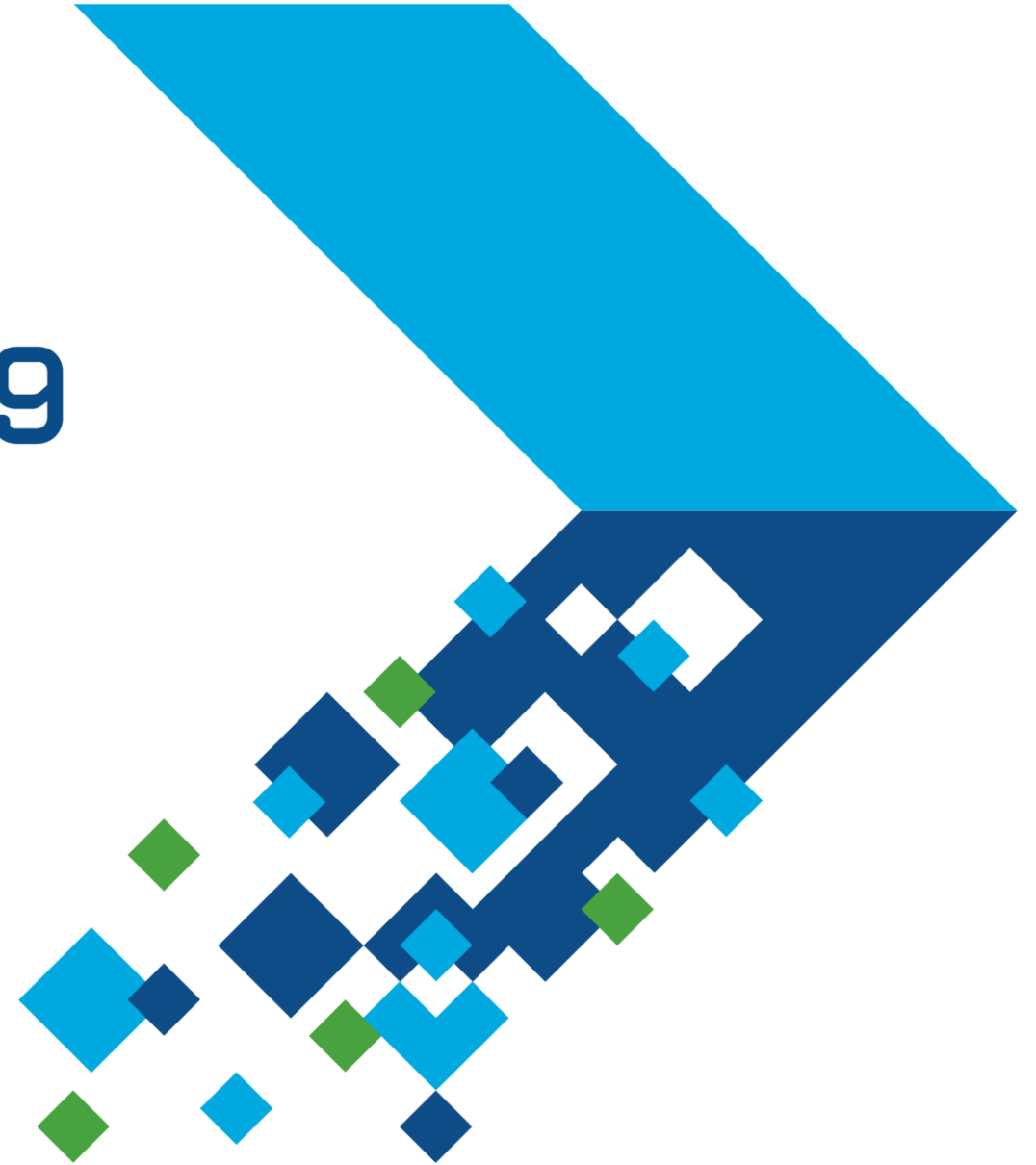


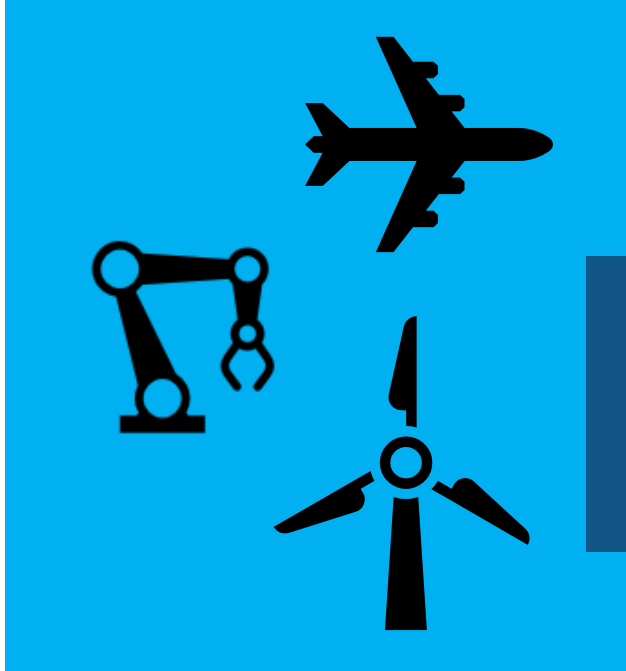
# MATLAB EXPO 2019

## Deploying AI for Near Real-Time Manufacturing Decisions

Antti Löytynoja, *Senior Application Engineer*



# The Need for Large-Scale Streaming Analytics



Jet engine: ~800TB per day  
Turbine: ~ 2 TB per day

## Predictive Maintenance

*Increase Operational Efficiency*  
*Reduce Unplanned Downtime*

**More applications require  
near real-time analytics**

## Medical Devices

*Patient Safety*  
*Better Treatment Outcomes*

## Connected Cars

*Safety, Maintenance*  
*Advanced Driving Features*



Car: ~25 GB per hour

## **Example Problem:** Develop a machine learning model to predict failures in industrial pumps

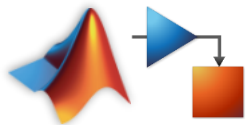
- We developed the machine learning model for the customer
- We wanted to go further:
  - Create a streaming application based on this real customer request
  - Develop application in a 3-4 week sprint
  - We believe this represents a realistic customer situation

# Our Project: Develop and operationalize a machine learning model to predict failures in industrial pumps



## Process Engineer

Develops models  
in MATLAB and  
Simulink



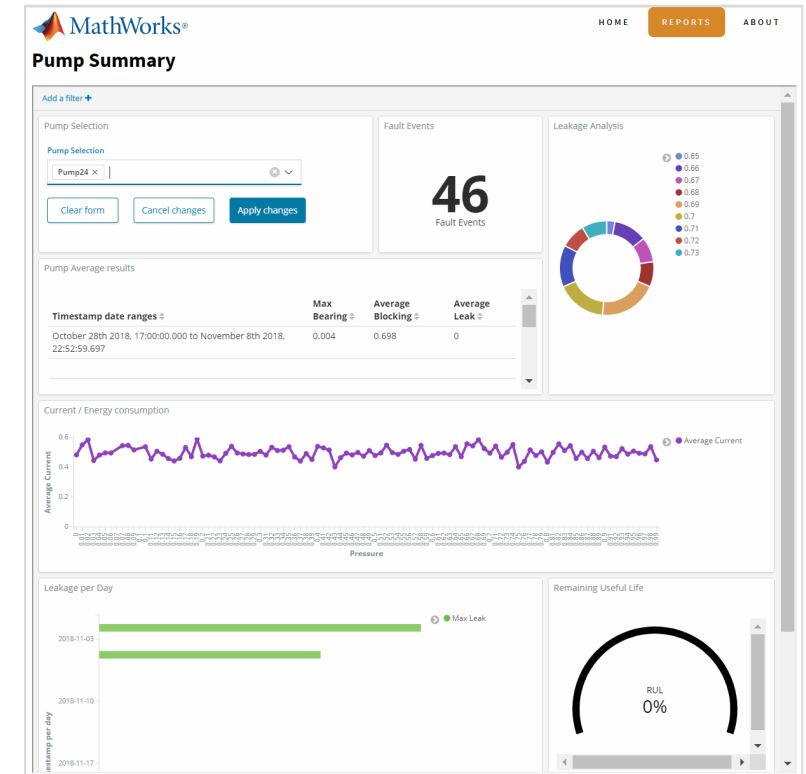
## System Architect

Deploys and  
operationalizes model  
on Azure cloud



## Operator

Makes operational  
decisions based  
on model output



Current system requires Operator to manually monitor operational metrics for anomalies. Their expertise is required to detect and take preventative action



