

# Ask the Expert

How Do I Streamline AI Project Workflows?

Daniele Cazzari, Sr Manager, IoT Product Innovation & Engineering  
Rik de Rooter, Senior Systems Architect





# Daniele Cazzari

Sr Manager, IoT Product Innovation & Engineering

Daniele brings over 10 years of experience in IoT architecture. He currently serves as a technical lead for cloud-native SAS IoT solutions and products. His primary objective is to streamline the processing and analysis of data derived from edge devices. Daniele has driven several key initiatives including ESP-ONNX integration, computer vision edge architecture and cloud marketplaces deployment.



# Rik de Ruiter

## Senior Systems Architect

Rik is a technical product manager in the SAS IoT division, where he is dedicated to improving real-time inferencing and open source integration for IoT applications. His focus is on making it easier to solve real business cases using AI.

# ASK THE EXPERT

## How Do I Streamline AI Project Workflows?

**Daniele Cazzari**  
Sr Manager, IoT Product  
Innovation & Engineering

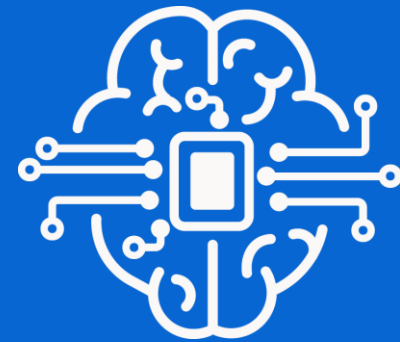
**Rik de Ruiter**  
Sr Systems Architect



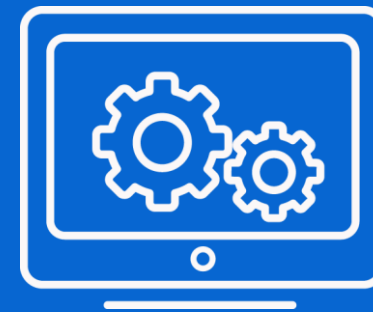
# Agenda



Introduction to  
ONNX and  
ESP



ONNX Integration  
with ESP



Demo:  
ONNX Object Detection



Azure  
Marketplace  
deployment



Q&A

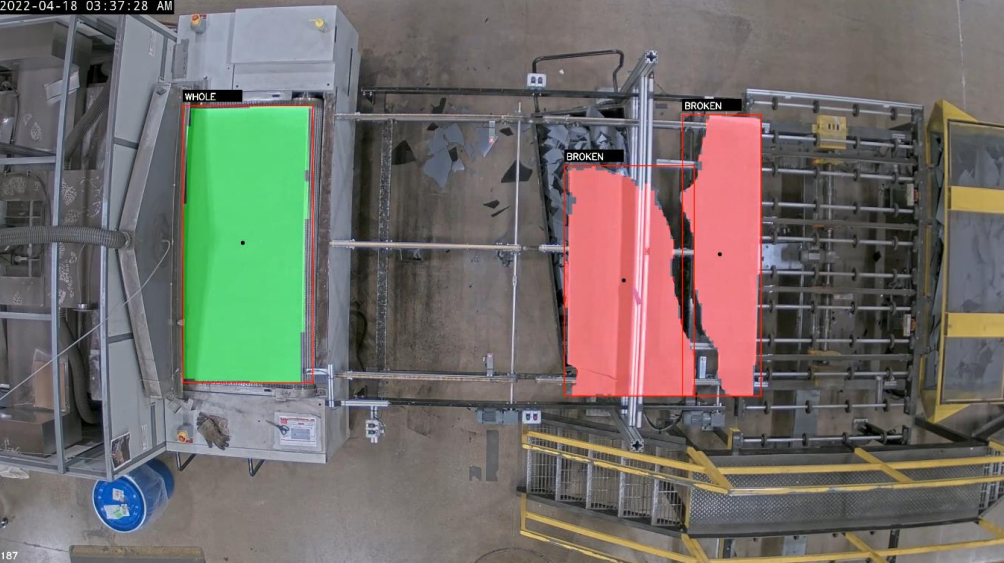
# Introduction to ONNX and ESP

What do these technologies help me with?

