SAS® Event Stream Processing 5.2: Using SAS® Event Stream Processing Studio

Overview to SAS Event Stream Processing Studio

Overview

SAS Event Stream Processing Studio is a web-based client that enables you to create, edit, upload, and test event stream processing models using SAS Event Stream Processing Studio Modeler. SAS Event Stream Processing Studio Modeler displays a model as a data flow diagram, enabling you to see and control how windows relate and flow into one another.

Requirements for Solution Access and Use

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

- a supported web browser has been installed

  Note: For more information about the browsers that are supported in SAS Event Stream Processing Studio, click ? in the application and then click **About**. Click **Supported Browsers and Platforms** to view the browsers that are supported. SAS Event Stream Processing Studio requires the use of cookies to maintain the session state. For more information about how to access SAS Event Stream Processing Studio, see "Accessing SAS Event Stream Processing Studio" on page 2.
- Your screen has a minimum resolution of 1,280 x 1,024
- JavaScript has been enabled in your browser

Starting the Event Stream Processing Server

Before opening or creating a model in SAS Event Stream Processing Studio, you must start the ESP server. To start the server on a UNIX system, run the following command:

```
$DFESP_HOME/bin/dfesp_xml_server -http port
```

On Windows systems, run the following command:

```
%DFESP_HOME%\bin\dfesp_xml_server -http port
```

Port specifies the port for the HTTP REST API. You can check the terminal log to confirm that you have instantiated the server successfully. In addition, the terminal log displays the port specified here. For information about the ESP server, see "Using the ESP Server" in SAS Event Stream Processing: Using the ESP Server.

In these examples, **`$DFESP_HOME`** is the installation directory on UNIX systems and **`%DFESP_HOME%`** is the installation directory on Windows systems.

Note: On Windows systems, if your installation directory contains a space, you must enclose this command in quotation marks, as shown here:

```
"%DFESP_HOME%\bin\dfesp_xml_server" -http 1234
```

Accessing SAS Event Stream Processing Studio

To access SAS Event Stream Processing Studio:

1. Open the following URL:

   `https://host/SASEventStreamProcessingStudio`

   The *host* assigned to the system where SAS Event Stream Processing Studio installs.

   The Sign In to SAS window appears.

   Note: The Sign In to SAS window appears only if your deployment has been configured to enable users to log on to SAS Event Stream Processing Studio. If your deployment has not been configured in this way, you are not required to enter a user ID and password to access SAS Event Stream Processing Studio.

2. Enter your user ID and password and click **Sign in**.

If you successfully access SAS Event Stream Processing Studio, the SAS Event Stream Processing Studio home page appears. If you are using SAS Event Stream Processing Studio for the first time, the initial SAS Event Stream Processing Studio window might not contain any models.

Understanding the User Interface

Pages

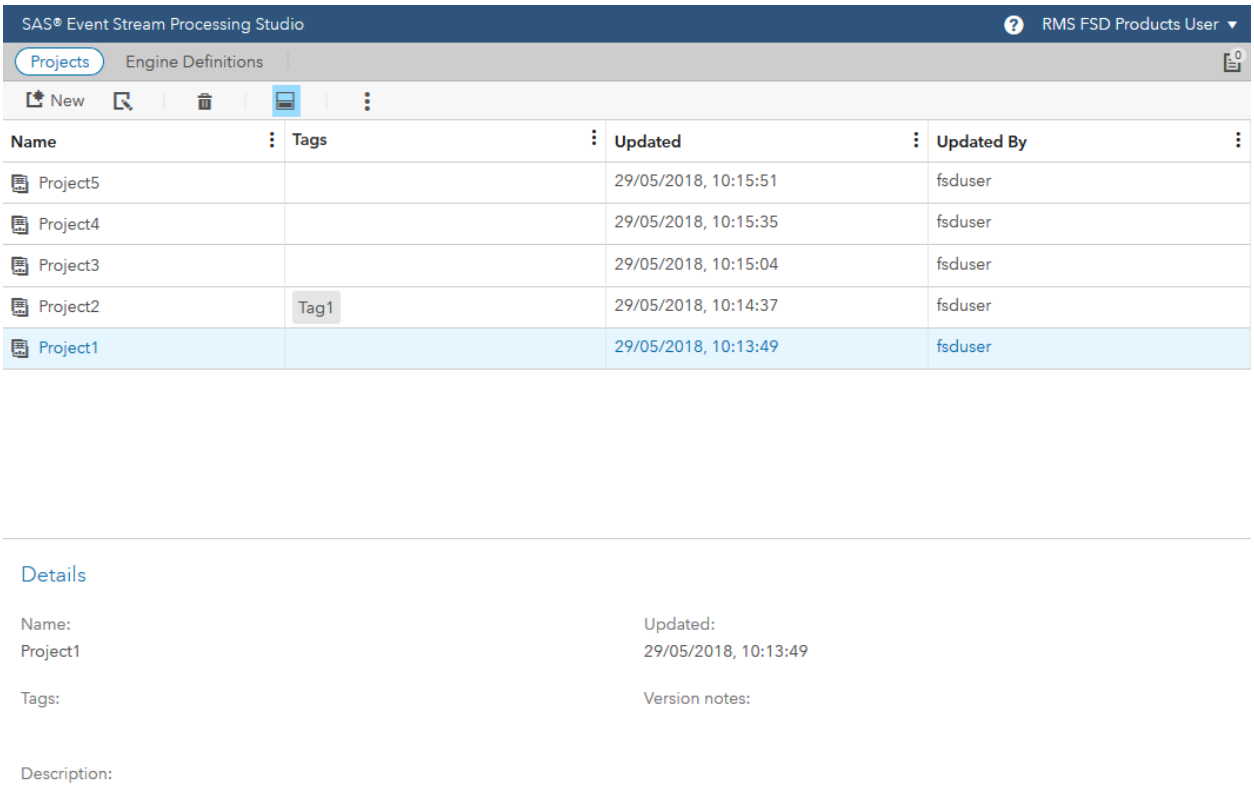
A *page* is the highest level container in the user interface. All other user interface elements are contained within the confines of a page.

SAS Event Stream Processing Studio contains the following main pages:

- the **Projects** page enables you to create, edit, upload, download, or delete the projects that contain your SAS Event Stream Processing models
- the **Engine Definitions** page enables you to create, edit, upload, download, or delete engine definitions

When you first access SAS Event Stream Processing Studio, the **Projects** page appears.

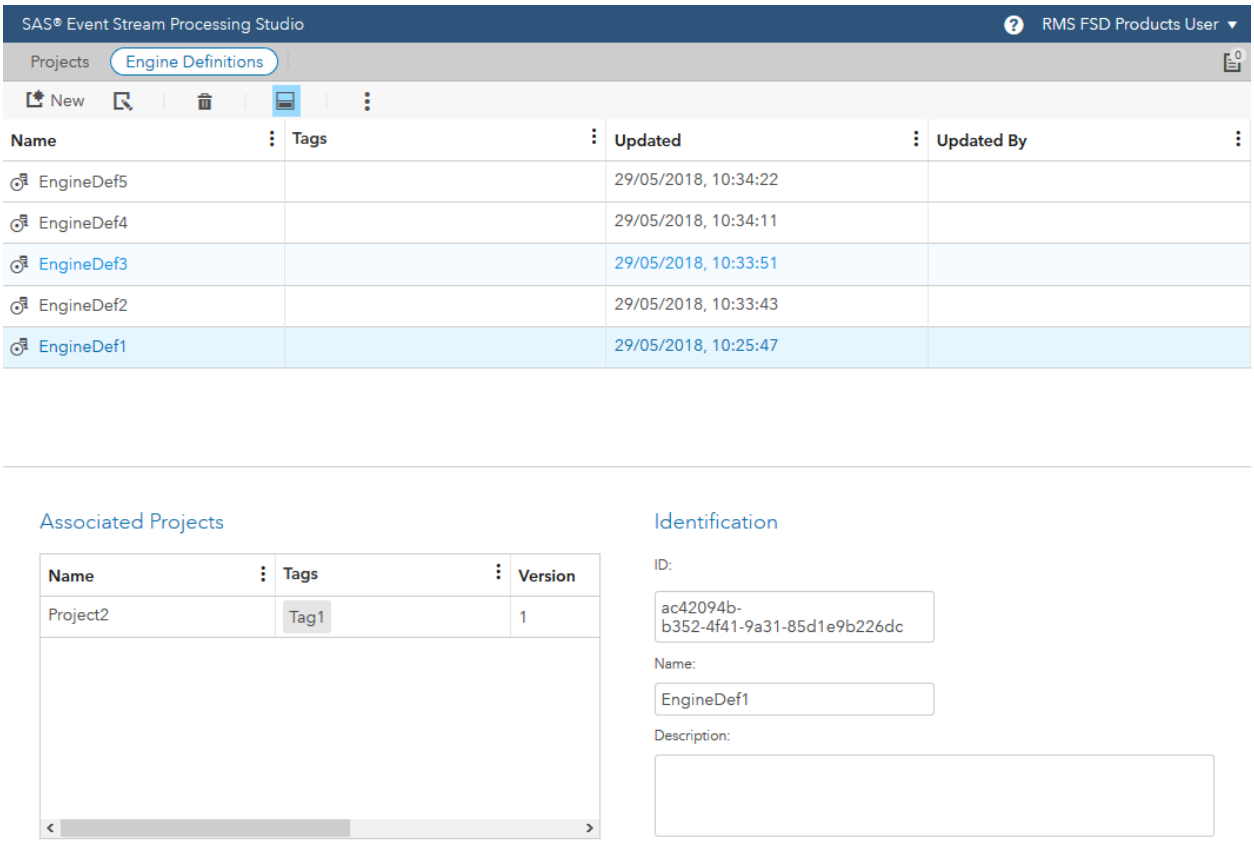
Figure 1 *The Projects Page Displaying Active Test Projects*



Panes

SAS Event Stream Processing Studio contains *panes* that enable you to view different types of information within the same page. The following figure displays the **Engine Definitions** page, which contains a bottom pane with two tiles: **Associated Projects** and **Identification**.

Figure 2 *Example of a Page with a Pane*



To resize a pane, drag a border in the appropriate direction. To resize a horizontal pane, drag a border up or down. To resize a vertical pane, drag the border left or right.

To hide a pane, click . To display it again, click .

Tiles

A *tile* is a self-contained block of information that resides within a pane or sometimes directly on a page.

Figure 3 *Example of a Tile*

Associated Projects

Name	Tags	Version	Updated
project155		1	22/05/2018, 11:14:17

Windows

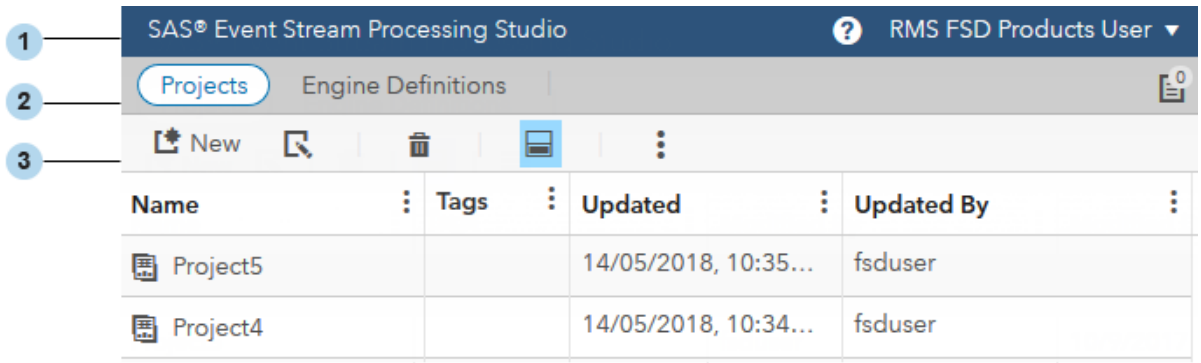
A *window* is a floating user interface element that often appears as a result of a user action. Windows generally provide a means by which to perform an action and can be closed to return you to the page from which the window was launched. The following figure shows a window that is used to upload an engine definition to SAS Event Stream Processing Studio.

Figure 4 *The Upload Engine Definition Window*

Note: The user interface element *window* in SAS Event Stream Processing Studio does not have the same meaning as a SAS Event Stream Processing window. In SAS Event Stream Processing, windows are components of a continuous query. A continuous query contains a source window and one or more derived windows. SAS Event Stream Processing windows are connected by edges, which have an associated direction. SAS Event Stream Processing Studio contains both user interface element windows and SAS Event Stream Processing windows.

Toolbars

Figure 5 *SAS Event Stream Processing Studio Toolbars*



There are three main toolbars in SAS Event Stream Processing Studio, as shown in the following table:

Item Number	Description	Name
1	Application bar	■ Displays your display name or user ID **Note:** If you have not set a display name, your user ID is displayed by default. If SAS Event Stream Processing Studio has been configured so that you do not need to log on with a user name and password, your user ID is not displayed. The screen shots in this section display a user ID. ■ Provides access to Help and product information ■ Enables you to sign out of SAS Event Stream Processing Studio (if applicable)
2	Menu bar	■ Provides access to the main SAS Event Stream Processing Studio pages: **Projects** and **Engine Definitions** ■ Provides access to each project, project version, model test, or engine definition that is currently open. The navigation overflow menu button displays the total number of these pages that are currently open, for example, .
3	Toolbar	■ Includes buttons or tabs associated with the open item ■ Enables you to perform actions associated with a project. For example, you can download a project to your computer by clicking ⋮ on the toolbar and selecting **Download**.

Sorting and Filtering Items

To make it easier to work with a large amount of information, you can sort and filter items displayed in tables. You can also show, hide, and reorder columns.

You can sort lists of items by ascending or descending order. To sort in ascending order, click the heading of the column that you want to sort. To sort in descending order, click the column again. To remove sorting, click the column a third time.

You can create filter criteria by which to display only a subset of information for a column. To create filter criteria, click ⋮ for the column that you want to apply filter criteria to, select **Filter**, and enter your filter criteria.

You can configure the columns that you want to display. To do this, click ⋮ 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.

SAS Event Stream Processing Studio Modeler

When you create a new project or open an existing project, a separate page that contains the project content appears. This project page displays SAS Event Stream Processing Studio Modeler, which enables you to design models in a visual way and to test them. SAS Event Stream Processing Studio Modeler also includes the XML Editor, which you can use as an alternative way to construct your model.

For more information about the modeler, see “Using SAS Event Stream Processing Studio Modeler” on page 17. For more information about the XML Editor, see “Using the XML Editor” on page 23.

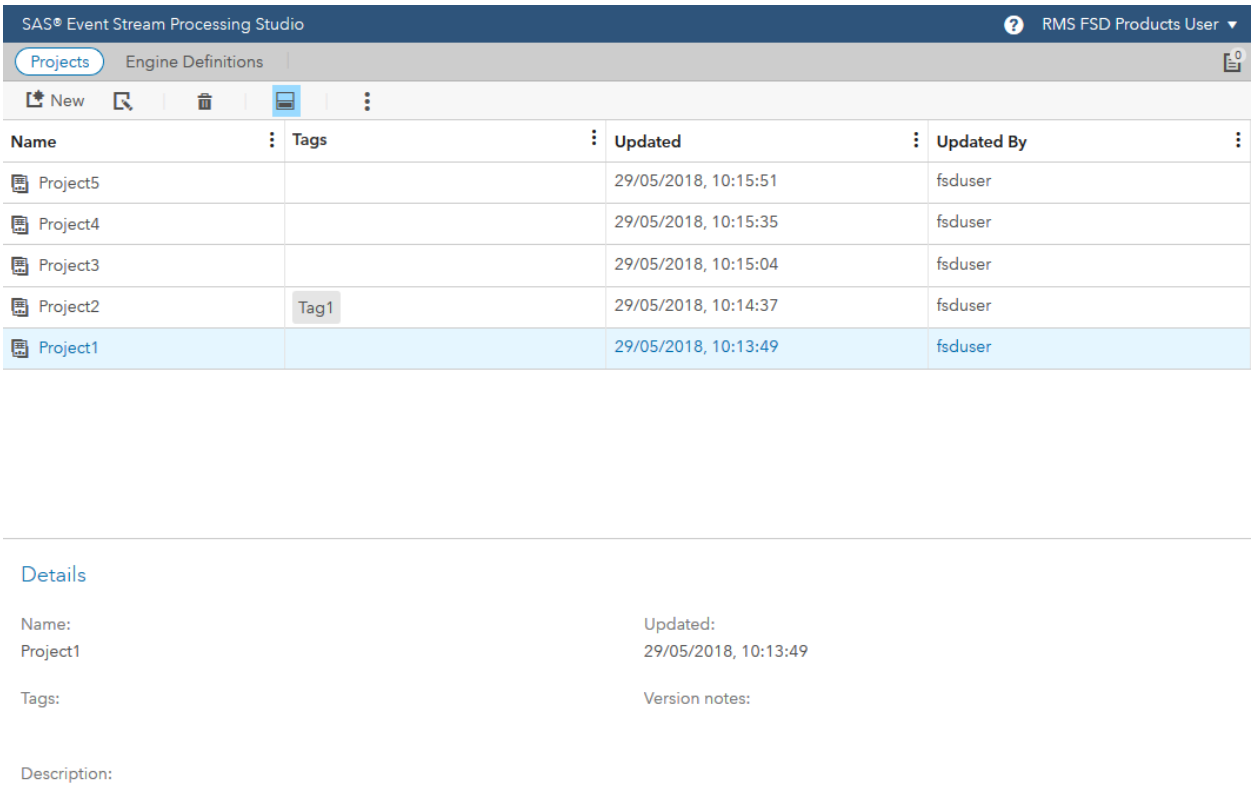
Working with Projects in SAS Event Stream Processing Studio

A *project* consists of one or more continuous queries. You can use SAS Event Stream Processing Studio to create, upload, download, and delete SAS Event Stream Processing projects. You can associate your project with an engine that is defined in SAS Event Stream Processing Studio. For more information, see "Creating Engine Definitions" on page 14.

Overview

The **Projects** page in SAS Event Stream Processing Studio enables you to view the projects in your deployment, along with their identification details and associated engines.

Figure 6 *The Projects Page*



Note: You can access the Manage ESP Servers window from the **Projects** page, enabling you to register additional ESP servers or to display the details of existing ESP servers. To do this, click ⋮ and select **Manage ESP servers** from the drop-down list. For more information, see "Managing ESP Servers in SAS Event Stream Processing Studio" on page 29.

Creating Projects

To create a new project:

1. On the **Projects** page, click New.

   The New Project window appears.

2. In the New Project window, do the following:

   a. In the **Name** field, enter a unique name for the project.

      Note: You must enter a unique project name. SAS Event Stream Processing Studio does not support duplicate project names.

      If a project has been published and then subsequently deleted from SAS Event Stream Processing Studio, you cannot reuse the deleted project's name.

   b. In the **Description** field, enter a description for the project.

   c. In the **Tags** field, enter any identifying keywords that describe the project.

   d. In the **Notes** field, enter any extra information relating to the project.

   e. Click **OK**.

      If you do not currently have any ESP servers configured, you are prompted to decide whether you want to configure an ESP server now.

   f. Click **Yes** to configure an ESP server now or click **No** to configure an ESP server later.

3. If you chose to configure an ESP server now, do the following:

   a. In the **Name** field, enter a name to identify the new ESP server that you want to create.

   b. In the **Host** field, enter the host name or IP address of the server that contains the new ESP server.

   c. In the **HTTP port** field, enter the new ESP server's administration port number.

   d. If required, change the setting for the **Authentication** field:

      - **None**: This is the default option.
      - **Authenticate using an OAuth token**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field appears where you must enter the OAuth token.
      - **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields appear where you must enter the user name and password.

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

   f. Click **OK**.

   Note: You can register additional ESP servers or you can view the details of a specific ESP server. To do this, in SAS Event Stream Processing Modeler, click **Manage ESP Servers** in the **ESP Server** drop-down list.

4. Click **OK**.

SAS Event Stream Processing Modeler appears.

Your project is created with a set of default properties. Before you start creating your model, configure your project's properties.

To configure your new project's properties:

a Review the default project properties in the right pane and modify them if necessary.

b You can also add or modify additional project properties, such as SAS Micro Analytic Service modules, user-defined properties, and connector orchestration.

5 Click .

Note: To create a copy of the project with a different filename, click and select **Save as**. Enter the relevant information into the Save As window and click **OK**.

Uploading Projects

Note: Project XML files uploaded to SAS Event Stream Processing Studio must be encoded in UTF-8 format. Uploading project XML files that are not encoded in UTF-8 format can display invalid characters in SAS Event Stream Processing Studio.

To upload existing projects:

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 to upload simultaneously. To do this, press and hold Ctrl, select the projects that you want to upload, and click **Open**. If your projects are located in different folders, click again, select the relevant project, and click **Open**.

4 Click **Upload**.

An icon appears indicating if the project successfully uploaded. Successfully uploaded projects are indicated by a icon. Projects that failed to upload are indicated by an icon.

