## Contents

*Using This Book* ........................................... vii

**Chapter 1 / SAS Event Stream Manager Overview** ............................................... 1
  What Is SAS Event Stream Manager? ................................................................. 2
  Key Concepts ........................................................................................................ 2
  Access SAS Event Stream Manager .................................................................... 5
  Understanding the User Interface ........................................................................ 5
  Overview of Managing Assets .............................................................................. 11
  Production Assets ................................................................................................. 12

**Chapter 2 / Working with Deployments** .............................................................. 15
  Create a Deployment ............................................................................................... 15
  Edit a Deployment .................................................................................................. 16
  Change the Production Status of a Deployment ................................................... 17
  Delete a Deployment ............................................................................................... 17

**Chapter 3 / Working with ESP Servers** ............................................................... 19
  Add ESP Servers .................................................................................................... 19
  Connecting Directly to an ESP Server .................................................................. 21
  Edit an ESP Server ................................................................................................. 23
  Remove ESP Servers .............................................................................................. 24

**Chapter 4 / Working with Projects** .............................................................. 27
  View a Project ......................................................................................................... 27
  (Optional) Upload a Project ................................................................................... 31
  Change the Production Status of a Project ........................................................... 33

**Chapter 5 / Working with Filters** ................................................................ 35
  Filters ...................................................................................................................... 35
  Working with Filters for a Specific Deployment .................................................... 36
  Managing Filters ..................................................................................................... 38
  Filter Expression Syntax ...................................................................................... 40
### Chapter 6 / Working with Job Templates

- Job Template ........................................... 43
- Create a Job Template ............................... 44
- Upload a Job Template .............................. 46
- Edit a Job Template .................................. 47
- Change the Production Status of a Job Template ........................................... 48
- Deploy a Job Template .............................. 48
- Download a Job Template .......................... 49
- Delete a Job Template ............................... 50

### Chapter 7 / Example Job Templates

- Overview of Example Job Templates ............ 51
- Example Job Template: Stock Trade .......... 52
- Example Job Template: Stop a Project ......... 62

### Chapter 8 / Job Template Content

- Overview of Job Template Content ............. 66
  - job-template ......................................... 66
  - localization-strings ......................... 69
  - parameters ........................................ 71
  - enumerations ..................................... 81
  - server-filters .................................... 82
  - instructions ....................................... 85
  - failure-instructions ......................... 98
  - initialization ..................................... 101

### Chapter 9 / Monitoring a Deployment

- Monitor Deployments .............................. 105
  - View Sample Data ................................. 111
  - Update SAS Micro Analytic Service Modules ........................................... 114
  - Monitor Unassigned ESP Servers ............... 116
  - Monitor Jobs ....................................... 118
  - Delete a Job ........................................ 121
  - Rerun a Job ......................................... 121
  - Stop a Running Job ............................... 122
Contents
Audience

The target audience for SAS Event Stream Manager: User’s Guide is SAS Event Stream Processing application administrators. SAS Event Stream Manager enables application administrators to effectively manage the SAS Event Stream Processing environment, including deploying SAS Event Stream Processing projects to ESP servers, monitoring deployments, and managing change.

Requirements for Solution Access and Use

Here are the requirements for accessing and using SAS Event Stream Manager:

- You have a user ID and password for logging on to SAS Event Stream Manager.

- A supported web browser has been installed.

  Note: For detailed information about supported browsers, in SAS Event Stream Manager click the user icon in the top-right corner and then click About. The About window appears. Click Supported browsers and platforms to view supported browsers. SAS Event Stream Manager requires the use of cookies to maintain the session state.

- Your screen has a minimum screen resolution of 1,280 x 1,024.

- JavaScript has been enabled in your browser.
Purpose of This Document

This document provides usage and operational assistance for the typical user of SAS Event Stream Manager.

Note: Each deployment of SAS Event Stream Manager is configured to meet the needs of a specific customer. The information and windows, including the color and layout themes, shown in this document are intended to show the functionality of SAS Event Stream Manager and are only examples.
What Is SAS Event Stream Manager?

SAS Event Stream Manager is a web-based client that enables you to manage your SAS Event Stream Processing environment.

You can use SAS Event Stream Manager to perform the following tasks:

- deploy SAS Event Stream Processing projects into production environments and test environments
- monitor the status of your deployments
- administer your deployments and manage change
- monitor your SAS Event Stream Processing metering servers

You can use SAS Event Stream Processing Studio to create the projects that you deploy to ESP servers using SAS Event Stream Manager.

Key Concepts

This topic introduces the key concepts in SAS Event Stream Manager.

Asset

An asset is any identifiable part of a SAS Event Stream Processing deployment that SAS Event Stream Manager can monitor or act upon. Examples of assets include projects and ESP servers.

SAS Event Stream Manager tracks changes to assets that it manages. Here are examples of details recorded by SAS Event Stream Manager for most assets:

- creation timestamp
- last update timestamp
Deployment

A deployment is a logical grouping of assets as a single unit for life cycle management and monitoring in SAS Event Stream Manager. That is, a deployment is a group of assets that together does something useful.

Project

A project is a data processing model that can be executed on an ESP server.

A project is stored as an XML document. You can create a project in SAS Event Stream Processing Studio, or you can upload a project to it. When you publish the project using SAS Event Stream Processing Studio, it becomes visible in SAS Event Stream Manager.

Here are examples of project metadata captured by SAS Event Stream Manager:

- version
- project repository ID
- whether the project is a production asset (for more information, see “Production Assets” on page 12)

ESP Server

In SAS Event Stream Processing, the ESP server is an engine-executable program that instantiates and executes SAS Event Stream Processing projects.
An ESP server and the equipment on which it is running can have different states. For example, an ESP server might not be running even though the server on which it is located is available.

Here are examples of ESP server metadata captured by SAS Event Stream Manager:

- status
- the host on which the ESP server is running
- the SAS Event Stream Processing version that is installed on the host on which the ESP server is running
- memory use and CPU use

**Job Template**

A job template is an XML document that contains a set of instructions to create a job. That is, a job template outlines the steps required to load, start, stop, or unload a project on an ESP server.

You can create a job template by using an editor provided within SAS Event Stream Manager. A job template contains the following high-level elements:

- localizations
- parameters
- enumerations
- initializations
- instructions
- failure instructions

When you deploy a job template, a job is created.

**Job**

A job is a set of tasks to be executed by SAS Event Stream Manager on various assets associated with one or more deployments.
A job includes the tasks to be performed on assets (such as projects and ESP servers) and the status of each task.

Here are examples of job metadata that SAS Event Stream Manager records:
- job status
- the deployment that the job relates to
- start date and time

Projects can include placeholders where values are injected to customize project behavior at run time. These placeholder values can be recorded and displayed as part of the details of a job.

Jobs cannot be changed after they are created. If you want to alter a job, you must create a new job instead.

---

**Access SAS Event Stream Manager**

1. Open the following URL:

   https://host/SASEventStreamManager

   The *host* is the system on which SAS Event Stream Manager is installed.

   The Sign In to SAS window appears.

2. Enter your user ID and password, and click **Sign In**.

---

**Understanding the User Interface**

**Pages**

A *page* is the highest level container in the user interface. All other user interface elements are contained within the confines of a page.
When you access SAS Event Stream Manager, the **Deployments** page appears.

**Figure 1.1  Deployments Page with Three Deployments**

SAS Event Stream Manager contains the following main pages:

- The **Deployments** page enables you to create deployments and then monitor and manage them.

- The **Projects** page enables you to view projects that have been published using SAS Event Stream Processing Studio.

- The **Unassigned Servers** page enables you to view available ESP servers that SAS Event Stream Manager is aware of, and to connect directly to other ESP servers. ESP servers that already belong to a deployment are not displayed.

- The **Job Templates** page enables you to create and upload job templates, which you can then deploy to create running jobs.
The Filters page enables you to manage filters that are available in SAS Event Stream Manager.

The Metering page enables you to monitor the metering servers that track usage data.

The Log page displays details of currently running jobs and historical jobs.

For more information about the main tasks that you perform using SAS Event Stream Manager, see “Overview of Managing Assets” on page 11.

**Panes**

SAS Event Stream Manager pages contain panes. The following figure shows the Filters page, which contains a bottom pane with a tile called Details.

*Figure 1.2 Example of a Page with a Pane*

<table>
<thead>
<tr>
<th>Name</th>
<th>Usage</th>
<th>Tags</th>
<th>Updated</th>
<th>Updated By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Deployment 1</td>
<td></td>
<td>13/03/2019, 10:01:22</td>
<td>fsduser</td>
</tr>
<tr>
<td>SSL</td>
<td>Any Deployment</td>
<td></td>
<td>13/03/2019, 09:58:31</td>
<td>fsduser</td>
</tr>
<tr>
<td>Test servers</td>
<td>Any Deployment</td>
<td>testing</td>
<td>13/03/2019, 09:57:55</td>
<td>fsduser</td>
</tr>
<tr>
<td>version 6.1</td>
<td>Deployment 2</td>
<td></td>
<td>13/03/2019, 10:00:15</td>
<td>fsduser</td>
</tr>
</tbody>
</table>

To resize a pane, drag a border that is marked with .... upward or downward.
To hide a pane, click 📦. To display it again, click the same button again.

**Tiles**

A *tile* is a self-contained block of information that resides within a pane or sometimes directly on a page. The same tile can appear on several pages. For example, the *Details* tile appears in the bottom pane on more than one page.

**Windows**

A *window* is a floating user interface element that often appears as a result of a user action. Windows generally provide a means by which to perform an action. Closing a window returns you to the page from which the window was launched. The following figure shows a window that can create a new deployment in SAS Event Stream Manager.

*Figure 1.3  Example of a Window*
Toolbars

There are three main toolbars in SAS Event Stream Manager, as shown in the following figure. For information about each toolbar, see the subsequent table.

Figure 1.4 SAS Event Stream Manager Toolbars

<table>
<thead>
<tr>
<th>Item Number</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application bar</td>
<td>Displays your user icon, which shows the first character of your name or user ID. Click the user icon to access the following functionality:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- View Help and product information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Sign out from SAS Event Stream Manager (unless your system has been configured so that you do not need to sign in)</td>
</tr>
<tr>
<td>2</td>
<td>Menu bar</td>
<td>- Provides access to the main SAS Event Stream Manager pages: Deployments, Projects, Unassigned Servers, Job Templates, Filters, Metering, and Log.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Provides access to each deployment, ESP server, project, job template, or job that is currently open. The navigation overflow menu button displays the total number of these pages that are currently open, for example, <img src="image" alt="Overflow Menu Button" /></td>
</tr>
<tr>
<td>3</td>
<td>Toolbar or tabs</td>
<td>Includes buttons or tabs specific to the open page. For example, the preceding figure shows toolbar buttons on the Job Templates page. The following figure shows tabs on a page for an open project.</td>
</tr>
</tbody>
</table>
Arrange Information in Tables

Sometimes a large amount of information is displayed in tables. To make it easier to work with a large amount of information, you can arrange this information in different ways.

You can sort the contents of many columns by ascending or descending order. To do this, click the heading of the column that you want to sort.

You can create filter criteria by which to display only a subset of information for a column. To create filter criteria, click \(^\text{Filter}\) for the column that you want to apply filter criteria to, select Filter, and enter your filter criteria. The use of filter criteria is not available for some columns.

You can configure the columns that you want to display. To do this, click \(^\text{Columns}\) in any column, select Columns, and deselect the columns that you do not want to appear.

You can re-order columns. To do this, click and hold the column heading and drag it to a different location.

In some tables, you can group information by column. To do this, click \(^\text{Group columns}\) and select Group columns. If this option is not available, it means that you cannot group.
information by columns in this table. A horizontal bar appears at the top of the table, with the text **Drag a column heading here to group by that column.** To group information by column, drag a column heading to the bar. If required, you can drag additional columns to the bar. In the following example, information about the **Job Templates** page has been grouped by the Production column:

*Figure 1.6  Example of Grouping Information by Column*

---

**Overview of Managing Assets**

Before SAS Event Stream Manager can interact with assets, you must do the following:

1. **Create a deployment.**
   
   For more information, see  “Create a Deployment” on page 15.

2. **Associate an ESP server with the deployment.**
   
   For more information, see  “Add ESP Servers” on page 19.

3. **Create or upload a project in SAS Event Stream Processing Studio, and publish it.**
   
   When you publish a project in SAS Event Stream Processing Studio, it becomes visible in SAS Event Stream Manager.

   For more information, see  “View a Project” on page 27.
4 Create or upload a job template.

For more information about creating a new job template, see “Create a Job Template” on page 44. For more information about uploading a job template, see “Upload a Job Template” on page 46.

5 Deploy a job template.

For more information, see “Deploy a Job Template” on page 48.

TIP Instead of deploying a job template that you have created or uploaded, in some cases you can use controls in the user interface to load and start a project. For more information, see “Load and Start a Project” on page 123.

For more information about the meaning of the terms mentioned here, see “Key Concepts” on page 2.

This guide includes an end-to-end example of how to use SAS Event Stream Manager to complete the tasks listed above. For more information, see “Overview of the Stock Trade Example” on page 133.

Production Assets

You can mark certain assets as production assets to prevent assets meant for testing from accidentally being used for production (that is, being used in a live environment). The following assets can be marked as production assets:

- Deployments
- Projects
- Job templates

You can mark these assets as production assets when you create them. You can change the production status to a different status later. For example, you can change a production deployment to a non-production deployment, or you can change a non-production deployment to a production deployment.
You can deploy a non-production job template only against a non-production deployment. In such a situation, SAS Event Stream Manager permits you to select either a production project or a non-production project to deploy. This choice enables you to test your current production projects or to test non-production projects to assess whether they are suitable for being marked as production projects in the future.

You can deploy a production job template against a production deployment or against a non-production deployment:

- If you deploy a production job template against a production deployment, SAS Event Stream Manager permits you to select only a production project to deploy.
- If you deploy a production job template against a non-production deployment, SAS Event Stream Manager permits you to select either a production project or a non-production project to deploy.

The following diagram illustrates this information:

*Figure 1.7  Options for Deploying Job Templates*
Working with Deployments

Create a Deployment ................................................................. 15
Edit a Deployment ................................................................. 16
Change the Production Status of a Deployment ......................... 17
Delete a Deployment ............................................................... 17

Create a Deployment

1 On the Deployments page, click .

The Deployment Properties window appears.

2 In the Name field, enter a unique name for the deployment.

3 In the Description field, enter a description for the deployment. For example, you can enter the purpose of the deployment to allow users to differentiate between deployments with similar names. This description is displayed as a tooltip when you place the cursor over the deployment name in the Name column on the Deployments page.

4 In the Tags field, you can attribute one or more tags to the deployment.

To add a tag, enter text and press Enter. Tags can be used to group and filter deployments. Tags are single-term descriptors for the deployment. Tags cannot contain spaces. Duplicate tags are not permitted on a single deployment.
If you want to create a production deployment, select the Production deployment check box. For more information, see “Production Assets” on page 12.

Click OK.

Your deployment appears on a new page. The following figure shows an example:

---

Edit a Deployment

1. On the Deployments page, select the deployment that you want to edit and click .

   The Deployment Properties window appears.

2. Edit the Description and Tags fields as required. You can also select or deselect the Production deployment check box to move a deployment from test to production or from production to test. For more information about these fields, see “Create a Deployment” on page 15.

   Note: You cannot edit the Name field.
If required, you can add or remove ESP servers, or filter ESP servers associated with the deployment:

- For more information about adding ESP servers, see “Add ESP Servers” on page 19.
- For more information about removing ESP servers, see “Remove ESP Servers” on page 24.
- For more information about filtering ESP servers, see “Working with Filters for a Specific Deployment” on page 36.

**Change the Production Status of a Deployment**

Certain assets can be marked as production assets. For more information, see “Production Assets” on page 12.

To change the production status of a deployment, on the **Deployments** page, right-click the deployment and select **Set as production** or **Remove as production**.

The Production column on the **Deployments** page changes to reflect your choice.

**Delete a Deployment**

You can delete a deployment if no ESP servers are associated with it. For more information, see “Remove ESP Servers” on page 24.

To delete a deployment:

1. On the **Deployments** page, select the deployment that you want to delete.
2. Click .

   The Delete Deployment window appears.
3 Click **Delete**.
Add ESP Servers

You can associate an ESP server with a deployment only if SAS Event Stream Manager is already aware of the ESP server. SAS Event Stream Processing informs SAS Event Stream Manager about ESP servers that exist in the SAS Event Stream Processing environment. You can configure the esm.xml file in SAS Event Stream Processing to enable SAS Event Stream Processing to pass this information to SAS Event Stream Manager. For more information about the esm.xml file in a Linux environment, see “Configure the ESP Server for SAS Event Stream Manager” in SAS Event Stream Manager for Linux: Deployment Guide. For more information about the esm.xml file in a Windows environment, see “Start the ESP Server” in SAS Event Stream Manager for Windows: Deployment Guide.