# Real-world Computer Vision use cases

Applicable to many industries

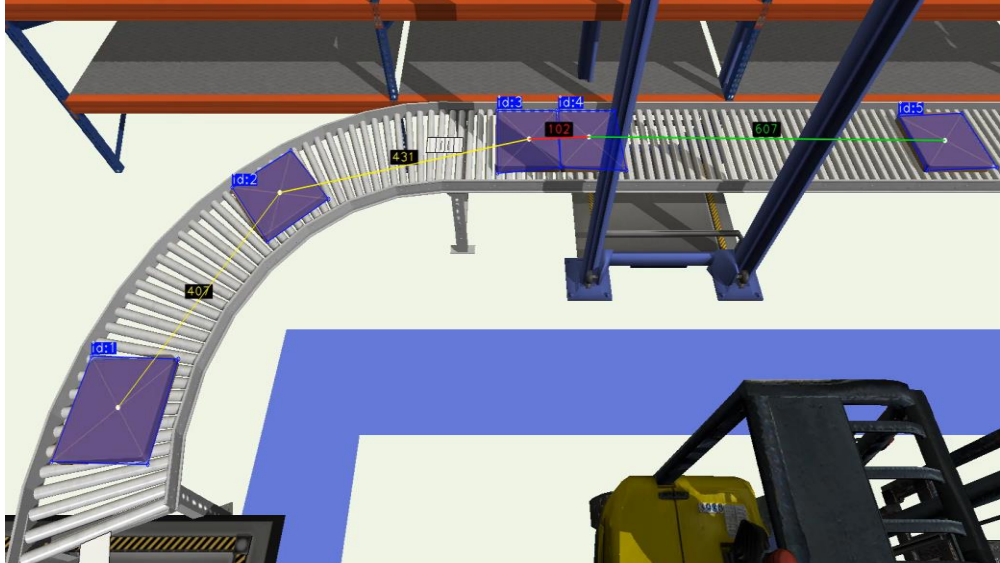
Product Quality



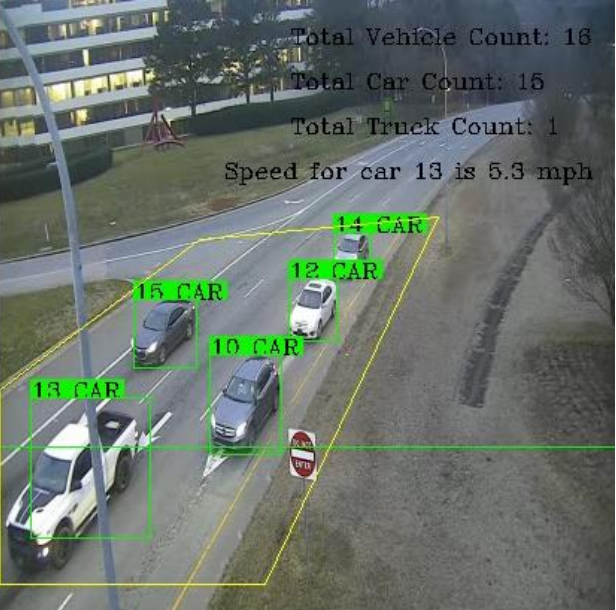
Safety in Workspace



Conveyor belt monitoring



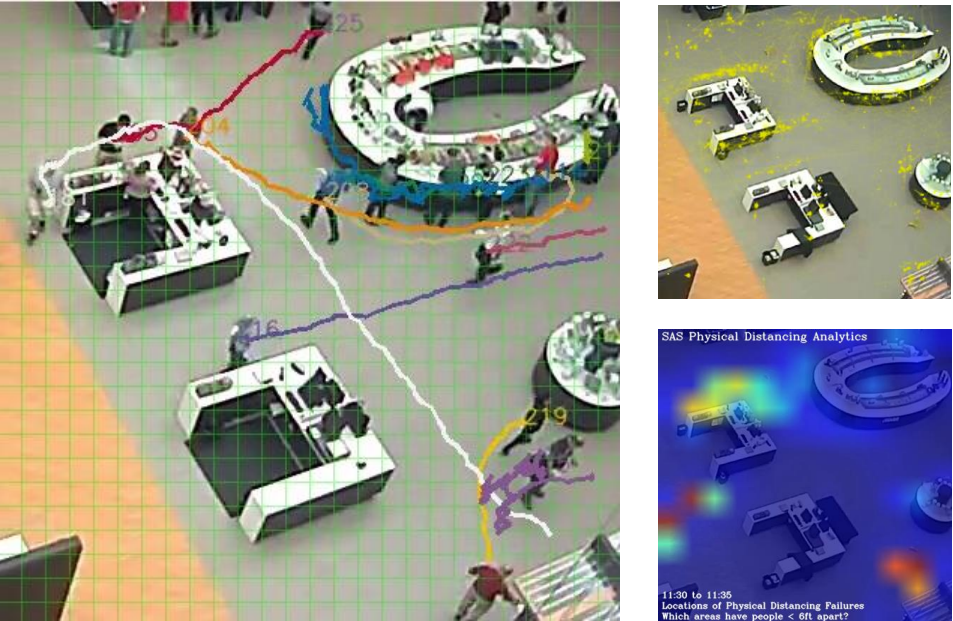
Traffic & Vehicle Monitoring



Traffic Light Monitoring

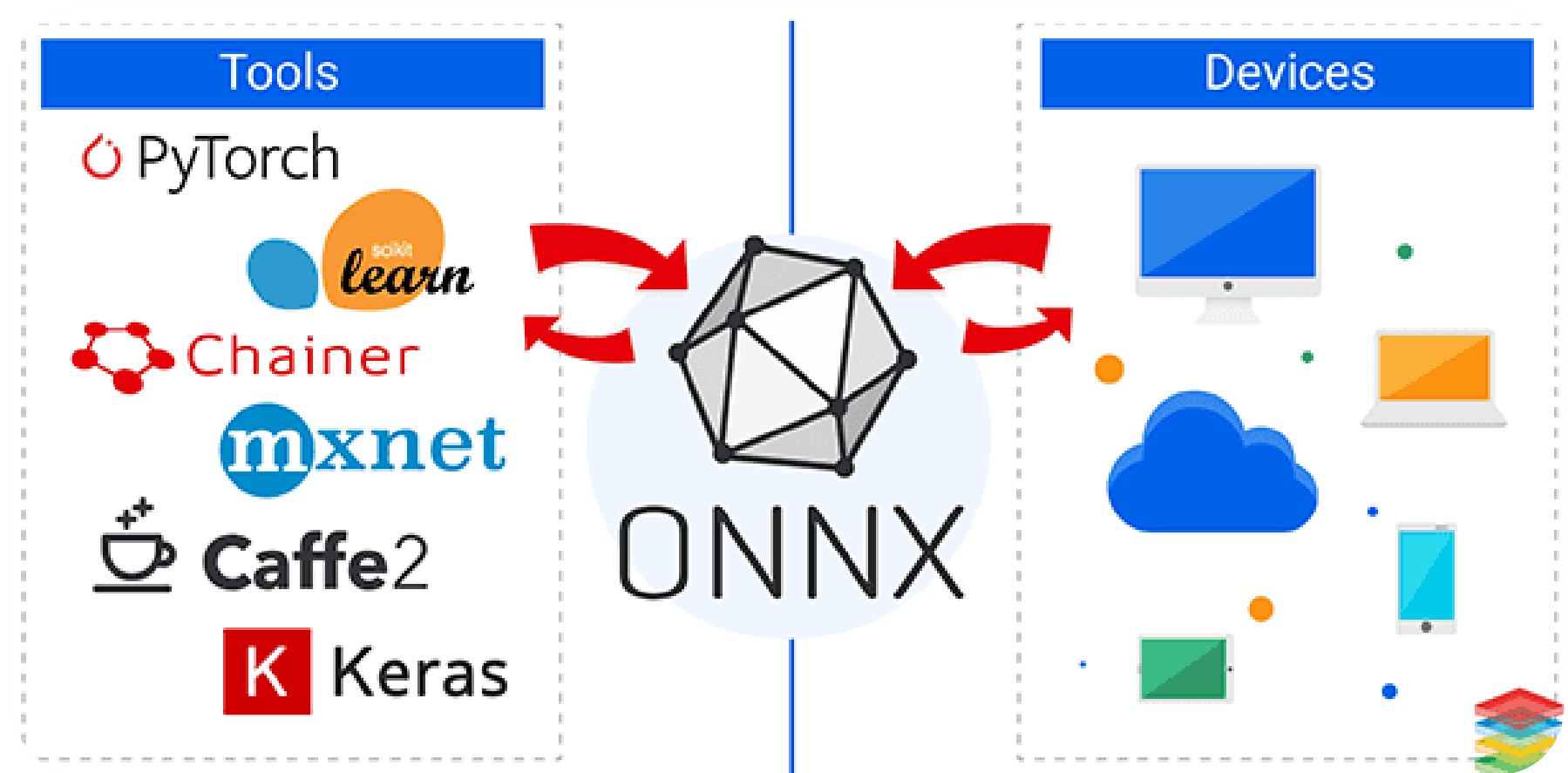


In-Store monitoring

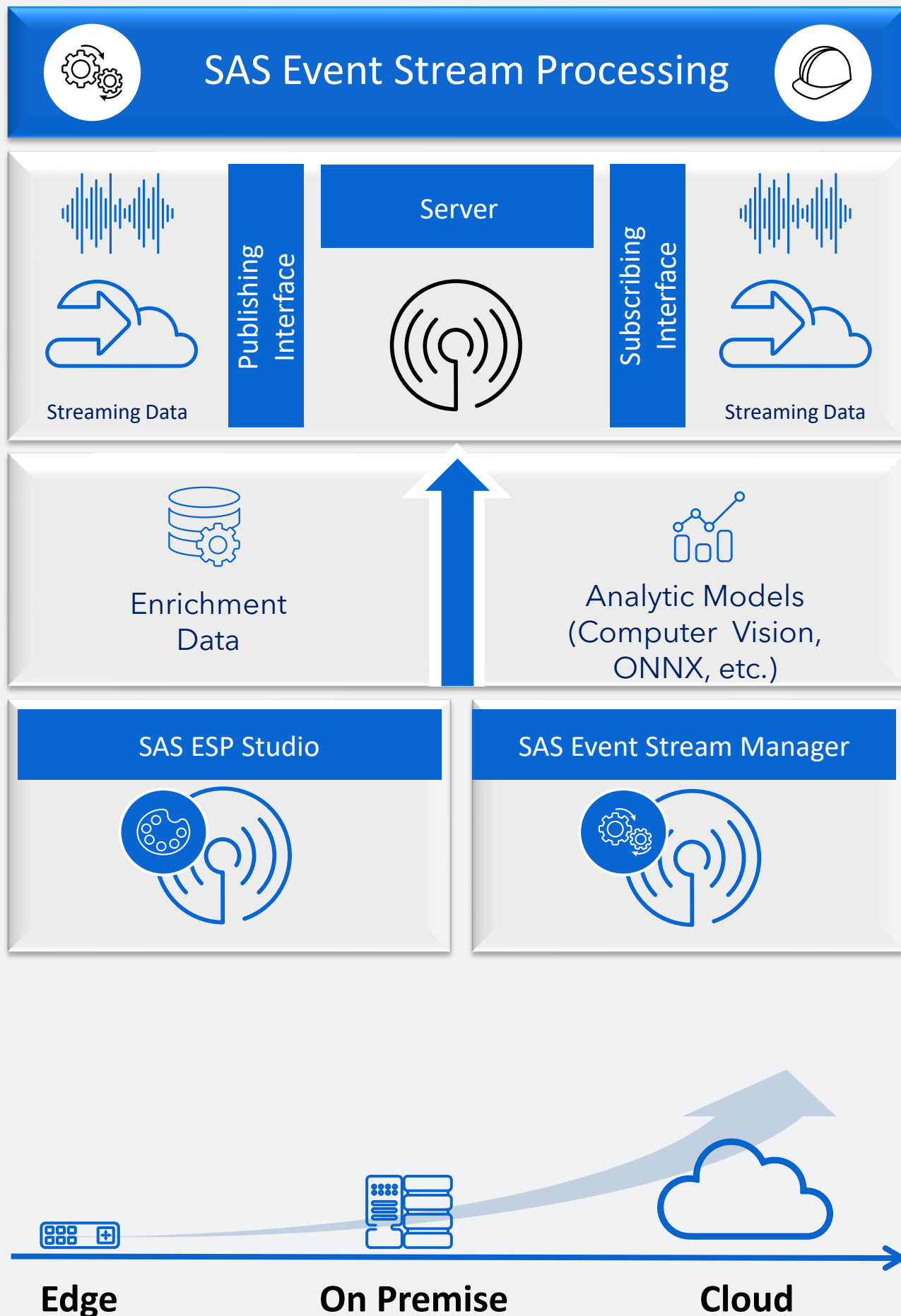


# What is ONNX?

- The **Open Neural Network Exchange (ONNX)** is open standards for representing machine learning algorithms to promote innovation and collaboration in the AI sector.
- **Framework interoperability**
  - Allow developers to more easily move between frameworks, some of which may be more desirable for specific phases of the development process, such as fast training, network architecture flexibility or inferencing on mobile devices.
- **Shared optimization**
  - *Allow hardware vendors and others to improve the performance of artificial neural networks of multiple frameworks at once by targeting the ONNX representation*







