■ Connection Timeout: The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified (or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server. <li data-bbox="524 747 1369 852">■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. <li data-bbox="524 894 1369 1304">■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank. <ul style="list-style-type: none"> <li data-bbox="570 1083 1369 1146">■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication. <li data-bbox="570 1167 1369 1304">■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication.
Default To	Enter a native service endpoint to route the request to in case all routing rules evaluate to False. Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box, as described for the Route To field above.

The "Load Balancing" Routing Rule Step (SOAP)

If you have a native service that is hosted at two or more endpoints, you can use the Load Balancing protocol to distribute requests among the endpoints.

The requests are intelligently routed based on the "round-robin" execution strategy. The load for a service is balanced by directing requests to two or more services in a pool, until the optimum level is achieved. The application routes requests to services in the pool sequentially, starting from the first to the last service without considering the individual performance of the services. After the requests have been forwarded to all the services in the pool, the first service is chosen for the next loop of forwarding.

Load-balanced endpoints also have automatic Failover capability. If a load-balanced endpoint is unavailable (for example, if a connection is refused), then that endpoint is marked as "down" for the number of seconds you specify in the **Suspend Interval** field (during which the endpoint will not be used for sending the request), and the next configured endpoint is tried. If all the configured load-balanced endpoints are down, then a SOAP fault is sent back to the client. After the suspension period expires, each endpoint marked will be available again to send the request.

To configure the "Routing Rule" step for "Load Balancing" routing (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
3. Expand **In Sequence**.
4. Click **Routing Rule** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) HTTP .
Routing Type	Select Load Balancing .
Route To	<p>Specify the URLs of two or more services in a pool to which the requests will be routed. The application routes the requests to services in the pool sequentially, starting from the first to the last service without considering the individual performance of the services. After the requests have been forwarded to all the services in the pool, the first service is chosen for the next loop of forwarding.</p> <p>To specify additional URLs, use the plus button next to the field to add rows.</p> <p>Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows. These properties will apply to all the endpoints.</p>

- **Optimization Method:** Select one of the following options:

Option	Description
	<ul style="list-style-type: none"> ■ None: The default. ■ MTOM: Indicates that CloudStreams expects to receive a request with a Message Transmission Optimization Mechanism (MTOM) attachment, and will forward the attachment to the native service. ■ SwA: Indicates that CloudStreams expects to receive a "SOAP with Attachment" (SwA) request, and will forward the attachment to the native service. <p>Note: Bridging between SwA and MTOM is not supported. If a consumer sends an SwA request, CloudStreams can only forward SwA to the native provider. The same is true for MTOM, and applies to responses received from the native provider. That is, an SwA or MTOM response received by CloudStreams from a native provider will be forwarded to the caller using the same format it received.</p> <p>Note: When sending SOAP requests that do not contain a MTOM or SWA attachment to a virtual service for a native provider endpoint that returns an MTOM or SWA response, the request <code>Accept</code> header must be set to 'multipart/related' (or the virtual service's Request Processing Step should include an IS service callout that sets the <code>BUILDER_TYPE</code> context variable to 'multipart/related'). This is necessary so CloudStreams knows how to parse the response properly.</p> <ul style="list-style-type: none"> ■ Connection Timeout: The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified (or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server. ■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. ■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank.

Option	Description
	<ul style="list-style-type: none"> ■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication. ■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication.
Suspend Interval	A numeric timeout value (in seconds) for a failed endpoint. Default: 30. When this timeout value expires, the system routes the execution of the virtual service to the next consecutive Web service endpoint specified in the Route To field.
Use Credentials from Incoming Request	Default. Authenticates requests based on the credentials specified in the HTTP header. CloudStreams passes the <code>Authorization</code> header present in the original client request to the native service.
Use Specific Credentials	Authenticates requests according to the values you specify in the User , Password and Domain fields.
Invoke Service Anonymously	Does not authenticate requests.
Use Existing HTTP Headers	Use the HTTP headers that are contained in the requests.
Customize HTTP Headers	Use the HTTP headers that you specify in the Name and Value columns on this page. If you need to specify multiple headers, use the plus button to add rows.

The "Out Sequence" Step (SOAP)

You configure the service's "Out Sequence" step (optional) to manipulate the response messages from the native service before they are returned to the consuming applications. This step includes the following sub-steps:

- The "Transform" step (optional), which invokes an XSLT transformation file to transform the SOAP response payloads from XML format to the format required by the consumer.
- The "Invoke IS Service" step (optional), which processes the response message from the native service provider before it is returned to the consuming application.

The "Transform" Step (Outbound, SOAP)

Add this step if you need to invoke an XSLT transformation file to transform the native service's response messages into the format required by the consumer applications, before CloudStreams returns the responses to the consumer applications. For more information about when to use the Transform step, see ["The "Transform" Step \(Inbound, SOAP\)" on page 68](#).

To add the "Transform" step (outbound, SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
3. Right-click **Out Sequence** and click **Transform**.

The Transform step is added to the Out Sequence step.

4. Click **Transform** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Transform .
XSLT File	The XSLT file to transform the response message before it is returned to the consuming application. Click Browse to select a file from your file system and click Save .

The XSL file uploaded by the user should not contain the XML declaration in it (`<?xml version="1.0" encoding="UTF-8">`). This is because when the virtual service is deployed to CloudStreams, CloudStreams embeds the XSL file in the virtual service definition (VSD), and since the VSD itself is in XML format, there cannot be an XML declaration line in the middle of it. This can lead to unexpected deployment issues which can be avoided by making sure the XSL file does not contain the declaration line.

Note: If you make changes to the XSLT file in the future, you must re-deploy the virtual service.

5. To create an additional Transform step, right-click **In Sequence** and select **Transform** again.

- To delete a Transform step, right-click **Transform** and click **Delete**.

The "Invoke IS Service" Step (Outbound, SOAP)

This optional step invokes a user-defined Integration Server flow service. You can invoke an IS service in the service's "Out Sequence" step to pre-process the response message from the native service before it is returned to the consuming application. For more information about IS services, see "[The "Invoke IS Service" Step \(Inbound, SOAP\)" on page 69](#).

To add the "Invoke IS Service" step (outbound, SOAP)

- Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
- In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.
- Right-click **Out Sequence** and click **Invoke IS Service**.

The Invoke IS Service step is added to the Out Sequence step.

- Click **Invoke IS Service** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Invoke IS Service .
Service	The IS service to pre-process the response message from the native service before it is returned to the consuming application.

- When you define the IS service, the **Pipeline In** section of the flow should have the following input variables:
 - `proxy.name`: This is the name of the virtual service.
 - `SOAPEnvelope`: This is of the Java type `org.apache.axiom.soap.SOAPEnvelope`.
 - `MessageContext`: CloudStreams will automatically place a `MessageContext` variable into the pipeline before executing the IS service call. `MessageContext` is of the Java type `org.apache.axis2.context.MessageContext`.

Integration Server users can use the Axis2 `MessageContext` object to manipulate the incoming SOAP request. The Integration Server provides built-in services (the `pub.soap.*` services) to work with the `MessageContext` object to get/set/modify the SOAP body, header, properties, etc. Integration Server users should

use these services to extract the information they need from the `MessageContext` to build the necessary business logic. Users do not need to understand Axis2 or Axiom (the xml object model based on StAX) to work with the SOAP request, because if they are familiar with the Integration Server `pub.soap.*` services, they can accomplish most of the tasks. For more information about these related Integration Server services, see the *webMethods Integration Server Built-In Services Reference*.

6. To create an additional "Invoke IS Service" step, right-click **In Sequence** and click **IS Service** again.
7. To delete an "Invoke IS Service" step, right-click **Invoke IS Service** and click **Delete**.

The "Error Sequence" Step (SOAP)

CloudStreams returns a default fault response to the consuming application, which you can customize with context variables. This fault response is used for faults returned by the native service provider as well as faults returned by internal CloudStreams exceptions (policy violation errors, cloud connection errors and cloud connector service errors). In addition, you can:

- Choose whether or not to send the native service provider's service fault content, or just send the response message.
- Invoke IS services to pre-process or post-process the error messages.

You use this step to configure error messaging for each virtual service individually. If you want to configure global error responses for *all* virtual services, set the Service Fault Configuration options, which are located in the Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Service Fault Configuration**). For details, see ["Setting the Service Fault Configuration Options" on page 40](#).

To configure the "Error Sequence" step (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.
3. Expand **Error Sequence**.
4. Click **Error Messaging** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Error Messaging .

Option	Description
Error Message	<p>Select one or both of the following options:</p> <p>Custom Failure Response Message: When you select this option, CloudStreams returns the following fault response to the consuming application:</p> <pre>CloudStreams encountered an error:\$ERROR_MESSAGE while executing operation:\$OPERATION service: \$SERVICE at time:\$TIME on date:\$DATE. The client ip was:\$CLIENT_IP. The current user:\$USER. The consumer application:\$CONSUMER_APPLICATION".</pre> <p>This fault response is returned in both of the following cases:</p> <ul style="list-style-type: none"> ■ When a fault is returned by the native service provider. <p>In this case, the <code>\$ERROR_MESSAGE</code> variable in the fault response will contain the message produced by the provider's exception that caused the error. This is equivalent to the <code>getMessage</code> call on the Java Exception. This maps to the <code>faultString</code> element for SOAP 1.1 or the <code>Reason</code> element for SOAP 1.2 catch. CloudStreams discards the native service provider's fault and does not return this content to the web service caller since it could be considered a security issue, especially if the native provider is returning a stack trace with its response.</p> ■ When a fault is returned by internal CloudStreams exceptions (policy violation errors, cloud connection errors and cloud connector service errors). <p>In this case, the <code>\$ERROR_MESSAGE</code> variable will contain the error messages generated by CloudStreams.</p> <p>The default fault response contains predefined fault handler variables (<code>\$ERROR_MESSAGE</code>, <code>\$OPERATION</code>, etc.), which are described in "The Fault Handler Variables" on page 87.</p> <p>You can customize the default fault response using the following substitution variables, where CloudStreams replaces the variable reference with the real content at run time:</p> <ul style="list-style-type: none"> ■ The predefined context variables listed in "The Predefined Context Variables" on page 100. ■ Custom context variables that you declare using the CloudStreams API (see "The API for Context Variables" on page 105).

Option	Description
Processing Method	<p>Note: If you want to reference a custom context variable that you have already defined in a context-based routing rule (as opposed to one you have declared using CloudStreams API for context variables), then you must add the prefix <code>\$mx</code> to the variable name in order to reference the variable. For example, if you defined the variable <code>TAXID</code>, you would reference it as <code>\$mx:TAXID</code>.</p>
	<p>Native Provider Fault: When you select this option, CloudStreams sends the native service provider's service fault content, if one is available. CloudStreams will send whatever content it received from the native service provider.</p> <p>If you select this option, the Custom Failure Response Message is ignored when a fault is returned by the native service provider. (Faults returned by internal CloudStreams exceptions will still be handled by the Custom Failure Response Message option.)</p>
	<p>Optionally select either of the following:</p> <ul style="list-style-type: none"> ■ Pre-Processing: Select this option if you want to invoke an IS service to manipulate the response message before the Error Sequence step is invoked. The IS service will have access to the response message context (the axis2 <code>MessageContext</code> instance) before it is updated with the custom error message. For example, you might want to send emails or perform custom alerts based on the response payload. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, SOAP)" on page 69. ■ Post-Processing: Select this option if you want to invoke an IS service to manipulate the service fault after the Error Sequence step is invoked. The IS service will have access to the entire service fault and the custom error message. You can make further changes to the fault message structure, if needed. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, SOAP)" on page 69.

The Fault Handler Variables

The following fault handler variables are used in fault response messages. You can define fault response messages either in a virtual service's "Error Sequence" step or in the global Service Fault Configuration page in Integration Server Administrator.

Error Handler Variable	Description
\$ERROR_MESSAGE	<p>When a fault is returned by the native service provider, this variable will contain the message produced by the provider's exception that caused the error. This is equivalent to the <code>getMessage</code> call on the Java Exception. This maps to the <code>faultString</code> element for SOAP 1.1 or the <code>Reason</code> element for SOAP 1.2 catch. For REST service calls, the message is returned inside an <code></Exception></code> tag. If the response is XML, the message is returned inside <code><Exception>'custom message'</Exception></code>. If the response is JSON, it will be returned inside <code>{"Exception": "Invalid response"}</code>.</p> <p>When a fault is returned by internal CloudStreams exceptions (policy violation errors, cloud connection errors and cloud connector service errors), this variable will contain the error messages generated by CloudStreams.</p>
\$OPERATION	The operation that was invoked when this error occurred.
\$SERVICE	<p>The service that was invoked when this error occurred.</p> <p>Note: Deprecated, but still a valid alias. Alternatively, use <code>\$SERVICE_NAME</code> (see "The Predefined Context Variables" on page 100).</p>
\$TIME	The time (as determined on the Container side) at which the error occurred.
\$DATE	The date (as determined on the Container side) at which the error occurred.
\$CLIENT_IP	<p>The IP address of the client invoking the service. This might be available for only certain invoking protocols, such as HTTP(S).</p> <p>Note: Deprecated, but still a valid alias. Alternatively, use <code>\$INBOUND_IP</code> (see "The Predefined Context Variables" on page 100).</p>

Error Handler Variable	Description
<code>\$(USER)</code>	The currently authenticated user. The user will be present only if the Transport/SOAP Message have user credentials.
<code>\$(CONSUMER_APPLICATION)</code>	The currently identified consumer application. If the service's policy does not contain the "Identify Consumer" action, <code>\$(CONSUMER_APPLICATION)</code> will return "null".

IS Service Constructs

As previously mentioned, an IS (Integration Server) service is a user-defined Integration Server flow service that you can invoke in the "In Sequence" and "Out Sequence" steps of virtual services (both SOAP-based and REST-based).

You can use the following constructs in an IS service:

- The Security API provided by CloudStreams (for SOAP-based services only). For more information, see ["Using the Security API in IS Services" on page 89](#).
- Predefined or custom context variables. For more information, see ["Using Context Variables in IS Services" on page 100](#).

Using the Security API in IS Services

Note: This API can only be used for SOAP-based virtual services.

CloudStreams provides Java services that you can use to support WS-Security functionality in an IS service that you invoke in the "In Sequence" step. These following services are included:

- ["pub.cloudstreams.security.ws.AddUserNameToken" on page 89](#)
- ["pub.cloudstreams.security.ws.AddX509Token" on page 93](#)
- ["pub.cloudstreams.security.ws.AddSamlSenderVouchesToken" on page 94](#)
- ["pub.cloudstreams.security.ws.AddTimestamp" on page 95](#)
- ["pub.cloudstreams.security.ws.AddWSAddressingHeaders" on page 97](#)

pub.cloudstreams.security.ws.AddUserNameToken

Adds the WS-Username Token 1.0 and 1.1 to the request. This service includes the following input parameters:

Parameter	Data Type	Description	Default Value
username	String	Required. Java Type: String. The value that will be added as the Username element in the token.	""
MessageContext	Object	Required. Java Type: org.apache.axis2.context.MessageContext. CloudStreams will place a MessageContext parameter into the pipeline before executing the IS service call.	org.apache.axis2.context.MessageContext instance
password	String	Java Type: String. The password for the token; must be specified if the passwordType (see below) is specified as either TEXT or DIGEST.	""
passwordType	String	Java Type: String. Specifies how the password will be added in the token. Valid values: <ul style="list-style-type: none"> ■ NONE: The password is not added. ■ TEXT: The password is added in plain text. ■ DIGEST: The password is added in digested form (as specified in the 	NONE

Parameter	Data Type	Description	Default Value
		UsernameToken profile).	
addNonce	Boolean	Java Type: Boolean. Specifies whether the <code>Nonce</code> element will be added to the token.	False
addCreated	Boolean	Java Type: Boolean. Specifies whether the <code>Created</code> element will be added to the token.	False
salt	byte[]	Java Type: byte[]. The value for the / <code>wsse11:UsernameToken/</code> <code>wsse:Salt</code> element. Its value is a 128 bit number serialized as <code>xs:base64Binary</code> .	null
iteration	int	Java Type: Integer. Indicates the number of times the hashing operation is repeated when deriving the key. It is expressed as a <code>xs:unsignedInteger</code> value. If it is not present, a value of 1000 is used for the iteration count.	1000
useMac	Boolean	Java Type: Boolean. Indicates whether the derived key will be used as a Message Authentication Code (MAC) or as	False

Parameter	Data Type	Description	Default Value
		a symmetric key for encryption.	
<code>useBasicAuthCredentials</code>	Boolean	Java Type: Boolean. If this parameter is set to <code>True</code> , <code>CloudStreams</code> will try to use the username and password from the <code>Authorization</code> HTTP header. In this case the <code>username</code> and <code>password</code> fields need not be specified.	<code>False</code>
<code>actor</code>	String	Java Type: String. Indicates the value of the SOAP <code>actor</code> attribute if a new security header is being added to the SOAP request. If the request already has a security header with the <code>actor</code> specified in it, then this value will not overwrite it.	<code>""</code>
<code>mustUnderstand</code>	Boolean	Java Type: Boolean. Specifies whether the security header will have the <code>mustUnderstand</code> attribute set to 0 or 1 (<code>false</code> / <code>true</code>). If the security header already has this attribute set, then this value will not overwrite it.	<code>False</code>

pub.cloudstreams.security.ws.AddX509Token

Adds an X.509 certificate (or certificate chain) as a `BinarySecurityToken` (BST) element in the outbound SOAP request. This service includes the following input parameters:

Parameter	Data Type	Description	Default Value
<code>MessageContext</code>	Object	Required. Java Type: <code>org.apache.axis2.context.MessageContext</code> . CloudStreams will place a <code>MessageContext</code> parameter into the pipeline before executing the IS service call.	<code>org.apache.axis2.context.MessageContext</code> instance
<code>keystoreFile</code>	String	Required. Java Type: String. The absolute path to a keystore file on the system where CloudStreams is running.	""
<code>keystorePassword</code>	String	Java Type: String. Required. The password for the keystore.	""
<code>keystoreType</code>	String	Java Type: String. The type of keystore represented by the file (can be <code>JKS</code> , <code>JCEKS</code> or <code>PKCS12</code>).	<code>JKS</code>
<code>keyAlias</code>	String	Java Type: String. Required. The key alias whose X509 certificate will be sent in the SOAP request as a BST.	""
<code>useCertificatePath</code>	Boolean	Java Type: Boolean. If set to <code>True</code> , CloudStreams will use the entire certificate chain represented by the key alias instead of just a single certificate.	<code>False</code>

Parameter	Data Type	Description	Default Value
actor	String	Java Type: String. Indicates the value of the SOAP <code>actor</code> attribute if a new security header is being added to the SOAP request. If the request already has a security header with this attribute specified in it, then this value will not overwrite it.	""
mustUnderstand	Boolean	Java Type: Boolean. Specifies whether the security header will have the <code>mustUnderstand</code> attribute set to 0 or 1 (false / true). If the security header already has this attribute set, then this value will not overwrite it.	False

pub.cloudstreams.security.ws.AddSamlSenderVouchesToken

This service enables a Security Token Service (STS) client to send a WS-Trust request to a configured STS to obtain a SAML v1/v2 assertion. For the details about configuring CloudStreams to act as an STS client, see ["Setting the STS Options" on page 36](#).

This service adds the obtained SAML assertion to the original request that is sent by the client to the native service, and includes the following parameters.

Parameter	Data Type	Description	Default Value
ConfigName	String	Required. Java Type: String. References a previously configured STS configuration name.	""
MessageContext	Object	Required. Java Type: <code>org.apache.axis2.context.MessageContext</code> . CloudStreams will place a <code>MessageContext</code> parameter	<code>org.apache.axis2.context.MessageContext</code> instance

Parameter	Data Type	Description	Default Value
		into the pipeline before executing the IS service call.	
<code>addTimeStamp</code>	Boolean	Java Type: Boolean. Adds a <code>Timestamp</code> element (with the duration specified in <code>timeToLive</code>) to the WS-Security header of the request, and includes it in the signature. (The other items that are signed are the body and SAML assertion.)	False
<code>timeToLive</code>	Integer	Java Type: Integer. If <code>addTimeStamp</code> is true, then <code>timeToLive</code> specifies the duration (in seconds) for which the request is valid.	300 seconds (5 minutes)
<code>actor</code>	String	Java Type: String. Indicates the value of the SOAP <code>actor</code> attribute if a new security header is being added to the SOAP request. If the request already has a security header with this attribute specified in it, then this value will not overwrite it.	""
<code>mustUnderstand</code>	Boolean	Java Type: Boolean. Specifies whether the security header will have the <code>mustUnderstand</code> attribute set to 0 or 1 (false / true). If the security header already has this attribute set, then this value will not overwrite it.	False

pub.cloudstreams.security.ws.AddTimestamp

Adds a timestamp to the outbound SOAP request WS-Security header. This service includes the following input parameters:

Parameter	Data Type	Description	Default Value
timeToLive	Integer	Java Type: Integer. Specifies the duration (in seconds) for which the request is valid.	300 seconds (5 minutes)
signTimestamp	Boolean	Java Type: Boolean. Indicates whether the generated timestamp must be signed by CloudStreams using the configured keystore and signing alias. Note: <code>signTimestamp</code> to work, you must ensure that a valid IS keystore and signing alias are configured in CloudStreams. For details, see " Setting the General Options " on page 28.	False
useMillisecondPrecision	Boolean	Java Type: Boolean. Indicates whether the generated timestamp must have millisecond precision.	True
MessageContext	Object	Required. Java Type: <code>org.apache.axis2.context.MessageContext</code> . CloudStreams will place a <code>MessageContext</code> parameter into the pipeline before executing the IS service call.	<code>org.apache.axis2.context.MessageContext</code> instance
actor	String	Java Type: String. Indicates the value	""

Parameter	Data Type	Description	Default Value
		of the SOAP <code>actor</code> attribute if a new security header is being added to the SOAP request. If the request already has a security header with this attribute specified in it, then this value will not overwrite it.	
<code>mustUnderstand</code>	Boolean	Java Type: Boolean. Specifies whether the security header will have the <code>mustUnderstand</code> attribute set to 0 or 1 (false / true). If the security header already has this attribute set, then this value will not overwrite it.	False

pub.cloudstreams.security.ws.AddWSAddressingHeaders

Adds WS-Addressing headers to a SOAP request sent by the client before CloudStreams forwards the request to the native service. This service includes the following input parameters:

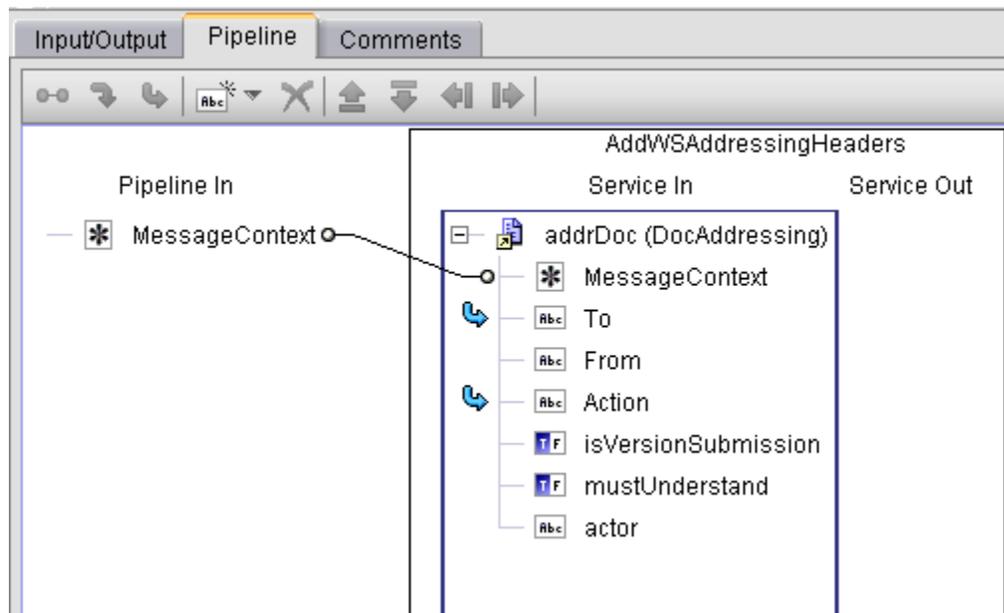
Parameter	Data Type	Description	Default Value
<code>isVersionSubmission</code>	Boolean	Java Type: Boolean. The WS-Addressing version that should be used. <ul style="list-style-type: none"> ■ If true, the WS-Addressing submission namespace <code>http://schemas.xmlsoap.org/ws/2004/08/addressing</code> will be used. 	False

Parameter	Data Type	Description	Default Value
		<ul style="list-style-type: none"> ■ If false, the Final specification namespace <code>http://www.w3.org/2005/08/addressing</code> will be used. 	
To	String	<p>Java Type: String. This value corresponds to the <code>/wsa:To</code> addressing header. You must specify a value that corresponds to the destination of the request message. If this value is not specified, then depending on the <code>isVersionSubmission</code> parameter value, one of the following anonymous EPR values will be sent:</p> <p>If <code>isVersionSubmission</code> is set to true, the anonymous EPR value is: <code>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</code>.</p> <p>If <code>isVersionSubmission</code> is set to false, the anonymous EPR value is: <code>http://www.w3.org/2005/08/addressing/anonymous</code>.</p>	
From	String	Java Type: String. This value corresponds to the <code>/wsa:From</code> addressing header and refers to the source of the message.	""
Action	String	Java Type: String. This value corresponds to the <code>/wsa:Action</code> addressing header. By default, this	URI identifying input operation corresponding to a WSDL

Parameter	Data Type	Description	Default Value
		property has the same value as the operation on the virtual service being invoked (which will usually correspond to the same operation on the native service). But you can specify a different value corresponding to the native service being called.	port type being called on the virtual service.
MessageContext	Object	Required. Java Type: <code>org.apache.axis2.context.MessageContext</code> . CloudStreams will place a <code>MessageContext</code> parameter into the pipeline before executing the IS service call.	<code>org.apache.axis2.context.MessageContext</code> instance
actor	String	Java Type: String. Indicates the value of the SOAP <code>actor</code> attribute if a new security header is being added to the SOAP request. If the request already has a security header with this attribute specified in it, then this value will not overwrite it.	""
mustUnderstand	Boolean	Java Type: Boolean. Specifies whether the security header will have the <code>mustUnderstand</code> attribute set to 0 or 1 (false / true). If the security header already has this attribute set, then this value will not overwrite it.	False

Example of using AddWSAddressingHeaders:

The sample service shown below is configured by providing the `MessageContext` parameter.



Using Context Variables in IS Services

CloudStreams provides predefined context variables, and you can declare your own custom context variables. You can use both predefined and custom context variables when you configure various processing steps of a virtual service. Specifically, you can use them in:

- An IS service that you create and invoke in the "In Sequence" and "Out Sequence" steps.
- In a routing rule that you create in a "Routing Rule" step of the type "Context-Based".

This section discusses:

- ["The Predefined Context Variables" on page 100.](#)
- Custom context variables (see ["The API for Context Variables" on page 105.](#))

The Predefined Context Variables

You can use the predefined context variables listed below. Any context variable state defined during the inbound request processing steps will still be available during the outbound response processing steps.

Note: To set, get or remove the predefined context variables, use the API for context variables provided by CloudStreams (see ["The API for Context Variables" on page 105.](#))

Note: You do not need to declare the predefined context variables. If you attempt to declare an existing predefined context variable, an error will occur.

Context Variable Display Name	Context Variable Name	Description
Average Response	AVG_SUCCESS_TIME	The average amount of time it took the service to complete all invocations in the current interval. Response time is measured from the moment CloudStreams receives the request until the moment it returns the response to the caller.
Client IP Address	INBOUND_IP	The IP address used to send the request to CloudStreams.
CloudStreams Host Name	HOSTNAME	CloudStreams host name.
CloudStreams IP Address	SERVER_IP	CloudStreams IP address.
CloudStreams Target Name	TARGET_NAME	CloudStreams target name.
Consumer	CONSUMER_APPLICATION	The name of the consumer application accessing the service, if known.
Fault Count	INTERVAL_FAULT_COUNT	The number of invocations of the service resulting in a fault in the current interval.
Inbound Content Type	MESSAGE_TYPE	A Content-Type defined in axis2.xml for a message formatter. This value must be a key in the axis2 message formatters list, since it is used to control message serialization. (The valid choices are defined as attributes of <messageFormatters/> group in Integration Server's axis2.xml configuration.)

Context Variable Display Name	Context Variable Name	Description
Inbound HTTP Method	INBOUND_HTTP_METHOD	The HTTP method used by the client to send the request (GET, POST, PUT, DELETE, CUSTOM).
Inbound Protocol	INBOUND_PROTOCOL	The protocol (HTTP or HTTPS) of the request.
Maximum Response	SLOWEST_SUCCESS_INVOKE	The maximum amount of time it should take for the service to complete an invocation. Response time is measured from the moment CloudStreams receives the request until the moment it returns the response to the caller.
Minimum Response	FASTEST_SUCCESS_INVOKE	The minimum amount of time it should take for the service to complete an invocation.
Outbound HTTP Method	ROUTING_METHOD	The HTTP method to be sent to the native service if the inbound HTTP method is custom. Otherwise, this value will be null. For more information, see "Changing the HTTP Method of a REST Virtual Service" on page 162 .
Success Count	INTERVAL_SUCCESS_COUNT	The number of successful invocations of the service in the current interval.
Total Count	INTERVAL_TOTAL_COUNT	The total number of invocations (successful or unsuccessful) of the service in the current interval.
Virtual Service Name	SERVICE_NAME	Virtual service name.
(not displayed)	BUILDER_TYPE	A Content Type defined in axis2.xml for a message builder. This value must be a key in the axis2 message builders list, since it is used to control building of native service

Context Variable Display Name	Context Variable Name	Description
		<p>response messages. (The valid choices are defined as attributes of <code><messageBuilders/></code> group in Integration Server's <code>axis2.xml</code> configuration.)</p>
(not displayed)	INBOUND_REQUEST_URI	<p>A partial reference to a virtual service (for HTTP/HTTPS only). The protocol, host and port are not part of the value. For example, if the following virtual service is invoked:</p> <pre>http://user:5555/ws/TC1</pre> <p>then the expected value of this variable would be <code>/ws/TC1</code>.</p> <p>For a REST or XML service, the URL might also include query string parameters. For example, if the following virtual service is invoked:</p> <pre>http://user:5555/ws/cars?vin=1234</pre> <p>then the expected value of this variable would be <code>/ws/cars?vin1234</code>.</p> <p>This is useful to know because by the time you are able to access the request inside of CloudStreams, the REST request would contain a top-level element that looks like this:</p> <pre><vin>1234</vin></pre> <p>So it is not obvious from an XSLT expression or an IS service callout what part of a REST request came in as a query parameter. Therefore, using this variable along with <code>INBOUND_HTTP_METHOD</code> and <code>INBOUND_PROTOCOL</code>, you can determine the exact entry point URI that was used when a CloudStreams virtual service was invoked.</p>

Context Variable Display Name	Context Variable Name	Description
(not displayed)	NATIVE_PROVIDER_ERROR	The "reason" returned by the native service provider in the case where it produced a SOAP fault. This variable will not contain CloudStreams-hosted errors such as policy violation errors; it will only contain the "reason" text wrapped in a SOAP fault.
(not displayed)	OPERATION	The service operation selected to execute a request.
(not displayed)	PROTOCOL_HEADERS	Contains a map of key-value pairs in the request, where the values are typed as strings. To get/set this variable, use <code>pub.cloudstreams.ctxvar.getContextVariable</code> (see " The API for Context Variables " on page 105).
(not displayed)	SOAP_HEADERS	(For use in IS services only.) Contains an array of the SOAP header elements in the request. To get/set this variable, use <code>pub.cloudstreams.ctxvar.getContextVariable</code> (see " The API for Context Variables " on page 105).
(not displayed)	USER	The value defined for the Integration Server session executing the request message. If the request is not authenticated, it will use a default unprivileged account. Otherwise, it will set the Integration Server session to the user credentials used for transport security. In the case where a consumer is sending a request with both transport credentials (HTTP Basic authentication) and message credentials (WSS Username or X.509 token), the message credentials take precedence over the transport credentials.

The API for Context Variables

CloudStreams provides an API that you can use to:

- Set, get, declare and remove custom context variables.
- Set and get the predefined context variables. (It is not necessary (or even legal) to declare or remove the predefined context variables.)

CloudStreams provides the following Java services (described below):

- `pub.cloudstreams.ctxvar:getContextVariable`
- `pub.cloudstreams.ctxvar:setContextVariable`
- `pub.cloudstreams.ctxvar:declareContextVariable`
- `pub.cloudstreams.ctxvar:removeContextVariable`

The following sample flow services are described in this section as well:

- Sample Flow Service: Getting a Context Variable Value
- Sample Flow Service: Setting a Context Variable Value

`pub.cloudstreams.ctxvar:getContextVariable`

Use this Java service to retrieve a context variable's value and assign it to a pipeline variable. All parameter names are case-sensitive.

Parameter	Pipeline Type	Data Type	Description
<code>MessageContext</code>	in	Object ref	This object is inserted into the pipeline by CloudStreams.
<code>varName</code>	in	String	Context variable name (predefined or custom). Examples: <code>PROTOCOL_HEADERS</code> , <code>SOAP_HEADERS</code> , <code>mx:CUSTOM_VAR</code>
<code>serValue</code>	out	Object ref	<code>java.io.Serializable</code> value. (Usually a string).

Notes on Getting and Setting the `PROTOCOL_HEADERS` and `SOAP_HEADERS` Variables

All context variable values are typed as either "string" or "int" except for the predefined context variables `PROTOCOL_HEADERS` and `SOAP_HEADERS`, which are of the type "IData". You can set/get values for `PROTOCOL_HEADERS` and `SOAP_HEADERS` in one of two ways:

1. Set/get the entire structure.

To set the entire structure, you must:

- Set the `varName` parameter in `pub.cloudstreams.ctxvar:setContextVariable` to `PROTOCOL_HEADERS` or `SOAP_HEADERS`.
- Use the method `RuntimeFacade.setContextVariableValue()`.

To get the entire structure, you must:

- Set the `varName` parameter in `pub.cloudstreams.ctxvar:getContextVariable` to `PROTOCOL_HEADERS` or `SOAP_HEADERS`.
- Use the method `RuntimeFacade.getContextVariableValue()`.

If `varName` is set to `PROTOCOL_HEADERS`, you will get/set the entire `IData` structure containing all of the transport headers. The key is the transport header name (for example, `Content-Type`) and the value is a `String`. The `IData` object for `PROTOCOL_HEADERS` will contain a set of string values where each `IData` string key matches the header name in the transport headers map. The set of possible keys includes the HTTP v1.1 set of headers as well as any custom key-value pairs you might have defined.

If `varName` is set to `SOAP_HEADERS`, you will get/set the entire `IData` structure containing all of the SOAP headers in the SOAP envelope. The key is the array position starting with "0", and the value is an `Axiom OMElement` containing that SOAP header block.

Alternatively, you can set the `varName` parameter to address a specific element in the array. For example, setting it to `PROTOCOL_HEADERS[Content-Type]` would apply to the `Content-Type` transport header. Similarly, setting it to `SOAP_HEADERS[0]` would return a `String` representation of the first SOAP header block (as opposed to an `Axiom OMElement`).

2. Set/get a nested value.

Set a nested value in one of the following ways:

- Set the `varName` parameter in `pub.cloudstreams.ctxvar:setContextVariable` to `PROTOCOL_HEADERS[arrayElement]`, where *arrayElement* refers to a specific element. For example, `PROTOCOL_HEADERS[Content-Type]` or `SOAP_HEADERS[0]` (to indicate the first array element in the set).
- Alternatively, use the method `RuntimeFacade.setContextVariableValue()`. You would use this method only if you are writing a Java service and you want to access it through the Java source code.

Get a nested value in one of the following ways:

- Set the `varName` parameter in `pub.cloudstreams.ctxvar:getContextVariable` to `PROTOCOL_HEADERS[arrayElement]`, where *arrayElement* refers to a specific element. For example, `PROTOCOL_HEADERS[Content-Type]` or `SOAP_HEADERS[0]` (to indicate the first array element in the set).
- Alternatively, use the method `RuntimeFacade.getContextVariableValue()`. You would use this method only if you are writing a Java service and you want to access it through the Java source code.

You can set/get a nested value inside `PROTOCOL_HEADERS` and `SOAP_HEADERS` via an additional `keyName`. In this case, the object reference will not be an `IData` object.

- For `PROTOCOL_HEADERS`, the `keyName` must match the transport header name in a case-sensitive manner (for example, `PROTOCOL_HEADERS[Content-Type]` or `PROTOCOL_HEADERS[Authorization]`). In this case, the `Serializable` value will be a string.
- For `SOAP_HEADERS`, the `keyName` must match the 0-based array element. If a request has a SOAP security header element (`</wsse:Security>`), then it would be addressed as `SOAP_HEADERS[0]`. In this case, the element will be in its string format.

`pub.cloudstreams.ctxvar:setContextVariable`

Use this Java service to set a value on a context variable. The pipeline variable containing the context variable value should be an object reference that implements `java.io.Serializable`. All parameter names are case-sensitive.

Parameter	Pipeline Type	Data Type	Description
<code>MessageContext</code>	in	Object ref	This object is inserted into the pipeline by CloudStreams.
<code>varName</code>	in	String	Context variable name (predefined or custom). Examples: <code>PROTOCOL_HEADERS</code> , <code>SOAP_HEADERS</code> , <code>mx:CUSTOM_VAR</code>
<code>serValue</code>	in	Object ref	<code>Java.io.Serializable</code> value. (Usually a string.)

`pub.cloudstreams.ctxvar:declareContextVariable`

Use this Java service to declare a *custom* context variable. All custom-defined context variables must be declared in a custom namespace that is identified by using the prefix `mx` (for example, `mx:CUSTOM_VARIABLE`). All parameter names are case-sensitive.

Note: It is not legal to use this service to declare the predefined context variables; you can only declare custom variables.

Parameter	Pipeline Type	Data Type	Description
<code>ctxVr</code>	in	Object ref	The document type defining the context variable object. Use the <code>ctxVar</code> Document Type provided in the Java

Parameter	Pipeline Type	Data Type	Description
			service pub.cloudstreams.ctxvar:ctxVar and map it to this input variable. Define the name of the context variable (for example, mx:CUSTOM_VARIABLE), the schema_type (string or int), and isReadOnly (true or false).
ctxVar	out	Object ref	The set Context variable document type.
varNameQ	out	Object ref	javax.xml.namespace.QName value. The QName of the variable.

Note the following:

- After declaring the context variable, you can use the setContext variable to set a value on the context variable.
- You do *not* need to declare the following kinds of context variables:
 - The predefined variables provided by CloudStreams. If you attempt to declare an existing predefined context variable, an error will occur.
 - Any custom context variable that you define in a routing rule that you create in a "Context-Based" Routing Rule step.
- Any custom context variables that you explicitly declare in source code using the API will have a declaration scope of SESSION.
- Any custom context variable's state that is defined during the inbound request processing steps will still be available during the outbound response processing steps.
- All context variable values are typed as either "string" or "int" (excluding the SOAP_HEADERS and PROTOCOL_HEADERS variables, which are of the type "IData").
- Valid names should be upper case (by convention) and must be a valid Java Identifier. In general, use alpha-numerics, \$ or _ symbols to construct these context names. Names with punctuation, whitespace or other characters will be considered invalid and will fail deployment.
- All custom context variables must be declared in a custom namespace that is identified by using an mx prefix (for example, mx:CUSTOM_VARIABLE).
- To reference a custom context variable in a flat string, you need to prepend a \$ symbol to the context variable name to indicate that variable's value should be referenced. Think of this usage as being similar to the "&" address operation for C variables.