5 Click **OK**.

The projects that you uploaded appear on the **Projects** page.

Note: You cannot upload a project to SAS Event Stream Processing Studio that has the same name as a project version that has previously been published. This also applies to a project that has the same name as a project version that has been subsequently deleted from SAS Event Stream Processing Studio.

Downloading Projects

To download a project, select the project that you want to download from the table on the **Projects** page, click , and select **Download**.

The project downloads to your computer.

Note: The location of the project that you downloaded might vary depending on your browser's configuration.

Deleting Projects

To delete a project, select the project that you want to delete from the table on the **Projects** page and click  . Click **Yes** to confirm the deletion.

The project is permanently deleted from SAS Event Stream Processing Studio.

Note: Only the working version of a project is deleted. Published versions can still be accessed by other applications through SAS Files and Folders Services.

Project Locks

Projects that are being edited in SAS Event Stream Processing Studio are automatically locked. This ensures that changes to a project cannot be unintentionally overwritten by another user. For more information, see "Project and Engine Definition Locking" on page 16.

Project Metadata

When you create a project, the following unique information that identifies a project is created automatically:

- The user ID of the user who created the project
- The user ID of the user who last modified the project
- The date on which the project was created
- The date on which the project was last modified

The project information is displayed within the `<metadata>` element in your project's XML code, but it is stored in the SAS Event Stream Processing Studio database.

Metadata is also created if you perform one of the following actions:

- Apply a tag to the project
- Make changes to your model in the workspace

  Note: When you make a change to your model in the workspace, a `<meta id="layout">` element is added. This element specifies the name of your model's continuous query, the names of the windows in your model, and each window's X and Y coordinates in the workspace.
- Import a model that contains a SAS Micro Analytic Service module that has been created in SAS Model Manager into SAS Event Stream Processing Studio. For more information, see "Example: Importing a Model Created in SAS Model Manager into SAS Event Stream Processing Studio" on page 77.

When you publish a version of a project, the following unique information that identifies the version is created automatically:

Note: The following elements are not displayed in the XML code of a working model. However, these elements are displayed in the model's XML code if you have published the model and you are viewing the model in Read-Only mode.

1. The project version's unique ID
2. The project version's major version number
3. The project version's minor version number

Here is an example of a published project version's metadata:

```
<metadata>
    <meta id="studioUploaded">2/26/2018</meta>
    <meta id="studioUploadedBy">espuser</meta>
    <meta id="studioModified">2/26/2018</meta>
    <meta id="studioModifiedBy">espuser</meta>
    <meta id="studioTags">Tag1</meta>
    <meta id="layout">{"cq1":{"Source1":{"x":-310,"y":-315}}}</meta>
    <meta id="studioVersionMajor">1>/meta>
    <meta id="studioVersionMinor">0</meta>
</metadata>
```

In this example, the project's version number is 1.0.

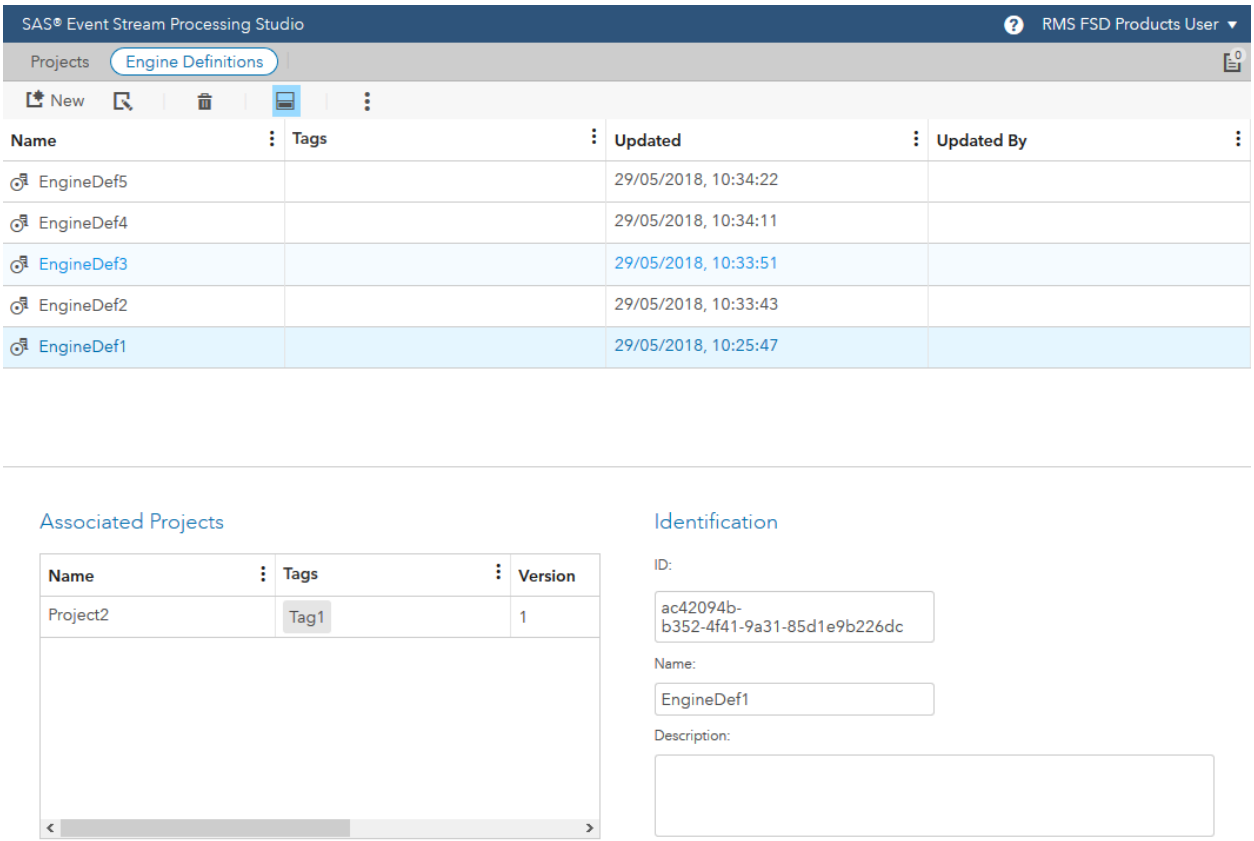
Working with Engine Definitions in SAS Event Stream Processing Studio

An *engine* is the top-level container in the SAS Event Stream Processing model hierarchy. Each model contains only one engine instance with a unique name. You can use SAS Event Stream Processing Studio to create, upload, download, and delete SAS Event Stream Processing engine definitions. You can associate each project that you produce or upload with an engine definition in SAS Event Stream Processing Studio.

Overview

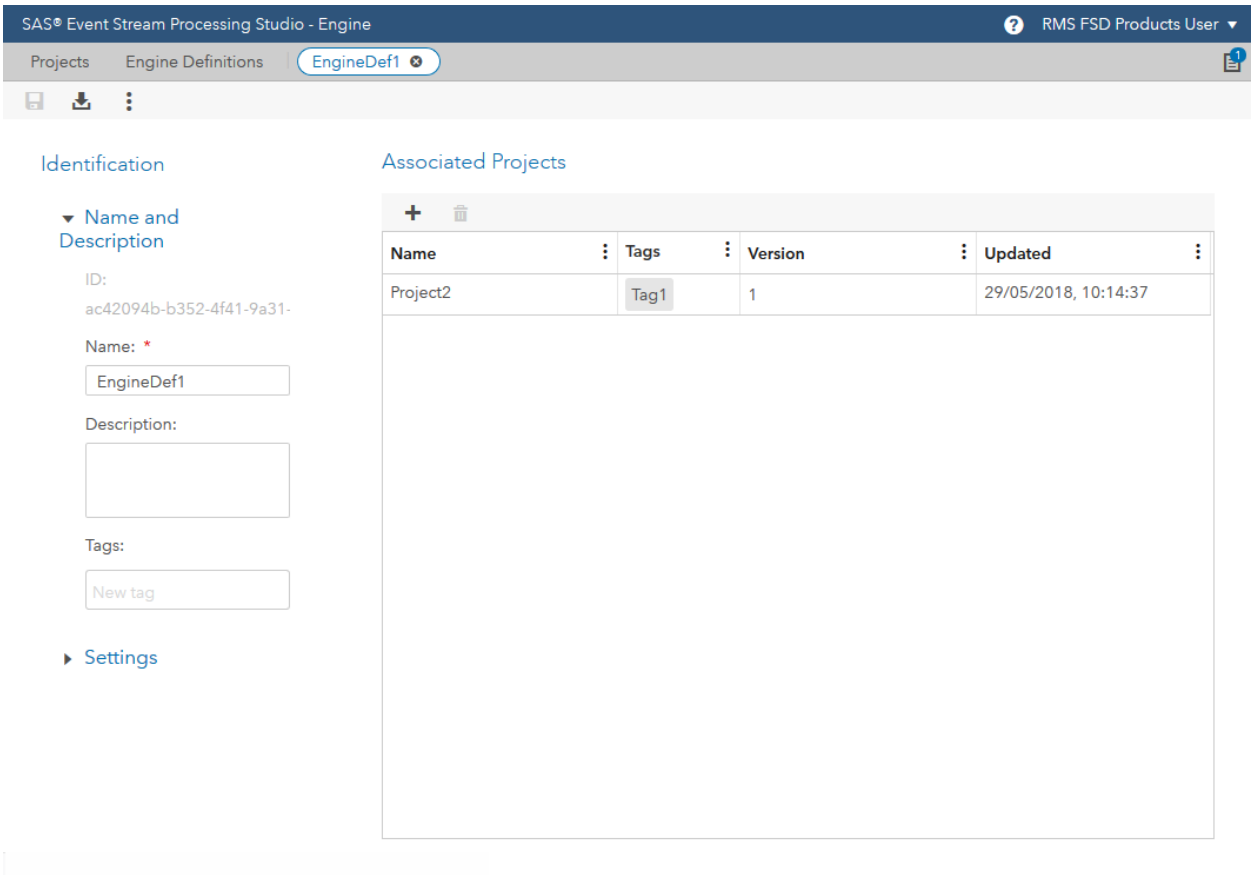
The **Engine Definitions** page in SAS Event Stream Processing Studio enables you to view all operational engines in your deployment, along with their identification details. You can view a list of the engine's associated projects in the **Associated Projects** tile.

Figure 7 *The Engine Definitions Page*



Double-clicking on an engine definition displays an engine definition page. This page consists of an **Identification** tile that contains identifying information about the engine definition. In addition, the page contains an **Associated Projects** tile that enables you to associate the engine definition with one or more projects. Associating an engine definition with one or more projects is useful when executing multiple projects as a single action by grouping the projects within an engine. This enables you to reuse projects without having to re-create individual projects within each new engine.

Figure 8 *An Engine Definition Page*



Creating Engine Definitions

To create a new engine definition:

1 On the **Engine Definitions** page, click New.

The New Engine Definition window appears.

2 In the **Engine definition name** field, enter a name for the engine definition that you are creating.

3 Click **OK**.

Your engine definition is created, and an **Engine Definition** page appears.

4 Review the information in the **Identification** tile:

- The **Name** field contains the engine definition's name
- In the **Description** field, enter a description for the engine definition that you are creating
- In the **Tags** field, enter any keywords that describe the engine definition that you are creating
- Update any other fields as required

5 Associate projects with your engine definition:

a In the **Associated Projects** pane, click +.

The Add Project window appears.

b In the Available projects table, select the project that you want your engine definition to be associated with and click .

The project that you selected appears in the Selected projects table.

c Click **Save**.

The newly associated project appears in the **Associated Projects** tile.

6 Click .

Uploading Engine Definitions

To upload an engine definition:

1 On the **Engine Definitions** page, click and select **Upload**.

The Upload Engine Definition window appears.

2 In the **File** field, click **Choose File**.

3 Navigate to the file that contains the engine definition that you want to upload and click **Open**.

4 In the **Engine definition name** field, if necessary, adjust the name of the engine definition that you are uploading.

5 In the **Description** field, enter a description for the engine definition that you are uploading.

6 In the **Tags** field, enter any keywords that describe the engine definition that you are uploading.

7 Click **OK**.

Downloading Engine Definitions

To download an engine definition, select the engine definition that you want to download from the table on the **Engine Definitions** page, click , and select **Download**.

Note: The location engine definition that you downloaded might vary depending on your browser's configuration.

Deleting Engine Definitions

To delete an engine definition, select the engine definition that you want to delete from the table on the **Engine Definitions** page and click . Click **Yes** to confirm the deletion.

The engine definition is deleted from SAS Event Stream Processing Studio.

Engine Definition Locking

Engine definitions that are being edited in SAS Event Stream Processing Studio are automatically locked. This ensures that changes to an engine definition cannot be unintentionally overwritten by another user. For more information, see "Project and Engine Definition Locking" on page 16.

Project and Engine Definition Locking

Projects and engine definitions that are being edited in SAS Event Stream Processing Studio are automatically locked. This ensures that changes to a project or to an engine definition cannot be unintentionally overwritten by another user. If you open a project or an engine definition, it is assumed that you intend to edit it. If a project or an engine definition has not been locked by another user, the project or the engine definition is locked immediately after you open it. If you are editing a project or an engine definition, the lock is released automatically if you close the tab that contains the project in the application. A lock is also released approximately two minutes after you close the application. For example, if you turn off your computer or close the browser, other users must wait two minutes until they can lock a project or lock an engine definition.

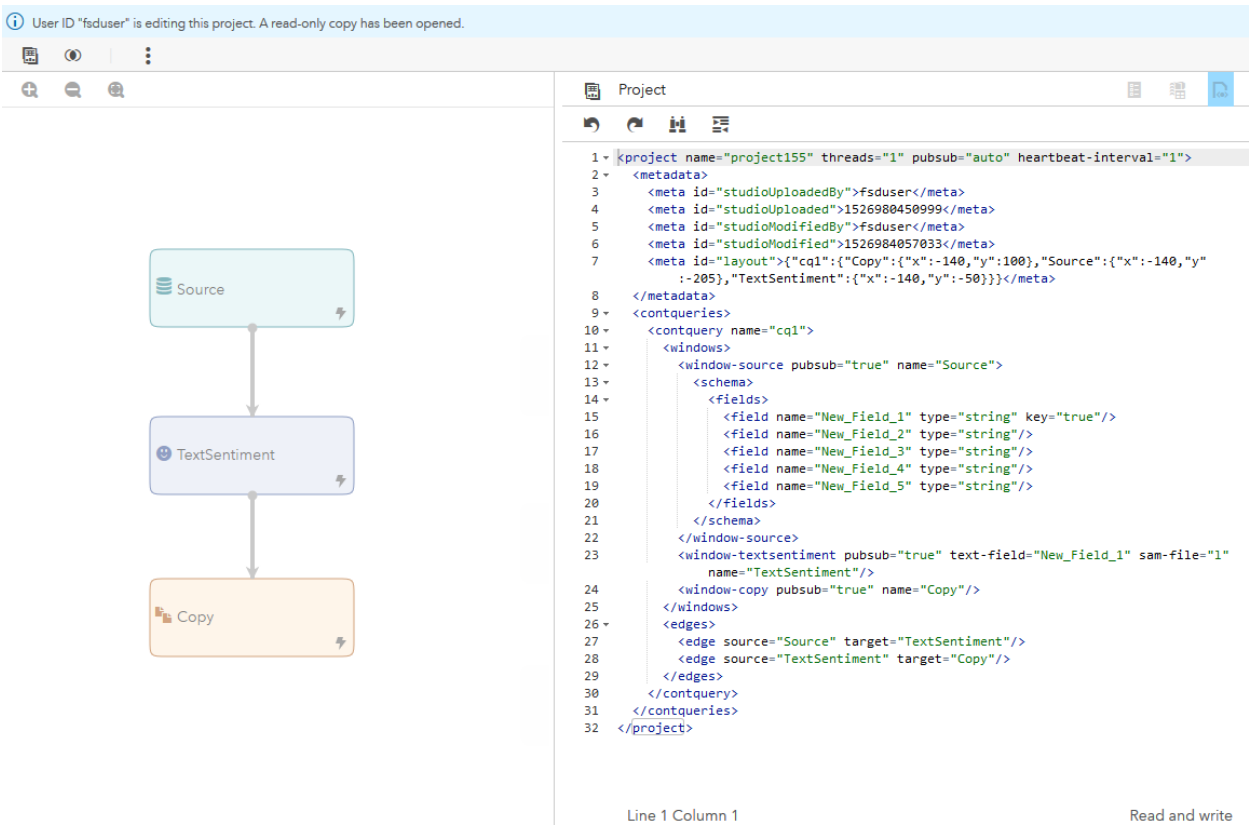
Project Locking

If you attempt to open a project that is locked by another user, you are informed that another user is editing the project. You are prompted to decide whether you want to continue. If you click **Yes**, the project opens in SAS Event Stream Processing Modeler in Read-Only mode. If you click **No**, you return to the **Projects** page.

Read-Only mode enables you to view the model and, if necessary, rearrange the position of the model's windows. However, you cannot save your changes.

Note: Projects are locked against your user name and by the instance of the application. If you are editing a project and you open the project in another browser tab or window, the lock remains valid.

Figure 9 *Viewing a Project in Read-Only Mode*



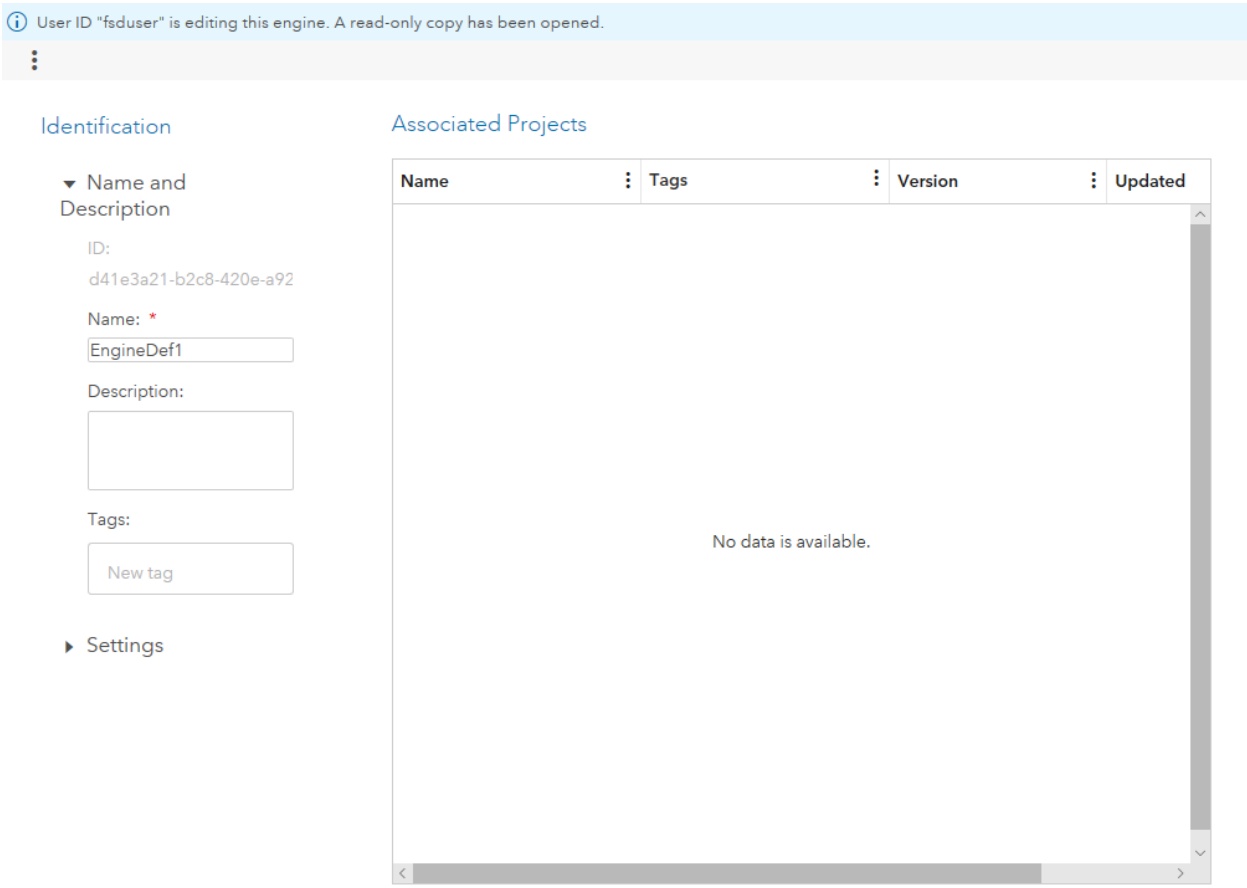
The **Windows** pane and magnification icons are unavailable in Read-Only mode. Therefore, you cannot add windows to your model using the **Windows** pane or adjust your model's magnification.

Engine Definition Locking

If you attempt to open an engine definition that is locked by another user, you are informed that another user is editing the engine definition. You are prompted to decide whether you want to continue. If you click **Yes**, the engine definition opens in Read-Only mode. If you click **No**, you return to the **Engine Definitions** page.

