Cloud-Native Capabilities

SAS Event Stream Processing 6.2 provides pre-built Docker images of the following applications:

- the ESP operator
- the ESP server
- the metering server
- SAS Event Stream Processing Studio
- SAS Event Stream Processing Streamviewer
- SAS Event Stream Manager

After you download these images through your Software Order Email (SOE), you can use tools in a GitHub project to develop, deploy, and test an open ESP server in a Kubernetes cluster. Those tools consist of a set of scripts, YAML template files, and sample projects (XML files) that you can run in the ESP server.

For more information about deploying the product in a Kubernetes cluster, see the SAS Event Stream Processing on Linux: Deployment Guide.
Enhancements to Streaming Analytics

New Online Models
The following online models are now available:
- Kalman Filter
- Slice Operations
- Smoothing

Added Support for Models That Are Stored in Analytic Store Files
Analytic store files that apply the following algorithms can now be used with SAS Event Stream Processing:
- Association Rule Mining
- Hidden Markov Model
- Kernel Principal Component Analysis (KPCA)

For more information, see "Loading Models Stored in Analytic Store Files " in SAS Event Stream Processing: Using Streaming Analytics.

Improvements to the Image Processing Algorithm
- Three types of coordinate specifications are now supported: COCO, RECT, and YOLO. Previously, only RECT was supported.
- Cropping parameters can now be received as part of the event stream. Previously these parameters had to be specified in the model file, and thus all images had to be cropped identically. They now can be supplied from a Score window that uses a deep learning algorithm in an analytic store (ASTORE) file. The input map of the Calculate window can now accept the streaming cropping parameters.

For more information, see "Processing Image Data" in SAS Event Stream Processing: Using Streaming Analytics.

Added Support for Deep Learning Functionality
The following deep learning functionality that is available in SAS Visual Data Mining and Machine Learning is now supported for use with SAS Event Stream Processing:
- Mask R-CNN for instance segmentation
- a new ROI (region of interest) alignment layer that is used in Mask R-CNN
- Layer Normalization
- a new embedding loss layer to train Siamese, triplet, and quartet networks
- a Survival layer type for the addLayer action (in the deepLearning action set) in order to create deep survival analysis analytic models
- the Cosine Margin Softmax loss function
- custom FCMP functions that you can use to define early stopping criteria during training
- data augmentations (such as cropping, mutating, and flipping) for object detection and segmentation tasks
- deep clustering unsupervised learning applications

**Miscellaneous Improvements**
- The Streaming Summary algorithm has a new alpha parameter and new output parameters.
- The Lag Monitoring algorithm has a new parameter to specify the correlation type. You now can specify a distance correlation as well as a Pearson correlation.
- The Recommender Model algorithm now supports Nonnegative Matrix Factorization (NMF) in the Train, Model Reader, and Score windows.

**Enhancements to SAS Event Stream Processing Studio**
Enhancements and changes to the user interface include the following:
- A new layout theme has been implemented to improve the user experience.
- The user interface for test mode is enhanced. For example, in the Logs pane, you now can view more than one type of log message simultaneously. In addition, you now can more easily identify your model's window and field types in test mode.
- You now can validate expressions that you enter from the user interface. This functionality validates the syntax of the code that you enter. It can be particularly useful when you enter several lines of codes or paste code from an external source.
- Windows that you have selected are displayed in the workspace with a dashed bounding box, making them easier to identify.
- You can now select multiple projects, engines, and ESP servers simultaneously, which enables you to delete them more efficiently.
- The Engine Definitions page is redesigned to improve usability.

SAS Event Stream Processing Studio has been enhanced with cloud-native capabilities, which enables you to use a containerized version of the application.
- You now can use the application in a Kubernetes cluster, which makes it easier to deploy and run ESP servers in the application.
- When a project is created in a Kubernetes cluster, an ESP server is created on demand. When the project is stopped, the ESP server is deleted from the cluster.

When using a Pattern window to detect events of interest (EOIs), you now can import your own Pattern logic templates and use them in SAS Event Stream Processing Studio.