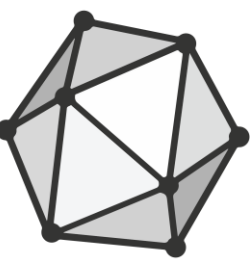
# SAS Event Stream Processing

## Empowering Computer Vision

- ✓ **Fast streaming data analysis:** Processes structured or unstructured data continuously, on the move, in-memory with very high speed and low latency.
- ✓ **End-to-end, streamlined, low-code solution** for real-time Computer Vision inferencing. Enable end-to-end use case from video ingestion to image annotation into a single, cohesive solution.
- ✓ **Optimized for deep learning inferencing:** Provide build-in capabilities to inference computer vision models, leveraging native ONNX runtime integration and hardware acceleration frameworks like Intel Openvino.
- ✓ **Operational Efficiency:** Enhances the ability to monitor, analyze, and optimize operations in real time, leading to significant improvements in productivity and cost savings.

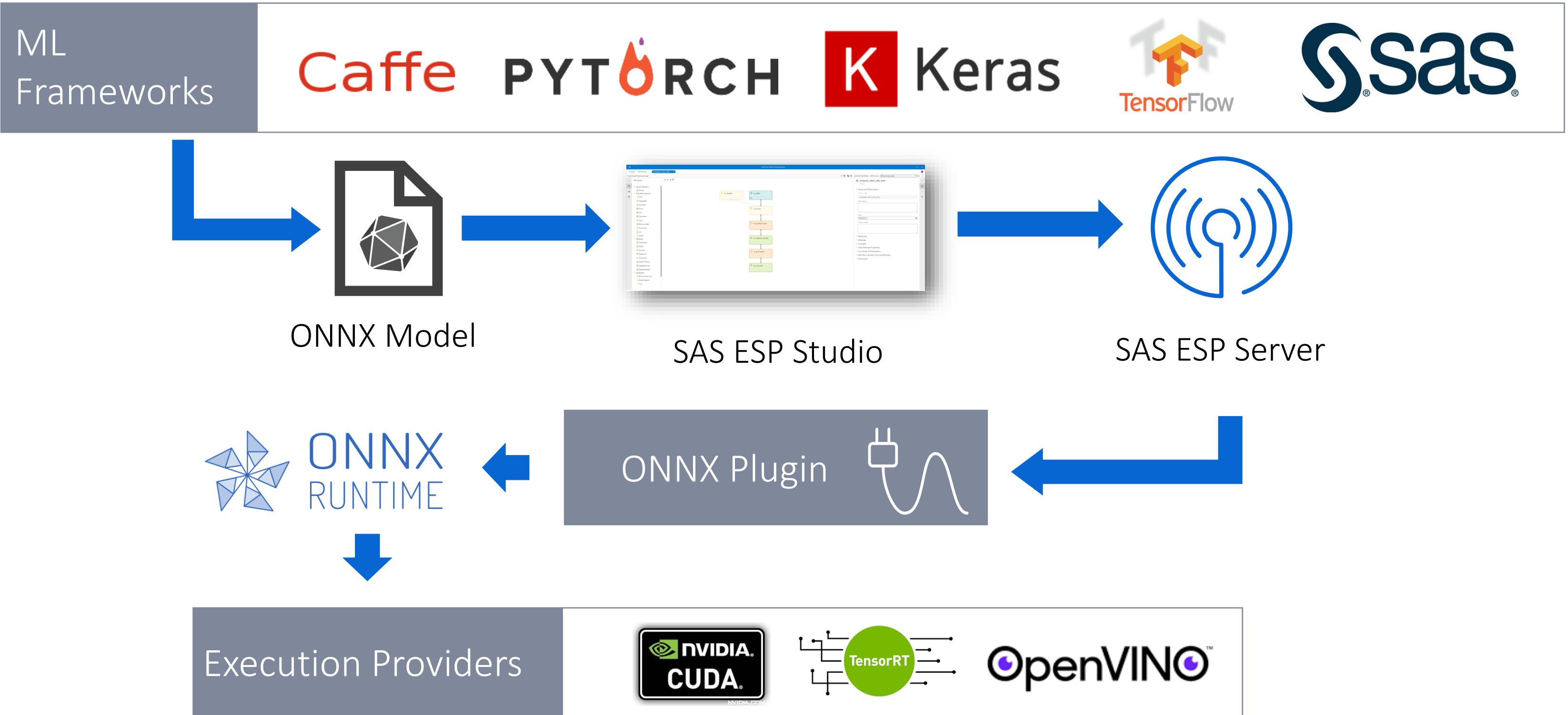
# ONNX Integration with ESP

How does it work?