Read-Only mode enables you to view the engine definition's properties and to download the engine definition to your computer. However, you cannot modify the engine definition's properties or save your changes.

Figure 10 *Viewing an Engine Definition in Read-Only Mode*



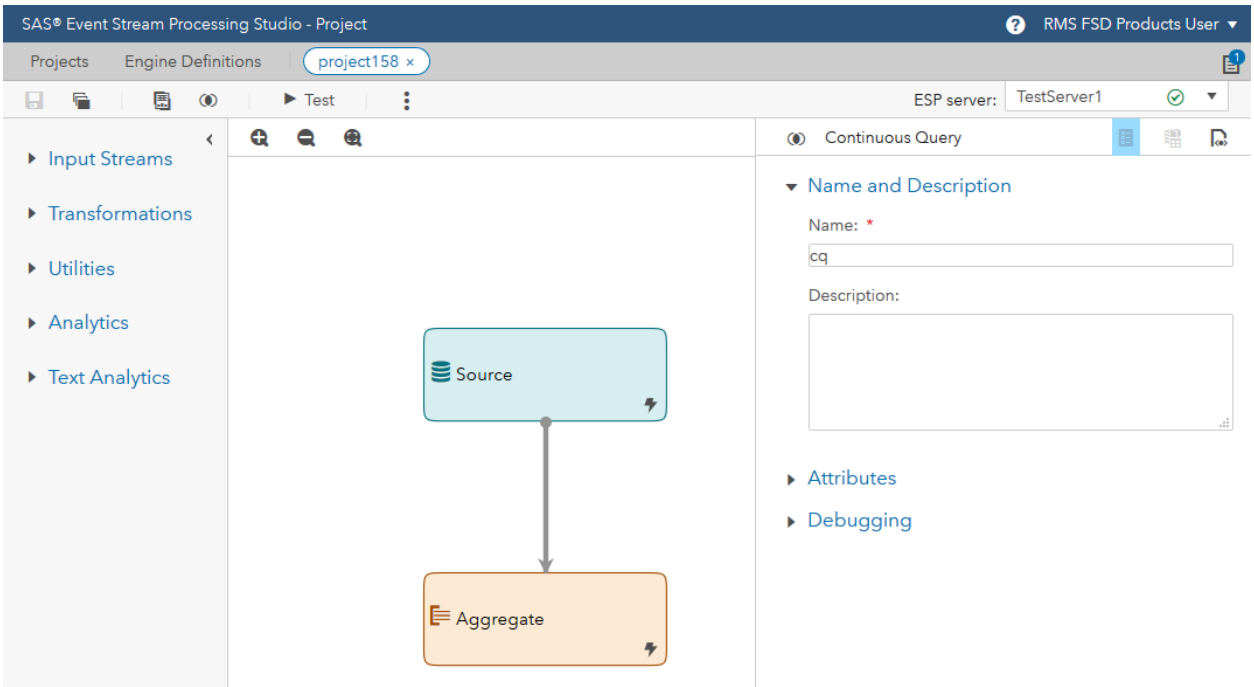
Using SAS Event Stream Processing Studio Modeler

Overview

SAS Event Stream Processing Studio Modeler enables you to construct, change, and test SAS Event Stream Processing models. A *model* specifies how a SAS Event Stream Processing engine analyzes and then transforms input event streams into meaningful results.

SAS Event Stream Processing Studio Modeler appears when you create a new project or open an existing project.

Figure 11 *SAS Event Stream Processing Studio Modeler*



Note: You can increase or decrease the magnification of your model by using the zoom buttons. Click to increase the magnification and click to decrease the magnification. To adjust the magnification of your model so that the entire model appears in the workspace, click .

The modeler displays one continuous query at a time. When you create a new model, a continuous query named `cq1` is created by default. To construct your model, you must configure at least one continuous query. For more information about configuring continuous queries, see "Configuring Continuous Queries in SAS Event Stream Processing Studio" on page 26.

Note: To pan your model, click anywhere in the workspace and then drag the cursor in the appropriate direction.

Configuring a Model's ESP Server

When you create a model, if you have not configured any ESP servers, you are prompted to decide whether you want to configure an ESP server. If you decide to create an ESP server, the ESP server that you create becomes the model's default ESP server and appears in SAS Event Stream Processing Studio Modeler:

If you want to test your model in test mode, you must create an ESP server and assign it to your model. However, creating an ESP server is optional when viewing or editing a model. For more information about testing your model, see "Testing Models in SAS Event Stream Processing Studio" on page 27. If you have configured multiple ESP servers, you can change the ESP server that is associated with your model by selecting an alternative ESP server from the **ESP server** drop-down list.

Note: Functionality is limited if you open a model that does not have an assigned ESP server. For example, connector properties are unavailable to models for which ESP servers are not assigned.

To manage your ESP servers, select **Manage ESP servers** from the **ESP server** drop-down list. For more information about managing ESP servers, see "Managing ESP Servers in SAS Event Stream Processing Studio" on page 29.

Adding Windows

To add windows to the continuous query that is currently displayed, drag a window from the **Windows** pane on the left to the workspace.

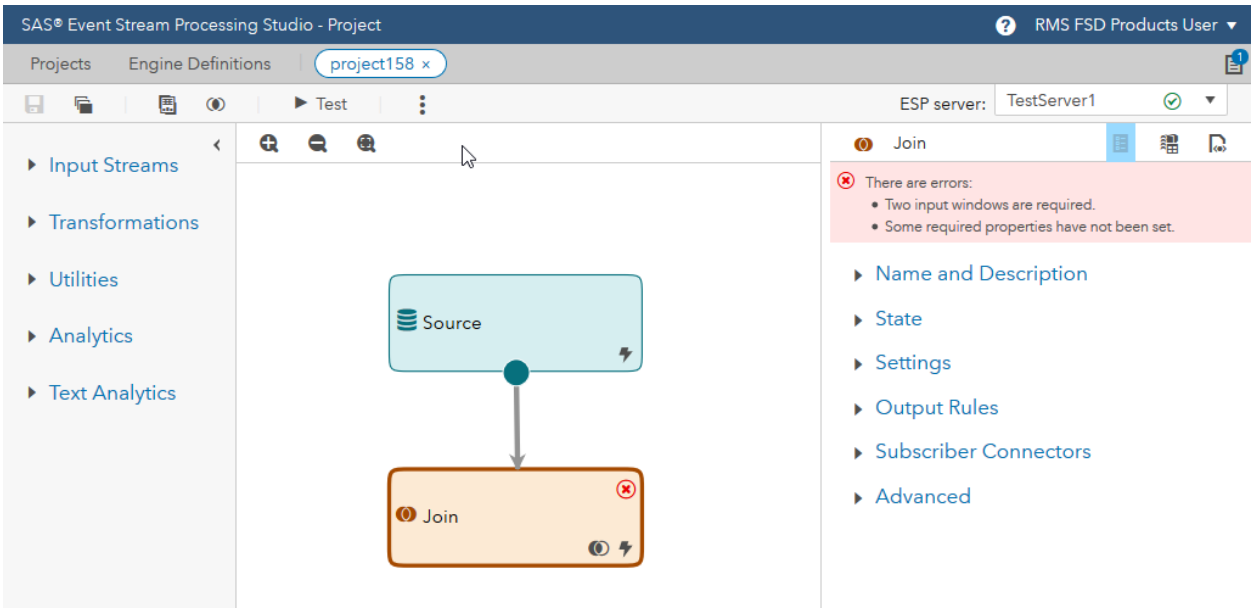
Note: Alternatively, you can add a window to the continuous query by double-clicking the relevant window in the **Windows** pane.

The windows are grouped into the following categories:

- Input Streams
- Transformations
- Utilities
- Analytics
- Text Analytics

You must ensure that you enter valid properties for each window. Entering invalid window properties causes the window to display an error icon. In addition, the right pane that displays the window's properties shows a warning message. Here is an example of a model where the Join window contains invalid properties:

Figure 12 A Window Validation Warning Message

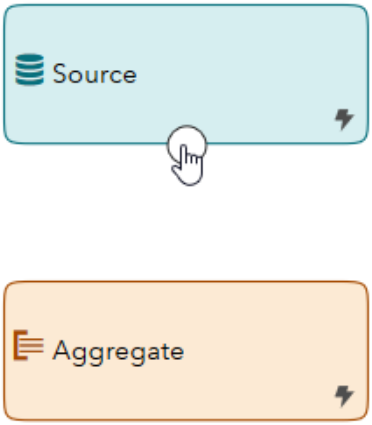


Connecting Windows

To connect a window to another window with an edge:

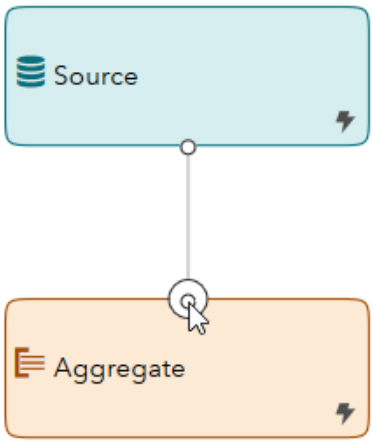
1 Position the cursor over the anchor point at the bottom of the window so that the anchor point color changes from black to white:

Figure 13 *Cursor over the Anchor Point*



2 Click the white anchor point, hold the left mouse button down, and draw a line to the anchor point of another window:

Figure 14 *An Edge Connecting Two Windows*



The edge automatically connects to the window.

Note: You cannot change a connection by moving an edge from one window to another. Instead, you must establish a new connection by creating a new edge. If you have created a connection between two windows in error, you must delete the edge. To do this, select the edge that you want to delete and press **Delete**.

Edge Display Types

Connecting edges can appear differently depending on the type of connection between windows.

For information about edge display types, see the following table:

Edge Display Type	Connecting Edge	Details
Event stream		Event stream edges connect input windows that contain event stream data.

Edge Display Type	Connecting Edge	Details
Supporting data	→	Supporting data edges connect input windows that contain geometric data (that is, where the edge role is set to `Geometries`). They can also connect secondary input windows to a Join window.
Non-data edges	- ->	Non-data edges connect windows that do not contain event stream data (that is, where the edge role is set to `Model` or `Request`).

Note: An invalid edge appears as a red dashed line in SAS Event Stream Processing Studio Modeler.

Edge Roles

You can use SAS Event Stream Processing Studio Modeler to configure the edge roles of connecting edges in your model. Edge roles must be specified for edges that connect streaming analytics windows and for edges that connect Geofence and Join windows. Each edge is assigned a role by default.

To change an edge's default role:

1. In the workspace, select the edge whose role you want to change.
2. In the right pane, in the **Role** field, change the default selection.

For information about edge roles, see the following table:

Window Name	Default Edge Role	Available Edge Roles	Dependencies
Join	Left table	Left table or Right table	If you assign an edge role to a Join window's connecting edge, the remaining connecting edge is automatically assigned the alternative edge role.
Geofence	Position	Position or Geometry	If you assign an edge role to a Geofence window's connecting edge, the remaining connecting edge is automatically assigned the alternative edge role.
Model Reader	Request	Request	Not applicable
Model Supervisor	Request	Request or Model	Not applicable
Calculate	Data	Data or Request	Not applicable
Train	Data	Data or Request	Not applicable
Score	Data	Data or Model	If you assign an edge role to a connecting edge, the remaining connecting edge is automatically assigned the alternative edge role.

Deleting Windows and Edges

To remove a selected window or an edge from the model, press **Delete**.

Note: Deleting a window from a model automatically deletes all its connecting edges.

Window Icons

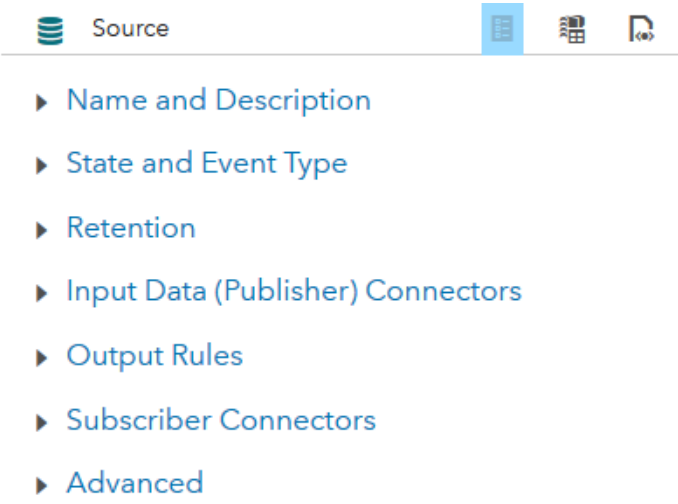
Each window in your model can display icons that represent its current state. For example, a Source window that contains a publisher connector displays . For information about each icon, see the following table:

Icon	Description
	Indicates that the window's schema is invalid.
	Indicates that the window contains a fully stateful index that is stored in memory. **Note:** If the window contains a non-stateful index, this icon is not displayed.
	Indicates that the window contains a fully stateful index that is stored on disk.
	Indicates that the Source window contains a publisher connector or a subscriber connector. **Note:** Connector icons that appear on the left of a window indicate that the window contains a publisher connector. Connector icons that appear on the right of a window indicate that the window contains a subscriber connector.
	Indicates that the Join window contains an inner join type.
	Indicates that the Join window contains a full outer join type.
	Indicates that the Join window contains a right outer join type.
	Indicates that the Join window contains a left outer join type.

Configuring the Properties of Windows

To configure the properties of a window, click on the window in the workspace. The right pane displays the properties for that window. Edit the properties as required.

Figure 15 *Properties of the Source Window*



Note: If the right pane displays XML code, you are viewing the XML Editor. Click in the right pane to display the properties instead.

Using the XML Editor

Overview

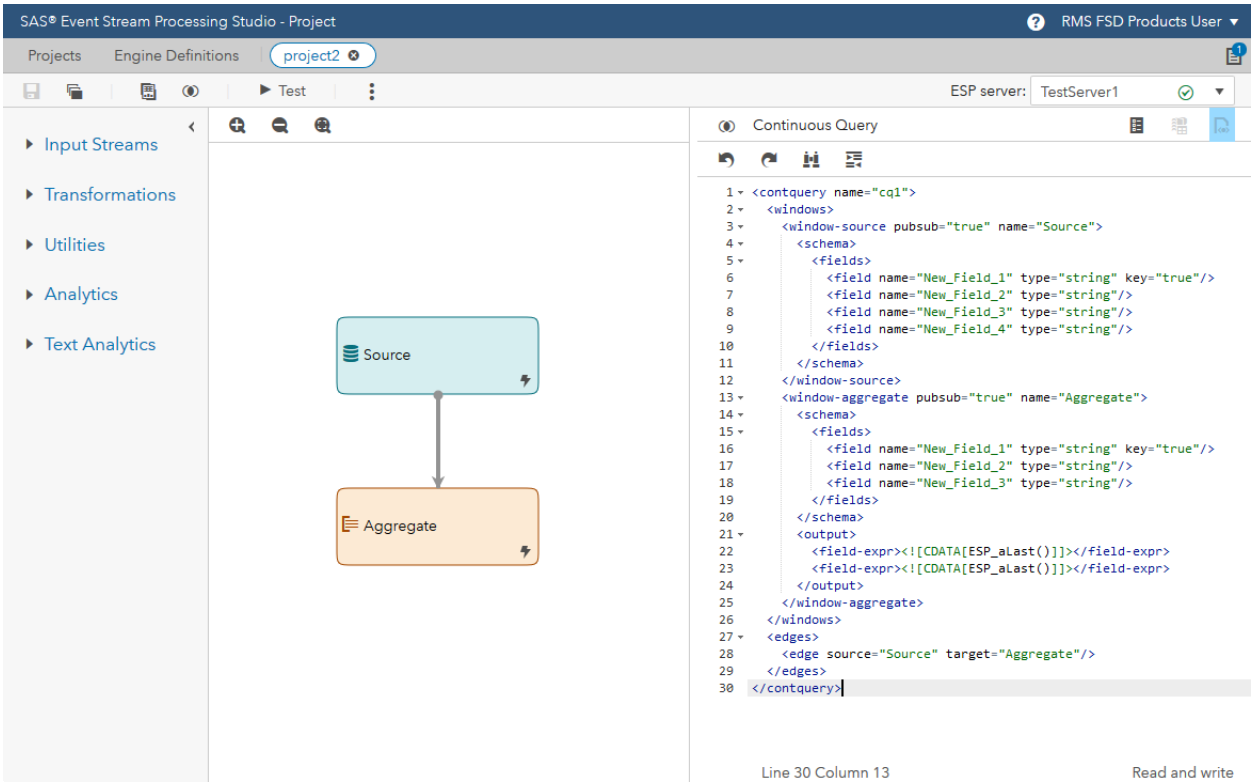
SAS Event Stream Processing Studio Modeler includes the XML Editor. You can use it as an alternative way of creating models, compared to the visual modeling capabilities in SAS Event Stream Processing Studio Modeler. The workspace displays a snapshot of your model's XML code. You can use the XML Editor to rename a window. To do this, select the window that you want to rename in the workspace and then change the window's name in the XML Editor.

CAUTION! Manually editing your model's XML code using the XML Editor can result in an invalid model. Using SAS Event Stream Processing Studio Modeler to construct your model limits the possibility of your model containing invalid XML code. You must correct any invalid XML in the XML Editor before you can switch back to the **Properties** pane. Changes that you make manually in the XML Editor are not always reflected in the workspace. Using the XML Editor to rename a window, without first selecting it, results in the window being replaced by a new window in a default position on the workspace. This invalidates your model. Any connections to or from the redundant window must then be deleted and then re-created in the workspace. Alternatively, you can manually edit the edges in the XML Editor.

To open the XML Editor, open a project and click in the right pane.

The right pane displays the XML Editor.

Figure 16 *The XML Editor within SAS Event Stream Processing Studio Modeler*



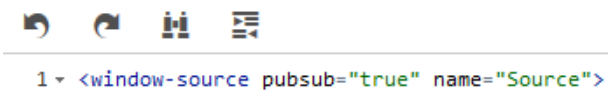
Selecting a specific element in your workspace reloads the XML Editor to display only the corresponding section of XML code. To display the entire project's XML again, click . If you have associated a project with an engine, the XML code that specifies the engine is not included in the project's XML code. This information is instead included as metadata in the SAS Event Stream Processing Studio database.

If you selected a specific ESP window, clicking an area of white space in your workspace reloads the XML Editor to display the XML relating to your model's continuous query.

If you want to add a comment to your XML code, you must ensure that the comment is enclosed within its relevant XML element. If you do not, the comment will appear at the top of the XML Editor the next time the XML code is automatically reordered. The XML code is automatically reordered to maintain consistency with the SAS Event Stream Processing schema.

When a model is created, optional attributes are not included in its XML code. Models also contain settings that are not directly specified in the model's XML code, but they are represented in the user interface. For example, if a Source window has a default window state of **(inherit from query) pi_HASH**, the implied setting is not displayed in the XML code. See the example shown here:

Figure 17 *The XML Editor Displaying a Source Window's XML Code without the Window's Default State*



Changing the window's state from its default value includes the attribute in the model's XML code, as shown here:

Figure 18 *The XML Editor Displaying a Source Window's XML Code with the Window's Default State*

```
1 <window-source pubsub="true" index="pi_EMPTY" name="Source">
```

Using Editing Tools and Keyboard Shortcuts

The XML Editor includes a toolbar that contains editing tools. These tools are also accessible using keyboard shortcuts.

Icon	Action	Keyboard Shortcut
	Reverts your previous change	Ctrl + Z
	Reverts the effects of the undo action	Ctrl + Y
	Prompts you to search for specific text. Pressing Ctrl + F again prompts you to replace the text that you have searched for.	Ctrl + F
	Formats the XML code that you have manually entered	Not available
No icon	Removes the code that you have selected from its original position	Ctrl + X
No icon	Copies the code that you have selected to the clipboard	Ctrl + C
No icon	Pastes the code on the clipboard at the cursor's position	Ctrl + V
No icon	Selects all of the code in the XML editor.	Ctrl + A

Validation

The XML Editor automatically validates the syntax of the code that you enter. If you enter invalid code, the XML Editor displays the following warning:

Figure 19 *Invalid XML Warning*

```
1  <window-source pubsub="true" index="pi_EMPTY" name="Source">
2    <schema>
3      <fields>
4        <field name="New_Field_1" type="string" key="true"/>
5        <field name="New_Field_2" type="string"/>
6        <field name="New_Field_3" type="string"/>
7        <field name="New_Field_4" type="string"/>
8        <field name="New_Field_5" type="string"/>
9      </fields>
10   </schema>
11 </window-source
```

Position the cursor over the warning icon to view a generic description of the error in your XML code. The icon displays the location of the error in your XML code. For a more detailed description of the error, position the cursor over the icon.

Efficiency Tips

For some window types, you can use SAS Event Stream Processing Studio Modeler to copy schema fields if there are windows in your model that use the same fields. In the Output Schema window, click to open the Copy Fields from Input Schema window. Select the schema fields that you want to copy and click **OK**. Alternatively, you can use the XML Editor to copy and paste the fields between the windows.