An expression that references a custom context variable might look like this:

```
$mx:TAXID=1234 or $mx:ORDER_SYSTEM_NAME="Pluto"
```

Notice that the values of the data type `int` are not enclosed in quotation marks, while the values of the data type `string` are. The quotation marks are only needed if a context variable expression (as opposed to a reference) is defined.

- Referencing an undefined context variable does not result in an error.
- Once a variable has been declared it cannot be declared again.

pub.cloudstreams.ctxvar:removeContextVariable

Use this Java service to remove a *custom* context variable from a request/response session. All parameter names are case-sensitive.

Note: It is not legal to use this service to remove any predefined context variables; you can only remove custom variables.

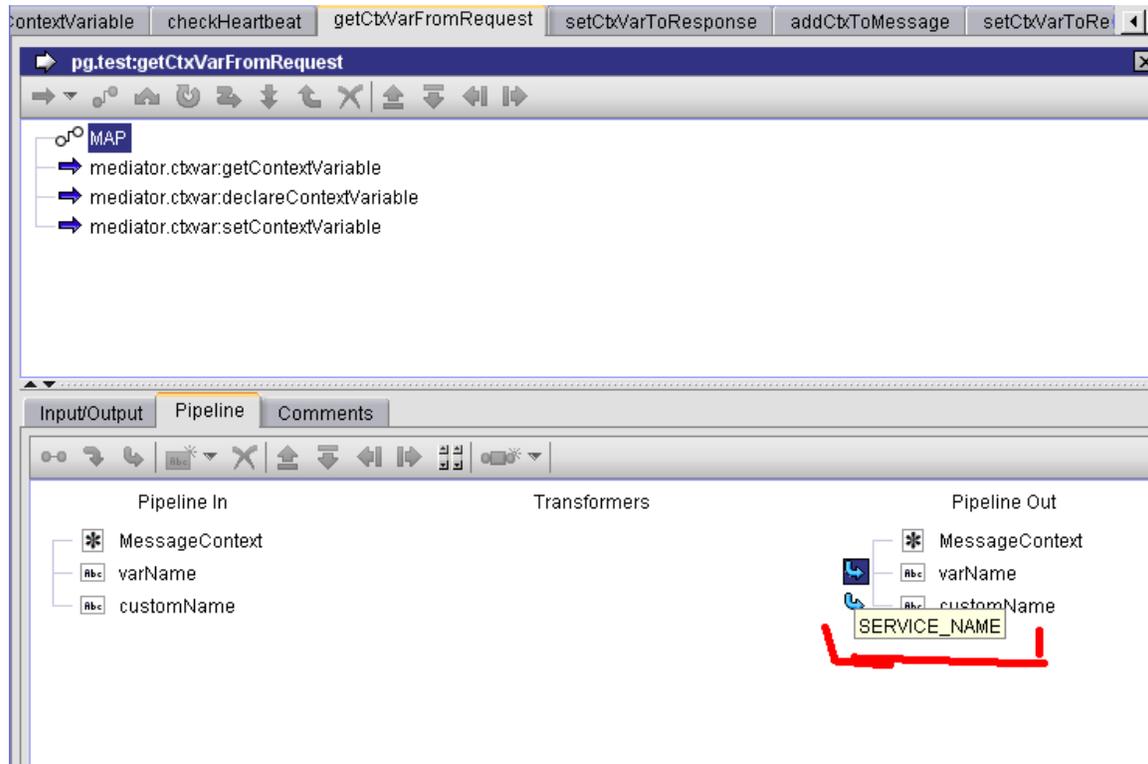
Note: Attempting to remove a non-existent context variable will not result in an error.

Parameter	Pipeline Type	Data Type	Description
MessageContext	in	Object ref	This object is inserted into the pipeline by CloudStreams.
varName	in	String	Custom context variable name. Example: <code>mx:CUSTOM_VAR</code> .

Sample Flow Service: Getting a Context Variable Value

This flow service gets the value of a custom context variable from a request. The service declares a context variable using the name defined in the pipeline variable `customName` (that is, `mx:COMP_TEST`). It is hard-coded to store the predefined context variable `SERVICE_NAME` in this custom context variable name.

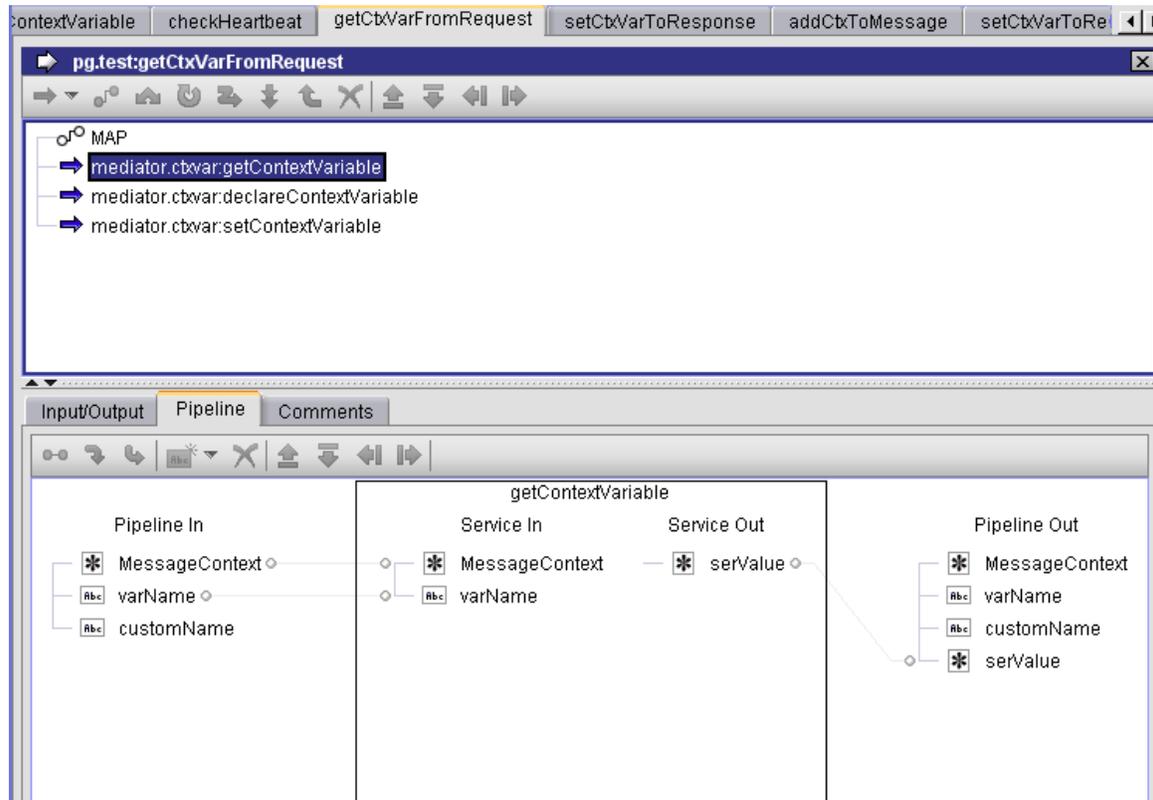
Step 1: Setting varName and customName



In this flow service, the following pipeline variables that are hard-coded as follows:

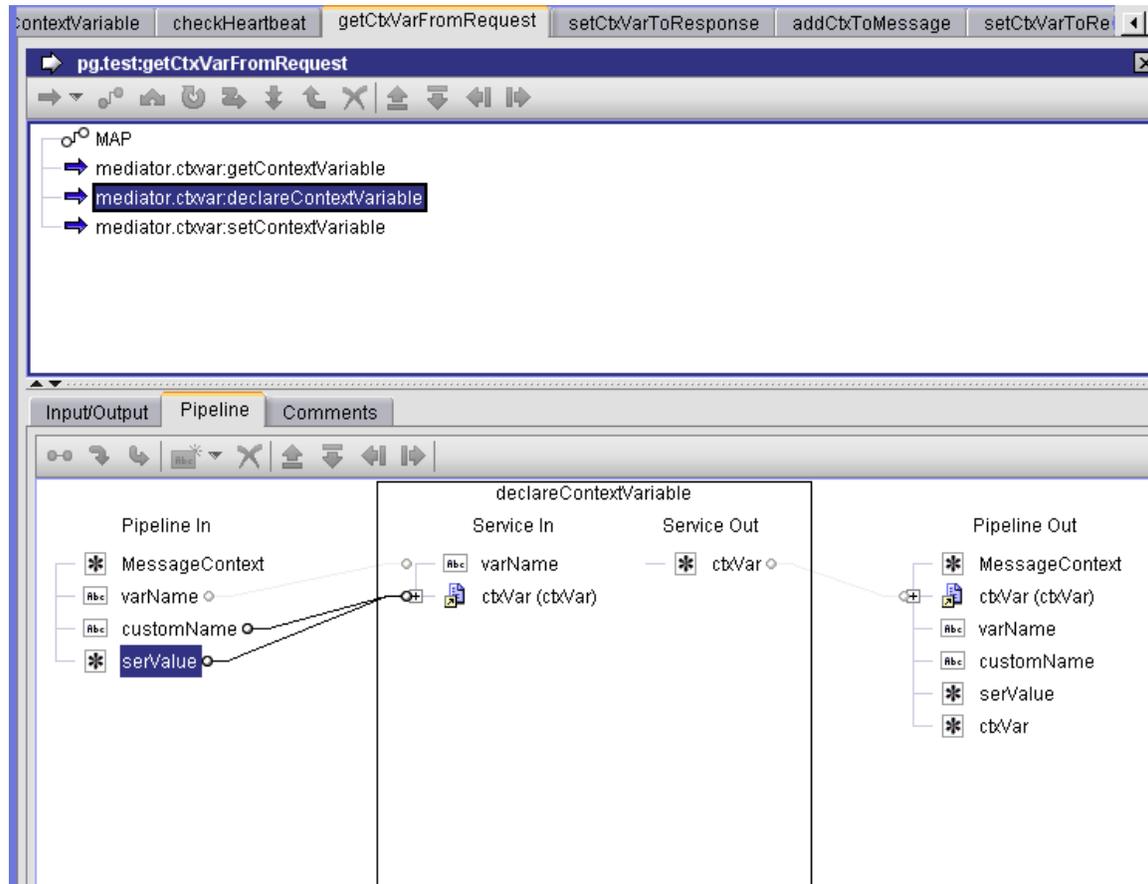
- varName is set to the predefined context variable SERVICE_NAME.
- customName is set to mx:COMP_TEST, and SERVICE_NAME is stored in customName in the "Pipeline Out".

Step 2: Calling the `pub.cloudstreams.ctxvar:getContextVariable` service



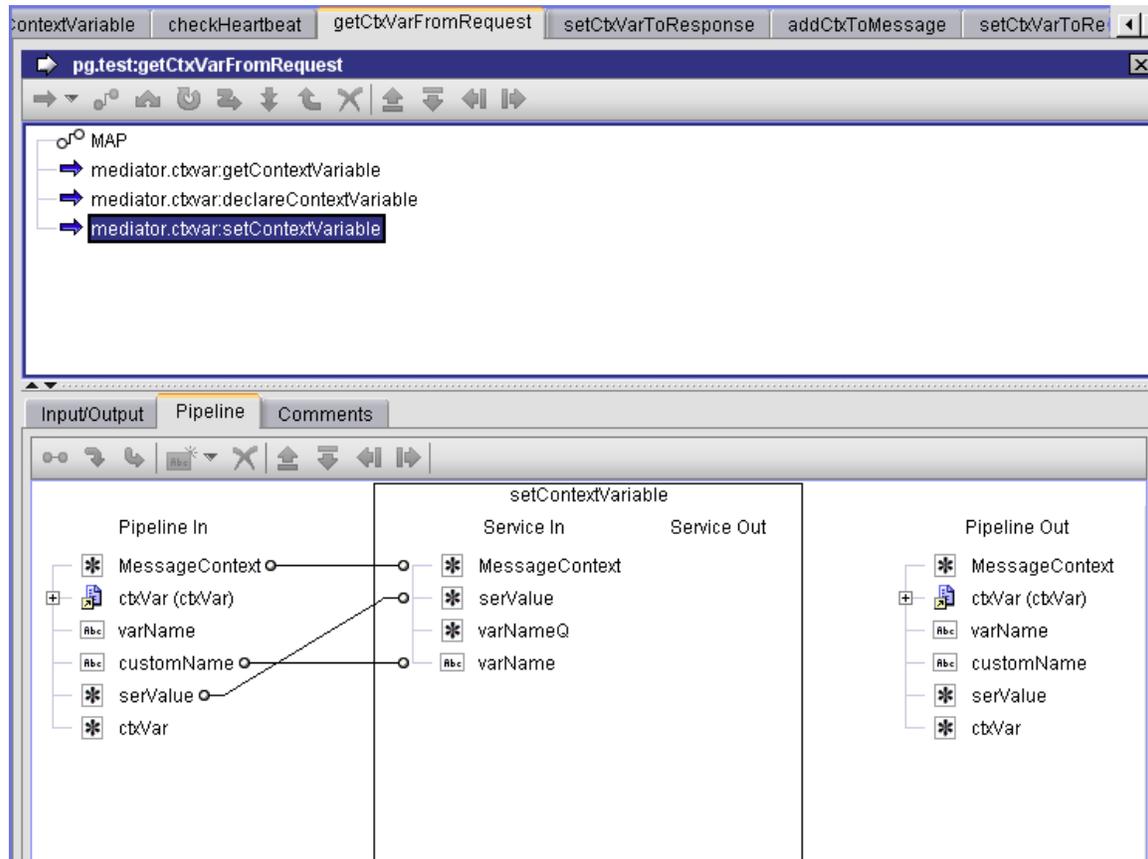
The call to `pub.cloudstreams.ctxvar:getContextVariable` looks up the context variable value for the "Pipeline In" variable `varName` and sets the `Serializable` value in the "Pipeline Out" variable `serValue`.

Step 3: Calling the `pub.cloudstreams.ctxvar:declareContextVariable` service



The call to `pub.cloudstreams.ctxvar:declareContextVariable` takes the serialized value we looked up for `varName` and assigns it to a newly declared custom context variable named `ctxVar`.

Step 4: Calling the `pub.cloudstreams.ctxvar:setContextVariable` service



The call to `pub.cloudstreams.ctxvar:setContextVariable` sets the context variable value back into `MessageContext` so it will be available for the rest of the service invocation steps.

Sample Flow Service: Setting a Context Variable Value

This flow service sets the value of a custom context variable to be used in a response.

This flow service declares a pipeline variable named `customName`, which is set to the value `mx:COMP_TEST`.

This flow service will retrieve the context variable for `customName` and create an element for its context variable value in the response message return to the consumer.

Step 1: Declaring customName

The screenshot displays the webMethods CloudStreams IDE interface. At the top, a breadcrumb trail shows the pipeline steps: contextVariable, checkHeartbeat, getCtxVarFromRequest, setCtxVarToResponse, and addCtxToMessage. The main window is titled 'pg.test:setCtxVarToResponse' and shows a sequence of steps: a SEQUENCE step, a MAP step, a mediator step 'mediator.ctxvar:getContextVariable', and a step 'pg.test:addCtxToMessage'. Below this, the 'Input/Output' tab is active, showing the configuration for the selected step. It includes fields for 'Specification Reference', 'Input', and 'Output'. There are checkboxes for 'Validate input' and 'Validate output'. The 'Input' list contains: proxy.name, SOAPEnvelope, MessageContext, EnvelopeString, varName, customName, and UpdatedSoapRequest. The 'Output' list contains: UpdatedSoapRequest, EnvelopeString, customName, and UpdatedSoapRequest.

We define the `customName` variable to be `mx:COMP_TEST` so we can use this variable to lookup the custom variable name that was seeded in the previous example.

Step 2: Setting customName to mx:COMP_TEST

The screenshot displays the webMethods CloudStreams interface. At the top, a tab bar shows several pipeline steps: contextVariable, checkHeartbeat, getCtxVarFromRequest, setCtxVarToResponse (selected), and addCtxToMessage. Below the tabs, a window titled 'pg.test:setCtxVarToResponse' is open, showing a tree view of the pipeline configuration. The tree includes a SEQUENCE node, a MAP node, and two child nodes: mediator.ctxvar.getContextVariable and pg.test.addCtxToMessage.

Below this window, the 'Pipeline' tab is active, showing a comparison between 'Pipeline In' and 'Pipeline Out' variables. The 'Pipeline In' variables are: proxy.name, SOAPEnvelope, MessageContext, EnvelopeString, varName, customName, and UpdatedSoapRequest. The 'Pipeline Out' variables are: proxy.name, SOAPEnvelope, MessageContext, EnvelopeString, varName, customName, and UpdatedSoapRequest. The 'UpdatedSoapRequest' variable in the 'Pipeline Out' column is highlighted with a blue selection box. Blue arrows indicate the mapping of variables between the input and output.

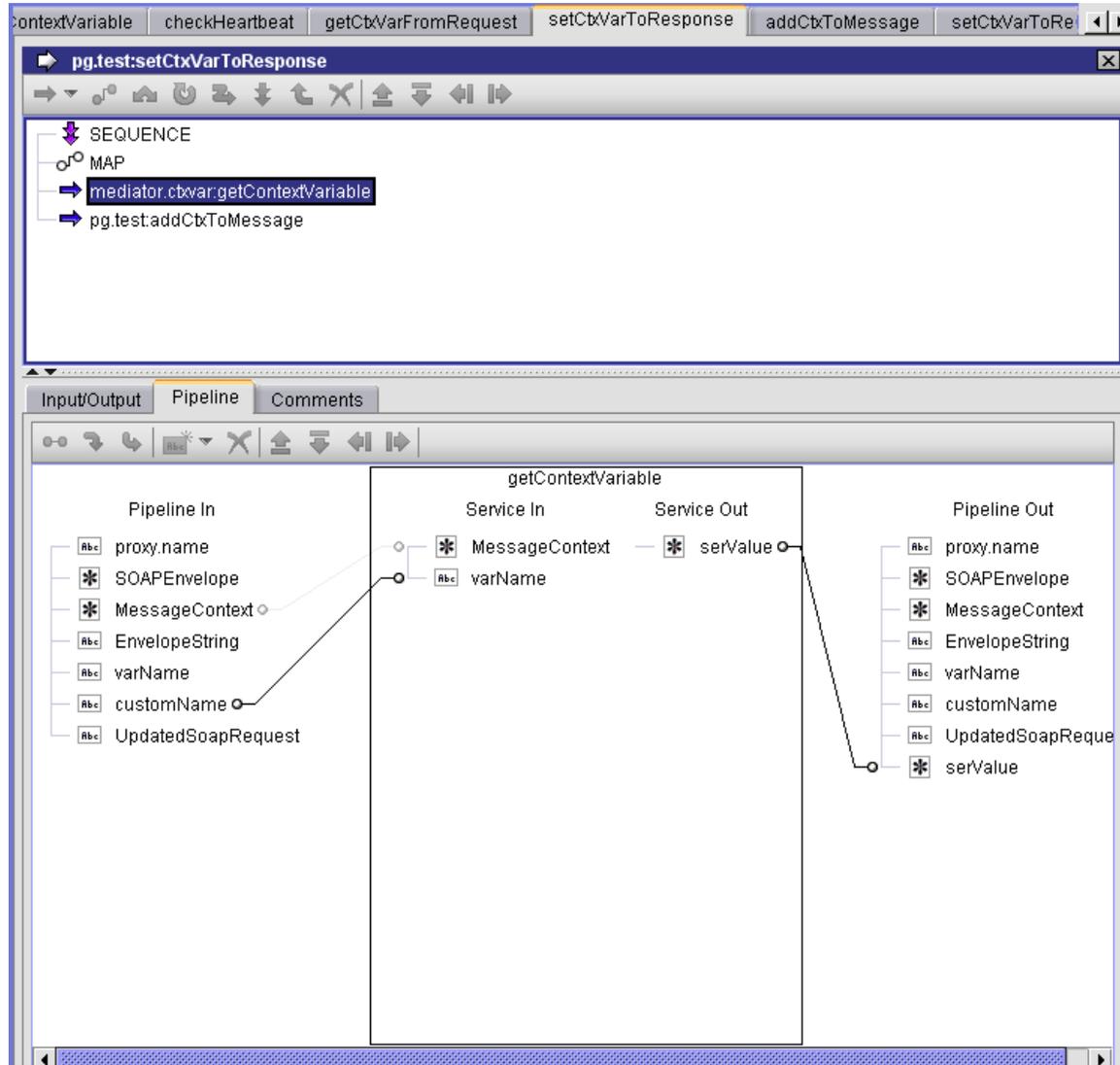
Clicking on the `customName` pipeline variable will display the name.

Step 3: Displaying the value of customName

The screenshot displays the CloudStreams IDE interface. At the top, a tab bar shows several pipeline steps: `ontextVariable`, `checkHeartbeat`, `getCtxVarFromRequest`, `setCtxVarToResponse` (selected), `addCtxToMessage`, and `setCtxVarToRe`. Below the tab bar, the main workspace shows a pipeline diagram for `pg.test:setCtxVarToResponse`. The diagram consists of a `SEQUENCE` container, followed by a `MAP` component, and then two actions: `mediator.ctxvar:getContextVariable` and `pg.test:addCtxToMessage`. An `Input/Output` window is open, titled "Input for 'customName'", with the text `mx:COMP_TEST` entered in the `customName` field. Below the text field are two checkboxes: `Overwrite pipeline value` (checked) and `Perform variable substitution` (unchecked). The `OK`, `Cancel`, and `Help` buttons are visible at the bottom of the dialog. To the right, the `Pipeline Out` window shows the output structure, including `proxy.name`, `SOAPEnvelope`, `MessageContext`, `EnvelopeString`, `varName`, `customName` (highlighted in blue), and `UpdatedSoapRequest`.

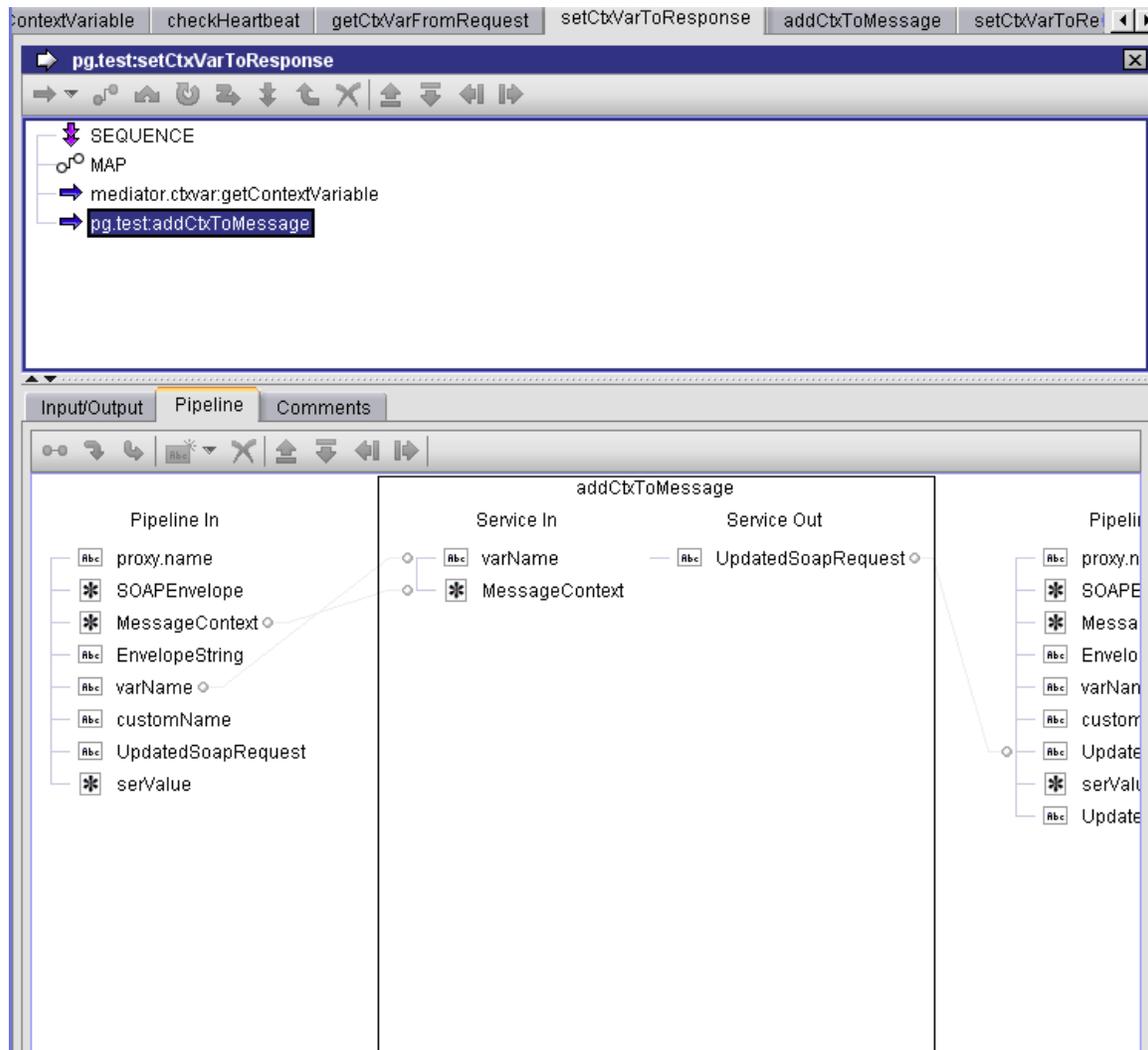
The call to `pub.cloudstreams.ctxvar:getContextVariable` retrieves the value of the custom context variable from the context variable map.

Step 4: Calling `pub.cloudstreams.ctxvar:getContextVariable`



This is just a sample Java service that takes the context variable and creates a top-level element in the response message using the same name and value.

Step 5: Sample service using the context variable



Creating a New Connector Virtual Service (SOAP)

A connector virtual service will:

- Handle the provider's response.
- Log the requests/responses.

CloudStreams provides a default connector virtual service for each metadata handler:

- The service WmCloudStreams.SoapVS (for the SOAP handler).
- The service WmCloudStreams.RestVS (for the REST handler).

The default services are located in the sample CloudStreams Governance project in the WmCloudStreams package. You cannot modify them.

Each default connector virtual service has a default policy (named Logging Policy), which logs all requests/responses and their payloads to a database. You cannot modify the default policy. Alternatively, you can create additional connector virtual services with custom policies.

If you create a connector virtual service with a custom policy, you can only include the actions in the "Monitoring" or "Additional" action categories; you cannot include the "WS-SecurityPolicy 1.2" actions. For example, you might want to create a custom policy that monitors run-time performance, customizes how the service invocations are logged, validates response messages against an XML schema, or optimizes server traffic. For complete details, see the chapter "[Policies](#)" on page 167.

To create a new connector virtual service (SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, right-click the CloudStreams Governance project and click **New > Connector Virtual Service** (or expand the project, right-click the **Connector Virtual Services** folder and click **New Connector Virtual Service**).
3. Complete the following fields in the New Connector Virtual Service wizard and click **Finish**.

Option	Description
CloudStreams Governance Project	Click Browse and select a CloudStreams Governance project in which to create the connector virtual service.
Name	Assign a name for the service. Unlike native services, the names of virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the virtual service name, then the names of the services themselves must not contain characters that are invalid in virtual service names.
	Note: If you want to change the service name after it has been created, right-click the service name in the Virtual Services folder and select Rename .
Version	The version is always set to 1.0.
Type	Select SOAP .

Option	Description
Description	Optional. A description for the service. This description appears when a user displays a list of services in the user interface.

The new connector virtual service is added to the CloudStreams Governance project, in the **Connector Virtual Services** folder.

The Properties of a Connector Virtual Service (SOAP)

To view the properties of a connector virtual service (SOAP)

1. OpenSoftware AG Designer and display theCloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In theCloudStreams Governance view, expand yourCloudStreams Governance project and click the virtual service name. (If the Properties view is not already open, click **Window > Show View > Other > General > Properties**.)
3. The **General** page in the Properties view displays the following properties:

Option	Description
Name	(Read-only field.) The service name. You can change the service name at any time by right-clicking the name in the Connector Virtual Services folder and clicking Rename .
Service Type	(Read-only field.) SOAP .
Created/Last Modified	(Read-only field.) The service's creation/modification timestamps.
Target Namespace	(Read-only field.) The value derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
Namespaces	Click the button next to this field to view the service's available namespaces (such as <code>wSDL</code> , <code>xsd</code> , <code>soap</code> , etc.).
Version	The version is always set to 1.0.
WSDL URL	(Read-only field.) The URL of the service's WSDL.
Description	You can change the service description.

4. View additional properties in the **Advanced** page.

Option	Description
Name	(Read-only field.) The service name.
Type	(Read-only field.) The type of the service (SOAP).
Target Namespace	(Read-only field.) The value derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
WSDL URL	Click this URL to display the contents of the service's abstract WSDL. This is an internal WSDL used by CloudStreams primarily to manage deployment of connector virtual services to a CloudStreams target. The native provider's WSDL that was defined for the connector is what is used when the connector service is configured.
Namespaces	Click the button next to this field to view the service's available namespaces (such as <code>wSDL</code> , <code>xsd</code> , <code>soap</code> , etc.).
Version	(Read-only field.)
Description	(Read-only field.) The service description.
VSD	<p>Click the button next to this field to view the service's "virtual service definition" (VSD). This button is disabled until you deploy the connector virtual service to a CloudStreams server.</p> <p>When you deploy the connector virtual service to a CloudStreams server, CloudStreams generates an XML document called a <i>virtual service definition (VSD)</i>. The VSD defines the connector virtual service for CloudStreams, and contains all the resources required to deploy the connector virtual service to a CloudStreams server, including the policy that applies to the service. You cannot edit the VSD.</p> <p>Note: If multiple policies apply to the service, CloudStreams combines all those policies into a single policy known as the <i>effective policy</i>. The effective policy is a simple UNION of the run-time actions specified in all policies that apply to a service. To create the effective policy, CloudStreams evaluates the combined list of actions from all policies, using a set of internal rules known as Policy Conflict Resolution rules. For details, see "Policy Conflict Resolution Rules" on page 203.</p>

Option	Description
Applicable Policies	Click the button next to this field to view a list of the active policies that apply to this service. Any inactive policy that applies to the service is not listed. To change the list of applicable policies, see "Modifying Policies" on page 172 .
Endpoint	(Read-only field). The endpoint is resolved when the connector service is configured and the user selects an enabled managed connection pool.
Deployed Status	(Read-only field.) Indicates whether the service is Deployed , Undeployed or Not Deployed (which is the initial status before you deploy the service). To deploy or undeploy a service, see "Deploying Virtual Services and Connector Virtual Services" on page 199 .
Resource	Not applicable to SOAP services.

Managing a Connector Virtual Service (SOAP)

You can rename, delete, deploy or undeploy a connector virtual service, and change the list of policies that apply to the it.

To manage a connector virtual service (SOAP)

1. OpenSoftware AG Designer and display theCloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In theCloudStreams Governance view, expand yourCloudStreams Governance project.
3. Right-click the connector virtual service name and choose any of the following tasks from the context menu:

Option	Description
Rename	<p>Use this option to assign a new name to the service. The service must be undeployed before you can rename it (check the Deployed Status field in the Advanced page of the service's Properties view).</p> <p>Unlike native services, the names of connector virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the connector virtual service name, then the names of</p>

Option	Description
	the services themselves must not contain characters that are invalid in connector virtual service names.
Delete	Deletes the connector virtual service from CloudStreams.
Deploy	Deploys the connector virtual service to a CloudStreams server target. For more information, see " Deploying Virtual Services and Connector Virtual Services " on page 199.
Undeploy	Undeploys the connector virtual service from a CloudStreams server target.

- To change the list of policies that apply to the connector virtual service, see "[Modifying Policies](#)" on page 172.

The "In Sequence" Step (Connector Virtual Service, SOAP)

The "In Sequence" step can contain the following sub-steps:

- The "Entry Step" (required), which specifies the protocol of the requests that the connector virtual service will accept.
- The "Routing Rule" step (required), which specifies how the connector virtual service will route the service requests to the native service endpoint.
- The "Invoke IS Service" step (optional), which pre-processes the request message before it is submitted to the native service.

The "Entry Step" (Connector Virtual Service, SOAP)

This step specifies the protocol of the requests that the connector virtual service will accept. You cannot modify this step, except for its name.

To edit the "Entry Step" (connector virtual service, SOAP)

- Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
- In the CloudStreams Governance view, expand your CloudStreams Governance project and click the connector virtual service name.

The virtual service editor displays the three main processing steps: **In Sequence**, **Out Sequence** and **Error Sequence**.

- Expand **In Sequence**.

By default, the **In Sequence** step contains the sub-steps **Entry Step** and **Routing Rule**. (You can add the optional **Invoke IS Service** step, as described later.)

- Click **Entry Step** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Entry Step .
Protocol	(Read-only field.) The protocol of the requests that the connector virtual service will accept. The value Local means that the service can only be called from Cloud Connector Services.

The "Routing Rule" Step (Connector Virtual Service, SOAP)

This step specifies how the connector virtual service will route the requests to the native service endpoint. You cannot modify this step, except for its name.

To edit the "Routing Rule" step (connector virtual service, SOAP)

- Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
- In the CloudStreams Governance view, expand your CloudStreams Governance project and click the connector virtual service name.
- Expand **In Sequence**.
- Click **Routing Rule** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) Protocol . You cannot change the protocol over which the connector virtual service will accept requests.
Routing Rule	(Read-only field.) The value Connection Pool means that the requests are sent to the provider via the CloudStreams connection pool. For more information about connection pools, see the documentation specific to your webMethods

Option	Description
	CloudStreams cloud provider (for example, <i>webMethods CloudStreams Provider for Salesforce.com Installation and User's Guide</i>).

The "Invoke IS Service" Step (Inbound, Connector Virtual Service, SOAP)

This optional step invokes a user-defined Integration Server flow service. An IS service is a user-defined Integration Server flow service that you can invoke in:

- The "In Sequence" step, to pre-process the request messages before CloudStreams submits the requests to the native service.
- The "Out Sequence" step, to pre-process the response messages from the native service before CloudStreams returns the responses to the consuming application.

You create an "Invoke IS Service" step for a connector virtual service the same way you create one for a virtual service. For details, see "[The "Invoke IS Service" Step \(Inbound, SOAP\)](#)" on page 69, or "[The "Invoke IS Service" Step \(Outbound, SOAP\)](#)" on page 84.

The "Out Sequence" Step (Connector Virtual Service, SOAP)

You configure the service's "Out Sequence" step (optional) to manipulate the response messages from the native service before CloudStreams returns the responses to the consuming applications. This step can include the "Invoke IS Service" step (optional), which pre-processes the response message from the native service provider before it is returned to the consuming application. See "[The "Invoke IS Service" Step \(Inbound, Connector Virtual Service, SOAP\)](#)" on page 125.

The "Error Sequence" Step (Connector Virtual Service, SOAP)

The default connector virtual services are configured to return the native service provider's service fault, if one is available. CloudStreams will send whatever content it received from the native service provider. You can optionally invoke IS services to pre-process or post-process the error messages.

To configure the "Error Sequence" step (connector virtual service, SOAP)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the connector virtual service name.
3. In the virtual service editor, expand **Error Sequence**.
4. Click the **Error Messaging** sub-step and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Error Messaging.
Processing Method	<p>Optionally select either of the following:</p> <ul style="list-style-type: none"> ■ Pre-Processing: Select this option if you want to invoke an IS service to manipulate the response message before the Error Sequence step is invoked. The IS service will have access to the response message context (the <code>axis2 MessageContext</code> instance) before it is updated with the custom error message. For example, you might want to send emails or perform custom alerts based on the response payload. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, SOAP)" on page 69. ■ Post-Processing: Select this option if you want to invoke an IS service to manipulate the service fault after the Error Sequence step is invoked. The IS service will have access to the entire service fault and the custom error message. You can make further changes to the fault message structure, if needed. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, SOAP)" on page 69.

Creating a New Virtual Service (REST)

To create a new virtual service (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, right-click the CloudStreams Governance project and click **New > Virtual Service** (or expand the project, right-click the **Virtual Services** folder and click **New Virtual Service**).
3. In the New Virtual Service wizard, complete the following fields.

Option	Description
CloudStreams Governance Project	Click Browse and select a CloudStreams Governance project in which to create the service.

Option	Description
Name	Assign a name for the service. Unlike native services, the names of virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the virtual service name, then the names of the services themselves must not contain characters that are invalid in virtual service names. Note: If you want to change the service name after it has been created, right-click the service name in the Virtual Services folder and select Rename .
Version	The version is always set to 1.0.
Service Type	Select REST .
Description	Optional. A description for the service. This description appears when a user displays a list of services in the user interface.

4. Click **Finish**.

The new virtual service is added to the CloudStreams Governance project, in the **Virtual Services** folder.

5. Click the new virtual service name in the CloudStreams Governance project.

The Properties view is displayed at the bottom of the editor.

6. On the **Advanced** page in the Properties view, click the **Resource** button.

7. In the REST Resources dialog box that appears, add a REST resource that the service will access by completing the following fields.

Option	Description
Resource	Specify the name of the REST resource that the service will access. Note: To add a REST resource to the service after the service has been created, right-click the service name and click Attach Resource .
HTTP Methods	(Read-only field.) The HTTP methods that the virtual service should be allowed to perform on the REST resource (GET ,

Option	Description
	POST, PUT, DELETE). You specify the HTTP methods when you configure the virtual service's Entry Step.
Content Type	(Read-only field.) Application/xml or Application/json .
Query String	Specify the search string as required. Any text typed in is not case sensitive. This field is used only for documentation purposes at this time. It is not included in the service's "virtual service definition" (VSD) that is deployed to CloudStreams, so this value has no impact at runtime.
Import from	Import the schemas that the virtual service will use. Choose one of the following options. (To select multiple schemas, click the plus button to add a row.) <ul style="list-style-type: none"> ■ URL: Specify the URL of a schema and click the Add button. Optionally choose the following option: <ul style="list-style-type: none"> ■ URL authentication: If you have specified a URL, and the site you want to access via the URL requires user authentication, select this option. This opens an Authentication sub-dialog in which you can enter a user name and password for authentication at the URL site. ■ File: Click Browse to select a schema from your local file system.

8. Click **OK**.

The Properties of a Virtual Service (REST)

To view or modify the properties of a virtual service (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name. (If the Properties view is not already open, click **Window > Show View > Other > General > Properties**.)
3. The **General** page in the Properties view displays the following properties:

Option	Description
Name	(Read-only field.) The service name. You can change the service name at any time by right-clicking the name in the Virtual Services folder and clicking Rename .
Service Type	(Read-only field.) REST .
Created/Last Modified	(Read-only field.) The service's creation/modification timestamps.
Target Namespace	(Read-only field.) The value is derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
Namespaces	Specify other namespaces (such as <code>wSDL</code> , <code>xsd</code> , etc.).
Version	The version is always set to 1.0.
WSDL URL	(Read-only field.) The URL of the REST resource's abstract WSDL.
Description	You can change the service description.

4. View additional properties on the **Advanced** page.

Option	Description
Name	(Read-only field.) The service name.
Type	(Read-only field.) REST .
Target Namespace	(Read-only field.) The value is derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
WSDL URL	Click this URL to display the contents of the REST resource's abstract WSDL. If a WSDL file was not added, this will be empty.
Namespaces	Specify other namespaces (such as <code>wSDL</code> , <code>xsd</code> , etc.).
Version	The version is always set to 1.0.
Description	(Read-only field.) The service description.