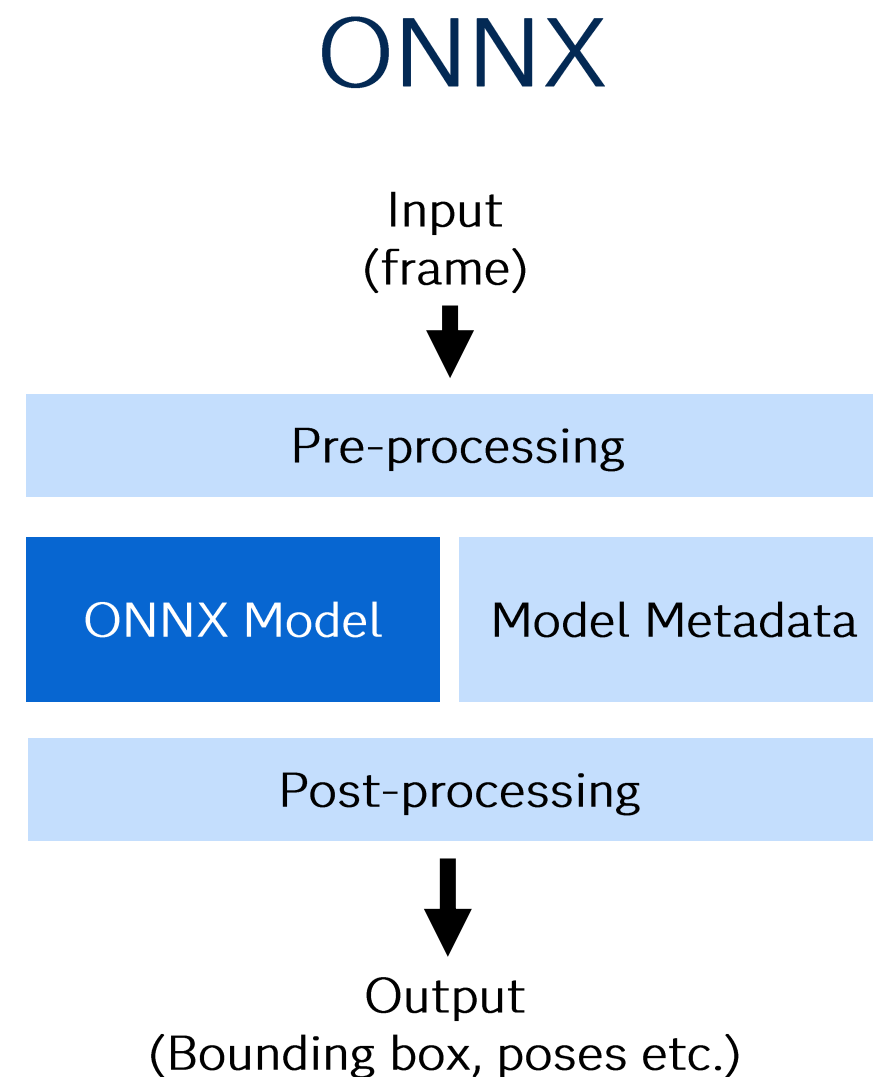
# ESP and ONNX

## Components



# ONNX scoring pipeline

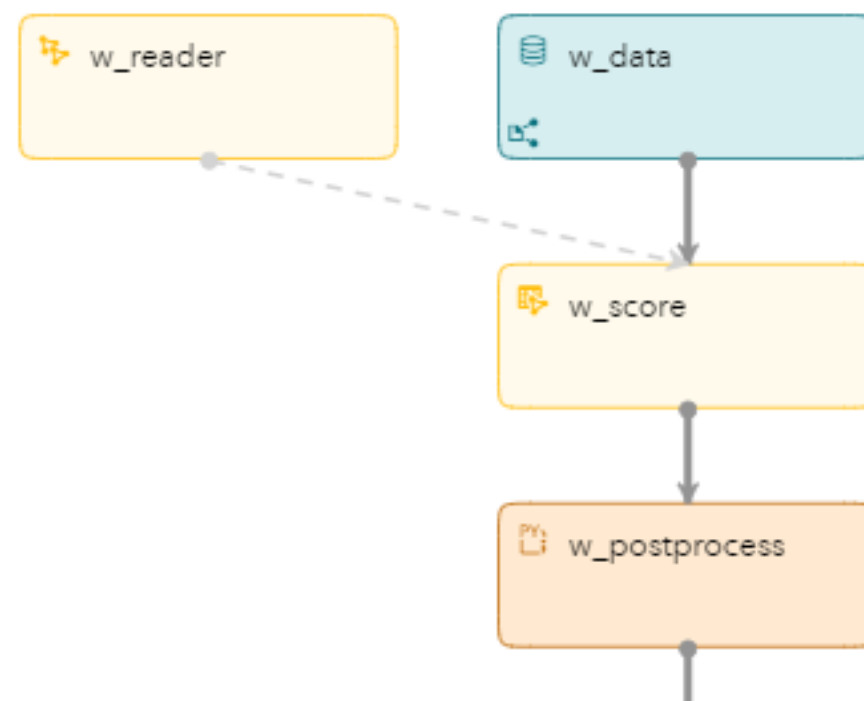
ONNX model is only a portion of the overall flow



- ONNX file is only the model and does not include pre-processing, post-processing or metadata
- Examples of preprocessing steps are resizing, BGR2RGB conversion, normalization, NCHW encoding
- Postprocessing includes decoding the output tensor of the ONNX model, and sometimes additional steps such as non-max suppression (NMS)
- Model metadata often includes a labels.txt file to map IDs back to class names later

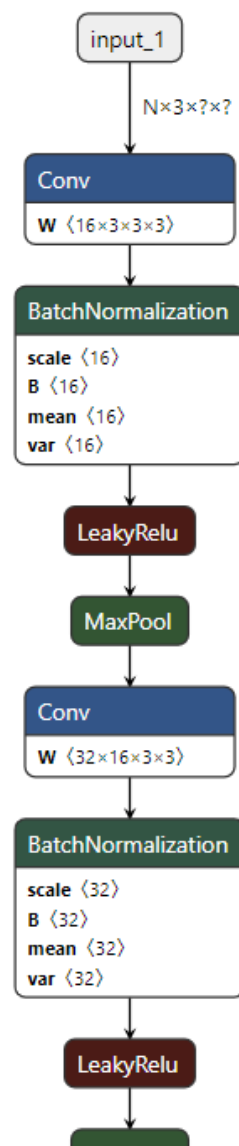
# ESP ONNX Pipeline

Low-code pipeline for real-time computer vision



- Reading video data via the Video Capture connector (**new!**)
- Preprocessing configuration via the UI (**new!**)
- Scoring the ONNX model via the Score window
- Postprocessing handled via a Python window (**new!**)
- Python window contains utility functions to make it easier to work with tensors and images (**new!**)

# ONNX Scoring: Retrieve Model Tensor Input and Output



**MODEL PROPERTIES**

format	ONNX v6
producer	keras2onnx 1.6.0
domain	onnx
imports	ai.onnx v11
subgraph	model_1