Note: This functionality is not available for window types where it is not appropriate for schema fields to be copied from another window. For example, you cannot copy schema fields to or from windows that contain schemas that are implied or have been internally generated.

Project Metadata

Unique identifying project information is displayed within the `<metadata>` element in your project's XML code. Although the metadata is displayed in your project's XML code, it is located in the SAS Event Stream Processing Studio database. Fore more information, see “Project Metadata” on page 11.

Configuring Continuous Queries in SAS Event Stream Processing Studio

Continuous queries allow SAS Event Stream Processing engines to analyze and manipulate data. *Continuous queries* are queries that run automatically and periodically on data in real time.

SAS Event Stream Processing models must contain at least one continuous query. SAS Event Stream Processing Studio Modeler creates a continuous query `cq1` by default. You can then add and configure windows within this continuous query. Your model can contain many continuous queries.

Your continuous query must contain at least one Source window. Source windows connect to one or more derived windows (for example, a Pattern or Join window). After you have created a Source window, you can then add derived windows to your model.

To configure the properties of a continuous query:

1. On the **Projects** page, right-click the project that contains the continuous query that you want to configure, and select **Open Project**.

   SAS Event Stream Processing Studio Modeler appears. The right pane displays the project’s properties.

2. Click on the toolbar.

   The right pane displays the properties of the continuous query that you selected when the project was last saved.

3. Configure the fields as required.

To add a new continuous query to your model, click on the toolbar and select **Add continuous query**. You can also delete continuous queries from your model by selecting **Delete continuous query**.

To switch between each continuous query in your model, select the continuous query that you want to view from the **Continuous Query** drop-down list on the toolbar.

Figure 20 *The Continuous Query Drop-down List on the Toolbar*

Testing Models in SAS Event Stream Processing Studio

You can use SAS Event Stream Processing Studio to verify that your model operates as intended. You can analyze how incoming data is transformed into meaningful event streams that can be consumed by subscribers.

Note: An *engine* is the top–level container in the SAS Event Stream Processing model hierarchy and can contain one or more projects. A project can contain one or more continuous queries. When you are running a model in test mode, only the project is tested. Test mode does not test engines.

1. On the **Projects** page, right-click the project that contains the model that you want to test.
2. Select **Open project** from the menu.

   The project appears in a new page.
3. Click ▶ Test.

   A page appears, enabling you to test your model.
4. From the **ESP server** drop-down list, select an ESP server to perform the test on.

   Note: If SAS Event Stream Processing Studio does not contain any registered ESP servers, you must register one before you can continue testing your model. You can register a new ESP server from the Manage ESP Servers window. To access the Manage ESP Servers window, select **Manage Test Servers** from the **ESP server** drop-down list.
5. To configure your test's settings, click .

   The Test Results Settings window appears.

   a. To return events to test mode in real time, select **Return events from the server as they happen**. Alternatively, to return events to test mode in pages, select **Return events from the server in pages**.

      Note: This version of SAS Event Stream Processing Studio uses the WebSocket protocol to subscribe to windows. Models that are executed in this version of SAS Event Stream Processing Studio might display events in a different sequence than models that were executed in previous versions. The order of the events delivered to the WebSocket subscriber will not match the order of the events received from the engine. If you are using the WebSocket subscriber, the event key and event state take precedence over the sequence of events received from the engine.

      If you selected **Return events from the server in pages**, do the following:

      i. In the **Maximum page size (events)** field, enter the maximum number of events to be displayed in a results page.

      Note: To prevent an excessive number of results being simultaneously returned from the ESP server, you can display results in paged format. This can improve your browser's performance. However, if the total number of results is greater than maximum page size, some results are not displayed.

ii In the **Interval (ms)**, enter an interval at which each page is to be returned from the ESP server (in milliseconds).

b Click **OK**.

6 From the list of windows on the left of the screen, select the windows whose results you want to view.

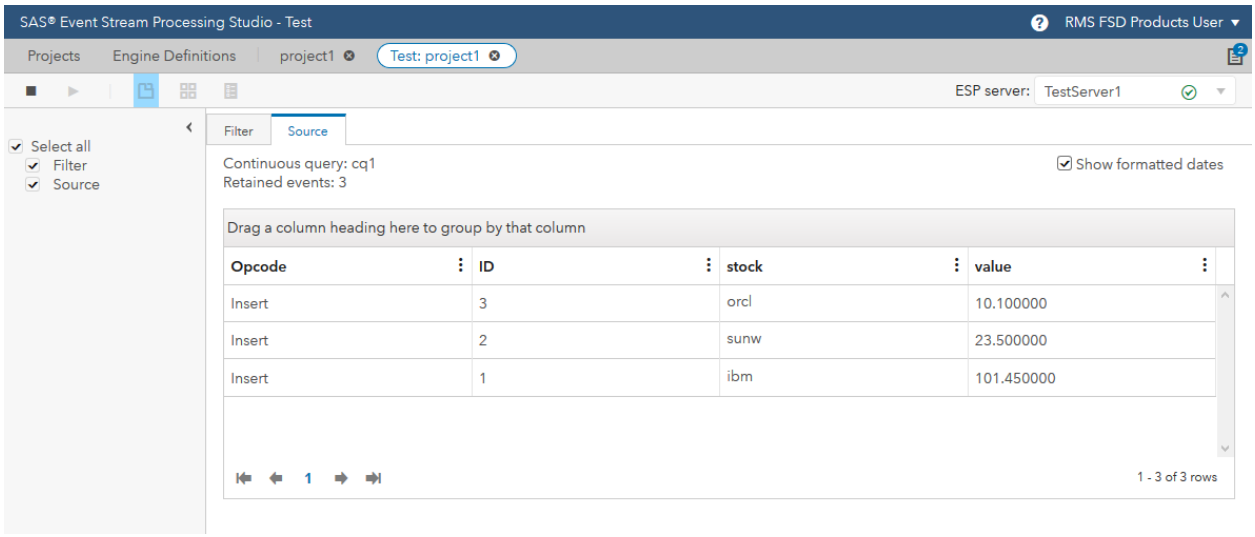
7 To run the test, click ▶.

Each window's results appear in their corresponding tab. Only windows whose event stream you have subscribed to can display data. If you subscribed to view the results of six windows or fewer, you can choose to view your test results in windowed format by clicking .

You can group information by column. The results table contains a horizontal bar at the top of the table, with the text **Drag a column header and drop it 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.

Alternatively, you can view your test results by opening the output file that you specified in your subscriber connector properties.

You can use the **Show formatted dates** check box to control whether information appears as Coordinated Universal Time (UTC) in ISO 8601 format (for example, 2018-10-22T13:33:26.000Z). The check box is selected by default. If you clear the check box, date and double data types appear in the format received from the ESP server instead (UNIX Epoch time).



Note: If your event stream contains more than 1,0000 results, only the last 1,0000 results in the stream are displayed in test mode.

8 To stop the test, click ■.

9 When you have finished testing your model, close the page.

Note: If the test fails and the resulting error message does not explain what caused the failure, see the SAS Event Stream Processing engine log files for additional information.

Managing ESP Servers in SAS Event Stream Processing Studio

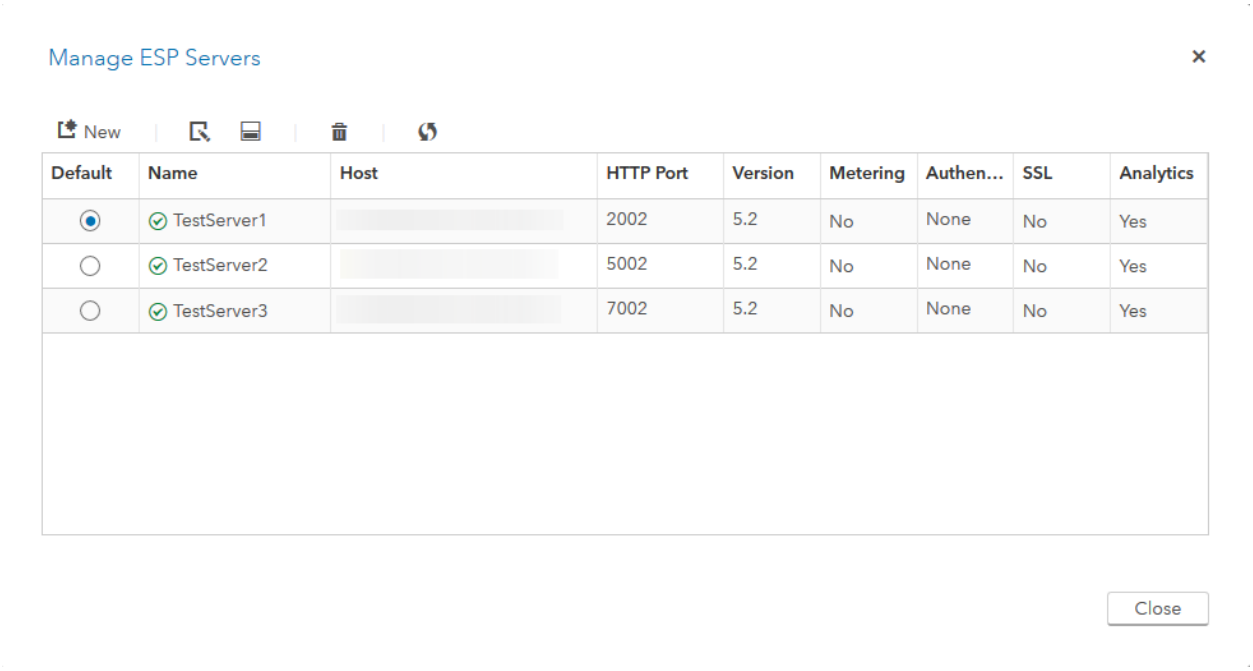
You can use the Manage ESP Servers window to view, create, edit, and delete ESP servers in SAS Event Stream Processing Studio.

About Managing ESP Servers

You can access the Manage ESP Servers window from the **Projects** page. To do this, click ⋮ and select **Manage ESP servers**. Alternatively, you can access the Manage ESP Servers window from SAS Event Stream Processing Studio Modeler by selecting **Manage ESP servers** from the **ESP server** drop-down list.

The following figure shows an example:

Figure 21 *The Manage ESP Servers Window, with Some ESP Servers Listed*



The Manage ESP Servers window displays the following information for each ESP server:

- The default ESP server.
- The ESP server's health.
- The ESP server's name.
- The host on which the ESP server is running.
- The port that is used for HTTP administration requests.
- The SAS Event Stream Processing version installed on the host on which the ESP server is running.
- Whether SAS Event Stream Processing has been enabled to meter the number of events that are processed on the ESP server.

- Whether authentication is enabled on the ESP server. If authentication is enabled, the time at which it was enabled appears.
- Whether the SAS Event Stream Processing ESP server is configured to require SSL encryption.

The Name column displays an icon alongside the ESP server's name summarizing the ESP server's condition. The condition of the ESP server is determined by the server's connectivity and availability. This information helps you focus on the ESP servers that have problems. The following icons can appear in the Name column:

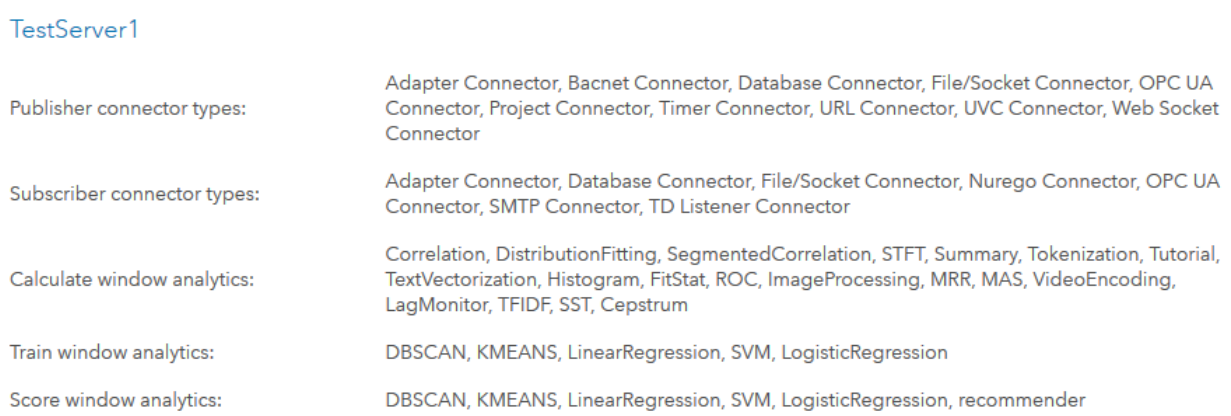
Icon	Description
Healthy	The ESP server is available and operating normally.
Errors Reported	The ESP server is not available.

You can also view additional information relating to each ESP server in your deployment, such as the connector types and analytics available.

To view an ESP server's additional information, select the ESP server whose additional information you want to view and click .

A tile appears that contains additional information relating to the ESP server, as shown here:

Figure 22 *ESP Server Details*



To hide the additional information again, click .

Create an ESP Server

To create a new ESP server:

1 Click New.

The ESP Server Properties window appears.

2 In the **Name** field, enter a name to identify the new ESP server.

3 In the **Host** field, enter the new ESP server's host name or IP address.

4 In the **HTTP port** field, enter the new ESP server's administration port number.

5 If required, change the setting for the **Authentication** field:

- **None**: This is the default option.
- **Authenticate using an OAuth token**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field appears where you must enter the OAuth token.
- **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields appear where you must enter the user name and password.

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

7 Click **OK**.

Edit an ESP Server

To edit an existing ESP server's details:

1 Select the ESP server that you want to open and click .

The Edit ESP Server Properties window appears.

2 Edit the information in the **Name**, **Host**, and **HTTP Port** fields as required.

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

4 Click **OK**.

Delete an ESP Server

Deleting an ESP server removes it from the table on the Manage ESP Servers window. Deleting ESP servers can be useful if the table contains ESP servers that are no longer used. You can delete an ESP server that is still running. To delete a specific ESP server from the table:

1 Select the ESP server that you want to delete and click .

The Remove ESP Server window appears.

2 Click **Yes** to confirm the deletion of the ESP server.

Refresh the Manage ESP Servers Window

To refresh the table of ESP servers, click .

Publishing Project Versions

About Version Control

You can use SAS Event Stream Processing Studio to create and manage multiple versions of a project. If your deployment contains a running instance of SAS Event Stream Manager, publishing a project version from SAS Event Stream Processing Studio makes the project version available to SAS Event Stream Manager.

Note: You cannot edit project versions after they have been published. A project that you can edit is designated as the working copy of the project. It does not become a project version until you publish it. The working copy of the project enables you to make changes to the project without affecting any project versions that you previously published.

When you publish a project version, the version's XML code is updated to display its unique ID number. Project versions that are published for the first time are assigned a version number of 1.0. The number to the left of the decimal point is the project's major version number. The number to the right of the decimal point is the project's minor version number. Publishing subsequent versions of a project increments the major version number. In SAS Event Stream Processing Studio, you can publish major project versions, but you cannot publish minor project versions. Minor versions are created when you make a change to the project version in SAS Event Stream Manager (for example, when you import content from SAS Model Manager directly into your project version in SAS Event Stream Manager). You can view a project's major versions and minor versions in the version hierarchy on the **Versioning** page.

Publish a Version

To publish a version of a project:

1. On the **Projects** page, right-click the relevant project and select **Open Project**.

   SAS Event Stream Processing Studio Modeler appears.

   Note: To create a new version of a project, the version must exist in a valid state.

2. Click .

   The **Versioning** page appears. This page contains a version hierarchy that displays the versions of the project that you are working on.

3. Click .

   The Publish — Version window appears.

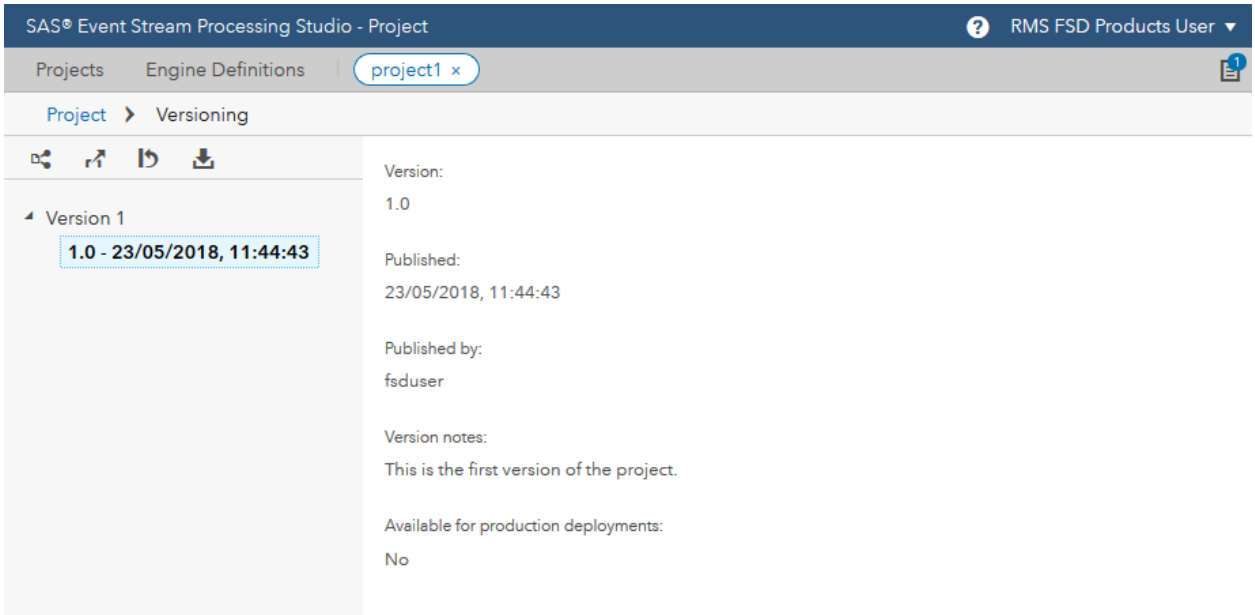
   a. In the **Version notes** field, enter any notes that relate to the version of the project that you want to publish. This enables you to maintain a record of a project's version history.

      Note: You cannot modify the version notes of a project version that has already been published.

   b. Click **OK**.

      The **Versioning** page displays your published project version in the version hierarchy.

4 To view information relating to the project version that you published, select the relevant version in the version hierarchy on the left.



When you publish a project version, the version's XML code is updated to display its unique ID number. The project's ID number, major version number, and minor version number are specified in the version's XML code as metadata. For more information, see "Working with Projects in SAS Event Stream Processing Studio" on page 8.

View a Published Version

To view a published version of a project:

1 On the **Projects** page, right-click the relevant project and select **Open Project**.

SAS Event Stream Processing Studio Modeler appears.

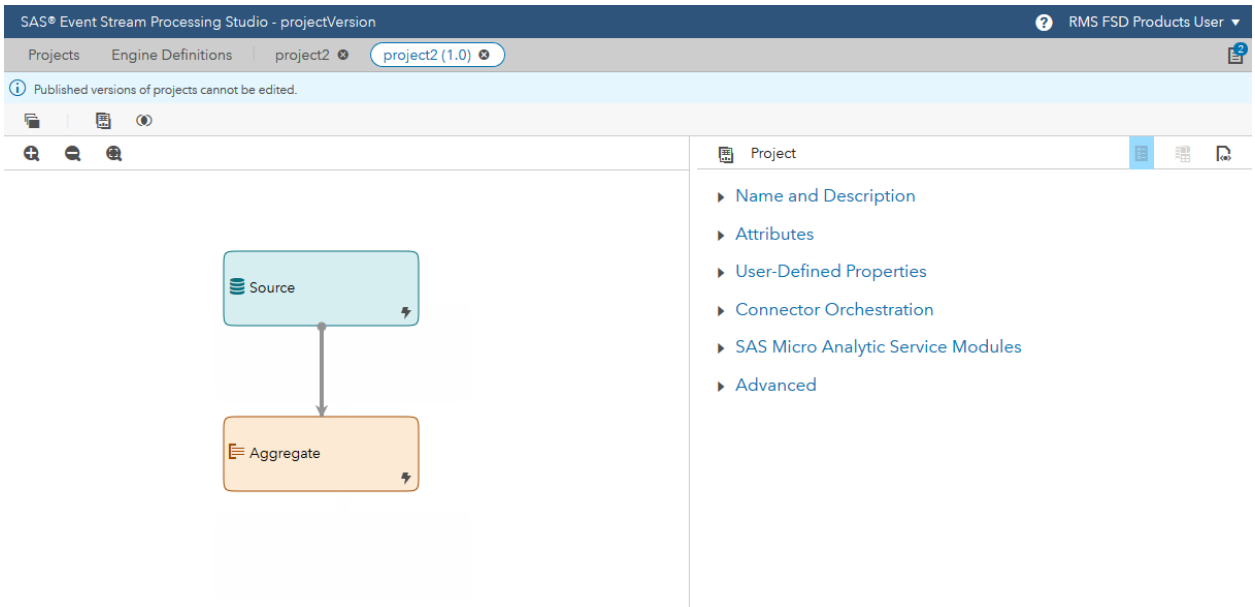
2 Click .