If SAS Event Stream Manager is not aware of a particular ESP server, you can connect directly to that ESP server. A direct connection to a specific ESP server might be necessary if the esm.xml file has not been configured or the ESP server is running with
an earlier version of SAS Event Stream Processing (version 5.2 or earlier). In this case, SAS Event Stream Manager becomes aware of the ESP server only after you have made the direct connection. For more information, see “Connecting Directly to an ESP Server” on page 21.

Each ESP server can belong to only one deployment. If you want to add an ESP server that already belongs to another deployment, you must first remove that ESP server from its original deployment. For more information, see “Remove ESP Servers” on page 24.

Each deployment can contain several ESP servers.

To add ESP servers to a deployment:

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .

   A separate page that displays the deployment opens.

2. Click and select Add an unassigned server.

   Note: The New server menu option, which is adjacent to the Add an unassigned server menu option, relates to making a direct connection to an ESP server that SAS Event Stream Manager has not yet detected. For more information, see “Connecting Directly to an ESP Server” on page 21.

   The Add and Remove ESP Servers window appears.

3. Move the required ESP servers from the Available servers table to the Selected servers table, using the buttons between the two tables:

   - To move a single ESP server, select the ESP server in the Available servers table and click .

   - To move all ESP servers that are displayed in the Available servers table, click .

4. Click OK.
The ESP servers appear in the table on the deployment page. To view more information about a particular ESP server, select it in the main table and view the information that appears in the tabs in the bottom pane. The following figure shows an example:

![Connecting Directly to an ESP Server](image)

**Connecting Directly to an ESP Server**

You can associate an ESP server with a deployment only if SAS Event Stream Manager is already aware of the ESP server. SAS Event Stream Processing informs SAS Event Stream Manager about ESP servers that exist in the SAS Event Stream Processing environment. You can configure the esm.xml file in SAS Event Stream Processing to enable SAS Event Stream Processing to pass this information to SAS Event Stream Manager. For more information about the esm.xml file in a Linux environment, see “Configure the ESP Server for SAS Event Stream Manager” in *SAS Event Stream Manager for Linux: Deployment Guide*. For more information about the esm.xml file in a Windows environment, see “Start the ESP Server” in *SAS Event Stream Manager for Windows: Deployment Guide*.
If SAS Event Stream Manager is not aware of a particular ESP server, you can connect directly to that ESP server. A direct connection to a specific ESP server might be necessary if the esm.xml file has not been configured or the ESP server runs with an earlier version of SAS Event Stream Processing (version 5.2 or earlier).

To connect directly to an ESP server:

1. On the **Unassigned Servers** page, click ![Unassigned Servers Icon].
   
The ESP Server Properties window appears.

2. In the **Name** field, enter a name for the ESP server.

3. In the **Host** field, enter the host name of the server that the ESP server runs on.

4. In the **HTTP port** field, replace the default value with the port number used for HTTP administration requests and for the HTTP publish/subscribe server.
   
   **Note:** This port is specified in the `-http` argument to the `dfesp_xml_server` command in SAS Event Stream Processing.

5. In the **Description** field, enter a description for the ESP server. For example, you can specify the purpose of this ESP server, to allow users to differentiate between ESP servers with similar names.

6. In the **Tags** field, you can attribute one or more tags to the ESP server.
   
   To add a tag, enter text and press Enter. Tags can be used to group and filter ESP servers. *Tags* are single-term descriptors for the ESP server. Tags cannot contain spaces. Duplicate tags are not permitted on a single ESP server.

7. If required, change the setting for the **Authentication** field:
   
   - **None**: This is the default option.
   
   - **Kerberos**: This option is relevant only if the ESP server is configured to require authentication using Kerberos.
- **OAuth token**: This option is relevant only if the ESP server is configured to require authentication using an OAuth token. If you select this option, an additional field appears where you must enter the OAuth token.

- **Username and password**: This option is relevant only if the ESP server is configured to require authentication using a user name and password (SASLogon Services). If you select this option, additional fields appear where you must enter the user name and password.

8. If required, select the **Connect using SSL** check box. Selecting this check box is relevant only if the ESP server is configured to require SSL encryption.

9. Click **OK**.

The ESP server is displayed on the **Unassigned Servers** page, along with your other ESP servers. The **Unassigned Servers** page does not indicate which ESP servers were detected automatically and which ones you connected to directly.

You can now associate the ESP server with a deployment. For more information, see “Add ESP Servers” on page 19.

---

**Edit an ESP Server**

1. Open the ESP server if it is not already open: on the **Unassigned Servers** page, select the ESP server that you want to open and click **Edit**.

   The Edit ESP Server Properties window appears.

2. Edit the fields as required:
   - **Name**: Edit the name of the ESP server.
   - **Tags**: You can attribute one or more tags to the ESP server. To add a tag, enter text and press Enter. Tags can be used to group and filter ESP servers. **Tags** are single-term descriptors for the ESP server. Tags cannot contain spaces. Duplicate tags are not permitted on a single ESP server.
Description: Edit the description for the ESP server. For example, you can specify the purpose of this ESP server, to allow users to differentiate between ESP servers with similar names.

3 If required, change the setting for the Authentication field:

- None: This is the default option.
- Kerberos: This option is relevant only if the ESP server is configured to require authentication using Kerberos.
- OAuth token: This option is relevant only if the ESP server is configured to require authentication using an OAuth token. If you select this option, an additional field appears where you must enter the OAuth token.
- Username and password: This option is relevant only if the ESP server is configured to require authentication using a user name and password (SASLogon Services). If you select this option, additional fields appear where you must enter the user name and password.

4 If required, select the Connect using SSL check box. Selecting this check box is relevant only if the ESP server is configured to require SSL encryption.

5 Click OK.

Remove ESP Servers

Remove ESP Servers from a Deployment

Removing an ESP server from a deployment enables you to add that ESP server to another deployment.

Before you can remove an ESP server from a deployment, you must stop and unload any projects that were previously running on the ESP server. For more information, see “Stop and Unload a Project” on page 124.
To remove ESP servers from a deployment:

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .

   A separate page that displays the deployment opens.

2. Select the ESP servers that you want to remove. You can select multiple ESP servers by holding down the Ctrl key and using your mouse to select the ESP servers. You can also select all ESP servers by clicking .

3. Click .

4. If you want to remove the ESP servers from the deployment, select **Return the ESP servers to the list of unassigned servers** if you want to remove the ESP servers from the deployment. If you want to remove the ESP servers from SAS Event Stream Manager, select **Remove the ESP servers permanently**.

Remove ESP Servers from SAS Event Stream Manager

Removing an ESP server from SAS Event Stream Manager means that SAS Event Stream Manager is not aware of that ESP server. The ESP server itself continues to exist.

To remove an ESP server from SAS Event Stream Manager, you must first remove it from any deployment that it belongs to. For more information, see “Remove ESP Servers from a Deployment” on page 24.

If you made a direct connection to an ESP server, you can remove the ESP server from SAS Event Stream Manager by following the steps in this topic. However, if SAS Event Stream Manager is aware of certain ESP servers because the esm.xml configuration file is used in SAS Event Stream Processing, you cannot remove those ESP servers using the SAS Event Stream Manager user interface. For more information about the esm.xml file in a Linux environment, see “Configure the ESP Server for SAS Event Stream Manager” in SAS Event Stream Manager for Linux: Deployment Guide. For more
information about the esm.xml file in a Windows environment, see “Start the ESP Server” in SAS Event Stream Manager for Windows: Deployment Guide.

To remove ESP servers that are not assigned to any deployment from SAS Event Stream Manager:

1. On the **Unassigned Servers** page, select the ESP servers that you want to remove.

2. Click ![trash can icon].
   
   The Remove ESP Server window appears.

3. Click **Yes**.

SAS Event Stream Manager is no longer aware of the ESP servers. However, the ESP servers themselves continue to exist.
Working with Projects

View a Project ................................................................................................. 27
Overview ......................................................................................................... 27
The Diagram Tab .......................................................................................... 28
The XML Tab ............................................................................................... 29
The Details Tab ............................................................................................ 29
The Versions Tab ........................................................................................ 30
The Files Tab ............................................................................................... 31

(Optional) Upload a Project .......................................................................... 31

Change the Production Status of a Project .................................................. 33

View a Project

Overview

You can create a project in SAS Event Stream Processing Studio or upload a project to SAS Event Stream Processing Studio. If you publish the project using SAS Event Stream Processing Studio, it becomes visible in SAS Event Stream Manager. If you delete a published project in SAS Event Stream Processing Studio, it remains visible in SAS Event Stream Manager.

Note: If SAS Event Stream Processing Studio is running as a stand-alone application, you cannot use it to publish projects. For a project to be visible in SAS Event Stream Manager, you must download it from SAS Event Stream Processing Studio and then
upload it to SAS Event Stream Manager. You can upload a project to SAS Event Stream Manager only if SAS Event Stream Processing Studio is running as a stand-alone application. For more information, see “(Optional) Upload a Project” on page 31.

To view detailed information about a project, select the project on the Projects page and click . A page with five tabs appears.

**The Diagram Tab**

The **Diagram** tab displays a graphical representation of the project’s windows and edges. Each window can display icons that represent its state. For more information, see “Window Icons” in *SAS Event Stream Processing: Using SAS Event Stream Processing Studio*. The following figure shows an example of the **Diagram** tab:

You can use the buttons on the toolbar to zoom in and out of the view, or to zoom to fit the view.
The XML Tab

The **XML** tab displays a read-only view of the XML used to construct the model. You can collapse XML elements to restrict your view to the elements that you are interested in. The following figure shows an example of the **XML** tab:

```
1. <xml version="1.0"/>
2. <project heartbeat-interval="1" name="filteredtrades" threads="8" publish="auto">  
3. <description>Find trades that match a specified stock code.</description>  
4. </project>
5. ...
6. ...
7. ...
8. ...
9. ...
10. ...
11. ...
12. ...
13. ...
14. <contquery name="cq">
15.     ...
```

Read-only

The Details Tab

The **Details** tab displays general information relating to the project entered when the current version of the project was uploaded. The following figure shows an example:
The Versions Tab

The **Versions** tab displays the current version and any previous versions of the project. The following figure shows an example:
To open a previous version, right-click it and select **Open project version**. A separate page appears where you can review the metadata and XML content relating to that version.

The major version is updated when you publish a new version of the project in SAS Event Stream Processing Studio. For example, the version is updated from 1.0 to 2.0.

The minor version is updated when you update a project that references a model that is stored in the SAS Model Manager common model repository, and a new champion model is declared in SAS Model Manager. For example, SAS Event Stream Manager updates the project from version 1.0 to 1.1. For more information, see “Update SAS Micro Analytic Service Modules” on page 114.

**The Files Tab**

The **Files** tab displays analytics model files that are loaded from the SAS Model Manager common model repository and are executed through SAS Micro Analytic Service modules and the Calculate window. The following figure shows an example:

![Files Tab Example](image)

(Optional) Upload a Project

If SAS Event Stream Processing Studio is running as a stand-alone application, you cannot use it to publish projects. For a project to be visible in SAS Event Stream Manager, you must download it from SAS Event Stream Processing Studio and then upload it to SAS Event Stream Manager. You can upload a project to SAS Event Stream Manager only if SAS Event Stream Processing Studio is running as a stand-alone application. For more information about downloading a project from SAS Event Stream

Before SAS Event Stream Manager accepts an uploaded project, it validates the project’s content against an active ESP server. Therefore, an ESP server must be available to SAS Event Stream Manager before you attempt to upload a project.

To upload a project to SAS Event Stream Manager:

1. On the Projects page, click : and select Upload projects.
   The Upload Projects window appears.

2. Click .

3. Navigate to the file that contains the project that you want to upload and click Open.
   
   **Note:** If you want to upload multiple projects that are located in the same folder, you can select the relevant projects and upload them simultaneously. To select multiple projects, hold down the Ctrl key and select each project that you want to upload, and click Open. If your projects are located in different folders, click again, select one or more relevant projects, and click Open.

4. Click Upload.
   An icon is displayed to indicate whether the project was successfully uploaded. Successfully uploaded projects are indicated by the icon . Projects that failed to upload are indicated by the icon .

5. Click OK.
   The projects that you uploaded appear on the Projects page.

If the project fails validation, an error message appears. You can use the information in the message to correct the issue and then upload the project again.

When you upload a new version of a project that was updated in SAS Event Stream Processing Studio, the project’s major version number is updated in SAS Event Stream
Manager. For example, in SAS Event Stream Processing Studio, the version number in the project XML file is incremented from 1.0 to 2.0, and the updated version is displayed in SAS Event Stream Manager. However, if you update the project XML file manually (that is, outside SAS Event Stream Processing Studio), then the version number is always set to 1.0. The version number is set to 1.0 even if the major and minor version numbers are set manually in the project XML file. To ensure that the version number is incremented as expected, update the project in SAS Event Stream Processing Studio.

### Change the Production Status of a Project

Certain assets can be marked as production assets. For more information, see “Production Assets” on page 12.

To change the production status of a project, on the Projects page, right-click the project and select Set as production or Remove as production.

The Production column on the Projects page changes to reflect your choice.
Filters

SAS Event Stream Manager enables you to search for ESP servers that match certain criteria and save those criteria as a filter that you can reference in job templates. For example, you might want to deploy a project to ESP servers that have the tag `primary`. If the number of ESP servers that have this tag changes, a filter that references this tag still finds any matching ESP servers.

A filter contains a filter expression. The filter expression specifies the search criteria for ESP servers.
You can specify that a filter is available only to a specific deployment or that a filter is available for use with all deployments. For example, this means that the filters available to a deployment called Deployment1 include filters specific to that deployment as well as filters available to all deployments. However, filters specific to another deployment called Deployment2 are not available to Deployment1.

In addition, the SAS Event Stream Manager user interface can show the result of applying a filter. This means that for a given deployment, the user interface lists the ESP servers that match the filter criteria. This functionality is different from applying a filter. A filter is applied when a job template that references the filter is deployed.

You can reference filters from the server-filters and server-filter-selector elements in job templates. You can also specify a filter expression directly in the server-filters element. For more information about the server-filters element, see “server-filters” on page 82. For more information about the server-filter-selector element, see “server-filter-selector” on page 80.

In addition to creating filters yourself, there is one situation where SAS Event Stream Manager constructs a filter expression that you can then save as a filter. If a job fails to execute on some ESP servers, SAS Event Stream Manager enables you to save a filter that references those failed servers. For more information, see “Rerun a Job” on page 121.

---

**Working with Filters for a Specific Deployment**

When you open a deployment, you can work with filters that are available for that specific deployment.

To work with filters that are available for use with all deployments, use the **Filters** page instead. For more information, see “Managing Filters” on page 38.
Show the Result of Applying a Filter

Showing the result of applying a filter means showing only those ESP servers that match a selected filter. This functionality is different from applying a filter. A filter is applied when a job template that references the filter is deployed.

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .
   A separate page opens to display the deployment.

2. Click on the toolbar and select Advanced Filtering.
   The Advanced Filtering area appears on the page.

3. In the drop-down list that displays the text Enter a filter expression, either select an existing filter or enter a filter expression. For more information, see “Filter Expression Syntax” on page 40.

4. Click Apply to see what the result of applying this filter would be.
   The table on the page changes to display only those ESP servers that match the filter expression.

You can save your filter for further use. For more information, see “Save a Filter” on page 37.

To display all ESP servers in the table again, click Clear.

Save a Filter

1. Complete the steps in “Show the Result of Applying a Filter” on page 37.

2. Click Save.
   The Filter Properties window appears.

3. Edit the fields as required:
Managing Filters

The Filters page enables you to manage filters that are available in SAS Event Stream Manager. This includes filters that are available for use only with specific deployments as well as filters that are available for use with all deployments.

Sort Filters

You can arrange the information in the table on the Filters page in several ways. However, you might find it particularly helpful to sort the table using the Usage column, so that filters available to different deployments or to all deployments are listed together.
For more information about sorting tables, see “Arrange Information in Tables” on page 10.

The following figure shows an example where filters have been sorted using the Usage column:

![Filters Window](image)

### Create a Filter

The **Filters** page enables you to create a filter by entering a filter expression. However, you cannot check the result of applying your filter on this page. This functionality is available only when you open the page for a specific deployment and enter a filter expression there. For more information, see “Working with Filters for a Specific Deployment” on page 36.

1. Click ![Filter Icon].

   The Filter Properties window appears.

2. Edit the fields as required:
   - **Name**: The name of the filter.
   - **Description**: A description of the purpose of the filter.
   - **Tags**: One or more tags attributed to the filter.