## Project statement: Develop end-to-end predictive maintenance system and demo in one 3-4 week sprint



Plant  
Operator

1. Monitor *flow*, *pressure*, and *current* of each pump so I always know their *operational state*
2. Need to know which component is causing problems so that I can fix the **right issue**
3. Continuous estimate of each pump's *remaining useful life (RUL)* so I can *schedule maintenance or replace* the asset



# Challenges of AI Deployment



Process  
Engineer

We don't have a large set of failure data, and it's too costly to generate real failures in our plant for this project

**Solution:** Use an accurate physics-based software model for the pump to develop synthetic training sets

# Challenges of AI Deployment



**System  
Architect**

We don't have a large IT/hardware budget, and we need to see results before committing to a particular platform or technology

**Solution:** Leverage cloud platform to quickly configure and provision the services needed to build the solution, while minimizing lock-in to a particular provider



# Challenges of AI Deployment

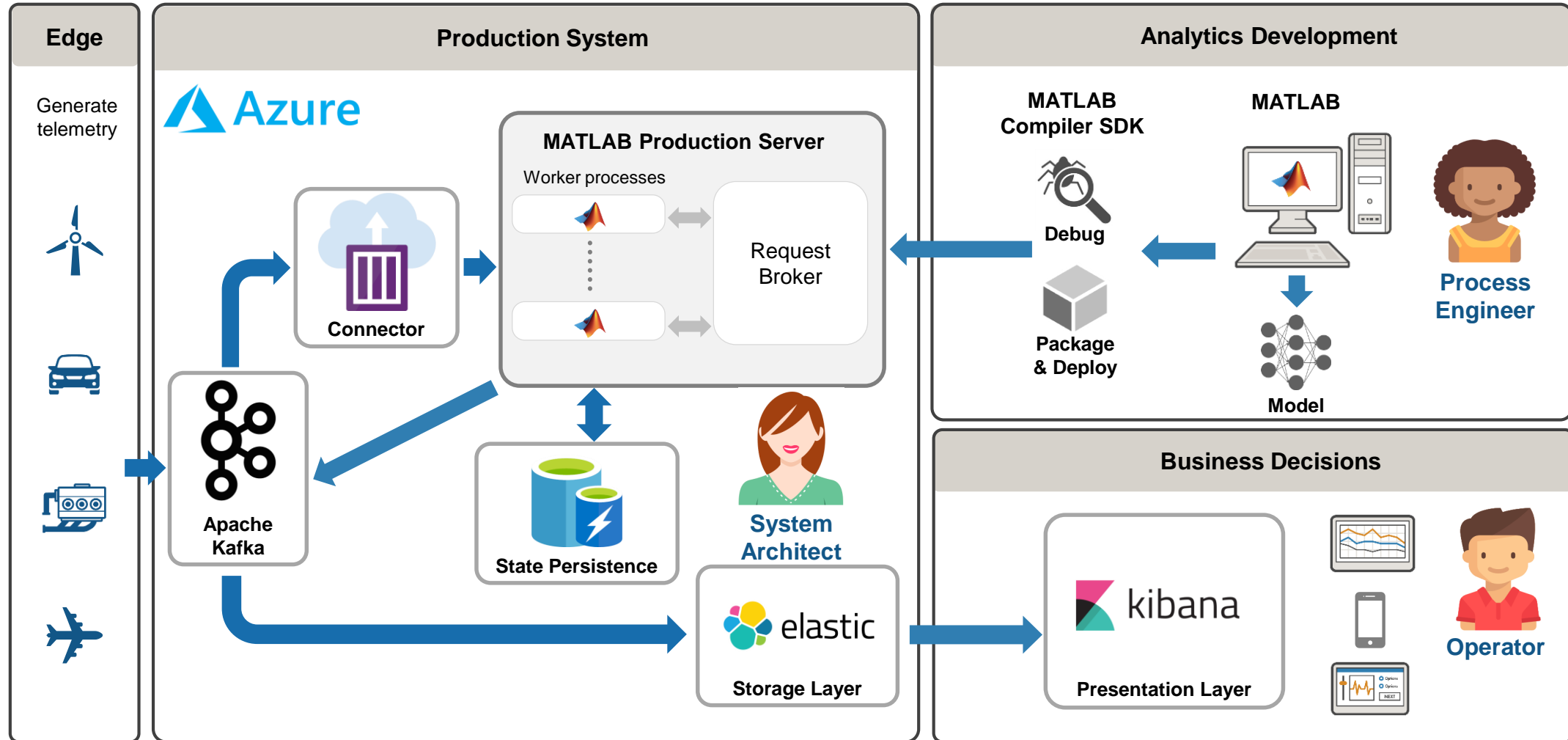


Process  
Engineer

Need software for multidisciplinary problem across teams, plus integration w/ IT

**Solution:** Use MATLAB and integrate with OSS

# Predictive Maintenance Architecture on Azure





Process  
Engineer

## Review model requirements



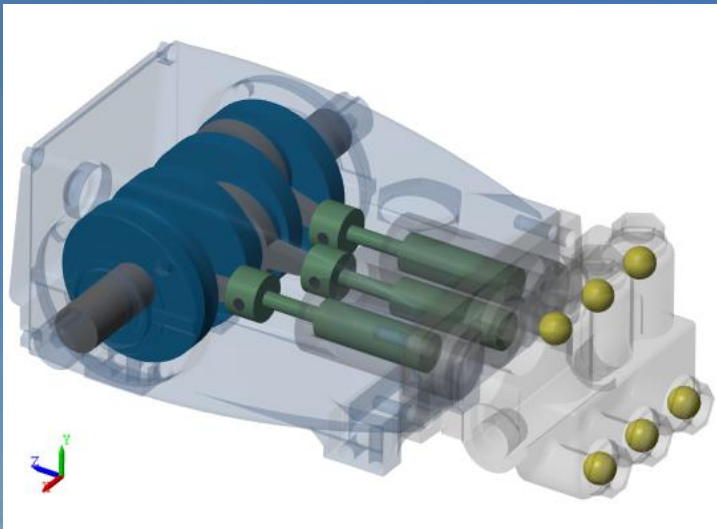
### Requirements From Operator

- Continuous predictions of type of fault
  - “Blocking”
  - “Leaking”
  - “Bearing”
  - Combination of above
- Continuous predictions of Remaining Useful Life [RUL]



### Requirements From System Architect

- Define window for streaming
- Define format of results, intermediate values
- Test code
- Scale code