**INPUTS**

input_1	name: <b>input_1</b>	-
	type: float32[N,3,,]	
image_shape	name: <b>image_shape</b>	-
	type: float32[N,2]	

**OUTPUTS**

yolonms_layer_1	name: <b>yolonms_layer_1</b>	-
	type: float32[1,,4]	
yolonms_layer_1:1	name: <b>yolonms_layer_1:1</b>	-
	type: float32[1,80,]	
yolonms_layer_1:2	name: <b>yolonms_layer_1:2</b>	-
	type: int32[1,,3]	

- Each model could one or more tensors as input or output
- A tool like Netron provides graphical information about input and output formats (<https://netron.app/>)
- ESP provides functionality to retrieve this information via Studio or command line
- Each input/output is a tensor that need to be encoded/decoded during the inferencing process

# UI-based pre-processing

Steps:



resize  Advanced properties

Property	Value
resizeType	letterbox ▼
width	@MODEL_WIDTH@
height	@MODEL_HEIGHT@

color

Property	Value
function*	BGR2RGB ▼

normalize

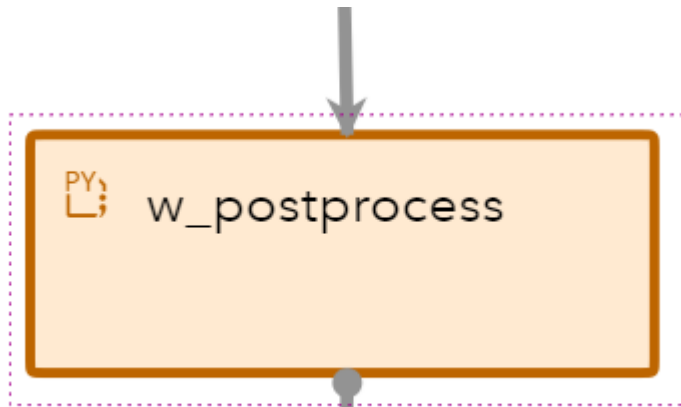
Property	Value
type*	zero_one ▼

encode

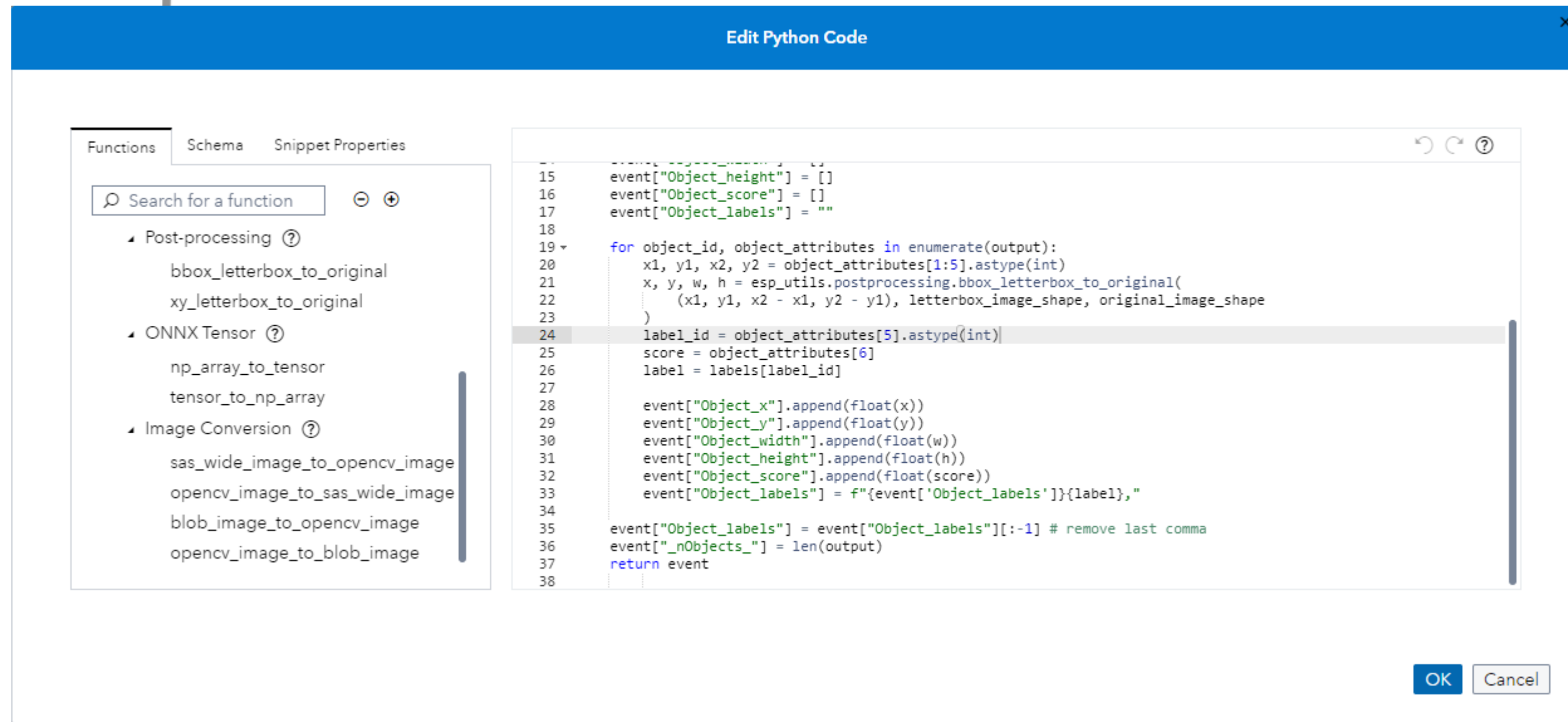
Property	Value
shape*	NCHW ▼

- UI based preprocessing:
  - Resize: basic, letterbox, ratio resize
  - Normalization: 0-1, mean vector, standard deviation vect.
  - Color Encoding: BGR, RGB
  - Other transformations: Image Encoding (NCHW), Padding/ Stride, Expand Dimension

# Post-processing



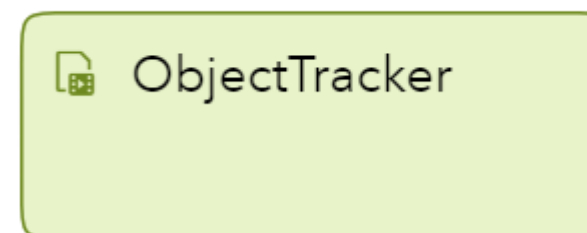
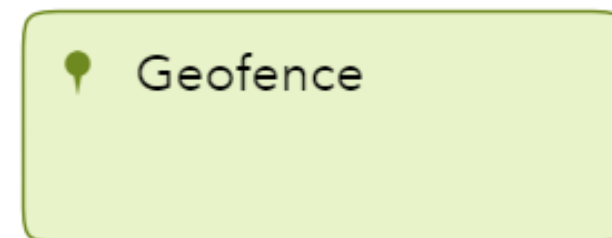
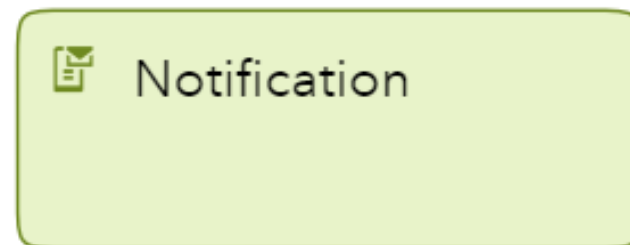
- Process the ONNX output tensors into ESP variables using a Python window
- Utility functions are available





# What's next?

- Further processing could include:
  - Object tracking
  - Geofencing
  - Annotation
  - Decision logic
  - Notifications



**w\_object\_tracker** Object Tracker window ?

▼ Name and Description

Name: \* ?