   To add a tag, enter text and press Enter. Tags can be used to group and sort filters. *Tags* are single-term descriptors for the filter. Tags cannot contain spaces. Duplicate tags are not permitted on a single filter.
Filter expression: A valid expression that defines the filter criteria. For more information, see “Filter Expression Syntax” on page 40.

Scope: Select One deployment and then select the desired deployment, or select Any deployment.

3 Click OK.

Edit a Filter

1 Click Edit.

The Filter Properties window appears.

2 Edit the fields as required. For more information about how to use the fields in this window, see “Create a Filter” on page 39.

3 Click OK.

Delete a Filter

1 Click Delete.

The Delete Filter window appears.

2 Click Delete.

Filter Expression Syntax

Here are examples of supported filter expressions:

- `hostname='server1234.example.com'`
  Finds ESP servers that are running on the host server1234.example.com.
- `label~'Test*'`
  Finds ESP servers whose names begin with the characters `Test` and are followed by any other characters. For example, this filter finds an ESP server with the name `Test_1` but not `ESP_Test_1` or `test_1`.

  **Note:** If the name of the ESP server contains an apostrophe, enclose the filter term in double quotation marks, and escape the apostrophe with a backslash or an apostrophe. For example, the filter `label~"SiteA\'s*"` finds an ESP server with the name `SiteA's_ESP_server`.

- `((tags='tag1')|(tags='tag2'))`
  Finds ESP servers that have the tag `tag1` or `tag2` or both.

- `((tags='tag1')&(tags='tag2'))`
  Finds ESP servers that have the tags `tag1` and `tag2`.

- `((tags='tag1')&(tags='tag3')&!(tags='tag5'))`
  Finds ESP servers that have the tags `tag1` and `tag3`, but do not have the tag `tag5`.

- `metering=true`
  Finds ESP servers whose hosts have metering enabled.

- `version='6.1'`
  Finds ESP servers that run with SAS Event Stream Processing 6.1.

- `ssl=true`
  Finds ESP servers with SSL authentication.

- `analyticsLicensed=true`
  Finds ESP servers whose hosts have a SAS Event Stream Processing Analytics license installed.
Working with Job Templates

Job Template ................................................................. 43
Create a Job Template .............................................. 44
Upload a Job Template ............................................... 46
Edit a Job Template .................................................... 47
Change the Production Status of a Job Template .......... 48
Deploy a Job Template .................................................. 48
Download a Job Template .............................................. 49
Delete a Job Template ................................................... 50

Job Template

A job template is an XML document that contains a set of instructions used to create a job. That is, a job template outlines the steps required to execute a project on an ESP server. When you deploy a job template, a job is created.

The job template also specifies what input you must supply when you deploy the job template. For example, when you deploy a job template that contains an instruction to load a project, SAS Event Stream Manager prompts you to select the project that you want to load.

A job template contains the following high-level elements:
- localization-strings – a collection of strings that display localized output
- parameters – a collection of user parameters that allow a user to enter data into the deployment and customize a job template when it is deployed
- enumerations – a collection of definitions that restrict user input when used with user parameters
- initialization – a collection of definitions to replace placeholders to alter a project when it is deployed
- server-filters – a collection of definitions that specify filters for ESP servers
- instructions – a collection of instructions that describe operations that must be performed to create or modify a deployment
- failure-instructions – a collection of instructions that attempt to return the system to normal operation after an instruction has failed to execute

If the job template contains a parameters element, you are prompted to enter each of the user parameters when you deploy the job template. SAS Event Stream Manager displays a list of projects and ESP servers that are available.

The string values from a job template’s localization-strings element are used to construct labels used when specifying user parameters. After you have selected your user parameters, SAS Event Stream Manager produces and executes a job derived from the job template and the user parameters that you entered. As the job executes, you receive progress updates. The string values from the localization-strings element are used to display labels for each instruction.

For example job templates, see “Overview of Example Job Templates” on page 51.

For more information about job template contents, see “Overview of Job Template Content” on page 66.

Create a Job Template

1. On the Job Templates page, click .
The Create New Job Template window appears.

2 In the **Name** field, enter a unique ID for the job template.

3 In the **Description** field, you can enter the purpose of the job template.

4 In the **Tags** field, you can attribute one or more *tags* to the job template.

   To add a tag, enter text and press Enter. Tags can be used to group and filter job templates. Tags are single-term descriptors for the job template. Tags cannot contain spaces. Duplicate tags are not permitted on a single job template.

5 If you want to create a production job template, select the **Production template** check box. For more information, see “**Production Assets**” on page 12.

6 In the **Version notes** field, you can add notes to further describe the job template.

   Adding notes is especially useful if you want to track changes between versions of the same job template within SAS Event Stream Manager.

7 Click **OK**.

   Your new job template appears on the **Job Templates** page. The following figure shows an example:

![Job Templates page example](image)

You must now add content to your job template to make it functional. For more information, see “**Edit a Job Template**” on page 47.
Upload a Job Template

To upload a job template:

1. On the **Job Templates** page, click ![Upload job template](image) and select **Upload job template**.

   The Upload Job Template window appears.

2. In the **File** field, click **Browse** and navigate to the location of your job template.

   Job templates must be in XML format.

3. In the **Tags** field, you can attribute one or more tags to the job template.

   To add a tag, enter text and press Enter. Tags can be used to group and filter job templates. **Tags** are single-term descriptors for the job template. Tags cannot contain spaces. Duplicate tags are not permitted on a single job template.

4. If you want this job template to be a production job template, select the **Production template** check box. For more information, see “Production Assets” on page 12.

5. In the **Version notes** field, you can add notes to further describe the job template.

   Adding notes is especially useful if you want to track changes between versions of the same job template within SAS Event Stream Manager.

6. Click **OK**.

   When successfully uploaded, the job template appears on the **Job Templates** page. The following figure shows an example:
Note: If the job template fails validation, an error message appears. You can use the information in the message to correct the issue and then upload a revised job template.

## Edit a Job Template

1. On the **Job Templates** page, select the job template that you want to edit and click ![Edit](edit_icon.png).

   A separate page appears, displaying the XML that defines the job template. The following figure shows an example:

   ![XML of a job template](xml_template.png)

   The following table describes the buttons on the XML editor toolbar:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Action</th>
<th>Keyboard Shortcut</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="rev_icon.png" alt="Rev" /></td>
<td>Reverts your previous change.</td>
<td>Ctrl + Z</td>
</tr>
</tbody>
</table>

2. Edit the content of the job template as required. For more information, see “Overview of Job Template Content” on page 66.

   The following table describes the buttons on the XML editor toolbar:
<table>
<thead>
<tr>
<th>Icon</th>
<th>Action</th>
<th>Keyboard Shortcut</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔄</td>
<td>Reverts the effects of the undo action.</td>
<td>Ctrl + Y</td>
</tr>
<tr>
<td>🔍</td>
<td>Searches for specific text.</td>
<td>Ctrl + F</td>
</tr>
<tr>
<td>📝</td>
<td>Formats manually entered XML code.</td>
<td>not available</td>
</tr>
</tbody>
</table>

3 Click 🔄. Alternatively, you can click 📝 and select **Save template as** to save your changes as a new job template.

**Change the Production Status of a Job Template**

Certain assets can be marked as production assets. For more information, see “Production Assets” on page 12.

To change the production status of a job template, on the **Job Templates** page, right-click the job template and select **Set as production** or **Remove as production**.

The Production column on the **Job Templates** page changes to reflect your choice.

**Deploy a Job Template**

1 On the **Job Templates** page, select a job template and click 🔄.

   The Job Template window appears.

2 In the **Deployment** field, select the deployment to which you want to deploy the job template.
The Job Template window is refreshed to display additional fields, which reflect the parameters element of the job template that you are deploying. Complete these additional fields.

**Note:** By default, SAS Event Stream Manager deploys the latest version of the project. To deploy a different version, select a value from the **Project version** dropdown list. You cannot specify the minor version number — instead, SAS Event Stream Manager always uses the latest minor version of the specified major version.

3. **Click OK.**

SAS Event Stream Manager creates a job from the job template that you deployed. The result of running the job is displayed on a new page. The following figure shows an example:

![Job Result](image)

You can now monitor the running job. For more information, see “Monitor Jobs” on page 118.

You can also monitor the deployment. For more information, see “Monitor Deployments” on page 105.

For information about stopping a running job, see “Stop a Running Job” on page 122.

---

**Download a Job Template**

1. On the **Job Templates** page, select the job template that you want to download.
2 Click ⚙️ and select **Download job template**.

The job template is downloaded to your computer. The location of the downloaded job template might vary depending on your browser’s configuration.

---

### Delete a Job Template

You can delete a job template if there are no running jobs that reference it.

To delete a job template:

1  On the **Job Templates** page, select the job template that you want to delete.

2  Click ⚡.

   The **Delete Job Template** window appears.

3  Click **Yes** to confirm the deletion.
Example Job Templates

Overview of Example Job Templates

Example Job Template: Stock Trade
Overview ................................................................. 52
Job Template Outline ............................................... 52
Adding an Enumeration ............................................. 54
Adding the load-project Instruction .............................. 55
Adding Parameters for the load-project Instruction ........ 57
Adding String References .......................................... 59
Adding the project-transformation Instruction ............... 59
Adding a Filter ....................................................... 60
The Complete File .................................................. 60

Example Job Template: Stop a Project ......................... 62

Overview of Example Job Templates

The topics in this chapter describe example job templates:

- For a job template that loads and starts a project, see “Example Job Template: Stock Trade” on page 52.
- For a job template that stops and unloads a project, see “Example Job Template: Stop a Project” on page 62.
These job templates are also used as part of an end-to-end example of how SAS Event Stream Manager works. For more information, see “Overview of the Stock Trade Example” on page 133.

Example Job Template: Stock Trade

Overview

This example describes a job template that loads and starts a SAS Event Stream Processing project, trades_connector_stocksymbol_job_template.xml.

An outline of the job template is created first, and then detail is added to each of the high-level elements in the job template. The complete job template XML is displayed at the end. For more information, see “The Complete File” on page 60.

The complete job template file is included in the SAS Event Stream Manager examples package, which you can download from SAS Event Stream Manager examples.

This job template is also used as part of another example, where it is used together with other resources to create a deployment and deploy a job. For more information, see “Overview of the Stock Trade Example” on page 133.

Job Template Outline

Here is the outline for the trades_connector_stocksymbol_job_template.xml file:

```xml
<?xml version="1.0"?>
<job-template id="FilteredTrades"
  localization-id="jobName"
  description-localization-id="jobDescription">

  <localization-strings default-language="en-gb">
    <language id="en-gb">
      <string id="jobName">Filter Trades by Stock Code</string>
      <string id="jobDescription">Deploy filtered trades demo with data connectors</string>
    </language>
  </localization-strings>
</job-template>
```
The top-level element, `job-template`, has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>A unique identifier given to the job template.</td>
</tr>
<tr>
<td>localization-id</td>
<td>A reference to a string value for the job template name used in SAS Event Stream Manager.</td>
</tr>
<tr>
<td>description-localization-id</td>
<td>A reference to a string value for the job template description used in SAS Event Stream Manager.</td>
</tr>
</tbody>
</table>

The `localization-strings` element in the job template outline stores the string identifier and values for each supported language. Within the `localization-strings` element, there is a `language` element whose `id` attribute is set to `en-gb` (United Kingdom English). This `language` element contains `string` elements. The `jobName` and `jobDescription` strings are referenced by the `localization-id` and `description-localization-id` attributes in the `job-template` element at the beginning of the file. When the job template is uploaded and then displayed in SAS Event Stream Manager, these values are displayed as metadata about the job template.

The `enumerations` element in the job template outline is populated with elements that specify user input options.
The parameters element in the job template outline is populated with elements to gather the user input required for executing instructions.

The initialization element in the job template outline is populated with elements to alter SAS Event Stream Processing projects as the projects are deployed.

The server-filters element in the job template defines a filter for ESP servers. When the job template is deployed, the filter resolves to a specific ESP server.

The instructions element in the job template outline guides SAS Event Stream Manager through the process of creating or modifying a deployment. It is populated with an instruction to start a SAS Event Stream Processing model.

Adding an Enumeration

An enumeration is a finite list of options that is presented to the user to restrict input. The enumeration is referenced from the parameters element of the job template. In this example, the user is limited to selecting from three stocks. The enumeration uses string identifiers for the localized string values that are displayed in SAS Event Stream Manager during execution.

The XML enumerations element looks like this:

```xml
<enumerations>
  <enumeration id="symcode">
    <enumeration-value id="MCRO" localization-id="mcr" />
    <enumeration-value id="SPY" localization-id="spy" />
    <enumeration-value id="LQD" localization-id="lqd" />
  </enumeration>
</enumerations>
```

The enumeration element has an attribute called id, which is a unique identifier given to this enumeration.

Each enumeration-value child element has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>A unique identifier given to the enumeration value.</td>
</tr>
</tbody>
</table>
### Attribute Table

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>localization-id</td>
<td>A reference to a string value for the enumeration value when it is displayed in SAS Event Stream Manager.</td>
</tr>
</tbody>
</table>

### Adding the load-project Instruction

Instructions tell SAS Event Stream Manager what to do with assets in a deployment. In this example, a `load-project` instruction is added to load and then start a SAS Event Stream Processing project.

Like other SAS Event Stream Manager instructions, the `load-project` instruction uses string identifiers for the localized string values that are displayed in SAS Event Stream Manager during execution. It also has XML attributes for parameters to identify required input for the project to load and identify the ESP server, as well as other additional parameters. These parameters can be specified as literal values in the XML or as references to user-defined input parameters. In this example, user-defined input parameters are specified.

The XML `instructions` element looks like this:

```xml
<instructions>
  <load-project
    id="load-project"
    localization-id="load-project"
    server-filter="filter1"
    project-id="{transformed-project.id}"
    project-version="{project.version}"
    running-project-name="{symcode}_{project.version}"
    start="true" />
</instructions>
```

The XML element name, `load-project`, is the instruction name. In this example, the XML line has been split to make it more readable. In a functional job template, it is more likely to be on a single line.
The **load-project** instruction has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></td>
<td>A unique identifier given to the instruction. All instructions require this attribute, and its value must be unique in the job template.</td>
</tr>
<tr>
<td><strong>localization-id</strong></td>
<td>A reference to a string value for the instruction when it is displayed in SAS Event Stream Manager.</td>
</tr>
<tr>
<td><strong>server-filter</strong></td>
<td>A reference to a filter for ESP servers. A filter resolves to a set of ESP servers to which the project will be deployed. The filter is defined in the <code>server-filters</code> element within the job template.</td>
</tr>
<tr>
<td><strong>project-id</strong></td>
<td>A unique identifier is created internally for each SAS Event Stream Processing project that has been published in SAS Event Stream Processing Studio. This value is specified by using a variable placeholder indicated by the use of the braces, <code>{</code> and <code>}</code>. The variable in this case is <code>{transformed-project.id}</code>, which is explained in the initialization section.</td>
</tr>
<tr>
<td><strong>project-version</strong></td>
<td>Each project that has been published in SAS Event Stream Processing Studio is versioned. This value is specified by using a variable placeholder indicated by the use of the braces, <code>{</code> and <code>}</code>. The variable in this case is <code>{project.version}</code>, which is referenced and explained in the parameters section.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Purpose</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>running-project-name</td>
<td>A name for the running project that will be created in SAS Event Stream Processing. This value must be unique to the running ESP server that will execute the project. This value is specified by using variable placeholders indicated by the use of the braces, { and }. In this case, two variables are specified with a literal underscore ( _ ) to create a project name, {symcode}_ {project.version}. These variables are explained in the parameters section.</td>
</tr>
<tr>
<td>start</td>
<td>The value of this attribute tells SAS Event Stream Manager to start running the SAS Event Stream Processing project after it has been loaded. In this case, the value is true. If it was false, an additional instruction, start-project, would be required to start the project.</td>
</tr>
</tbody>
</table>
Three selectors are added to the `parameters` element:

```xml
<parameters>
  <project-selector id="project"
                  localization-id="project" />
  <server-selector id="server"
                  localization-id="server" />
  <enum-selector id="symcode"
               localization-id="symcode"
               required="true"
               source="symcode" />
</parameters>
```

In this example, each selector element is split onto several lines to make it more readable. In a functional job template, it is more likely to be on a single line.

The `localization-id` attributes are discussed later (for more information, see “Adding String References” on page 59).