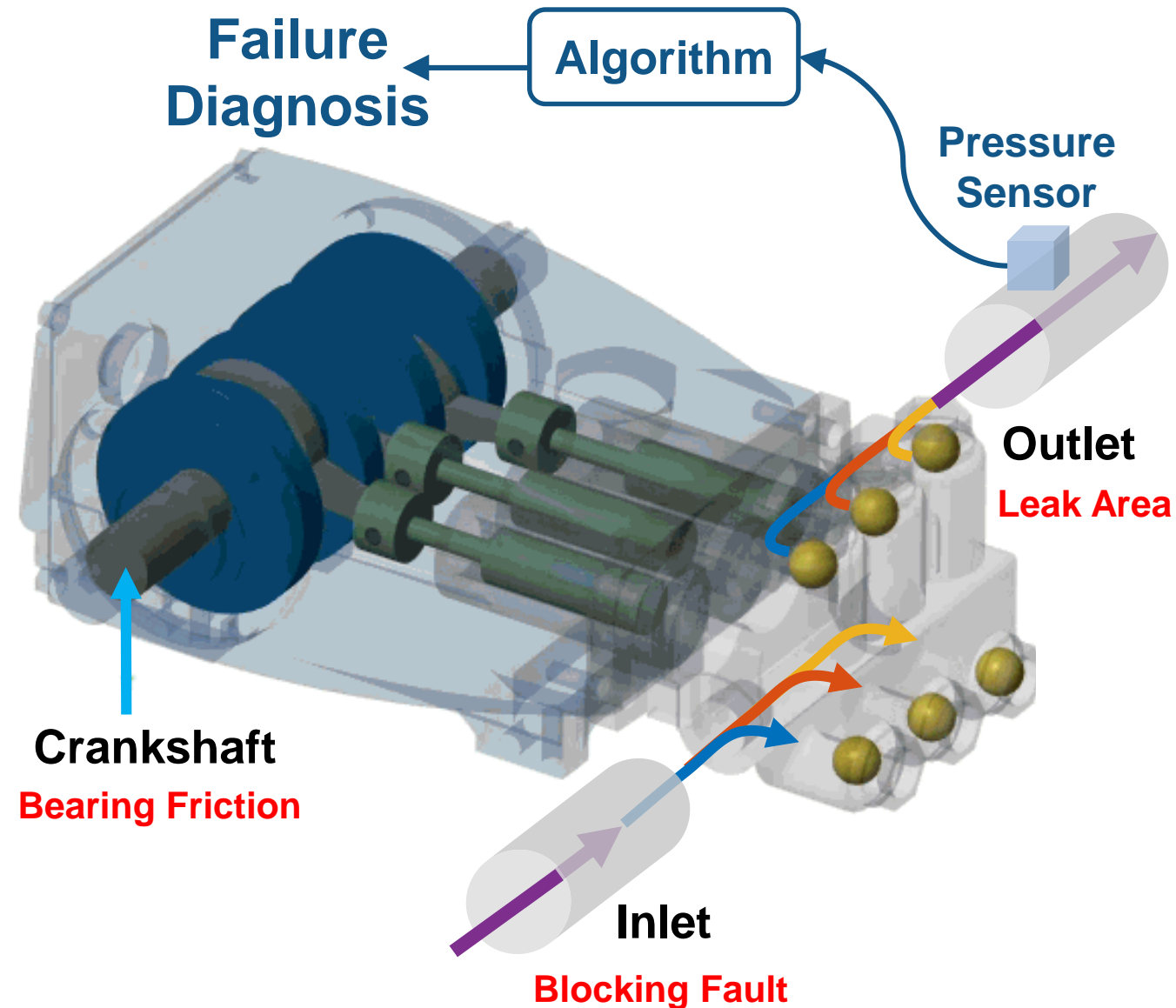
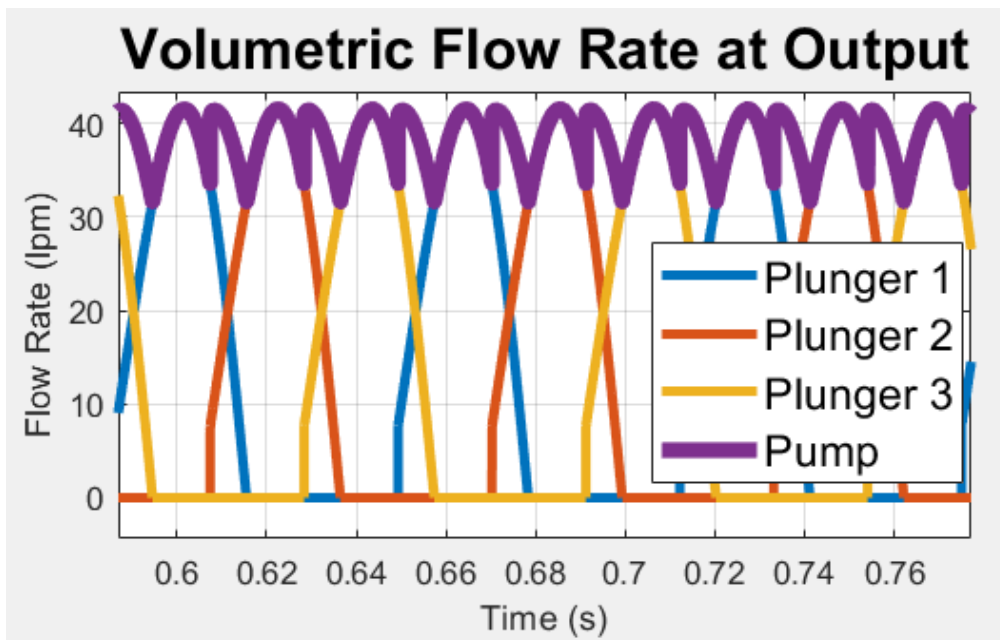




Process  
Engineer

# Physics of Triplex Pump

- Crankshaft drives three plungers
  - Each 120 degrees out of phase
  - One chamber always discharging
  - Three types of **failures**

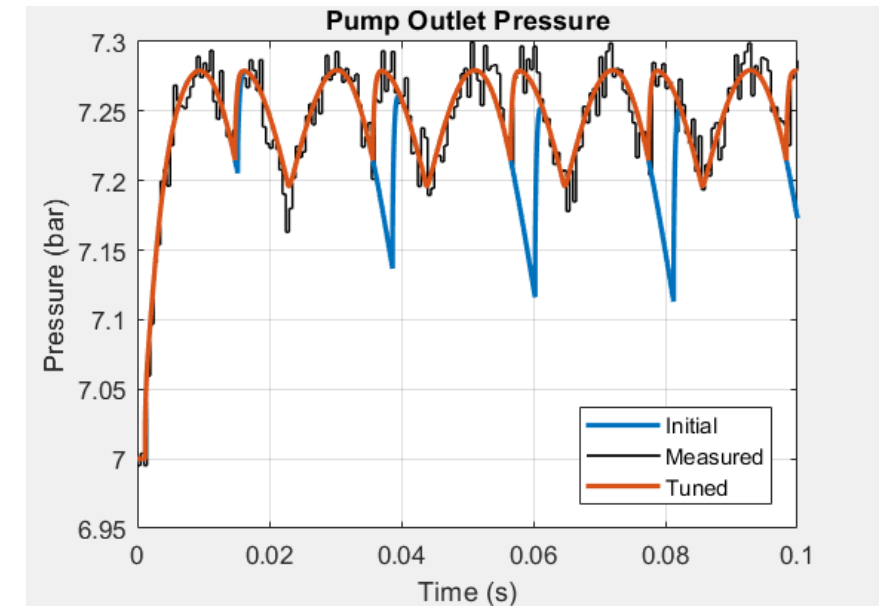
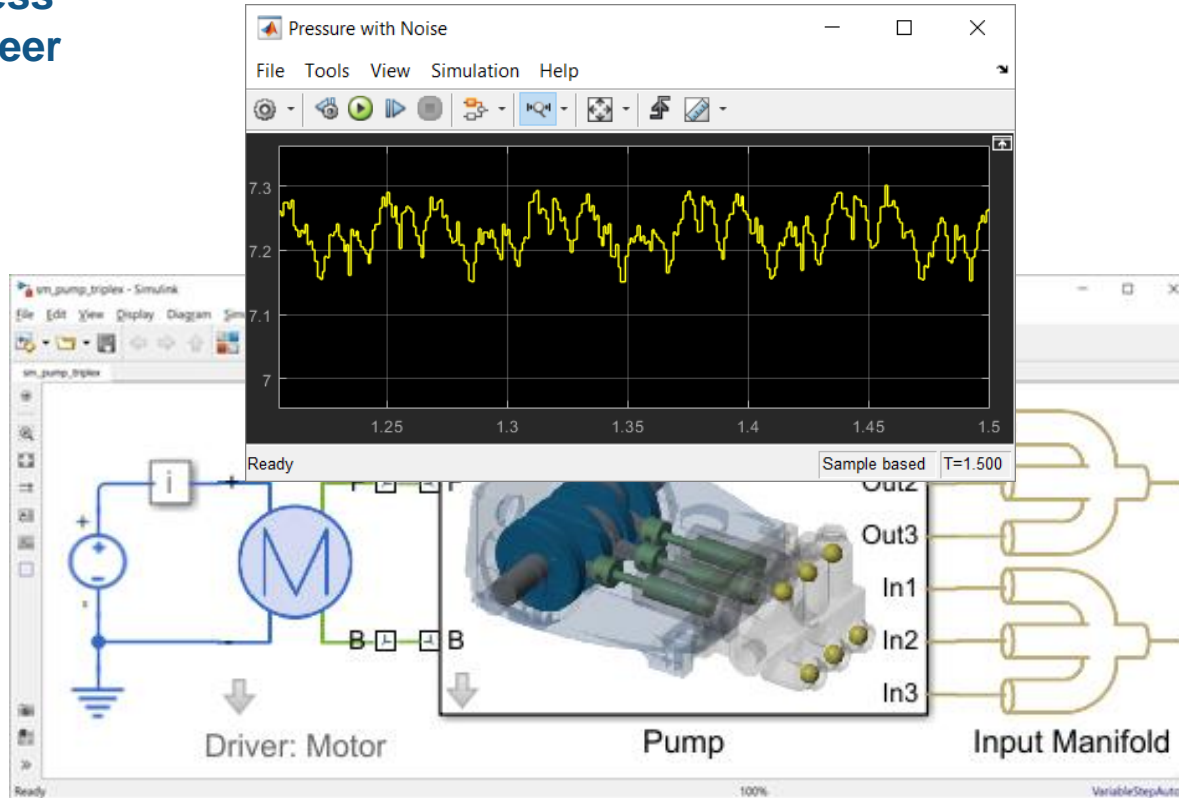