Description:

> Settings

▼ Tracking Properties ?

Tracking method: \*

IOU

Sigma score:

Low:

High:

Sigma thresholds:

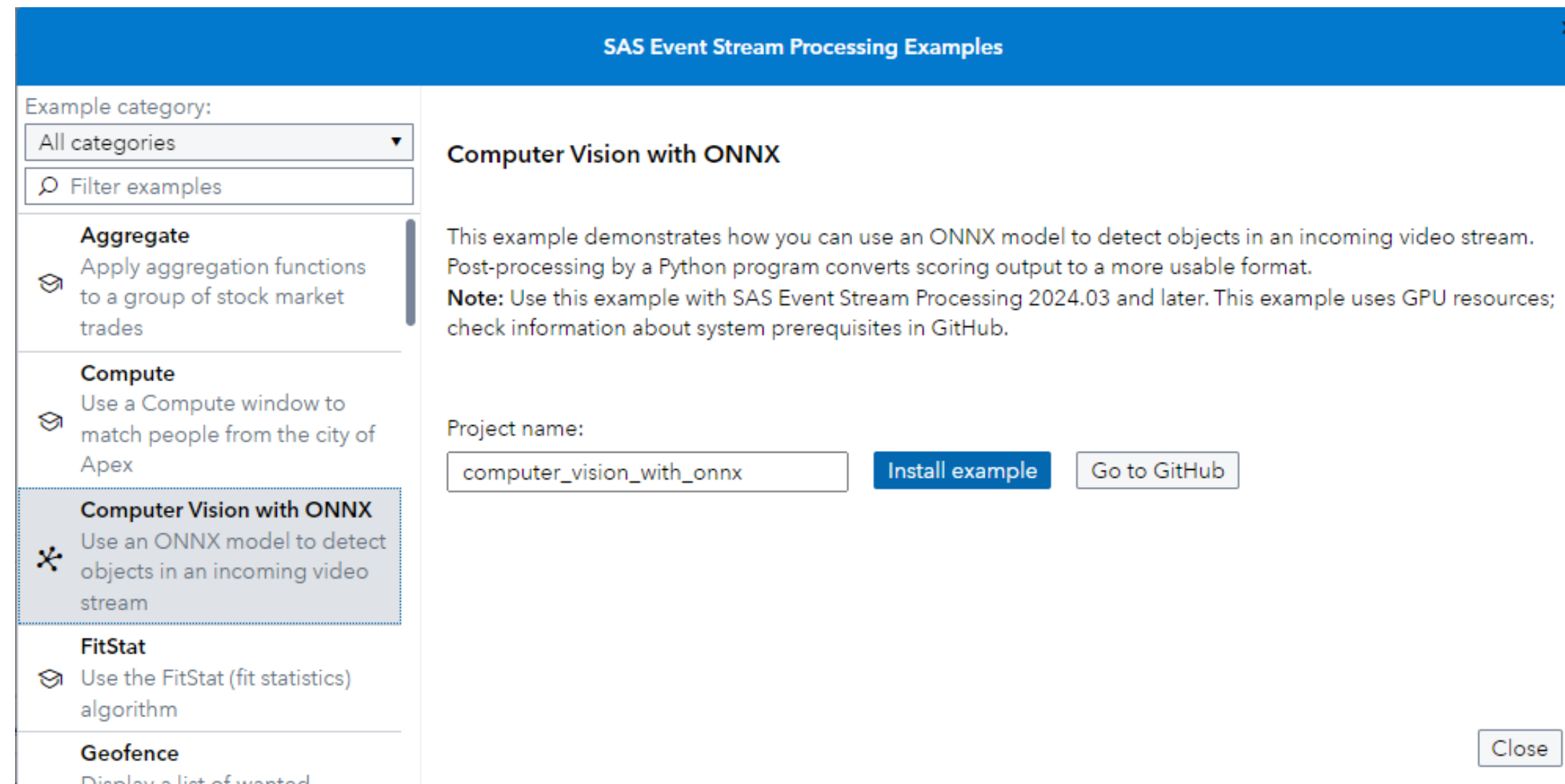
First:

Second:

# Installable example for Computer Vision

Runs out of the box, comes with all scripts and files to run

- Object detection example



The screenshot shows a web interface titled "SAS Event Stream Processing Examples". On the left, there is a sidebar with a search bar and a list of example categories. The "Computer Vision with ONNX" category is selected and highlighted. The main content area displays the details for this example, including a description, a note about system prerequisites, a project name input field, and buttons for "Install example" and "Go to GitHub". A "Close" button is located in the bottom right corner of the interface.

**SAS Event Stream Processing Examples**

Example category:  
All categories

Filter examples

**Aggregate**  
Apply aggregation functions to a group of stock market trades

**Compute**  
Use a Compute window to match people from the city of Apex

**Computer Vision with ONNX**  
Use an ONNX model to detect objects in an incoming video stream

**FitStat**  
Use the FitStat (fit statistics) algorithm

**Geofence**  
Display a list of wanted

**Computer Vision with ONNX**

This example demonstrates how you can use an ONNX model to detect objects in an incoming video stream. Post-processing by a Python program converts scoring output to a more usable format.

**Note:** Use this example with SAS Event Stream Processing 2024.03 and later. This example uses GPU resources; check information about system prerequisites in GitHub.

Project name:  
computer\_vision\_with\_onnx

Install example

Go to GitHub

Close

# Demo ONNX Object Detection

# Azure Marketplace

Enablement and deployment

# Event Stream Processing

## Azure Marketplace Deployment

- Simple and intuitive provisioning of a cost-effective Kubernetes infrastructure that fully support SAS Event Stream Processing enabled with Git and Grafana
- Support for scalable CPU and GPU workload tailored for you application needs
- Automatic configuration of managed Postgres DB that could be also used to store streaming analysis outcome
- Optional integration with EventHub for data ingestion and external blob storage

**Create SAS Event Stream Processing** ...

Basics License Accounts Networking Sizing Integrations Review + create

This page is blank when you update the application.

**Compute Size of your Kubernetes Node**

Node size: \* ⓘ **1x Standard F8s v2**  
8 vcpus, 16 GB memory  
[Change size](#)

Maximum number of node instances: ⓘ  2  
Num

**Compute Size of your Kubernetes GPU Node**

Add GPU node: ⓘ

Node size: \* ⓘ **1x Standard NC4as T4 v3**  
4 vcpus, 28 GB memory  
[Change size](#)

Minimum number of GPU node instances: ⓘ  1  
Num

Maximum number of GPU node instances: ⓘ  2  
Num

Time slicing count: ⓘ  4  
Num

**Azure Database for PostgreSQL Server**

SAS Event Stream Processing deploy PostgreSQL to store settings and project. If needed you could increase the default sizing to enable storing streaming results. If Cores selected are more than 1 a esp-data schema will be generated and proper connection configuration will be stored.  
[Learn more](#)

# GPU support in Azure Marketplace

- Easy to deploy a GPU, just a checkbox
- Reduce costs by autoscaling to 0 when the GPU is not used
- Multiple projects can share the GPU (time slicing support) to fully utilize a GPU

The screenshot shows the 'Create SAS Event Stream Processing' page in the Microsoft Azure Marketplace. The 'Sizing' tab is selected, showing configuration options for the Kubernetes Compute Node Size and the Compute Size of your Kubernetes GPU Node.