The attributes for the selectors are similar. Each of the attributes is described as follows:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>A unique identifier given to the user parameter.</td>
</tr>
<tr>
<td>localization-id</td>
<td>A reference to a string value for this parameter when it is displayed in SAS Event Stream Manager.</td>
</tr>
<tr>
<td>required</td>
<td>If set to <code>true</code>, the user must provide an input value.</td>
</tr>
<tr>
<td>source</td>
<td>This attribute is used only for <code>enum-selector</code>. It provides information about the source of the data. The value must be the name of a defined enumeration.</td>
</tr>
</tbody>
</table>

The `project-selector` element provides the user with a drop-down list that contains the SAS Event Stream Processing projects that have been published in SAS Event Stream Processing Studio. When a value is selected, the `id` and `version` properties of the selected project can be referenced throughout the job template. You can reference these properties by using the `id` attribute, with either a `.id` or `.version` suffix.
The server-selector element provides the user with a drop-down list that contains the ESP servers that have been added to SAS Event Stream Manager. When a value is selected, the id and version properties of the selected project can be referenced throughout the job template. You can reference these properties by using the id attribute, with either a .id or .version suffix.

The enum-selector element provides the user with a drop-down list that contains values from the referenced enumeration. That is, this selector provides a list of stock codes.

**Adding String References**

The load-project instruction and the three selectors within the parameters element all specify localization-id attribute values that need to be added to the localization-strings element of the job template. Each id attribute must be associated with a string value. These string values are used by SAS Event Stream Manager when prompting for input or when showing progress or error messages when executing a job created from the job template.

The string references are added under the language id="en-gb" element from the outline XML defined earlier. The string references are as follows:

```
<string id="project">Project</string>
<string id="server">ESP server</string>
<string id="load-project">Load Project</string>
<string id="symcode">Stock symbol code</string>
<string id="mcr">Micro Focus</string>
<string id="spy">SPDR</string>
<string id="lqd">LQD</string>
```

**Adding the project-transformation Instruction**

An instruction to replace placeholders when a project is deployed needs to be added to the initialization element. Replacing the placeholders is achieved by using the project-transformation instruction.

The XML initialization element looks like this:

```
<initialization>
  <project-transformations>
    ...
  </project-transformations>
</initialization>
```

Example Job Template: Stock Trade
Variables used for \{project\} are resolved at execution when the user selects a project. The placeholder \{symcode\} is replaced with the stock code selected by the user.

**Adding a Filter**

A server-filters element must be added to define a filter for ESP servers. In this example, the filter resolves to the ESP server selected by the user when the job template is deployed. However, in a more complicated example, the filter might resolve to a larger set of ESP servers, and the project might be deployed to more than one ESP server.

The XML server-filters element looks like this:

```xml
<server-filters>
  <server-filter id="filter1"
    filter-expression="objectId='{server.id}'" />
</server-filters>
```

The filter expression contains a \{server.id\} variable. This variable references the server-selector element within the parameters element.

In this example, the server-filters element is added between the initialization and instructions elements.

**The Complete File**

Here is the complete trades_connector_stocksymbol_job_template.xml file:

```xml
<?xml version="1.0"?>
<job-template id="FilteredTrades"
  localization-id="jobName"
  description-localization-id="jobDescription">
  <localization-strings default-language="en-gb">
    <language id="en-gb">
```
<string id="jobName">Filter Trades by Stock Code</string>
<string id="jobDescription">Deploy filtered trades demo with data connectors</string>
<string id="project">Project</string>
<string id="server">ESP server</string>
<string id="load-project">Load Project</string>
<string id="symcode">Stock symbol code</string>
<string id="mcro">Micro Focus</string>
<string id="spy">SPDR</string>
<string id="lqd">iBoxx</string>
</language>
</localization-strings>
<enumerations>
  <enumeration id="symcode">
    <enumeration-value id="MCRO" localization-id="mcro" />
    <enumeration-value id="SPY" localization-id="spy" />
    <enumeration-value id="LQD" localization-id="lqd" />
  </enumeration>
</enumerations>
<parameters>
  <project-selector id="project" localization-id="project" />
  <server-selector id="server" localization-id="server" />
  <enum-selector id="symcode" localization-id="symcode"
    required="true" source="symcode" />
</parameters>
<initialization>
  <project-transformations>
    <project-transformation id="transformed-project"
      project-id="\{project.id\}"
      project-version="\{project.version\}">
      <placeholder id="symcode">\{symcode\}</placeholder>
    </project-transformation>
  </project-transformations>
</initialization>
<server-filters>
  <server-filter id="filter1"
    filter-expression="objectId='\{server.id\}'" />
</server-filters>
<instructions>
  <load-project id="load-project"
    localization-id="load-project"
    server-filter="filter1"
    project-id="\{transformed-project.id\}"
Example Job Template: Stop a Project

This example describes a simple job template, stop_project_job_template.xml, that stops a running project and unloads it from the ESP server.

The complete job template file is included in the SAS Event Stream Manager examples package, which you can download from SAS Event Stream Manager examples.

This job template is also used as part of another example. For more information, see “Overview of the Stock Trade Example” on page 133.

Here is the complete stop_project_job_template.xml file:

```xml
<?xml version="1.0"?>
<job-template id="clean-server" localization-id="jobName"
    description-localization-id="jobDescription">
    <localization-strings>
        <language id="en-gb">
            <string id="jobName">Remove a Running Project</string>
            <string id="jobDescription">Unload a project from a specified ESP server</string>
            <string id="target-server">ESP server</string>
            <string id="project-to-stop">Project to unload</string>
            <string id="stop-project">Stop Project</string>
            <string id="unload-project">Unload Project</string>
        </language>
    </localization-strings>
    <parameters>
        <server-selector id="server"
            localization-id="target-server"
            required="true" />
        <running-project-selector id="project-to-stop"
            localization-id="project-to-stop"
            required="true" />
    </parameters>
</server-filters>
```
The localization-strings element stores the string identifier and values for each supported language. Within the localization-strings element, there is a language element whose id attribute is set to en-gb (United Kingdom English). This language element contains string elements. The jobName and jobDescription strings are referenced by the localization-id and description-localization-id attributes in the job-template element at the beginning of the file. When the job template is uploaded and then displayed in SAS Event Stream Manager, these values are displayed as metadata about the job template.

The parameters element contains selectors that gather the user input required to execute instructions.

The server-filters element defines a filter for ESP servers. The {server.id} variable references the server-selector element within the parameters element. In this example, the filter resolves to the ESP server selected by the user when the job template is deployed. However, in a more complicated example the filter might resolve to a larger set of ESP servers.

The instructions element contains instructions for stopping the running project specified by the user and unloading it from the ESP server. For more information about the stop-project instruction, see “stop-project” on page 95. For more information about the unload-project instruction, see “unload-project” on page 95.
Job Template Content

Overview of Job Template Content ................................................................. 66

job-template .................................................................................................. 66

localization-strings ......................................................................................... 69

parameters ...................................................................................................... 71
  Overview of the parameters Element ............................................................ 71
  Common Attributes of Selectors .................................................................. 73
  project-selector ............................................................................................. 74
  server-selector .............................................................................................. 75
  running-project-selector ............................................................................. 76
  text-input-selector ....................................................................................... 77
  query-selector ............................................................................................... 79
  window-selector ........................................................................................... 79
  enum-selector ................................................................................................. 79
  server-filter-selector .................................................................................. 80

enumerations ..................................................................................................... 81

server-filters .................................................................................................. 82

instructions ..................................................................................................... 85
  instructions .................................................................................................... 85
  Common Attributes of Instructions and the group Element ....................... 88
  load-project .................................................................................................. 90
  start-project ................................................................................................ 92
  start-connectors ......................................................................................... 93
Overview of Job Template Content

The topics in this chapter contain information about the content that you can add to job templates.

For a high-level introduction to job templates, see “Job Template” on page 43.

For an overview of example job templates, see “Overview of Example Job Templates” on page 51.

job-template

The job-template element is the top-level element in a job template.

Here is an example:

```xml
<job-template id="test1"
  localization-id="template-name"
  description-localization-id="template-description"/>

<!-- The rest of the job template contents are added here. -->

</job-template>
```

Table 8.1 Properties of the job-template XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>id</td>
</tr>
<tr>
<td>localization-id</td>
</tr>
<tr>
<td>description-localization-id</td>
</tr>
<tr>
<td>Child Elements</td>
</tr>
<tr>
<td>localization-strings</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Element</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>parameters</td>
</tr>
<tr>
<td>enumerations</td>
</tr>
<tr>
<td>initialization</td>
</tr>
<tr>
<td>server-filters</td>
</tr>
<tr>
<td>instructions</td>
</tr>
</tbody>
</table>
failure-instructions

This element contains instructions that attempt to return the system to normal operation after an instruction has failed to execute.

- Required: No.
- For more information, see “failure-instructions” on page 98.

localization-strings

The localization-strings element of the job template contains the string values for the labels displayed by SAS Event Stream Manager when executing that job template. Each string value is fully localizable within one or more language groups. The localization-strings element requires a default language.

Here is an example:

```xml
<job-template id="test1"
  localization-id="template-name"
  description-localization-id="template-description">
  <localization-strings default-language="en-us">
    <language id="en-us">
      <string id="template-name">test1</string>
      <string id="template-description">Loads a project</string>
    </language>
  </localization-strings>
</job-template>
```

Each string element has an id attribute that is referred to by other elements of the job template when a string value is required. In the example here, template-name and template-description are referred to in the top-level job-template element.

In the SAS Event Stream Manager user interface, the string values are displayed in the following locations:

- on the Job Templates page
- in the window that appears when you deploy a job template from the Job Templates page
on the Log page when the job created from the job template is executing

**Table 8.2  Properties of the localization-strings XML Element**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default-language</td>
<td>This attribute specifies the language to be used if the user’s current locale is not supported. Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: This value must correspond to a defined language identifier. The value must be a valid ISO locale code. The locale code must be lowercase and in the format aa-bb (for example: en-us).</td>
</tr>
</tbody>
</table>

**Child Element**

| language           | There must be one language element for each supported language and at least one that corresponds to the default-language attribute of the parent localization-strings element. |

**Table 8.3  Properties of the language XML Element**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the locale code for this language. Required: Yes. Valid values: A valid ISO locale code. The locale code must be lowercase and in the format aa-bb (for example: en-us).</td>
</tr>
</tbody>
</table>

**Child Element**

| string    | There must be one string element for each localization string.                                                                           |
Overview of the parameters Element

The parameters element of the job template tells SAS Event Stream Manager what user input to capture and assign to internally stored variables that can be referenced throughout the job template. The parameters element includes child elements whose purpose is to guide user input. These selectors restrict the user's text input or enable the user to select an option from a list. In the following example, the project-selector element enables the user to select from a list of projects stored in SAS Event Stream Manager’s repository:

```xml
<parameters>
  <project-selector id="projectSelector"
    localization-id="projectSelectorLabel"
    required="true"/>
</parameters>
```

You can nest some selectors inside other selectors. For example, information might be needed to populate available values for the selector in the user interface. In addition, you might need to show some parameters only upon selection of a valid value for another parameter. In such cases, the selectors should be nested inside each other, as shown in the following example:

```xml
<parameters>
  <project-selector id="projectSelector"
    localization-id="projectSelectorLabel"
    required="true">
    <query-selector id="querySelector"
      localization-id="checkQuery"
      required="true">
      <window-selector id="windowSelector"
        localization-id="checkWindow"
        required="true"/>
    </query-selector>
  </project-selector>
</parameters>
```
### Table 8.4 Properties of the parameters XML Element

<table>
<thead>
<tr>
<th>Child Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>project-selector</td>
<td>This element enables the user to select from a list of projects stored in SAS Event Stream Manager’s repository. For more information, see “project-selector” on page 74.</td>
</tr>
<tr>
<td>server-selector</td>
<td>This element enables the user to select from a list of ESP servers that are managed by SAS Event Stream Manager. For more information, see “server-selector” on page 75.</td>
</tr>
<tr>
<td>running-project-selector</td>
<td>This element enables the user to select a project that is either running on a specific SAS Event Stream Processing server or contained in a project that has been published in SAS Event Stream Processing Studio. For more information, see “running-project-selector” on page 76.</td>
</tr>
<tr>
<td>text-input-selector</td>
<td>This element enables the user to enter a text string that is checked against a regular expression. For more information, see “text-input-selector” on page 77.</td>
</tr>
<tr>
<td>query-selector</td>
<td>This element enables the user to select a currently running continuous query inside a project on a specific SAS Event Stream Processing server. For more information, see “query-selector” on page 79.</td>
</tr>
</tbody>
</table>
| window-selector              | This element enables the user to select a currently running window within a continuous query inside a project on a specific SAS Event Stream Processing server. For more information, see “window-selector” on page 79. }
enum-selector

This element enables the user to select from a list of restricted values of an enumeration defined either globally or locally in the deployment. For more information, see “enum-selector” on page 79.

server-filter-selector

This element enables the user to select a previously created filter. For more information, see “server-filter-selector” on page 80.

Common Attributes of Selectors

Selectors have some common attributes, as shown in the following table:

**Table 8.5  Common Attributes of Selectors**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the unique identifier of the selector. This identifier is placed into the execution context.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: This value must match the regular expression [0-9A-z][0-9A-Z_-]+.</td>
</tr>
<tr>
<td>localization-id</td>
<td>This attribute specifies the localization string ID.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: This value must exist in the default language information within the localization-strings element.</td>
</tr>
</tbody>
</table>
required

This attribute indicates whether it is mandatory for the user to provide input. If you set this attribute to `true`, the user must provide input, and an asterisk (*) is displayed next to the user interface control to indicate this to the user.

- Required: No.
- Valid values: `true` and `false`.

validation-localization

This attribute specifies the localization string ID that is displayed if validation fails.

- Required: Yes.
- Valid values: This value must exist in the default language information within the `localization-strings` element.

When a user selects the value of a selector, that value is stored internally by SAS Event Stream Manager. You can reference this value in the job template by the selector’s `id` value. To do this, enclose the `id` value in braces: `{ }`.

An `id` value can have one or more attributes. To access these attributes, use a dot (.) notation. For example, if the selector’s `id` value is `alpha` and this `id` value has the attribute `beta`, you can reference the selector by writing `{alpha.beta}`. Additional examples are provided in the following sections, where each selector is discussed in more detail.

**project-selector**

The `project-selector` element enables the user to select from a list of projects stored in SAS Event Stream Manager’s repository. This action produces a project object that has the following variables:

- `id`: the unique identifier of the project that has been selected by the user
- `version`: the major version number of the project that has been selected by the user
- `friendlyName`: the name of the project that has been selected by the user
The friendlyName project object can be particularly useful if you need to identify the project using its name rather than its identifier. For example, some SAS Event Stream Processing adapters need to reference the project name.

The project-selector element is a top-level selector and does not need to be nested inside any other selectors to function properly.

When deploying a job template, the user is prompted to select the project version. By default, the latest major version of the project is selected.

The project-selector element is useful for choosing a project that is then published to the ESP server. An example from a simplified job template is shown here:

```xml
<parameters>
  <project-selector id="projectSelector"
    localization-id="projectSelectorLabel"
    required="true" />
  <server-selector id="serverSelector"
    localization-id="serverLabel"
    required="true" />
</parameters>

<server-filters>
  <server-filter id="serverFilter"
    filter-expression="(objectId='{serverSelector.id}')" />
</server-filters>

<instructions>
  <load-project id="loadProjectInstruction"
    localization-id="loadProjectLabel"
    server-filter="serverFilter"
    project-id="{projectSelector.id}"
    project-version="{projectSelector.version}"
    running-project-name="{projectSelector.friendlyName}"
    start="true" />
</instructions>
```

**server-selector**

The server-selector element enables the user to select from a list of ESP servers that are associated with the selected deployment. This action produces an ESP server object that has the following variable:

**id**: the unique identifier of the ESP server that has been selected by the user
The server-selector element is a top-level selector and does not need to be nested inside any other selectors to function properly.