1

Access and Explore Data

Use sensor data to tune the digital twin

Process  
Engineer

Simulate faults

Pump sensor data

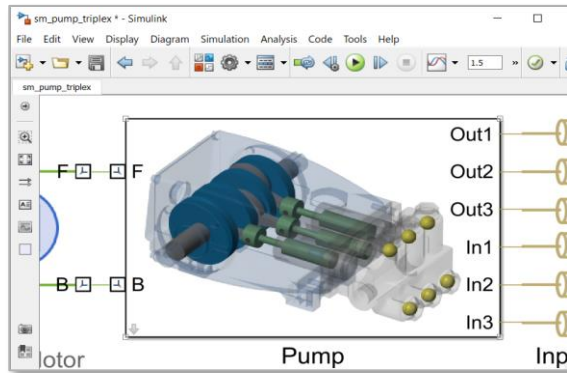


1

Access and Explore Data

Process  
Engineer

Simulate data with many failure conditions

**Leak Area = [1e-9 0.036]****Bearing Friction = [0 6e-4]****Blocking Fault = [0.5 0.8]**





2

Preprocess Data

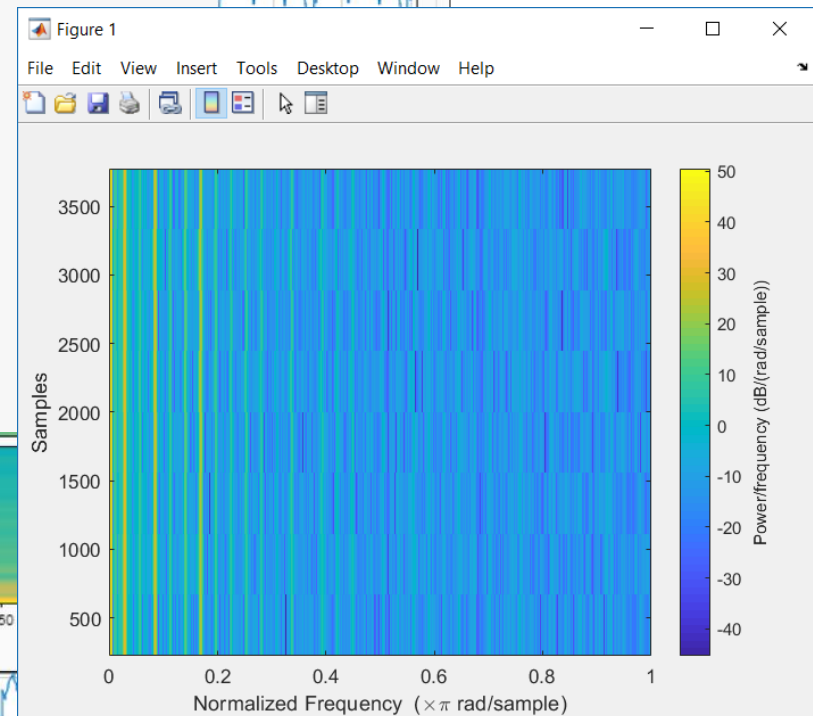
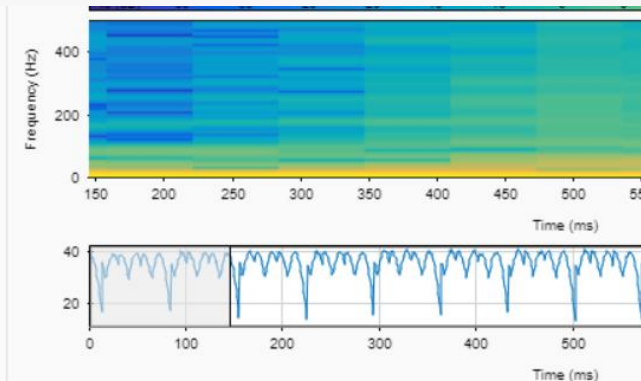
# Represent signal information

Process  
Engineer

## Signal processing

```
[Spectrum,Frequencies] = pspectrum(data.Flow);
[pLow,pHigh] = bounds(Spectrum);
fPeak = Frequencies(Spectrum==pHigh);
qPeak2Peak = peak2peak(data.Flow);
qCrest = peak2rms(data.Flow);
qRMS = rms(data.Flow);
qMAD = mad(data.Flow);
```

allfaults	1000×3	timetable
bearingPump	1000×3	timetable
blockedPu...	1000×3	timetable
healthyPump	1000×3	timetable
leakingPump	1000×3	timetable





3

Develop Predictive Models

# Develop Predictive Models in MATLAB

Process Engineer

2001x1 timetable	None
2001x1 timetable	Leak
2001x1 timetable	None
2001x1 timetable	Leak
2001x1 timetable	None
2001x1 timetable	Leak
2001x1 timetable	None

Label Faults

Scale

```
tt = tall(ds);
tt = preprocessData(tt);
model = TreeBagger(50,tt,'Event');
```

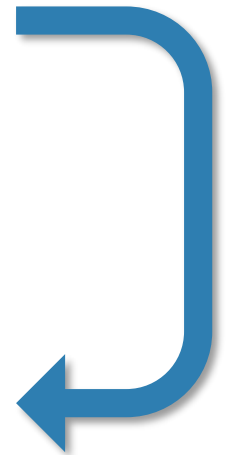
Evaluating tall expression using the Spark Cluster:

- Pass 1 of 2: Completed in 11 sec
  - Pass 2 of 2: Completed in 2.3333 min
- Evaluation completed in 2.6167 min

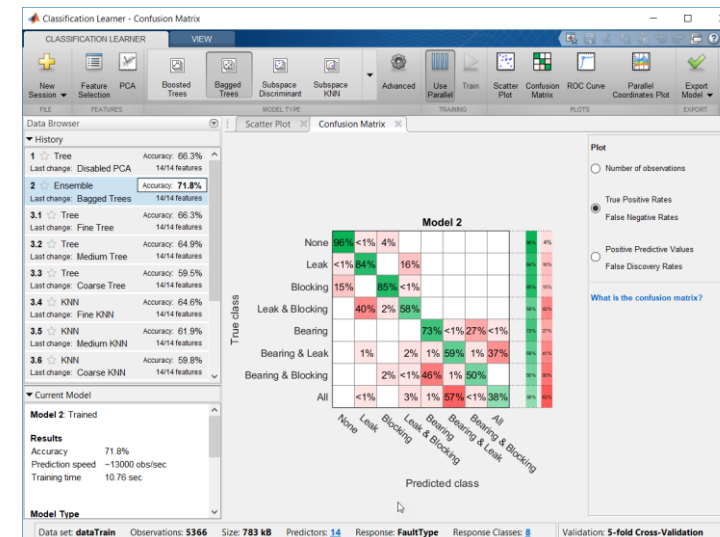


qVar	qSkewness	qKurtosis	FaultType
7.8516	-0.7236	2.6917	None
15.1621	-0.2604	2.4295	Leak
12.8083	-1.3068	5.2105	Blocking
7.5085	-0.7057	2.7118	None
7.5758	-0.7311	2.6837	Blocking
7.5540	-0.7157	2.7543	Blocking
13.1398	-1.2569	5.0460	Blocking
7.8751	-0.6167	2.5937	Blocking
18.4527	-1.7095	6.8370	Blocking