For more information, see *SAS Event Stream Processing: Using SAS Event Stream Processing Studio*. 
Enhancements to SAS Event Stream Processing Streamviewer

- The application layout theme is changed to improve the look of the user interface.
- The user interface is enhanced with new and clearer icons to improve usability.
- SAS Event Stream Processing Streamviewer is enhanced with cloud-native capabilities enabling you to use a containerized version of the application.
  - You now can use the application in a Kubernetes cluster, which makes it easier to deploy and run ESP servers in the application.
  - You now can visualize the event streams of a project where the ESP server runs within a Kubernetes cluster.

For more information, see *SAS Event Stream Processing: Using SAS Event Stream Processing Streamviewer*.

Pylon Connector and Adapter Now Supports a USB Attached Camera

The Pylon connector and adapter now enables you to communicate with a USB attached camera in order to continuously publish captured frames into a Source window. For more information, see “Using the Pylon Publisher Connector and Adapter” in *SAS Event Stream Processing: Connectors and Adapters*.

New OPC-DA Publisher Adapter

A new OPC-DA publisher adapter can be used for communication with industrial equipment and systems for data collection and control. OPC Data Access (OPC DA) is precursor to OPC Unified Architecture (OPC UA). For more information, see “Using the OPC-DA Publisher Adapter” in *SAS Event Stream Processing: Connectors and Adapters*.

Miscellaneous Enhancements to Connectors and Adapters

- The Avro format library now supports the conversion of an Avro decimal logical type to an ESP_MONEY datavar. The Avro decimal logical type represents an arbitrary precision signed decimal number. The ESP_MONEY datavar is a 192–bit fixed decimal.
- The Cassandra adapter now supports ESP blob fields.
- The database connector and adapter now supports the SQL_BINARY format.
- Added an ESP publish/subscribe server configuration parameter to bind to a specific NIC.
- The file and socket subscriber connector and adapter now supports the following:
  - opaquestring data
opaque_binary files

- The JMS adapter now supports copying MapMessage to and from an ESP blob field.
- Added support for configurable consumer groups to Kafka publisher connector and adapter.
- Added an argument to suppress the event opcode from outbound Avro messages in the Kafka subscriber, Kinesis subscriber, and Kafka client transport.
- Enhanced the Kafka publisher connector and adapter and client transport to honor the configured offset when a consumer group is configured.
- The Kinesis connector and adapter now supports Assume Role credentials.
- The PI publisher connector and adapter now support copying all values to an ESP string field.
- A new, optional parameter to the Pylon connector and adapter is available to compress images in YUV422_YUYV_Packed format into JPEG format.
- A configurable heartbeat to the RabbitMQ C++ and Java client transports is now available.
- The sniffer connector and adapter now support copying packet payloads to an ESP blob field.
- You now can use the following as the message ID for hot failover:
  - the RabbitMQ delivery tag
  - the Kafka offset

For more information, see *SAS Event Stream Processing: Connectors and Adapters*.

---

Enhancements to ESPPy

ESPPy now enables you to group windows that are connected in a specific way into templates, which then can be reused within a Python program. For more information, see “Creating and Using Templates” in *SAS Event Stream Processing: Using the Python Interface*.

ESPPy now provides the capability to visualize ESP models in action within a Jupyter Lab environment. You can create various charts, an ESP model viewer, and an ESP log viewer. For more information, see “Visualization Programming Objects in ESPPy” in *SAS Event Stream Processing: Using the Python Interface*.

---

New ESP Server Connection API

A new API that uses the WebSocket protocol is available that enables ESP functions to operate with minimal system resources. The API sets up a single WebSocket connection to handle all ESP server activities. You can have a single WebSocket connection that does the following actions:

- subscribes to any number of ESP windows
- publishes events
- receives server status information
- monitors the server log

For more information, see “Using the ESP Server Connection API” in *SAS Event Stream Processing: WebSocket API Reference*.
New Capabilities for the RESTful API

The ESP server now supports HTTP requests over Kafka and MQTT message queues, processing those requests as if they were sent over a TCP/IP socket. For more information, see “Using the RESTful API over Message Queues” in SAS Event Stream Processing: Using the ESP Server.

New Configuration Capabilities

The configuration file esp-properties.yml has been expanded to include two new sections:

- The env section enables you to set additional environment variables.
- The prometheus section enables you to expose ESP server metrics to Prometheus, an open-source application used to monitor events and alerts in a cloud-based environment.

For more information, see “Configuring the ESP Server” in SAS Event Stream Processing: Using the ESP Server.