Alongside the project-selector element, the server-selector element is one of the most commonly used parameter types. The server-selector element is typically used to help define a server-filter element, as shown in the simplified example job template here:

```xml
<parameters>
  <project-selector id="projectSelector"
      localization-id="projectSelectorLabel"
      required="true" />
  <server-selector id="serverSelector"
      localization-id="serverLabel"
      required="true" />
</parameters>
<server-filters>
  <server-filter id="serverFilter"
      filter-expression="(objectId='{serverSelector.id}')" />
</server-filters>
<instructions>
  <load-project id="loadProjectInstruction"
      localization-id="loadProjectLabel"
      server-filter="serverFilter"
      project-id="{projectSelector.id}"
      project-version="{projectSelector.version}"
      running-project-name="{projectSelector.friendlyName}"
      start="true" />
</instructions>
```

**running-project-selector**

The running-project-selector element enables the user to select a project that is either running on a specific SAS Event Stream Processing server or contained in a project that has been published in SAS Event Stream Processing Studio. This action produces a project object that has the following variable:

id: the unique identifier of the project that has been selected by the user

The running-project-selector element must be nested inside a server-selector element to function properly.
Not all job templates relate to new installations. In some cases, you might need to make changes to existing projects or unload existing projects to complete the deployment successfully. The example here demonstrates how the following tasks are performed:

1. prompts the user to select an ESP server
2. prompts the user to select a project
3. stops the selected project running on that server

```xml
<parameters>
  <server-selector id="serverSelector"
      localization-id="serverLabel"
      required="true"
  >
    <running-project-selector
      id="runningProjectSelector"
      localization-id="runningProjectLabel"
      required="true" />
  </server-selector>
</parameters>
<server-filters>
  <server-filter id="serverFilter"
      filter-expression="(objectId='{serverSelector.id}')" />
</server-filters>
<instructions>
  <stop-project id="stopProjectInstruction"
      localization-id="stopProjectLabel"
      server-filter="serverFilter"
      running-project-name="{runningProjectSelector.friendlyName}" />
</instructions>
```

**Note:** The `stop-project` instruction stops the project. To entirely unload a project, use the `unload-project` instruction. For more information, see “unload-project” on page 95.

### text-input-selector

The `text-input-selector` element enables the user to enter a text string that is checked against a regular expression. This action produces a string object that can be used throughout the job template.

The `text-input-selector` element is a top-level selector and does not need to be nested inside any other selectors to function properly.
In addition to properties common to all selectors (for more information, see “Common Attributes of Selectors” on page 73), the text-input-selector element has the XML attribute shown in the following table:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validation-expression</td>
<td>This attribute defines the regular expression that is used to perform validation.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: The valid regular expression.</td>
</tr>
</tbody>
</table>

In the following example, the user’s choice of a project name is restricted so that the name must begin with a letter. The remainder of the string must contain letters, numbers, or underscores. If validation fails, the localized string defined in project-not-valid is displayed to the user.

```xml
<parameters>
  <project-selector id="projectSelector"
    localization-id="projectSelectorLabel"
    required="true" />
  <server-selector id="serverSelector"
    localization-id="serverLabel"
    required="true" />
  <text-input-selector id="projectName"
    localization-id="invalidProjectLabel"
    required="true"
    validation-expression="^[A-z][A-z0-9_]+$"
    validation-localization-id="project-not-valid" />
</parameters>
<server-filters>
  <server-filter id="filter1"
    filter-expression="(objectId='{server.id}')" />
</server-filters>
<instructions>
  <load-project id="loadProjectInstruction"
    localization-id="loadProjectLabel"
    server-filter="serverFilter"
    project-id="{projectSelector.id}"
    project-version="{projectSelector.version}""
    running-project-name="{projectSelector}"
```
query-selector

The query-selector element enables the user to select a currently running continuous query inside a project on a particular SAS Event Stream Processing server. This action produces a query object with the following variable:

id: the unique identifier of the query that has been selected by the user

The query-selector element must be nested inside a project-selector element or a running-project-selector element to function properly.

window-selector

The window-selector element enables the user to select a currently running window within a continuous query inside a project on a particular SAS Event Stream Processing server. This action produces a query object with the following variable:

id: the unique identifier of the window that has been selected by the user

The window-selector element must be nested inside a query-selector element to function properly.

enum-selector

The enum-selector element enables the user to select from a list of restricted values of an enumeration defined either globally or locally in the deployment. This action produces a string that can then be used anywhere in the job template.

The enum-selector element is a top-level selector and does not need to be nested inside any other selectors to function properly.

In addition to properties common to all selectors (for more information, see “Common Attributes of Selectors” on page 73), the enum-selector element has the XML attribute shown in the following table:
Table 8.7 Properties of the enum–selector XML Element

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>This attribute provides additional information about the source of the data.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: The name of a defined enumeration.</td>
</tr>
</tbody>
</table>

For more information about enumerations, see “enumerations” on page 81.

**server-filter-selector**

The `server-filter-selector` element enables the user to select a filter. A separate window is displayed to the user. The window lists filters available only to the specific deployment as well as filters available for use with all deployments. The window also provides additional details (such as the filter expression) for each filter to help the user select an appropriate filter.

For more information, see “Filters” on page 35.

The `server-filter-selector` element does not need to be nested inside any other selectors to function properly.

Here is an example:

```xml
<parameters>
  <server-filter-selector id="failedServerFilter"
   required="true"
   localization-id="serverFilterLabel" />
</parameters>

<server-filters>
  <server-filter id="filter1"
    filter-expression="\{failedServerFilter.filterExpression\}">
  </server-filter>
</server-filters>

<instructions>
  <group id="group1" localization-id="groupLabel"
   server-filter="filter1">
    <!-- appropriate instructions are added here -->
  </group>
</instructions>
```
This simplified example shows a server-filter-selector element called serverFilter. This element is then referenced in a server-filter element called filter1. This means that the set of ESP servers included in filter1 is the set of ESP servers returned by whichever filter a user selects for the serverFilter selector. Within the instructions element, the group element called group1 then refers to filter1. In this way, you can specify a group of instructions to execute against a set of ESP servers that a user selects. The strings shown in this example must be referenced in the localization-strings element, which is not shown in the example code here.

**enumerations**

An *enumeration* is a finite list of options that is presented to the user to restrict input.

Each job template can support the declaration of local localized enumerations. Such declarations are optional. The enumerations element of the job template specifies a set of enumerations that can be used in the deployment. Each enumeration is fully localized and produces output to the user in the user’s specified locale (subject to the job template supporting the locale). Here is an example enumerations element that defines a single enumeration:

```
<enumerations>
  <enumeration id="country">
    <enumeration-value id="USA" localization-id="united-states" />
    <enumeration-value id="UK" localization-id="united-kingdom" />
  </enumeration>
</enumerations>
```

The enumeration with the id value of country can be referenced in the job template from the parameters element when you are limiting the input options the user can select from. In this example, the user is limited to the values USA and UK.

<table>
<thead>
<tr>
<th>Table 8.8</th>
<th>Properties of the enumerations XML Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Child Element</td>
<td></td>
</tr>
</tbody>
</table>
There must be one `enumeration` element for each enumeration.

**Table 8.9  Properties of the enumeration XML Element**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>id</code></td>
<td>This attribute specifies the unique identifier of the enumeration. If a global enumeration with the same identifier is defined, this one takes precedence.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: This value must match the regular expression <code>[0-9A-z][0-9A-Z_-]+</code>.</td>
</tr>
</tbody>
</table>

**Child Element**

- `enumeration-value`: Each distinct value in the enumeration must be listed as a child element.

---

**server-filters**

The `server-filters` element specifies filters for ESP servers. A filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated. For more information about how filters work, see “Filters” on page 35.

**Table 8.10  Properties of the server-filters XML Element**

<table>
<thead>
<tr>
<th>Child Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>server-filter</code></td>
<td>This element specifies a filter for ESP servers.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>id</td>
<td>This attribute specifies the unique identifier of the server-filter element.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must match the regular expression [0-9A-z][0-9A-Z_-]+.</td>
</tr>
<tr>
<td>localization-id</td>
<td>This attribute specifies the localization string ID.</td>
</tr>
<tr>
<td></td>
<td>If a job fails, SAS Event Stream Manager enables the user to create a filter that references the failed ESP servers (for more information, see “Rerun a Job” on page 121). The job template that was used to create the failed job can include multiple server-filter elements. In this case, SAS Event Stream Manager creates a separate filter for each ESP server group on which the job failed. (For more information about the group element, see “group” on page 96.) The id attribute of the relevant server-filter element is added to the filter name that is entered by the user.</td>
</tr>
<tr>
<td></td>
<td>If the server-filter element includes a localization-id attribute, this attribute's value is used instead of the id attribute to provide a friendly, localizable name. If a translated string is available in the user's language, that translation is used. If no translated string is available, the value of the id attribute is used instead.</td>
</tr>
<tr>
<td></td>
<td>- Required: No.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must exist in the default language information within the localization-strings element.</td>
</tr>
</tbody>
</table>
filter-expression

This attribute contains a filter expression or references a previously saved filter (which contains a filter expression).

- Required: Yes.
- Valid values: A valid filter expression or the name of a previously saved filter. For more information, see “Filter Expression Syntax” on page 40.

Here is an example where the filter-expression attribute references an existing filter called failedserverfilter1:

```xml
<server-filters>
  <server-filter id="filter1" filter-expression="\{failedserverfilter1.filterExpression\}" />
</server-filters>
```

Here is a second example, where the filter-expression attribute includes a filter expression:

```xml
<server-filters>
  <server-filter id="filter1" filter-expression="\{objectId='\{server.id\}\}'" />
</server-filters>
```

In this second example, the filter expression contains a \{server.id\} variable. This code extract is from the Stock Trade example, where the \{server.id\} variable references the server-selector element (within the parameters element). The filter resolves to the ESP server selected by the user when the job template is deployed. For more information, see “Example Job Template: Stock Trade” on page 52.

The server-filters element can also be used together with the server-filter-selector element (within the parameters element). In addition, a server-filter element can be referenced from a group element (within the instructions element). For more information, including additional example code, see “server-filter-selector” on page 80.
instructions

The instructions element is at the core of the job template. It contains the commands that SAS Event Stream Manager executes. The instructions often contain identifiers for selectors used with the parameters element, as well as string identifiers from the localization-strings element.

instructions

The instructions element is the parent element that contains specific instructions.

Here is an example that shows the outline of the instructions element. The details of the stop-project and unload-project instructions have been removed.

```xml
<instructions>
  <stop-project <!-- The details for the stop-project instruction are added here. --> />
  <unload-project <!-- The details for the unload-project instruction are added here. --> />
</instructions>
```

The instructions element can also contain a group element that groups instructions together.

Here is an example. The details of the elements have been removed.

```xml
<instructions>
  <group <!-- The attributes of the first group element are added here. --> >
    <load-project <!-- The details for the load-project instruction are added here. --> />
    <start-project <!-- The details for the start-project instruction are added here. --> />
    <start-connectors <!-- The details for the start-connectors instruction are added here. --> />
  </group>
  <group <!-- The attributes of the second group element are added here. --> >
    <load-project <!-- The details for the load-project instruction are added here. --> />
    <start-project <!-- The details for the start-project instruction are added here. --> />
</instructions>
```
Table 8.12  Properties of the instructions XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server-filter</td>
<td>This attribute specifies a filter for ESP servers. The filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated.</td>
</tr>
<tr>
<td></td>
<td>- Required: Each instruction must have a server-filter attribute. This attribute can be included in the instruction itself or in the enclosing group element or instructions element.</td>
</tr>
<tr>
<td></td>
<td>- A server-filter attribute on a child element overrides a server-filter attribute on a parent element. For example, a server-filter attribute on a specific instruction overrides a server-filter attribute on the parent instructions element.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: The ID of a server-filter element. For more information, see “server-filters” on page 82.</td>
</tr>
</tbody>
</table>
on-failure

This attribute specifies how SAS Event Stream Manager handles an instruction that fails to execute. By default, if an instruction fails to execute, any instructions that depend on the failed instruction are canceled. You can use the on-failure attribute to specify different behavior.

- Required: No. This attribute can be included in a specific instruction or in the enclosing group element or instructions element.

An on-failure attribute on a child element overrides an on-failure attribute on a parent element. For example, an on-failure attribute on a specific instruction overrides an on-failure attribute on the parent instructions element.

- Valid values: continue. Setting the on-failure attribute to continue means that dependent instructions will be executed even if the original instruction fails. If you do not want this to happen, remove the on-failure attribute.

Child Elements

load-project

This instruction takes a specific project that has previously been published in SAS Event Stream Processing Studio and loads it to an active ESP server.

- Required: No.

- For more information, see “load-project” on page 90.

start-project

This instruction starts an already loaded SAS Event Stream Processing project.

- Required: No.

- For more information, see “start-project” on page 92.
**Common Attributes of Instructions and the group Element**

Instructions and the `group` element have some common attributes, as shown in the following table:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Required</th>
<th>For More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>start-connectors</td>
<td>This instruction starts any connectors associated with a published SAS Event Stream Processing project.</td>
<td>No</td>
<td>start-connectors on page 93.</td>
</tr>
<tr>
<td>modify-project</td>
<td>This instruction updates an existing running project, subject to the rules imposed by SAS Event Stream Processing.</td>
<td>No</td>
<td>modify-project on page 93.</td>
</tr>
<tr>
<td>stop-project</td>
<td>This instruction stops a published SAS Event Stream Processing project. Stopping a project does not remove the project, but it stops any new connections to it.</td>
<td>No</td>
<td>stop-project on page 95.</td>
</tr>
<tr>
<td>unload-project</td>
<td>This instruction unloads a published SAS Event Stream Processing project.</td>
<td>No</td>
<td>unload-project on page 95.</td>
</tr>
</tbody>
</table>

This element groups instructions together.

<table>
<thead>
<tr>
<th>Required</th>
<th>For More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>group on page 96.</td>
</tr>
</tbody>
</table>
### Table 8.13  Common Attributes of Instructions and the group Element

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>This attribute specifies the unique identifier of the instruction or the group element.</td>
</tr>
<tr>
<td></td>
<td>■ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>■ Valid values: This value must match the regular expression <code>[0-9A-z] [0-9A-Z_ -] +</code>.</td>
</tr>
<tr>
<td>localization-id</td>
<td>This attribute specifies the localization string ID.</td>
</tr>
<tr>
<td></td>
<td>■ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>■ Valid values: This value must exist in the default language information within the localization-strings element.</td>
</tr>
<tr>
<td>server-filter</td>
<td>This attribute specifies a filter for ESP servers. The filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated.</td>
</tr>
<tr>
<td></td>
<td>■ Required: Each instruction must have a server-filter attribute. This attribute can be included in the instruction itself or in the enclosing group element or instructions element.</td>
</tr>
<tr>
<td></td>
<td>■ Valid values: The ID of a server-filter element.</td>
</tr>
<tr>
<td></td>
<td>A server-filter attribute on a child element overrides a server-filter attribute on a parent element. For example, a server-filter attribute on a specific instruction overrides a server-filter attribute on the parent instructions element.</td>
</tr>
<tr>
<td></td>
<td>■ Valid values: The ID of a server-filter element.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “server-filters” on page 82.</td>
</tr>
</tbody>
</table>
on-failure

This attribute specifies how SAS Event Stream Manager handles an instruction that fails to execute. By default, if an instruction fails to execute, any instructions that depend on the failed instruction are canceled. You can use the on-failure attribute to specify different behavior.

- Required: No. This attribute can be included in a specific instruction or in the enclosing group element or instructions element.

An on-failure attribute on a child element overrides an on-failure attribute on a parent element. For example, an on-failure attribute on a specific instruction overrides an on-failure attribute on the parent instructions element.

- Valid values: continue. Setting the on-failure attribute to continue means that dependent instructions will be executed even if the original instruction fails. If you do not want this to happen, remove the on-failure attribute.

depends-on

This attribute lists the instructions that this instruction or group depends on.

- Required: No.

- Valid values: The values of id attributes of other instructions in the same job template.

The common attributes have been omitted from subsequent sections, which describe additional XML properties for each instruction and the group element.

load-project

This instruction takes a specific project that has previously been published in SAS Event Stream Processing Studio and loads it to an active ESP server.
In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 88), the load-project instruction has the XML attributes shown in the following table:

**Table 8.14 Properties of the load-project XML Element**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>project-id</strong></td>
<td>This attribute specifies the ID of the project to be loaded to the SAS Event Stream Processing factory server. The project ID is generated by SAS Event Stream Processing Studio when a project is published. The ID is available using the project-selector element, or the literal value can be obtained from the application when viewing the project metadata.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: A valid project ID stored internally by SAS Event Stream Manager.</td>
</tr>
<tr>
<td><strong>project-version</strong></td>
<td>This attribute specifies the major version of the project to be published to the SAS Event Stream Processing factory server.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: A valid project version, which is specified by using the {project.version} placeholder.</td>
</tr>
<tr>
<td></td>
<td>Project XML files created in SAS Event Stream Processing Studio 5.2 and later versions contain separate variables that correspond to a project’s major version number and a project’s minor version number. This attribute specifies a project’s major version number as follows: project-version=&quot;{project.version}&quot;. You cannot specify the minor version number. Instead, SAS Event Stream Processing always uses the latest minor version of the specified major version.</td>
</tr>
</tbody>
</table>
running-project-name

This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.

- **Required:** Yes.
- **Valid values:** This value must match the regular expression `[A-z0-9_] +`.

**start**

This attribute specifies whether the project starts automatically.

If set to `true`, the project automatically starts upon publication. Otherwise, a manual `start-project` instruction is required. The default value is `true`.

- **Required:** No.
- **Valid values:** `true` or `false`.

**start-project**

This instruction starts an already loaded SAS Event Stream Processing project.