Represent Signals



Train Model



Validate Model

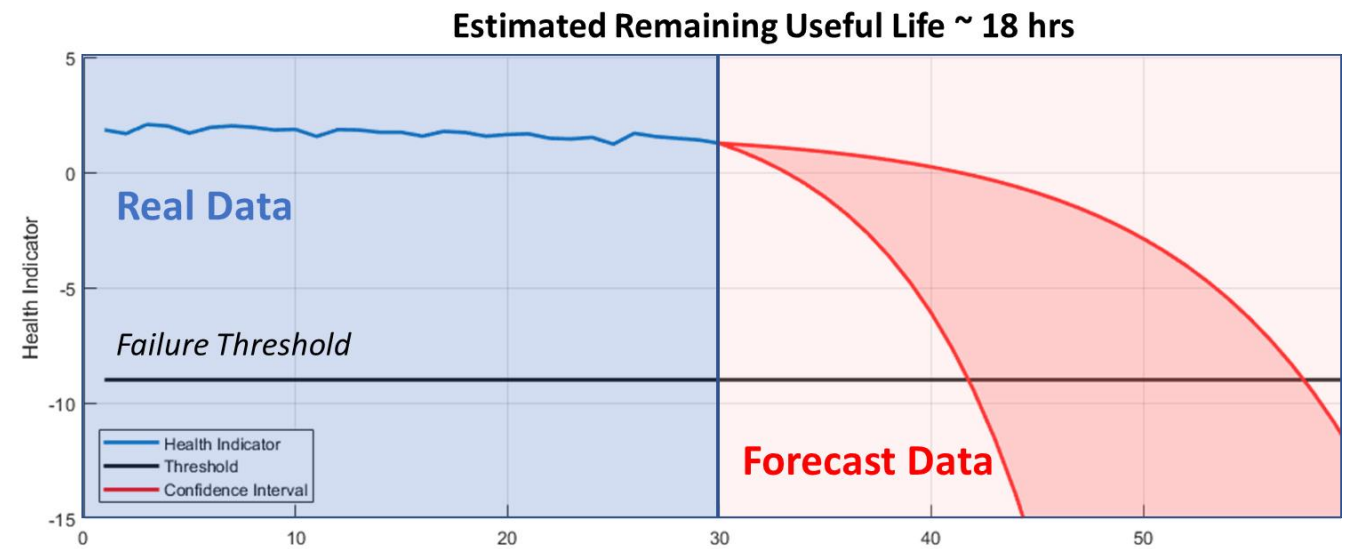
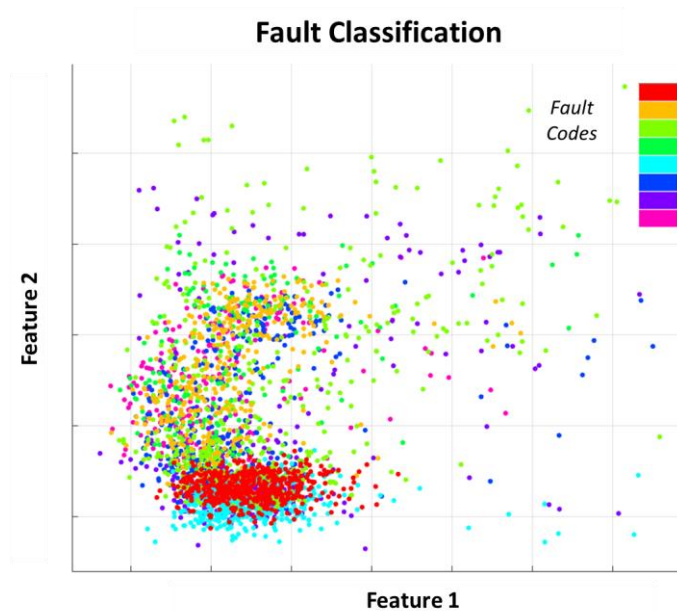