**Kubernetes Compute Node Size**

- Node size \* ⓘ: **1x Standard D8ds v4** (8 vcpus, 32 GB memory) [Change size](#)
- Maximum number of node instances ⓘ:  2

**Compute Size of your Kubernetes GPU Node**

- Add GPU node ⓘ:
- Node size \* ⓘ: **1x Standard NC4as T4 v3** (4 vcpus, 28 GB memory) [Change size](#)
- Minimum number of GPU node instances ⓘ:  1
- Maximum number of GPU node instances ⓘ:  2
- Time slicing count ⓘ:  4

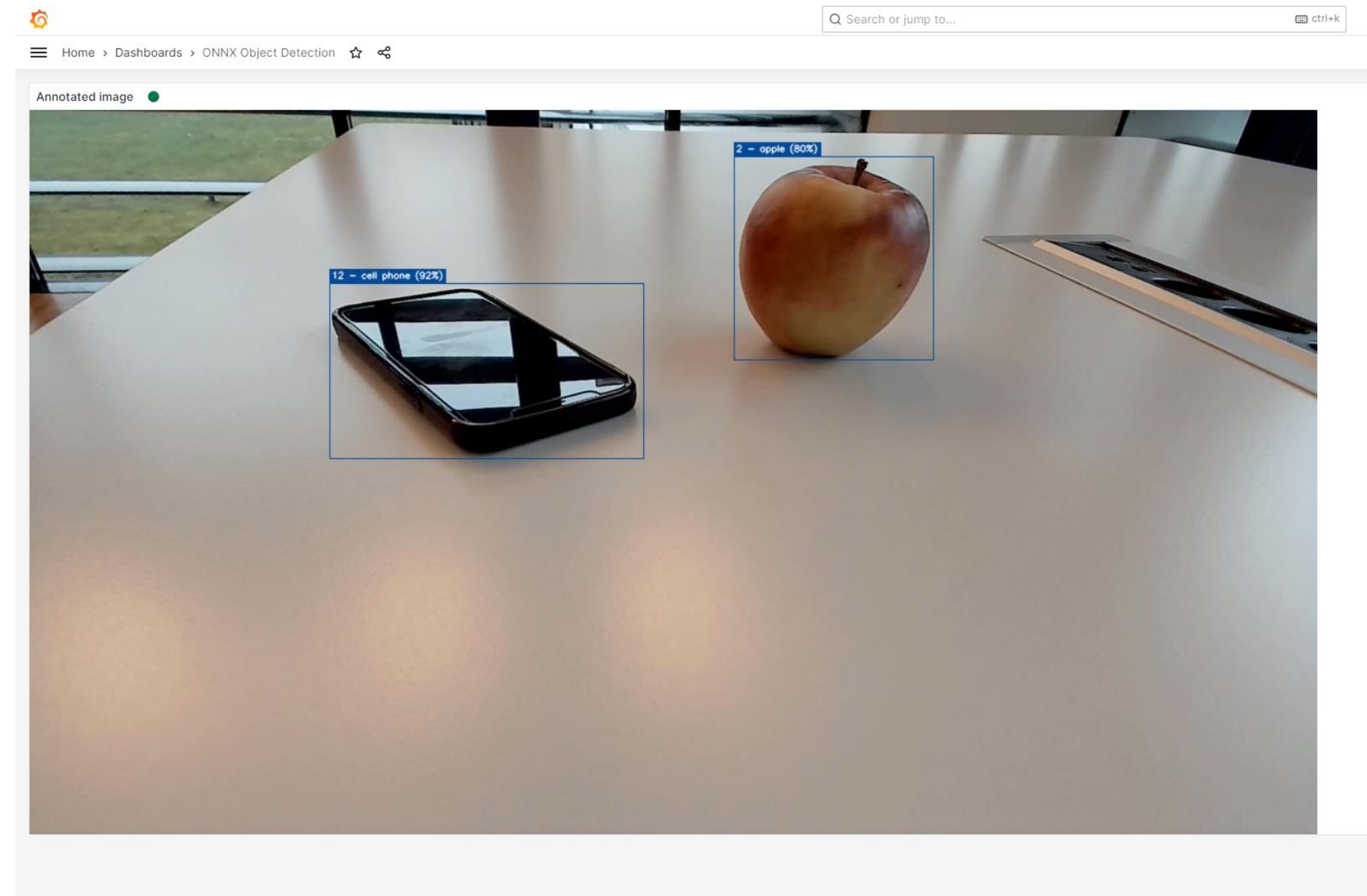
# Grafana integration

Deployed automatically with ESP on Azure Marketplace

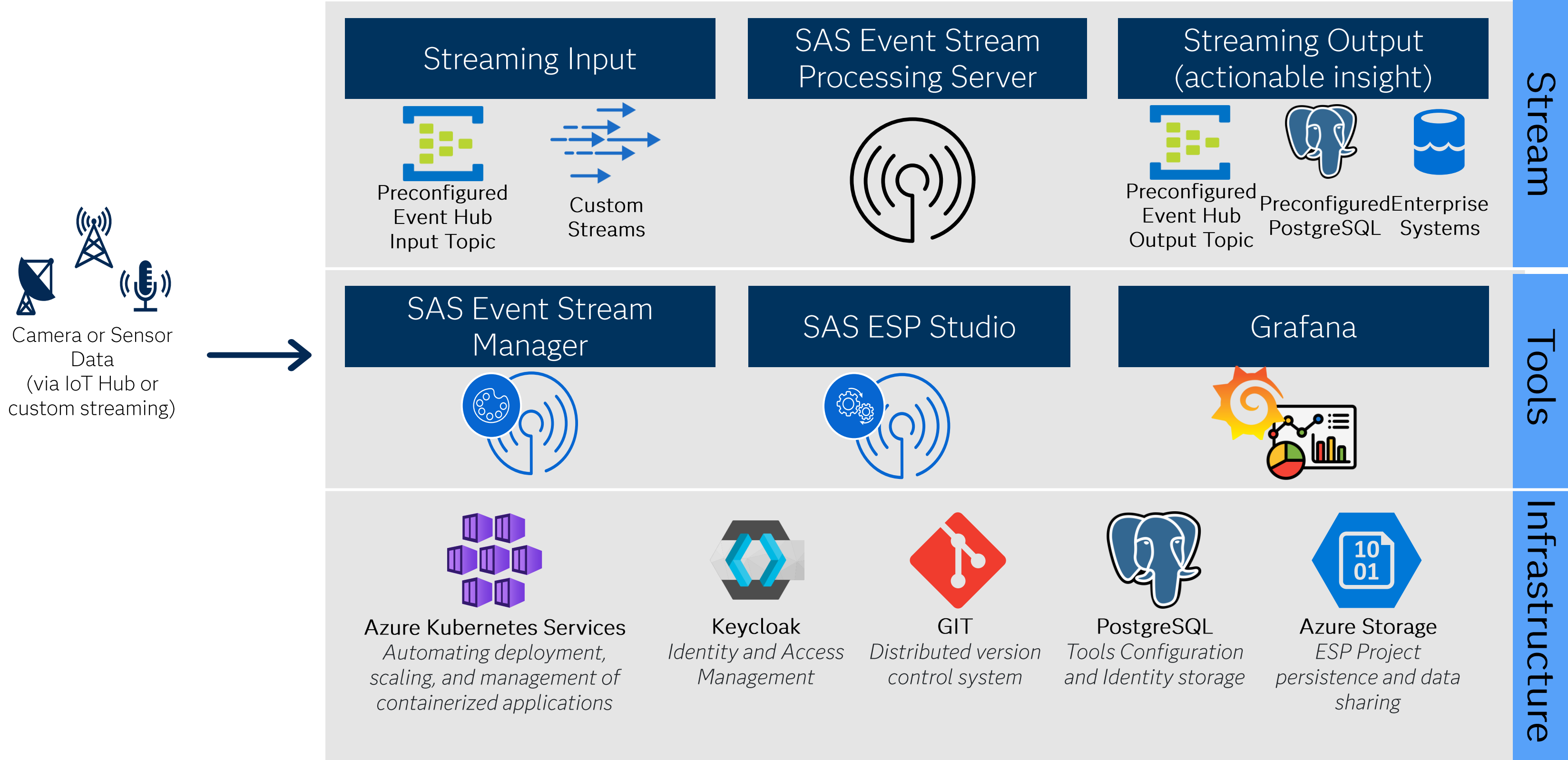


The Grafana ESP plugin...

- Allows to visualize and report on running ESP projects in Grafana
- Discovers ESP server pods in the cluster
- Makes efficient connections to the running projects
- Supports Computer Vision use cases



# Azure Marketplace Architecture



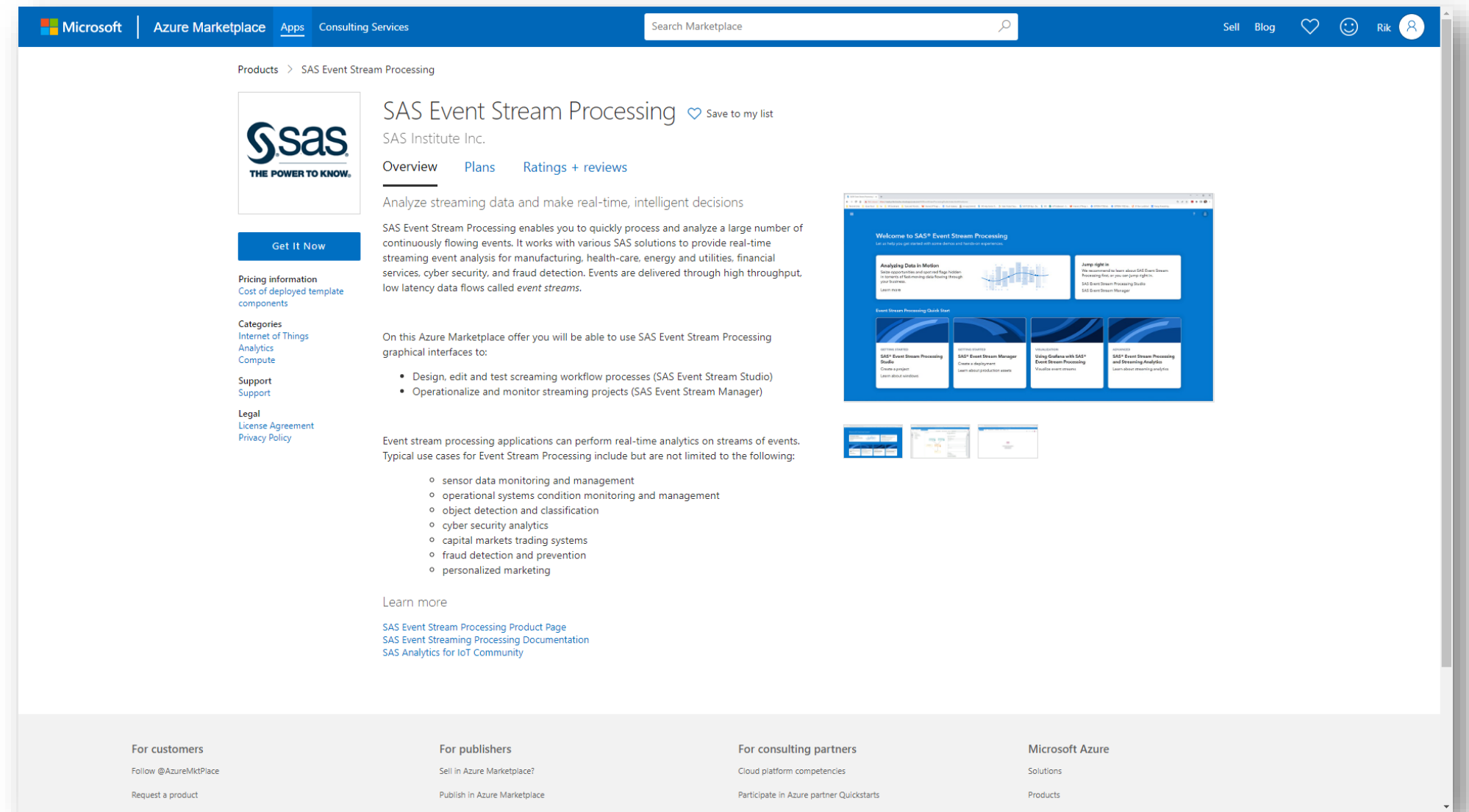


# SAS Event Stream Processing

Available on Azure Marketplace

You could deploy it with your existing license now!

For a trial license or for any question please write to [iotcontact@sas.com](mailto:iotcontact@sas.com)



The screenshot shows the Azure Marketplace page for SAS Event Stream Processing. The page includes the SAS logo, a 'Get It Now' button, pricing information, categories, support, and legal links. The main content area describes the product's