In addition to properties common to all instructions (for more information, see "Common Attributes of Instructions and the group Element" on page 88), the `start-project` instruction has the XML attributes shown in the following table:

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>running-project-name</td>
<td>This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.</td>
</tr>
<tr>
<td></td>
<td><strong>Required:</strong> Yes.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong> This value must match the regular expression <code>[A-z0-9_] +</code>.</td>
</tr>
</tbody>
</table>
**start-connectors**

This instruction starts any connectors associated with a published SAS Event Stream Processing project.

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 88), the start-connectors instruction has the XML attributes shown in the following table:

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>running-project-name</td>
<td>This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.</td>
</tr>
<tr>
<td></td>
<td>▪ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>▪ Valid values: This value must match the regular expression [A-z0-9_] +.</td>
</tr>
</tbody>
</table>

**modify-project**

This instruction updates an existing running project, subject to the rules imposed by SAS Event Stream Processing.

The modify-project instruction uses the following SAS Event Stream Processing command to set the state of the project to modified:

```
PUT http://<server>:<port>/SASESP/projects/<projectId>/state?value=modified
```

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 88), the modify-project instruction has the XML attributes shown in the following table:
### Table 8.17 Properties of the modify-project XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
</table>
| project-id          | This attribute specifies the ID of the project to be loaded to the SAS Event Stream Processing factory server. The project ID is generated by SAS Event Stream Processing Studio when a project is published.  
  The ID is available using the `project-selector` element, or the literal value can be obtained from the application when viewing the project metadata.  
  - Required: Yes.  
  - Valid values: A valid project ID stored internally by SAS Event Stream Manager.                                                                 |
| project-version     | This attribute specifies the version of the project to be published to the SAS Event Stream Processing factory server.  
  - Required: Yes.  
  - Valid values: A valid project version, which is specified by using the `{project.version}` placeholder.  
  Project XML files created in SAS Event Stream Processing Studio 5.2 and later versions contain separate variables that correspond to a project’s major version number and a project’s minor version number.  
  This attribute specifies a project's major version number as follows:  
  `project-version="{project.version}"`.  
  You cannot specify the minor version number. Instead, SAS Event Stream Processing always uses the latest minor version of the specified major version. |
running-project-name

This attribute specifies the name of the SAS Event Stream Processing project that you want to use to publish the model.

- Required: Yes.
- Valid values: This value must match the regular expression [A-z0-9_]+.

---

stop-project

This instruction stops a published SAS Event Stream Processing project. Stopping a project does not remove the project, but it stops any new connections to it.

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 88), the stop-project instruction has the XML attributes shown in the following table:

**Table 8.18  Properties of the stop-project XML Element**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>running-project-name</td>
<td>This attribute specifies the name of the SAS Event Stream Processing project to stop.</td>
</tr>
<tr>
<td></td>
<td>- Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>- Valid values: This value must match the regular expression [A-z0-9_]+.</td>
</tr>
</tbody>
</table>

For an example of how to use the stop-project instruction, see “Example Job Template: Stop a Project” on page 62.

unload-project

This instruction unloads a published SAS Event Stream Processing project.

In addition to properties common to all instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 88), the unload-project instruction has the XML attributes shown in the following table:
Table 8.19  Properties of the unload-project XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>running-project-name</td>
<td>This attribute specifies the name of the SAS Event Stream Processing project to unload.</td>
</tr>
<tr>
<td></td>
<td>Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>Valid values: This value must match the regular expression [A-z0-9_] +.</td>
</tr>
</tbody>
</table>

For an example of how to use the `unload-project` instruction, see “Example Job Template: Stop a Project” on page 62.

If a project is reported as missing (🚫) in the Running Projects tile, you might not be able to unload it using the `unload-project` instruction. For more information, see “Stop a Running Job” on page 122.

**group**

The `group` element enables you to group instructions together. Grouping instructions can make it easier to manage common dependencies. Grouping instructions also enables you to execute a job on several ESP servers.

In addition to properties common to instructions (for more information, see “Common Attributes of Instructions and the group Element” on page 88), the `group` element has the XML properties shown in the following table:

Table 8.20  Properties of the group XML Element

<table>
<thead>
<tr>
<th>Child Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>instruction</td>
<td>Any type of instruction is permitted inside a group element.</td>
</tr>
</tbody>
</table>
Here is an example in which two projects are deployed to two different sets of ESP servers:

```xml
<instructions>
  <group id="startgroup1" localization-id="startgroup1"
       server-filter="filter1">
    <load-project id="load-project1"
      localization-id="load-project"
      project-id="{project1.id}"
      project-version="{project1.version}"
      running-project-name="{project1display}" start="false" />
    <start-project id="start-project1"
      localization-id="start-project"
      running-project-name="{project1}"
      depends-on="load-project1" />
    <start-connectors id="start-connectors1"
      localization-id="start-connectors"
      running-project-name="{project1}"
      depends-on="start-project1" />
  </group>
  <group id="startgroup2" localization-id="startgroup2"
       server-filter="filter2">
    <load-project id="load-project2"
      localization-id="load-project"
      project-id="{project2.id}"
      project-version="{project2.version}"
      running-project-name="{project2display}" start="false" />
    <start-project id="start-project2"
      localization-id="start-project"
      running-project-name="{project2}"
      depends-on="load-project2" />
    <start-connectors id="start-connectors2"
      localization-id="start-connectors"
      running-project-name="{project2}"
      depends-on="start-project2" />
  </group>
</instructions>
```

The first group element provides instructions for loading and starting a project and for starting connectors. When a user deploys the template, the user selects the project to deploy (project1) and enters a display name for the project (project1display). These strings must be referenced in the localization-strings element (these are not shown in the example code here). The project is deployed to a set of ESP servers
that match a filter (filter1). This filter must be referenced in the server-filters element (this is not shown in the example code here).

Similarly, the second group element enables the user to select a second project to deploy to another set of ESP servers.

failure-instructions

An element called failure-instructions is activated if any of the commands in the instructions element of the job template fail.

The failure-instructions element has the same child elements as the instructions element. That is, the failure-instructions element can contain any instructions as well as the group element. For more information, see “instructions” on page 85.

Table 8.21 Properties of the failure-instructions XML Element

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>localization-id</td>
<td>This attribute specifies the localization string ID.</td>
</tr>
<tr>
<td></td>
<td>□ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>□ Valid values: This value must exist in the default language information within the localization-strings element.</td>
</tr>
</tbody>
</table>
This attribute specifies a filter for ESP servers. The filter resolves to a set of ESP servers to which the project will be deployed or with which the project is associated.

- Required: Each instruction must have a `server-filter` attribute. This attribute can be included in the instruction itself or in the enclosing `group` element or `failure-instructions` element.

A `server-filter` attribute on a child element overrides a `server-filter` attribute on a parent element. For example, a `server-filter` attribute on a specific instruction overrides a `server-filter` attribute on the parent `failure-instructions` element.

- Valid values: The ID of a `server-filter` element. For more information, see “server-filters” on page 82.
This attribute specifies how SAS Event Stream Manager handles a failure instruction that fails to execute. By default, if an instruction fails to execute, any instructions that depend on the failed instruction are canceled. You can use the on-failure attribute to specify different behavior.

- **Required:** No. This attribute can be included in a specific instruction or in the enclosing group element or failure-instructions element.

  An on-failure attribute on a child element overrides an on-failure attribute on a parent element. For example, an on-failure attribute on a specific instruction overrides an on-failure attribute on the parent failure-instructions element.

- **Valid values:** continue. Setting the on-failure attribute to continue means that dependent instructions will be executed even if the original instruction fails. If you do not want this to happen, remove the on-failure attribute.

### Child Elements

The failure-instructions element has the same child elements as the instructions element. That is, the failure-instructions element can contain any instructions as well as the group element. For more information, see “instructions” on page 85.

### Here is an example of a failure-instructions element:

```xml
<failure-instructions localization-id="recovery-steps">
  <unload-project id="unload" localization-id="unload"
      server-filter="filter1" project="example_proj" />
</failure-instructions>
```
You can use the initialization element of the job template to alter SAS Event Stream Processing projects as the projects are deployed by using the project-transformation instruction. The transform looks for a placeholder variable in the project and replaces it with the contents of the XML element. Here is an example:

```xml
<initialization>
  <project>
    <project-transformation id="transformed-project"
      project-id="{project.id}"
      project-version="{project.version}"/>
    <placeholder id="lang">{language}</placeholder>
  </project-transformation>
</projects>
</initialization>
```

Variables used for `{project}` are resolved at execution when the user selects a project. This is defined in the parameters element of the job template. The placeholder element within the initialization element has an attribute id that is set to the value of the placeholder in the project. In the example here, the project for deployment uses the SAS Event Stream Processing Twitter adapter. The placeholder allows the user to set the language to filter on when deploying:

```xml
<expression>tw_Lang == '{lang}'</expression>
```

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></td>
<td>This attribute specifies the unique identifier of this instruction.</td>
</tr>
<tr>
<td></td>
<td>▶ Required: Yes.</td>
</tr>
<tr>
<td></td>
<td>▶ Valid values: This value must match the regular expression <code>[0-9A-z] [0-9A-z_]</code> +.</td>
</tr>
</tbody>
</table>
**project-id**

This attribute specifies the ID of the project to be transformed. The ID is available using the `project-selector` element (for more information, see “project-selector” on page 74), or the literal value can be obtained from the application when viewing the project metadata.

- Required: Yes.
- Valid values: A valid project ID stored internally by SAS Event Stream Manager.

**project-version**

This attribute specifies the version of the project to be transformed.

- Required: Yes.
- Valid values: A valid project version.

Project XML files created in SAS Event Stream Processing Studio 5.2 and later versions contain separate variables that correspond to a project’s major version number and a project’s minor version number. This attribute specifies a project’s major version number as follows:

```
project-version="{project.version}"
```

You cannot specify the minor version number. Instead, SAS Event Stream Processing always uses the latest minor version of the specified major version.

**Child Elements**

**placeholder**

One or more `placeholder` elements can be defined to replace placeholders in the project.

---

**Table 8.23  Properties of the placeholder XML Element**

**Attributes**
| id | This attribute specifies the placeholder as it appears in the project.  
|    | - Required: Yes.  
|    | - Valid values: This value must match the value of the placeholder in the project. |

| Content |  
| placeholder | This content specifies the value that the placeholder resolves to. The content can be a job template variable enclosed in braces: `{ and }`. |
## Monitor Deployments

You can use SAS Event Stream Manager to review details of the active deployments in a SAS Event Stream Processing environment.

### Monitor Deployments

- Monitor All Deployments: 106
- Monitor a Specific Deployment: 108

### View Sample Data

- 111

### Update SAS Micro Analytic Service Modules

- 114

### Monitor Unassigned ESP Servers

- 116

### Monitor Jobs

- View All Jobs: 118
- View Job Details: 120

### Delete a Job

- 121

### Rerun a Job

- 121

### Stop a Running Job

- 122

### Load and Start a Project

- 123

### Stop and Unload a Project

- 124
Monitor All Deployments

The **Deployments** page displays any active deployments in your SAS Event Stream Processing environment.

**Figure 9.1  Deployments Page with Three Deployments**

The **Deployments** page displays the following information about the deployment:

- The number of ESP servers associated with the deployment.
- The status of the ESP servers associated with the deployment.
- The deployment’s name. If the deployment has a description, it is displayed as a tooltip when you place the cursor over the deployment name.
- Whether the deployment is a production deployment. For more information, see “Production Assets” on page 12.
The tags attributed to the deployment. Tags can be used to group and filter deployments. You can attribute tags to a deployment by editing the deployment. For more information, see “Edit a Deployment” on page 16.

The date on which the deployment was last updated.

The user ID or user name of the account that last updated the deployment.

The Status column provides a summary of the condition of the ESP servers associated with the deployment. The condition of an ESP server is determined by comparing the state of its projects with their expected state. This information helps you focus on those deployments that have problems. The number of ESP servers associated with the deployment is shown in the center of the icon. The following icons can appear in the Status column:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Green Circle" /> 2</td>
<td>Available (good) — The ESP servers associated with this deployment are available and do not have any projects in a warning or error state.</td>
</tr>
</tbody>
</table>
| ![Red Circle](image2.png) 2 | Available (project error) — The ESP servers associated with this deployment are available, but at least one ESP server has one or more projects that are in an error state.
For more information about the status of ESP servers, see “Monitor a Specific Deployment” on page 108. |
| ![Green Circle](image1.png) 2 | Available (unmanaged) — The ESP servers associated with this deployment are available and do not have any projects that are in an error state. However, at least one ESP server has one or more unmanaged projects (that is, projects that were not initiated from within SAS Event Stream Manager). |
| ![Gray Circle](image3.png) 2 | Unavailable — The ESP servers associated with this deployment are not available. |
| ![Black Circle](image4.png) 0 | This deployment (for example, a newly created deployment) does not contain any associated ESP servers. |
The icons shown in the preceding table assume that the deployment has multiple ESP servers that are all in the same state. If a deployment has more than one ESP server that are in different states, the icon in the Status column is displayed in multiple colors. These colors indicate the status of each of the ESP servers associated with the deployment. For example, the following icon shows a deployment that has five ESP servers — three are in the state Available (good), one is Available (unmanaged), and one is Unavailable:

You can arrange information in the table in several ways. For more information, see “Arrange Information in Tables” on page 10.

Clicking a deployment populates the ESP Server Status and Running Projects tiles:

- The ESP Server Status tile shows the number of ESP servers in each state.
- The Running Projects tile shows the number of running instances for each project in the deployment and the state of the instances. The project names and versions displayed in this tile correspond to the project names and versions displayed on the Projects page. The name of a running instance can be different from the project name, but in this tile such running instances are grouped under the project name.

Monitor a Specific Deployment

To view more details for a specific deployment, select the deployment on the Deployments page and click . A separate page appears, displaying information about the ESP servers assigned to the deployment and the projects running on those ESP servers. Clicking an ESP server in the main table populates the bottom pane with tabs that contain further information about that ESP server.
The main table displays the following information for each server or device defined as an ESP server:

- The ESP server’s status.
- The ESP server’s name.
- The tags attributed to the ESP server. Tags can be used to group and filter ESP servers. You can attribute tags to an ESP server by editing the ESP server. For more information, see “Edit an ESP Server” on page 23.
- The host on which the ESP server is running.
- The port for HTTP administration requests and for the HTTP publish/subscribe server.
- The SAS Event Stream Processing version installed on the host on which the ESP server is running.
- Whether a SAS Event Stream Processing Analytics license is installed on the host. To deploy a project that contains SAS Event Stream Processing Analytics windows, an appropriate license must be available.
Whether SAS Event Stream Processing has been enabled to meter the number of events that are processed on the ESP server.

A count of projects with different statuses. An example is (4) 2 Running, 1 Loaded, 1 Stopped.

The Status column provides a summary of the ESP server's condition. The condition of an ESP server is determined by comparing the state of its projects with their expected state. This information helps you focus on those ESP servers that have problems. The following icons can appear in the Status column:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🟢</td>
<td>Available (good) — The ESP server is available and does not have any projects in a warning or error state.</td>
</tr>
</tbody>
</table>
| 🔴 | Available (project error) — The ESP server is available, but at least one project is in an error state.  
For more information about the status of ESP servers, see "Monitor Unassigned ESP Servers" on page 116. |
| 🟠 | Available (unmanaged) — The ESP server is available and does not have any projects that are in an error state. However, it is running at least one unmanaged project (that is, a project that was not initiated from within SAS Event Stream Manager). |
| 🔴 | Unavailable — The ESP server is not available. |

You can arrange information in the table in several ways. For more information, see “Arrange Information in Tables” on page 10.

Clicking an ESP server populates the bottom pane with tabs that contain information relating to that ESP server:

The Projects tab displays projects that are running on the ESP server.

The status icon provides information that is similar to the status icon in the main table on the page for a specific deployment: the status of a project is determined by comparing its state with its expected state. For example, if the status and the
expected status do not match, the icon is red and its tooltip shows the message **Unexpected status**.

- The **Server Properties** tab displays the ESP server’s basic properties. You can click **Edit properties** to change most of these properties.
- The **Server Configuration** tab displays more detailed information about the ESP server.
- The **Performance** tab provides information about memory use. It also provides information about CPU use for each project and window.

On this page you can also perform actions on projects associated with your deployment:

- Load and start projects. For more information, see “Load and Start a Project” on page 123.
- View sample data for running projects. For more information, see “View Sample Data” on page 111.
- Stop and unload projects. For more information, see “Stop and Unload a Project” on page 124.

---

**View Sample Data**

To view sample data for a running project:

1. On the **Deployments** page, select a deployment that has a running project and click.

   **TIP** You can use the **Running Projects** tile on the **Deployments** page to identify deployments that have running projects. For more information, see “Monitor Deployments” on page 105.