3

Develop Predictive Models

Process Engineer

# Develop Predictive Models in MATLAB



**Type of Fault  
(Classification)**



Plant Operator

**Remaining Useful Life  
(Regression)**

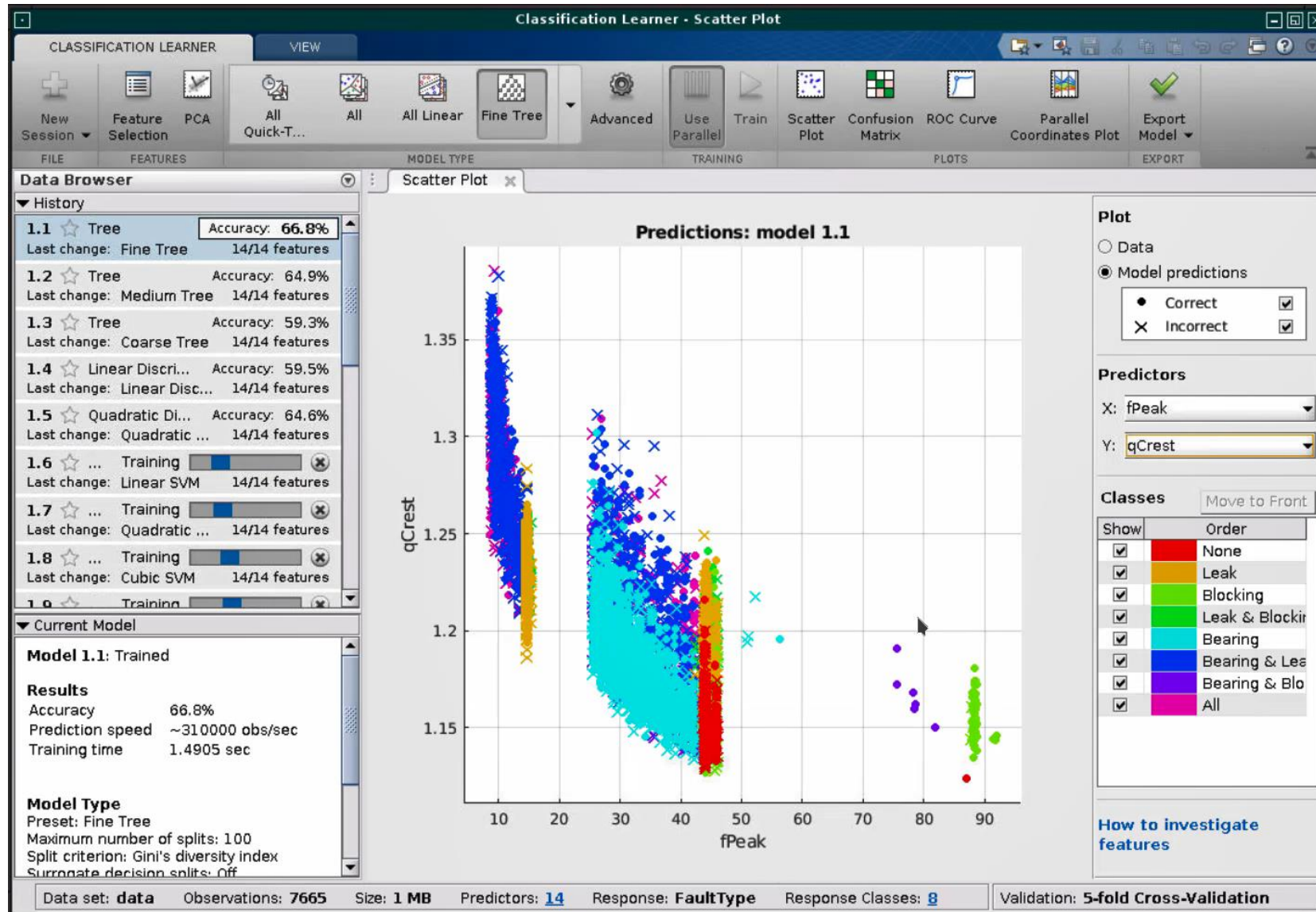


3

Develop Predictive Models

Process Engineer

# Develop Machine Learning Models

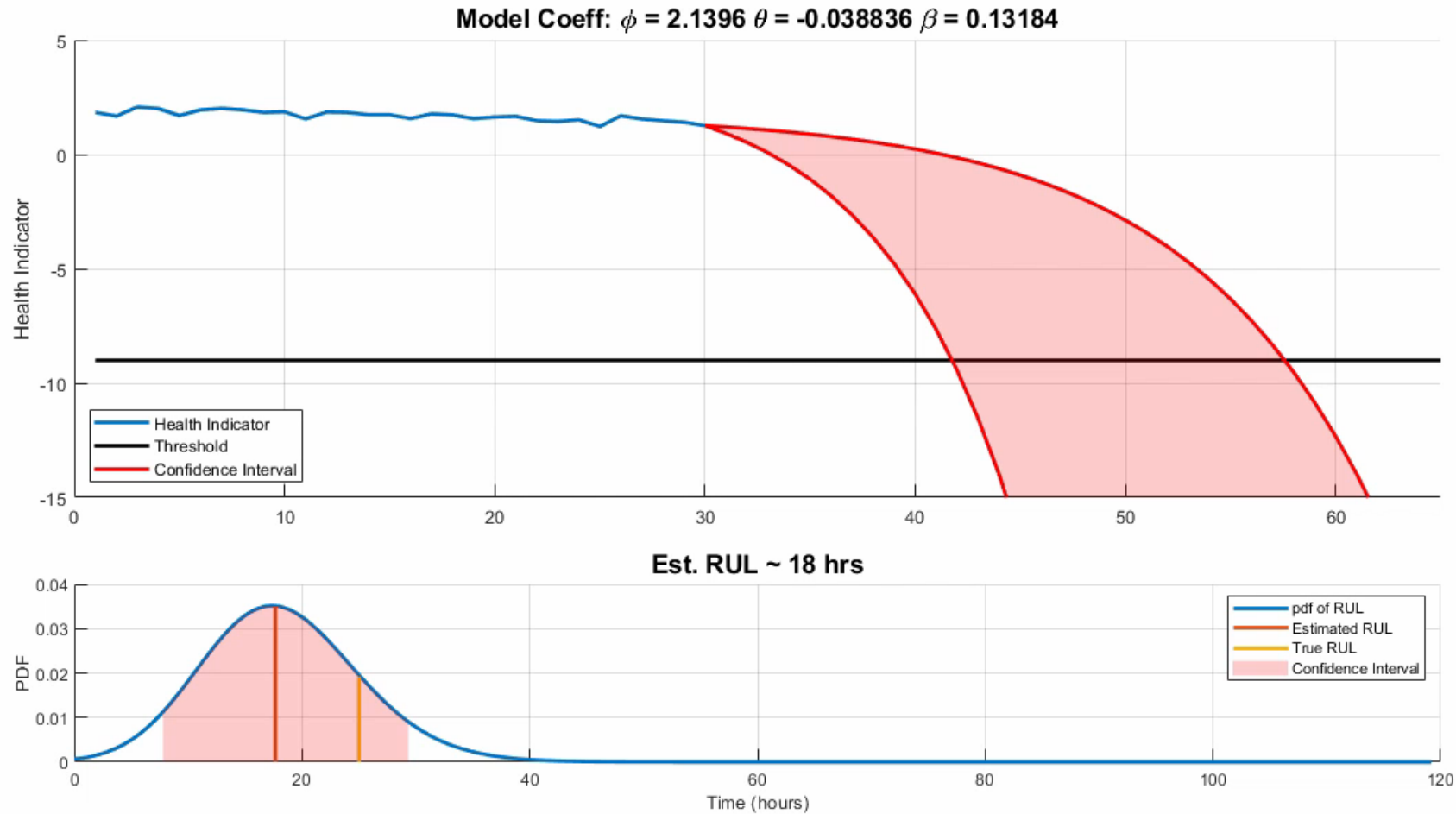




3

Develop Predictive  
ModelsProcess  
Engineer

# Estimate Remaining Useful Life



$$S(t) = \phi + \theta(t) e^{(\beta(t)t + \epsilon(t) - \frac{\sigma}{2})}$$

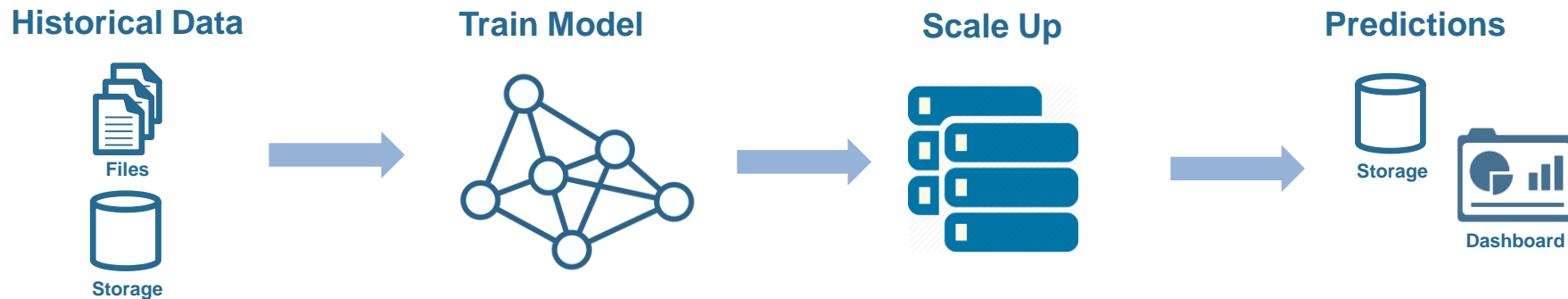


4

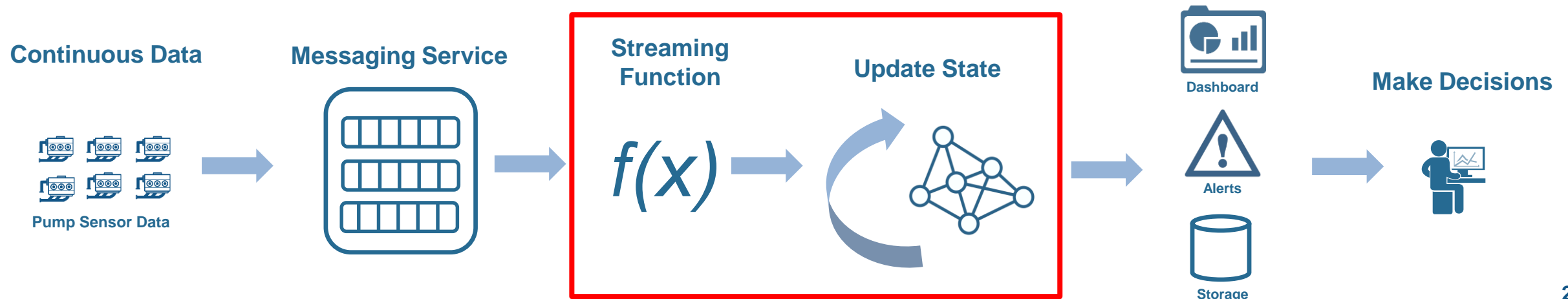
Integrate with  
Production  
SystemsProcess  
Engineer

# Develop a Stream Processing Function

- **Batch Processing:** Build and test model on simulated data



- **Stream Processing:** Apply model to sensor data in near real-time







Process  
Engineer

4

Integrate with  
Production  
Systems

## Develop a Stream Processing Function

### Streaming Function

```
function new_state = streamingFunction(data,old_state)
```

Process each window of  
data as it arrives

### Preprocess signals

```
[data,features] = preprocessData(data);
```

### Predict faults

```
[Leak,Blocking,Bearing] = predictFaultValues(features);  
FaultType = predictFault(features);  
[RUL,Model] = predictUpdateRUL(data.Timestamp,data.Flow,500);
```