The **Versioning** page appears. This page displays a version hierarchy containing the current and previous versions of the project.

3 In the version hierarchy on the left, select the version of the project that you want to view in SAS Event Stream Processing Studio Modeler.

4 Click .

SAS Event Stream Modeler displays the version in Read-Only mode.



Note: You cannot make changes to a version of a project that has already been published. If you want to make more changes to a project, a new working copy is made available for you to edit in SAS Event Stream Processing Studio.

Revert to a Previous Version

To revert to a previous version:

1 On the **Projects** page, right-click the relevant project and select **Open Project**.

SAS Event Stream Processing Studio Modeler appears.

2 Click .

The **Versioning** page appears. This page displays a version hierarchy containing the current and previous versions of the project.

3 In the version hierarchy on the left, select the version of the project that you want to revert to.

4 Click .

The Revert to Version window appears.

5 Click **Yes** to confirm the reversion.

The working version of the project reverts to the published version that you selected in the version hierarchy. You cannot undo this operation.

Download a Version

To download a previously published version:

1 On the **Projects** page, right-click the relevant project and select **Open Project**.

SAS Event Stream Processing Studio Modeler appears.

2 Click .

The **Versioning** page appears. This page displays a version hierarchy containing the current and previous versions of the project.

3 In the version hierarchy on the left, select the version of the project that you want to download.

4 Click  .

The project version downloads to your computer.

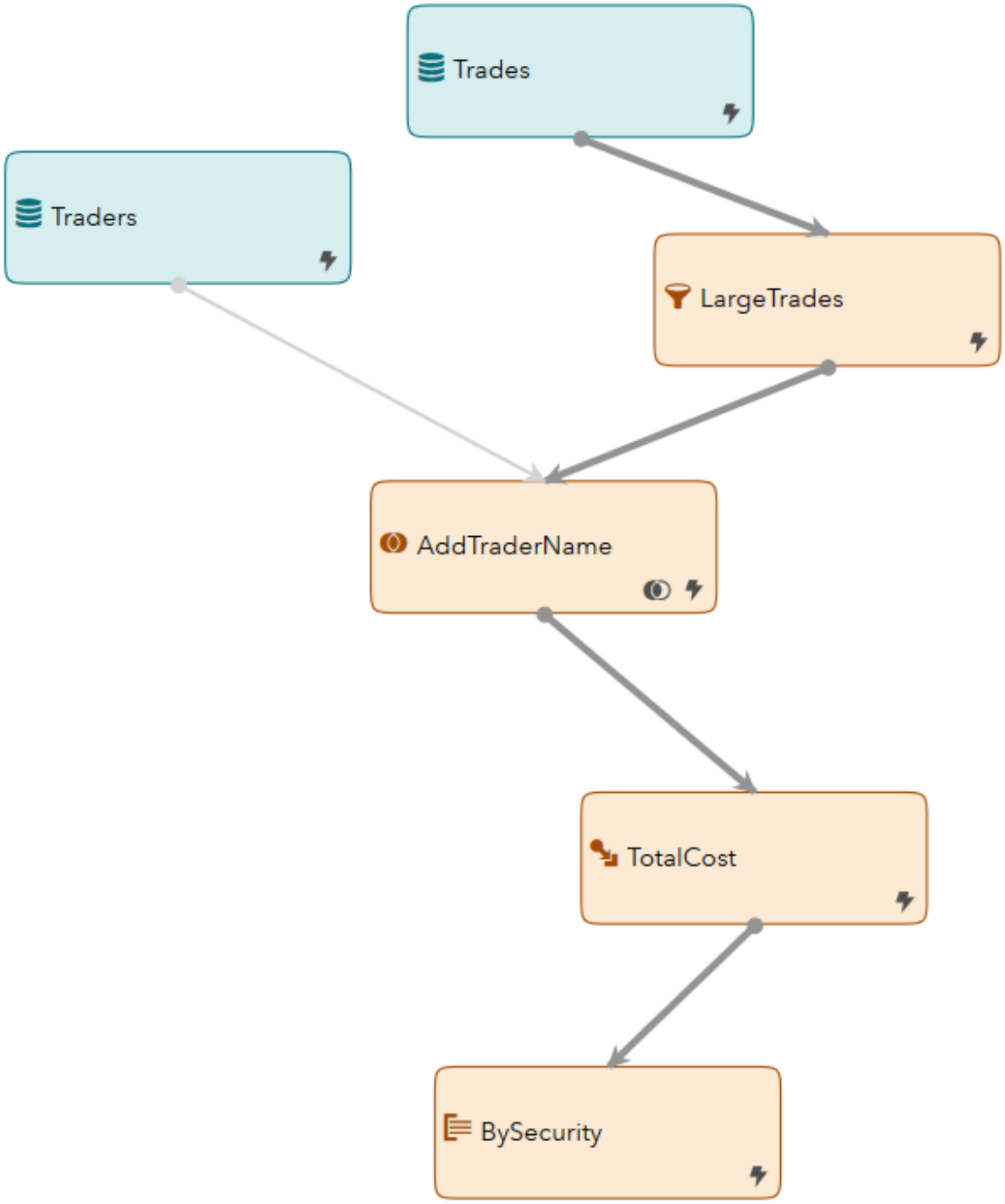
Note: The location of the project version that you downloaded might vary depending on your browser's configuration.

Example: Processing Trades

This example creates a model that processes stock market trades. The model identifies large securities transactions and the traders who were involved in those trades. The model performs the following actions:

- events about securities transactions are streamed to a Source window called Trades
- receives information about traders using a Source window called Traders
- identifies large trades using a Filter window called LargeTrades
- matches the large trades with the traders who made those trades using a Join window called AddTraderName
- computes the total cost of the large trades using a Compute window called TotalCost
- aggregates the large trades by security using an Aggregate window called BySecurity

Figure 23 *Diagram of the Trades Model*



Note: The comma-separated value (CSV) data and model XML code that are used in this example are available within your SAS Event Stream Processing installation, typically in the following location: `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/trades_xml`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

1 On the **Projects** page, click New.

   The New Project window appears.

2 In the New Project window, do the following:

   a In the **Name** field, enter `Trades`.

b In the **Description** field, enter a description. Here is an example: `This model can be used to identify large securities transactions and the traders who were involved in those trades.`

c Click **OK**.

If you do not currently have any ESP servers configured, you are prompted to decide whether you want to configure an ESP server now.

Note: It is assumed that you do not have any ESP servers configured. If you already have ESP servers configured, go to step 5.

3 Click **Yes** to configure an ESP server now.

The ESP Server Properties window appears.

4 Configure an ESP server:

a In the **Name** field, enter a name to identify the new ESP server that you want to create.

b In the **Host** field, enter the host name or IP address of the new ESP server.

c In the **HTTP port** field, enter the new ESP server's HTTP publish/subscribe port.

d If required, change the setting for the **Authentication** field:

- **None**: This is the default option.
- **Authenticate using an OAuth token**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field appears where you must enter the OAuth token.
- **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields appear where you must enter the user name and password.

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

f Click **OK**.

Your project is created with a set of default properties.

5 In the right pane, configure your project's properties:

a In the **Name** field, change the default name to `trades_proj`.

b Expand **Attributes**.

c In the **Threads** field, change the thread pool size to `4`.

6 Configure the project's continuous query:

a Click .

b In the right pane, in the **Name** field, change the continuous query's default name `cq1` to `trades_cq`.

7 Expand **Input Streams** on the **Windows** pane on the left and drag a Source window to the workspace.

The right pane displays the Source window's properties.

This window receives events about securities transactions.

8 Specify a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `Trades`.

b In the **Description** field, enter `Trades Source window`.

9 Specify an output schema for the Trades window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values in the rows:

Key	Field Name	Type
Y	tradeID	String
N	security	String
N	quantity	Int32
N	price	Double
N	traderID	Int64
N	time	Timestamp

d Click **OK**.

10 Configure the Trades window to stream events from a file called trades.csv that contains securities transactions. You can find this example CSV file in the `trades_xml` folder in the `examples` directory. To add a connector to this CSV file:

a In the right pane, click .

b Expand **Input Data (Publisher) Connectors**.

c Click .

The Connector Configuration window appears.

d In the **Name** field, replace the default value with `TradesConnector`.

e In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/trades_xml/trades.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

f In the **Fstype** drop-down list, select **csv**.

g Configure the TradesConnector connector's properties:

i Click **All properties**.

The All Properties window appears.

ii Enter `%d/%b/%Y:%H:%M:%S` in the **dateformat** property's **Value** field.

iii Click **OK**.

h Click **OK**.

11 Collapse **Input Data (Publisher) Connectors**.

12 Specify a state for the Trades window:

a Expand **State and Event Type**.

b In the **Window state and index** drop-down list, select **Stateful (pi_RBTREE)**.

13 Expand **Input Streams** on the **Windows** pane on the left and drag another Source window to the workspace.

The right pane displays the Source window's properties.

Configure this window to receive information about stock market traders.

14 Specify a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `Traders`.

b In the **Description** field, enter `Traders Source window`.

15 Specify an output schema for the Traders window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values:

Key	Name	Type
Y	ID	Int64
N	name	String

d Click **OK**.

16 Configure the Traders window to receive information from a file called traders.csv that contains details of stock market traders. You can find this example CSV file in the `trades_xml` folder in the `examples` directory. To add a connector to this CSV file:

a In the right pane, click .

b Expand **Input Data (Publisher) Connectors**.

c Click .

The Connector Configuration window appears.

d In the **Name** field, replace the default value with `TradersConnector`.

e In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/trades_xml/traders.csv`. Replace <*release*> with the release number in your SAS Event Stream Processing installation directory path.

f In the **Fstype** drop-down list, select **csv**.

g Click **OK**.

17 Expand **Transformations** on the **Windows** pane on the left and drag a Filter window to the workspace.

Configure this window to identify large trades. In this example, a trade is regarded as a large trade if the quantity of stock traded equals or exceeds 100.

18 Click the newly created Filter window on the workspace.

The right pane displays the Filter window's properties.

19 Specify a name and description for the Filter window:

a In the **Name** field, change the default name to `LargeTrades`.

b In the **Description** field, enter `LargeTrades Filter window`.

20 Specify a filter expression for the LargeTrades window:

a Expand **Filter**.

b In the **Expression** field, enter `quantity >= 100`

c Collapse **Filter**.

21 Connect the Trades window to the LargeTrades window with an edge:

a Position the cursor over the anchor point at the bottom of the Trades window so that the anchor point changes from black to white.

b Click the white anchor point, hold the mouse button down, and draw a line to the anchor point in the LargeTrades window.

The LargeTrades window now accepts trades from the Trades window.

22 Expand **Transformations** on the **Windows** pane on the left and drag a Join window to the workspace.

Configure this window to match large trades with the traders who made those trades.

23 Specify a name and description for the Join window:

a In the right pane, in the **Name** field, change the default name to `AddTraderName`.

b In the **Description** field, enter `AddTraderName Join window`.

24 Connect the LargeTrades window to the AddTraderName window with an edge.

The AddTraderName window now accepts trades from the LargeTrades window.

25 Connect the Traders window to the AddTraderName window with an edge.

The AddTraderName window now accepts trader names from the Traders window.

26 Click the AddTraderName window on the workspace.

The right pane displays the AddTraderName window's properties.

27 Configure the AddTraderName window’s settings:

a In the right pane, expand **Settings** and notice that the LargeTrades window is regarded as the left window and the Traders window is regarded as the right window. This is due to the order in which you added the edges.

b In the **Output field calculation method field**, select **Select fields**.

c Collapse **Settings**.

28 Configure the AddTraderName window's join conditions:

a Expand **Join Conditions**.

b In the **Join Conditions** section, click to add a row to the table.

c Click the cell in the Left: LargeTrades column, and select **traderID**.

d Click the cell in the Right: Traders: column, and select **ID**.

e Collapse **Join Conditions**.

29 Configure the AddTraderName window's join criteria:

a Expand **Join Criteria** if it is not already expanded.

b Confirm that the **Join Type** drop-down list has a default value of **LeftOuter**.

c Collapse **Join Criteria**.

30 Specify a schema for the AddTraderName window:

a In the right pane, click .

b Click .

The Output Schema window appears. Use this window to configure the fields as shown in the following table. The schema fields that are required have already been defined previously. Click to open the Copy Fields from Input Schema window. Select the following schema fields and click **OK**.

Window	Field	Type
LargeTrades	security	String
LargeTrades	quantity	Int32
LargeTrades	price	Double
LargeTrades	traderID	Int64
LargeTrades	time	Timestamp
Traders	name	String

The Output Schema window displays the fields that you selected.

c Click **OK**.

31 Click .

32 Expand **Transformations** on the **Windows** pane on the left and drag a Compute window to the workspace.

The right pane displays the Compute window's properties.

Configure this window to compute the total cost of the large trades.

33 Specify a name and description for the Compute window:

a In the **Name** field, change the default name to `TotalCost`.

b In the **Description** field, enter `TotalCost Compute window`.

34 Connect the AddTraderName window to the TotalCost window with an edge.

The TotalCost window now accepts trades from the AddTraderName window.

35 Click the TotalCost window to display the Compute window's properties in the right pane again.

36 Specify an output schema for the TotalCost window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table.

Enter the following values:

Note: Alternatively, you can copy schema fields that you have previously defined. Click to open the Copy Fields from Input Schema window. Select the schema fields that you want to copy and click **OK**.

Key	Field Name	Type	Expression
Y	tradeID	String	(not applicable)
N	security	String	security
N	quantity	Int32	quantity
N	price	Double	price
N	totalCost	Double	price*quantity
N	traderID	Int64	traderID
N	time	Timestamp	time
N	name	String	name

d Click **OK**.

37 Click .

38 Expand **Transformations** on the **Windows** pane on the left and drag an Aggregate window to the workspace.

The right pane displays the Aggregate window's properties.

Configure this window to compute the total cost of the large trades.

39 Specify a name and description for the Aggregate window:

a In the **Name** field, change the default name to `BySecurity`.

b In the **Description** field, enter `This window computes the total cost of the large trades`.

40 Connect the TotalCost window to the BySecurity window with an edge.

The BySecurity window now accepts trades from the TotalCost window.

41 Click the BySecurity window to display the Aggregate window's properties in the right pane again.

42 Specify an output schema for the BySecurity window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. Enter the following values:

Key	Field Name	Type	Aggregate function	Parameters
Y	security	String	(not applicable)	(not applicable)
N	quantityTotal	Double	ESP_aSum	quantity
N	costTotal	Double	ESP_aSum	totalCost

d Click **OK**.

43 The model is now complete. Click to save your model.

44 Click ▶ Test.

A new page called **Test: trades_proj** appears.

45 In the **ESP Server** drop-down list, select the ESP server on which you want to test the model.

46 Click ▶.

The results for each window appear on separate tabs:

- The **Trades** tab lists the securities transactions
- The **Traders** tab lists the traders
- The **LargeTrades** tab lists the large trades
- The **AddTraderName** tab lists the large trades and includes an additional column that shows trader names
- The **TotalCost** tab includes an additional column that shows the total cost of each transaction. You can use this information to identify high-value transactions.

- The **BySecurity** tab shows all the inserts, deletes, and update blocks for the large trades. The newest event is shown at the top of the table. The total cost of transactions for each security is displayed: 601300 for IBM and 91950 for SAP.

Note: If the table is empty, check that the publisher connectors for the Trades and Traders windows are set correctly to point to the CSV files.

47 To stop the test, click ■.

The project stops and then unloads from the SAS Event Stream Processing server.

Example: Streaming Analytics with Scoring and Training

This example demonstrates the use of the machine learning algorithm *k-means*, which is often used for cluster analysis in data mining. *K-means* clustering partitions observations into clusters with the nearest mean. The algorithm assigns data points to their nearest cluster centroid. Each cluster centroid is then recomputed based on the average of the cluster's data points.

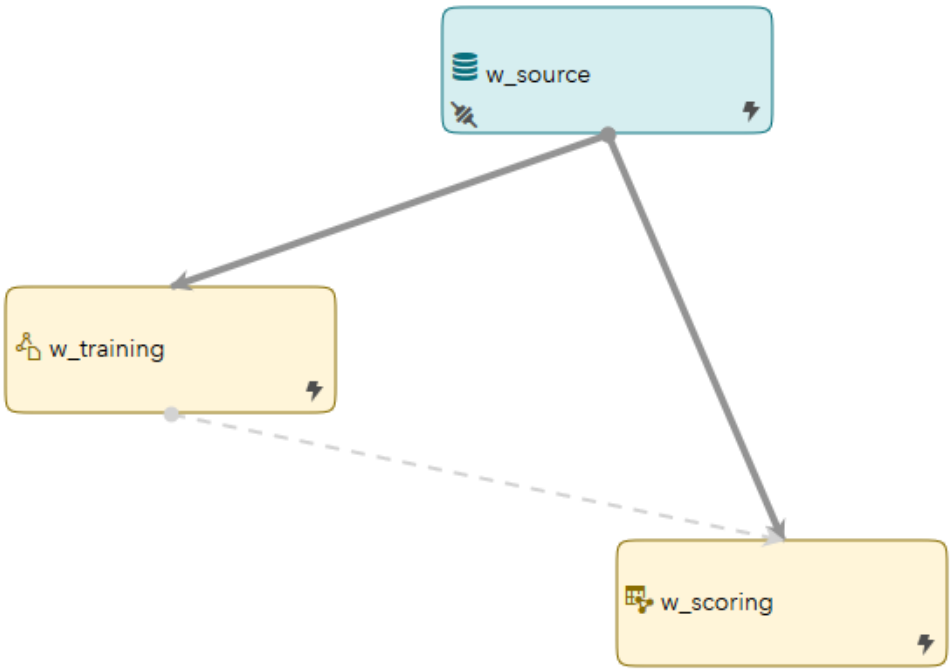
The model contains the following windows:

- A Source window that receives data to be scored
- A Train window that generates and periodically updates the *k-means* model
- A Score window that performs the scoring

In *k-means* clustering, the input event is augmented with a cluster number. This indicates the cluster that the observation falls into.

Note: The CSV data and model XML code that are used in this example are available within your SAS Event Stream Processing installation, typically in the following location: **`/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/analytics_kmeans`**. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

Figure 24 *Diagram of the Streaming Analytics Model with Scoring and Training*



1 On the **Projects** page, click New.

The New Project window appears.

2 In the New Project window, do the following:

a In the **Name** field, enter `Scoring_and_Training`.

b In the **Description** field, enter: `This model demonstrates the use of the K-means machine learning algorithm for clustering.`

c Click **OK**.

If you do not currently have any ESP servers configured, you are prompted to decide whether you want to configure an ESP server now.

Note: It is assumed that you do not have any ESP servers configured. If you already have ESP servers configured, go to step 5.

3 Click **Yes** to configure an ESP server now.

The ESP Server Properties window appears.

4 Configure an ESP server:

a In the **Name** field, enter a name to identify the new ESP server that you want to create.

b In the **Host** field, enter the host name of the new ESP server.

c In the **HTTP port** field, enter the new ESP server's HTTP publish/subscribe port.

d If required, change the setting for the **Authentication** field:

- **None**: This is the default option.

- **Authenticate using an OAuth token**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field appears where you must enter the OAuth token.
- **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields appear where you must enter the user name and password.

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

f Click **OK**.

SAS Event Stream Processing Modeler appears.

5 In the right pane, configure your project's properties:

a Expand **Attributes**.

b Select the **Compress open patterns** check box.

6 Expand **Input Streams** on the **Windows** pane on the left and drag a Source window to the workspace.

The right pane displays the Source window's properties.

7 Enter a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `W_source`.

b In the **Description** field, enter `Source window`.

8 Specify an output schema for the W_source window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values in the rows:

Key	Field Name	Type
Y	id	Int64
N	x_c	Double
N	y_c	Double

d Click **OK**.

9 The W_source window will stream events from a file called input.csv that contains example data. You can find this CSV file in the `analytics_kmeans` folder in the `examples` directory. To add a connector to this CSV file:

a In the right pane, click .

b Click the W_source window.

c Expand **Input Data (Publisher) Connectors** and click .

The Publisher Connectors window appears.

d In the **Name** field, replace the default value with `Source_File`.

e In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/analytics_kmeans/input.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

f In the **Fstype** drop-down list, select **csv**.

g Configure the Source_File connector's properties:

i Click **All properties...**.

The All Properties window appears.

ii Select **true** from the drop-down list in the **Value** field of the **transactional** property.

iii Enter `1` in the **Value** field of the **blocksize** property.

iv Click **OK**.

h Click **OK**.

i Collapse **Input Data (Publisher) Connectors**.

10 Configure an output rule for the W_source window:

a Expand **Output Rules**.

b Select the **Only output "insert" events** check box.

11 Expand **Analytics** on the **Windows** pane on the left and drag a Train window to the workspace.

This window uses the *k-means* algorithm to periodically generate a new clustering model.