A page that displays information about the selected deployment appears.
2 In the main table, select the ESP server that contains the running project that you are interested in.

Tabs appear in the bottom pane.

3 Right-click a running project in the **Projects** tab in the bottom pane, and select **Open running project**.

A new page opens, where sample data is displayed using tables, with a tab for each window. The following figure shows an example:

![SAS Event Stream Manager - Running Project](image)

**Note:** SAS Event Stream Manager displays a real-time view of the data. If your data source contains a limited number of events (for example, the data source is a CSV file rather than a stream of data), the table will be empty after all the events have been displayed.

You can arrange information in the table in several ways. For more information, see “Arrange Information in Tables” on page 10.
You can use the **Show formatted fields** check box to choose whether data appears exactly as it was received from the ESP server or with additional formatting. Here are some examples of additional formatting that is applied when the check box is selected:

- Dates are shown as Coordinated Universal Time (UTC) in ISO 8601 format. An example is 2018-11-30T13:33:47.000Z. If you clear the check box, dates appear in UNIX Epoch time, as this is the format in which the data is received from the ESP server.

- A dot is used as a separator in certain types of numerical data, rather than another separator, such as a comma. If you clear the check box and your locale is set to a locale that uses another separator, that separator is displayed instead of a dot.

- Opcodes are displayed using their localized names if your locale is not set to an English-language locale. If you clear the check box, opcodes are always shown in English, as this is how the data is received from the ESP server.

You can use the buttons on the navigation toolbar below the table to move between pages of sample data, or to jump to a specific page of sample data. Here is an example:

![Sample Data Table](image)

The following table describes the buttons on the navigation toolbar:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>←</td>
<td>Displays the first page of sample data.</td>
</tr>
<tr>
<td>←</td>
<td>Displays the previous page of sample data.</td>
</tr>
<tr>
<td>→</td>
<td>Displays the next page of sample data.</td>
</tr>
<tr>
<td>Icon</td>
<td>Action</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
</tr>
<tr>
<td><img src="image" alt="" /></td>
<td>Displays the last page of sample data.</td>
</tr>
</tbody>
</table>

## Update SAS Micro Analytic Service Modules

SAS Event Stream Processing projects can reference models that are stored in the SAS Model Manager common model repository. When a project is deployed, the model is retrieved from the SAS Model Manager common model repository and written to the ESP server. SAS Micro Analytic Service modules are used to accommodate such models. These models can be subsequently updated if a new champion model is declared. If your running project in SAS Event Stream Manager references a model stored in the SAS Model Manager common model repository, you can update the running project to reflect updates to the model.

If an update is available, a notification bar is displayed on the **Deployments** page. The following figure shows an example. You can update all projects that are running the specified version of the model, or you can select the projects that you want to update.
To hide the notification for a specific update temporarily, click **Ignore**. The notification for that update is hidden until you return to the **Deployments** page.

To accept the update for one or more projects that are running the specified version of the model:

1. Click **Deploy** on the notification bar.
   
   The Update SAS Micro Analytic Service modules window appears.

2. Click **All projects running this version** or **Select projects**, as required.

3. If you clicked **Select projects**, a drop-down list is displayed. Use this drop-down list to select the projects that you want to update. The drop-down list displays all the projects that reference the updated model. For each project, the ESP server that the project is currently deployed to is shown in brackets.
4 Click **OK**.

SAS Event Stream Manager runs a job to fetch and deploy the new champion model for the selected projects. To view the status of this job, open the **Log** page.

SAS Event Stream Manager updates the SAS Micro Analytic Service store with the model content and then replaces the SAS Micro Analytic Service module that each selected project is using. The schemas for the existing windows in the project are not altered during this operation. The new champion model must contain function input and output variables that are compatible with the existing schema.

The minor version of the project is updated. For example, the project version is updated from 1.2 to 1.3. The new version is also available in SAS Event Stream Processing Studio.

---

**Monitor Unassigned ESP Servers**

You can use the **Unassigned Servers** page to view all ESP servers that SAS Event Stream Manager is aware of, and which are available to be assigned to a deployment. ESP servers that already belong to a deployment are not displayed. The following figure shows an example:
The **Unassigned Servers** page does not indicate whether SAS Event Stream Manager is aware of a particular ESP server, for one of these reasons:

- SAS Event Stream Manager received information about the specific ESP server from SAS Event Stream Processing.
- You made a direct connection to the specific ESP server.

For more information, see “Connecting Directly to an ESP Server” on page 21.

The information about the status and properties of ESP servers is similar to the information about the page for a specific deployment. For more information, see “Monitor a Specific Deployment” on page 108.
Monitor Jobs

View All Jobs

The Log page displays the status of active and historical jobs, except for any jobs that you have deleted from the page. You can use this information to help resolve problems with your deployments. The following information is displayed for each job:

- the name of the job template from which the job was created
- the name of the deployment that the job template belongs to
- the user who deployed the job template
- the time at which the job template was deployed
- the time at which the job was completed
- the time it took for all the tasks in this job to be completed (whether they were completed successfully or not)
- the status of the job
- the job’s progress (specified as a percentage)
To refresh the table, click .

If the table contains too much information, you can arrange the information in several ways to find the jobs that you are interested in. For more information, see “Arrange Information in Tables” on page 10. In addition, you can use the drop-down menu that has the text Jobs started today to select a time period. You can also delete jobs from the table. For more information, see “Delete a Job” on page 121.

To view more information for a specific job, select the job in the table at the top of the page. The bottom pane displays the following information:

- The Progress tab shows how many tasks were completed and whether tasks were completed on all relevant ESP servers.
- The Failed Tasks tab lists tasks that were not completed.
- The Parameters tab lists parameters in the job template along with the value entered by the user for each parameter.
- The Details tab shows the job template description and other basic information.
View Job Details

To view further job details, select a job in the main table on the Log page and click . The job appears on a separate page. The following figure shows an example:

**Figure 9.5  Open Job**

The top of the page shows a summary and contains information about how many tasks were completed and whether tasks were completed on all relevant ESP servers.

The table on this page shows all tasks for the job, in the order in which the instructions are executed. You can expand items in the Task column to view the required level of detail. In the example, the Load Project, Start Project, and Start Pub/Sub Connectors rows relate to instructions in this specific example. That is, the names of the rows are derived from the job template. The table also shows that each task was completed on the same ESP server fraud.esp5.1.

You can also use the following check boxes to filter the contents of the table: Running, Not started, Completed, and Completed with failures.
Delete a Job

Deleting a job removes it from the main table on the Log page. Deleting jobs can be useful if the table contains jobs that are no longer relevant.

Deleting a job affects only the information that appears in the table; it does not affect job execution. Nevertheless, you cannot delete a job that is still running.

1. Do one of the following:
   - To delete a specific job from the Log page, select the job and click .
   - To delete several jobs, click and select Delete all canceled jobs, Delete all failed jobs, or Delete all successful jobs.

   The Delete Job window appears.

2. Click OK.

Rerun a Job

If a job fails to complete successfully, you can run it again. For example, a job might not complete successfully because an ESP server was unavailable or a data file was not present. After you have resolved the problem, you can rerun the job quickly from the page for that specific job rather than deploying the job template again. You can also rerun jobs that completed successfully.

Rerunning a job creates a new job, as opposed to starting the original job again.

To rerun a job:

1. Open the job if it is not already open: on the Log page, select the job and click .
2 Click 🔄.

   The Rerun Job window appears.

3 In the **Strategy** field, select your preferred option:
   - **Rerun all instructions**
   - **Rerun all instructions on servers that failed to complete successfully**
   - **Save failed servers to a filter**

   Select this option to create a filter that references the failed ESP servers. If you select this option, you must enter a name in the **Filter name** field. You can then reference this filter in another job template that addresses errors or cleans up ESP servers. For example, consider a situation where a project was loaded but a connector was not started because files were not available. A job template that addresses this error might include the `start-connectors` instruction. Consider another situation where a project could not be loaded because it had already been loaded. A cleanup job template might contain the `unload-project` and `stop-project` instructions.

   If the original job template needs to be run instead, use the **Rerun all instructions on servers that failed to complete successfully**.

   For more information about how you can use filters, see “Filters” on page 35.

4 If required, select **Skip failed instructions**.

5 Click **OK**.

SAS Event Stream Manager reruns the job. The result appears on a new page and also on a new row in the table on the **Log** page.

---

**Stop a Running Job**

Stopping a running job can be helpful if, for example, you want to stop a job that is taking a long time to complete. Stopping a running job means canceling the execution of
instructions that have not yet been executed. That is, when you stop a running job, the instruction that is currently being processed will be executed, but any subsequent instructions will not be executed.

Stopping a running job does not stop a running project. For more information, see “Stop and Unload a Project” on page 124.

To stop a running job:

1. Select the job in the main table of the Log page and click .
   
   The Cancel Job window appears.

2. Click OK.

---

### Load and Start a Project

You can load and start projects by including the load-project and start-project instructions in a job template. For more information, see “instructions” on page 85. You can also load and start projects by using the user interface controls on the page for a specific deployment. Loading and starting projects using these user interface controls is appropriate in many situations. However, for complex situations, using instructions in a job template might be more useful. For example, in the Stock Trade example, where you collect user input about selected stock codes, using instructions in a job template is appropriate. For more information, see “Overview of the Stock Trade Example” on page 133.

To load or start a project with user interface controls:

1. Open the deployment if it is not already open: on the Deployments page, select the deployment that you want to open and click .

2. In the main table, select the ESP servers on which you want to load or start a project. To select multiple ESP servers, hold down the Ctrl key and click the ESP servers. You can also select all ESP servers by clicking .
3 Click and select **Load project**, **Start project**, or **Load and start project**, as required.

The Load Project window, the Start Loaded or Stopped Project window, or the Load and Start Project window appears.

4 Select the project, and if prompted for select the version, and click **OK**.

A new page appears, displaying details about the job. For more information, see “View Job Details” on page 120.

---

**Stop and Unload a Project**

You can stop and unload projects by including the `stop-project` and `unload-project` instructions in a job template. For an example of a job template like this, see “Example Job Template: Stop a Project” on page 62. For more information, see “instructions” on page 85.

You can also stop and unload projects by using the user interface controls on the page for a specific deployment.

To stop or unload a project with user interface controls:

1 Open the deployment if it is not already open: on the **Deployments** page, select the deployment that you want to open and click .

2 In the main table, select the ESP servers on which you want to load or start a project. To select multiple ESP servers, hold down the Ctrl key and click the ESP servers. You can also select all ESP servers by clicking .

3 Click and select **Stop project**, **Unload project**, or **Stop and unload project**, as required.

The Stop Running Project window, the Unload Project window, or the Stop and Unload Running Project window appears.
4 Select the project and click **OK**.

A new page appears, displaying details about the job. For more information, see “View Job Details” on page 120.

When a project has stopped, the **Projects** tab in the bottom pane of the **Deployments** page displays the project’s status as stopped. When a project has been unloaded, it no longer appears in the **Projects** tab.

If a project is reported as missing, the Status column in the **Projects** tab displays the value **Missing** and the status icon is red (●). In this case, you might not be able to unload the project using an instruction. A project is reported as missing if the ESP server that it is running on is restarted. A project might be reported as missing if it was started, but never stopped, by SAS Event Stream Manager, and now is no longer available to the ESP server. To clean up a missing project, right-click the ESP server that contains the affected project and select **Clean up missing projects**. The affected project no longer appears in the **Projects** tab.
SAS Event Stream Manager enables you to monitor your metering servers. This ensures that your production ESP servers are in compliance with the terms of your software license. You must run at least one metering server to track usage data (that is, event counts) on your production ESP servers.

You do not need to track events on development servers because they do not contribute to the event volume assigned to your software license.

Add a Metering Server

To monitor an ESP metering server, you must add it to the list of defined metering servers in SAS Event Stream Manager.
To add a metering server:

1. On the **Metering** page, click ![ meterserver icon].

   The Metering Server Properties window appears.

2. In the **Host** field, enter the host name of the server containing the metering server.

3. In the **Metering port** field, either accept the default value (31001) or replace it with the network port defined on the metering server.

4. Click **OK**.

   The metering server is displayed on the **Metering** page, along with any other metering servers that were added previously.

---

**Monitor a Metering Server**

**View Metering Server Details**

To view details about a metering server, select the required metering server on the **Metering** page.

**Note:** If the metering server that you want to monitor is not listed, you must add it to SAS Event Stream Manager. For more information, see “Add a Metering Server” on page 127.

The Events tile displays each license associated with the selected metering server. It also displays the total number of events recorded for that license for each month or year in the selected time period.

To change the time period, click the drop-down list and select the required time period. For example, select **Show yearly totals**, **Show last 12 months**, or **Show monthly breakdown for year**. The event information is shown for the selected time period.
You can click to refresh the results in the **Events** tile to reflect any further metered events that have taken place after you selected the metering server.

Expanding each license in the Events tile enables you to view the total number of events for each host that uses the license. For metering servers that run with SAS Event Stream Processing 6.1 or a later release, each host can be expanded to view events for individual servers.

The following figure shows an example:
Export Detailed Events Information

You can export detailed information about metering server events to a comma-separated values (CSV) file. The CSV file contains the following events information, with separate rows for each unique combination: license, host, ESP server, server type, event date, event time (for each hour), continuous query, project, and window. The total number of events for the specific hour is displayed in the final column.

To export a CSV file:

1. On the Events tile, click the drop-down list and select a time period.

2. (Optional) Select a row in the table to specify the license, host, or server that you want to export detailed event information for. If you do not select a row in the table, detailed event information is exported for all licenses, hosts, and servers on the metering server.

   Note: All rows within a group are included in the export file. For example, if you select a license, the export file will contain details of all hosts that use that license and all servers on each of those hosts.

3. Click .

   The Export Metering Data window appears. The Months field contains a filter token for each month in the selected time period. Here is an example of a filter token:

   ![3/2019](example)

   Note: To add a month to the export, click in the Months field and select a month from the drop-down list. To remove a month from the export, click in the corresponding filter token. If you remove all filter tokens, so that the Months field is empty, the CSV file will contain details of all events on the metering server regardless of date.

4. Edit the file name as required and click OK.

   The CSV file containing the exported data is downloaded to your computer.
Group Metering Servers

You can arrange the information in the table on the Metering page in several ways. However, you might find it particularly helpful to group the table by the Host column, so that all ESP metering servers on a particular host are grouped together.

For more information, see “Arrange Information in Tables” on page 10.
Overview of the Stock Trade Example

This chapter provides an end-to-end example of how SAS Event Stream Manager works. By following this example, you learn the following tasks:

- create a deployment
- associate an ESP server with the deployment
- upload a project and a job template
- deploy the job template
monitor the deployment

stop a running job

The example uses the five files listed below. Four of these files are provided for you by SAS (for more information, see “Prepare the Example Files for Use” on page 138), and one file is created when you deploy the job template.

- trades_connector_stocksymbol_project.xml is the project used in the example. For more information about the content of this project, see “The Stock Trade Project” on page 134.
- trades_connector_stocksymbol_job_template.xml is a job template that loads and starts a project. This job template is discussed in detail earlier in this guide. For more information about the content of this job template, see “Example Job Template: Stock Trade” on page 52.
- trades1M.csv contains stock trade data used as input events in this example.
- placeholder_filtered_trades.csv is an output file. When you deploy the job template, the project processes stock trades and filters out trades that match a specified stock code. The project then writes these trades into the placeholder_filtered_trades.csv file.
- stop_project_job_template.xml is a job template that you can deploy to stop the running project when you have finished exploring this example. This job template is discussed in detail earlier in this guide. For more information about the content of this job template, see “Example Job Template: Stop a Project” on page 62.

The Stock Trade Project

The trades_connector_stocksymbol_project.xml file contains three windows:

- The source_win window is a source window. This window is where trades data from the trades1M.csv file enters the model.
- The symcode_filter window is a filter window. It contains a filter expression that identifies events with a specified stock symbol code.
The `aggFromFilt` window is an aggregate window. This window places events into aggregate groups and calculates the weighted average price, maximum price, minimum price, and standard deviation. This window also writes the results in the `placeholder_filtered_trades.csv` file.