### Update state

```
new_state = updateState(data,old_state);
```

### Write results

```
writeResults(Leak,Blocking,Bearing,FaultType,RUL,Model)  
end
```

Previous state

Current window of data to  
be processed



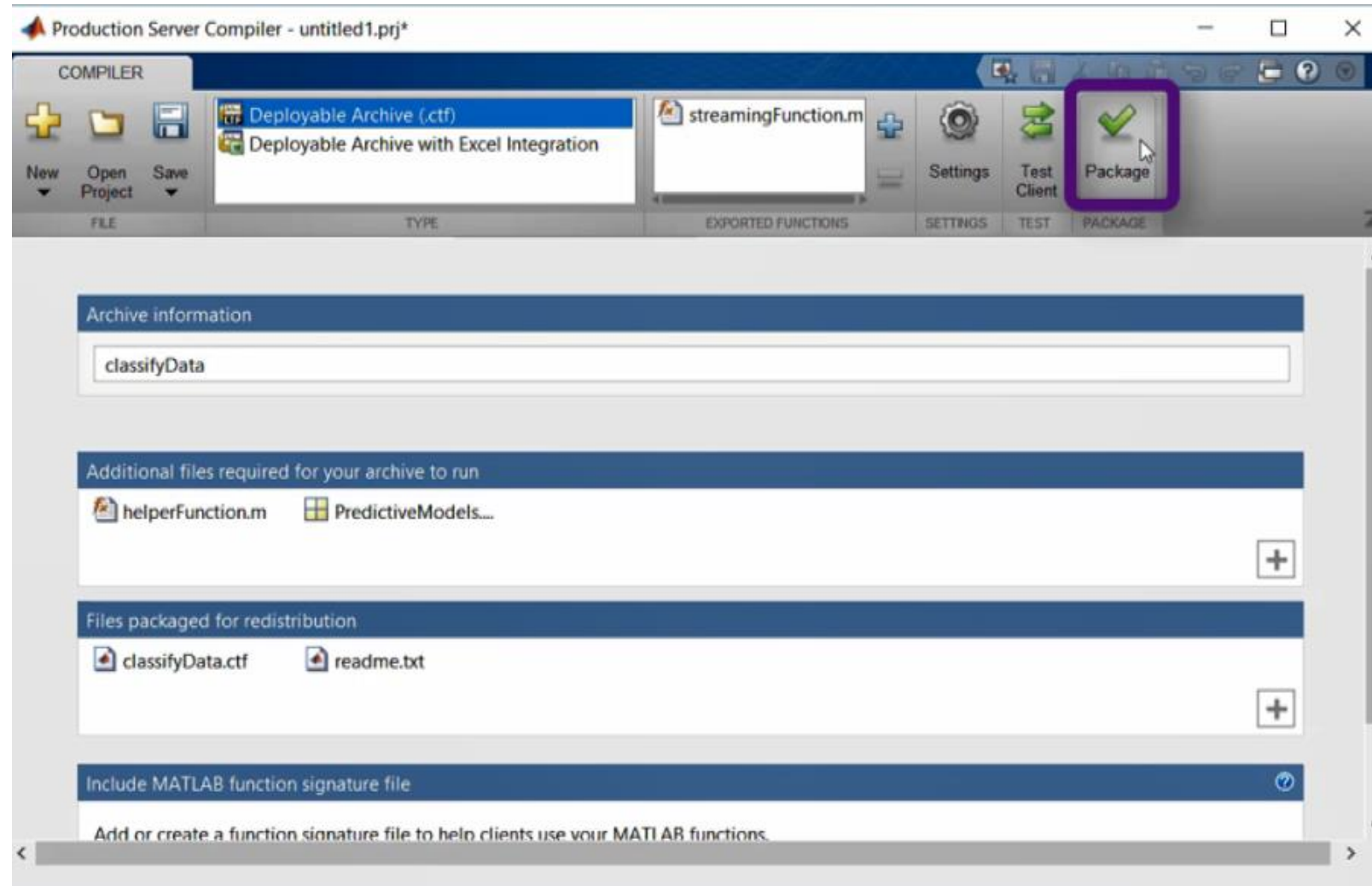


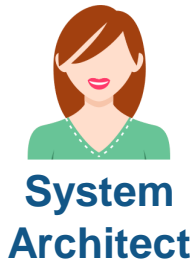
4

Integrate with  
Production  
Systems

Process  
Engineer

# Package Stream Processing Function





4

Integrate with  
Production  
Systems

## Review System Requirements

- Requirements from the Process Engineer
  - Every millisecond, each pump generates a time-stamped record of flow, pressure, and current
  - Model expects 1 sec. window of data per pump
  - Initially, 1's – 10's of devices, but quickly scale to 100's
- Requirements from the Operator
  - Classification of fault type
  - Continuous estimating of RUL for each pump