Option	Description
VSD	<p>Click the button next to this field to view the service's "virtual service definition" (VSD). This button is disabled until you deploy the virtual service to a CloudStreams server.</p> <p>When you deploy the virtual service to a CloudStreams server, CloudStreams generates an XML document called a <i>virtual service definition</i> (VSD). The VSD defines the virtual service for CloudStreams, and contains all the resources required to deploy the virtual service to a CloudStreams server, including the policy that applies to the service. You cannot edit the VSD.</p> <p>Note: If multiple policies apply to the service, CloudStreams combines all those policies into a single policy known as the <i>effective policy</i>. The effective policy is a simple UNION of the run-time actions specified in all policies that apply to a service. To create the effective policy, CloudStreams evaluates the combined list of actions from all policies, using a set of internal rules known as Policy Conflict Resolution rules. For details, see "Policy Conflict Resolution Rules" on page 203.</p>
Applicable Policies	<p>Click the button next to this field to view a list of the active policies that apply to this service. Any inactive policy that applies to the service is not listed.</p>
Endpoint	<p>Click the button next to this field to view the endpoint to which the virtual service is deployed, if applicable.</p>
Deployed Status	<p>(Read-only field.) Indicates whether the service is Deployed, Undeployed or Not Deployed (which is the initial status before you deploy the service). To deploy or undeploy a service, see "Deploying Virtual Services and Connector Virtual Services" on page 199.</p>
Resource	<p>Click the button next to this field to view the REST resource that the service will access. For details, see "Creating a New Virtual Service (REST)" on page 126.</p> <p>Note: To add a REST resource to the service, right-click the service name and click Attach Resource.</p>

Managing a Virtual Service (REST)

You can rename, delete, deploy or undeploy a virtual service, and attach a REST resource to it.

To manage a virtual service (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project.
3. Right-click the virtual service name and choose one of the following tasks from the context menu:

Option	Description
Rename	<p>Use this option to assign a new name to the service. The service must be undeployed before you can rename it (check the Deployed Status field in the Advanced page of the service's Properties view).</p> <p>Unlike native services, the names of virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the virtual service name, then the names of the services themselves must not contain characters that are invalid in virtual service names.</p>
Delete	Deletes the virtual service from CloudStreams.
Attach Resource	Use this option to attach a REST resource to an existing virtual service. The new resource will override the old one.
Deploy	Deploys the virtual service to a CloudStreams server target. For more information, see " Deploying Virtual Services and Connector Virtual Services " on page 199.
Undeploy	Undeploys the virtual service from a CloudStreams server target.

4. To change the list of policies that apply to the virtual service, see "[Modifying Policies](#)" on page 172.

The "In Sequence" Step (REST)

You configure the service's "In Sequence" step to manipulate the request messages. This step can include the following sub-steps:

- The "Entry Step" (required), which specifies the protocol (HTTP or HTTPS) of the requests that the virtual service will accept, and the HTTP methods that the virtual service should be allowed to perform on the REST resource.
- The "Transform" step (optional), which performs an XSLT message transformation on the request message before it is submitted to the native service.
- The "Invoke IS Service" step (optional), which pre-processes the request message before it is submitted to the native service.
- The "Routing Rule" step (required), which specifies how the virtual service will route the requests to the native service endpoint(s). There are four ways to route HTTP or HTTPS requests:
 - "Straight Through" routing (to route requests directly to the native service endpoint).
 - "Context-Based" routing (to route specific types of messages to specific endpoints according to context-based routing rules).
 - "Content-Based" routing (to route specific types of messages to specific endpoints based on specific values that appear in the request message).
 - "Load Balancing" routing (to distribute requests among multiple endpoints).

The "Entry Step" (REST)

The Entry Step (required) specifies the protocol (HTTP or HTTPS) over which the virtual service will accept requests, and the HTTP methods (GET, POST, PUT, DELETE) that the virtual service should be allowed to perform on a REST resource.

This step allows you to bridge protocols between the consuming application and the native service. For example, suppose you have a native service that is exposed over HTTPS and a consuming application that submits SOAP requests over HTTP. In this situation, you can configure the virtual service's Entry Step to accept HTTP requests and configure its Routing Rule step to route the request to the Web service using HTTPS.

To configure the "Entry Step" (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.

The virtual service editor displays the three main processing steps: **In Sequence**, **Out Sequence** and **Error Sequence**.

3. In the virtual service editor, expand **In Sequence**.

By default, the **In Sequence** step contains the sub-steps **Entry Step** and **Routing Rule**. (You can add the optional **Transform** and **Invoke IS Service** steps, as described later.)

4. Click **Entry Step** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Entry Step .
HTTP Methods	<p>The HTTP methods that the virtual service should be allowed to perform on the REST resource (GET, POST, PUT, DELETE).</p> <p>It is important to specify all the HTTP methods that are supported for the virtual service. For example, if the virtual service is deployed to CloudStreams and only the GET method was selected in the virtual service definition, then CloudStreams will only permit GET invocations. A POST request will be rejected with a return of statusCode 405 even if the native service happens to support POSTs. It is also important for the native service to return the correct <code>Content-Type</code> header with its responses.</p> <p>At run time, CloudStreams determines the type of a request message based on the message's HTTP method and its Content-Type. See "HTTP Method/Content-Type Combinations" on page 161.</p>
Protocol	The protocol (HTTP or HTTPS) over which the virtual service will accept requests. To specify HTTPS, select both HTTP and SSL . CloudStreams supports HTTP v1.1 only.

The "Transform" Step (Inbound, REST)

The optional "Transform" step specifies how the request message is to be transformed before it is submitted to the native service.

As long as a consumer sends a REST request with the proper Content-Type (see "[HTTP Method/Content-Type Combinations](#)" on page 161) and as long as the HTTP method and endpoint URI are in the expected form, then CloudStreams can determine the correct service and operation; in this case, no message transformation is required. However, in some cases a virtual REST service might need to transform XML messages.

For example, you might need to accommodate differences between the message content that a consuming application is capable of submitting and the message content that a native service expects. For example, if the consuming application submits an order record using a slightly different structure than the structure expected by the native

service, you can use the Transform step to transform the record submitted by the consuming application to the structure required by the Web service.

In this case, you would need to create two Transform steps:

- One in the "In Sequence" step, to transform the request messages into the format required by the native service, before CloudStreams sends the requests to the native services. To do this, you pass the message to an XSLT transformation file. (Additionally in this case, the transformation is required if the virtual service has a schema validation policy that validates the requests.)
- One in the "Out Sequence" step, to transform the native service's response messages into the format required by the consumer applications, before CloudStreams returns the responses to the consumer applications.

To add the "Transform" step (inbound, REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
3. Right-click **In Sequence** and select **Transform**.

The Transform step is added under the Entry Step. You cannot change the order of the steps.

4. Click **Transform** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only.) Transform .
XSLT File	<p>The XSLT file to transform the request message before it is submitted to the native service. Click Browse to select a file from your file system and click Save.</p> <p>The XSL file uploaded by the user should not contain the XML declaration in it (<code><?xml version="1.0" encoding="UTF-8"></code>). This is because when the virtual service is deployed to CloudStreams, CloudStreams embeds the XSL file in the virtual service definition (VSD), and since the VSD itself is in XML format, there cannot be an XML declaration line in the middle of it. This can lead to unexpected deployment issues which can be avoided by making sure the XSL file does not contain the declaration line.</p>

Option	Description
	Note: If you make changes to the XSLT file in the future, you must re-deploy the virtual service.

- To create an additional Transform step, right-click **In Sequence** and select **Transform** again.
- To delete a Transform step, right-click **Transform** and click **Delete Step**.

The "Invoke IS Service" Step (Inbound, REST)

This optional step invokes a user-defined Integration Server flow service. You can invoke an IS service in:

- The "In Sequence" step, to pre-process the request message before it is submitted to the native service.
- The "Out Sequence" step, to pre-process the response message from the native service before it is returned to the consuming application.

The IS service you invoke must be running on the same Integration Server as CloudStreams. It can call out a C++ or Java or .NET function. It can also call other IS services to manipulate the SOAP message.

You can use predefined or custom context variables in an IS service. For more information, see ["Using Context Variables in IS Services" on page 100](#).

You can create multiple "Invoke IS Service" steps.

To add the "Invoke IS Service" step (inbound, REST)

- Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
- In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
- Right-click **In Sequence** and click **Invoke IS Service**.

The "Invoke IS Service" step is added under the Entry Step (and under a Transform step, if one exists). You cannot change the order of the steps.

- Click **Invoke IS Service** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only.) Invoke IS Service .

Option	Description
Service	Specify the IS service to pre-process the request message before it is submitted to the native service.

5. When you define the IS service, the **Pipeline In** section of the flow should have the following input variables:
 - `proxy.name`: This is the name of the virtual service.
 - `SOAPEnvelope`: This is of the Java type `org.apache.axiom.soap.SOAPEnvelope`.
 - `MessageContext`: CloudStreams will automatically place a `MessageContext` variable into the pipeline before executing the IS service call. `MessageContext` is of the Java type `org.apache.axis2.context.MessageContext`.

Integration Server users can use the `Axis2 MessageContext` object to manipulate the incoming SOAP request. The Integration Server provides built-in services (the `pub.soap.*` services) to work with the `MessageContext` object to get/set/modify the SOAP body, header, properties, etc. Integration Server users should use these services to extract the information they need from the `MessageContext` to build the necessary business logic. Users do not need to understand Axis2 or Axiom (the xml object model based on StAX) to work with the SOAP request, because if they are familiar with the Integration Server `pub.soap.*` services, they can accomplish most of the tasks. For more information about these related Integration Server services, see the *webMethods Integration Server Built-In Services Reference*.
6. To create an additional "Invoke IS Service" step, right-click **In Sequence** and select **Invoke IS Service** again.
7. To delete an "Invoke IS Service" step, right-click **Invoke IS Service** and click **Delete**.

The "Routing Rule" Step (REST)

The Routing Rule step is required. You can choose one of the following protocols in your Routing Rule step for routing the requests:

- "Straight Through" routing (to route requests directly to the native service endpoint).
- "Context-Based" routing (to route specific types of messages to specific endpoints according to context-based routing rules).
- "Content-Based" routing (to route specific types of messages to specific endpoints based on specific values that appear in the request message).
- "Load Balancing" routing (to distribute requests among multiple endpoints).

The "Straight Through" Routing Rule Step (REST)

When you select the "Straight Through" routing protocol, the virtual service will route the requests directly to the native service endpoint you specify. You may specify how to authenticate requests (as with all routing protocols).

To configure the "Routing Rule" step for "Straight Through" routing (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
3. Expand **In Sequence**.
4. Click **Routing Rule** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) HTTP .
Routing Type	Select Straight Through .
Default To	<p>The URL of the service to which to route the request.</p> <p>Click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows:</p> <ul style="list-style-type: none"> ■ HTTP Properties <ul style="list-style-type: none"> ■ Connection Timeout: The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified (or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server. ■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. ■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank. ■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication.

Option	Description
	<ul style="list-style-type: none"> ■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication.
HTTP Method	<p>The HTTP method to pass to the native service.</p> <p>Typically you want to pass each request to the native service with the same HTTP method that is contained in the request. For example, if a request contains a GET method, you allow the GET method to be passed to the native service. In this case, select the value CUSTOM, which is a context variable that contains the HTTP method that is contained in the request.</p> <p>However, in other cases you might want to change the HTTP method of a request to a different HTTP method. In this case, you can specify the different method explicitly (by selecting the GET, POST, PUT or DELETE value), or you can specify the different method dynamically on a per-request basis. If you want to specify the method dynamically, select the CUSTOM option and then write an IS service to set the value of the context variable. For more information, see "Changing the HTTP Method of a REST Virtual Service" on page 162.</p>
Use credentials from incoming request	<p>This is the default HTTP Authentication option. Authenticates requests based on the credentials specified in the HTTP header. CloudStreams passes the <code>Authorization</code> header present in the original client request to the native service.</p>
Use specific credentials	<p>Authenticates requests according to the values you specify in the User and Password fields, and optionally the Domain field.</p>
Invoke service anonymously	<p>Does not authenticate requests.</p>
Use existing HTTP Headers	<p>Use the HTTP headers that are contained in the requests.</p>
Customize HTTP Headers	<p>Use the HTTP headers that you specify in the Name and Value columns on this page. If you need to specify multiple headers, use the plus button to add rows.</p>

The "Context-Based" Routing Rule Step (REST)

If you have a native service that is hosted at two or more endpoints, you can use the Context-Based protocol to route specific types of messages to specific endpoints.

The requests are routed according to the context-based routing rules you create. A routing rule specifies where the requests should be routed, and the criteria by which they should be routed there. For example, requests can be routed according to certain consumers, certain dates/times, or according to requests that exceed/fall below a specified metric (Total Count, Success Count, Fault Count, etc.). You can create one or more rules. For example, you might use this capability to route requests from certain high-priority consumers to endpoints on a fast machine.

To configure the "Routing Rule" step for "Context-Based" routing (REST)

- OpenSoftware AG Designer and display theCloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
- In theCloudStreams Governance view, expand yourCloudStreams Governance project and click the virtual service name.
- Expand **In Sequence**.
- Click **Routing Rule** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) HTTP .
Routing Type	Select Context-Based .
Rule Name	Assign a name for the rule.

Then, click the icon next to this field and choose one of the following variables in the Configure Routing Rule dialog box that appears: **Time**, **IP Address** (IPv4 or IPv6 format), **Date**, **Consumer**, **Predefined Context Variable** or **Custom Context Variable** (see "[Using Context Variables in IS Services](#)" on page 100). Then specify a value and operator appropriate for your chosen variable. If you need to specify multiple variables, use the plus button at the end of the row to add rows.

Option	Description
Route To	<p>Note: If you select the value Custom Context Variable, you must write an IS service to get/set the custom context variable, and then invoke that service in an Invoke IS Service step. CloudStreams provides an API to get/set custom context variables. For more information, see "The API for Context Variables" on page 105. CloudStreams automatically declares any custom context variables you have specified; there is no need for you to declare them.</p> <p>Specify the URL of the native service to route the request to, if the rule criteria are met.</p> <p>Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows:</p> <ul style="list-style-type: none"> ■ HTTP Properties <ul style="list-style-type: none"> ■ Connection Timeout: The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified (or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server. ■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. ■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank. <ul style="list-style-type: none"> ■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication. ■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication.
HTTP Method	The HTTP method to pass to the rule.
Default To	Enter a native service endpoint to route the request to in case all routing rules evaluate to False. Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box, as described for the Route To field above.

Option	Description
HTTP Method	<p>The HTTP method to pass to the native service.</p> <p>Typically you want to pass each request to the native service with the same HTTP method that is contained in the request. For example, if a request contains a GET method, you allow the GET method to be passed to the native service. In this case, select the value CUSTOM, which is a context variable that contains the HTTP method that is contained in the request.</p> <p>However, in other cases you might want to change the HTTP method of a request to a different HTTP method. In this case, you can specify the different method explicitly (by selecting the GET, POST, PUT or DELETE value), or you can specify the different method dynamically on a per-request basis. If you want to specify the method dynamically, select the CUSTOM option and then write an IS service to set the value of the context variable. For more information, see "Changing the HTTP Method of a REST Virtual Service" on page 162.</p>
Use credentials from incoming request	<p>Default. Authenticates requests based on the credentials specified in the HTTP header. CloudStreams passes the <code>Authorization</code> header present in the original client request to the native service.</p>
Use specific credentials	<p>Authenticates requests according to the values you specify in the User, Password and Domain fields.</p>
Invoke service anonymously	<p>Does not authenticate requests.</p>
Use existing HTTP Headers	<p>Use the HTTP headers that are contained in the requests.</p>
Customize HTTP Headers	<p>Use the HTTP headers that you specify in the Name and Value columns on this page. If you need to specify multiple headers, use the plus button to add rows.</p>

The "Content-Based" Routing Step (REST)

If you have a native service that is hosted at two or more endpoints, you can use the Content-Based routing protocol to route specific types of messages to specific endpoints based on specific values that appear in the request message.

You might use this capability, for example, to determine which operation the consuming application has requested, and route requests for complex operations to an endpoint on a fast machine.

The requests are routed according to the content-based routing rules you create. That is, they are routed based on the successful evaluation of one or more XPath expressions that are constructed utilizing the content of the request payload. For example, a routing rule might allow requests for half of the methods of a particular service to be routed to Service A, and the remaining methods to be routed to Service B.

To configure the "Routing Rule" step for "Content-Based" routing (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.
3. Expand **In Sequence**.
4. Click **Routing Rule** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) HTTP .
Routing Type	Select Content-Based .
Rule Name	Assign a name to the rule.
XPath	Create an XPath expression as follows: <ol style="list-style-type: none"> a. Click the icon next to this field to display the XPath Editor. b. In the XPath Editor that appears, view the Namespace tab, which displays all predefined namespaces. If you want to add custom namespaces, click Add Custom Namespace/prefix, specify a name and value for the namespace, and click OK. To add additional rows, use the plus button at the end of the row to add them. c. In the XPath Editor's Nodes tab, expand the namespace's node, choose the method you want for the XPath expression, and click OK. d. In the XPath Editor's Evaluator tab, evaluate the XPath expression by specifying an argument in the XPath Expression field, and clicking Evaluate.

Option	Description
Route To	<p>The true/false result of the evaluation is displayed in the Result field.</p> <p>To specify additional XPath expressions, use the plus button at the end of the row to add them.</p> <p>Note: Currently, you cannot define namespaces URIs. So the XPath expressions should not include a namespace prefix.</p> <p>Specify where to route the request if the rule criteria are met. Specify either the URL of a native service or a connection pool name.</p> <p>Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows:</p> <ul style="list-style-type: none"> ■ Connection Timeout: The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified (or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server. ■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. ■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank. <ul style="list-style-type: none"> ■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication. ■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication.
HTTP Method	The HTTP method to pass to the rule.
Default To	Enter a native service endpoint to route the request to in case all routing rules evaluate to False. Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box, as described for the Route To field above.

Option	Description
HTTP Method	<p>The HTTP method to pass to the native service.</p> <p>Typically you want to pass each request to the native service with the same HTTP method that is contained in the request. For example, if a request contains a GET method, you allow the GET method to be passed to the native service. In this case, select the value CUSTOM, which is a context variable that contains the HTTP method that is contained in the request.</p> <p>However, in other cases you might want to change the HTTP method of a request to a different HTTP method. In this case, you can specify the different method explicitly (by selecting the GET, POST, PUT or DELETE value), or you can specify the different method dynamically on a per-request basis. If you want to specify the method dynamically, select the CUSTOM option and then write an IS service to set the value of the context variable. For more information, see "Changing the HTTP Method of a REST Virtual Service" on page 162.</p>
Use credentials from incoming request	<p>Default. Authenticates requests based on the credentials specified in the HTTP header. CloudStreams passes the <code>Authorization</code> header present in the original client request to the native service.</p>
Use specific credentials	<p>Authenticates requests according to the values you specify in the User, Password and Domain fields.</p>
Invoke service anonymously	<p>Does not authenticate requests.</p>
Use existing HTTP Headers	<p>Use the HTTP headers that are contained in the requests.</p>
Customize HTTP Headers	<p>Use the HTTP headers that you specify in the Name and Value columns on this page. If you need to specify multiple headers, use the plus button to add rows.</p>

The "Load Balancing" Routing Rule Step (REST)

If you have a Web service that is hosted at two or more endpoints, you can use the Load Balancing option to distribute requests among the endpoints.

The requests are intelligently routed based on the "round-robin" execution strategy. The load for a service is balanced by directing requests to two or more services in a pool, until the optimum level is achieved. The application routes requests to services in the pool sequentially, starting from the first to the last service, without considering the

individual performance of the services. After the requests have been forwarded to all the services in the pool, the first service is chosen for the next loop of forwarding.

Load-balanced endpoints also have automatic Failover capability. If a load-balanced endpoint is unavailable (for example, if a connection is refused), then that endpoint is marked as "down" for the number of seconds you specify in the **Suspend Interval** field (during which the endpoint will not be used for sending the request), and the next configured endpoint is tried. If all the configured load-balanced endpoints are down, then a SOAP fault is sent back to the client. After the suspension period expires, each endpoint marked will be available again to send the request.

To configure the "Routing Rule" step for "Load Balancing" routing (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.
3. Expand **In Sequence**.
4. Click **Routing Rule** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) HTTP .
Routing Type	Select Load Balancing .
Route To	The URLs of two or more services in a pool to which the requests will be routed. The application routes the requests to services in the pool sequentially, starting from the first to the last service without considering the individual performance of the services. After the requests have been forwarded to all the services in the pool, the first service is chosen for the next loop of forwarding.

To specify additional services, use the plus button next to the field to add rows.

Then, click the icon next to this field and complete the Configure Endpoint Properties dialog box as follows. These properties will apply to all the endpoints.

Option	Description
	<ul style="list-style-type: none"> <li data-bbox="524 327 764 359">■ HTTP Properties <ul style="list-style-type: none"> <li data-bbox="570 380 1365 516">■ Connection Timeout: The time interval (in seconds) after which a connection attempt will timeout. If a value is not specified (or if the value 0 is specified), CloudStreams will use the default value specified in Integration Server. <li data-bbox="570 537 1365 667">■ Read Timeout: The time interval (in seconds) after which a socket read attempt will timeout. If a value is not specified (or if the value 0 is specified), the default is 30 seconds. <li data-bbox="524 716 1365 884">■ SSL Options: Optional. To enable SSL client authentication for the endpoint, you must specify values for both the Client Certificate Alias field and the IS Keystore Alias field. If you specify a value for only one of these fields, a deployment error will occur. You may leave both fields blank. <ul style="list-style-type: none"> <li data-bbox="570 905 1365 978">■ Client Certificate Alias: The client's private key to be used for performing SSL client authentication. <li data-bbox="570 999 1365 1129">■ IS Keystore Alias: The keystore alias of the instance of Integration Server on which CloudStreams is running. This value (along with the value of Client Certificate Alias) will be used for performing SSL client authentication.
Suspend Interval	A numeric timeout value (in seconds) for a failed endpoint. Default: 30. When this timeout value expires, the system routes the execution of the virtual service to the next consecutive Web service endpoint specified in the Route To field.
HTTP Method	<p data-bbox="524 1381 1133 1413">The HTTP method to pass to the native service.</p> <p data-bbox="524 1444 1321 1644">Typically you want to pass each request to the native service with the same HTTP method that is contained in the request. For example, if a request contains a GET method, you allow the GET method to be passed to the native service. In this case, select the value CUSTOM, which is a context variable that contains the HTTP method that is contained in the request.</p> <p data-bbox="524 1686 1321 1892">However, in other cases you might want to change the HTTP method of a request to a different HTTP method. In this case, you can specify the different method explicitly (by selecting the GET, POST, PUT or DELETE value), or you can specify the different method dynamically on a per-request basis. If you want to specify the method dynamically, select the CUSTOM</p>

Option	Description
Use Credentials from Incoming Request	option and then write an IS service to set the value of the context variable. For more information, see "Changing the HTTP Method of a REST Virtual Service" on page 162.
Use Specific Credentials	This is the default HTTP Authentication option. Authenticates requests based on the credentials specified in the HTTP header. CloudStreams passes the <code>Authorization</code> header present in the original client request to the native service.
Invoke Service Anonymously	Authenticates requests according to the values you specify in the User , Password and Domain fields.
Use Existing HTTP Headers	Does not authenticate requests.
Customize HTTP Headers	Use the HTTP headers that are contained in the requests.
Customize HTTP Headers	Use the HTTP headers that you specify in the Name and Value columns on the tab. If you need to specify multiple headers, use the plus button to add rows.

The "Out Sequence" Step (REST)

You configure the service's "Out Sequence" step (optional) to manipulate the response messages from the native service before they are returned to the consuming applications. This step can include:

- The "Transform" step (optional), which invokes an XSLT transformation file to transform the SOAP response payloads from XML format to the format required by the consumer.
- The "Invoke IS Service" step (optional), which processes the response message from the native service provider before it is returned to the consuming application.

The "Transform" Step (Outbound, REST)

Add this step if you need to invoke an XSLT transformation file to transform the native service's response messages into the format required by the consumer applications, before CloudStreams returns the responses to the consumer applications. For more information about when to use the Transform step, see ["The "Transform" Step \(Inbound, REST\)"](#) on page 133.

To add the "Transform" step (outbound, REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the virtual service name.
3. In the virtual service editor, expand **Out Sequence**.
4. To add the Transform step, right-click **Out Sequence** and select **Transform**.

The **Transform** step is added under the **Out Sequence** step.

5. Click **Transform** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
XSLT File	<p>The XSLT file to transform the response message before it is returned to the consuming application. Click Browse to select a file from your file system and click Save.</p> <p>The XSL file uploaded by the user should not contain the XML declaration in it (<code><?xml version="1.0" encoding="UTF-8"></code>). This is because when the virtual service is deployed to CloudStreams, CloudStreams embeds the XSL file in the virtual service definition (VSD), and since the VSD itself is in XML format, there cannot be an XML declaration line in the middle of it. This can lead to unexpected deployment issues which can be avoided by making sure the XSL file does not contain the declaration line.</p>

Note: If you make changes to the XSLT file in the future, you must re-deploy the virtual service.

6. To create an additional "Transform" step, right-click **In Sequence** and select **Transform** from the context menu.
7. To delete a "Transform" step, right-click **Transform** and click **Delete**.

The "Invoke IS Service" Step (Outbound, REST)

This optional step invokes a user-defined Integration Server flow service. You can invoke an IS service in the service's "Out Sequence" step to pre-process the response message from the native service before it is returned to the consuming application. For

more information about IS services, see "[The "Invoke IS Service" Step \(Inbound, REST\)](#)" on page 135.

To add the "Invoke IS Service" step (outbound, REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
3. Right-click **Out Sequence** and click **Invoke IS Service**.

The Invoke IS Service step is added to the Out Sequence step.

4. In the Properties view, complete the following fields.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Invoke IS Service .
Service	Specify the IS Service to pre-process the response message before it is returned to the consuming application.

5. When you define the IS service, the **Pipeline In** section of the flow should have the following input variables:
 - `proxy.name`: This is the name of the virtual service.
 - `SOAPEnvelope`: This is of the Java type `org.apache.axiom.soap.SOAPEnvelope`.
 - `MessageContext`: CloudStreams will automatically place a `MessageContext` variable into the pipeline before executing the IS service call. `MessageContext` is of the Java type `org.apache.axis2.context.MessageContext`.

Integration Server users can use the Axis2 `MessageContext` object to manipulate the incoming SOAP request. The Integration Server provides built-in services (the `pub.soap.*` services) to work with the `MessageContext` object to get/set/modify the SOAP body, header, properties, etc. Integration Server users should use these services to extract the information they need from the `MessageContext` to build the necessary business logic. Users do not need to understand Axis2 or Axiom (the xml object model based on StAX) to work with the SOAP request, because if they are familiar with the Integration Server `pub.soap.*` services, they can accomplish most of the tasks. For more information about these related Integration Server services, see the *webMethods Integration Server Built-In Services Reference*.

6. To create an additional Invoke IS Service step, right-click **In Sequence** and click **IS Service** again.

- To delete an Invoke IS Service step, right-click **Invoke IS Service** and click **Delete**.

The "Error Sequence" Step (REST)

CloudStreams returns a default fault response to the consuming application, which you can customize with context variables. This fault response is used for faults returned by the native service provider as well as faults returned by internal CloudStreams exceptions, such as policy violation errors, connection timeouts, etc. In addition, you can:

- Choose whether or not to send the native service provider's service fault content, or just send the response message.
- Invoke IS services to pre-process or post-process the error messages.

You use this step to configure error messaging for each virtual service individually. If you want to configure global error responses for *all* virtual services, set the Service Fault Configuration options, which are located in the Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Service Fault Configuration**). For details, see ["Setting the Service Fault Configuration Options" on page 40](#).

To configure the "Error Sequence" step (REST)

- Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
- In the CloudStreams Governance view, expand your CloudStreams Governance project and click the virtual service name.
- Expand **Error Sequence**.
- Click **Error Messaging** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Error Messaging .
Error Message	Select one or both of the following options: Custom Failure Response Message: When you select this option, CloudStreams returns the following fault response to the consuming application: <pre>CloudStreams encountered an error:\$ERROR_MESSAGE while executing operation:\$OPERATION service: \$SERVICE at time:\$TIME on date:\$DATE. The client</pre>

Option	Description
	<pre data-bbox="524 310 1256 373">ip was:\$CLIENT_IP. The current user:\$USER. The consumer application:\$CONSUMER_APPLICATION".</pre> <p data-bbox="524 420 1308 449">This fault response is returned in both of the following cases:</p> <ul data-bbox="524 474 1364 1186" style="list-style-type: none"> <li data-bbox="524 474 1364 976">■ When a fault is returned by the native service provider. In this case, the <code>\$ERROR_MESSAGE</code> variable in the fault response will contain the message produced by the provider's exception that caused the error. This is equivalent to the <code>getMessage</code> call on the Java Exception. For REST service calls, the message is returned inside an <code></Exception></code> tag. If the response is XML, the message is returned inside <code><Exception>'custom message'</Exception></code>. If the response is JSON, it will be returned inside <code>{"Exception":"Invalid response"}</code>. CloudStreams discards the native service provider's fault and does not return this content to the web service caller since it could be considered a security issue, especially if the native provider is returning a stack trace with its response. <li data-bbox="524 993 1364 1186">■ When a fault is returned by internal CloudStreams exceptions (policy violation errors, cloud connection errors and cloud connector service errors). In this case, the <code>\$ERROR_MESSAGE</code> variable will contain errors generated by CloudStreams. <p data-bbox="524 1224 1300 1325">The default fault response contains predefined fault handler variables (<code>\$ERROR_MESSAGE</code>, <code>\$OPERATION</code>, etc.), which are described in "The Fault Handler Variables" on page 87.</p> <p data-bbox="524 1346 1333 1446">You can customize the default fault response using the following substitution variables, where CloudStreams replaces the variable reference with the real content at run time:</p> <ul data-bbox="524 1470 1328 1656" style="list-style-type: none"> <li data-bbox="524 1470 1328 1539">■ The predefined context variables listed in "The Predefined Context Variables" on page 100. <li data-bbox="524 1556 1328 1656">■ Custom context variables that you declare using the CloudStreams API (see "The API for Context Variables" on page 105). <div data-bbox="524 1682 1364 1869" style="background-color: #f0f0f0; padding: 10px;"> <p data-bbox="537 1703 1349 1869">Note: If you want to reference a custom context variable that you have already defined in a context-based routing rule (as opposed to one you have declared using CloudStreams API for context variables), then you must add the prefix <code>\$mx</code> to the variable name in order to</p> </div>

Option	Description
	<p>reference the variable. For example, if you defined the variable <code>TAXID</code>, you would reference it as <code>\$mx:TAXID</code>.</p> <p>Native Provider Fault: When you select this option, CloudStreams sends the native service provider's service fault content, if one is available. CloudStreams will send whatever content it received from the native service provider.</p> <p>If you select this option, the Custom Failure Response Message is ignored when a fault is returned by the native service provider. (Faults returned by internal CloudStreams exceptions will still be handled by the Custom Failure Response Message option.)</p>
<p>Processing Method</p>	<p>Optionally select either of the following:</p> <ul style="list-style-type: none"> ■ Pre-Processing: Select this option if you want to invoke an IS service to manipulate the response message before the Error Sequence step is invoked. The IS service will have access to the response message context (the axis2 <code>MessageContext</code> instance) before it is updated with the custom error message. For example, you might want to send emails or perform custom alerts based on the response payload. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, REST)" on page 135. ■ Post-Processing: Select this option if you want to invoke an IS service to manipulate the service fault after the Error Sequence step is invoked. The IS service will have access to the entire service fault and the custom error message. You can make further changes to the fault message structure, if needed. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, REST)" on page 135.

Creating a New Connector Virtual Service (REST)

A connector virtual service will:

- Handle the provider's response.
- Log the requests/responses.

CloudStreams provides a default connector virtual service for each metadata handler: the service `WmCloudStreams.RestVS` (for the REST handler) and the service `WmCloudStreams.SoapVS` (for the SOAP handler). The default services are located in

the sample CloudStreams Governance project in the WmCloudStreams package. You cannot modify these default services.

Each default connector virtual service has a default policy (named Logging Policy), which logs all request/response payloads to a database. You cannot modify the default policy. Alternatively, you can create additional connector virtual services with custom policies. If you create a connector virtual service with a custom policy, you can only include the actions in the "Monitoring" or "Additional" action categories; you cannot include the "WS-SecurityPolicy 1.2" actions. For example, you might want to create a custom policy that monitors run-time performance, customizes how the service invocations are logged, validates response messages against an XML schema, or optimizes server traffic. For complete details, see the chapter "[Policies](#)" on page 167.

To create a new connector virtual service (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, right-click the CloudStreams Governance project and click **New > Connector Virtual Service** (or expand the project, right-click the **Connector Virtual Services** folder and click **New Connector Virtual Service**).
3. In the New Connector Virtual Service wizard, complete the following fields and click **Finish**.

Option	Description
CloudStreams Governance Project	Click Browse and select a CloudStreams Governance project in which to create the connector virtual service.
Name	Assign a name for the service. Unlike native services, the names of virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the virtual service name, then the names of the services themselves must not contain characters that are invalid in virtual service names. Note: If you want to change the service name after it has been created, right-click the service name in the Connector Virtual Services folder and select Rename .
Version	The version is always set to 1.0.
Service Type	(Read-only field.) REST .

Option	Description
Description	Optional. A description for the service. This description appears when a user displays a list of services in the user interface.

The new connector virtual service is added to the CloudStreams Governance project, in the **Connector Virtual Services** folder.

The Properties of a Connector Virtual Service (REST)

To view the properties of a connector virtual service (REST)

1. OpenSoftware AG Designer and display theCloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In theCloudStreams Governance view, expand yourCloudStreams Governance project and click the virtual service name. (If the Properties view is not already open, click **Window > Show View > Other > General > Properties**.)
3. The **General** page in the Properties view displays the following properties:

Option	Description
Name	(Read-only field.) The service name. You can change the service name at any time by right-clicking the name in the Virtual Services folder and clicking Rename .
Service Type	(Read-only field.) REST .
Created/Last Modified	(Read-only field.) The service's creation/modification timestamps.
Target Namespace	(Read-only field.) The value is derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
Namespaces	Click the button next to this field to view the service's available namespaces (such as <code>wSDL</code> , <code>xsd</code> , etc.).
Version	The version is always set to 1.0.
WSDL URL	(Read-only field.) The URL of the service's WSDL.
Description	You can change the service description.

4. View additional properties in the **Advanced** page.

Option	Description
Name	(Read-only field.) The service name.
Type	(Read-only field.) The type of the service (REST).
Target Namespace	(Read-only field.) The value derived from the <code>targetNamespace</code> attributes of the WSDL's definition element.
WSDL URL	Click this URL to display the contents of the service's abstract WSDL. If a WSDL file was not added, this will be empty.
Namespaces	Click the button next to this field to view the service's available namespaces (such as <code>wSDL</code> , <code>xsd</code> , etc.).
Version	The version is always set to 1.0.
Description	(Read-only field.) The service description.
VSD	Click the button next to this field to view the service's "virtual service definition" (VSD). This button is disabled until you deploy the connector virtual service to a CloudStreams server. When you deploy the connector virtual service to a CloudStreams server, CloudStreams generates an XML document called a <i>virtual service definition (VSD)</i> . The VSD defines the connector virtual service for CloudStreams, and contains all the resources required to deploy the connector virtual service to a CloudStreams server, including the policy that applies to the service. You cannot edit the VSD.
Applicable Policies	Click the button next to this field to view a list of the active policies that apply to this service. Any inactive policy that applies to the service is not listed.

Note: If multiple policies apply to the service, CloudStreams combines all those policies into a single policy known as the *effective policy*. The effective policy is a simple UNION of the run-time actions specified in all policies that apply to a service. To create the effective policy, CloudStreams evaluates the combined list of actions from all policies, using a set of internal rules known as Policy Conflict Resolution rules. For details, see "[Policy Conflict Resolution Rules](#)" on page 203.

Option	Description
Endpoint	(Read-only field). The endpoint is resolved when the connector service is configured and the user selects an enabled managed connection pool.
Deployed Status	(Read-only field.) Indicates whether the service is Deployed , Undeployed or Not Deployed (which is the initial status before you deploy the service). To deploy or undeploy a service, see "Deploying Virtual Services and Connector Virtual Services" on page 199 .
Resource	Click the button next to this field to view the REST resources that the service will access.

Managing a Connector Virtual Service (REST)

You can rename, delete, deploy or undeploy a connector virtual service, and change the list of policies that apply to it.

To manage a connector virtual service (REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project.
3. Right-click the connector virtual service name and choose any of the following tasks from the context menu:

Option	Description
Rename	<p>Use this option to assign a new name to the service. The service must be undeployed before you can rename it (check the Deployed Status field in the Advanced page of the service's Properties view).</p> <p>Unlike native services, the names of connector virtual services cannot contain spaces or special characters except _ (underscore) and - (hyphen). Consequently, if you adopt a convention that involves using the name of the service as part of the connector virtual service name, then the names of the services themselves must not contain characters that are invalid in connector virtual service names.</p>
Delete	Deletes the connector virtual service from CloudStreams.

Option	Description
Deploy	Deploys the connector virtual service to a CloudStreams server target. For more information, see "Deploying Virtual Services and Connector Virtual Services" on page 199.
Undeploy	Undeploys the connector virtual service from a CloudStreams server target.

- To change the list of policies that apply to the connector virtual service, see ["Modifying Policies"](#) on page 172.

The "In Sequence" Step (Connector Virtual Service, REST)

The "In Sequence" step can contain the following sub-steps:

- The "Entry Step" (required), which specifies the protocol of the requests that the connector virtual service will accept, and the HTTP methods that the connector virtual service should be allowed to perform on a REST resource.
- The "Routing Rule" step (required), which specifies how the connector virtual service will route the service requests to the native service endpoint.
- The "Invoke IS Service" step (optional), which pre-processes the request message before it is submitted to the native service.

The "Entry Step" (Connector Virtual Service, REST)

This step specifies the protocol of the requests that the connector virtual service will accept, and the HTTP methods that the virtual services are allowed to perform on the REST resources.

To edit the "Entry Step" (connector virtual service, REST)

- OpenSoftware AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
- In the CloudStreams Governance view, expand your CloudStreams Governance project and click the connector virtual service name.

The virtual service editor displays the three main processing steps: **In Sequence**, **Out Sequence** and **Error Sequence**.

- Expand **In Sequence**.

By default, the **In Sequence** step contains the sub-steps **Entry Step** and **Routing Rule**. (You can add the optional **Invoke IS Service** step, as described later.)

- Click **Entry Step** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Entry Step.
Protocol	(Read-only field.) The protocol of the requests that the connector virtual service will accept. The value Local means that the service can only be called from Cloud Connector Services.
HTTP Method	<p>The HTTP methods that the virtual services should be allowed to perform on the REST resources (GET, POST, PUT, DELETE).</p> <p>It is important to specify all the HTTP methods that are supported for the virtual services. For example, if a virtual service is deployed to CloudStreams and only the GET method was selected here, then CloudStreams will only permit GET invocations. A POST request will be rejected with a return of statusCode 405 even if the native service happens to support POSTs. It is also important for the native service to return the correct <code>Content-Type</code> header with its responses.</p> <p>At run time, CloudStreams determines the type of a request message based on the message's HTTP method and its Content-Type. For a list of valid HTTP method/Content-Type combinations, see "HTTP Method/Content-Type Combinations" on page 161.</p>

The "Routing Rule" Step (Connector Virtual Service, REST)

This step specifies how the connector virtual service will route the requests to the native service endpoint. You cannot modify this step, except for its name.