The right pane displays the Train window's properties.

12 Specify a name and description for the Train window:

a In the right pane, in the **Name** field, change the default name to `W_training`.

b In the **Description** field, enter `W_training window`.

13 Connect the W_source window to the W_training window with an edge:

a Position the cursor over the anchor point at the bottom of the W_source window so that the anchor point changes from black to white.

b Click the white anchor point, hold the mouse button down, and draw a line to the anchor point in the W_training window.

The W_training window now accepts events from the W_source window.

14 Click the W_training window on the workspace.

15 Expand **Settings**.

16 In the **Algorithm** drop-down list, select **KMEANS**.

17 Expand **KMEANS**:

a Expand **Parameters**.

b In the **nClusters** field, confirm that the default number of clusters is set to `2`.

c In the **initSeed** field, enter `1` to specify the random seed that is used during initialization when each point is assigned to a random cluster.

d In the **dampingFactor** field, confirm that the damping factor's default value for old data points is set to `0.8`.

e In the **fadeOutFactor** field, confirm that the default value for determining whether an existing cluster is fading out is set to `0.05`.

Note: If a cluster weight is smaller than the maximal cluster weight among other clusters multiplied by θ, then this cluster is considered to be fading.

f In the **disturbFactor** field, confirm that the default value for the disturbance factor when splitting a cluster is set to `0.01`.

g In the **nInit** field, confirm that the default value for the number of data events that are used during initialization is set to `50`.

h In the **velocity** field, enter `5` to specify the number of events that arrive at a single timestamp.

i In the **commitInterval** field, confirm that the default value for the number of timestamps to elapse before committing a model to downstream scoring is set to `25`.

j Collapse **Parameters**.

k Expand **Input Map**.

l Click the **inputs** field twice to display the drop-down list and select **x_c** and **y_c** from the drop-down list to specify the variable names to use in clustering.

18 Expand **Analytics** on the **Windows** pane on the left and drag a Score window to the workspace.

The right pane displays the Score window's properties.

This window scores incoming events.

19 In the right pane, in the **Name** field, change the default name to `W_scoring`:

a In the right pane, in the **Name** field, change the default name to `W_scoring`.

b In the **Description** field, enter `Score window`.

20 Specify a schema for the W_scoring window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values in the rows:

Key	Field Name	Type
Y	id	Int64

Key	Field Name	Type
N	x_c	Double
N	y_c	Double
N	seg	Int32
N	min_dist	Double
N	model_id	Int64

21 Click **OK**.

22 Connect the W_source window to the W_scoring window with an edge.

The W_scoring window can now score events that originate from the W_source window.

23 Configure the settings for the W_scoring window:

a Click the W_scoring window on the workspace.

b Expand **Settings** if it is not already expanded.

c Specify an algorithm to use to score incoming events:

i In the **Configured algorithms** field, click .

The Configured Algorithms window appears.

ii Select the **KMEANS** check box.

iii Click **OK**.

d Expand **KMEANS**.

e Configure an input map:

i Expand **Input Map**.

ii Click the **inputs** field twice to display the drop-down list and select **x_c** and **y_c** from the drop-down list to specify the variable names to use in clustering.

iii Collapse **Input Map**.

f Configure an output map:

i Expand **Output Map**.

ii Specify the output variable name in the output schema that stores the cluster label. In the **labelOut** row, click the **Name** field twice to display the drop-down list and select **seg**.

iii Specify the output variable name in the output schema that stores the distance to the nearest cluster. In the **minDistanceOut** row, click the **Name** field twice to display the drop-down list and select **min_dist**.

iv Specify the output variable name in the output schema that stores the ID of the model from which the score is computed. In the **modelIdOut** row, click the **Name** field twice to display the drop-down list and select **model_id**.

24 Connect the W_training window to the W_scoring window with an edge.

25 Configure the project's continuous query:

a Click .

b In the right pane, in the **Name** field, change the default name to `scoretrain_cq`.

c Expand **Debugging**.

d In the **Enable trace logging for this query** field, select **W_scoring** and **W_training**.

26 Click .

27 Click ▸ Test.

A new page called **Test: Scoring_and_Training** appears.

28 In the **Test Server** drop-down list, select the ESP server on which you want to test the model.

29 Click ▶.

The results for each window appear in separate tabs:

- The **w_source** tab displays events to be scored
- The **w_training** tab displays the generated clustering model using the *k-means* algorithm
- The **w_scoring** tab displays the scored events

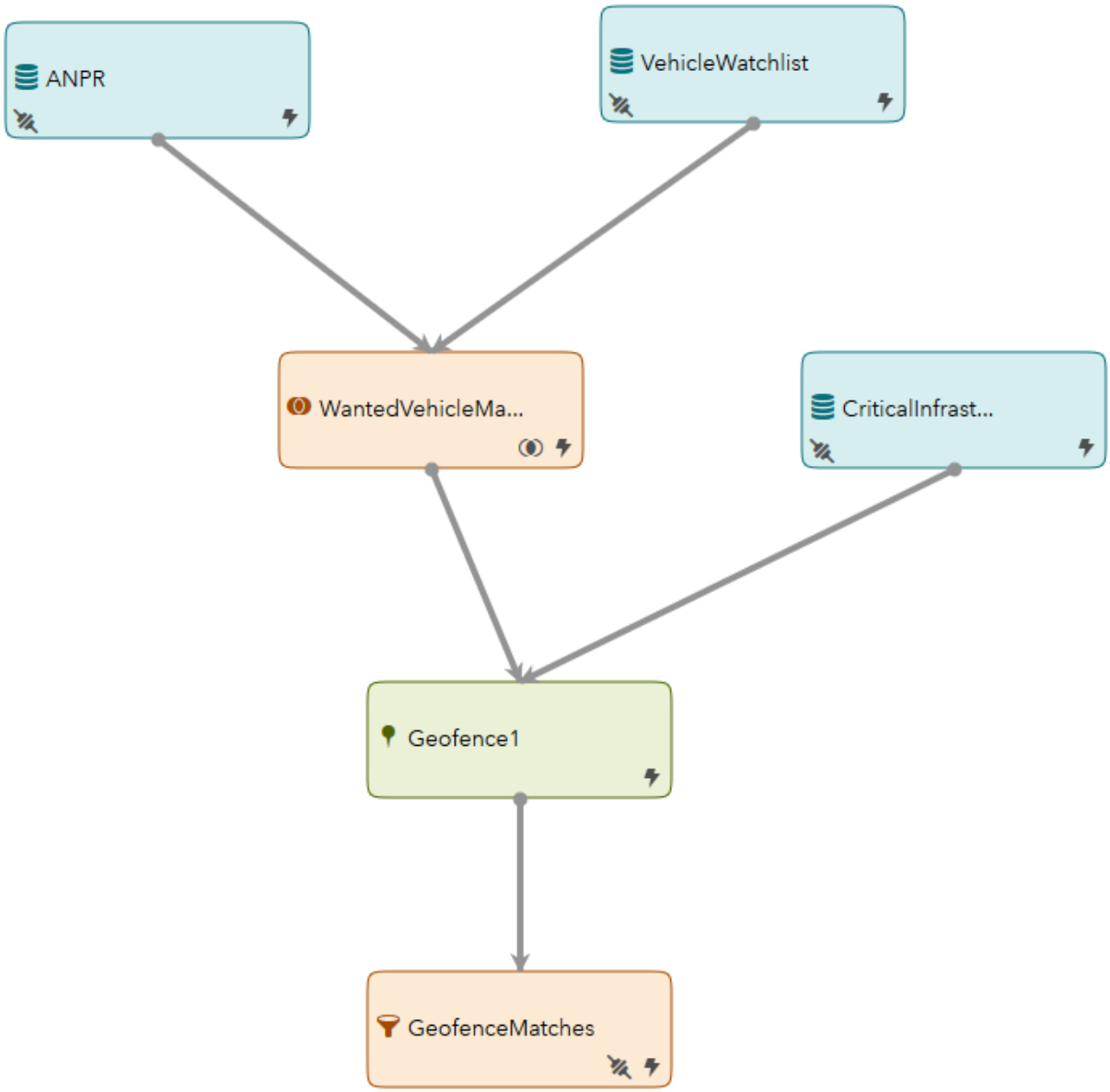
30 To stop the test, click ■.

Example: Geofence

This example creates a model that displays a list of wanted vehicles found in close proximity of critical infrastructure sites. The model performs the following actions:

- streams a list of vehicles, including vehicle locations
- streams a list of vehicles that are included on a vehicle watch list
- streams a list of critical infrastructure sites, including site locations
- processes the list of vehicles and attempts to match any wanted vehicles that are in close proximity to critical infrastructure sites
- produces a list of wanted vehicles found in close proximity to critical infrastructure sites

Figure 25 *Diagram of the Geofence Model*



Note: The CSV data and model XML code that are used in this example are available within your SAS Event Stream Processing installation, typically in the following location: `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/geofence2_xml`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

1 On the **Projects** page, click New.

   The New Project window appears.

2 In the New Project window, do the following:

   a In the **Project name** field, enter `geofence_demo`.

   b In the **Description** field, enter a description. Here is an example: `This model can be used to identify wanted vehicles found in close proximity to critical infrastructure sites.`

c Click **OK**.

If you do not currently have any ESP servers configured, you are prompted to decide whether you want to configure an ESP server now.

Note: It is assumed that you do not have any ESP servers configured. If you already have ESP servers configured, skip to step 5.

3 Click **Yes** to configure an ESP server now.

The ESP Server Properties window appears.

4 Configure an ESP server:

a In the **Name** field, enter a name to identify the new test server to create.

b In the **Host** field, enter the host name or the IP address of the test server.

c In the **HTTP port** field, enter the test server's HTTP publish/subscribe port.

d If required, change the setting for the **Authentication** field:

- **None**: This is the default option.
- **Authenticate using an OAuth token**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field appears where you must enter the OAuth token.
- **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields appear where you must enter the user name and password.

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

f Click **OK**.

Your project is created with a set of default properties.

5 Expand **Input Streams** on the **Windows** pane on the left and drag a Source window to the workspace.

The right pane displays the Source window's properties.

6 Specify a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `ANPR`.

b In the **Description** field, enter `ANPR Source window`.

7 Configure the ANPR window to accept only "Insert" events and to automatically generate the key field:

a Expand **State and Event Type**.

b Select the **Accept only "Insert" events** check box.

c Select the **Automatically generate the key field** check box.

8 Specify an output schema for the ANPR window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values:

Key	Field Name	Type
N	vrm	String
N	lat	Double
N	long	Double
N	date	Timestamp
Y	pkey	String

d Click **OK**.

9 The ANPR window streams a list of vehicles from a file called anpr.csv that contains example data. You can find this CSV file in the `geofence2_xml` folder in the `examples` directory. To add a connector to this CSV file:

a In the right pane, click .

b Expand **Input Data (Publisher) Connectors** and click .

The Connector Configuration window appears.

c In the **Name** field, replace the default value with `anpr_csv_read`.

d In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/geofence2_xml/anpr.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

e In the **Fstype** drop-down list, select **csv**.

f Configure the anpr_csv_read connector's properties:

i Click **All properties**.

The All Properties window appears.

ii Enter `%Y-%m-%d %H:%M:%S` in the **Value** field of the **dateformat** property.

iii Enter `1` in the **Value** field of the **header** property.

iv Select **true** from the drop-down list in the **Value** field of the **ignorecsvparseerrors** property.

v Select **true** from the drop-down list in the **Value** field of the **noautogenfield** property.

vi Click **OK**.

g Click **OK**.

10 Expand **Input Streams** on the **Windows** pane on the left and drag another Source window to the workspace.

The right pane displays the Source window's properties.

11 Specify a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `VehicleWatchList`.

b In the **Description** field, enter `VehicleWatchList Source window`.

12 Specify an output schema for the VehicleWatchList window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. Enter the following values:

Key	Field Name	Type
Y	vrm	String

d Click **OK**.

13 The VehicleWatchList window streams a list of wanted vehicles from a file called wantedvehicle.csv that contains example data. You can find this CSV file in the `geofence2_xml` folder in the `examples` directory. To add a connector to this CSV file:

a In the right pane, click .

b Expand **Input Data (Publisher) Connectors** and click .

The Connector Configuration window appears.

c In the **Name** field, replace the default value with `vehicle_watchlist`.

d In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/geofence2_xml/wantedvehicle.csv`. Replace <*release*> with the release number in your SAS Event Stream Processing installation directory path.

e In the **Fstype** drop-down list, select **csv**.

f Configure the vehicle_watchlist connector's properties:

i Click **All properties**.

The All Properties window appears.

ii Enter `1` in the **Value** field of the **header** property.

iii Click **OK**.

g Click **OK**.

14 Expand **Transformations** on the **Windows** pane on the left and drag a Join window to the workspace.

The right pane displays the Join window's properties.

15 Specify a name and description for the Join window:

a In the right pane, in the **Name** field, change the default name to `WantedVehicleMatch`.

b In the **Description** field, enter `WantedVehicleMatch Join window`.

16 Connect the ANPR window to the WantedVehicleMatch window with an edge:

a Position the cursor over the anchor point at the bottom of the ANPR window so that the anchor point changes from black to white.

b Click the white anchor point, hold the left mouse button down, and draw a line to the anchor point in the WantedVehicleMatch window.

The WantedVehicleMatch window now accepts values from the ANPR window.

17 Connect the VehicleWatchlist window to the WantedVehicleMatch window with an edge.

The WantedVehicleMatch window now accepts values from the VehicleWatchlist window.

Note: Each window in your model displays specific icons that represent window properties. For example, if a Source window contains a publisher connector, the window displays the corresponding publisher connector icon. For more information about window icons, see Window Icons on page 22. The WantedVehicleMatch window displays an error icon indicating that an invalid join type has been set. The occurrence of this error is expected behavior and will be resolved later when you set a valid join type.

18 Click the WantedVehicleMatch window in the workspace.

The right pane displays the Join window's properties.

19 Configure the calculation method for the WantedVehicleMatch window's output fields:

a In the right pane, expand **Settings**.

b Inspect the **Left window** and **Right window** fields. Notice that the ANPR window is regarded as the left window and the VehicleWatchList window is regarded as the right window. This is due to the order in which you added the edges.

c In the **Output field calculation method** field, select **Select fields** from the drop-down list.

d Collapse **Settings**.

20 Configure the WantedVehicleMatch window's join criteria:

a Expand **Join Criteria**.

b In the **Join Type** drop-down list, select **Inner**.

c Collapse **Join Criteria**.

21 Configure the WantedVehicleMatch window's join conditions:

a Expand **Join Conditions**.

b In the **Join Conditions** field, click to add a join condition.

c Click the cell in the Left: ANPR column twice, and select **vrm** from the drop-down list.

d Click the cell in the Right: VehicleWatchList column twice, and select **vrm** from the drop-down list.

22 Specify an output schema for the WantedVehicleMatch window:

a In the right pane, click .

b Click to add a row to the schema table.

The Edit Output Schema window appears. Use this window to configure the fields as shown in the following table. The schema fields required have already been defined previously. Click to open the Copy Fields from Input Schema window. Select the following schema fields and click **OK**.

Field Name	Window	Field	Type
vrm	ANPR	vrm	String
lat	ANPR	lat	Double
long	ANPR	long	Double
date	ANPR	date	TimeStamp

The Edit Output Schema window displays the fields that you selected.

c Click **OK**.

23 Click .

24 Expand **Input Streams** on the **Windows** pane on the left and drag another Source window to the workspace.

The right pane displays the Source window's properties.

25 Specify a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `CriticalInfrastructure`.

b In the **Description** field, enter `CriticalInfrastructure Source window`.

26 Specify an output schema for the CriticalInfrastructure window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row. Enter the following values:

Key	Field Name	Type
Y	name	String
N	lat	Double
N	long	Double
N	location	String
N	county	String
N	region	String
N	type	String

Key	Field Name	Type
N	capacity	String
N	opened	String
N	closed	String
N	demolished	String
N	notes	String

d Click **OK**.

27 The CriticalInfrastucture window will stream a list of sites that contain critical infrastructure from a file called infrastructure.csv that contains example data. You can find this CSV file in the `geofence2_xml` folder in the `examples` directory. To add a connector to this CSV file:

a In the right pane, click .

b Expand **Input Data (Publisher) Connectors**.

c Click .

The Connector Configuration window appears.

d In the **Name** field, replace the default value with `infrastructure_csv_reader`.

e In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/geofence2_xml/infrastructure.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

f In the **Fstype** drop-down list, select **csv**.

g Configure the infrastructure_csv_reader connector's properties:

i Click **All properties**.

The All Properties window appears.

ii Enter `1` in the **Value** field of the **header** property.

iii Select **true** from the drop-down list in the **Value** field of the **ignorecsvparseerrors** property.

iv Click **OK**.

h Click **OK**.

28 Expand **Utilities** on the **Windows** pane on the left and drag a Geofence window to the workspace.

The right pane displays the Geofence window's properties.

29 Specify a name and description for the Geofence window:

a In the right pane, in the **Name** field, change the default name to `Geofence`.

b In the **Description** field, enter `Geofence window`.

30 Connect the WantedVehicleMatch window to the Geofence window with an edge.

31 Connect the CriticalInfrastructure window to the Geofence window with an edge.

32 Click the Geofence window in the workspace.

The right pane displays the Geofence window's properties.

33 Configure the Geofence window's positional settings:

a Expand **Positions**.

b In the **Longitude** field, select **long** from the drop-down list.

c In the **Latitude** field, select **lat** from the drop-down list.

d In the **Default lookup distance (meters)** field, enter `100`.

e Collapse **Positions**.

34 Configure the Geofence window's geometric settings:

a Expand **Geometries**.

b In the table, in the **Longitude** row, select **long** from the **Field** drop-down list.

c In the table, in the **Latitude** row, select **lat** from the **Field** drop-down list.

d In the **Default radius (meters)** field, enter `100`.

e Collapse **Geometries**.

35 Configure the Geofence window's geofence algorithm properties:

a Expand **Geofence Algorithm Properties**.

b Select the **Record invalid geometries in the standard output log** check box.

c Collapse **Geofence Algorithm Properties**.

36 Configure the Geofence window's output map properties:

a Expand **Output Map**.

b In the **Geometry ID** field, enter `geoid`.

c In the **Event number** field, enter `eventnum`.

d Collapse **Output Map**.

37 Expand **Transformations** on the **Windows** pane on the left and drag a Filter window to the workspace.

The right pane displays the Filter window's properties.

38 Specify a name and description for the Filter window:

a In the **Name** field, change the default name to `GeofenceMatches`.

b In the **Description** field, enter `GeofenceMatches Filter window`.

39 Configure a subscribe connector for the GeofenceMatches window:

a Expand **Subscriber Connectors**.

b Click .

The Connector Configuration window appears.

c In the **Name** field, enter `sub`.

d Select the **Snapshot** check box.

e In the **Fsname** field, enter the path to the output file: **result.out**. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/geofence2_xml/result.out`. Replace <*release*> with the release number in your SAS Event Stream Processing installation directory path.

f In the **Fstype** drop-down list, select **csv**.

g Click **OK**.