*Figure 11.1  Diagram of the Stock Trade Project*

Here is the project XML:

```xml
<project name="filteredtrades" pubsub="auto" threads="8">
  <description>
  This project finds trades that match a specified stock code.
  </description>
  <contqueries>
    <contquery name="cq">
      <windows>
        <window-source name="source_win" index="pi_EMPTY" insert-only="true" collapse-updates="true">
          <schema>
            <fields>
```


<field name="ID" type="int32" key="true"/>
<field name="symbol" type="string"/>
<field name="currency" type="int32"/>
<field name="udate" type="int64"/>
<field name="msecs" type="int32"/>
<field name="price" type="double"/>
<field name="quantity" type="int32"/>
<field name="venue" type="int32"/>
<field name="broker" type="int32"/>
<field name="buyer" type="int32"/>
<field name="seller" type="int32"/>
<field name="buysellflg" type="int32"/>
<field name="time" type="stamp"/>
</fields>
</schema>
<connectors>
<connector name="csv_data_provider" class="fs">
<properties>
<property name="type">
<! [CDATA [pub]]>
</property>
<property name="fstype">
<! [CDATA [csv]]>
</property>
<property name="fsname">
<! [CDATA [/data/input/trades1M.csv]]>
</property>
<property name="transactional">
<! [CDATA [true]]>
</property>
<property name="blocksize">
<! [CDATA [1]]>
</property>
<property name="dateformat">
<! [CDATA [%Y-%m-%d %H:%M:%S]]>
</property>
</properties>
</connector>
</connectors>
</window-source>
>window-filter name="symcode_filter">
<description>
This is a Filter window with a filter expression to get events that have a specified stock symbol code. The Filter window has the same schema as the input window. As a result, there is no schema defined for the Filter window.
<window-filter>
    <expression>symbol == '{{$symcode}}'</expression>
</window-filter>

<window-aggregate name="aggFromFilt" collapse-updates="true">
    <schema>
        <fields>
            <field name="symbol" type="string" key="true"/>
            <field name="awap" type="double"/>
            <field name="minPrice" type="double"/>
            <field name="maxPrice" type="double"/>
            <field name="std" type="double"/>
        </fields>
    </schema>
    <output>
        <field-expr><![CDATA[ESP_aWAve(quantity,price)]]></field-expr>
        <field-expr><![CDATA[ESP_aMin(price)]]></field-expr>
        <field-expr><![CDATA[ESP_ESP_aMax(price)]]></field-expr>
        <field-expr><![CDATA[ESP_aStd(price)]]></field-expr>
    </output>
    <connectors>
        <connector name="Output_CSV" class="fs">
            <properties>
                <property name="type"><![CDATA[sub]]></property>
                <property name="fstype"><![CDATA[csv]]></property>
                <property name="fsname"><![CDATA[/data/placeholder_filtered_trades.csv]]></property>
                <property name="snapshot"><![CDATA[true]]></property>
            </properties>
        </connector>
    </connectors>
</window-aggregate>
</windows>
Stock Trade Example Steps

Prepare the Example Files for Use

1. Download the SAS Event Stream Manager examples package from SAS Event Stream Manager examples.

2. Save the following files, contained in the examples ZIP file, to a temporary location on your computer:
   - trades_connector_stocksymbol_project.xml
   - trades_connector_stocksymbol_job_template.xml
   - stop_project_job_template.xml

3. Locate the trades1M.csv file, which is provided within your SAS Event Stream Processing installation. This file is typically in the following location: `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/vwap_xml`. Replace `<release>` with the release number in your SAS Event Stream Processing installation directory path.

4. Update the trades_connector_stocksymbol_project.xml file to point to the location of the trades1M.csv file:
   a. Open the trades_connector_stocksymbol_project.xml file in a text editor.
   b. Locate the following line (for example, by searching):
      ```xml
      <![CDATA[/data/input/trades1M.csv]]>
      ```
c Update the directory path to point to the location of the trades1M.csv file on your SAS Event Stream Processing server.

d Locate the following line (for example, by searching):

```xml
<![CDATA[/data/placeholder_filtered_trades.csv]]>
```

e Update the directory path to point to a location on your SAS Event Stream Processing server where the output file can be written.

f Save the trades_connector_stocksymbol_project.xml file.

---

**Create the Stock Trade Deployment**

1 On the **Deployments** page, click ![Deployments Icon].

   The Deployment Properties window appears.

2 In the **Name** field, enter **Stock Trade**.

3 In the **Description** field, enter **A deployment for processing stock trades**.

4 In the **Tags** field, enter **stock**.

5 Click **OK**.

The **Stock Trade** deployment appears on a new page:
Add an ESP Server

You must associate an ESP server with the Stock Trade deployment.

**Note:** Ensure that you have an ESP server that you can use for this example. You can view available ESP servers that SAS Event Stream Manager has detected on the Unassigned Servers page. SAS Event Stream Manager detects some ESP servers automatically. You can also connect directly to an ESP server. In this case, SAS Event Stream Manager becomes aware of the ESP server only after you have connected. For more information, see “Connecting Directly to an ESP Server” on page 21.

To add an ESP server to the Stock Trade deployment:

1. Click ![Add Server](image) and select **Add an unassigned server**.
   - The Add and Remove ESP Servers window appears.

2. Move the desired ESP server from the **Available servers** table to the **Selected servers** table.

3. Click **OK**.
The ESP server appears in the table on the Stock Trade deployment page:

![SAS Event Stream Manager - Deployment](image)

<table>
<thead>
<tr>
<th>Status</th>
<th>Name</th>
<th>Tags</th>
<th>Host</th>
<th>HTTP Port</th>
<th>Version</th>
<th>Analytics</th>
<th>Metering</th>
<th>Project Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fraud.esp2_1</td>
<td></td>
<td></td>
<td>3002</td>
<td>6.1</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

**Upload and Publish the Stock Trade Project**

Use SAS Event Stream Processing Studio to upload and publish the stock trade project:

1. Upload the trades_connector_stocksymbol_project.xml file to SAS Event Stream Processing Studio. For more information, see “Upload a Project” in *SAS Event Stream Processing: Using SAS Event Stream Processing Studio*.

   The filteredtrades project appears on the Projects page in SAS Event Stream Processing Studio.

2. Publish the filteredtrades project. For more information, see “Publish a Version” in *SAS Event Stream Processing: Using SAS Event Stream Processing Studio*.

Afterward, the filteredtrades project appears on the Projects page in SAS Event Stream Manager:

![SAS Event Stream Manager](image)

<table>
<thead>
<tr>
<th>Name</th>
<th>Production</th>
<th>Tags</th>
<th>Latest Version</th>
<th>Published</th>
<th>Published By</th>
</tr>
</thead>
<tbody>
<tr>
<td>filteredtrades</td>
<td>No</td>
<td></td>
<td>1.0</td>
<td>22/03/2019</td>
<td>fsduser</td>
</tr>
</tbody>
</table>

**Note:** To explore the project’s contents, right-click the filteredtrades project and select Open project. The project opens and you can click the Diagram, XML, Details, Versions, and Files tabs to explore their contents.
Upload the Stock Trade Job Template

1. On the **Job Templates** page, click ![Upload job template](image) and select **Upload job template**.

   The Upload Job Template window appears.

2. Fill out the fields as follows:
   - **File**: Navigate to the trades_connector_stocksymbol_job_template.xml file.
   - **Tags**: Enter `stock`.
   - **Production template**: Leave this check box deselected.
   - **Version notes**: Enter `First version`.

3. Click **OK**.

The Filter Trades by Stock Code job template appears on the **Job Templates** page:

![Job Template 'Filter Trades by Stock Code' v.1](image)

**Note**: To explore the job template’s contents, right-click the Filter Trades by Stock Code job template and select **Open**.

Deploy the Stock Trade Job Template

1. On the **Job Templates** page, select the Filter Trades by Stock Code job template and click ![Deploy](image).

   The Job Template ‘Filter Trades by Stock Code’ v.1 window appears.
2 Fill out the fields as follows:

- **Deployment**: Select **Stock Trade**.
- **Project**: Select **filteredtrades**.
- **Latest available version**: This field is automatically set to **1.0**.
- **ESP server**: Select your ESP server.
- **Stock symbol code**: Select **Micro Focus**.

When the job template is deployed, trades for Micro Focus stock are identified with the **MCRO** stock code.

**Note**: The **Deployment** field always appears in this window. The other fields appear because they are specified in the job template parameters.

3 Click **OK**.

The **Filter Trades by Stock Code** page appears. When the **Load Project** task has completed successfully, its status is displayed as **✓**.

### Monitor the Deployment

1 On the **Deployments** page, select the **Stock Trade** deployment and click **✓**.

The **Stock Trade** page appears.
2 In the main table, select your server.
   The MCRO_1 project appears in the bottom pane.

3 Right-click the MCRO_1 project and select Open running project.
   The MCRO_1 page appears.

4 Select each tab in turn to view the results for each window:
   - The source_win tab displays a snapshot of input events in real time.
     The source_win window has no index. When all the events in the CSV file have been displayed, the table will be empty.
   - The symcode_filter tab lists trades for the selected stock code, MCRO.
     The symcode_filter window has an index. When all the events in the CSV file have been displayed, the table still contains some events.
   - The aggFromFilt tab displays aggregated results.
     The aggFromFilt window has an index. When all the events in the CSV file have been displayed, the table still contains some events. However, to see all the results, open the placeholder_filtered_trades.csv file on your SAS Event Stream Processing server:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I</td>
<td>N</td>
<td>MCRO</td>
<td>26.97</td>
<td>26.97</td>
<td>26.97</td>
</tr>
<tr>
<td>2</td>
<td>UB</td>
<td>N</td>
<td>MCRO</td>
<td>26.97</td>
<td>26.97</td>
<td>26.97</td>
</tr>
<tr>
<td>3</td>
<td>D</td>
<td>N</td>
<td>MCRO</td>
<td>26.97</td>
<td>26.97</td>
<td>26.97</td>
</tr>
<tr>
<td>4</td>
<td>UB</td>
<td>N</td>
<td>MCRO</td>
<td>26.97</td>
<td>26.97</td>
<td>26.97</td>
</tr>
<tr>
<td>5</td>
<td>D</td>
<td>N</td>
<td>MCRO</td>
<td>26.97</td>
<td>26.97</td>
<td>26.97</td>
</tr>
<tr>
<td>6</td>
<td>UB</td>
<td>N</td>
<td>MCRO</td>
<td>26.96099</td>
<td>26.84</td>
<td>26.97</td>
</tr>
<tr>
<td>7</td>
<td>D</td>
<td>N</td>
<td>MCRO</td>
<td>26.97</td>
<td>26.97</td>
<td>26.97</td>
</tr>
<tr>
<td>8</td>
<td>UB</td>
<td>N</td>
<td>MCRO</td>
<td>26.95721</td>
<td>26.84</td>
<td>26.97</td>
</tr>
<tr>
<td>9</td>
<td>D</td>
<td>N</td>
<td>MCRO</td>
<td>26.96099</td>
<td>26.84</td>
<td>26.97</td>
</tr>
</tbody>
</table>

If you want to see results for the other two stocks, deploy the project again and select a different stock.
Stop the Stock Trade Job

When you have finished exploring this example, you can stop the running job by deploying another job template file, stop_project_job_template.xml. Deploying this job template stops the selected project and unloads it from the ESP server.

1. On the **Job Templates** page, click ![Upload job template](image) and select **Upload job template**.

   The Upload Job Template window appears.

2. In the **File** field, select the stop_project_job_template.xml file.

3. Click **OK**.

4. Select the **Remove a Running Project v.1** job template and click ![Select](image).

   The Job Template ‘Remove a Running Project’ v.1 window is displayed.

5. Fill out the fields as follows:

   - **Deployment**: Select **Stock Trade**.
   - **ESP server**: Select your ESP server.
   - **Project to unload**: Select **MCRO_1**.

   **Note**: If you selected a different stock code when exploring the example, select the code for that stock instead.

6. Click **OK**.

   The **Remove a Running Project** page appears.

   This job template has two tasks: **Stop Project** and **Unload Project**. When each task has completed successfully, the status for each job is displayed as ![Status](image).
Security

Overview

This chapter describes configuration changes that you can make to the SAS Event Stream Manager environment. These changes should be made by your system administrator.

(Optional) Enable Encryption for SAS Event Stream Manager

Enabling the Transport Layer Security (TLS) protocol for connections between SAS Event Stream Manager and the ESP server is optional.

By default, the CA file is named `ca.pem`.

To enable TLS:

1. Obtain the CA file for the system where SAS Event Stream Manager is installed and for the clients that access the user interface.
2 On the machines where users access SAS Event Stream Manager, import the client certificate to the certificates store of your preferred web browser.

3 On the machine where SAS Event Stream Manager is running, import the client certificate to the Java keystore:

   **Note:** Specify the command on a single line. Multiple lines are used here to improve readability.

   ```bash
   $JAVA_HOME/jre/bin/keytool -importcert -keystore keystore-location -file path-to-file -storepass password -noprompt -alias alias
   
   Here is an example that assumes that you use `ca.pem` as the CA file:

   ```bash
   $JAVA_HOME/jre/bin/keytool -importcert -keystore /opt/sas/viya/config/etc/SASSecurityCertificateFramework/cacerts/trustedcerts.jks -file ca.pem -storepass changeit -noprompt -alias myalias
   ```

4 Restart the SAS Event Stream Manager service. Run the appropriate command:

   For Red Hat Enterprise Linux 6.7:

   ```bash
   sudo service sas-viya-esm-service-default stop
   sudo service sas-viya-esm-service-default start
   ```

   For Red Hat Enterprise Linux 7.x or SUSE Linux:

   ```bash
   sudo systemctl stop sas-viya-esm-service-default
   sudo systemctl start sas-viya-esm-service-default
   ```
Recommended Reading

SAS Event Stream Processing documentation

For a complete list of SAS publications, go to sas.com/store/books. If you have questions about which titles you need, please contact a SAS Representative:

SAS Books
SAS Campus Drive
Cary, NC 27513-2414
Phone: 1-800-727-0025
Fax: 1-919-677-4444
Email: sasbook@sas.com
Web address: sas.com/store/books
**adapter**

a standalone program that publishes event streams into or subscribes event streams from an event stream processing engine.

**asset**

See *event stream asset*.

**connector**

See *ESP connector*.

**container**

a standardized package of applications and their dependencies that can be run reliably in various computing environments.

**continuous query**

a container that holds a set of windows that are connected by an edge element, and are usually represented by a directed graph.

**data event**

an event that streams input data to be processed by the receiving window.

**data flow (flow)**

a data processing model that can be executed on a SAS Event Stream Processing engine.

**deployment**

a logical grouping of assets as a single unit for lifecycle management and monitoring in SAS Event Stream Manager.
**derived window**
a window that processes events that have been fed through other windows and that performs computations or transformations on these incoming events.

**directed graph**
a set of nodes connected by edges, where the edges have a direction associated with them.

**edge element**
an XML element that defines a conduit for events between windows.

**edge role**
an assignment that corresponds to the event type that is coming into, or streaming of, one of the following window types: Score, Train, Calculate, Model Reader, or Model Supervisor.

**edge server**
a server that resides between a device that generates data and a computer system that analyzes that data.

**engine**
the top level container in an event stream processing model.

**ESP**
See event stream processing.

**ESP connector (connector)**
an inline process that publishes event streams to, or subscribes event streams from, an event stream processing engine.

**ESP server**
See event-stream processing server.

**event block**
a grouping of binary events with a unique ID for use in a continuous query.
event stream (stream)
a continuous flow of events.

event stream asset (asset)
an identifiable part of a deployment that can be modified and monitored by SAS Event Stream Manager. Examples include projects and ESP servers.

event stream processing (ESP)
a process that enables real-time decision making by continuously analyzing large volumes of data as it is received.

event-stream processing server (ESP server)
an engine-executable program that instantiates and executes projects.

events engine
the top level container in an event stream processing model.

events project
See project.

flow
See data flow.

job template
an XML file that contains instructions for creating a set of tasks to be executed by SAS Event Stream Manager.

metering server
a server that aggregates event counts based on license, source window, and hour of the day.

model event
an event that streams model metadata into a Score window or into a Model Supervisor window.
**offline model**

a model that are specified, developed, trained, and stored separately from the ESP server.

**online model**

a model that uses an algorithm that is packaged with SAS Event Stream Processing Analytics.

**opcode (operation code)**

an instruction that specifies an action to be performed.

**operation code**

See opcode.

**persist-and-restore operation**

a process that preserves a model's state on one computer system and restores it on another system.

**project (events project)**

a container that holds one or more continuous queries. See also events engine, continuous query.

**publish/subscribe API**

a library that enables you to publish event streams into an event stream processor, or to subscribe to event streams, within the event stream processing model. The publish/subscribe API also includes C, JAVA, and Python event stream processing object support libraries.

**request event**

an event that initiates an action, such as reconfiguring a model.

**Source window**

a window that has no windows feeding into it and is the entry point for publishing events into the continuous query.
stream
See event stream.

thread pool
a set of threads that can be used to execute tasks, post work items, process asynchronous I/O, wait on behalf of other threads, and process timers.

window
a processing node in an event stream processing model. Source and derived windows can perform aggregations, computations, pattern matching, and other operations.
Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore for additional books and resources.