To edit the "Routing Rule" step (connector virtual service, REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and double-click the connector virtual service name.
3. In the virtual service editor, expand **In Sequence**.
4. Click **Routing Rule** and complete the following fields in the **General** page in the Properties view.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Routing Rule .
Protocol	(Read-only field.) Protocol . You cannot change the protocol over which the connector virtual service will accept requests.
Routing Rule	(Read-only field.) The value Connection Pool means that the requests are sent to the provider via the CloudStreams connection pool. For more information about connection pools, see the documentation specific to your CloudStreams connector (for example, <i>webMethods CloudStreams Provider for Salesforce.com Installation and User's Guide</i>).

The "Invoke IS Service" Step (Inbound, Connector Virtual Service, REST)

This optional step invokes a user-defined Integration Server flow service. An IS service is a user-defined Integration Server flow service that you can invoke in:

- The "In Sequence" step, to pre-process the request messages before CloudStreams submits the requests to the native service.
- The "Out Sequence" step, to pre-process the response messages from the native service before CloudStreams returns the responses to the consuming application.

You create an "Invoke IS Service" step for a connector virtual service the same way you create one for a virtual service. For details, see "[The "Invoke IS Service" Step \(Inbound, REST\)](#)" on page 135, or "[The "Invoke IS Service" Step \(Outbound, REST\)](#)" on page 148.

The "Out Sequence" Step (Connector Virtual Service, REST)

You configure the service's "Out Sequence" step (optional) to manipulate the response messages from the native service before CloudStreams returns the responses to the consuming applications. This step can include the "Invoke IS Service" step (optional), which processes the response message from the native service provider before it is returned to the consuming application. See "[The "Invoke IS Service" Step \(Inbound, Connector Virtual Service, REST\)](#)" on page 159.

The "Error Sequence" Step (Connector Virtual Service, REST)

The default connector virtual services are configured to return the native service provider's service fault, if one is available. CloudStreams will send whatever content it

received from the native service provider. You can optionally invoke IS services to pre-process or post-process the error messages.

To configure the "Error Sequence" step (connector virtual service, REST)

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project and click the connector virtual service name.
3. Expand **Error Sequence**.
4. Click **Error Messaging** and complete the fields in the **General** page in the Properties view as follows.

Option	Description
Name	You can optionally change the step name to any other name. There are no naming restrictions.
Type	(Read-only field.) Error Messaging .
Processing Method	<p>Optionally select either of the following:</p> <ul style="list-style-type: none"> ■ Pre-Processing: Select this option if you want to invoke an IS service to manipulate the response message before the Error Sequence step is invoked. The IS service will have access to the response message context (the axis2 <code>MessageContext</code> instance) before it is updated with the custom error message. For example, you might want to send emails or perform custom alerts based on the response payload. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, SOAP)" on page 69. ■ Post-Processing: Select this option if you want to invoke an IS service to manipulate the service fault after the Error Sequence step is invoked. The IS service will have access to the entire service fault and the custom error message. You can make further changes to the fault message structure, if needed. For more information about IS services, see "The "Invoke IS Service" Step (Inbound, SOAP)" on page 69.

Important Considerations for REST Virtual Services

Keep the following topics in mind when you configure REST virtual services:

- ["HTTP Method/Content-Type Combinations" on page 161](#)

- ["Changing the HTTP Method of a REST Virtual Service" on page 162](#)

HTTP Method/Content-Type Combinations

When you configure a REST virtual service it is important to specify all the HTTP methods that are supported for the service. For example, if the REST virtual service is deployed to CloudStreams and only the GET method was selected in the REST virtual service definition, then CloudStreams will only permit GET invocations. A POST request will be rejected with a return of statusCode 405 even if the native service happens to support POSTs.

At run time, CloudStreams determines the type of a request message based on the message's HTTP method and its Content-Type. (The absence of the `soapAction` header will indicate to CloudStreams that the message is an XML message.)

The valid HTTP method/Content-Type combinations are as follows:

This method...	Can be included in a message of this type...
POST	application/xml application/json application-x-www-form-urlencoded multipart/form-data
PUT	application/xml application/json application-x-www-form-urlencoded multipart/form-data
GET	application-x-www-form-urlencoded
DELETE	application-x-www-form-urlencoded

Note that:

- If CloudStreams receives a request sent with an HTTP method that is not specified in the REST virtual service definition, it will return a 405 error.
- If CloudStreams receives a request sent with a wrong Content-Type, it will return a 415 error. In addition, if the wrong Content-Type is used with a GET or DELETE, then the query parameters contained in the message (if any) will not be processed.

Changing the HTTP Method of a REST Virtual Service

When configuring the routing step of a REST virtual service, you specify whether to route the requests to the native service with the same HTTP method that is contained in the requests (GET, POST, PUT, DELETE), or whether to route the requests with a different HTTP method. This applies only to virtual services, not to connector virtual services.

Typically you want to pass each request to the native service with the same HTTP method that is contained in the request. For example, if a request contains a GET method, you allow the GET method to be passed to the native service. However, there might be rare cases in which you want to change the HTTP method of a request to a different HTTP method. For example, you might want to:

- Expose an XML service as a REST service.

In this case, the service you create would be a Virtual XML service that exposes the HTTP methods GET, POST, PUT and DELETE, but the routing method would always be POST.

- Expose a REST service whose virtual REST service only exposes the POST method

To change the HTTP method of a REST request, in the REST virtual service's "Routing Rule" step, set the value of the **HTTP Method** field either statically (by explicitly setting the value to **GET**, **POST**, **PUT** or **DELETE**) or dynamically (by setting the value to **CUSTOM**).

In order to use the **CUSTOM** option to set the field dynamically, you must write a IS service that sets a value of GET, POST, PUT or DELETE for a predefined context variable named ROUTING_METHOD. You need to invoke this service in the virtual service's "In Sequence" step. For details, see ["Changing HTTP Methods in Requests Dynamically Using a Context Variable" on page 165](#).

Note: Use this feature carefully, since changing HTTP methods to certain other HTTP methods could result in unintended results or errors. For example, changing an inbound GET request to a DELETE request would be a serious mistake if the deletion was not intended and the native REST service actually deleted a resource when invoked with a DELETE method. Additionally, an incoming POST or PUT request cannot be translated into a GET or DELETE if the request has nested elements. For more information, see ["Implications of Changing HTTP Methods" on page 162](#).

Implications of Changing HTTP Methods

Note the following.

When changing from...	To...	Note that...
GET	POST	<ul style="list-style-type: none"> ■ The Content-Type of the changed request is sent as application/xml or application/json, and the charset is UTF-8. ■ Depending on the structure of the native service, be aware that the native service might not be expecting the same payload structure that is being sent. In this case, you would need to transform the request message into the format required by the native service before CloudStreams sends the requests to the native service. For more information, see "Sample XSLT Transformation for GET-to-POST or GET-to-PUT" on page 165.
GET	PUT	Identical to GET-to-POST, except that CloudStreams changes the request's HTTP method from GET to PUT.
GET	DELETE	No comment.
POST	GET	<ul style="list-style-type: none"> ■ CloudStreams will translate the POSTed request elements into query string parameters, in a root element. An incoming POST or PUT request cannot be translated into a GET or DELETE if the request has nested elements. For example: <pre>(this is correct) <person> <lastName>Smith</lastName> </person> (this is incorrect) <person> <name> <last>Smith</last> </name> </person></pre> ■ If you want to send additional parameters as part of the request URL, you can transform this payload. To do this, you can use an XSLT file or a webMethods IS service call to add parameters before the request is sent to the native service. For more information, see "The "In Sequence" Step (REST)" on page 131.

When changing from...	To...	Note that...
POST	DELETE	Identical to POST-to-GET, except that CloudStreams changes the request's HTTP method from POST to DELETE.
POST	PUT	The Content-Type of the changed request is sent as application/xml or application/json, and the charset is UTF-8.
PUT	GET	Identical to POST-to-GET, except that CloudStreams changes the request's HTTP method from PUT to GET.
PUT	POST	The Content-Type of the changed request is sent as application/xml or application/json, and the charset is UTF-8.
PUT	DELETE	Identical to POST-to-DELETE, except that CloudStreams changes the request's HTTP method from PUT to DELETE.
DELETE	GET	No comment.
DELETE	POST	Identical to GET-to-POST, except that CloudStreams changes the request's HTTP method from DELETE to POST.
DELETE	PUT	Identical to GET-to-PUT, except that CloudStreams changes the request's HTTP method from DELETE to PUT.
GET, POST, PUT or DELETE	CUSTOM	See "Changing HTTP Methods in Requests Dynamically Using a Context Variable" on page 165.
GET or DELETE	POST or PUT	Note that the query parameters will be picked off the URL and stored as top-level elements when the message is sent to the native service. The query parameters are ignored on the endpoint URL and lost when we POST to the native provider (that is, do not change the protocol method).

Changing HTTP Methods in Requests Dynamically Using a Context Variable

Alternatively, instead of changing an HTTP method explicitly (statically) to PUT, POST, GET or DELETE, you can change the HTTP method to the value of a predefined context variable (ROUTING_METHOD) that dynamically resolves to a different HTTP method (PUT, POST, GET or DELETE, as appropriate).

To change the HTTP method dynamically, you create an IS service and invoke it in the virtual service's "In Sequence" step. This IS service should reference the predefined context variable ROUTING_METHOD (see ["The Predefined Context Variables" on page 100](#)). To set the value of ROUTING_METHOD, use the `setContextVariableValue` method, which is defined in the following class:

`com/softwareag/cloudstreams/api/RuntimeFacade.java`

For example:

```
public static final void updateHttpMethod(IData pipeline)
    throws ServiceException {
    String mcKey = "Message Context";
    Object obj = IDataUtil.get(pipeline.getCursor(), mcKey);
    if (obj!=null && obj instanceof
org.apache.axis2.context.MessageContext) {
        MessageContext msgCtx = (MessageContext) obj;
        QName varName =
new QName(ContextVariableType.ROUTING_METHOD.getName());
        RuntimeFacade.setContextVariableValue(varName, "PUT", msgCtx );
    }
}
```

Sample XSLT Transformation for GET-to-POST or GET-to-PUT

As stated in the above table, depending on the structure of the native service, the native service might not be expecting the same payload structure that is being sent. In this case, you would need to transform the request message into the format required by the native service before CloudStreams sends the requests to the native service. To do this, you invoke an XSLT file during the "In Sequence" step.

Assume that:

- The native service name is "authors:."
- The REST virtual service for "authors:" is named "vs-authors:" and is made available in CloudStreams at this endpoint: `http://localhost:5555/ws/vs-authors/Invoke`. The `targetNamespace` of the REST virtual service is "http://example.com/authors".

Following is a sample XSLT transformation file for the GET-to-POST or GET-to-PUT scenario.

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:ns="http://example.com/authors"
    version="1.0">
    <xsl:output method="xml" omit-xml-declaration="no" standalone="yes"
indent="yes"/>
    <xsl:strip-space elements="*" />
    <xsl:template match="node()|@"*>
```

```
<xsl:copy>
  <xsl:apply-templates select="node()|@*" />
</xsl:copy>
</xsl:template>
<xsl:template match="//ns:invoke/node()">
  <xsl:element name="{local-name(.)}">
    <xsl:value-of select="." />
  </xsl:element>
</xsl:template>
<xsl:template match="//ns:invoke">
  <xsl:element name="authors">
    <xsl:apply-templates />
  </xsl:element>
</xsl:template>
</xsl:stylesheet>
```

4 Policies

- Overview of Policies 168
- Creating Policies 171
- Modifying Policies 172
- The Policy Actions 173

Overview of Policies

A policy adds run-time governance capabilities to a virtual service or a Connector Virtual Service. A *policy* is a sequence of actions that are carried out by CloudStreams when a consumer requests a particular service through CloudStreams.

The actions in a policy perform activities such as identifying/authenticating consumers, validating digital signatures and capturing performance measurements. An *action* is a single task that is included in a policy and is evaluated by CloudStreams at run time. Actions have one or more parameters, which you configure when you insert the actions into a policy. For example, an action that identifies consumers specifies one or more identifiers to identify the consumers who are trying to access the services.

CloudStreams provides built-in action templates. A built-in action template is a definition of an action that can be used in a policy. An action template specifies the set of parameters associated with a particular policy action. You can use these action templates to create actions for policies.

When you create a policy, you:

1. Specify the service(s) to which the policy should apply. A policy can apply to one or more virtual services or one or more Connector Virtual Services, but not to both types of services.
2. Add the desired actions to the policy, and configure their parameters.
3. Activate the policy when you are ready to put it into effect. When you deploy the virtual services or Connector Virtual Services to which the policy is applied, the policy will also be deployed.

Following is a summary of the kinds of actions you can include in policies for virtual services and Connector Virtual Services. Detailed descriptions of the actions appear later, in ["The Policy Actions" on page 173](#).

Policies for Virtual Services

You should create a policy for each virtual service. You can apply a single policy to one or more virtual services. A policy for a virtual service can include the following kinds of actions:

WS-SecurityPolicy 1.2 Actions

There are two kinds of WS-SecurityPolicy 1.2 actions:

- Authentication actions, to verify that the requests for virtual services contain a specified WS-SecurityPolicy element. CloudStreams provides the following authentication actions:

- **Require SAML Token:** Requires that a WSS Security Assertion Markup Language (SAML) assertion token be present in the SOAP message header to validate service consumers.
- **Require WSS Username Token:** Requires that a WSS username token and password be present in the SOAP message header to validate service consumers.
- **Require X.509 Token:** Requires that a WSS X.509 token be present in the SOAP message header to validate service consumers.
- XML security actions, to provide confidentiality (through encryption) and integrity (through signatures) for request and response messages. CloudStreams provides the following XML security actions:
 - **Require Signing:** Requires that a request's XML element (which is represented by an XPath expression) be signed.
 - **Require Encryption:** Requires that a request's XML element (which is represented by an XPath expression) be encrypted.
 - **Require SSL:** Requires that requests be sent via SSL client certificates, and can be used with both SOAP and REST services.
 - **Include Timestamps:** Requires that timestamps be included in the request header. CloudStreams checks the timestamp value against the current time to ensure that the request is not an old message. This serves to protect your system against attempts at message tampering, such as replay attacks.

Monitoring Actions

CloudStreams provides the following run-time monitoring actions:

- **Monitor Service Performance:** This action monitors a user-specified set of run-time performance conditions for a virtual service, and sends alerts to a specified destination when these performance conditions are violated.
- **Monitor Service Level Agreement:** This action provides the same functionality as "Monitor Service Performance", but this action is different because it enables you to monitor a virtual service's run-time performance for particular consumers. You can configure this action to define a *Service Level Agreement (SLA)*, which is set of conditions that defines the level of performance that a specified consumer should expect from a service.
- **Throttling Traffic Optimization:** This action limits the number of service invocations allowed during a specified time interval, and sends alerts to a specified destination when the performance conditions are violated. You can use this action to avoid overloading the back-end services and their infrastructure, to limit specific consumers in terms of resource usage, etc.

Additional Actions

CloudStreams provides additional actions that you can use in conjunction with the actions above.

- **Identify Consumer:** You use this action in conjunction with an authentication action ("Require WSS Username Token", "Require X.509 Token", or "Require HTTP Basic Authentication"). Alternatively, this action can be used alone to identify consumers only by host name or IP address.
- **Require HTTP Basic Authentication:** This action uses HTTP Basic authentication to verify the consumer's authentication credentials contained in the request's Authorization header. CloudStreams authorizes the credentials against the list of users registered in the Integration Server on which CloudStreams is running. This action supports WS-SecurityPolicy 1.2.
- **Authorize User:** This action authorizes consumers against a list of users and/or a list of groups registered in the Integration Server on which CloudStreams is running. You use this action in conjunction with an authentication action ("Require WSS Username Token", "Require SAML Token", or "Require HTTP Basic Authentication").
- **Log Invocation:** Logs request/response information (and optionally their payloads) to a destination you specify.
- **Validate Schema:** Validates all XML request and/or response messages against an XML schema referenced in the WSDL.

Policies for Connector Virtual Services

Each default Connector Virtual Service has a default policy, which logs the requests/responses to a database. You cannot modify the default Connector Virtual Service or its policy. However, you can create additional Connector Virtual Services with custom policies. If you create a Connector Virtual Service with a custom policy, you can include the actions from the "Monitoring" or "Additional" action categories (except for "Validate Schema"); the "WS-SecurityPolicy 1.2" actions are not needed. For example, you might want to create a custom policy that monitors run-time performance, customizes how the service invocations are logged, or optimizes server traffic.

The default Connector Virtual Service policies are named Logging Policy, located in the sample CloudStreams Governance project in the WmCloudStreams package.

The remainder of this section describes:

- ["Creating Policies" on page 171](#)
- ["Modifying Policies" on page 172](#)
- ["The Policy Actions" on page 173](#)

Creating Policies

To create a policy

1. Open Software AG Designer and display the CloudStreams Development perspective.
2. In the CloudStreams Governance view, expand your CloudStreams Governance project folder, right-click the **Policies** folder and click **New Policy**.
3. In the New Policy wizard, complete the following fields.

Option	Description
CloudStreams Project	Click Browse and select the CloudStreams project in which to create the policy. Select a CloudStreams Governance project if the policy is for a virtual service or a Connector Virtual Service, or select a CloudStreams Provider project if the policy is for a custom cloud connector.
Name	A name for the new policy. A policy name can contain any character, including spaces. Note: If you want to change the policy name after it has been created, right-click the service name in the Policies folder and select Rename .
Service Type	Specify whether the policy will be applied to a virtual service or to a Connector Virtual Service.
Description	Optional. A description for the new policy. This description appears when a user displays a list of policies in the user interface.

4. Click **Next**.
5. Click the button next to the **Criteria** field to apply the policy to services that meet certain criteria. In the Criteria dialog box that appears, specify the criteria as follows:
 - a. In the **Criteria** field, specify a value (**SOAP** or **REST**) for the **Service Type** attribute. That is, specify whether the policy should apply to SOAP services or REST services.
 - b. Click the plus button next to the **Criteria** field to add a new row, and choose another criteria attribute (**Name** or **Description**) and choose an operator and value for it. For example, specify whether the policy should apply to all SOAP services that are equal to, or start with, or contain the prefix `Abc_`.

- c. Repeat step 2 if desired to specify additional criteria.
- d. If you specify multiple criteria, use the **Condition** field to connect the criteria by the AND or OR operator. The default operator is AND.

Notes:

- If you specify no criteria, the policy will apply to all virtual services or connector virtual services (based on your selection in the wizard).
- **Caution:** CloudStreams checks for policy conflicts when you deploy a virtual service. If the service has only one policy applied to it (that is, the policy you are applying here), that policy will be deployed to CloudStreams, and CloudStreams executes the policy's actions in the order in which they are specified in the policy. However, if the service has multiple policies applied to it, a policy conflict might occur. A policy conflict might have unintended consequences. CloudStreams will warn you of policy conflicts. For more information, see ["What Happens When You Deploy a Service?" on page 202](#).

6. In the policy editor, click **Add Action** and select the first action to include in the policy.

Some actions have input parameters, which are displayed in the **Action Values** section of the editor. For example, the Identify Consumer action has the parameter `Anonymous Usage Allowed`.

7. Specify values for the input parameters (if any) displayed in the **Action Values** section.
8. Right-click the action name in the **Applied Actions** list. Some actions have additional parameters you can select. For detailed descriptions of all policy action parameters, see ["The Policy Action Reference" on page 177](#).
9. Select another action to include in the policy, if desired.

Note: Make sure to select the actions in the order in which you want them to run when the policy is enforced. You cannot change the order of the actions after you add them; you can only delete them and then add a new list of actions.

10. Activate the policy when you are ready to put it into effect by right-clicking the policy name in the **Policies** folder and clicking **Active**. You will not be allowed to activate the policy unless all of its action parameters have been set. If the activation is successful, the value of the **Status** field in the Properties view will change to **Active**.

When you deploy the virtual services or Connector Virtual Services to which the policy is applied, the policy will be deployed as well.

Modifying Policies

To modify a policy

1. Open Software AG Designer and display the CloudStreams Development perspective.

2. In the CloudStreams Governance view, click the policy name in the **Policies** folder.
3. You can modify:
 - a. The policy name: Right-click the policy name in the **Policies** folder and click **Rename**.
 - b. The policy description in the Properties view.
 - c. The policy's criteria (to change the services to which the policy applies): Click the button next to the **Criteria** field in the Properties view (or right-click the policy name in the **Policies** folder and click **Criteria**). The **Criteria** dialog box is described above, in ["Creating Policies" on page 171](#).
 - d. The policy's actions and their parameters in the **Applied Actions** and **Action Values** sections.
 - e. To delete the policy, right-click the policy name in the **Policies** folder and click **Delete**.
4. Activate the changed policy when you are ready to put it into effect by right-clicking the policy name in the **Policy** folder and clicking **Active**. You will not be allowed to activate the policy unless all of its action parameters have been set. If the activation is successful, the value of the **Status** field in the Properties view will change to **Active**.

When you deploy the virtual services or Connector Virtual Services to which the policy is applied, the policy will be deployed as well.

The Policy Actions

This section describes the set of policy actions that is installed with CloudStreams. The content is organized under the following sections:

- ["Action Evaluation Order and Dependencies" on page 173](#): Lists the order in which CloudStreams evaluates the actions, lists action dependencies, and presents usage cases for authenticating/identifying consumers.
- ["The Policy Action Reference" on page 177](#): An alphabetic reference of the actions and their parameters.

Action Evaluation Order and Dependencies

The following table shows:

- The order in which CloudStreams evaluates the actions to determine possible policy conflicts. CloudStreams resolves policy conflicts as described in ["What Happens When You Deploy a Service?" on page 202](#).
- Any action dependencies (that is, whether an action must be used in conjunction with another particular action).

Evaluation Order	Action	Dependency
1	Require SSL	None
2	Require HTTP Basic Authentication	The "Identify Consumer" action (if the Authenticate Credentials option is selected).
3	Require WSS Username Token	None
4	Require X.509 Token	None
5	Require SAML Token	None
6	Require Signing	None
7	Require Encryption	None
8	Require Timestamps	The actions "Require SSL", "Require Signing" <i>and</i> "Require Encryption".
9	Identify Consumer	<p>If "Identify Consumer"'s identifier field is set to:</p> <ul style="list-style-type: none"> ■ HTTP Token, the action "Require HTTP Basic Authentication" may also be required. ■ WSS Header Token, the action "Require WSS Username Token" is also required. ■ Consumer Certificate, the actions "Require X.509 Token" or "Require Signing" are also required.
10	Authorize User	The "Require HTTP Basic Authentication", "Require WSS Username Token" <i>or</i> "Require SAML Token" action.
11	Validate Schema	None

Evaluation Order	Action	Dependency
12	Log Invocation	None
13	Monitor Service Performance	None
14	Monitor Service Level Agreement	"Identify Consumer"
15	Throttling Traffic Optimization	"Identify Consumer" (if the Limit Traffic for Applications option is selected).

Usage Cases for Identifying/Authenticating Consumers

When deciding which type of identifier to use to identify a consumer application, consider the following points:

- Whatever identifier you choose to identify a consumer application, it must be unique to the application. Identifiers that represent user names are often not suitable because the identified users might submit requests for multiple applications.
- Identifying applications by IP address or host name is often a suitable choice, however, it does create a dependency on the network infrastructure. If a consumer application moves to a new machine, or its IP address changes, you must update the identifiers in the application asset.
- Using X.509 certificates or a custom token that is extracted from the SOAP message itself (using an XPATH expression), is often the most trouble-free way to identify a consumer application.

Following are some combinations of actions you can use to identify/authenticate consumers.

- Scenario 1: Identify consumers by IP address or host name

The simplest way to identify consumers is to use the Identify Consumer action and select either the **IP Address** or **Host Name** parameter.

- Scenario 2: Authenticate consumers by HTTP authentication token

Use the following actions:

- Identify Consumer action, and select the **HTTP Authentication Token** parameter (to identify consumers using the token derived from the HTTP header).
- Require HTTP Basic Authentication action.

- Authorize User action (to authorize a list of users and/or groups registered in the Integration Server on which CloudStreams is running).
- Scenario 3: Authenticate consumers by WS-Security authentication token
Use the following actions:
 - Identify Consumer action, and select the **WS-Security Authentication Token** parameter (to identify consumers using the token derived from the WSS Header).
 - Require WSS Username Token action.
 - Authorize User action (to authorize a list of users and/or groups registered in the Integration Server on which CloudStreams is running).
- Scenario 4: Authenticate consumers by WSS X.509 token
Use the following actions:
 - Identify Consumer action, and select the **Consumer Certificate** parameter (to identify consumers using the WSS X.509 token).
 - Require X.509 Token action.
 - Require SSL action.

Multiple Security Elements in Requests/Responses

SOAP allows you to send multiple security elements in the SOAP header of a request or response. When CloudStreams receives a message with multiple security elements in the SOAP header, it will process only the first security element listed. It ignores all other security elements in the request.

In order for CloudStreams to process the security element, the virtual service must be configured with the proper policy action to handle the element (for example, "Require WSS Username Token", "Require X.509 Token", etc.). If the proper policy actions are *not* configured for the virtual service, CloudStreams will not process the security header (even if the `mustUnderstand` attribute of the security element is set to 1 (true)). In this case, CloudStreams will forward the message to the native service or the consumer unchanged.

If the proper policy actions are configured for the virtual service, CloudStreams processes the requests/responses as follows:

1. CloudStreams processes the first security element found in the message.
2. CloudStreams removes the security element from the message before sending it to the native service or the consumer.
3. If the security policy has been violated, CloudStreams sends a policy violation event notification (assuming that the policy is configured for policy violation event notifications).
4. Processing is complete; CloudStreams ignores all but the first security element.

The Policy Action Reference

This section describes, in alphabetic order, each policy action provided by CloudStreams:

- "Authorize User" on page 177
- "Identify Consumer" on page 178
- "Include Timestamps" on page 179
- "Log Invocation" on page 180
- "Monitor Service Performance" on page 181
- "Monitor Service Level Agreement (SLA)" on page 184
- "Require Encryption" on page 188
- "Require HTTP Basic Authentication" on page 190
- "Require Signing" on page 192
- "Require SSL" on page 194
- "Require SAML Token" on page 190
- "Require WSS Username Token" on page 194
- "Require X.509 Token" on page 195
- "Throttling Traffic Optimization" on page 195
- "Validate Schema" on page 198

Authorize User

Note: Dependency requirement: A policy that includes this action must also include *one* of the following actions: Require HTTP Basic Authentication, Require WSS Username Token or Require SAML Token.

This action authorizes consumers against a list of users and/or a list of groups registered in the Integration Server on which CloudStreams is running.

Input Parameters

User Right-click the action name and click **Add User** to authorize consumers against a list of users who are registered in the Integration Server instance on which CloudStreams is running. You can select **Add User** multiple times to add multiple users.

Group Right-click the action name and click **Add Group** to authorize consumers against a list of groups who are registered in the

Integration Server instance on which CloudStreams is running. You can select **Add Group** multiple times to add multiple groups.

Identify Consumer

Specifies the kind of consumer identifier (IP address, HTTP authorization token, etc.) that CloudStreams will use to identify consumer applications. You can select only one identifier. Alternatively, you can allow anonymous users to send requests, without restriction.

Input Parameters

Anonymous Usage Allowed

Allows all users to send requests, without restriction.

- **True:** Allows all users to send requests.
- **False:** Default. Allows only the users identified by your selected identifier (below) to send requests.

IP Address

Right-click the action name and click **Add IP Address** to identify consumer applications based on their originating IP addresses.

Host Name

Right-click the action name and click **Add Host Name** to identify consumer applications based on a host name.

HTTP Token

Right-click the action name and click **Add HTTP Token** if you want to use HTTP Basic authentication to verify the consumer's authentication credentials contained in the request's Authorization header. CloudStreams authorizes the credentials against the list of users registered in the Integration Server on which CloudStreams is running. This type of consumer authentication is referred to as "preemptive authentication". If you want to use "preemptive authentication", you should also include the action "Require HTTP Basic Authentication" in the policy.

If you choose to omit "Require HTTP Basic Authentication", the client will be presented with a security challenge. If the client successfully responds to the challenge, the user is authenticated. This type of consumer authentication is referred to as "non-preemptive authentication". For more information, see "[Require HTTP Basic Authentication](#)" on page 190.

Note: If you select the value **HTTP Token**, do not include the "Authorize Against Registered Consumers" action in the policy. This is an invalid combination.

WSS Header Token	Right-click the action name and click Add WSS Header Token to validate user names and passwords that are transmitted in the SOAP message header in the WSS Username Token. If you select this option, you should also include the action Require WSS Username Token in the policy.
XPATH Token	Right-click the action name and click Add XPATH Token to validate consumer applications based on an XML element (represented by an XPATH expression you specify in the XPATH to Identify Token field).
Consumer Certificate	Right-click the action name and click Add Consumer Certificate to identify consumer applications based on information in a WSS X.509 certificate. If you select this option, you should also include the Require X.509 Token or the Require Signing action in the policy.
User ID	<p>If you are applying the Identify Consumer action to a connector virtual service, right-click the action name and click Add User ID. No other identifier is valid if you are applying the Identify Consumer action to a connector virtual service.</p> <p>User ID identifies consumer applications based on a list of users who are registered in the Integration Server on which CloudStreams is running. (You need to apply the Identify Consumer action to a Connector Virtual Service if you apply the following actions to the Connector Virtual Service: "Monitor Service Level Agreement" or "Throttling Traffic Optimization" (if you select its Limit Traffic for Applications option)).</p>

Include Timestamps

Note: Dependency requirement: A policy that includes this action must also include *all* of the following actions: Require SSL, Require Signing, and Require Encryption.

If you include this action, CloudStreams will require that timestamps be included in the request header. This action supports WS-SecurityPolicy 1.2 and cannot be used with REST services or connector virtual services.

CloudStreams checks the timestamp value against the current time to ensure that the request is not an old message, which serves to protect your system against attempts at message tampering, such as replay attacks.

CloudStreams rejects the request if either of the following things happen:

- CloudStreams receives a timestamp that exceeds the time defined by the timestamp element.
- A timestamp element is not included in the request.

Input Parameters

None.

Log Invocation

Logs request/response payloads. You can log the payloads in the database and/or send the payloads in the form of email alerts. This action also logs other information about the request/response, including the service name, operation name, the Integration Server user, a timestamp, the response time, and more.

Input Parameters

Log Generation Frequency

Specifies how frequently to log the payload.

- **Always:** Log all requests and/or responses.
- **On Success:** Log only the successful requests and/or responses.
- **On Failure:** Log only the failed requests and/or responses.

Log the Following Payloads

Specifies whether to log all request payloads, all response payloads, or both.

- **Log Request:** Log all request payloads.
- **Log Response:** Log all response payloads.

Send Data To

Specify where to log the payloads.

- **Database:** Default. Logs the payloads in the CloudStreams Analytics database.

Note: Be sure to also select the **Database Publishing** option in Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Database**), as described in ["Setting the Database Options for Publishing Performance Metrics and Events"](#) on page 33.

- **Server Log:** Logs the payloads in the server log of the Integration Server on which CloudStreams is running.

Also choose a value in the **Log Level** field:

- **Info:** Logs error-level, warning-level, and informational-level alerts.

- Warn: Logs error-level and warning-level alerts.
- Error: Logs only error-level alerts.

Note: The Integration Server Administrator's logging level for CloudStreams should match the logging level specified for this action (in the Integration Server Administrator go to **Settings > Logging > Server Logger**).

Alert Email

Right-click the action name and click **Add Alert Email** to send the payloads in an email alert to the email address you specify in the **Email ID** field. You can select **Add Alert Email** multiple times to add multiple email addresses.

Note: Be sure to set the email options in Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Email**), as described in "[Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts](#)" on page 31.

Monitor Service Performance

This action monitors a user-specified set of run-time performance conditions for a virtual service, and sends alerts to a specified destination when the conditions are violated.

For the counter-based metrics (Total Request Count, Success Count, Fault Count), CloudStreams sends an alert as soon as the performance condition is violated, without having to wait until the end of the metrics tracking interval. CloudStreams sends only one alert the first time the condition is violated during the interval. (It will send another alert the next time a condition is violated during a subsequent interval.) For information, see "[The Intervals for Metric Publishing](#)" on page 34.

For the aggregated metrics (Average Response Time, Minimum Response Time, Maximum Response Time), CloudStreams aggregates the response times at the end of the interval, and then sends an alert if the performance condition is violated.

Note: By default, this action does not include metrics for failed invocations. To include metrics for failed invocations, set the `pg.PgMetricsFormatter.includeFaults` parameter to true in `IntegrationServer_directory\instances\instance_name\packages\WmCloudStreams\config\resources\wst-config.properties`. For more information, see "[Advanced Settings](#)" on page 289.

Input Parameters

Alert Interval	The time period (in minutes) in which to monitor performance before sending an alert if a condition is violated.
Alert Frequency	<p>Specifies how frequently to issue alerts.</p> <ul style="list-style-type: none">■ Every Time: Issue an alert every time one of the specified conditions is violated.■ Only Once: Issue an alert only the first time one of the specified conditions is violated.
Alert Message	A text message to include in the alert.
Send Data To	<p>Specify where to log the alerts.</p> <ul style="list-style-type: none">■ Database: Default. Logs the alerts in the CloudStreams Analytics database. <div data-bbox="570 932 1367 1136"><p>Note: Be sure to also select the Database Publishing option in Integration Server Administrator (go to Solutions > CloudStreams > Administration > Database), as described in "Setting the Database Options for Publishing Performance Metrics and Events" on page 33.</p></div> <ul style="list-style-type: none">■ Server Log: Logs the alerts in the server log of the Integration Server on which CloudStreams is running. <p>Also choose a value in the Log Level field:</p> <ul style="list-style-type: none">■ Info: Logs error-level, warning-level, and informational-level alerts.■ Warn: Logs error-level and warning-level alerts.■ Error: Logs only error-level alerts. <div data-bbox="618 1486 1367 1690"><p>Note: The Integration Server Administrator's logging level for CloudStreams should match the logging level specified for this action (in the Integration Server Administrator go to Settings > Logging > Server Logger).</p></div>
Metrics Collection Level	The run-time performance metrics for a virtual service (which is invoked only in the inbound run-time scenario), are collected at the service level. That is, the metrics for all invocations of a single virtual services are aggregated together

during your specified metrics publishing interval and then published.

In contrast, the metrics for a connector virtual service (which is invoked only in the outbound run-time scenario) can be collected at two different levels of metric collection:

- **Cloud Connector Service:** Remember that a single connector virtual service can be used by multiple cloud connector services. Select this option if you want to collect the metrics for the connector virtual service broken down by each separate cloud connector service that uses it. For example, if a connector virtual services is used by three cloud connector services, then this option will collect the metrics of that service separately, broken down by each of the three cloud connector services that use it.
- **Connector Virtual Service**(default): Select this option if you want to aggregate all the metrics for a single connector virtual service, even if it is used by multiple cloud connector services. For example, if a connector virtual service is used by three cloud connector services, then this option will collect the combined metrics for the connector virtual service by all three of the cloud connector services that use it.

Action Configuration

Right-click the action name and click **Add Action Configuration** to specify a condition to monitor. To do this, select a **condition Name** (the metric to monitor), an **Operator**, and a **Value** for the condition. You can select **Add Action Configuration** multiple times to add multiple conditions. Multiple conditions are connected by the AND operator.

Name: The metric to monitor, which can be:

- **Availability:** Indicates whether the service was available to the specified consumers in the current interval. A value of 100 indicates that the service was always available. If invocations fail due to policy violations, this parameter could still be as high as 100. That is, SOAP faults returned by the native provider or faults due to CloudStreams policy enforcements do not impact Availability. Only errors that CloudStreams interprets as a provider service being down will impact Availability.
- **Average Response:** The average amount of time it took the service to complete all invocations in the current interval. Response time is measured from the moment CloudStreams receives the request until the moment it returns the response to the caller.

- **Fault Count:** The number of faults returned in the current interval.
- **Maximum Response :** The maximum amount of time to respond to a request in the current interval.
- **Minimum Response:** The minimum amount of time to respond to a request in the current interval.
- **Successful Request Count:** The number of successful requests in the current interval.
- **Total Request Count:** The total number of requests (successful and unsuccessful) in the current interval.

Alert Email

Right-click the action name and click **Add Alert Email** if you want to send the monitoring alerts to an email address you specify in the **Email ID** field. You can select **Add Alert Email** multiple times to add multiple email addresses.

Note: Be sure to also set the email options in Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Email**), as described in "[Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts](#)" on page 31.

Monitor Service Level Agreement (SLA)

Note: Dependency requirement: A policy that includes this action must also include the Identify Consumer action.

This action is similar to the Monitor Service Performance action. Both actions can monitor the same set of run-time performance conditions for a virtual service, and then send alerts when the performance conditions are violated. This action is different because it enables you to monitor run-time performance for *one or more specified consumers*.

You can configure this action to define a *Service Level Agreement (SLA)*, which is set of conditions that defines the level of performance that a consumer should expect from a service. You can use this action to identify whether a service's threshold rules are met or exceeded. For example, you might define an agreement with a particular consumer that sends an alert to the consumer if responses are not sent within a certain maximum response time. You can configure SLAs for each virtual service/consumer application combination.

For the counter-based metrics (Total Request Count, Success Count, Fault Count), CloudStreams sends an alert as soon as the performance condition is violated, without having to wait until the end of the metrics tracking interval. You can choose whether to

send an alert only once during the interval, or every time the violation occurs during the interval. (CloudStreams will send another alert the next time a condition is violated during a subsequent interval.) For information about intervals, see ["The Intervals for Metric Publishing" on page 34](#).

For the aggregated metrics (Average Response Time, Minimum Response Time, Maximum Response Time), CloudStreams aggregates the response times at the end of the interval, and then sends an alert if the performance condition is violated.

Note: By default, this action does not include metrics for failed invocations. You can include metrics for failed invocations by setting the `pg.PgMetricsFormatter.includeFaults` parameter to true in `IntegrationServer_directory\instances\instance_name\packages\WmCloudStreams\config\resources\wst-config.properties`. For more information, see ["Advanced Settings" on page 289](#).

Input Parameters

Alert Interval The time period (in minutes) in which to monitor performance before sending an alert if a condition is violated.

Alert Frequency Specifies how frequently to issue alerts.

- **Every Time:** Issue an alert every time one of the specified conditions is violated.
- **Only Once:** Issue an alert only the first time one of the specified conditions is violated.

Alert Message A text message to include in the alert.

Send Data To Specify where to log the alerts.

- **Database:** Default. Logs the alerts in the CloudStreams Analytics database.

Note: Be sure to also select the **Database Publishing** option in Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Database**), as described in ["Setting the Database Options for Publishing Performance Metrics and Events" on page 33](#).

- **Server Log:** Logs the alerts in the server log of the Integration Server on which CloudStreams is running.

Also choose a value in the **Log Level** field:

- **Info:** Logs error-level, warning-level, and informational-level alerts.

- Warn: Logs error-level and warning-level alerts.
- Error: Logs only error-level alerts.

Note: The Integration Server Administrator's logging level for CloudStreams should match the logging level specified for this action (in the Integration Server Administrator go to **Settings > Logging > Server Logger**).

Metrics Collection Level

The run-time performance metrics for a virtual service (which is invoked only in the inbound run-time scenario), are collected at the service level. That is, the metrics for all invocations of a single virtual services are aggregated together during your specified metrics publishing interval and then published.

In contrast, the metrics for a connector virtual service (which is invoked only in the outbound run-time scenario) can be collected at two different levels of metric collection:

- **Cloud Connector Service:** Remember that a single connector virtual service can be used by multiple cloud connector services. Select this option if you want to collect the metrics for the connector virtual service broken down by each separate cloud connector service that uses it. For example, if a connector virtual services is used by three cloud connector services, then this option will collect the metrics of that service separately, broken down by each of the three cloud connector services that use it.
- **Connector Virtual Service**(default): Select this option if you want to aggregate all the metrics for a single connector virtual service, even if it is used by multiple cloud connector services. For example, if a connector virtual service is used by three cloud connector services, then this option will collect the combined metrics for the connector virtual service by all three of the cloud connector services that use it.

Action Configuration

Right-click the action name and click **Add Action Configuration** to specify a condition to monitor. To do this, select a condition **Name** (the metric to monitor), an **Operator**, and a **Value** for the condition. You can select **Add Action Configuration** multiple times to add multiple conditions. Multiple conditions are connected by the AND operator.

Name: The metric to monitor, which can be:

- **Availability:** Indicates whether the service was available to the specified consumers in the current interval. A value of 100

indicates that the service was always available. If invocations fail due to policy violations, this parameter could still be as high as 100. That is, SOAP faults returned by the native provider or faults due to CloudStreams policy enforcements do not impact Availability. Only errors that CloudStreams interprets as a provider service being down will impact Availability.

- **Average Response:** The average amount of time it took the service to complete all invocations in the current interval. Response time is measured from the moment CloudStreams receives the request until the moment it returns the response to the caller.
- **Fault Count:** The number of faults returned during the current interval.
- **Maximum Response :** The maximum amount of time to respond to a request in the current interval.
- **Minimum Response:** The minimum amount of time to respond to a request in the current interval.
- **Successful Request Count:** The number of successful requests in the current interval.
- **Total Request Count:** The total number of requests (successful and unsuccessful) in the current interval.

Alert for Applications

Right-click the action name and click **Add Alert for Applications** to specify the consumer application to which this Service Level Agreement will apply. You can select **Add Alert for Applications** multiple times to add multiple consumer applications.

Add Alert Email

Right-click the action name and click **Add Alert Email** if you want to send the monitoring alerts to an email address you specify in the **Email ID** field. You can select **Add Alert Email** multiple times to add multiple email addresses.

Note: Be sure to also set the email options in Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Email**), as described in "[Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts](#)" on page 31.

Require Encryption

This action requires that an XML element (which is represented by an XPath expression) be encrypted. This action supports WS-SecurityPolicy 1.2 and cannot be used with REST services or connector virtual services.

Prerequisites:

1. Configure Integration Server: Set up keystores and truststores in Integration Server (see the section *Securing Communications with the Server* in the document *webMethods Integration Server Administrator's Guide*).
2. Configure CloudStreams: In the Integration Server Administrator, navigate to **Solutions > CloudStreams > Administration > General** and complete the **IS Keystore Name**, **IS Truststore Name** and **Alias (signing)** fields, as described in "[Setting the General Options](#)" on page 28).

When this policy action is set for the virtual service, CloudStreams provides decryption of incoming requests and encryption of outgoing responses. CloudStreams can encrypt and decrypt only individual elements in the SOAP message body that are defined by the XPath expressions configured for the policy action. CloudStreams requires that requests contain the encrypted elements that match those in the XPath expression. You must encrypt the entire element, not just the data between the element tags. CloudStreams rejects requests if the element name is not encrypted.

Note: Do not encrypt the entire SOAP body because a SOAP request without an element will appear to CloudStreams to be malformed.

CloudStreams attempts to encrypt the response elements that match the XPath expressions with those defined for the policy. If the response does not have any elements that match the XPath expression, CloudStreams will not encrypt the response before sending. If the XPath expression resolves a portion of the response message, but CloudStreams cannot locate a certificate to encrypt the response, then CloudStreams sends a SOAP fault exception to the consumer and a Policy Violation event notification to CloudStreams.

How CloudStreams Encrypts Responses:

The Require Encryption action encrypts the response back to the client by dynamically setting a public key alias at run time. CloudStreams determines the public key alias as follows:

1. If CloudStreams can access the X.509 certificate of the client (based on the incoming request signature), it will use `useReqSigCert` as the public key alias.

OR

2. If the Identify Consumer action is present in the policy (and it successfully identifies a consumer application), then CloudStreams will look for a public key alias with that consumer name in the `IS Keystore Name` property. The `IS Keystore Name` property is specified in the Integration Server Administrator, under **Solutions >**

CloudStreams > Administration > General. This property should be set to an Integration Server keystore that CloudStreams will use.

For an Identify Consumer action that allows for anonymous usage, CloudStreams does not require a consumer name in order to send encrypted responses. In this case, CloudStreams can use one of the following to encrypt the response in the following order, depending on what is present in the security element:

- A signing certificate.
- Consumer name.
- WSS username, SAML token or X.509 certificate.
- HTTP authorized user.

OR

3. If CloudStreams can determine the current IS user from the request (that is, if an Integration Server WS-Stack determined that Subject is present), then the first principal in that subject is used.

OR

4. If the above steps all fail, then CloudStreams will use either the WS-Security username token or the HTTP Basic-Auth user name value. There should be a public key entry with the same name as the identified username.

Input Parameters

Element Required to be Encrypted An XPath expression that represents the XML element that is required to be encrypted.

Namespace Prefix Optional. Right-click the action name and click **Add Namespace Prefix** if you want to specify the namespace prefix of the element required to be encrypted. Enter the namespace prefix in the following format:

xmlns:prefix-name

For example:

xmlns:soapenv

For more information, see the XML Namespaces specifications at <http://www.w3.org/TR/REC-xml-names/#ns-decl>.

See below for an example XPath element generated in the policy.

Example Generated XPath Element

The generated XPath element in the policy should look similar to this:

```
<sp:SignedElements xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
```

```
<sp:XPath xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"//soapenv:Body</sp:XPath>  
</sp:SignedElements>
```

Require HTTP Basic Authentication

This action uses HTTP Basic authentication to verify the consumer's authentication credentials contained in the request's Authorization header. CloudStreams authorizes the credentials against the list of users registered in the Integration Server on which CloudStreams is running. This action supports WS-SecurityPolicy 1.2.

If the user/password value in the Authorization header cannot be authenticated as a valid Integration Server user (or if the Authorization header is not present in the request), a 500 SOAP fault is returned, and the client is presented with a security challenge. If the client successfully responds to the challenge, the user is authenticated. If the client does not successfully respond to the challenge, a 401 "WWW-Authenticate: Basic" response is returned, and the invocation is not routed to the policy engine. As a result, no events are recorded for that invocation, and its key performance indicator (KPI) data are not included in the performance metrics.

If you choose to omit the Require HTTP Basic Authentication action (regardless of whether an Authorization header is present in the request or not), then:

- CloudStreams forwards the request to the native service, without attempting to authenticate the request.
- The native service returns a 401 "WWW-Authenticate: Basic" response, which CloudStreams will forward to the client; the client is presented with a security challenge. If the client successfully responds to the challenge, the user is authenticated.

In the case where a consumer is sending a request with both transport credentials (HTTP Basic authentication) and message credentials (WSS Username or X.509 token), the message credentials take precedence over the transport credentials when Integration Server is determining which credentials it should use for the session. For more information, see ["Require WSS Username Token" on page 194](#) and ["Require X.509 Token" on page 195](#). In addition, you must ensure that the consumer that connects to the virtual service has an Integration Server user account.

Input Parameters

Authenticate Credentials	Authorizes consumers against the list of users registered in the Integration Server on which CloudStreams is running. If you select this option, you must also include the Identify Consumer action in the policy.
---------------------------------	--

Require SAML Token

Requires that a WSS Security Assertion Markup Language (SAML) assertion token be present in the SOAP message header to validate service consumers. CloudStreams

supports SAML 1.1 and 2.0 tokens. This action supports WS-SecurityPolicy 1.2 and cannot be used with REST services or connector virtual services.

Note: When a Require SAML Token action is generated, CloudStreams also implicitly selects the `timestamp` and `signing` assertions. You should not add the Include Timestamps and Require Signing policy actions to a virtual service if the Require SAML Token action is already applied.

Input Parameters

SAML Version **String.** Specifies the version of the WSS SAML Token to use (1.1 or 2.0).

Run-Time Behavior

When a service consumer sends a request that includes a SAML token to a virtual service, CloudStreams validates the SAML token to ensure it is valid. If the token is valid, Integration Server uses its included JAAS login modules to process the SAML assertion and map the client public key in the assertion to a valid Integration Server user.

If the service consumer invokes the virtual service without a SAML assertion in the request, then CloudStreams sends the following SOAP fault to the service consumer to indicate that the request does not match the security policy being enforced: `SAML Token missing in request`.

Prerequisites

In order to use a SAML token, CloudStreams requires that you:

- Determine which Security Token Services (STS) to trust. The STS generates the SAML tokens that clients will submit. The client can use any STS provider that generates SAML 1.1 or 2.0 tokens. The generated SAML token must:
 - Contain the certificate of the user/client (service consumer) in the assertion if Integration Server is to use Holder-of-Key (HOK) type SAML assertions.
 - Be signed by the STS.
- Identify the trusted STS to Integration Server. For more information, see "[Setting the STS Options](#)" on page 36.
- Provide a truststore alias that points to a truststore containing the issuer's certificate. For information about providing a truststore alias, see the section *Securing Communications with the Server* in the document *webMethods Integration Server Administrator's Guide*.
- If Integration Server is to process Holder-of-Key (HOK) type SAML assertions, which contain the client's public key, you must map the client's public key to an Integration Server user. For more information about configuring and mapping

client certificates, see the section *Authenticating Clients* in the document *webMethods Integration Server Administrator's Guide*.

To Identify a Trusted STS to Integration Server

To identify a trusted STS to Integration Server

1. Open the Integration Server Administrator if it is not already open.
2. In the Navigation panel, select **Security > SAML**.
3. Click **Add SAML Token Issuer**, set the following parameters and click **Save Changes**:

Option	Description
Issuer Name	Name of a SAML token issuer from which Integration Server should accept and process SAML assertions. Integration Server will reject SAML assertions from issuers not configured on this screen and will log a message to the Server log similar to the following: 2010-06-09 23:35:38 EDT [ISS.0012.0025E This value must match the value of the Issuer field in the SAML assertion.
Truststore Alias	Text identifier for the truststore, which contains the public keys of the SAML token issuer.
Certificate Alias	Text identifier for the certificate associated with the truststore alias.
Clock Skew	Clock difference between your Integration Server and the SAML token issuer.

Require Signing

Note: Dependency requirement: A policy that includes this action must also include the Identify Consumer action.

This action requires that a request's XML element (represented by an XPath expression) be signed. This action supports WS-SecurityPolicy 1.2 and cannot be used with REST services or connector virtual services.

Prerequisites:

1. Configure Integration Server: Set up keystores and truststores in Integration Server (see the section *Securing Communications with the Server* in the document *webMethods Integration Server Administrator's Guide*).

2. Configure CloudStreams: In the Integration Server Administrator, navigate to **Solutions > CloudStreams > Administration > General** and complete the **IS Keystore Name**, **IS Truststore Name** and **Alias (signing)** fields, as described in "[Setting the General Options](#)" on page 28). CloudStreams uses the signing alias specified in the **Alias (signing)** field to sign the response.

When this policy action is set for the virtual service, CloudStreams validates that the requests are properly signed, and provides signing for responses. CloudStreams provides support both for signing an entire SOAP message body or individual elements of the SOAP message body.

CloudStreams uses a digital signature element in the security header to verify that all elements matching the XPath expression were signed. If the request contains elements that were not signed or no signature is present, then CloudStreams rejects the request.

Note: You must map the public certificate of the key used to the sign the request to an Integration Server user. If the certificate is not mapped, CloudStreams returns a SOAP fault to the caller.

Input Parameters

Element Required to be Signed An XPath expression that represents the XML element that is required to be signed.

Namespace Prefix Optional. Right-click the action name and click **Add Namespace Prefix** if you want to specify the namespace prefix of the element required to be signed. Enter the namespace prefix in the following format:

xmlns:prefix-name

For example:

xmlns:soapenv

For more information, see the XML Namespaces specifications at <http://www.w3.org/TR/REC-xml-names/#ns-decl>.

See below for an example XPath element generated in the policy.

Example Generated XPath Element

The generated XPath element in the policy should look similar to this:

```
<sp:SignedElements xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
  <sp:XPath
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">//soapenv:Body</sp:XPath>
</sp:SignedElements>
```

Require SSL

This action ensures that requests are sent to the server using the HTTPS protocol (SSL). This action supports WS-SecurityPolicy 1.2 but can be used with both SOAP and REST services.

In addition, setting the **Client Certificate Required** parameter to True allows CloudStreams to verify the client sending the request.

Ensure that you specify an HTTPS port, as described in ["Setting the General Options" on page 28](#).

Input Parameters

Client Certificate Required

Specifies whether client certificates are required for the purposes of:

- Verifying the signature of signed SOAP requests or decrypting encrypted SOAP requests.
- Signing SOAP responses or encrypting SOAP responses.

Values:

- **True:** Requires client certificates. If a valid client certificate is not presented, CloudStreams rejects the message. Ensure that the Integration Server HTTPS port is configured to request or require a client certificate.
- **False:** Default. Does not require client certificates.

Require WSS Username Token

Note: Dependency requirement: A policy that includes this action must also include the Identify Consumer action.

Requires that a WSS username token and password be present in the SOAP message header to validate service consumers. This action supports WS-SecurityPolicy 1.2 and cannot be used with REST services or connector virtual services.

CloudStreams rejects requests that do not include the username token and password of an Integration Server user. CloudStreams only supports clear text passwords with this kind of authentication.

In the case where a consumer is sending a request with both transport credentials (HTTP basic authentication) and message credentials (WSS username or X.509 token), the message credentials take precedence over the transport credentials when Integration Server is determining which credentials it should use for the session. For more information, see ["Require HTTP Basic Authentication" on page 190](#).

Input Parameters

None.

Require X.509 Token

Note: Dependency requirement: A policy that includes this action must also include the Identify Consumer action.

Requires that a WSS X.509 token be present in the SOAP message header to validate service consumers. This action supports WS-SecurityPolicy 1.2 and cannot be used with REST services or connector virtual services.

In the case where a consumer is sending a request with both transport credentials (HTTP basic authentication) and message credentials (X.509 token or WSS username), the message credentials take precedence over the transport credentials when Integration Server is determining which credentials it should use for the session. For more information, see ["Require HTTP Basic Authentication" on page 190](#). In addition, you must ensure that the service consumer that connects to the virtual service has an Integration Server user account.

Input Parameters

None.

Throttling Traffic Optimization

Note: Dependency requirement: A policy that includes this action must also include the Identify Consumer action if the **Limit Traffic for Applications** option is selected.

This action limits the number of service invocations allowed during a specified time interval, and sends alerts to a specified destination when the performance conditions are violated.

Reasons for limiting the service invocation traffic include:

- To avoid overloading the back-end services and their infrastructure.
- To limit specific consumers in terms of resource usage (that is, you can use the Monitor Service Level Agreement action in addition to Throttling Traffic Optimization).
- To shield vulnerable servers, services, and even specific operations.
- For service consumption metering (for example, billable pay-per-use services).

Input Parameters

Soft Limit	<p>Number. Optional. Specifies the maximum number of invocations allowed per Interval before issuing an alert. Reaching the soft limit will not affect further processing of requests (until the Hard Limit is reached).</p> <p>The limit is reached when the total number of invocations coming from all the consumer applications (specified in the Limit Traffic for Applications field) reaches the limit.</p>
Hard Limit	<p>Number. Required. Specifies the maximum number of invocations allowed per Interval before stopping the processing of further requests and issuing an alert. Typically, this limit should be higher than the soft limit.</p> <p>The limit is reached when the total number of invocations coming from all the consumer applications (specified in the Limit Traffic for Applications field) reaches the limit.</p>
Alert Interval	<p>Number. Specifies the amount of time for the soft limit and hard limit to be reached.</p>
Alert Frequency	<p>Number. Specifies how frequently to issue alerts.</p> <ul style="list-style-type: none">■ Every Time: Issue an alert every time one of the specified conditions is violated.■ Only Once: Issue an alert only the first time one of the specified conditions is violated.
Alert Message for Soft Limit	<p>Optional. Specify a text message to include in the soft limit alert.</p>
Alert Message for Hard Limit	<p>Optional. Specify a text message to include in the hard limit alert.</p>
Send Data To	<p>Specify where to log the alerts.</p> <ul style="list-style-type: none">■ Database: Default. Logs the alerts in the CloudStreams Analytics database.

Note: Be sure to also select the **Database Publishing** option in Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Database**), as described in "[Setting the Database Options for Publishing Performance Metrics and Events](#)" on page 33.

- **Server Log:** Logs the alerts in the server log of the Integration Server on which CloudStreams is running.

Also choose a value in the **Log Level** field:

- **Info:** Logs error-level, warning-level, and informational-level alerts.
- **Warn:** Logs error-level and warning-level alerts.
- **Error:** Logs only error-level alerts.

Note: The Integration Server Administrator's logging level for CloudStreams should match the logging level specified for this action (in the Integration Server Administrator go to **Settings > Logging > Server Logger**).

Metrics Collection Level

The run-time performance metrics for a virtual service (which is invoked only in the inbound run-time scenario), are collected at the service level. That is, the metrics for all invocations of a single virtual services are aggregated together during your specified metrics publishing interval and then published.

In contrast, the metrics for a connector virtual service (which is invoked only in the outbound run-time scenario) can be collected at two different levels of metric collection:

- **Cloud Connector Service:** Remember that a single connector virtual service can be used by multiple cloud connector services. Select this option if you want to collect the metrics for the connector virtual service broken down by each separate cloud connector service that uses it. For example, if a connector virtual services is used by three cloud connector services, then this option will collect the metrics of that service separately, broken down by each of the three cloud connector services that use it.
- **Connector Virtual Service**(default): Select this option if you want to aggregate all the metrics for a single connector virtual service, even if it is used by multiple cloud connector services. For example, if a connector virtual service is used by three cloud connector services, then this option will collect the combined metrics for the connector virtual service by all three of the cloud connector services that use it.

Limit Traffic for Application

Right-click the action name and click **Add Limit Traffic for Application** if you want to specify the consumer application that this action applies to. You can select **Add Limit Traffic**

for **Application** multiple times to add multiple consumer applications.

Alert Email

Right-click the action name and click **Add Alert Email** if you want to send the monitoring alerts to an email address you specify in the **Email ID** field. You can select **Add Alert Email** multiple times to add multiple email addresses.

Note: Be sure to also set the email options in Integration Server Administrator (go to **Solutions > CloudStreams > Administration > Email**), as described in "[Setting the E-mail Options for Logging Payloads and Sending Performance Monitoring Alerts](#)" on page 31.

Validate Schema

This action validates all XML request and/or response messages against an XML schema referenced in the WSDL. This action cannot be used for connector virtual services.

CloudStreams can enforce this policy action for messages sent between services. When this policy is set for a virtual service, CloudStreams validates XML request messages, response messages, or both, against the XML schema referenced in the WSDL. Be sure to provide a schema.

Input Parameters

Validate SOAP Message(s)

Validates the request and/or response messages.

Note: Be aware that CloudStreams does not remove `wsu:Id` attributes that may have been added to a request by a consumer as a result of security operations against request elements (signatures and encryptions). In this case, to avoid schema validation failures you would have to add a Transform sub-step to the virtual service's In Sequence step so that the requests are passed to an XSLT transformation file that removes the `wsu:Ids`.

5 Deploying Virtual Services and Connector Virtual Services

■ Overview of Deployment	200
■ Before You Deploy Virtual Services or Connector Virtual Services	200
■ Deploying All Services in a CloudStreams Governance Project	201
■ Deploying a Single Service	202
■ Undeploying Services	202
■ What Happens When You Deploy a Service?	202

Overview of Deployment

You can deploy virtual services and Connector Virtual Services in two ways:

- Deploy all Virtual Services, and any custom Connector Virtual Services, that are contained in a particular CloudStreams Governance project.

OR

- Deploy a single Virtual Service or custom Connector Virtual Service. Generally, this is useful for testing purposes.

Note: CloudStreams does not support sharing of Connector Virtual Services, Virtual Services, and Policies across nodes in a clustered setup. These artifacts should be manually deployed to a clustered node as needed.

Note: CloudStreams automatically deploys the *default* Connector Virtual Services, WmCloudStreams.SoapVS and WmCloudStreams.RestVS; there is no need for you to deploy them.

When you execute the deployment operation, CloudStreams will immediately deploy the service(s) and the following items to the CloudStreams server target(s) that you specify:

- The policies of each service, as well as any other artifacts that you associated with the services when you created them.
- The VSD (virtual service definition) of each virtual service or Connector Virtual Service.

When you deploy a Virtual Service or a Connector Virtual Service, CloudStreams generates an XML document called a *virtual service definition (VSD)*. The VSD defines the virtual service or Connector Virtual Service for CloudStreams, and contains all the resources required to deploy the service to a CloudStreams server, including the policy that applies to the service. You cannot edit the VSD, but you can view it in the **Advanced** page in the Properties view of each service.

Before You Deploy Virtual Services or Connector Virtual Services

You should:

- Ensure that all policies of the services are Active; the value of the **Status** field in the Properties view of each policy should be **Active**. If it is not, right-click the policy name in the CloudStreams Governance view and click **Active**. You will not be allowed to activate a policy unless all of its action parameters have been set.

- Ensure that at least one CloudStreams server target has already have been defined, as described in ["Defining and Managing a CloudStreams Server Target" on page 59](#).
- Ensure that each target's specified deployment URL is active and the user credentials of Integration Server are valid.
- Ensure that you specify an HTTP or HTTPS port, as described in ["Setting the General Options" on page 28](#).
- Test the server connection by going to the Software AG Designer menu, selecting **Window > Preferences > Software AG > CloudStreams Servers**, and clicking the **Test** button. If the connection is not active and valid, activate the deployment endpoint and modify the user credentials as required.

Deploying All Services in a CloudStreams Governance Project

When you deploy a CloudStreams Governance project, all virtual services in the project are deployed at once. If any custom Connector Virtual Services are defined in the project, they are deployed too.

When you deploy a project, Integration Server automatically creates a package to support the project. The package name is the same as your project name. This package includes startup/shutdown services to manage the registration of the virtual service definitions (VSDs) when Integration Server restarts.

To deploy all services within a project

1. Open Software AG Designer and display the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Governance view, right-click the CloudStreams Governance project you want to deploy, and click **Deploy**.
3. In the Deploy dialog box, choose one or more CloudStreams server targets to which to deploy the services and click **OK**.

Note: If only one CloudStreams server target is available to select, this dialog box will not appear.

The services are immediately deployed to the following location:

```
IntegrationServer_directory \instances \instance_name \packages  
\<PackageName>\<ProjectName> \config \proxies \VirtualService
```

If CloudStreams encounters a problem deploying or redeploying a service, it displays a failure message in the deployment log at the bottom of the page. In this case, the CloudStreams administrator should take corrective action and redeploy the service. For more information, see ["What Happens When You Deploy a Service?" on page 202](#).

If one of the virtual services fails deployment, you do not need to redeploy the entire project; just correct the problem with the failed virtual service and then redeploy it separately.

Deploying a Single Service

To deploy a single virtual service or Connector Virtual Service

1. OpenSoftware AG Designer and display theCloudStreams Development perspective.
2. Right-click the service name and click **Deploy** in the context menu.
3. In the Deploy dialog box, choose one or more CloudStreams server targets to which to deploy the service and click **OK**.

Note: If only one CloudStreams server target is available to select, this dialog box will not appear.

The service is immediately deployed to the following location:

```
IntegrationServer_directory \instances\instance_name \packages  
\<PackageName><ProjectName \config\proxies\VirtualService
```

If CloudStreams encounters a problem deploying or redeploying a service, it displays a failure message in the deployment log at the bottom of the page. In this case, the CloudStreams administrator should take corrective action and redeploy the service. For more information, see "[What Happens When You Deploy a Service?](#)" on [page 202](#).

Undeploying Services

To undeploy a virtual service or Connector Virtual Service

1. OpenSoftware AG Designer and display theCloudStreams Development perspective.
2. In theCloudStreams Governance view, expand theCloudStreams Governance project that contains the services you want to undeploy.
3. Right-click either the project name (to undeploy all services) or the service name and click **Undeploy** in the context menu.

CloudStreams immediately undeploys the service(s).

What Happens When You Deploy a Service?

When you deploy virtual services or Connector Virtual Services to CloudStreams targets, keep the following points in mind.

Policy Validation

When you deploy a service, CloudStreams automatically validates the service's policy (or policies) to ensure that all action dependencies are properly met. CloudStreams will warn you of any violation, and you will need to correct the violations before deploying the service. For more information about dependencies, see ["The Policy Actions" on page 173](#).

Policy Conflict Resolution Rules

When the combined list of actions is evaluated, policy conflicts can arise due to multiple authentication or authorization actions. Conflicts between these actions are resolved as follows.

Action Type	Resolution
Authentication actions: <ul style="list-style-type: none"> ■ Require HTTP Basic Authentication ■ Require WSS Username Token ■ Require X.509 Token ■ Require SAML Token 	One occurrence of each authentication action that appears in the list is chosen.
Authorization actions: <ul style="list-style-type: none"> ■ Authorize User 	All occurrences of authorization actions are chosen.

The Policy Conflict Resolution rules evaluate the actions from all policies in the following order. The rules choose the actions for the effective policy as follows:

Order	Action	Result Chosen for Effective Policy
1	Require SSL	One occurrence of the action is chosen if one of the actions has its <code>Client Certificate Required</code> parameter set to Yes.
2	Require HTTP Basic Authentication	One occurrence of the action is chosen.
3	Require WSS Username Token	One occurrence of the action is chosen.

Order	Action	Result Chosen for Effective Policy
4	Require WSS X.509 Token	One occurrence of the action is chosen.
5	Require WSS SAML Token	One occurrence of the action is chosen.
6	Require Signing	One occurrence of the action is chosen.
7	Require Encryption	One occurrence of the action is chosen.
8	Require Timestamps	One occurrence of the action is chosen.
9	Identify Consumer	One occurrence of the action is chosen. CloudStreams evaluates the consumer identifiers in all the actions and selects the action with the highest-priority consumer identifier, as follows: 1.) IP address 2.) Host name 3.) HTTP Authentication token 4.) WS-Security token 5.) Custom identification token 6.) Consumer certificate 7.) User ID.
10	Authorize User	All occurrences of the action are chosen.
11	Validate Schema	If at least one occurrence of the action is configured to validate requests, and at least one occurrence of the action is configured to validate responses, then an action is constructed to validate both requests and responses.
12	Log Invocations	All occurrences of the action are chosen.
13	Monitor Service Performance	All occurrences of the action are chosen.
14	Monitor Service Level Agreement	All occurrences of the action are chosen.
15	Throttling Traffic Optimization	All occurrences of the action are chosen.

What if You Need to Modify Deployed Services?

If you need to modify a virtual service that is already deployed, you must redeploy it after you modify it. CloudStreams does not monitor for updates to deployed assets. If you make changes to a virtual service's processing steps, for example, you must manually redeploy the virtual service to put those changes into effect.

6 CloudStreams Analytics

■ Overview	208
■ Installation and Configuration Tasks	210
■ Using the CloudStreams Analytics Dashboard	220

Overview

CloudStreams can publish key performance indicator (KPI) metrics and run-time events to the CloudStreams Analytics dashboard. The dashboard enables you to view and analyze the KPI metrics and run-time events of your CloudStreams connectors, cloud connector services and consumers.

The dashboard is powered by a MashApp called CloudStreams Analytics, which is powered by Software AG MashZone. CloudStreams Analytics uses data feeds from Software AG MashZone and uses the Integration Server's JDBC connection pool to publish this data at regular intervals.

Note: The datafeed of Software AG MashZone fetches data from an HTTP server. However, CloudStreams uses a WmTomcat server located inside Integration Server to retrieve metrics and events from the database and to send it back to Software AG MashZone. Currently, CloudStreams cannot fetch data from an HTTPS server. Thus, only the metrics and events generated by the outbound services are displayed in the CloudStreams Analytics dashboard. The metrics and events generated by *inbound* virtual services are *not* displayed in the CloudStreams Analytics dashboard, but this data is captured and is available in your destination database (which you configure in your JDBC connection pool, as described later in this chapter).

The dashboard displays the following analytic views:

View	Can be filtered by...
Connector performance and availability.	Provider and time period.
API usage of connectors (total requests and data volume).	Provider and time period.
Error trends by consumer and provider.	Provider and time period.
Service Level Agreement (SLA) violations and alerts, shown by consumer and provider.	Provider and time period.
Cloud connector service performance and availability.	Service and time period.
The results of service request transactions.	CloudStreams server target, event type and time period.

The types of KPI metrics and run-time events in the views are listed below.

The Run-Time Events

The types of run-time events that can be published are as follows:

Event Type	Description
Lifecycle	A Lifecycle event occurs each time a CloudStreams connector is started or shut down.
Transaction	A Transaction event occurs each time a cloud connector service is invoked (successfully or unsuccessfully).
Error	An Error event occurs each time an invocation of a cloud connector service results in an error.
Policy Violation	A Policy Violation event occurs each time an invocation of a cloud connector service violates a policy that was set for its associated virtual service.
Monitoring	CloudStreams publishes key performance indicator (KPI) metrics, such as the average response time, fault count, and availability of all connectors and connector services.

The Key Performance Indicator (KPI) Metrics

The types of KPI metrics that can be published are as follows:

Metric	Description
Availability	<p>Indicates whether the connector or service was available to the specified consumers in the current interval. For information about intervals, see "Setting the Database Options for Publishing Performance Metrics and Events" on page 33.</p> <p>A value of 100 indicates that the connector or service was always available. If invocations fail due to policy violations, this parameter could still be as high as 100.</p>
Average Response Time	The average amount of time it took the connector or cloud connector service to complete all invocations in the current interval. Response time is measured from the moment

Metric	Description
	CloudStreams receives the request until the moment it returns the response to the caller.
Fault Count	The number of failed invocations made by a connector, cloud connector service, or consumer in the current interval.
Success Count	The number of successful service invocations made by a connector, cloud connector service, or consumer in the current interval.
Total Request Count	The total number of requests made by a connector, cloud connector service, or consumer in the current interval.

To enable CloudStreams to publish KPI metrics and events to the CloudStreams Analytics dashboard, you must set the options in ["Setting the Database Options for Publishing Performance Metrics and Events"](#) on page 33.

Note: The metrics do not include metrics for failed invocations unless you set the `pg.PgMetricsFormatter.includeFaults` parameter to true in `IntegrationServer_directory\instances\instance_name\packages\WmCloudStreams\config\resources\wst-config.properties`. For more information, see ["Advanced Settings"](#) on page 289.

Installation and Configuration Tasks

Before you can use CloudStreams Analytics, you must perform the following tasks:

1. ["Installing MashZone and Starting the MashZone Server"](#) on page 211
2. ["Copying the Required Resource Files for CloudStreams Analytics"](#) on page 211
3. ["Creating the Schema and Tables Required by CloudStreams Analytics"](#) on page 212
4. ["Creating a JDBC Connection Pool for CloudStreams "](#) on page 215
5. ["Configuring the Data Feed"](#) on page 218
6. ["Importing the CloudStreams Analytics MashApp"](#) on page 218
7. ["Creating New Users for CloudStreams Analytics"](#) on page 219
8. ["Setting the Publishing Options for Performance Metrics and Events"](#) on page 219

Installing MashZone and Starting the MashZone Server

To install MashZone and start the server

1. Install Software AG MashZone from the Software AG Installer, as described in the document *Installing webMethods and Intelligent Business Operations Products* .

On the Configure page of the wizard, accept the default ports and then specify the desired system configuration (Small, Medium or Large).
2. Start the MashZone server.
3. In your browser, connect to `http://localhost:4080/mashzone/login.jsp` and log in using the user name `system` and the password `manager`.
4. Click **Open central user management**.
5. Click the **Licenses** tab under Central User Management.
6. Enter the license by uploading the MashZone license ZIP archive `MashZone_license.zip` (to do this, click the icon in the upper right corner of the screen).
7. Click the **User Management** tab under Central User Management.
8. Under the Users tab, click on the user **system**.
9. Click the **Privilege** tab and then click **License privileges**.
10. Click the **Granted** check box for **MashZone User** and **MashZone Viewer**.

Copying the Required Resource Files for CloudStreams Analytics

To copy the required resource files

1. Copy the file `SAG_HOME \CloudStreamsAnalytics\ mashapp \CloudStreams Analytics.mzp` into `MASHZONE_HOME \server\bin\work\work_mashzone_m \mashzone_data\importexport` folder.

where `MASHZONE_HOME` is the base directory that you specified when you installed MashZone (for example, `C:\SoftwareAG\ppmmashzone`).
2. Copy the contents of the folder `SAG_HOME \CloudStreamsAnalytics\ mashapp \assets\colorschemes` into the `MASHZONE_HOME server\bin\work \work_mashzone_m\mashzone_data\assets\colorschemes` folder.
3. Copy the contents of the folder `SAG_HOME \CloudStreamsAnalytics\ mashapp \resources` into the `MSAHZONE_HOME \server\bin\work\work_mashzone_m \mashzone_data\resources` folder.

Creating the Schema and Tables Required by CloudStreams Analytics

To create the schema and tables

1. Start the Database Component Configurator GUI as follows:

System	Action
Windows	On the Start menu, go to Program > Software AG > Tools > Database Component Configurator .
UNIX	Go to <i>Software AG_directory</i> and run the command <code>dbConfigurator.sh</code> .

2. In the **Action** area, in the **Type** list, click **create**.

The **create** action type creates the database components you select in the **Action** area, and lets you create a database user and storage.

3. Click **Component**, and in the text box next to **Component**, select **CloudStreamsEvents**.
4. In the **Version** field, select **Latest**.

The configurator will create the latest version of the database component.

5. In the **Connection** area, in the **RDBMS** field, select the RDBMS in which to create the database component.

A URL specific to the selected database appears in the **URL** field.

6. Modify the URL to point to the database where the CloudStreams events should be collected.

Sample URL formats for the DataDirect Connect JDBC 5.0 driver are displayed. Below is additional information for completing this field.

- For Oracle, if you are going to create storage and the Data Purge database component, you must specify the `sysLoginRole` connection option on the URL (for example: `;sysLoginRole=sysdba`).
- For DB2, if you are going to create database components in a schema other than the default schema for the specified database user, you must specify these connection options in the URL, where `AlternateID` is the name of the default schema used to qualify unqualified database objects in dynamically prepared SQL statements:

```
;AlternateId=schema ; "InitializationString=(SET CURRENT
PATH=current_path,schema )"
```

- For information about options supported by the DataDirect Connect JDBC 5.0 driver used by webMethods products, see *DataDirect Connect for JDBC User's Guide and Reference 5.0* in the *Software AG_directory / _documentation* directory.
7. In the **User ID** and **Password** fields, your entries depend on the task you are going to perform as follows:

If you are...	Specify...
Creating a database user and storage in Oracle or SQL Server	<p>The database user and password to create.</p> <p>Note:For SQL Server, the user will be created and a default schema named dbo will be assigned to that user.</p> <p>Note:For Oracle, do not use the SYSTEM user to create the database components in the SYSTEM schema.</p>
Creating a database user and storage in DB2 for LUW	The OS user to which to grant permissions, and the password for that user.

8. Select the **Create Tablespaces** check box. (To create tablespaces, you must have Admin privileges for the database.)

- Note:** This check box is labeled **Create Database and Database User** for SQL Server, and **Create Tablespaces and Grant Permissions to OS User** for DB2.
- Note:** For DB2 on Linux systems, tablespaces are created for each DB2 database. If you are creating webMethods database components in more than one DB2 database, either the tablespace directory or the tablespace names must be unique for each DB2 database.

9. In the **Admin ID** field, identify the database user or operating system user that has database administrator credentials to create the database user and storage. Supply the password for the user in the **Admin Password** field.
10. The next field and your entry depend on your RDBMS.

RDBMS	Field and Entry
Oracle or DB2 for LUW	In the Tablespace Directory field, identify the directory in which to create the tablespaces. For Oracle, use this field only if the DB_CREATE_FILE_DEST parameter is not set for your Oracle instance.

RDBMS	Field and Entry
SQL Server	In the Database field, specify the database to create.

11. The default tablespace names are as follows:

RDBMS	Data Storage	Index Storage	BLOB Storage
Oracle	WEBMDATA	WEBMINDX	WEBMDATA
SQL	Primary	Primary	Primary
DB2 for LUW	WEBMDATA	WEBINDX	WEBMDATA
DB2 for iSeries	Default	Default	Default

12. For Oracle or DB2 for LUW, you can select the **Use Custom Tablespace Names** check box and specify custom tablespace names in the fields. You must have Admin privileges to do this.

- For Oracle, the custom tablespace names will replace the defaults WEBMDATA and WEBMINDX.
- For DB2 for LUW, the custom tablespace names will replace the defaults WEBMDATA, WEBMINDX, and WEBMBLOB. You can also specify a custom name to use for the buffer pool for webMethods products (WEBMBUFF by default).

Note: webMethods products support all tablespace configurations deployed by users.

13. Click **Execute**.

The execution information is displayed on the **Results** tab and is written to the log file `dcc_yyyymmddHHMMss` in the `Software AG_directory \common\db\logs` directory. If the execution is successful, all required tables are available.

If you intend to run the configurator more than once, you can set the current field values as the defaults for subsequent runs by clicking **Save Settings as Default**. You can also export field values to .xml files by clicking **Export**, then later import the values from a file by clicking **Import Configuration**. In each case, the values for the two **Password** fields are not saved.

Creating a JDBC Connection Pool for CloudStreams

To create a JDBC connection pool for CloudStreams

1. Start Integration Server and open Integration Server Administrator.
2. Go to the **Settings > JDBC Pools** page.
3. On the **Settings > JDBC Pools** page, click **Create a new Pool Alias Definition** and complete the fields to point to the tables you created in "[Creating the Schema and Tables Required by CloudStreams Analytics](#)" on page 212, as follows.

Field	Entry
Alias Name	Specify a name for the connection pool.
Alias Description	Description for the pool.
Associated Driver Alias	Database driver to use.
Database URL	<p>URL for the database server. Sample URL formats for the DataDirect Connect JDBC 5.0 driver are displayed.</p> <p>Use the DataDirect Connect connection option <code>MaxPooledStatements=35</code> on the database URL. This connection option improves performance by caching prepared statements.</p> <p>For DB2, if Integration Server will connect to a schema other than the default schema for the specified database user, you must specify these connection options in the URL:</p> <pre><code>;AlternateId=schema ;"InitializationString=(SET CURRENT PATH=current_path,schema)" ;MaxPooledStatements=35</code></pre> <p><code>AlternateId</code> specifies the name of the default schema that is used to qualify unqualified database objects in dynamically prepared SQL statement.</p>
User ID	Database user for Integration Server to use to communicate with the database.
Password	Password for the database user.

Field	Entry
Minimum Connections	<p>Minimum number of connections the pool must keep open at all times.</p> <p>If you use this pool alias for more than one function, each pool instance keeps the specified number of connections open. For example, if you specify keeping at least 3 connections open, and the IS Core Audit Log and the Document History database components both use this pool, the pool keeps a total of 6 connections open: 3 for the IS Core Audit Log pool instance and 3 for the Document History pool instance.</p> <p>If your logging volume has sudden spikes, you can improve performance by making sure the connections needed to handle the increased volume open quickly. You can minimize connection startup time during spikes by setting this value higher, so that more connections remain open at all times.</p>
Maximum Connections	<p>Maximum number of connections the pool must keep open at all times.</p> <p>Calculate this value as part of the total possible number of connections that could be opened simultaneously by all functions and applications that write to the database. Make sure the total number does not exceed the database's connection limit. If one of the applications opens more connections than the database allows, the database will reject subsequent requests for connections from any application.</p>
Idle Timeout	<p>Period of time, in milliseconds, the pool can keep an unused connection open. After the specified period of time, the pool closes unused connections that are not needed to satisfy the Minimum connections value.</p>

Note: You should also consider setting *connection* timeouts as described in "[SQL Statement Execution and Connection Timeouts](#)" on page 217.

4. Click **Save Settings**.
5. Click **Return to JDBC Pool Definitions**.
6. In the **Functional Alias Definitions** table, locate the functional name **CloudStreams**.
7. Click **Edit** under the **Edit Association** column.

8. In the **JDBC Functional Alias** field under **Associated Pool Alias**, select the connection pool alias you created.
9. Make sure Integration Server can connect to the database by clicking **Test Connection**.
10. Click **Save Settings**.

SQL Statement Execution and Connection Timeouts

The CloudStreams event-sending component relies on the JDBC pools connection properties for configuring the DataDirect driver properties.

It is expected that any DataDirect driver settings will be configured via property settings in the connection URL. Minimally, it is recommended that timeouts be configured for connections and SQL statement execution since by default the DataDirect driver uses "0" for its default settings (which implies that no timeouts are used).

Following is a sample connection URL with the property settings `QueryTimeout` and `LoginTimeout`:

```
jdbc:wm:oracle://<server>:<1521|
port>;serviceName=<value>;QueryTimeout=<seconds>;LoginTimeout=<seconds>
```

Note the following:

- The `LoginTimeout` property controls connection timeouts.

If `LoginTimeout` is not specified, the connection pool will use the value of the global Integration Server property `watt.server.jdbc.loginTimeout` to attempt to connect to the database.

The `watt.server.jdbc.loginTimeout` property is located in `IntegrationServer_directory\instances\instance_name\config\server.cnf`. This property sets the maximum time, in seconds, that the JDBC driver on Integration Server is to wait for a response while attempting to connect to a database. If Integration Server does not receive a response in the allotted time, it terminates the request, logs the error and moves on. The default value is 60 seconds.

Note: The DataDirect property `LoginTimeout` takes precedence over the global Integration Server property `watt.server.jdbc.loginTimeout`, which means you can override the global property on a case-by-case basis.

- The `QueryTimeout` property controls timeouts for queries and for inserting events.

Although the name `QueryTimeout` suggests its scope is limited to queries, it actually applies to *all* SQL statements, including INSERTs.

If The `QueryTimeout` property is not specified, there is no timeout for SQL statements executing on connections in this pool. So it is recommended that you specify a `QueryTimeout` value for this pool.

If you experience timeouts while inserting events, you should consider increasing the `QueryTimeout` property (and/or reducing the CloudStreams property `pg.jdbc.batchSize`, which is described below).

- The CloudStreams property `pg.jdbc.batchSize` controls the size of the events batch.

CloudStreams uses the optimized JDBC batch API to send its events to the relational database using prepared statements. The CloudStreams property `pg.jdbc.batchSize` controls how many events the sender can insert into a given batch (25 events by default). So, a batch of events is sent to the database every time the value of `pg.jdbc.batchSize` is reached. Increasing the batch size can improve database performance, but will also increase the statement execution time.

The `pg.jdbc.batchSize` property is located in:

```
IntegrationServer_directory \instances \instance_name \packages \WmCloudStreams
\config \resources \wst-config.properties
```

Additionally, if more than 3 seconds elapse since the last batch was sent, any received events will be sent to the database, even if the batch threshold has not been reached. This ensures there is not a substantial delay between the time an event is created and when it is inserted into the table.

Configuring the Data Feed

To configure the data feed

1. Open the file `config.xml` located in the `MASHZONE_HOME \server \bin \work \work_mashzone_m \mashzone_data \resources` folder.
2. Set the `host` and `port` to point to the Integration Server's host and port.
The default host and port is `localhost` and `5555`, respectively.

Importing the CloudStreams Analytics MashApp

To import the CloudStreams Analytics MashApp

1. Login to MashZone using your credentials. If you selected the default MashZone administrator credentials during installation, the default credentials are:
 - User name: `system`
 - Password: `manager`
2. Go to **Administration > Licensing** and add the License Key.
3. Go to **Administration > Import/Export/Delete** and click the **Import** button.
4. In the Select File dialog box, select **CloudStreams Analytics.mzp** and click **OK**.

5. Go to the MashZone Home page, where you should see the MashApp named **CloudStreams Analytics**.
6. Verify the imported MashApp as follows:
 - a. Login to MashZone using your credentials. If you selected the default MashZone administrator credentials during installation, the default credentials are: User name is `system` and Password is `manager`.
 - b. Click the **MashApps** tab and select the **CloudStreams Analytics** MashApp.
 - c. Double-click **CloudStreams Analytics** to open the MashApp.
 - d. You should see the **Cloud Connector Performance Summary** tab.

Creating New Users for CloudStreams Analytics

To enable users to access CloudStreams Analytics, you must create those users in MashZone.

You can create users using the following procedure, or you can import users and user groups from your LDAP system to MashZone and update the data of already imported LDAP users. For details about importing users from your LDAP system, see the chapter "Import LDAP Users and User Groups" chapter in the *Software AG MashZone Online Help*.

Note: Prerequisite: You must have MashZone Administrator privileges.

To create a new user in MashZone

1. Login to MashZone using your credentials. If you selected the default MashZone administrator credentials during installation, the default credentials are:
 - User name: `system`
 - Password: `manager`
2. Click the **User Management Tab > User** tab.
3. Create a new user.
4. The new user can login to CloudStreams Analytics using the login and password entered here.

Setting the Publishing Options for Performance Metrics and Events

CloudStreams uses the Integration Server's JDBC connection pool to publish the performance metrics and events. To enable CloudStreams to publish performance metrics and events to the CloudStreams Analytics dashboard, you must set the options described in "[Setting the Database Options for Publishing Performance Metrics and Events](#)" on page 33.

Using the CloudStreams Analytics Dashboard

After you have completed the "[Installation and Configuration Tasks](#)" on page 210, login to the dashboard as follows.

To login to the dashboard

1. Login to MashZone using your credentials. If you selected the default MashZone administrator credentials during installation, the default credentials are:
 - User name: `system`
 - Password: `manager`
2. Click the **MashApps** tab and select the **CloudStreams Analytics** MashApp.
3. Double-click **CloudStreams Analytics** to open the MashApp.
4. The MashApp's first view, called Cloud Connector Performance Summary, is displayed.

The dashboard provides the following views:

- "[Cloud Connector Performance Summary View](#)" on page 220
- "[Cloud Connector API Usage View](#)" on page 221
- "[Cloud Connector Performance View](#)" on page 222
- "[Cloud Connector Error View](#)" on page 224
- "[Cloud Connector \"Service Level Agreement\" \(SLA\) View](#)" on page 225
- "[Service View](#)" on page 226
- "[Transaction View](#)" on page 228

Cloud Connector Performance Summary View

This view provides a summary of all CloudStreams connectors' performance and availability. It lists all connectors and their availability, performance and status.

To use the Cloud Connector Performance Summary view

1. Open the CloudStreams Analytics MashApp, as described in "[Using the CloudStreams Analytics Dashboard](#)" on page 220.

By default, the Cloud Connector Performance Summary view is the first view displayed in the dashboard.
2. This view displays the following information about all CloudStreams connectors installed in CloudStreams:

Field	Description
Status	There are two possible statuses: <ul style="list-style-type: none"> ■ Error: The reason for the error appears in the Performance and Availability Summary column. ■ OK: The connector is operating normally.
Cloud Connector	The cloud connector's name, for example Salesforce.
Performance and Availability Summary	The reason for the connector's status. One reason for Error status would be Availability of less than 80%.
Performance	The average response time (in milliseconds) that it took the connector to complete all invocations in the current interval. Response time is measured from the moment CloudStreams receives the request until the moment it returns the response to the caller. For information about intervals, see " Setting the Database Options for Publishing Performance Metrics and Events " on page 33.
Availability	The percentage of time that the connector was available during the current interval. A value of 100 indicates that the connector was always available. Availability of less than 80% results in Error status.

Cloud Connector API Usage View

This view shows the API usage of one or all connectors, filtered by cloud provider and time period.

To use the Cloud Connector API Usage view

1. Open the CloudStreams Analytics MashApp, as described in "[Using the CloudStreams Analytics Dashboard](#)" on page 220.
2. Click the **Usage** tab in the dashboard.
3. Filter the metrics you want to view as follows:

Filter	Description
Cloud Provider	Choose one cloud provider or choose All .

Filter	Description
Period	The time period for which to view API usage: <ul style="list-style-type: none"> ■ Last 90 Days: The previous 90 days, including today. ■ Last 30 Days: The previous 30 days, including today. ■ Last 7 Days: The previous 7 days, including today. ■ Last 1 Day: The previous 24 hours. ■ Last 1 Hour: The previous hour.

4. This view displays the following information:

Field	Description
Cloud Providers by Number of Requests	Shows each provider's percentage of the total number of requests received during the specified time period.
Cloud Providers by Data Volume	Shows each provider's percentage of the request/response payload size sent and received during the time period.
Top 5 - Cloud Connector Services by Number of Requests	Shows the five cloud connector services that received the most requests during the specified period. Point to a bar to view the exact number.
Top 5 - Cloud Connector Services by Data Volume	Shows the five cloud connector services that received/sent the largest amount of request/response payload data (in Kilobytes). Point to a bar to view the exact number.
Top 5 - Consumers by Number of Requests	Shows the five consumers that sent the most requests. Point to a bar to view the exact number.
Top 5 - Consumers by Data Volume	Shows the five consumers that sent the largest amount of request/response payloads (in Kilobytes). Point to a bar to view the exact number.

Cloud Connector Performance View

This view shows the following metrics, filtered by cloud provider and time period:

- Performance (average response time) of cloud connector services. Performance is shown by provider, consumer and individual cloud connector services.

Average response time is the average amount of time (in milliseconds) it took the cloud connector service to complete all invocations in the current interval. Response time is measured from the moment CloudStreams receives the request until the moment just before it returns the response to the caller. By default, the average response time does not include metrics for failed invocations. For information about intervals, see ["Setting the Database Options for Publishing Performance Metrics and Events" on page 33](#).

- Availability of cloud connector services. Availability is shown by consumer and individual connector services.

Availability indicates the percentage of time that the cloud connector service was available during the current interval. A value of 100 indicates that the cloud connector service was always available. Only the time when the service is unavailable counts against this metric. If invocations fail due to policy violations, this parameter could still be as high as 100. For information about intervals, see ["Setting the Database Options for Publishing Performance Metrics and Events" on page 33](#).

To use the Cloud Connector Performance view

1. Open the CloudStreams Analytics MashApp, as described in ["Using the CloudStreams Analytics Dashboard" on page 220](#).
2. Click the **Performance** tab in the dashboard.
3. Filter the metrics you want to view as follows:

Filter	Description
Cloud Provider	Choose one cloud provider or choose All .
Period	The time period for which to view performance: <ul style="list-style-type: none"> ■ Last 90 Days: The previous 90 days, including today. ■ Last 30 Days: The previous 30 days, including today. ■ Last 7 Days: The previous 7 days, including today. ■ Last 1 Day: The previous 24 hours. ■ Last 1 Hour: The previous hour.

4. This view displays the following information:

Field	Description
Cloud Provider Performance Trends	Shows the average response time for each provider.

Field	Description
Cloud Consumer Performance Trends	Shows the average response time for each consumer.
Top 5 - Cloud Connector Services by Average Response Time	Shows the average response time of the five fastest cloud connector services.
Slowest 5 - Cloud Connector Services by Average Response Time	Shows the average response time of the five slowest cloud connector services.
Top 5 - Cloud Connector Services by Availability	Shows the three cloud connector services with the highest availability.
Lowest 5 - Cloud Connector Services by Availability	Shows the three cloud connector services with the lowest availability.

Cloud Connector Error View

This view shows service request error trends.

To use the Cloud Connector Error view

1. Open the CloudStreams Analytics MashApp, as described in "[Using the CloudStreams Analytics Dashboard](#)" on page 220.
2. Click the **Error** tab in the dashboard.
3. Filter the metrics you want to view as follows:

Filter	Description
Cloud Provider	Choose one cloud provider or choose All .
Period	The time period for which to view performance: <ul style="list-style-type: none"> ■ Last 90 Days: The previous 90 days, including today. ■ Last 30 Days: The previous 30 days, including today. ■ Last 7 Days: The previous 7 days, including today. ■ Last 1 Day: The previous 24 hours. ■ Last 1 Hour: The previous hour.

4. This view displays the following information:

Field	Description
Cloud Provider Error Trends	Shows the number of service request errors returned by each provider.
Consumer Error Trends	Shows the number of service request errors encountered by each consumer.
Top 5 - Services by Fault Count	Shows the five cloud connector services that returned the highest number of faults. The fault count indicates the number of failed requests for the service for the current interval. For information about intervals, see "Setting the Database Options for Publishing Performance Metrics and Events" on page 33.
Lowest 5 - Services by Fault Count	Shows the five cloud connector services that returned the fewest number of faults.

Cloud Connector "Service Level Agreement" (SLA) View

This view shows the violations and alerts that were generated by the "Monitor Service Level Agreement" action, filtered by cloud provider and time period. (You can include the "Monitor Service Level Agreement" action in the policies for virtual services and Connector Virtual Services; for details, see ["Monitor Service Level Agreement \(SLA\)"](#) on page 184.)

To use the Cloud Connector "Service Level Agreement" (SLA) view

1. Open the CloudStreams Analytics MashApp, as described in ["Using the CloudStreams Analytics Dashboard"](#) on page 220.
2. Click the **SLA** (Service Level Agreement) tab in the dashboard.
3. Filter the metrics you want to view as follows:

Filter	Description
Cloud Provider	Choose one cloud provider or choose All .
Period	The time period for which to view performance: <ul style="list-style-type: none"> ■ Last 90 Days: The previous 90 days, including today. ■ Last 30 Days: The previous 30 days, including today.

Filter	Description
	<ul style="list-style-type: none"> ■ Last 7 Days: The previous 7 days, including today. ■ Last 1 Day: The previous 24 hours. ■ Last 1 Hour: The previous hour.

4. This view displays the following information:

Field	Description
SLA Violation Trends	Shows the number of Service Level Agreement (SLA) violations received by each cloud provider.
SLA Violations by Cloud Provider	Shows the total number of SLA violations received by each cloud provider.
SLA Violations by Consumer	Shows the total number of SLA violations sent by each consumer.
Top 5 - Services by SLA Violation	Shows the five cloud connector services that received the most SLA violations.
Top 5 - Services by Monitoring Alerts	Shows the five cloud connector services that received the most monitoring alerts generated by the SLA violations. (Remember that alerts may or may not be sent every time a violation occurs; this is controlled by the Alert Frequency parameter of the "Monitor Service Level Agreement" action.)
Monitoring Alerts by Event Source	Shows the number of monitoring alerts generated by the various run-time events.
SLA Violations by Event Source	Shows the number of SLA violations generated by the various run-time events.

Service View

This view shows the run-time performance of cloud connector services. It can be filtered by provider, connector and date range.

To use the Service View

1. Open the CloudStreams Analytics MashApp, as described in "[Using the CloudStreams Analytics Dashboard](#)" on page 220.
2. Click the **Service View** tab in the dashboard.
3. Filter the metrics you want to view as follows:

Filter	Description
Cloud Provider	Choose one cloud provider or choose All .
Cloud Connector	Choose one cloud connector or choose All .
Date Range	<p>Choose one of the following values or click Timeframe and choose the start and end dates from the calendars.</p> <ul style="list-style-type: none"> ■ Last 90 Days: The previous 90 days, including today. ■ Last 30 Days: The previous 30 days, including today. ■ Last 7 Days: The previous 7 days, including today. ■ Last 1 Day: The previous 24 hours. ■ Last 1 Hour: The previous hour.

4. This view displays the following information:

Field	Description
Provider Name	The cloud provider name.
Connector Name	The connector name.
Service Name	The cloud connector service name.
Total Requests	Total number of requests for the service, sorted in descending order (by the services requested the most).
Fault Count	Number of faults that the service returned.
Success Count	Number of successful requests for the service.

Field	Description
Average Response Time	Average response time for the service.
Total Bytes of Data	Total size of all request/response payloads.

Transaction View

This view shows the event results of all request transactions for cloud connector services. It can be filtered by provider, connector, service, event type and date range.

To use the Transaction View

1. Open the CloudStreams Analytics MashApp, as described in "[Using the CloudStreams Analytics Dashboard](#)" on page 220.
2. Click the **Transaction View** tab in the dashboard.
3. Filter the metrics you want to view as follows:

Filter	Description
Cloud Provider	Choose a cloud provider or choose All .
Cloud Connector	Choose a cloud connector or choose All .
Service Name	Choose a service or choose All .
Event Type	Choose any of the following event types or choose All : <ul style="list-style-type: none"> ■ Error: An Error event occurs each time a cloud connector service invocation results in an error. ■ Policy Violation: A Policy Violation event occurs each time a cloud connector service invocation violates a policy that was set for its associated virtual service. ■ Transaction: A Transaction event occurs each time a cloud connector service is invoked (successfully or unsuccessfully). ■ Monitoring: CloudStreams publishes key performance indicator (KPI) metrics, such as the average response time, fault count, and availability of all connectors and connector services.

Filter	Description
	<p>Note: To enable CloudStreams to publish these events, you must set the options described in "Setting the Database Options for Publishing Performance Metrics and Events" on page 33.</p>
Date Range	<p>Choose one of the following values or click Timeframe and choose the start and end dates from the calendars.</p> <ul style="list-style-type: none"> ■ Last 90 Days: The previous 90 days, including today. ■ Last 30 Days: The previous 30 days, including today. ■ Last 7 Days: The previous 7 days, including today. ■ Last 1 Day: The previous 24 hours. ■ Last 1 Hour: The previous hour.

4. This view displays the following information:

Field	Description
Provider Name	The cloud provider name.
Connector Name	The connector name.
Service Name	The cloud connector service that was requested.
Event Type	The event type that resulted (Error, Policy Violation, Transaction or Monitoring).
Session ID	The SOAP invocation session ID that generated the event.
Target Name	The CloudStreams server target that generated the event.
Event Create Timestamp	The event timestamps, sorted in descending order (the most recent timestamps first).
Error/Alert Source	The source of errors and alerts generated by Error, Policy Violation and Monitoring events.

Field	Description
Error/Alert Description	Descriptions of the errors and alerts generated by Error, Policy Violation and Monitoring events.
Lifecycle Status	The status generated by a Lifecycle event.

7 Creating Custom Cloud Connectors

■ Overview of Creating Custom Cloud Connectors	232
■ Creating and Managing CloudStreams Provider Projects	233
■ Creating Custom SOAP or REST Cloud Connectors	234
■ Configuring Custom SOAP Cloud Connectors	238
■ Configuring Custom REST Cloud Connectors	255
■ Support for multiple root JSON data format with anonymous collection	278
■ Viewing the Constraints Applied to Variables	279
■ Deploying Custom Cloud Connectors	280
■ Managing Custom Cloud Connectors	280
■ Managing the Custom Cloud Connector Packages	281

Overview of Creating Custom Cloud Connectors

Although CloudStreams provides predefined CloudStreams connectors for some SaaS applications (for example, Salesforce.com), you might want to create custom cloud connectors for other SaaS applications, which you can deploy to CloudStreams. You use the CloudStreams Development plug-in to create and configure custom cloud connectors.

The three high-level steps to create a custom cloud connector are:

1. Create a CloudStreams Provider project in which you will define the cloud connector.
2. Create the custom cloud connector by specifying:
 - The SaaS provider's WSDL (for SOAP connectors) or XSD (for REST connectors).
 - A metadata handler provided by CloudStreams, which provides the appropriate SOAP or REST data model for the SaaS provider.
 - The necessary cloud connector services, as follows:
 - For a SOAP-based SaaS provider, you should create at least one cloud connector service for each operation defined in the provider's WSDL. For example, for a query operation, you might create a cloud connector service to query accounts and another to query contacts.
 - For a REST-based SaaS provider, you should create at least one cloud connector service for each REST resource on which the connector can operate.
 - One or more run-time connections to the SaaS application.
3. Finally, you need to develop the governance functionality for the custom cloud connector, the same way you develop governance for a predefined CloudStreams connector. That is, you will:
 - a. Use the CloudStreams Development plug-in to configure:
 - Virtual services and connector virtual services (as described in "[Virtual Services](#)" on page 55).
 - Policies that provide governance capabilities for the virtual services and connector virtual services (as described in "[Policies](#)" on page 167).
 - The CloudStreams Analytics dashboard (as described in "[CloudStreams Analytics](#)" on page 207).
 - b. Use the Integration Server Administrator to set the CloudStreams configuration options (as described in "[CloudStreams Configuration Options](#)" on page 27).

Creating and Managing CloudStreams Provider Projects

You must create a CloudStreams Provider project in which you will define the cloud connector. You can perform several management tasks for Provider projects, such as exporting them to (and importing them from) your local file system, deleting them, and more.

Creating CloudStreams Provider Projects

To create a CloudStreams Provider project

1. In Software AG Designer, open the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. Click the CloudStreams Connectors view on the left side of the page.
3. Click **File > New > CloudStreams Provider Project** from the menu.
4. In the New CloudStreams Provider Project wizard, assign a name to the project and click **Finish**.

Field	Description
Project Name	<p>Assign a name to the new project using any combination of letters, numbers, and the underscore character. The name must be a valid resource name on your operating system. The name must not be null and cannot be an empty string.</p> <p>CloudStreams Provider project names are validated according to the same rules that apply to Integration Server package names. Keep the following guidelines in mind when naming new packages:</p> <ul style="list-style-type: none"> ■ Start all package names with an uppercase letter and capitalize the first letter of subsequent words (for example, PurchaseOrder). ■ Keep package names short. Use abbreviations instead of full names. For example, instead of ProcessPurchaseOrder, use ProcessPO. ■ Make sure the package name describes the functionality and purpose of the services it contains. ■ Control characters and special characters like periods (.), including: ■ Avoid creating package names with random capitalization (for example, cOOLPkgTest).

Field	Description
	<ul style="list-style-type: none"> ■ Avoid using articles (for example, “a,” “an,” and “the”) in the package name. For example, instead of TestTheService, use TestService. ■ Avoid using the prefix “Wm”. Integration Server and Designer use the “Wm” prefix for predefined packages that contain services, IS document types, and other files. ■ Avoid using control characters and special characters like periods (.) in a package name. The <code>watt.server.illegalNSChars</code> setting in the <code>server.cnf</code> file (which is located in the <code>IntegrationServer_directory \instances \instance_name \config</code> directory) defines all the characters that you cannot use when naming packages. Additionally, the operating system on which you run the Integration Server might have specific requirements that limit package names.

Managing CloudStreams Provider Projects

To manage a CloudStreams Provider project

1. In Software AG Designer, open the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. Click the CloudStreams Connectors view on the left side of the page.
3. Right-click the CloudStreams Provider project and select a menu item such as:
 - **Export:** Exports the Provider project to your local file system. Complete the wizard to indicate the export destination and click **Finish**. A project with deployed artifacts should be undeployed before being exported.
 - **Import:** Imports the Provider project from your local file system. Complete the wizard to indicate the import destination and click **Finish**.
 - **Delete:** Deletes the Provider project from the CloudStreams Development perspective.

Creating Custom SOAP or REST Cloud Connectors

The CloudStreams Development plug-in provides a wizard and editors you use to create and configure a cloud connector.

To create a custom cloud connector

1. Ensure that you have defined a CloudStreams server target to which you will deploy the cloud connector. For the procedure to do this, see ["Defining and Managing a CloudStreams Server Target" on page 59](#).
2. Ensure that you have created a CloudStreams Provider project in which you will define the cloud connector, as described in ["Creating and Managing CloudStreams Provider Projects" on page 233](#).
3. Right-click the **Connectors** folder and select **New Cloud Connector** (or right-click the CloudStreams Provider project you just created and select **New Cloud Connector**). Complete the fields in the New Cloud Connector wizard as follows.

Parameter	Description
Project	Click Browse and select the CloudStreams Provider project you just created.
Name	<p>Assign a name for the cloud connector. The name must start with an alpha character and may contain numerals and the following special characters:</p> <ul style="list-style-type: none"> ■ - (hyphen) ■ _ (underscore) ■ ' (apostrophe) ■ , (comma) ■ " (double quotes) ■ . (period) ■ & (ampersand) ■ () (parentheses) ■ ! (exclamation)
Version	<p>Assign a version identifier for the cloud connector. The version identifier is a required attribute that you should increment when a connector is modified to indicate that the cloud connector has been updated. This is a "public" version identifier that Designer shows to developers when it displays the list of connectors. The name may contain numerals and the following special characters:</p> <ul style="list-style-type: none"> ■ - (hyphen) ■ _ (underscore)

Parameter	Description
ID	<ul style="list-style-type: none"> ■ . (period) <p>Assign a unique identifier for the connector. This value must start with an alpha character and may contain numerals and the following special characters:</p> <ul style="list-style-type: none"> ■ - (hyphen) ■ _ (underscore) ■ . (period)
Provider Group	<p>Optional. You can specify that the connector should belong to a particular group. A group is a logical grouping of a collection of connectors.</p> <p>For example, if you have multiple connectors for the same provider, you might want to group them together for convenience. Suppose you have two connectors for the Amazon provider: one connector that is provided by Software AG, for example, the webMethods CloudStreams Amazon EC2 Connector, and another connector for the Amazon provider that was custom-built. You might want to group them together into a group called AmazonProvider.</p> <p>To specify a group for the connector, type a group name in this field or select an existing group name from the drop-down list. The name must start with an alpha character and may contain numerals and the following special characters:</p> <ul style="list-style-type: none"> ■ - (hyphen) ■ _ (underscore) ■ ' (apostrophe) ■ , (comma) ■ " (double quotes) ■ . (period) ■ & (ampersand) ■ () (parentheses) ■ ! (exclamation) <p>If you do not specify a group, the connector will be added to your CloudStreams Provider project. If you specify a group,</p>

Parameter	Description
	<p>the connector will be added to the <code><group_name></code> folder (under CloudStreams > Providers).</p> <p>Note: The group will be visible in the Integration Server Administrator's Connector List (under Solutions > CloudStreams > Providers) if at least one connector is in the deployed state.</p>
Metadata Handler	<p>CloudStreams provides two kinds of metadata handlers, which provide the appropriate SOAP data model for the SaaS provider. Click Browse and select a metadata handler.</p> <ul style="list-style-type: none"> ■ For a SOAP-based SaaS provider, select the SOAP metadata handler WsdConnectorMetadataHandler. ■ For a REST-based SaaS provider, select the REST metadata handler RestConnectorMetadataHandler.
Description	Optional. Type a description of the cloud connector.

- Click **Next** and create a cloud connector service based on the provider's WSDL or XSD that you specify as follows.

Note: Optionally, instead of clicking **Next** you can click **Finish**. If you choose to click **Finish**, you can create the cloud connector service later, using the Services Configuration page as described in "[Configuring Cloud Connector Services \(SOAP\)](#)" on page 249.

Parameter	Description
WSDL (SOAP only)	<p>Specify the SaaS provider's WSDL that CloudStreams will use to generate the document types by selecting either:</p> <ul style="list-style-type: none"> ■ File. Specify the WSDL file name, or click Browse and select the WSDL from your local file system. ■ URL. Specify the URL of the WSDL.
XSD (REST only)	<p>Specify the SaaS provider's XSD that CloudStreams will use to generate the document types by selecting either:</p> <ul style="list-style-type: none"> ■ Namespace. Specify the XSD's namespace (for example <code>myConnector_v1.doctypes</code>). Alternatively, click Browse and select a <code>doctype</code> that has already been generated in Integration Server.

Parameter	Description
	<ul style="list-style-type: none"> ■ XSD. Specify the URL of the XSD in the text field, or click Browse and select an XSD.
Package	<p>Assign a name for the DocTypes package, which will be deployed to the CloudStreams target server.</p> <p>The name must be a valid resource name on your operating system. The name must not be null and cannot be an empty string. The name cannot contain:</p> <ul style="list-style-type: none"> ■ Reserved words and characters that are used in Java or C/C++ (such as <i>for</i>, <i>while</i>, and <i>if</i>). ■ Digits as their first character. ■ Spaces. ■ Control characters and special characters like periods (.), including: ■ Characters outside of the basic ASCII character set, such as multi-byte characters. <p>CloudStreams will generate the document types from the provider's WSDL or XSD. However, if the Namespace option is chosen in the XSD field, then doctypes will not be generated.</p> <p>CloudStreams automatically generates a name for the namespace that will contain the provider's document types. The name's form is: <i>packageName.connector_ID.doctypes</i>. When additional services are added, CloudStreams adds the document types to this namespace.</p>
Description	Optional. Type a description of the cloud connector service.

5. Click **Finish**.

The new cloud connector is added to your CloudStreams Provider project.

In addition, if you completed page two of the wizard, CloudStreams creates a service for the cloud connector, based on the WSDL or XSD you specified in the wizard.

6. Next, perform the steps in "[Configuring Custom SOAP Cloud Connectors](#)" on page 238 or "[Configuring Custom REST Cloud Connectors](#)" on page 255.

Configuring Custom SOAP Cloud Connectors

After you create a cloud connector as described above, you need to configure it as follows.

- ["Creating Run-Time Connections \(SOAP\)" on page 239.](#)
- ["Configuring Cloud Connector Services \(SOAP\)" on page 249.](#)

Creating Run-Time Connections (SOAP)

You need to create one or more run-time connections to the SaaS provider.

To create a run-time connection (SOAP)

1. Start Software AG Designer and open the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Connectors view, expand your CloudStreams Provider project and click the cloud connector you just created (as described in ["Creating Custom SOAP or REST Cloud Connectors" on page 234](#)).

The Overview page is displayed, showing the general information you defined in the New Cloud Connector wizard.

3. Click the **Connections** link in the Connector Content section of the page (or click the **Connections** tab at the bottom of the page).

The Connections Configuration page is displayed.

4. Create a run-time connection by right-clicking the **Connections** node, selecting **Add Connection** and assigning a name and optional description to the connection. The name cannot contain special characters.

CloudStreams creates the connection in the **Connections** node, and by default it will contain a **Groups** node.

5. Right-click on the **Groups** node and click **Add Group**. The **New Group** window appears. Select one or more groups in the **New Group** window that should be allowed to access the connection.

The **Groups** node contains a default group called **connection**. You cannot select any other types for the connection group in the Configuration page. The Type, Name, and Description of the groups are also displayed. Select the Groups, click **OK**, and then complete the following fields in the Configuration section of the page.

Field	Description
Name	The group name. You may rename this group. Connection group names cannot have spaces or use special characters reserved by Integration Server or Designer. For more information about the use of special characters, see the <i>Designer Service Development</i> online help.
Group Type	You can select only one instance of each group type shown below for a provider connector's connection configuration.

Field	Description
	<ul style="list-style-type: none"> ■ oauth: This group will be used if you configure 'OAuth Tokens' in the Integration Server > CloudStreams > Administration > OAuth Tokens section. ■ oauth_v10a: Indicates that the group is defined with the details of OAuth V1.0a authentication scheme. ■ oauth_v20: Indicates that the group is defined with the details of OAuth V2.0 authentication scheme. ■ protocol: Indicates that the group is defined with the HTTP transport protocol that the connection will use. ■ connection: Indicates that the group is defined with the login endpoint to initiate communication with the SaaS provider. ■ requestHeaders: Indicates that the group is defined with the names of the HTTP request headers to include when sending the login request. ■ credentials: Indicates that the group is defined with a user account on the SaaS provider that the connection will use to connect to the SaaS provider. ■ aws_v2: Indicates that the user will use Signature Version 2 to sign Amazon Web Services Query API requests. ■ aws_s3: Indicates that the group is defined for the Amazon S3 authentication scheme and it uses the Access Key and the Secret Key of the client to authenticate the requests. ■ aws_v4: Indicates that the user will use Signature Version 4 to sign Amazon Web Services Query API requests. ■ custom: A user-defined group.

Description Optional. Type a description for the connection group.

Fields Based on the group type you selected above, CloudStreams displays the applicable fields for which you should specify values. Required fields are marked with an asterisk. Refer to the table below.

Based on the group type you selected above, CloudStreams displays the applicable fields for which you should specify values, as follows:

If the group type is...	The available fields are...
oauth	oauthAliasKey: The alias of a configured OAuth token.

If the group type is...	The available fields are...
oauth_v10a	<ul style="list-style-type: none"> <li data-bbox="553 321 1357 422">■ Consumer ID: The 'Consumer Key' issued by the Service Provider and used by the consumer to identify itself to the Service Provider. <li data-bbox="553 443 1357 510">■ Consumer Secret: A secret used by the Consumer to establish ownership of the 'Consumer Key'. <li data-bbox="553 531 1357 632">■ Access Token: A value used by the Consumer to gain access to the Protected Resources on behalf of the User, instead of using the User's Service Provider credentials. <li data-bbox="553 653 1357 720">■ Access Token Secret: A secret used by the Consumer to establish ownership of a given 'Access Token'.
oauth_v20	<ul style="list-style-type: none"> <li data-bbox="553 762 1369 829">■ Consumer ID: A 'client identifier' issued to the client to identify itself to the authorization server. <li data-bbox="553 850 1369 917">■ Consumer Secret: A secret matching to the 'client identifier'. <li data-bbox="553 938 1369 1005">■ Access Token: A token used by the client to make authenticated requests on behalf of the resource owner. <li data-bbox="553 1026 1369 1127">■ Instance URL: Optional field, used to specify a runtime host, if applicable. This may be required in some back ends like Salesforce. <li data-bbox="553 1148 1369 1732">■ Refresh Access Token: Option to refresh the 'Access Token'. OAuth 2.0 access tokens typically have a very short lifetime. When an access token expires, the OAuth profile does not automatically refresh the expired access token. Select this option if you want an expired access token to be refreshed automatically. If you select this option, you must also specify the relevant refresh parameters. The access token is refreshed whenever the session expires. Session expiration is handled according to the setting of the Session Management property in your connection. Note that if Session Management is set to "none", then you must manually modify the access token in the OAuth alias. (The Refresh Access Token option will not be applicable in this case.). Default is 'false'. If you want to refresh the 'Access Token' automatically, set Session Management to either 'fixed' or 'idle'. The Timeout value should be based on the backend settings. <li data-bbox="553 1753 1369 1852">■ Refresh Token: A token used by the client to obtain a new access token without having to involve the resource owner.

If the group type is... The available fields are...

- Refresh URL: The provider specific URL to refresh an 'Access Token'. This is required when 'Refresh Access Token' is enabled, that is, configured to 'true', and the Refresh URL Request is configured to 'URL Query String' or 'Body Query String'.
- Refresh URL Request: Options for sending the parameters in the 'Access Token' refresh request. The options are 'URL Query String', 'Body Query String', and 'Custom ESB Service'. Default is 'Body Query String'.
- URL Query String: The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the URL of the POST request. Example:

```
www.examplebackend.com/o/oauth2/token?grant_type=refresh_token&
client_id=842428530070-pubfebfqkqkj6t54m4ns6&client_secret=
4adQT95cAtUxWINbDxGP9SJ4&refresh_token=
1%2Fn072P4BXpuNObjCLUtiZTc4fMH6YersmxBIv8QN3bhw
```

Body Query String: The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the body of the POST request. Example:

```
POST /o/oauth2/token HTTP/1.1
Host: accounts.backend.com
Content-length: 163
content-type: application/x-www-form-urlencoded
client_secret4adQT95cAtUxWINbDxGP9SJ4&grant_type
=refresh_token&refresh_token=1%2Fn072P4BXpuNObjC
LUtiZTc4fMH6YersmxBIv8QN3bhw&client_id=
407408718192
```

- Custom ESB Service: If the backend requires the refresh request in a custom format, for example, requests which need more parameters than the ones specified by OAuth v2.0, or the backend uses some custom way of organizing parameters, or expects some other HTTP method request (other than POST), use the "Custom ESB Service" option.

Refresh Custom ESB Service: User implemented service for refreshing the 'Access Token'. This is required when the 'Custom ESB Service' option is selected as the 'Refresh URL Request'. This service must strictly conform to the specification:

```
- wm.cloudstreams.service.common.lookup.specs:
oauthTokenRefreshServiceSpec
```

If the group type is...	The available fields are...
	<p>Authorization Header Prefix: The prefix to be used with the 'Access Token' in the Authorization header. Options are 'Bearer' and 'OAuth'. Default is 'Bearer'.</p>
protocol	<ul style="list-style-type: none"> ■ elementCharSet: The encoding to use for the HTTP request line, headers, etc. ■ contentCharSet: The encoding to use for the request message. ■ httpVersion: The HTTP version (HTTP/0.9, HTTP/1.0 or HTTP/1.1. The default value for the connection factory is HTTP/1.1. ■ userAgent: The value to the connection configuration will send for the User-Agent request header. ■ useExpectContinue: If true, use the Expect/Continue HTTP/1.1 handshake and send the Expect request header. ■ waitForContinue: The number of milliseconds that the connection factory's client connection should wait for a "100 Continue" response from the server. ■ useStrictTransferEncoding: If true, the connection factory connection raise an exception if the "Transfer-Encoding" header is invalid. ■ useChunking: If true, use HTTP/1.1 chunking, using a chunk size that matches the socket buffer size. ■ flowServerRedirects: If true, follow server redirects. ■ serverRedirectMax: Maximum number of times to follow a server redirect.
connection	<ul style="list-style-type: none"> ■ providerURL: The native provider endpoint target for the connection configuration. The default configuration field provided with the connection factory is <code>cn.providerURL</code>. ■ minPoolConnections: The minimum number of socket connections to reserve for a connection configuration alias. ■ maxPoolConnections: The maximum number of socket connections to reserve for a connection configuration alias. ■ connectTimeout: The number of milliseconds a connection attempt will wait before giving up. (0 will wait indefinitely.)

If the group type is...	The available fields are...
	<ul style="list-style-type: none"> <li data-bbox="553 323 1369 430">■ readTimeout: The number of milliseconds in which the the client must read a response message from the server. (0 will wait indefinitely.) <li data-bbox="553 443 1369 550">■ useStaleChecking: If true, the connection factory performs additional processing to test the socket to see if it is still functional each time it is used. <li data-bbox="553 562 1369 669">■ retryCount: How many times should the connection factory attempt to execute a failed invocation. <li data-bbox="553 682 1369 789">■ retryOnResponseFailure: If true, the retry mechanism will be used for failed responses even if the request was sent successfully. <li data-bbox="553 802 1369 909">■ useTcpNoDelay: If true, do not use Nagles algorithm as a socket optimization technique. <li data-bbox="553 921 1369 1029">■ bufferSize: The size of the read and write socket buffers, in bytes. <li data-bbox="553 1041 1369 1148">■ reuseAddr: If true, the socket will be reused even if it is in TIME_WAIT due to a previous socket closure. <li data-bbox="553 1161 1369 1184">■ sessionToken: Session token for a stateful session. <li data-bbox="553 1197 1369 1304">■ webproxyAlias: The alias to a web proxy server configuration in Integration Server. <li data-bbox="553 1316 1369 1423">■ truststoreAlias: Alias for the Integration Server trust store configuration. <li data-bbox="553 1436 1369 1659">■ hostnameVerifier: Fully qualified class name that implements the Apache HC <code>org.apache.http.conn.ssl.X509HostnameVerifier</code> interface; helps guard against "man-in-the-middle" attacks. Also set the IS property <code>watt.security.cert.wmChainVerifier.trustByDefault</code> to false and ensure the outbound connector's connection configuration property <code>Hostname verifier</code> is set to its default value, <code>org.apache.http.conn.ssl.StrictHostnameVerifier</code>. <li data-bbox="553 1671 1369 1694">■ linger: Determines how quickly a socket should close.
requestHeaders	<ul style="list-style-type: none"> <li data-bbox="553 1692 1369 1833">■ names: An array of request header names to include for this connection configuration. The value should be a comma-delimited list of header names; for example <code>Content-Type, SOAPAction</code>. <li data-bbox="553 1845 1369 1919">■ values: An array of request header values to include for this connection configuration. The value should be comma-

If the group type is...	The available fields are...
credentials	<p data-bbox="602 304 1307 373">delimited list of values in the same order as the header names; for example, <code>text/xml,login</code>.</p> <ul style="list-style-type: none"> <li data-bbox="553 415 1258 485">■ username: The username credentials for the current connection configuration. <li data-bbox="553 506 1258 575">■ password: The password credentials for the current connection configuration <li data-bbox="553 596 1339 695">■ preemptiveAuth: If true, basic auth credentials will be included when a request is sent. (It will not wait for a 401 response challenge.) <li data-bbox="553 716 1347 785">■ authorizationType: The string identifying the authentication protocol scheme to use for the connection configuration. <li data-bbox="553 806 1315 875">■ domainName: The domain/security realm for the current connection configuration. <li data-bbox="553 896 1299 966">■ keystoreAlias: Alias for the Integration Server key store configuration. <li data-bbox="553 987 1315 1056">■ clientKeyAlias: Alias to reference a key inside a key store file.
custom	User-defined fields of a custom group.
Connection Management Properties	<ul style="list-style-type: none"> <li data-bbox="553 1161 1323 1302">■ Enable Connection Pooling: Whether connection pooling is enabled for a connection. Valid values: true: Connection pooling is enabled for this connection. false: Connection pooling is disabled for this connection. Default: true. <li data-bbox="553 1323 1356 1463">■ Initial Pool Size: The minimum number of connection objects that remain in the connection pool at all times, if connection pooling is enabled. When the connector creates the pool, it creates this number of connections. Default: 1. <li data-bbox="553 1484 1356 1715">■ Maximum Pool Size: The maximum number of connection objects that can exist in the connection pool if connection pooling is enabled. When the connection pool has reached its maximum number of connections, the connector will reuse any inactive connections in the pool, or, if all connections are active, it will wait for a connection to become available. Default: 10. <li data-bbox="553 1736 1339 1869">■ Pool Increment Size: The number of connections by which the pool will be incremented, up to the maximum pool size, if connection pooling is enabled and connections are needed. Default: 1.

<u>If the group type is...</u>	<u>The available fields are...</u>
--------------------------------	------------------------------------

- **Block Timeout (msec):** The number of milliseconds that Integration Server will wait to obtain a connection with the SaaS provider before the connection times out and returns an error. For example, you have a pool with Maximum Pool Size of 20. If you receive 30 simultaneous requests for a connection, 10 requests will be waiting for a connection from the pool. If you set the Block Timeout to 5000, the 10 requests will wait for a connection for 5 seconds before they time out and return an error. If the services using the connections require 10 seconds to complete and return connections to the pool, the pending requests will fail and return an error message stating that no connections are available. If you set the Block Timeout value too high, you may encounter problems during error conditions. If a request contains errors that delay the response, other requests will not be sent. This setting should be tuned in conjunction with the Maximum Pool Size to accommodate such bursts in processing. Default: 1000.
- **Expire Timeout (msec):** The number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool, if connection pooling is enabled. The connection pool will remove inactive connections until the number of connections in the pool is equal to the Initial Pool Size. The inactivity timer for a connection is reset when the connection is used by the connector. This setting should be tuned in conjunction with the Initial Pool Size to avoid excessive opening/closing of connections during normal processing. Default: 1000.
- **Startup Retry Count:** The number of times that the system should attempt to initialize the connection pool at startup if the initial attempt fails. The retry mechanism is invoked only when the connection is configured correctly, but the target server URL cannot be reached or a network issue occurs while attempting to initialize the connection. Default: 0 (a single attempt).
- **Startup Backoff Timeout (sec):** The number of seconds that the system should wait between attempts to initialize the connection pool. This value is ignored if Startup Retry Count is 0. Default: 10.
- **Session Management:** The type of timeout for the connection session. Select the type of session management that fits the requirements of your SaaS provider backend. It is

If the group type is...	The available fields are...
	<p>recommended that you set this field to idle if you want the CloudStreams server to manage the session. Valid values:</p> <p>none: The CloudStreams server does not manage session timeout. The session times out based on the settings of the SaaS provider backend.</p> <p>idle: If no activity happens for the time specified in Session Timeout, the session times out. If the session is not idle (it is used actively), the session will not timeout.</p> <p>The CloudStreams server takes into account the idle timeout. For example, if the session is idle for the time specified in Session Timeout, the server renews the session before making the service call.</p> <p>fixed: The session will timeout at a fixed time interval (specified in Session Timeout) irrespective of the session usage or current activity. The CloudStreams server renews the session as soon as the fixed timeout value expires.</p> <ul style="list-style-type: none"> ■ Session Timeout (min): The maximum number of minutes a session can remain active (in other words, how long you want the server to wait before terminating a session). The value should be equal to the session timeout value specified at the SaaS provider backend.
6.	Add additional connection groups, if desired, by right-clicking the Connections folder and selecting Add Group .
7.	If your provider requires a Login Sequence, configure it as follows:
a.	Right-click the Login Sequence node and select Add Operation . The Login Sequence/Logout Sequence will be enabled for SOAP based connections only if you have configured any Login/Logout Operation under Services. The Login Sequence/Logout Sequence will be enabled by default for REST based connections.
b.	Right-click the login operation under the Login Sequence node and select Add Mapping
c.	In the Configuration section of the page assign a name to the sequence (for example <code>XYZSoapService:login</code>), select the Login operation and optionally enter a description.
	A Mapping node is added under the login operation.
d.	Now the Configuration section of the page shows the document types that were generated for the login service WSDL for the request and response messages. Define the mappings by inserting values into the request message or extracting values from the response message as needed. For the Salesforce.com connector, for example, the username and password values are inserted into the request message by selecting the configured values that are defined in the managed

- connection page. Then, the session token and a server URL are extracted from the response message to be inserted into special connection factory fields. These fields are used when invoking any of the Simple or Complex operations.
- e. Right-click a request message field you want to map (for example, **HDR1:username**) and select **Set configuration**.
 - f. In the Set Configuration dialog that appears, complete the following fields.

Field	Description
Name	Assign a name to the mapping.
Type	<p>Select the data source type:</p> <ul style="list-style-type: none"> ■ Cookie: HTTP cookies. For example, the Salesforce.com connector inserts the session token from a valid login in the SOAP header of the business operations to be executed. Other providers do the same thing; however, the token is managed via cookies instead of the SOAP header. ■ Header: Request or response HTTP transport headers for the support connection factory implementation. ■ IData: IData is a variable name that is associated with the connection instance when the service is invoked. It is a subset of those configuration fields from the selected groups. Not all fields are eligible for use as a mapping step since their values may not be changed or have any useful meaning in this context. ■ Literal: A constant value. ■ Parameter: Primarily used for REST handlers. ■ Service: The Service type applies to the source, not to the target. Select Service if you want the source to call a given service to perform certain tasks and to map the output of the service to a target. The service must adhere to <code>wm.cloudstreams.service.common.lookup.specs:mapServiceSpec</code>. If an error occurs related to service validation or execution, CloudStreams throws a Mapping Exception. ■ XPATH: Enables you to define an XPath expression.
Value	If you selected the Service type, the Value field's drop-down list will display all services that conform to the specification <code>wm.cloudstreams.service.common.lookup.specs:mapServiceSpec</code> .

Field	Description
	When you choose a service, the Formatter Service Arguments Configuration dialog appears, where you can add/delete/reset the input details specified in the service and specify arguments.
	If you selected a different type, choose the appropriate value from the drop-down list (for example, the connection user name key field cr.username).

g. Click **OK**.

A green check mark is shown next to the request message field you just mapped.

8. Configure a **Logout Sequence** step in a similar manner. The Login Sequence/Logout Sequence will be enabled for SOAP based connections only if you have configured any Login/Logout Operation under Services. The Login Sequence/Logout Sequence will be enabled by default for REST based connections.
9. Next, configure your cloud connector services, as described below.

Configuring Cloud Connector Services (SOAP)

In this task, you will configure the cloud connector service you just created in the New Cloud Connector wizard. You will:

- Configure the operations for the cloud connector service and map their signatures.
- Select any abstract objects that will be needed by the service's Complex type operations (if any).

To configure a cloud connector service (SOAP)

1. Start Software AG Designer and open the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Connectors view, expand your CloudStreams Provider project and click the cloud connector you just created (as described in "[Creating Custom SOAP or REST Cloud Connectors](#)" on page 234).

The Overview page is displayed, showing the general information you defined in the New Cloud Connector wizard.

3. Click the **Services** link in the Connector Content section of the page (or click the **Services** tab at the bottom of the page).
 - If you already created a cloud connector service on page 2 of the New Cloud Connector wizard, the Services Configuration page will show your new service and the default nodes **Abstract Objects** and **Operations**.
 - If you did *not* create a cloud connector service in the New Cloud Connector wizard, create it now by right-clicking the **Services** node, selecting **Add Service**

and specifying the SaaS provider's WSDL that CloudStreams will use to generate the document types. For a description of the **Document Type Package** field, see ["Creating Custom SOAP or REST Cloud Connectors" on page 234](#).

4. Add any abstract objects that will be needed by the service's Complex type operations.

CloudStreams supports the abstract object types `businessObject` and `schemaObject`.

For a `businessObject`, perform the following steps:

- a. Click the **Abstract Objects** node.
- b. In the Configuration section of the page, in the **Type** field select **businessObject**.
- c. In the **Document Type** field, select the document name of the abstract object from the drop-down list.
- d. Next to the **Business > List** field, click the Add icon.

The Lookup Selection dialog appears, displaying the list of lookup services that implement the `boList` lookup type.

- e. Next to the **Business Object > Describe** field, click the Add icon. The Lookup Selection window appears, displaying the list of lookup services that implement the `boDescribe` lookup type.
- f. In the Lookup Selection dialog boxes, select the lookup services that you implemented for the abstract object and click **OK**.

The selected services will be shown in the **Service** table.

For a `schemaObject`, perform the following steps:

- a. Click the **Abstract Objects** node.
- b. In the Configuration section of the page, in the **Type** field, select **schemaObject**.
- c. In the **Document Name** field, select the document name of the abstract object from the drop-down list.
- d. Next to the **Schema Object > List** field, click the Add icon.

The **Lookup Selection** window appears, displaying the list of lookup services that implement the `soList` lookup type.

- e. In the Lookup Selection dialog boxes, select the lookup service that you implemented for the abstract object and click **OK**.

The selected service will be shown in the **Service** table.

5. Click the **Operations** node and select the operation type for each operation in the Configuration section of the page as follows:

Field	Description
Name	(Read-only field.) The operation name.

Field	Description
Type	<p>Select the type of the operation:</p> <ul style="list-style-type: none"> ■ SIMPLE: This is any operation that has all the information needed to build the messages for a connector's operation based solely on the WSDL. A simple operation neither requires dynamic signature updating nor uses an abstract type definition. ■ COMPLEX: This is any operation that requires additional efforts to capture the metadata needed to build messages and to invoke a provider's service because the WSDL by itself does not suffice. Not every provider will have Complex operations. ■ METADATA: Metadata operations are special operations provided by the provider which are used by the connector to capture the additional information that is needed for a complex operation. Not every provider will have Metadata operations. ■ LOGIN/LOGOUT: The Login and Logout operations are those that are invoked when a managed connection pool is enabled (login) or disabled (logout). Some providers require login/logout methods to be invoked in order to set up a session for the user to use when invoking any of the business operations. Other providers do not require them.
Hide	If you select this option for any operation, the operation will not be visible while configuring a Cloud connector service.
Description	Optional. Enter a description for the operation.

Alternatively, you can select the types for the operations individually, by clicking an operation and then selecting its type.

6. Add the operation's input parameters as follows.

You can add predefined parameters (if the operation has any) and you can add additional ones as well.

a. Right-click the operation and select **Add Parameters**.

The Configuration section of the page will display the operation's predefined parameters (if any).

b. If the operation has any predefined parameters, select the check box next to each one you want to add. Mandatory parameters are selected by default and you cannot un-select them.

c. Complete the following fields in the Configuration section.

Field	Description
Name	The parameter's name. (For predefined parameters, this is Read-only.)
Parameter Type	The parameter's type which determines how the parameter should be used, e.g., <code>URI_CONTEXT</code> , <code>QUERYSTRING_PARAM</code> or <code>CFG_PARAM</code> . (For predefined parameters, this is Read-only.) For a list of the supported parameter types, see "Parameter Types" on page 272 .
Default Value	You can provide a default value for any parameter by typing a value in the Default Value field. If the variable is null in the input pipeline, this default value will be used at run time. The value given at run time always takes precedence over the default value. However, if the existing default value is of type "fixed default", the overwrite will fail. <p>Note: Parameters have various data validation constraints that apply to them. These constraints are indicated by an icon next to each parameter. For information about these constraints, see "Viewing the Constraints Applied to Variables" on page 279.</p>
Description	Optional description of the parameter.
Required	Specifies whether the parameter is required.

- d. If you want to add additional parameters, click the **Add** button and complete the fields in the Configuration section (described above).
 - e. Click **OK**.
7. Specify the input/output mapping for an operation as follows:
 - a. Right-click the operation and select **Add Mapping**.
 - b. In the Configuration section of the page, right-click a parameter you want to map, select **Set Configuration** and complete the following fields.

Field	Description
Name	Assign a name to the mapping.
Type	Select the data source type: <ul style="list-style-type: none"> ■ Cookie: HTTP cookies. For example, the Salesforce.com connector inserts the session token

Field	Description
	<p>from a valid login in the SOAP header of the business operations to be executed. Other providers do the same thing; however, the token is managed via cookies instead of the SOAP header.</p> <ul style="list-style-type: none"> ■ Header: Request or response HTTP transport headers for the support connection factory implementation. ■ IData: IData is a variable name that is associated with the connection instance when the service is invoked. It is a subset of those configuration fields from the selected groups. Not all fields are eligible for use as a mapping step since their values may not be changed or have any useful meaning in this context. ■ Literal: A constant value. ■ Parameter: Primarily used for REST handlers. ■ Service: The Service type applies to the source, not to the target. Select Service if you want the source to call a given service to perform certain tasks and to map the output of the service to a target. The service must adhere to <code>wm.cloudstreams.service.common.lookup.specs:mapServiceSpec</code>. If an error occurs related to service validation or execution, CloudStreams throws a Mapping Exception. ■ XPath: Enables you to define an XPath expression.
Value	<p>If you selected the Service type, the Value field's drop-down list will display all services that conform to the specification <code>wm.cloudstreams.service.common.lookup.specs:mapServiceSpec</code>. When you choose a service, the Formatter Service Arguments Configuration dialog appears, where you can add/delete/reset the input details specified in the service and specify arguments.</p> <p>If you selected a different type, choose the appropriate connection group field from the drop-down list. The list of fields displayed in the drop-down list depends on the type of the connection group you configure in the Connection Configuration page (as described in "Creating Run-Time Connections (SOAP)" on page 239).</p>

8. Define global mappings, if required by your provider.

Global mappings are mapping statements that must be executed for all Simple and Complex type operations (if required by your provider). These global mapping steps are used by the provider to inject the session token into the SOAP header before invoking the service. Some providers might have configuration values that could be included for every operation. Other providers have configuration content in their SOAP header.

To specify global mapping, right-click the service name, select **Global Mapping** and complete the fields in the Configuration section of the page as follows:

Field	Description
Name	Assign a name for the global mapping.
Parameter Type	Select the parameter's type which determines how the parameter should be used, e.g., <code>URI_CONTEXT</code> , <code>QUERYSTRING_PARAM</code> or <code>CFG_PARAM</code> . For a list of the supported parameter types, see " Parameter Types " on page 272.
Required	Specify whether this is a required mapping.
Fixed	A Fixed mapping means its values cannot be changed at run time.
Mapping Key Type	Select the source mapping type: <ul style="list-style-type: none"> <input type="checkbox"/> Cookie <input type="checkbox"/> Header <input type="checkbox"/> IData <input type="checkbox"/> Literal <input type="checkbox"/> Parameter <input type="checkbox"/> Service <input type="checkbox"/> XPath <p>For descriptions, see the previous step.</p>
Mapping Key	Enter the source mapping key. For example, if your source mapping type is <code>IData</code> , you might enter the key <code>cr.username</code> .
Mapping Value Type	Select the target mapping type, which can be: <ul style="list-style-type: none"> <input type="checkbox"/> Cookie <input type="checkbox"/> Header

Field	Description
	<ul style="list-style-type: none"> ■ IData ■ Literal ■ Parameter ■ Service ■ XPath
Mapping Value Key	Enter the target mapping key. For example, if your target mapping type is XPath, you might enter the key <code>tns:login/tns:us</code> .
Document Name	The document type generated for that SOAP operation.

9. You can add/delete additional services in the Services page, as long as the connector is not deployed.
10. View the **Manifest** tab, which shows read-only fields containing attributes whose values were captured from your custom cloud connector definition. The version of the CloudStreams Development Plug-in (Created-By) is also displayed. For imported plug-ins created in a Designer release prior to 9.7, this information is not shown.

Configuring Custom REST Cloud Connectors

After you create a cloud connector as described above, you need to configure it as follows.

- ["Creating Run-Time Connections \(REST\)" on page 255.](#)
- ["Configuring Cloud Connector Services \(REST\)" on page 265.](#)

Creating Run-Time Connections (REST)

You need to create one or more run-time connections to the SaaS provider.

To create a run-time connection (REST)

1. Start Software AG Designer and open the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Connectors view, expand your CloudStreams Provider project and click the cloud connector you just created (as described in ["Creating Custom SOAP or REST Cloud Connectors" on page 234](#)).

The Overview page is displayed, showing the general information you defined in the New Cloud Connector wizard.

- Click the **Connections** link in the Connector Content section of the page (or click the **Connections** tab).

The Connections Configuration page is displayed.

- Create a run-time connection by right-clicking the **Connections** node, selecting **Add Connection** and assigning a name and optional description to the connection. The name cannot contain special characters.

CloudStreams creates the connection in the **Connections** node, and by default the connection will contain a **Groups** node.

- Right-click on the **Groups** node and click **Add Group** to select one or more groups in the **New Group** window that should be allowed to access the connection.

The **Groups** node contains a default group called **connection**. You cannot select any other types for the connection group in the Configuration page. The Type, Name, and Description of the groups are also displayed. Select the groups, click **OK** and complete the fields in the Configuration section of the page as follows.

Field	Description
Name	<p>The group name. You may rename this group.</p> <p>Connection group names cannot have spaces or use special characters reserved by Integration Server or Designer. For more information about the use of special characters, see the <i>Designer Service Development</i> online help.</p>
Group Type	<p>Only one instance of each group type shown below may be selected for a provider connector's connection configuration.</p> <ul style="list-style-type: none"> ■ oauth: This group will be used if you configure 'OAuth Tokens' in the Integration Server > CloudStreams > Administration > OAuth Tokens section. ■ oauth_v10a: Indicates that the group is defined with the details of authentication type OAuth V1.0a. ■ oauth_v20: Indicates that the group is defined with the details of authentication type OAuth V2.0. ■ protocol: Indicates that the group is defined with the HTTP transport protocol that the connection will use. ■ connection: Indicates that the group is defined with the login endpoint to initiate communication with the SaaS provider. ■ requestHeaders: Indicates that the group is defined with the names of the HTTP request headers to include when sending the login request.

Field	Description
	<ul style="list-style-type: none"> ■ credentials: Indicates that the group is defined with a user account on the SaaS provider that the connection will use to connect to the SaaS provider. ■ aws_v2: Indicates that the user will use Signature Version 2 to sign Amazon Web Services Query API requests. ■ aws_v4: Indicates that the user will use Signature Version 4 to sign Amazon Web Services Query API requests. ■ aws_s3: Indicates that the group is defined for the Amazon S3 authentication scheme and it uses the Access Key and the Secret Key of the client to authenticate the requests. ■ custom: A user-defined group.

Description Optional. Type a description for the connection group.

Fields Based on the group type you selected above, CloudStreams displays the applicable fields for which you should specify values. Required fields are marked with an asterisk. Refer to the table below.

Based on the group type you selected above, CloudStreams displays the applicable fields for which you should specify values, as follows:

If the group type is...	The available fields are...
oauth	oauthAliasKey: The alias of an OAuth token that was configured in Integration Server Administrator. For details, see <i>webMethods Integration Server Administrator's Guide</i> .
oauth_v10a	<ul style="list-style-type: none"> ■ Consumer ID: The 'Consumer Key' issued by the Service Provider and used by the consumer to identify itself to the Service Provider. ■ Consumer Secret: A secret used by the Consumer to establish ownership of the 'Consumer Key'. ■ Access Token: A value used by the Consumer to gain access to the Protected Resources on behalf of the User, instead of using the User's Service Provider credentials. ■ Access Token Secret: A secret used by the Consumer to establish ownership of a given 'Access Token'.

If the group type is...	The available fields are...
oauth_v20	<ul style="list-style-type: none"> <li data-bbox="570 317 1328 390">■ Consumer ID: A 'client identifier' issued to the client to identify itself to the authorization server. <li data-bbox="570 411 1390 447">■ Consumer Secret: A secret matching to the 'client identifier'. <li data-bbox="570 468 1333 541">■ Access Token: A token used by the client to make authenticated requests on behalf of the resource owner. <li data-bbox="570 562 1398 653">■ Instance URL: Optional field, used to specify a runtime host, if applicable. This may be required in some back ends like Salesforce. <li data-bbox="570 695 1386 1283">■ Refresh Access Token: Option to refresh the 'Access Token'. OAuth 2.0 access tokens typically have a very short lifetime. When an access token expires, the OAuth profile does not automatically refresh the expired access token. Select this option if you want an expired access token to be refreshed automatically. If you select this option, you must also specify the relevant refresh parameters. The access token is refreshed whenever the session expires. Session expiration is handled according to the setting of the Session Management property in your connection. Note that if Session Management is set to "none", then you must manually modify the access token in the OAuth alias. (The Refresh Access Token option will not be applicable in this case). Default is 'false'. If you want to refresh the 'Access Token' automatically, set Session Management to either 'fixed' or 'idle'. The Timeout value should be based on the backend settings. <li data-bbox="570 1304 1382 1377">■ Refresh Token: A token used by the client to obtain a new access token without having to involve the resource owner. <li data-bbox="570 1398 1409 1530">■ Refresh URL: The provider specific URL to refresh an 'Access Token'. This is required when 'Refresh Access Token' is enabled (configured to 'true') and the Refresh URL Request is configured to 'URL Query String' or 'Body Query String'. <li data-bbox="570 1551 1414 1684">■ Refresh URL Request: Options for sending the parameters in the 'Access Token' refresh request. The options are 'URL Query String', 'Body Query String', and 'Custom ESB Service'. Default is 'Body Query String'. <li data-bbox="570 1705 1349 1837">■ URL Query String: The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the URL of the POST request.

Example:

If the group type is...**The available fields are...**

```
www.examplebackend.com/o/oauth2/
token?grant_type=refresh_token&client_id=
842428530070-pubfebfqfgkj6t54m4ns6&client_secret=
4adQT95cAtUxWINbDxGP9SJ4&refresh_token=
1%2Fn072P4BXpuNObjCLUtiZTc4fMH6YersmxBIv8QN3bhw
```

- **Body Query String:** The refresh request parameters, for example, `refresh_token`, `grant_type`, and so on, and their values are sent as query strings in the body of the POST request.

Example:

```
POST /o/oauth2/token HTTP/1.1
Host: accounts.backend.com
Content-length: 163
content-type: application/x-www-form-urlencoded
client_secret4adQT95cAtUxWINbDxGP9SJ4&grant_type
=refresh_token&refresh_token=1%2Fn072P4BXpuNOb
jCLUtiZTc4fMH6YersmxBIv8QN3bhw&client_id
=407408718192
```

- **Custom ESB Service:** If the backend requires the refresh request in a custom format, for example, requests which need more parameters than the ones specified by OAuth v2.0, or the backend uses some custom way of organizing parameters, or expects some other HTTP method request (other than POST), use the "Custom ESB Service" option.

Refresh Custom ESB Service: User implemented service for refreshing the 'Access Token'. This is required when the 'Custom ESB Service' option is selected as the 'Refresh URL Request'. This service must strictly conform to the specification:

```
- wm.cloudstreams.service.common.lookup.
specs:oauthTokenRefreshServiceSpec
```

Authorization Header Prefix: The prefix to be used with the 'Access Token' in the Authorization header. Options are 'Bearer' and 'OAuth'. Default is 'Bearer'.

protocol

- **elementCharSet:** The encoding to use for the HTTP request line, headers, etc.
- **contentCharSet:** The encoding to use for the request message.
- **httpVersion:** The HTTP version (HTTP/0.9, HTTP/1.0 or HTTP/1.1). The default value for the connection factory is HTTP/1.1.
- **userAgent:** The value to the connection configuration will send for the User-Agent request header.

If the group type is...	The available fields are...
connection	<ul style="list-style-type: none"> <li data-bbox="565 321 1406 394">■ useExpectContinue: If true, use the Expect/Continue HTTP/1.1 handshake and send the Expect request header. <li data-bbox="565 411 1406 516">■ waitForContinue: The number of milliseconds that the connection factory's client connection should wait for a "100 Continue" response from the server. <li data-bbox="565 533 1406 638">■ useStrictTransferEncoding: If true, the connection factory connection raise an exception if the "Transfer-Encoding" header is invalid. <li data-bbox="565 655 1406 728">■ useChunking: If true, use HTTP/1.1 chunking, using a chunk size that matches the socket buffer size. <li data-bbox="565 745 1406 781">■ flowServerRedirects: If true, follow server redirects. <li data-bbox="565 798 1406 871">■ serverRedirectMax: Maximum number of times to follow a server redirect. <li data-bbox="565 905 1406 1010">■ providerURL: The native provider endpoint target for the connection configuration. The default configuration field provided with the connection factory is <code>cn.providerURL</code>. <li data-bbox="565 1026 1406 1100">■ minPoolConnections: The minimum number of socket connections to reserve for a connection configuration alias. <li data-bbox="565 1117 1406 1190">■ maxPoolConnections: The maximum number of socket connections to reserve for a connection configuration alias. <li data-bbox="565 1207 1406 1281">■ connectTimeout: The number of milliseconds a connection attempt will wait before giving up. (0 will wait indefinitely.) <li data-bbox="565 1297 1406 1402">■ readTimeout: The number of milliseconds in which the the client must read a response message from the server. (0 will wait indefinitely.) <li data-bbox="565 1419 1406 1524">■ useStaleChecking: If true, the connection factory performs additional processing to test the socket to see if it is still functional each time it is used. <li data-bbox="565 1541 1406 1614">■ retryCount: How many times should the connection factory attempt to execute a failed invocation. <li data-bbox="565 1631 1406 1736">■ retryOnResponseFailure: If true, the retry mechanism will be used for failed responses even if the request was sent successfully. <li data-bbox="565 1753 1406 1827">■ useTcpNoDelay: If true, do not use Nagles algorithm as a socket optimization technique. <li data-bbox="565 1843 1406 1862">■ linger: Determines how quickly a socket should close.

If the group type is...	The available fields are...
	<ul style="list-style-type: none"> <li data-bbox="565 321 1414 394">■ bufferSize: The size of the read and write socket buffers, in bytes. <li data-bbox="565 411 1414 485">■ reuseAddr: If true, the socket will be reused even if it is in TIME_WAIT due to a previous socket closure. <li data-bbox="565 501 1414 537">■ sessionToken: Session token for a stateful session. <li data-bbox="565 554 1414 627">■ webproxyAlias: The alias to a web proxy server configuration in Integration Server. <li data-bbox="565 644 1414 718">■ truststoreAlias: Alias for the Integration Server trust store configuration. <li data-bbox="565 735 1414 1010">■ hostnameVerifier: Fully qualified class name that implements the Apache HC <code>org.apache.http.conn.ssl.X509HostnameVerifier</code> interface; helps guard against "man-in-the-middle" attacks. Also set the IS property <code>watt.security.cert.wmChainVerifier.trustByDefault</code> to false and ensure the outbound connector's connection configuration property <code>Hostname verifier</code> is set to its default value, <code>org.apache.http.conn.ssl.StrictHostnameVerifier</code>.
requestHeaders	<ul style="list-style-type: none"> <li data-bbox="565 1047 1414 1184">■ names: An array of request header names to include for this connection configuration. The value should be a comma-delimited list of header names; for example <code>Content-Type, SOAPAction</code>. <li data-bbox="565 1201 1414 1339">■ values: An array of request header values to include for this connection configuration. The value should be comma-delimited list of values in the same order as the header names; for example, <code>text/xml, login</code>.
credentials	<ul style="list-style-type: none"> <li data-bbox="565 1377 1414 1451">■ username: The username credentials for the current connection configuration. <li data-bbox="565 1467 1414 1541">■ password: The password credentials for the current connection configuration <li data-bbox="565 1558 1414 1665">■ preemptiveAuth: If true, basic auth credentials will be included when a request is sent. (It will not wait for a 401 response challenge.) <li data-bbox="565 1682 1414 1755">■ authorizationType: The string identifying the authentication protocol scheme to use for the connection configuration. <li data-bbox="565 1772 1414 1831">■ domainName: The domain/security realm for the current connection configuration.

If the group type is...	The available fields are...
aws_v2	<ul style="list-style-type: none"> <li data-bbox="565 321 1317 394">■ keystoreAlias: Alias for the Integration Server key store configuration. <li data-bbox="565 411 1386 447">■ clientKeyAlias: Alias to reference a key inside a key store file. <li data-bbox="565 485 1417 558">■ Signing Algorithm: Explicitly specify the signing algorithm (e.g. HMAC-SHA1 Signatures) used to sign the message. <li data-bbox="565 575 1360 711">■ Access Key: This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same AWS Access Key. <li data-bbox="565 728 1398 865">■ Secret Key: This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as asterisk or dots. <li data-bbox="565 882 1011 919">■ Region: An area-specific value.
aws_s3	<ul style="list-style-type: none"> <li data-bbox="565 961 1360 1098">■ Access Key: This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same AWS Access Key. <li data-bbox="565 1115 1398 1251">■ Secret Key: This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as asterisk or dots. <li data-bbox="565 1268 1011 1312">■ Region: An area-specific value.
aws_v4	<ul style="list-style-type: none"> <li data-bbox="565 1354 1360 1491">■ Access Key: This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same AWS Access Key. <li data-bbox="565 1507 1398 1644">■ Secret Key: This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as asterisk or dots. <li data-bbox="565 1661 1011 1705">■ Region: An area-specific value.
custom	User-defined fields of a custom group.

You can create additional connection groups by right-clicking the **Connections** node and selecting **Add Group**.

6. If your provider requires a Login Sequence, configure one as follows:
- Right-click the **Login Sequence** node, select **Add Resource**, select the Type **Login**, and click **Next**. The Login Sequence/Logout Sequence will be enabled for SOAP based connections only if you have configured any Login/Logout Operation under Services. The Login Sequence/Logout Sequence will be enabled by default for REST based connections.
 - Configure a Login resource by completing the fields as follows.

Field	Description
Name	Specify a name for the Login resource.
Document Reference	Click Browse and select a Document Reference.
Content Type	Specify the Content-Type of the Document Reference.

- Click **Next**.
- Create a Request for the Login resource by completing the fields as follows.

Field	Description
Name	Specify a name for the Request.
Document Reference	Click Browse and select a Document Reference, for example myConnector_v1.customDoctypes:docTypeRef_LoginInput .
Content Type	Specify the Content-Type of the Document Reference, for example, application/xml .

- Click **Next**.
- Create one or more Responses for the Login resource by completing the fields as follows.

Field	Description
Name	Specify a name for the Response.
Document Reference	Click Browse and select a Document Reference, for example myConnector_v1.customDoctypes:docTypeRef_LoginOutput .

Field	Description
Code	Specify a code for the Response, such as 201 or 400. Response codes should be comma separated.
Content Type	Specify the Content-Type of the Document Reference for example, application/xml .

- g. Click **Next**.
- h. Create one or more parameters for the Login resource by completing the fields as follows.

Field	Description
Name	Specify a name for the parameter.
Active	Specifies whether the parameter is active. If the Active option is disabled, CloudStreams will not include the parameter in the input/output signature at runtime.
Parameter Type	Select the parameter's type which determines how the parameter should be used, e.g., <code>URI_CONTEXT</code> , <code>QUERYSTRING_PARAM</code> or <code>CFG_PARAM</code> . For a list of the supported parameter types, see "Parameter Types" on page 272 .
Default Value	Specify a default value for the parameter. In pipeline, and input/output validation, the data validation applies <i>constraints</i> to its variables. Designer displays small symbols next to a variable icon to indicate validation constraints. For more information about icons for constrained variables, see "Viewing the Constraints Applied to Variables" on page 279 .
Description	Optional description of the parameter.

- i. Click **Finish**.
- j. Define the input/output mappings for the Login Sequence by right-clicking the Login Sequence name, selecting **Add Mapping**, and completing the fields as follows.

Field	Description
Parameter Type	Select IN or OUT .
Mapping fields	<p>For example, for input fields, you might enter the display names <code>Username</code> and <code>Password</code>, and the values <code>cr.username</code> (the connection user name key field) and <code>cr.password</code>, respectively.</p> <p>And for output fields, you might enter the following display names and values:</p> <ul style="list-style-type: none"> ■ <code>Server URL</code> (with value <code>cx.serverUrl</code>). ■ <code>Provider URL</code> (with value <code>cn.providerUrl</code>). ■ <code>Session Token</code> (with value <code>cn.sessionToken</code>). ■ <code>Fault String</code> (with value <code>cx.faultString</code>).

7. Configure a **Logout Sequence** in a similar manner.
8. Next, configure your cloud connector services, as described below.

Configuring Cloud Connector Services (REST)

In this task, you will configure the cloud connector service you just created in the New Cloud Connector wizard. You need to create at least one cloud connector service for each REST resource on which the connector can operate.

To configure a cloud connector service (REST)

1. Start **Software AG Designer** and open the **CloudStreams Development** perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the **CloudStreams Connectors** view, expand your **CloudStreams Provider** project and click the cloud connector you just created (as described in "[Creating Custom SOAP or REST Cloud Connectors](#)" on page 234).

The **Overview** page is displayed, showing the general information you defined in the wizard.

3. Click the **Services** link in the **Connector Content** section of the page (or click the **Services** tab at the bottom of the page).
 - If you already defined a cloud connector service on page 2 of the New Cloud Connector wizard, the **Services Configuration** page will show your new service and the default node **Resources**.
 - If you did *not* define a cloud connector service on the New Cloud Connector wizard, define it now by right-clicking the **Services** node, selecting **Add Service** and specifying the SaaS provider's XSD that CloudStreams will use to generate

the document types. For a description of the **Document Type Package** field, see ["Creating Custom SOAP or REST Cloud Connectors" on page 234](#).

4. Define any necessary global mappings to be used by all the REST resources in the service. To do this, right-click the service, select **Add Global Mapping** and complete the fields as follows.

Field	Description
Mapping Name	Assign a name for the global mapping.
Parameter Type	Select the parameter's type which determines how the parameter should be used, e.g., URI_CONTEXT, QUERYSTRING_PARAM or CFG_PARAM. For a list of the supported parameter types, see "Parameter Types" on page 272 .
Required	Specify whether this is a required mapping.
Fixed	A Fixed mapping means its values cannot be changed at run time.
Mapping Key Type	Select the data source type: <ul style="list-style-type: none"> ■ Cookie: HTTP cookies. For example, the Salesforce.com connector inserts the session token from a valid login in the SOAP header of the business operations to be executed. Other providers do the same thing; however, the token is managed via cookies instead of the SOAP header. ■ Header: Request or response HTTP transport headers for the support connection factory implementation. ■ IData: IData is a variable name that is associated with the connection instance when the service is invoked. It is a subset of those configuration fields from the selected groups. Not all fields are eligible for use as a mapping step since their values may not be changed or have any useful meaning in this context. ■ Literal: A constant value. ■ Parameter: Primarily used for REST handlers. ■ Service: The Service type applies to the source, not to the target. Select Service if you want the source to call a given service to perform certain tasks and to map the output of the service to a target. The service must adhere to <code>wm.cloudstreams.service.common.lookup.specs:mapServiceSpec</code>.

Field	Description
	<p>If an error occurs related to service validation or execution, CloudStreams throws a Mapping Exception.</p> <ul style="list-style-type: none"> ■ XPath: Enables you to define an XPath expression.
Mapping Key	Enter the source mapping key. For example, if your source mapping type is IData, you might enter the source mapping key <code>cn.sessionToken</code> .
Mapping Value Type	<p>Select the target mapping type:</p> <ul style="list-style-type: none"> ■ Cookie ■ Header ■ IData ■ Literal ■ Parameter ■ Service ■ XPath
Mapping Value Key	Enter the target mapping key. For example, if your target mapping type is Header, you might enter the target mapping key <code>X-SFDC-Session</code> .
Document Name	The document type, based on the XSD.

5. Add a REST resource by right-clicking the **Resources** node, selecting **Add REST Resource** and completing the following fields in the wizard.

Field	Description
Name	Assign a name for the resource.
Method	Select an available HTTP method (GET, PUT, POST, DELETE).
Path	Specify the path to the resource.
Description	Optional description for the resource.

6. Click **Next**.
7. Add a Request to a REST resource by completing the fields as follows.

Field	Description
Request Name	Specify the request name, for example <code>tns:jobInfo</code> .
Incoming Parsing Type	The Content-Type of the request, for example, application/octet+idateref .
Outgoing Serialization Type	The Content-Type for the response, for example, application/xml .
Document Reference Type	Click Browse and select a document reference. For example, <code>myConnector_v1.doctypes:docTypeRef_tns_JobInfo</code> .
Exclude document root	Select this option to exclude the root element of the JavaScript Object Notation (JSON) document type, for REST resource's request and response.

- Click **Next**.
- Add one or more Responses to a REST resource by completing the fields as follows.

Field	Description
Response Name	Specify the request name, for example <code>tns:jobInfo</code> .
Response Codes	Specify one or more codes for the response, such as 201 or 400. Response codes should be comma separated.
Incoming Parsing Type	For example, application/xml .
Outgoing Serialization Type	For example, application/octet+idateref .
Document Reference Type	Click Browse and select a document reference. For example, <code>myConnector_v1.doctypes:docTypeRef_tns_JobInfo</code> or <code>myConnector_v1.doctypes:docTypeRef_tns_Error</code> .
Exclude document root	Select this option to exclude the root element of the JavaScript Object Notation (JSON) document type, for REST resource's request and response.

- Click **Next**.

11. Select the predefined input parameters for the REST resource as desired. Note that:
 - Mandatory parameters are selected by default and you cannot un-select them.
 - You can provide a default value for any parameter by typing a value in the **Default Value** field. If the variable is null in the input pipeline, this default value will be used at run time. The value given at run time always takes precedence over the default value. However, if the existing default value is of type "fixed default", the overwrite will fail.
 - Parameters have various data validation constraints that apply to them. These constraints are indicated by an icon next to each parameter. For information about these constraints, see "[Viewing the Constraints Applied to Variables](#)" on page 279.
12. Optionally create additional input parameters for the REST resource by clicking the **Add** button and completing the following fields.

Field	Description
Name	Specify a parameter name.
Document Reference	For QUERYSTRING_PARAM type parameters only. If the parameter is a complex parameter involving indices where an IS document type is required, specify the fully qualified name of the IS document type.
Formatter Service	For QUERYSTRING_PARAM type parameters only. If you have used a parameter formatter to format input data into a custom format, specify your custom formatter service here, and then provide the input in the form of "key:value" pairs for the arguments specified in the service. For details about parameter formatters, see " Parameter Formatters for REST Connector Services " on page 274.
Parameter Type	Select the parameter's type, which determines how the parameter should be used, e.g., URI_CONTEXT, QUERYSTRING_PARAM or CFG_PARAM. For more information about the supported parameter types, see " Parameter Types " on page 272.
Default Value	Specify a default value for the parameter. If the variable is null in the input pipeline, this default value will be used at run time. The value given at run time always takes precedence over the default value. However, if the existing default value is of type "fixed default", the overwrite will fail.

Field	Description
	<p>Note: Parameters have various data validation constraints that apply to them. These constraints are indicated by an icon next to each parameter. For information about these constraints, see "Viewing the Constraints Applied to Variables" on page 279.</p>
Description	Optional description of the parameter.
Required	Specifies whether the parameter is required.

13. Click **Finish**.

14. Define the local mappings for the REST resource by right-clicking the resource name, selecting **Add Mapping** and completing the fields as follows.

Field	Description
Name	Assign a name for the local mapping. This mapping will be used only by this particular resource.
Parameter Type	Select the parameter's type which determines how the parameter should be used, e.g., <code>URI_CONTEXT</code> , <code>QUERYSTRING_PARAM</code> or <code>CFG_PARAM</code> . For a list of the supported parameter types, see " Parameter Types " on page 272.
Required	Specify whether this is a required mapping.
Fixed	A Fixed mapping means its values cannot be changed at run time.
Mapping Key Type	<p>Select the data source type:</p> <ul style="list-style-type: none"> ■ Cookie: HTTP cookies. For example, the Salesforce.com connector inserts the session token from a valid login in the SOAP header of the business operations to be executed. Other providers do the same thing; however, the token is managed via cookies instead of the SOAP header. ■ Header: Request or response HTTP transport headers for the support connection factory implementation. ■ IData: IData is a variable name that is associated with the connection instance when the service is invoked. It is a subset of those configuration fields from the selected groups. Not all fields are eligible for use as a mapping step since their

Field	Description
	<p>values may not be changed or have any useful meaning in this context.</p> <ul style="list-style-type: none"> ■ Literal: A constant value. ■ Parameter: Primarily used for REST handlers. ■ Service: The Service type applies to the source, not to the target. Select Service if you want the source to call a given service to perform certain tasks and to map the output of the service to a target. The service must adhere to <code>wm.cloudstreams.service.common.lookup.specs:mapServiceSpec</code>. If an error occurs related to service validation or execution, CloudStreams throws a Mapping Exception. ■ XPath: Enables you to define an XPath expression.
Mapping Key	Enter the source mapping key. For example, if your source mapping type is Literal, you might enter the key <code>application/xml</code> .
Mapping Value Type	<p>Select the target mapping type:</p> <ul style="list-style-type: none"> ■ Cookie ■ Header ■ IData ■ Literal ■ Parameter ■ Service ■ XPath
Mapping Value Key	Enter the target mapping key. For example, if your target mapping type is Header, you might enter the key <code>Content-Type</code> .
Document Name	The document type, based on the XSD.

15. View the **Manifest** node, which shows read-only fields containing attributes whose values were captured from your cloud connector definition. The version of the CloudStreams CloudStreams Development Plug-in (Created-By) is also displayed. For imported plug-ins created in a Designer release prior to 9.7, this information is not shown.

Understanding REST Parameters

CloudStreams REST connector services allow you to set parameters which become part of the outgoing request.

REST services rely on HTTP methods (GET, POST, PUT, DELETE) to make requests to a SaaS provider. Thus the parameters are closely tied to these HTTP methods, as they are sent as part of these HTTP method requests.

The parameters are typically part of the HTTP URI which may be of the following format:

```
[scheme:][//[authority][path][?query]
```

Parameter Types

CloudStreams supports the following parameter types:

Type	URI Scope
"URI_CONTEXT" on page 272	path
"QUERYSTRING_PARAM" on page 273	query
"CFG_PARAM" on page 273	authority, path

URI_CONTEXT

URI_CONTEXT parameters are passed as the path component of a REST resource URI, and the parameter names correspond to the URI path variable names specified in the {} annotation. For example, a sample request might look like this:

```
services/async/25.0/job/{jobId}
```

In the above sample request, the URI path variable name `jobId` is specified as a parameter to the job resource. The annotation {} is set to the variable name `jobId`. At service run-time, the variable is substituted with its runtime value to form the dynamic path.

A sample resource definition with *URI_CONTEXT* parameter:

```
<resource name="Job" method="POST" path="services/async/25.0/job/{jobId}">
  <parameters>
    <parameter name="jobId" isRequired="true" style="URI_CONTEXT"/>
  </parameters>
</resource>
```

QUERYSTRING_PARAM

QUERYSTRING_PARAM parameters are passed as the query component of a REST resource invocation request.

The following example demonstrates a typical HTTP GET request with parameters that form a query string of the resource URI.