40 Specify a filter expression for the GeofenceMatches window:

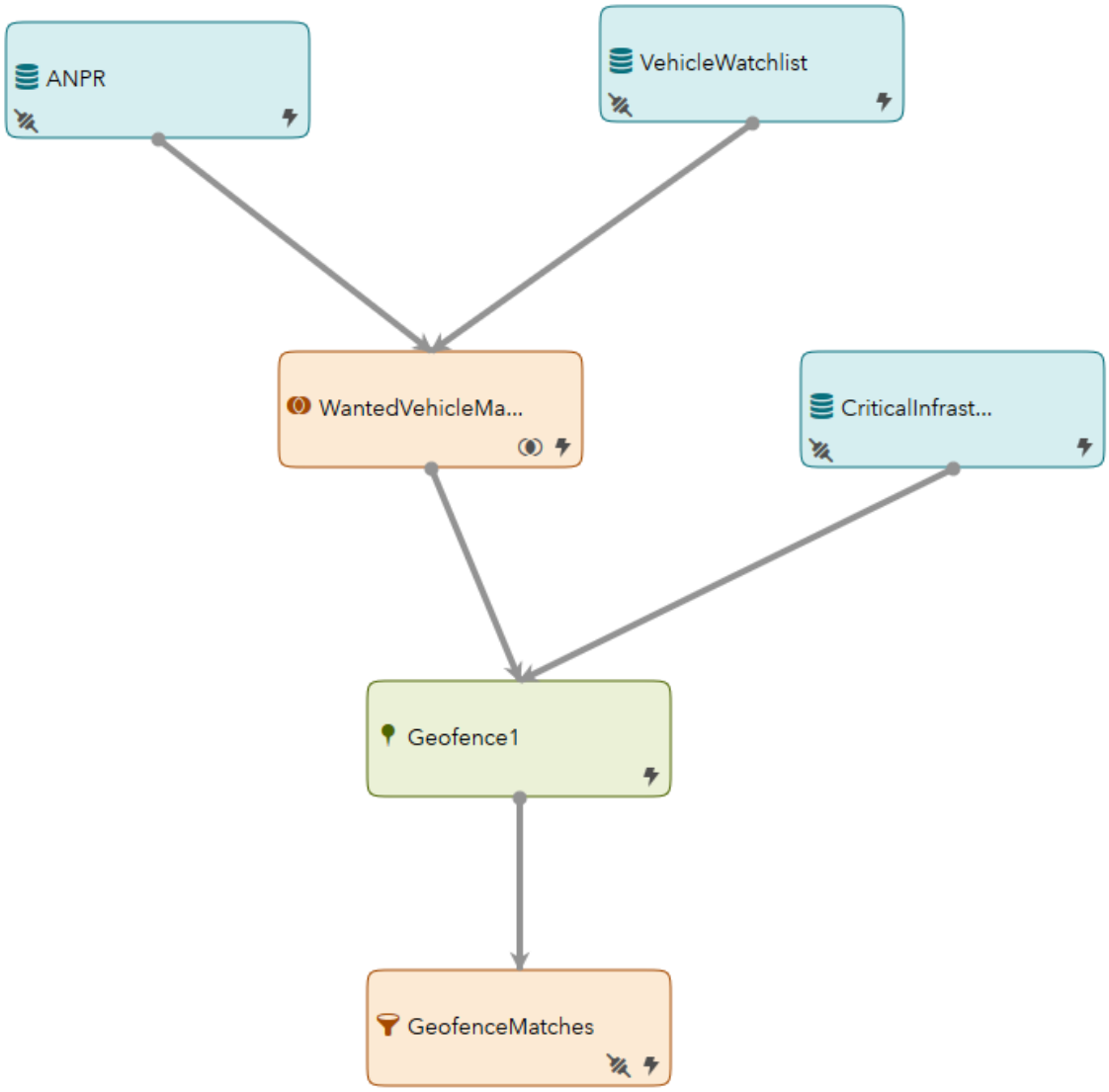
a Expand **Filter**.

b In the **Expression** field, enter `geoid != ''`

c Collapse **Filter**.

41 Connect the Geofence window to the GeofenceMatches window with an edge.

Figure 26 *The Geofence Window Connected to the GeofenceMatches Window*



42 Configure your model's connector orchestration:

a Click .

b In the right pane, expand **Connector Orchestration**.

c Click below the **Connector groups** label.

The Connector groups window appears.

d In the **Name** field, enter `sub1`.

e Click below the **Connectors** label.

f In the Connector column, click the newly created row and select **cq1/GeofenceMatches/sub** from the drop-down list.

g In the Target state column, select **Running** from the drop-down list.

h Click **OK**.

i Click below the **Connector groups** label.

The Connector groups window appears.

j In the **Name** field, enter `pub1`.

k Click below the **Connectors** label.

l In the Connector column, click the newly created row and select **cq1/ANPR/anpr_csv_read** from the drop-down list.

m In the Target state column, confirm that **Finished** is selected from the drop-down list.

n Click **OK**.

o Click below the **Connector groups** label.

The Connector groups window appears.

p In the **Name** field, enter `pub2`.

q Click below the **Connectors** label.

r In the Connector column, click the newly created row and select **cq1/CriticalInfrastructure/infrastructure_csv_reader** from the drop-down list.

s In the Target state column, confirm that **Finished** is selected from the drop-down list.

t Click **OK**.

u Click below the **Connector groups** label.

The Connector groups window appears.

v In the **Name** field, enter `pub3`.

w Click below the **Connectors** label.

x In the Connector column, click the newly created row and select **cq1/VehicleWatchlist/vehicle_watchlist** from the drop-down list.

y In the Target state column, confirm that **Finished** is selected from the drop-down list.

z Click **OK**.

aa Configure the dependency rules. Click below the **Dependency rules** label. After you add a row, click again to add the next row.

Enter the following values in the rows:

Row	Controlling Group	Dependent Group
1	sub1	pub1
2	pub2	pub1
3	pub2	pub3

ab In the right pane, click .

The XML Editor appears.

ac Locate the following line in the XML code: `<edge source="sub1" target="pub1"/>`

ad Amend this line to the following: `<edge source="sub1" target="pub1 pub2 pub3"/>`

43 Configure your model's threading level:

a In the right pane, click .

b In the right pane, expand **Attributes**.

c In the **Threads** field, enter `8`.

44 The model is now complete. Click to save your model.

45 Click Test.

A new page called **Test: geofence_demo** appears.

46 In the **ESP Server** drop-down list, select the ESP server on which you want to test the model.

47 Click .

The results for each window appear on separate tabs:

- The **ANPR** tab lists all vehicles within close proximity of critical infrastructure sites
- The **VehicleWatchlist** tab lists all vehicles on the vehicle watch list
- The **WantedVehicleMatch** tab lists the wanted vehicles found within close proximity of critical infrastructure sites
- The **Geofence** tab lists the geofencing information that relates to the matched vehicles
- The **CriticalInfrastructure** tab lists sites that contain critical infrastructure
- The **GeofenceMatches** tab shows any wanted vehicles found within close proximity of critical infrastructure sites

Note: If the table is empty, check that the publisher connectors for the ANPR, VehicleWatchList, and CriticalInfrastructure windows are set correctly to point to the CSV files.

48 To stop the test, click .

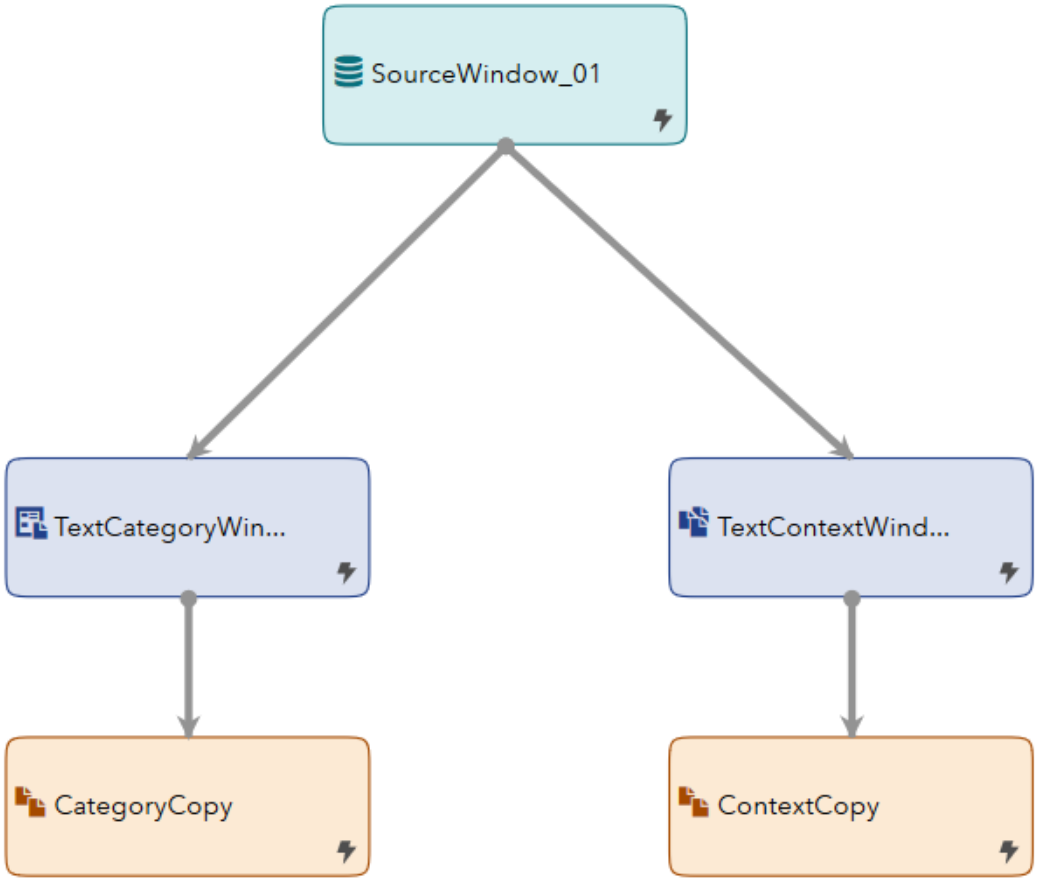
The project stops and then unloads from the ESP server.

Example: Working with Text Analytics

This example demonstrates how text can be analyzed, categorized, and sorted in context using SAS Event Stream Processing Studio. Text from a CSV file is processed and then cross-referenced against a set of text rules. Using the rules defined in an MCO file, the text is analyzed and categorized. The model also generates contextual information that is based on a set of rules defined in a LITI file.

Note: To successfully complete this example, you must have access to a SAS Contextual Analysis license.

Figure 27 *Diagram of the Text Analytics Model*



Note: The data files and model XML code that are used in this example are available from the following location: **`ftp://ftp.sas.com/techsup/download/esp/esp-5.1.TextAnalyticsExample.zip`**.

1 Download the esp-5.1.TextAnalyticsExample.zip file from **`ftp://ftp.sas.com/techsup/download/esp/esp-5.1.TextAnalyticsExample.zip`**.

2 Navigate to the esp-5.1.TextAnalyticsExample.zip file that you downloaded and extract its contents.

   Note: It is recommended that you extract the files into a folder called **`text_Analytics`**. Note the location that you extracted the files to.

3 On the **Projects** page, click New.

   The New Project window appears.

4 In the New Project window, do the following:

   a In the **Project name** field, enter **`text_analytics_demo`**.

   b In the **Description** field, enter a description. Here is an example: **`This model uses predefined rules to analyze, categorize, and sort information in context.`**

   c Click **OK**.

   If you do not currently have any ESP servers configured, you are prompted to decide whether you want to configure an ESP server now.

Note: It is assumed that you do not have any ESP servers configured. If you already have ESP servers configured, go to step 7.

5 Click **Yes** to configure an ESP server now.

The ESP Server Properties window appears.

6 Configure an ESP server:

a In the **Name** field, enter a name to identify the new ESP server that you want to create.

b In the **Host** field, enter the host name or IP address of the new ESP server.

c In the **HTTP port** field, enter the new ESP server's HTTP publish/subscribe port.

d If required, change the setting for the **Authentication** field:

- **None**: This is the default option.
- **Authenticate using an OAuth token**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field appears where you must enter the OAuth token.
- **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields appear where you must enter the user name and password.

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

f Click **OK**.

Your project is created with a set of default properties.

7 Configure your model's threading level:

a In the right pane, confirm that the project's properties appear. If they do not appear, click .

b Expand **Attributes**.

c In the **Threads** field, enter `4`.

8 Expand **Input Streams** on the **Windows** pane on the left and drag a Source window to the workspace.

The right pane displays the Source window's properties.

9 Specify a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `SourceWindow_01`.

b In the **Description** field, enter `This window processes the event stream enabling the model's derived windows to analyze the text data`.

10 Specify an output schema for the SourceWindow_01 window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values:

Key	Field Name	Type
Y	ID	Int32
N	tstamp	Date
N	msg	String

d Click **OK**.

11 The SourceWindow_01 window will stream a list of vehicles from a file called textanalytics.csv that contains example data. To add a connector to this CSV file:

a In the right pane, click .

b Expand **Input Data (Publisher) Connectors** and click .

The Connector Configuration window appears.

c In the **Name** field, replace the default value with `DataIn`.

d In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/text_Analytics/textanalytics.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

e In the **Fstype** drop-down list, select **csv**.

f Configure the DataIn connector's properties:

i Click **All properties**.

The All Properties — DataIn window appears.

ii Enter `%Y-%m-%d %H:%M:%S` in the **Value** field of the **dateformat** property.

iii Enter `100000` in the **Value** field of the **repeatcount** property.

iv Click **OK**.

g Click **OK**.

12 Expand **Text Analytics** on the **Windows** pane on the left and drag a Text Category window to the workspace.

The right pane displays the Text Category window's properties.

13 Specify a name and description for the Text Category window:

a In the right pane, in the **Name** field, change the default name to `TextCategoryWindow`.

b In the **Description** field, enter `This window processes text from a CSV file. The text is cross-referenced against a set of text rules in an MCO file. Using these rules, the text is analyzed and categorized.`

14 Connect the SourceWindow_01 window to the TextCategoryWindow window with an edge:

a Position the cursor over the anchor point at the bottom of the SourceWindow_01 window so that the anchor point color changes from black to white.

b Click the white anchor point, hold the mouse button down, and draw a line to the anchor point in the TextCategoryWindow window.

The TextCategoryWindow window now accepts values from the SourceWindow_01 window.

15 Click the TextCategoryWindow window on the workspace.

16 Configure the TextCategoryWindow window's state and index:

a In the right pane, expand **State**.

b In the **Window state and index** field, select **Stateless (pi_EMPTY)** from the drop-down list.

c Collapse **State**.

17 Configure the TextCategoryWindow window's text categorization properties:

a Expand **Text Category** if the section is not expanded by default.

b In the **Text field** field, confirm that **msg** is selected by default.

c In the **Categorization binary (MCO) file full path** field, enter the path to your IPTC.mco file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/text_Analytics/IPTC.mco`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

18 Expand **Transformations** on the **Windows** pane on the left and drag a Copy window to the workspace.

The right pane displays the Copy window's properties.

19 Specify a name and description for the Copy window:

a In the right pane, in the **Name** field, change the default name to `CategoryCopy`.

b In the **Description** field, enter `This window retains categorized events for five minutes.`

20 Connect the TextCategoryWindow window to the CategoryCopy window with an edge.

21 Configure the CategoryCopy window's state and index:

a Click the CategoryCopy window on the workspace.

b Expand **State**.

c In the **Window state and index** field, select **Stateful (pi_RBTREE)** from the drop-down list.

22 Configure the CategoryCopy window's retention properties:

a Expand **Retention** if the section is not already expanded by default.

b Select the **Limit event retention** check box.

c Confirm that the **Type** field is set to **By time, sliding**.

d In the **Time limit** field, enter `5` and select **Minutes** from the drop-down list.

e Collapse **Retention**.

23 Expand **Text Analytics** on the **Windows** pane on the left and drag a Text Context window to the workspace.

The right pane displays the Text Context window's properties.

24 Specify a name and description for the Text Context window:

a In the right pane, in the **Name** field, change the default name to `TextContextWindow`.

b In the **Description** field, enter `This window outputs information in context based on a set of rules defined in a LITI file.`

25 Connect the SourceWindow_01 window to the TextContextWindow window with an edge.

26 Configure the TextContextWindow window's text categorization properties:

a Click the TextContextWindow on the workspace.

b Expand **Text Context** if the section is not expanded by default.

c In the **Text field** field, confirm that **msg** is selected by default.

d In the **LITI files full paths** field, click .

A new row is created in the table.

e Click the new row and enter the path of your LITI file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/text_Analytics/citng_en-ne.li`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

f Select the **Generate a null event if no match is found** check box.

27 Expand **Transformations** on the **Windows** pane on the left and drag a Copy window to the workspace.

The right pane displays the Copy window's properties.

28 Specify a name and description for the Copy window:

a In the right pane, in the **Name** field, change the default name to `ContextCopy`.

b In the **Description** field, enter `This window retains events that have been sorted in context for five minutes.`

29 Connect the TextContextWindow window to the ContextCopy window with an edge.

30 Configure the ContextCopy window's retention properties:

a Click the ContextCopy window in the workspace.

b Expand **Retention**.

c Select the **Limit event retention** check box.

d Confirm that the **Type** field is set to **By time, sliding**.

e In the **Time limit** field, enter `5` and select **Minutes** from the drop-down list.

31 Configure the project's continuous query:

a Click .

b In the **Name** field, enter `contquery_01`.

c Expand **Debugging**.

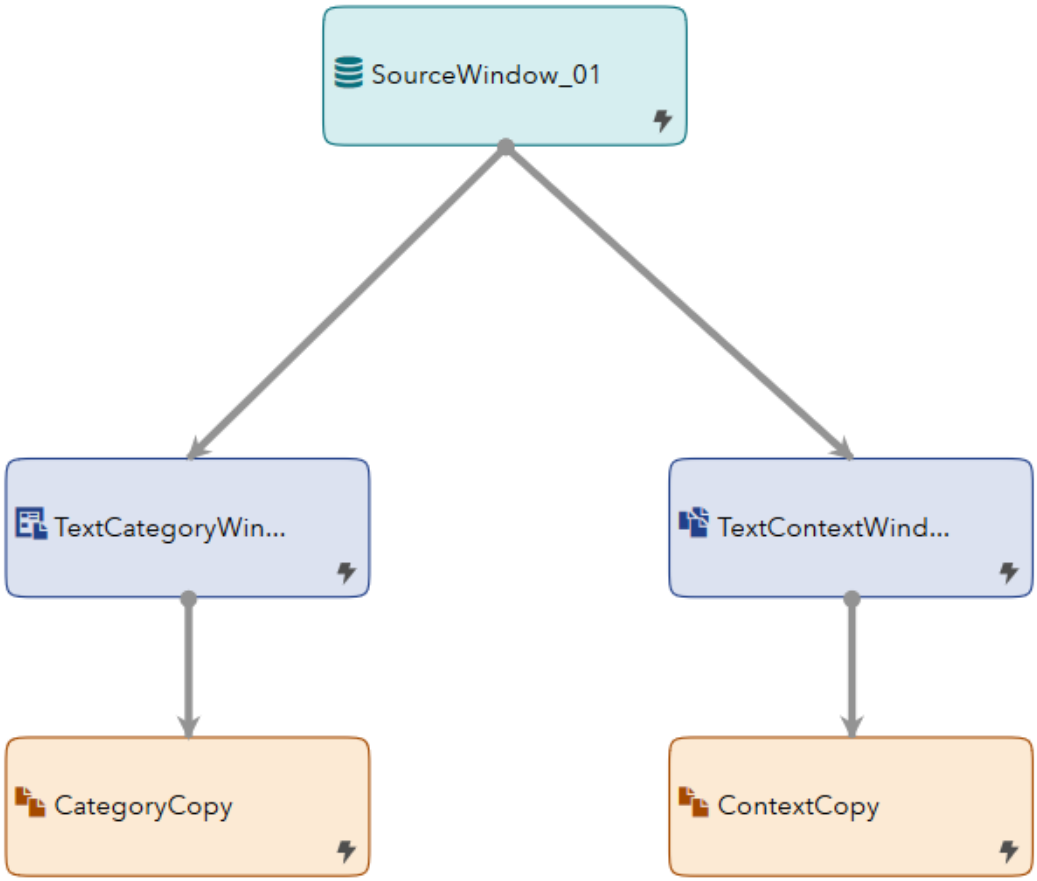
d Select the **Log warnings for long computation times** check box.

e In the **Threshold (µs)** field, enter `100`.

f In the **Trace in server log** field, select **TextCategoryWindow** and **TextContextWindow**.

32 Click .

Figure 28 Diagram of the Text Analytics Model



33 Click ▸ Test.

A new page called **Test: text_analytics_demo** appears.

34 In the **Test Server** drop-down list, select the ESP server on which you want to test the model.

35 Click ▶.

The results for each window appear on separate tabs:

- The **CategoryCopy** tab displays categorized events with a retention policy of 5 minutes
- The **ContextCopy** tab displays contextual events with a retention policy of 5 minutes
- The **TextContextWindow** tab displays information in contextual format
- The **SourceWindow_01** tab displays the processed event stream
- The **TextCategoryWindow** tab displays information in categorized format

36 To stop the test, click ■.

The project stops and then unloads from the ESP server.

Example: Split Generated Events across Output Slots

This model enables you to send generated stock market events across a set of output slots. It contains a Source window, a Compute window, and three Copy windows. The Compute window uses an expression to determine what output slot or slots should be used for a newly generated stock market event. The Copy windows connect to the Compute window using different output slots.

Filtering events using window splitters with only one output slot can be more efficient than using multiple Filter windows. This is because the filtering is performed at the window splitter only, rather than at multiple times for each filter. For example, performing an alpha-split across a set of trades results in less data movement and data processing than performing an alpha-split across multiple Filter windows.

1 On the **Projects** page, click New.

2 In the New Project window, do the following:

   a In the **Project name** field, enter `modelingSplitterExp`.

   b Click **OK**.

   If no ESP servers are currently configured, you are prompted to decide whether you want to configure an ESP server now.

   Note: It is assumed that no ESP servers are currently configured. If some already are, go to step 5.

3 Click **Yes** to configure an ESP server now.

   The ESP Server Properties window appears.

4 Configure an ESP server:

   a In the **Name** field, enter a name to identify the new ESP server that you want to create.

   b In the **Host** field, enter the host name or the IP address of the new ESP server.

   c In the **HTTP port** field, enter the new ESP server's HTTP publish/subscribe port.

   d If required, change the setting for the **Authentication** field:

      - **None**: This is the default option.
      - **Authenticate using an OAuth token**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, an additional field appears where you must enter the OAuth token.
      - **Authenticate using a user name and password**: This option is relevant only if the ESP server is configured to require authorization. If you select this option, additional fields appear where you must enter the user name and password.