capabilities for analyzing streaming data and making real-time decisions. It lists use cases such as sensor data monitoring, operational systems condition monitoring, object detection, cyber security, capital markets trading, fraud detection, and personalized marketing. The page also features a 'Learn more' section with links to the product page, documentation, and the SAS Analytics for IoT Community.

# References

Additional Materials

# Enablement & Support

ONNX example:

[https://github.com/sassoftware/esp-studio-examples/tree/main/Advanced/onnx\\_object\\_detection](https://github.com/sassoftware/esp-studio-examples/tree/main/Advanced/onnx_object_detection)

ONNX Model Zoo:

<https://github.com/onnx/models>

ESP Documentation:

[https://go.documentation.sas.com/doc/en/espcdc/v\\_047/espan/p0b1zsgwrsirbln1typkfoz428y9.htm](https://go.documentation.sas.com/doc/en/espcdc/v_047/espan/p0b1zsgwrsirbln1typkfoz428y9.htm)

ESP on Azure Marketplace:

[https://azuremarketplace.microsoft.com/en/marketplace/apps/sas-institute-560503.sas\\_esp?tab=Overview](https://azuremarketplace.microsoft.com/en/marketplace/apps/sas-institute-560503.sas_esp?tab=Overview)

Microsoft Azure Documentation:

<https://learn.microsoft.com/en-us/azure/?product=popular>

# THANK YOU

For more information, visit: [www.sas.com/esp](http://www.sas.com/esp)



sas.com