```
GET /status?key1=value1&key2=value2 HTTP/1.1
Host: www.softwareag.com
Content-Length: 0
```

Notice that the parameters are added to the path after a "?", and specified as ampersand (&) separated list of key-value pairs, with the corresponding Content-Length set accordingly. The key & values passed as parameters are URL encoded by CloudStreams.

A sample resource definition with *QUERYSTRING_PARAM* parameter:

```
<resource name="GetStatus" method="GET" path="/status">
  <parameters>
    <parameter name="key1" isRequired="true" style="QUERYSTRING_PARAM"/>
    <parameter name="key2" isRequired="false" style="QUERYSTRING_PARAM"/>
  </parameters>
</resource>
```

CFG_PARAM

CFG_PARAM parameters signify an internal contract between the connector tier and the connector-specific authentication scheme along with the virtual runtime layer.

The allowed set of parameter name, which can be specified as *CFG_PARAM* depends on the authentication scheme used for the connector definition.

Example

`aws.bucketName` is a parameter, which is used by the Amazon Authentication Scheme version 3, for specifying the dynamic host based on the parameter value.

While executing a resource with such parameter, the service endpoint is prefixed with the appropriate bucket name.

```
<resource name="GetBucket" method="GET" path="/">
  <parameters>
    <parameter name="aws.bucketName" isRequired="true" style="CFG_PARAM"/>
  </parameters>
</resource>
```

Parameter Data Types

CloudStreams currently supports the data types *String* and *Record*.

String

String data type represents a simple parameter which is an individual string key-value parameter. This is the most commonly used data type and the default type, if not specified explicitly.

Record

Record data type handles complex parameters involving a collection of related key-value pairs. These parameters can have indices's for one or more parameter entry. Such parameters can ideally be represented with an IS document type, and configured as a "Record" data type.

Example

Consider a complex parameter structure as follows:

```
+ Filter[]
  - Name
  + Value[]
```

This needs to be represented as a query string like:

```
Filter.1.Name=instance-
type&Filter.1.Value.1=m1.small&Filter.1.Value.2=t1.micro
```

Such a parameter can be defined by creating an IS document type, and referring it as a "Record" data type parameter within the resource definition.

CloudStreams provides the capability to represent and transform such complex parameters into the desired format using its predefined formatter implementation. The resource's parameter definition can be enhanced to refer to a formatter, as follows:

```
<parameter name="Filter" dataType="Record"
documentRef="documents.paramType:FilterList"
style="QUERYSTRING_PARAM">
  <formatter service="wm.cloudstreams.service.util.formatters:paramFormatter"
type="paramFormatter"/>
</parameter>
```

At runtime, the resource PATH would look like:

```
GET /?Filter.1.Name=instance-type&Filter.1.Value.1=m1.small&Filter.1.Value.2=t1.micro
```

For more details on formatters and customizing the formatter behavior, see ["Parameter Formatters for REST Connector Services"](#) on page 274.

Parameter Formatters for REST Connector Services

CloudStreams provides a parameter formatter which enables you to format input data into a custom format as desired.

Alternatively, in cases where the default parameter formatter is not sufficient to achieve the desired parameter behavior, you can write a custom implementation for the parameter formatter.

The input will typically be data present in a complex structure (e.g., an IS document type).

The output will be data organized in a desired output format (e.g., a URL query string). The format in which the output data is generated depends on the implementation of the formatter.

Typically, you will use the `QUERYSTRING_PARAM` parameter type to format the input data into a desired output format (e.g., a URL query string).

In order to form the query string, you can write a custom formatter service by implementing the bundled IS specification `wm.cloudstreams.service.common.lookup.specs:formatterSpec`.

At run time CloudStreams will give control to the formatter service to format the data appropriately and to return the formatted string data. Then, CloudStreams will encode the data during a post-processing step, and the final query string will be formed.

For details, see:

- ["Using the Default Parameter Formatter" on page 275.](#)
- ["Implementing a Custom Parameter Formatter" on page 276.](#)

Using the Default Parameter Formatter

For convenience, CloudStreams provides a default formatter implementation which handles the most common formatter usage:

`wm.cloudstreams.service.util.formatters:paramFormatter`

This formatter allows you to specify a complex IS document type, and to implement the desired output. The input will typically be data in a complex structure (e.g., an IS document type). The output will be a data organized in a desired output format (e.g., a URL query string). The format in which the output data is generated depends on the implementation of the formatter. Typically, the formatter is used in conjunction with the `QUERYSTRING_PARAM` parameter type.

Example of using the default parameter formatter

Consider a SaaS back-end expecting the following parameter:

```
BlockDeviceMapping.n.DeviceName
BlockDeviceMapping.n.VirtualName
BlockDeviceMapping.n.Ebs.NoDevice
BlockDeviceMapping.n.Ebs.VolumeSize
```

where `n` represents a number.

1. Create IS document types for the above parameter definition.

The IS document structure would look like this:

```
BlockDeviceMapping[]
+ DeviceName
+ VirtualName
+ Ebs
  + NoDevice
  + VolumeSize
```

2. Set the `dataType` of the parameter to `Record`.
3. Set the document reference (`documentRef`) to the newly created IS document type in step 1.

4. Specify a formatter element within the parameter element, along with the service name (*service* attribute) and type (*type* attribute).
5. In case the implementation contains supported arguments, specify any customization by adding the appropriate argument name and value.

The parameter definition for a resource using the above, within the Connector Descriptor, would look like this:

```
<parameter name="BlockDeviceMapping" dataType="Record"
documentRef="document.paramType:BlockDeviceList" style="QUERYSTRING_PARAM">
  <formatter service = "wm.cloudstreams.service.util.formatters:paramFormatter"
type = "paramFormatter">
    <arg name="startIndex" value="1" />
    <arg name="format" value="." />
  </formatter>
</parameter>
```

Note: The default value for the type attribute, within the scope of a parameter definition is "paramFormatter". This type refers to the bundled IS Specification:

```
wm.cloudstreams.service.common.lookup.specs:formatterSpec
```

The generated representation of the above parameter would look like this:

```
BlockDeviceMapping.1.DeviceName=/dev/sdj&;BlockDeviceMapping
.1.Ebs.NoDevice=true&;BlockDeviceMapping
.2.DeviceName=/dev/sdh&;BlockDeviceMapping
.2.Ebs.VolumeSize=300&;BlockDeviceMapping
.3.DeviceName=/dev/sdc&;BlockDeviceMapping
.3.VirtualName=ephemeral1
```

The default formatter can be configured with the following supported arguments:

Argument	Default Value	Description
startIndex	0	Determines the index values of the keys e.g. should the n present in BlockDeviceMapping.n.DeviceName start with 0 or 1 or some number of your choice.
format	.(period)	Specifies how the document entries be separated. For example, BlockDeviceMapping.3.DeviceName is separated by a period (.).

Implementing a Custom Parameter Formatter

In cases where the default parameter formatter is not sufficient to achieve the desired parameter behavior, you can write a custom implementation for the parameter formatter.

Implement a custom formatter as follows:

1. Write an IS service implementation, adhering to the bundled IS specification `wm.cloudstreams.service.common.lookup.specs:formatterSpec`.
2. The fully-qualified service namespace should be referred with the `formatter` service attribute, under the parameter definition. Ensure that the `formatter` type attribute is set to `"paramFormatter"`.
3. Optionally, any service arguments can be specified within the `formatter` element definition.

Example

Consider a custom service implementation:

```
my.custom.formatter:sample
```

The parameter definition for a resource using the above, within the Connector Descriptor, would look like this:

```
<parameter name="SampleParam" dataType="Record"
documentRef="my.custom.document:sample" style="QUERYSTRING_PARAM">
  <formatter service = "my.custom.formatter:sample" type = "paramFormatter">
    <arg name="separator" value=";" />
  </formatter>
</parameter>
```

Handling JSON representations of REST resources

If you are creating resources which have JavaScript Object Notation (JSON) data representation and you want to have your outgoing requests in JSON data format and response received in JSON format, you must set the `excludeRoot` properties correctly. Ensure that you create your IS document representations in accordance with the structure of the data.

JSON data format typically has data organized in `key:value` pairs enclosed within `{ }` brackets. A JSON data snippet may not have a root element, so if you want to send requests using an IS document (whose name represents a data root), you need to exclude that root element because you may want only the child key and value elements to be sent. Similarly, in response, you may want to exclude or consider the root element based on the representation of the resource from the SaaS provider.

To handle such cases, consider the `excludeRoot` property exposed through the connector development plug-in. This property is applicable only when dealing with resources having JSON representation, not XML.

For requests, if the property is selected, outgoing requests will have `key:value` pairs without the parent root, for example, `data - {"first": "John", "last": "Doe"}`. For responses, it means that the root is excluded in the response. Further, it may have two or more cases.

The following information describes how you should create the corresponding IS documents and how the property should be set:

1. If the JSON response has just `key:value` pairs and no top level parent root, you can have a document type with custom root, having the expected `key:value` pairs. In

the response resource, `excludeRoot` should be set to `true` or the property should be selected. For example, `{"first": "John", "last": "Doe", "age": 20, "sex": "M", "salary": 50000, "registered": true }`.

2. If the JSON response has key:value pairs with a top level parent root present, you can set the `exclude root` to `false` in response as your document matches the response. For example, `{ "error": { "errors": [{ "domain": "local", "reason": "credError", "message": "Invalid Credentials" }], "code": "401", "message": "Invalid Credentials" }}`

This example already has a root - `error`, so your IS document type can be mapped if its root name is `error` too. Within it, the child elements appear as in the response data.

It is important to create appropriate IS documents and set the correct property value for the responses to be mapped correctly.

Support for multiple root JSON data format with anonymous collection

The multiple root feature enables CloudStreams to support JavaScript Object Notation (JSON) APIs, which return anonymous collection, that is, collection without a key, of JSON objects provided by a JSON provider. CloudStreams can now support anonymous object collection or array in the response design time of a cloud service.

You can use CloudStreams to access those APIs which return anonymous collection of JSON objects for a JSON backend. The number of APIs supported now by CloudStreams is more. Most of the social connectors support anonymous collection in most of the APIs.

This feature also helps you to get error messages for scenarios where the `exclude root` is marked as `true`. You can experience a broader connectivity to all sets of JSON backends, especially social connectors.

The multiple root feature allows you to consume all types of APIs and does not restrict you to a subset of APIs. You can create cloud services that communicate with the JSON backends, which expect an anonymous collection from the backend. With added support in the Software AG Designer, you can now create document types of the JSON objects whose collection is expected from the JSON backend and can also map it as a logical collection in the design time of the cloud service. You need not worry about root less array of JSON objects because at runtime, CloudStreams automatically handles the contents of each object separately.

Note: This feature is supported on version 9.8 and later versions of CloudStreams Server and Software AG Designer.

To enable the multiple root feature, set the property `wst.rest.json.operation.multiroot.wrapPayload` to `true` in the CloudStreams `wst-config.properties` file available at: `IntegrationServer_directory\instances\instance_name\packages\WmCloudStreams\config\resources`. The default value is `true`.

If you do not want to enable the multiple root feature, for example, in cases where you will not consume JSON APIs that return anonymous collection, set this property `wst.rest.json.operation.multiroot.wrapPayload` to *false*.

Viewing the Constraints Applied to Variables

Designer displays small symbols next to a variable icon to indicate the constraints applied to the variable as follows.

Variable	Constraint Status	Variable Properties
 String	Required field.	The <code>Required</code> property is set to <code>True</code> .
 String	Optional field.	The <code>Required</code> property is set to <code>False</code> .
 String	Required field with content type constraint.	The <code>Content type</code> property specifies an IS schema or XML schema.
 String	Optional field with content type constraint.	The <code>Required</code> property is set to <code>False</code> , and the <code>Content type</code> property specifies an IS schema or XML schema.
 String	Required field with default value.	The <code>Fixed</code> property is set to <code>False</code> , and the <code>defaultValue</code> property specifies a default value. The variable has a default value, but you can override this default value with any other valid values while executing the service or mapping the variables.
 String	Required field with fixed value.	The <code>Fixed</code> property is set to <code>True</code> , and the <code>defaultValue</code> property specifies a null value. The variable has a null value assigned to it by default and you cannot override this value. You cannot map this variable to another variable or assign any input values to this variable during service execution.
 String	Required field with fixed default value.	The <code>Fixed</code> property is set to <code>True</code> , and the <code>defaultValue</code> property specifies a default value.

Variable	Constraint Status	Variable Properties
		The variable has a default value and you cannot override this value. You cannot map this variable to another variable or assign any input values to this variable during service execution.

Deploying Custom Cloud Connectors

Note: If you have not specified the provider's WSDL or XSD, then you will not be allowed to deploy the connector. You should have specified the provider's WSDL or XSD either in the New Cloud Connector wizard, or when you created a cloud connector service in the Services Configuration page (see ["Configuring Cloud Connector Services \(SOAP\)" on page 249](#)).

To deploy a custom cloud connector

1. Start Software AG Designer and open the CloudStreams Development perspective by clicking **Window > Open Perspective > CloudStreams Development**.
2. In the CloudStreams Connectors view, expand your CloudStreams Provider project, right-click the cloud connector and select **Deploy**.

If the deployment is successful, a message is displayed. These messages are not persisted, so the next time Software AG Designer is started, the prior messages will not be shown.

3. Next, you must enable the deployed cloud connector as described in ["Managing Custom Cloud Connectors" on page 280](#).

Managing Custom Cloud Connectors

After you deploy a custom cloud connector, you must enable it as described below. In addition, you can:

- View its configured connection pool instance(s).
- View other details about the connector.
- Configure the properties of the connector as desired.
- Delete the connector.

Note: When a new connector version is released, in the provider package, the new connector version is enabled, and all older connector versions are disabled. If you still want to work with an older connector version, you can manually enable the older connector version, if available, in the package. To manually

enable a cloud connector, from Integration Server Administrator, go to **Solutions > CloudStreams > Providers**. Select a provider and on the **Connector List** screen, for the connector you want to enable, click on **No** in the **Enabled** column.

To manage a custom cloud connector

1. In the Integration Server Administrator, click **Solutions > CloudStreams > Providers > your_provider_name**.

The list of custom cloud connector(s) that have been defined for that provider will appear in the Connector List.

2. You can do the following in the Connector List:

Column	Description
Connector Name	Click the connector name to display the connector's configured connection pool instance(s). To create a new connection pool instance, click Configure New Connection and complete the fields on the Configure Connection page.
View	Click the icon to display details about the connector.
Enabled	Click No or Yes to enable/disable the connector.
Delete	Click the icon to delete the connector. You must disable the connector before you can delete it.
Configure	Click the icon to display the Connector Configuration page, which displays the connector properties you can configure. You must disable the connector before you can edit the properties.

Managing the Custom Cloud Connector Packages

When you deploy a custom cloud connector, *two* packages are deployed: One package for the cloud connector you created and one for the *provider*, which CloudStreams automatically creates. When the connector is distributed, only the provider package is shipped. You can manage both packages as described below.

To manage the custom cloud connector packages

1. In the Integration Server Administrator, click **Packages > Management**.
2. Here you can enable/disable the packages, reload them, archive them, and delete them.

A Admin Folder

■ Summary of Elements in this Folder	284
■ pub.cloudstreams.admin.connection:disableConnection	284
■ pub.cloudstreams.admin.connection:enableConnection	285
■ pub.cloudstreams.admin.connection:getConnectionStatistics	285
■ pub.cloudstreams.admin.connection:queryConnectionState	286
■ pub.cloudstreams.admin.service:update	286

Summary of Elements in this Folder

The following elements are available in this folder:

Element	Description
pub.cloudstreams.admin.connection:disableConnection	Disables a connection node.
pub.cloudstreams.admin.connection:enableConnection	Enables an existing connection node.
pub.cloudstreams.admin.connection:getConnectionStatistics	Returns current usage statistics for a connection node.
pub.cloudstreams.admin.connection:queryConnectionState	Returns the current connection state (enabled/disabled) and error status for a connection node.
pub.cloudstreams.admin.service:update	Updates a CloudStreams connector service node.

pub.cloudstreams.admin.connection:disableConnection

Disables a connection node.

Input Parameters

connectionAlias **String.** Name of the connection node you want to disable.

Output Parameters

status **String.** Indicates whether the connection is disabled successfully. Values are:

- `true` - The connection is disabled.

- `false` - Disabling the connection failed.

statusMessage **String.** The status message of the error.

pub.cloudstreams.admin.connection:enableConnection

Enables an existing connection node.

Input Parameters

connectionAlias **String.** Name of the connection node you want to enable.

Output Parameters

status **String.** Indicates whether the connection is enabled successfully.
Values are:

- `true` - The connection is enabled.
- `false` - Enabling the connection failed.

statusMessage **String.** The status message of the error.

pub.cloudstreams.admin.connection:getConnectionStatistics

Returns current usage statistics for a connection node.

Input Parameters

connectionAlias **String.** Name of the connection node for which you want usage statistics returned.

Output Parameters

connectionStatistics **Document List.** Information for each connection node. Keys are:

- *totalConnections* . **String.** Current number of connection instances.
- *busyConnections* . **String.** Number of connections currently in use by services, notifications, and listeners.

- *freeConnections* . **String**. Total number of connections created and available for use.
- *totalHits* . **String**. Number of times this connection node successfully provided connections since the last reset.
- *totalMisses* . **String**. Number of times this connection node unsuccessfully provided connections since the last reset (when the request timed out).

pub.cloudstreams.admin.connection:queryConnectionState

Returns the current connection state (enabled/disabled) and error status for a connection node.

Input Parameters

connectionAlias **String**. Name of the connection node for which you want the connection state and error status returned.

Output parameters

response **Document List**. Information about the connection state and error status. Keys are:

- *connectionState* . **String**. Current connection state (enabled/disabled).
- *hasError* . **Boolean**. Flag indicating if any error was detected on the connection. Values are:
 - `true` - An error was detected.
 - `false` - No error was detected.
- *lastErrorTime* . **String**. Indicates the last time an error occurred.

pub.cloudstreams.admin.service:update

This service allows you to update a CloudStreams connector service node for various attributes such as the *Connection Alias* and the *Virtual Service Name* . If the connector service node was created with an earlier version of CloudStreams, you must first migrate the service node before using the update service. For information on how to migrate a connector service node, refer the document *Upgrading webMethods and Intelligent Business Operations Products*.

Input Parameters

<i>cloudServiceName</i>	String. Required. Name of the CloudStreams connector service node which needs to be updated.
<i>connectionAlias</i>	String. Optional. Name of the new connection alias. The CloudStreams connector service node will get updated to use this new connections alias. The new connection alias must be from the same CloudStreams connector.
<i>virtualServiceName</i>	String. Optional. Name of the new virtual service name. The CloudStreams connector service node will get updated to use this new virtual service. For more information on virtual services, see "Virtual Services" on page 55.

Output parameters

<i>success</i>	Boolean String. Required. <ul style="list-style-type: none">■ <code>true</code> – If the CloudStreams connector service node was updated successfully.■ <code>false</code> – If the CloudStreams connector service node update has failed.
<i>errorMessage</i>	String. Optional. Information on the cause of the error, if the update has failed.

B Advanced Settings

■ Introduction	291
■ pg.backupFailedProxies	291
■ pg.CollectionPool.	291
■ pg.CollectionWorkQueue.	292
■ pg.debug.	292
■ pg.delayedRefresher.	292
■ pg.ehcache.config.	292
■ pg.email.	292
■ pg.endpoint.	294
■ pg.failedProxies.	294
■ pg.http.	294
■ pg.IntervalPool.	294
■ pg.jaxbFileStore.	295
■ pg.jdbc.	295
■ pg.keystore.	295
■ pg.lb.	296
■ pg.passman.	296
■ pg.PgMenConfiguration.	296
■ pg.PgMenSharedCacheManager.	297
■ pg.PgMetricsFormatter.	297
■ pg.policygateway.	298
■ pg.proxyLoader	298
■ pg.rampartdeploymenthandler.	298
■ pg.ReportingPool.	298
■ pg.ReportingWorkQueue.	299
■ pg.serviceReader.	299

■ wst.rest.json.operation.multiroot.wrapPayload.	299
■ wst.cloudConnectorService.	299
■ wst.connfactory.	300
■ wst.default.tenant.	300
■ wst.product.	300

Introduction

This appendix describes the parameters you can specify in the CloudStreams properties file:

```
Integration Server_directory \instances\instance_name \packages\WmCloudStreams  
\config\resources\wst-config.properties
```

You can edit this file only by using a text editor. Before you edit the file, you should either:

- Shut down the Integration Server, make your changes and then restart the server,
OR
- Make your changes and then reload the WmCloudStreams package.

Note: Some parameters present in the wst-config.properties file can be edited from the CloudStreams Administration console in Integration Server Administrator. You do not have to restart Integration Server when you change parameters through the CloudStreams Administration console.

pg.backupFailedProxies

pg.backupFailedProxies

Indicates whether the backup services failed. The default is `false`.

pg.CollectionPool.

pg.CollectionPool.minThreads

Sets the minimum number of threads used for the Policy Engine. The default is 1.

pg.CollectionPool.maxThreads

Sets the maximum number of threads used for the data collection work queue. The default is 8.

pg.CollectionPool.forcefulShutdown

Specifies whether the data collection thread pool should shut down immediately or wait for queued tasks to complete during CloudStreams shutdown. The default is `false`.

pg.CollectionPool.poolName

Sets the name for the data collection thread pool. The default is `CollectionPool`.

pg.CollectionWorkQueue.

pg.CollectionWorkQueue.queueCapacity

Sets the capacity available for data collection tasks stored in the memory work queue. The default is 10000.

pg.debug.

pg.debug.eventLoggerActive

This is an internal parameter. Do not modify.

pg.delayedRefresher.

CloudStreams cannot query for updates or receive deployed services until Integration Server is running. If Integration Server is not yet fully operational when CloudStreams starts, a delayed refresh helper is used to wait for Integration Server. This helper will periodically check on Integration Server's status.

pg.delayedRefresher.napMillis

Specifies the amount of time (in milliseconds) the delayed refresher helper waits before checking to see whether Integration Server is running. The default is 500.

pg.ehcache.config.

pg.ehcache.config.file

The path to the ehcache configuration file. This path will be provided to Integration Server during the first startup, and subsequent startups will not use the local file; it will use the one under IS/config/Caching. Do not modify.

pg.email.

pg.email.charset

Specifies the character set to use for the subject line, email addresses, and message body of the emails when sending alerts or events. The default is US-ASCII.

pg.email.debug

This is an internal parameter. Do not modify.

pg.email.from

Specifies the email address used when sending events by email. The default is targetName@IS-hostname.

You can edit this parameter from the **From** field on the **CloudStreams > Administration > Email** page in Integration Server Administrator.

pg.email.resourceMimeType

Specifies the MIME type that CloudStreams uses to send the request and response payload attachments for transaction events that are sent by email. CloudStreams supports the following values:

- `application/gzip (.gz)`
- `application/zip (.zip)`
- `text/xml (.xml)`

The default is `application/gzip`.

pg.email.SenderActive

Specifies whether an email server is configured for CloudStreams. The default is `false`.

You can edit this parameter from the **SMTP Host Name/IP Address** field on the **CloudStreams > Administration > Email** page in Integration Server Administrator.

pg.email.smtpHost

Specifies the host name of the email server.

You can edit this parameter from the **SMTP Host Name/IP Address** field on the **CloudStreams > Administration > Email** page in Integration Server Administrator.

pg.email.smtpPort

Specifies the email port for the SMTP or SMTPS protocol. The default is `25`.

You can edit this parameter from the **Port** field on the **CloudStreams > Administration > Email** page in Integration Server Administrator.

pg.email.timeOut

Specifies the time out period (in milliseconds) when connecting to an e-mail server and sending e-mails. The default is `1000`.

pg.email.TLSEnabled

Specifies whether to use one-way transport-layer security (TLS). If set to true, the truststore configured for CloudStreams must include a certificate in the email server's certificate chain. The default is `false`.

You can edit this parameter from the **TLS Enabled** check box on the **CloudStreams > Administration > Email** page in Integration Server Administrator.

pg.email.userName

Specifies the user name of the email account used to log into the SMTP server.

You can edit this parameter from the **User** field on the **CloudStreams > Administration > Email** page in Integration Server Administrator.

pg.endpoint.

pg.endpoint.connectionTimeout

Specifies the time interval (in seconds) after which an HTTP connection attempt will timeout. Default: 30 seconds.

This is a global property that applies to the endpoints of all virtual services. If you prefer to specify a connection timeout for the endpoints of virtual services individually, set the **HTTP Connection Timeout** parameter in the virtual service's "Routing Protocols" processing step. This parameter takes precedence over `pg.endpoint.connectionTimeout`.

pg.endpoint.readTimeout

Specifies the time interval (in seconds) after which a socket read attempt will timeout. Default: 30 seconds.

This is a global property that applies to all virtual services. If you prefer to specify a read timeout for virtual services individually, set the **Read Timeout** parameter in the virtual service's "Routing Protocols" processing step. This parameter takes precedence over `pg.endpoint.readTimeout`.

pg.failedProxies.

pg.failedProxies.backupDir

The absolute or relative path to the `config` directory.

pg.http.

pg.http.ports

Specifies the HTTP ports on which CloudStreams is available.

You can edit this parameter from the **HTTP Ports Configuration** field on the **CloudStreams > Administration > General** page in Integration Server Administrator.

pg.https.ports

Specifies the HTTPS ports on which CloudStreams is available.

You can edit this parameter from the **HTTPS Ports Configuration** field on the **CloudStreams > Administration > General** page in Integration Server Administrator.

pg.IntervalPool.

The interval pool is used to schedule the processing of recurring tasks.

pg.IntervalPool.minThreads

Specifies the minimum thread count for this interval pool. The default is 1.

pg.IntervalPool.maxThreads

Specifies the maximum thread count for this interval pool. The default is 1.

pg.IntervalPool.forcefulShutdown

Specifies whether the interval thread pool should wait for queued tasks to complete during CloudStreams shutdown. Setting this parameter to true will cause CloudStreams to shut down immediately, without waiting for the tasks to finish. The default is `false`.

pg.IntervalPool.poolName

Specifies the name of the interval processor pool. The default is `IntervalPool`.

pg.jaxbFileStore.

pg.jaxbFileStore.consumerFileStore

Specifies the location of the locally persisted consumer applications that CloudStreams receives. This file is updated periodically. The default is `resources\consumers\consumers.xml`. The consumers are populated from the **CloudStreams > Administration > Consumers** page.

pg.jdbc.

pg.jdbc.active

Specifies whether to send events to CloudStreams JDBC tables (assuming no policies are violated).

You can edit this parameter from the **Database Publishing** check box on the **CloudStreams > Administration > Database options** page in Integration Server Administrator.

pg.jdbc.batchSize

Specifies the maximum number of records that the JDBC sender should send in a batch. The default is 25 records.

When the Policy Engine sends events to the database, it uses JDBC batch API for efficiency purposes. This parameter determines the size of the batch before the Datadirect driver sends records from the Integration Server JVM over to the database.

pg.keystore.

pg.keystore.keyStoreHandle

This is an internal parameter. Do not modify.

pg.keystore.trustStoreHandle

This is an internal parameter. Do not modify.

pg.lb.

pg.lb.http.url

Specify the primary HTTP load balancer URL and port number to use in `http://hostname:portnumber` format.

You can edit this parameter from the **Load Balancer URL (HTTP)** field on the **CloudStreams > Administration > General** page in Integration Server Administrator.

pg.lb.https.url

Specify the primary HTTPS load balancer URL and port number to use in `http://hostname:portnumber` format.

You can edit this parameter from the **Load Balancer URL (HTTPS)** field on the **CloudStreams > Administration > General** page in Integration Server Administrator.

pg.lb.failoverOnDowntimeErrorOnly

Controls CloudStreams's behavior for load-balanced endpoints. The default is `false`, which means that in a load-balanced routing scenario, if a service fault is encountered in the response coming from endpoint 1, then CloudStreams will immediately try the next configured endpoint. There is no distinction on the type of fault present in the response from endpoint 1. However, if this parameter is set to `true`, then CloudStreams will failover only if the service fault is a "downtime" error (that is, if it matches one of the strings defined in the file:

```
Integration_Server_directory \instances\instance_name \packages\WmCloudStreams
\config\resources\downtime-patterns.txt
```

pg.passman.

pg.passman.configFile

This is an internal parameter. Do not modify.

pg.PgMenConfiguration.

pg.PgMenConfiguration.perfDataEnabled

Controls whether CloudStreams sends run-time performance data to the relational database destination. The default is `true`.

Note: If this parameter is disabled, CloudStreams will not collect performance metrics.

You can edit this parameter from the **Publish Performance Metrics** check box on the **CloudStreams > Administration > Database options** page in Integration Server Administrator.

pg.PgMenConfiguration.publishErrorEvents

Specifies whether to publish Error events to CloudStreams. The default is `false`.

You can edit this parameter from the **Publish Events (Error, Lifecycle, Policy Violation)** check box on the **CloudStreams > Administration > Database options** page in Integration Server Administrator.

pg.PgMenConfiguration.publishLifeCycleEvents

Specifies whether to publish Life Cycle events to CloudStreams. The default is `false`.

You can edit this parameter from the **Publish Events (Error, Lifecycle, Policy Violation)** check box on the **CloudStreams > Administration > Database options** page in Integration Server Administrator.

pg.PgMenConfiguration.publishPolicyViolationEvents

Specifies whether to publish Policy Violation events to CloudStreams. The default is `false`.

You can edit this parameter from the **Publish Events (Error, Lifecycle, Policy Violation)** check box on the **CloudStreams > Administration > Database options** page in Integration Server Administrator.

pg.PgMenConfiguration.reportInterval

Specifies the how often (in minutes) CloudStreams publishes performance data to CloudStreams. The default is 60 minutes.

You can edit this parameter from the **Publish Interval** field on the **CloudStreams > Administration > Database options** page in Integration Server Administrator.

pg.PgMenConfiguration.sieFlushInterval

Specifies the number of seconds before the accumulated invoked service events are pushed into the shared cache. The default is 2.

pg.PgMenConfiguration.tickInterval

Specifies the amount of time (in seconds) between each interval processor iteration. This must be an evenly divisible fraction of the smallest policy interval, which is one minute. The default is 15.

pg.PgMenSharedCacheManager.

pg.PgMenSharedCacheManager.lockTimeOut

Specifies the amount of time (in seconds) the shared cache waits to obtain a lock before timing out. The default is 5 seconds.

pg.PgMetricsFormatter.

pg.PgMetricsFormatter.includeFaults

Specifies whether service faults are included in the aggregated performance metrics. If set to true, the average, minimum, and maximum response times will include failed

requests in the calculations. The default is `false`. For more information, "[The Key Performance Indicator \(KPI\) Metrics](#)" on page 209.

pg.policygateway.

pg.policygateway.targetName

Sets the name of the CloudStreams instance configured as a target. It is used to identify this instance of CloudStreams. The default is `your-target-name-here`.

You can edit this parameter from the **Target Name** field on the **CloudStreams > Administration > General** page in Integration Server Administrator.

pg.policygateway.repositoryLocation

This is an internal parameter. Do not modify.

pg.policygateway.deleteTempArtifacts

Specifies whether to delete artifacts that are temporarily persisted by the deployment receiver. The default is `true`.

pg.proxyLoader

pg.proxyLoader.proxyLocation

This is an internal parameter. Do not modify.

pg.rampartdeploymenthandler.

pg.rampartdeploymenthandler.signingCertAlias

Specifies the signing alias used to sign the outgoing response from CloudStreams to the original request service consumer.

You can edit this parameter from the **Alias (Signing)** field on the **CloudStreams > Administration > General** page in Integration Server Administrator.

pg.rampartdeploymenthandler.usernameTokenUser

This is an internal parameter. Do not modify.

pg.ReportingPool.

Reporting pool options affect outbound event publishing. The pool includes all events, including performance data, JDBC and SMTP.

pg.ReportingPool.minThreads

Specifies the minimum thread count for this reporting pool. Enabling more threads means that CloudStreams can send more events to the event destination faster. The default is 2.

pg.ReportingPool.maxThreads

Specifies the maximum thread count for this reporting pool. Enabling more threads means that CloudStreams can send more events to the event destination faster. The default is 4.

pg.ReportingPool.forcefulShutdown

Specifies whether the reporting pool should wait for queued tasks to complete during CloudStreams shutdown. Setting this parameter to true will cause CloudStreams to shut down immediately, without waiting for the tasks to finish. The default is true.

pg.ReportingPool.poolName

Specifies the name of the reporting pool. The default is ReportingPool.

pg.ReportingWorkQueue.

pg.ReportingWorkQueue.queueCapacity

Sets the capacity available for data collection tasks stored in the reporting work queue. Smaller in-memory collection and reporting queues reduce memory load on Integration Server, but can result in some messages being discarded. If you configure more memory for Integration Server, it can support more events in the queues. The default is 5000.

pg.serviceReader.

pg.serviceReader.interval

This is an internal parameter. Do not modify.

wst.rest.json.operation.multiroot.wrapPayload.

wst.rest.json.operation.multiroot.wrapPayload

Set this parameter available at *IntegrationServer_directory*\instances *instance_name*\packages\WmCloudStreams\config\resources\wst-config.properties to true if you want to enable the multiple root feature. The default is true.

wst.cloudConnectorService.

wst.cloudConnectorService.validation

Set this parameter to true if you want cloud connector services to be validated against the IS document types present in the service signature. The default is true.

wst.connfactory.

wst.connfactory.wirelogEnabled

Set this parameter to `true` if you want to capture additional debug messages that show the request's HTTP content that is sent over the wire to the native provider. The messages also show the response content that was received. This can be useful when debugging connector service invocations.

You can edit this parameter from the **Connection Factory Wire Logging** check box on the **CloudStreams > Administration > General** page in Integration Server Administrator.

wst.default.tenant.

wst.default.tenant.id

This is an internal parameter. Do not modify.

wst.product.

wst.product.name

This is an internal parameter. Do not modify.