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

   f Click **OK**.

   Your project is created with a set of default properties.

5 Configure the project's continuous query:

a Click .

b In the **Name** field, enter `cq_01`.

6 Expand **Input Streams** on the **Windows** pane on the left and drag a Source window to the workspace.

The right pane displays the Source window's properties.

7 Specify a name and description for the Source window:

a In the right pane, in the **Name** field, change the default name to `src_win`.

b In the **Description** field, enter `This window receives an event stream of stock market trades.`

8 Configure the src_win window's state and event type:

a In the right pane, expand **State and Event Type**.

b Select **Stateful (pi_RBTREE)** from the **Window state and index** drop-down list.

9 Specify an output schema for the src_win window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values:

Key	Field Name	Type
Y	ID	Int32
N	symbol	String
N	price	Double

d Click **OK**.

10 The src_win window streams a list of vehicles from a file called input.csv that contains example data. To add a connector to this CSV file:

a Click .

b Expand **Input Data (Publisher) Connectors** and click .

The Connector Configuration window appears.

c In the **Name** field, replace the default value with `pub`.

d In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/slot_exp_xml/input.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

e In the **Fstype** drop-down list, select **csv**.

f Configure the pub connector's properties:

i Click **All properties**.

The All Properties window appears.

ii Select `true` in the **Value** field of the **transactional** property.

iii Enter `1` in the **Value** field of the **blocksize** property.

iv Click **OK**.

g Click **OK**.

11 Expand **Transformations** on the **Windows** pane on the left and drag a Compute window to the workspace.

The right pane displays the Compute window's properties.

12 Specify a name and description for the Compute window:

a In the right pane, in the **Name** field, change the default name to `compute_win`.

b In the **Description** field, enter `This window uses expressions to calculate each field. The first field uses the expression to calculate the count. The last two fields are just passing through what is in the input window.`

13 Connect the src_win window to the compute_win window with an edge:

a Position the cursor over the anchor point at the bottom of the src_win window so that the anchor point changes from black to white.

b Click the white anchor point, hold the left mouse button down, and draw a line to the anchor point in the compute_win window.

The compute_win window now accepts values from the src_win window.

14 Specify an output schema for the compute_win window:

a In the right pane, click .

b Click .

The Output Schema window appears.

c Click to add a row to the schema table. After you add a row, click again to add the next row.

Enter the following values:

Key	Field Name	Type	Expression
Y	ID	Int32	not applicable
N	counter	Int32	counter=counter+1 return counter
N	symbol	String	symbol
N	price	Double	price

d Click **OK**.

15 Configure the compute_win window's split method:

a Click .

b Expand **Advanced**.

c Select the **Split output** check box.

d Confirm that **Expression** is selected in the **Split Method** field.

e Enter `ID%2` in the **Expression** field.

f Collapse **Advanced**.

16 Configure the compute_win window's compute settings:

a Expand **Compute Settings**.

b Confirm that **Expressions** is selected in the **Compute method** field.

c Select the **Include engine initialization expression** check box.

d Select **Int32** in the **Return type** drop-down list.

e In the **Expression** field, enter:

```
integer counter counter=0
```

f Collapse **Compute Settings**.

17 Configure a subscriber connector for the compute_win window:

a Expand **Subscriber Connectors** and click .

The Connector Configuration window appears.

b In the **Name** field, replace the default value with **`sub`**.

c In the **Fsname** field, enter the path to the CSV file. For example, you might enter **`/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/slot_exp_xml/compute.csv`**. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

d In the **Fstype** drop-down list, select **csv**.

e Click **OK**.

18 Expand **Transformations** on the **Windows** pane on the left and drag a Copy window to the workspace.

The right pane displays the Copy window's properties.

19 Specify a name and description for the Copy window:

a In the right pane, in the **Name** field, change the default name to **`cw_01`**.

b In the **Description** field, enter **`This Copy window connects to the Compute window using the output slot 0.`**

20 Configure a subscriber connector for the cw_01 window:

a Expand **Subscriber Connectors** and click .

The Connector Configuration window appears.

b In the **Name** field, replace the default value with `sub1`.

c In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/slot_exp_xml/cw_01.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

d Select the **Snapshot** check box.

e In the **Fstype** drop-down list, select **csv**.

f Click **OK**.

21 Connect the compute_win window to the cw_01 window with an edge:

a Position the cursor over the anchor point at the bottom of the compute_win window so that the anchor point changes from black to white.

b Click the white anchor point, hold the left mouse button down, and draw a line to the anchor point in the compute_win window.

The cw_01 window now accepts values from the compute_win window.

22 Assign a slot number to the edge:

a Click the edge that connects the compute_win window with the cw_01 window.

b In the right pane, enter `0` in the **Slot** field.

23 Expand **Transformations** on the **Windows** pane on the left and drag a Copy window to the workspace.

The right pane displays the Copy window's properties.

24 Specify a name and description for the Copy window:

a In the right pane, in the **Name** field, change the default name to `cw_02`.

b In the **Description** field, enter `This Copy window connects to the Compute window using the output slot 1.`

25 Configure a subscriber connector for the cw_02 window:

a Expand **Subscriber Connectors** and click .

The Connector Configuration window appears.

b In the **Name** field, replace the default value with `sub2`.

c In the **Fsname** field, enter the path to the CSV file. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/slot_exp_xml/cw_02.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

d Select the **Snapshot** check box.

e In the **Fstype** drop-down list, select **csv**.

f Click **OK**.

26 Connect the compute_win window to the cw_02 window with an edge:

a Position the cursor over the anchor point at the bottom of the compute_win window so that the anchor point changes from black to white.

b Click the white anchor point, hold the left mouse button down, and draw a line to the anchor point in the compute_win window.

The cw_02 window now accepts values from the compute_win window.

27 Assign a slot number to the edge you just created:

a Click the edge that connects the compute_win window with the cw_02 window.

b In the right pane, enter `1` in the **Slot** field.

28 Expand **Transformations** on the **Windows** pane on the left and drag a Copy window to the workspace.

The right pane displays the Copy window's properties.

29 Specify a name and description for the Copy window:

a In the right pane, in the **Name** field, change the default name to `cw_03`.

b In the **Description** field, enter `This Copy window connects to the Compute window using the output slot -1.`

30 Configure a subscriber connector for the cw_03 window:

a Expand **Subscriber Connectors** and click .

The Connector Configuration window appears.

b In the **Name** field, replace the default value with `sub3`.

c In the **Fsname** field, enter the path to the CSV file that will contain the output. For example, you might enter `/opt/sas/viya/home/SASEventStreamProcessingEngine/<release>/examples/xml/slot_exp_xml/cw_03.csv`. Replace *<release>* with the release number in your SAS Event Stream Processing installation directory path.

d Select the **Snapshot** check box.

e In the **Fstype** drop-down list, select **csv**.

f Click **OK**.

31 Connect the compute_win window to the cw_03 window with an edge:

a Position the cursor over the anchor point at the bottom of the compute_win window so that the anchor point changes from black to white.

b Click the white anchor point, hold the left mouse button down, and draw a line to the anchor point in the compute_win window.

The cw_03 window now accepts values from the compute_win window.

32 Assign a slot number to the edge you just created:

a Click the edge that connects the compute_win window with the cw_03 window.

b In the right pane, enter `-1` in the **Slot** field.

33 The model is now complete. Click to save your model.

34 Click ▶ Test .

A new page called **Test: modellingSpliter_demo** appears.

35 In the **Test Server** drop-down list, select the ESP server on which you want to test the model.

36 Click ▶.

The results for each window appear on separate tabs:

- The **src_win** tab lists the securities transactions
- The **compute_win** tab lists the computed fields
- The **cw_01** tab lists the trades output to slot 0
- The **cw_02** tab lists the trades output to slot 1
- The **cw_03** tab lists the trades output to slot –1

Note: If the table is empty, check that the publisher connector for the src_win window correctly points to the relevant CSV file.

37 To stop the test, click ■.

The project stops and then unloads from the ESP server.

Working with SAS Micro Analytic Service Modules in SAS Event Stream Processing Studio

You can use SAS Micro Analytic Service modules to create input handler functions in SAS Event Stream Processing Studio using Python, DS2, and C.

SAS Event Stream Processing projects can also 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 the imported content that was created in SAS Model Manager. The module is uploaded and then referenced from the model's Calculate window's input handler. For more information, see "Example: Importing a Model Created in SAS Model Manager into SAS Event Stream Processing Studio" on page 77.

Create a SAS Micro Analytic Service Module

To create a new module:

1 Open your project and click .

2 In the right pane, expand **SAS Micro Analytic Service Modules**.

3 Click .

The SAS Micro Analytic Service Module window appears.

4 In the **Name** field, enter a name for the module.

5 In the **Language** drop-down list, select the language that you want to use to write the module.

6 In the **Description** field, enter a description of the module.

7 In the **Function names** field, enter a comma-separated list of function names.

8 In the **Code source** field, select one of the following options:

- **Embedded code** to enter your own code
- **External file** to use code located in an external file
- **SAS Micro Analytic Service store** to use code in an analytic store (ASTORE) file

9 If you selected **Embedded code** in the **Code source** field, enter your code in the **Embedded code** field.

10 If you selected **External file** in the **Code source** field, enter the file path to the external file in the **External file** field.

11 If you selected **SAS Micro Analytic store** in the **Code source** field:

a In the **External file** field, enter the file path to the ASTORE file.

b In the **SAS Micro Analytic Service store** field, enter the module store location.

c In the **SAS Micro Analytic Service store version** field, enter the version of the module store location.

d In the **Module Members** field, enter your module's member names. To add a new module member, click and fill in the applicable fields.

12 Click **OK**.

The module that you created appears in the SAS Micro Analytic Service Modules table.

Upload a SAS Micro Analytic Service Module

To upload an existing module:

1 Open your project and click .

2 In the right pane, expand **SAS Micro Analytic Service Modules**.

3 Click .

The Import a SAS Micro Analytic Service Module from SAS Model Manager window appears.

4 In the **ZIP file** field, click **Choose File**.

5 Select the SAS Model Manager ZIP file that you want to upload and click **Open**.

The Import a SAS Micro Analytic Service Module from SAS Model Manager window reloads to display information about the module's roles.

6 Review the module's role properties and modify them if necessary.

7 To copy the input schema for future use, click **View input schema** and then copy the input schema to your clipboard. To copy the output schema for future use, click **View output schema** and then copy the output schema to your clipboard.

8 Click **OK.**

The module that you uploaded appears in the SAS Micro Analytic Service Modules table.

Delete a SAS Micro Analytic Service Module

To delete a module, select the module that you want to delete from the SAS Micro Analytic Service Modules table and click  .

The module is deleted from the SAS Micro Analytic Service Modules table.

Example: Importing a Model Created in SAS Model Manager into SAS Event Stream Processing Studio

This example demonstrates how to import a model created in SAS Model Manager to process input data. 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 the imported content that was created in SAS Model Manager. The module is uploaded and then referenced from the Calculate window's input handler. Here is an example of a DS2 code file that might be imported from the SAS Model Manager common model repository to SAS Event Stream Processing Studio:

```
ds2_options sas;
package module_1/overwrite=yes;
    method score(int quantity, double price, in_out int volume);
    volume = quantity * price;
    end;
endpackage;
```

This example code generates the volume of a set of stock market trades.

Note: A pre-configured model is not provided with this example. To successfully complete this example, you must provide your own champion model and data files.

Before you start, ensure that SAS Model Manager is installed at your deployment and that the model that you plan to import is correctly configured in it. You must also ensure that you have noted the order and type of your module's output fields. In SAS Model Manager, assign the following variables to your model:

Name	Data Type	Input/Output Field
price	Decimal	Input
quantity	Integer	Input
security	Character	Input
time	Character	Input
tradeID	Character	Input
traderID	Integer	Input
volume	Integer	Output

To successfully import a model created in SAS Model Manager into a specific Calculate window in your model, the model that you want to import must meet the following criteria:

- The model is a champion of the project.
- The model has a score code type of dataStep, ds2Package, or dsMultiType that meets the following criteria:
  - The dataStep model contains a score file with a file role of score code.
  - The ds2Package model contains a score file with a file role of score code.
  - The dsMultiType file contains ASTORE content with a DS2 score code file named dmcas_packagescorecode.sas.
- The model is a single DS2 code file (DS2 Package) or a DS2 code file with one or more analytic store files.

1. On the **Projects** page, click New.

   The New Project window appears.

2. In the New Project window, do the following:

   a. In the **Name** field, enter `MM_Import`.

   b. In the **Description** field, enter: `This example demonstrates how to import a champion model from SAS Model Manager`.

   c. Click **OK**.

   It is assumed that an ESP server is already configured.

3. Click .

4. In the right pane, configure your project's properties. Update the fields as required.

5. Expand **Input Streams** on the **Windows** pane on the left and drag a Source window to the workspace.

   The right pane displays the Source window's properties.

6. Expand **Utilities** on the **Windows** pane on the left and drag a Calculate window to the workspace.

   The right pane displays the Calculate window's properties.

7. Connect the Source window to the Calculate window with an edge:

   a. Position the cursor over an anchor point in the Source window so that the anchor point color changes to white.

   b. Click the white anchor point, hold the left mouse button down, and draw the line to an anchor point in the Calculate window.

   The Calculate window now accepts events from the Source window.

8. Click the Calculate window in the workspace.

9. Expand **Settings**:

   a. In the **Calculation** drop-down list, select **User-specified**.

   b. Expand **Handlers** if it is not already expanded:

      i. Click the row that contains the Source window.

      ii. Click .

The Input Handler window appears.

iii In the **Handler Type** field, select **Import from SAS Model Manager**.

iv The Import from SAS Model Manager window appears, showing SAS Model Manager repositories in a collapsed state.

v Navigate to the SAS Model Manager project that contains the model that you want to import.

vi Select the model that you want to import.

The window refreshes to display additional information about the model that you have selected.

vii Inspect the additional information about the model.

viii If necessary, you can add the input schema fields and output schema fields that you are importing to the window. You can also copy the input schema fields and output schema fields for future use. To add the input schema, select **Add input schema to window** and select the window that you want to import the schema to from the list. To add the output schema, select **Add output schema to window** and select the window that you want to import the schema to from the list.

Alternatively, to copy the input schema for future use, click **View input schema** and then copy the input schema to your clipboard. To copy the output schema for future use, click **View output schema** and then copy the output schema to your clipboard.

ix Click **OK**.

The Input Handler window appears.

Your model's XML code is updated to reference the module that you imported. The Input Handler window displays the name of the imported module and the function to call from the Calculate window.

x Inspect and, if necessary, modify the information in the Input Handler window.

xi Click **OK**.

The import is completed. The imported code is written to the ESP server, typically to the `/opt/sas/viya/config/etc/SASEventStreamProcessingEngine/default/mas-modules` directory. This directory might be different depending on how your deployment has been configured.

10 Configure a publisher connector:

Note: Make a note of the name, order, and type of the fields defined in your publisher connector's input file. The name, order, and type of the fields defined in your input file must match the name, order, and type of the fields defined in your Source window's output schema.

a Click the Source window in the workspace.

b Expand **Input Data (Publisher) Connectors** and click .

The Connector Configuration window appears.

c In the **Name** field, enter `Connector`.

d Configure the remaining fields as required.

e Click **OK**.

11 Configure the output schema:

a Click the Source window in the workspace.

b In the right pane, click .

c Click .

The Output Schema window appears.

d Inspect your output schema to establish if a key field is defined. If a key field is not defined, define a key field.

e Inspect your output schema to establish if its fields match the name, order, and type of the output fields defined in the SAS Micro Analytic Service module. Your output schema fields must also match the name, order, and type of the fields defined in your input file. These fields must match for the imported model to run successfully in test mode.

12 Click .

13 Click Test.

A new page called **Test: MM_example** appears.

14 Click .

The results for each window appear in separate tabs.

15 To stop the test, click .

After the import has completed, your model's XML code is updated to display the following information:

- A `<metadata>` element that contains a set of unique identifiers. These identifiers are required to obtain the model's content when you run the model in test mode. Here is an example of an imported model's `<metadata>` element:

```
<metadata>
    <meta id="layout">{"trades_traders_cq":{"Trades":{"x":50,"y":50},"pw1":{"x":50,"y"
    :171.99652862548828}}}</meta>
    <meta id="mm_linked_module_1">fae64eb3-2de3-45ac-931a-088a56a49062,83455da4-5b04-41bd
    -944e-a5c1b2bf63cb,ds2Package</meta>
</metadata>
```

  Note: If you change the content of the `<metadata>` element, SAS Event Stream Processing Studio might be unable to locate your model's content.

- A `<mas-modules>` element that contains identifying information about the module that you imported. Here is an example of an imported model's `<mas-modules>` element:

```
<mas-modules>
       <mas-module module="module_1" language="ds2" func names="score" mas-store="fae64eb3-2de3-45ac
    -931a-088a56a49062_champion_45c08c8c-8490-4646-e8e518430379" mas-store-version="1.0">
    <code-file><![CDATA[score.as.ds2]]</code-file>
</mas-modules>
```

  In the example code here, the `<mas-module>` element defines an input handler to the SAS Micro Analytic Service engine. The `<mas-store>` attribute contains the SAS Micro Analytic Service store name. The store name is `fae64eb3-2de3-45ac-931a-088a56a49062`, the ID for the ESP project is `45c08c8c-8490-4646-e8e518430379`, and the store version number is `1.0`. The model type is also specified as `champion`.

  The `<code-file>` element encloses the name of the score.as.ds2 file that contains the code used as an input handler.

  Because the imported code is written to the ESP server, not the model, you can view only the code from the ESP server. In the example code here, the imported code is written to the /opt/sas/viya/config/etc/

SASEventStreamProcessingEngine/default/mas-modules/fae64eb3-2de3-45ac-931a-088a56a49062_champion_45c08c8c-8490-4646-e8e518430379/1.0/score.mas.ds2 file. This file's location might be different depending on how your deployment has been configured.

- New schema fields for the Source and Calculate windows (if the model that you imported contained additional schema fields).
- A `<mas-map>` element in the Calculate window that references the newly created SAS Micro Analytic Service module and its function. The `<mas-map>` element binds the function that is defined in the `<mas-module>` element to the input stream in your model's Calculate window. Here is an example of an imported model's `<mas-map>` element:

```
<mas-map>
    <window-map module="module_1" function="score" revision="0" source="Trades"/>
</mas-map>
```

Working with Input Handlers

You can register event stream input handlers for the Procedural and Calculate windows. *Input handlers* process incoming event streams in your model. You can import score code created in SAS Model Manager directly into a specific Calculate window in your model. You define a SAS Micro Analytic Service map in a Calculate window to bind a function to an input window. This binding acts as the input handler for the Calculate window.

Creating Input Handler Functions in Procedural Windows

To create an input handler function within a Procedural window:

1. Open the relevant project.
2. Click the relevant Procedural window.

   The right pane displays the properties of the Procedural window.
3. In the right pane, expand **Input Handlers**.
4. Click the row for the input window that you want to link the import handler function to.
5. Click .

   The Input Handler window appears.
6. In the **Handler type** field, select one of the following handler types:
   - **Plug-in** – enables you to reference a plug-in library
   - **DS external file** – enables you to reference an external file that contains DATA step code

     Note: When you configure a model that contains a Procedural window that executes DATA step code, you must add the `ds-initialize` element to your project using the XML Editor.

   The Input Handlers window reloads to display fields that relate to the handler type that you selected.
7. Update any other fields as required.
8. Click **OK**.

   The Input handler functions section refreshes to display the imported function.

You can use SAS Event Stream Processing Studio to specify a function that returns derived context using the cxx-plugin-context plug-in. To do this:

1 Open your model and click the relevant Procedural window in the workspace.

   The right pane displays the properties of the Procedural window.

2 In the right pane, expand **Input Handlers**.

3 Select the **Use a context plug-in** check box.

4 In the **Plug-in library name** field, enter a name for the relevant plug-in library.

5 In the **Function** field, enter the function name.

6 To create a property list, in the **Property map** field, click to add a new row to the table. Click again to add an additional row to the table:

   a In the **Name** field, enter a name for the property.

   b In the **Value** field, enter a value for the property.

Creating Input Handler Functions in Calculate Windows

To create an input handler function within a Calculate window:

1 Open the relevant project.

2 Click the relevant Calculate window.

   The right pane displays the properties of the Calculate window.

3 In the right pane, expand **Settings**.

4 In the **Calculation** drop-down list, select **User-specified**.

5 Click the row for the input window that you want to link the import handler function to.

6 Click .

   The Input Handler window appears.

7 In the **Handler type** field, select one of the following handler types:

   - **SAS Micro Analytic Service** – enables you to reference a SAS Micro Analytic Service module
   - **Import a module from SAS Model Manager** – enables you to reference a SAS Micro Analytic Service module
   - **Import a module from SAS Model Manager ZIP file** – enables you to reference a SAS Micro Analytic Service module

   The Input Handlers window reloads to display fields that relate to the handler type that you selected.

8 Update any other fields as required.

9 Click **OK**.

   The Handlers section refreshes to display the imported function.