**Process Engineer**



**Operator**

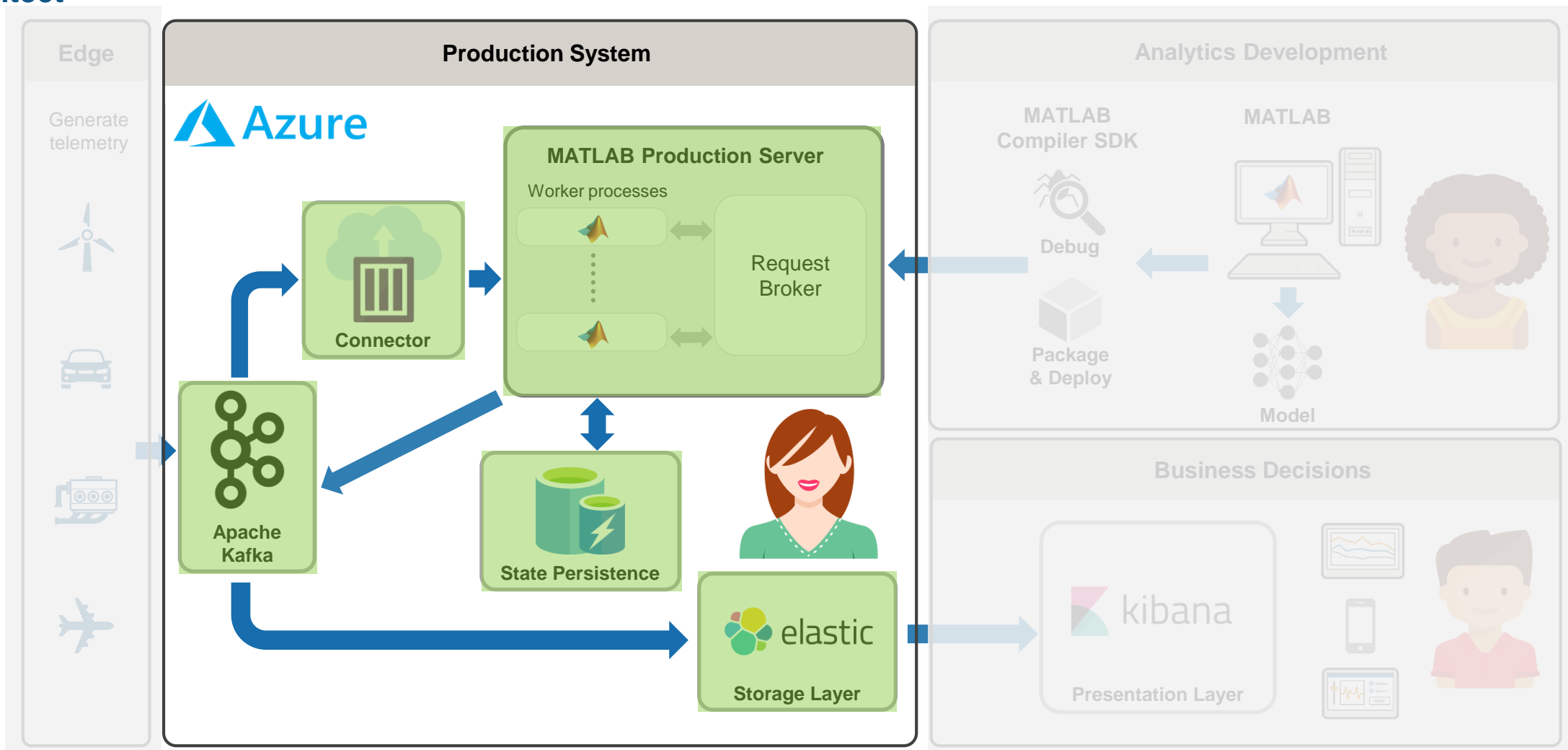


4

Integrate with  
Production  
Systems

System  
Architect

# Integrate Analytics with Production Systems

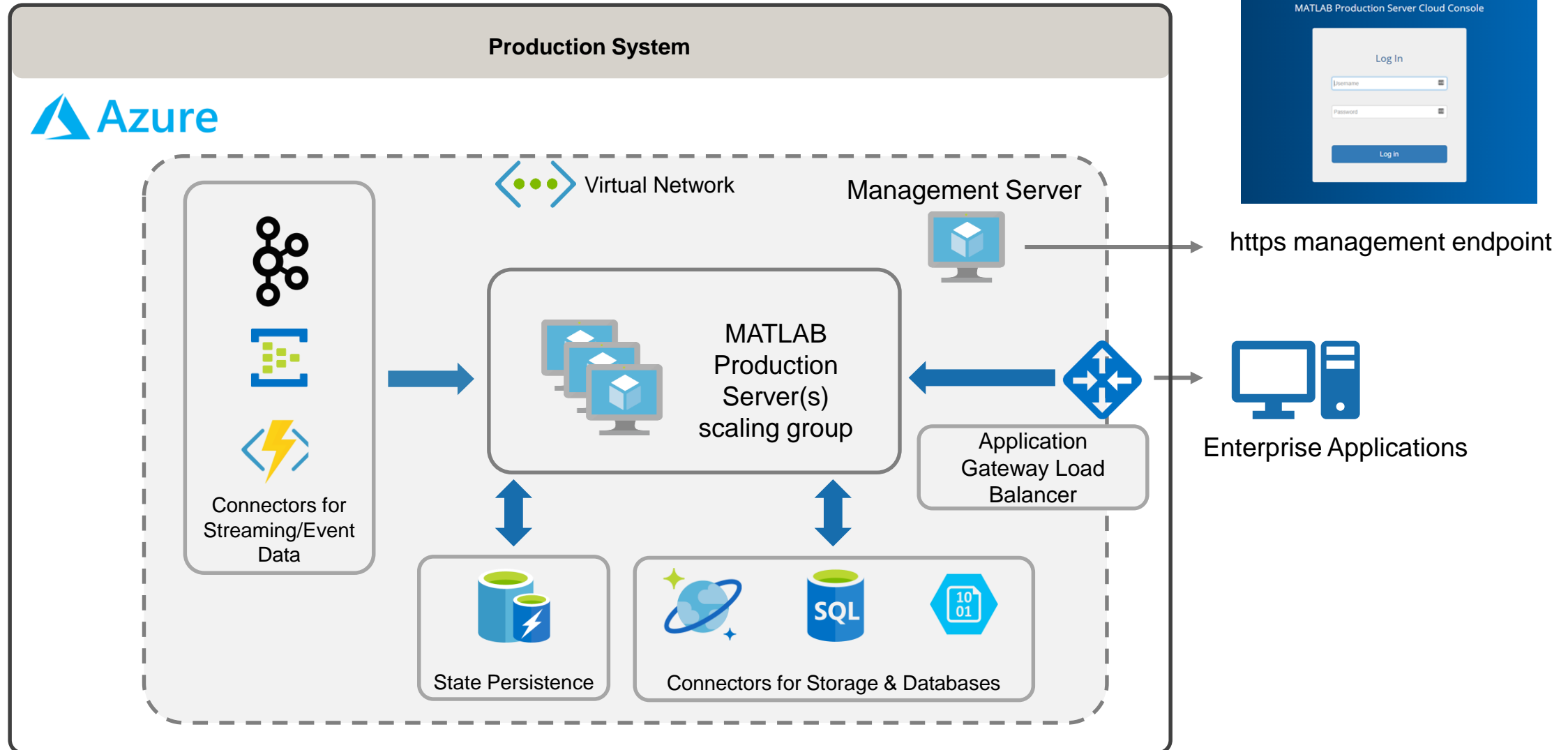




4

Integrate with  
Production  
Systems

# MATLAB Production Server on Azure

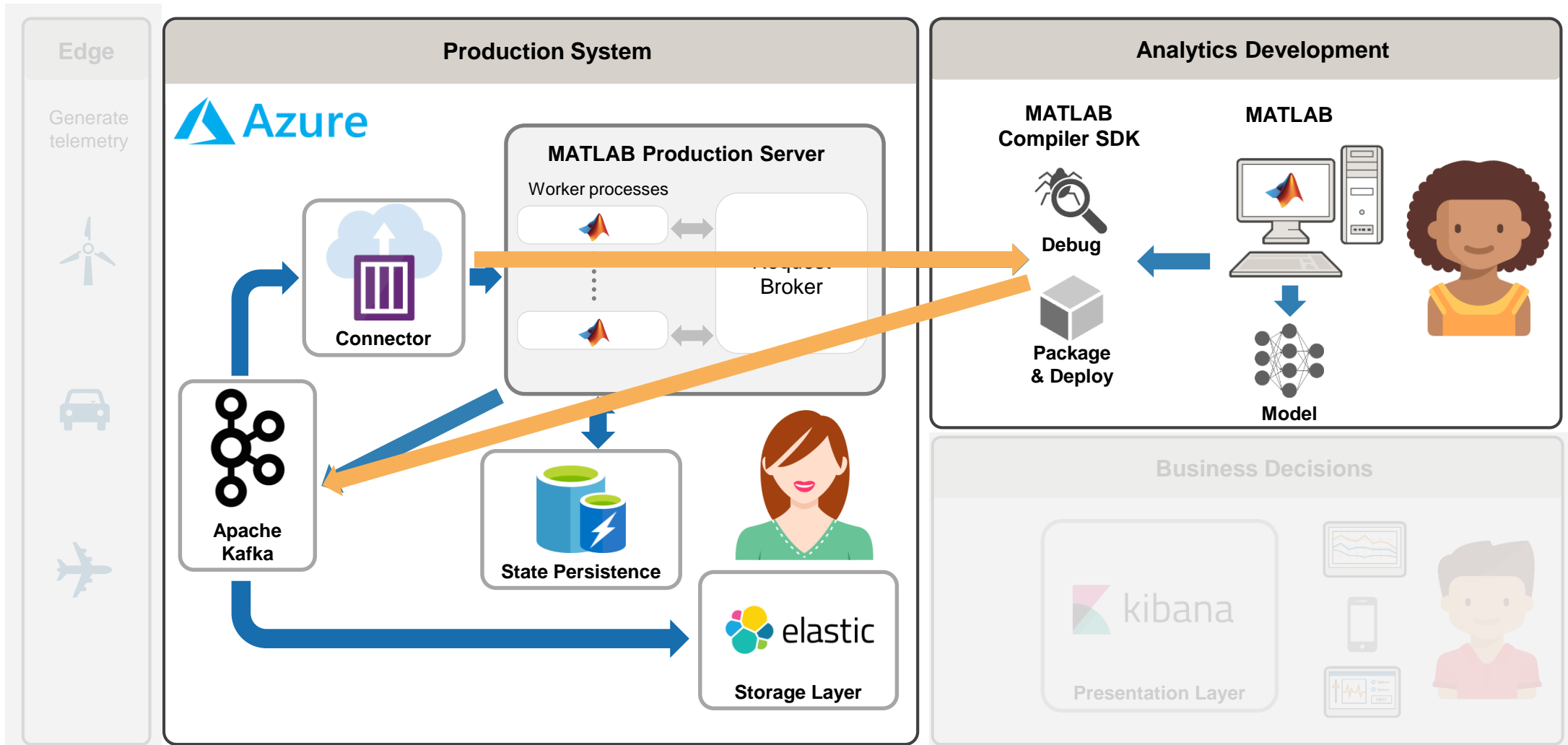




4

Integrate with  
Production  
Systems

# Debug your streaming function on live data

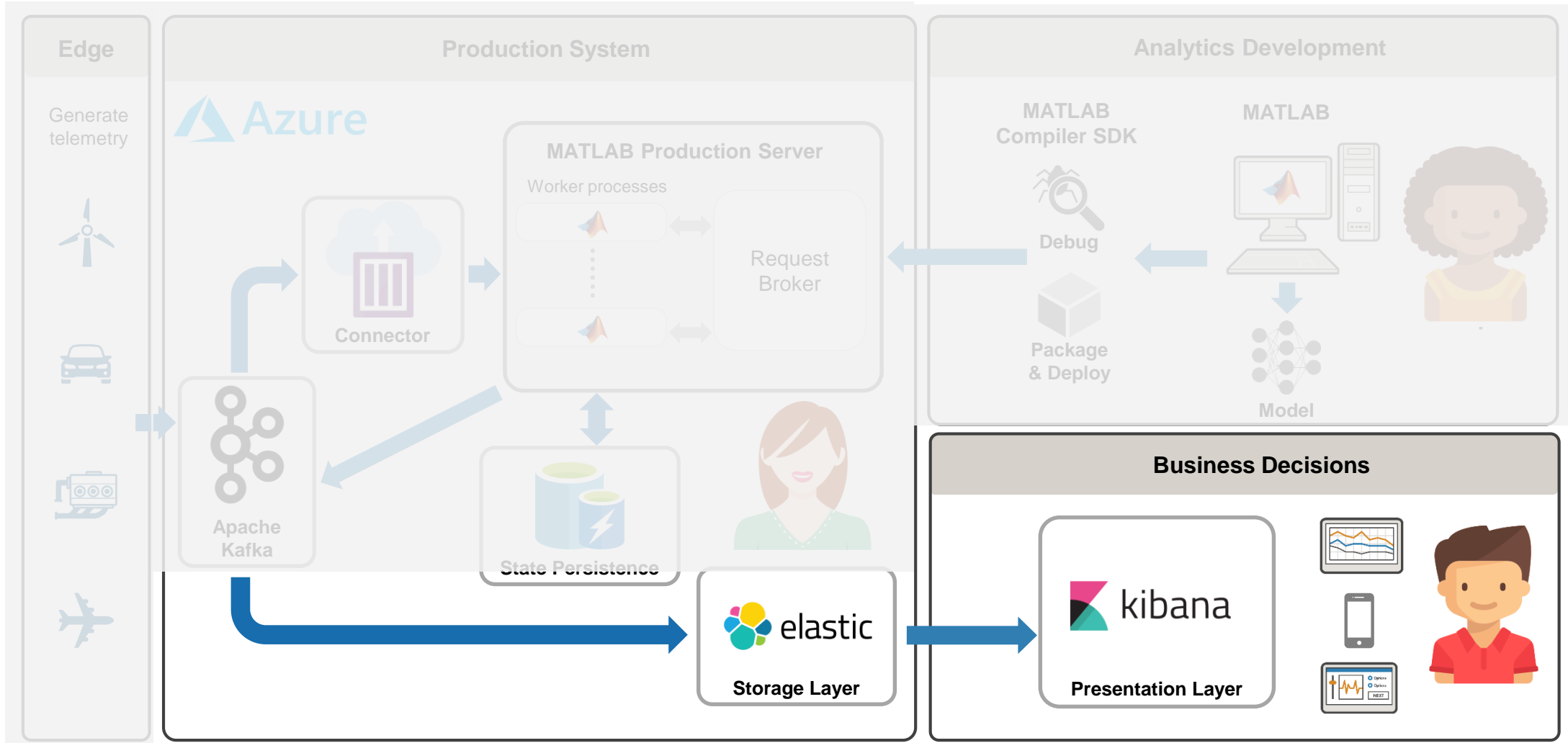




4

**Integrate with  
Production  
Systems**

# Complete your application



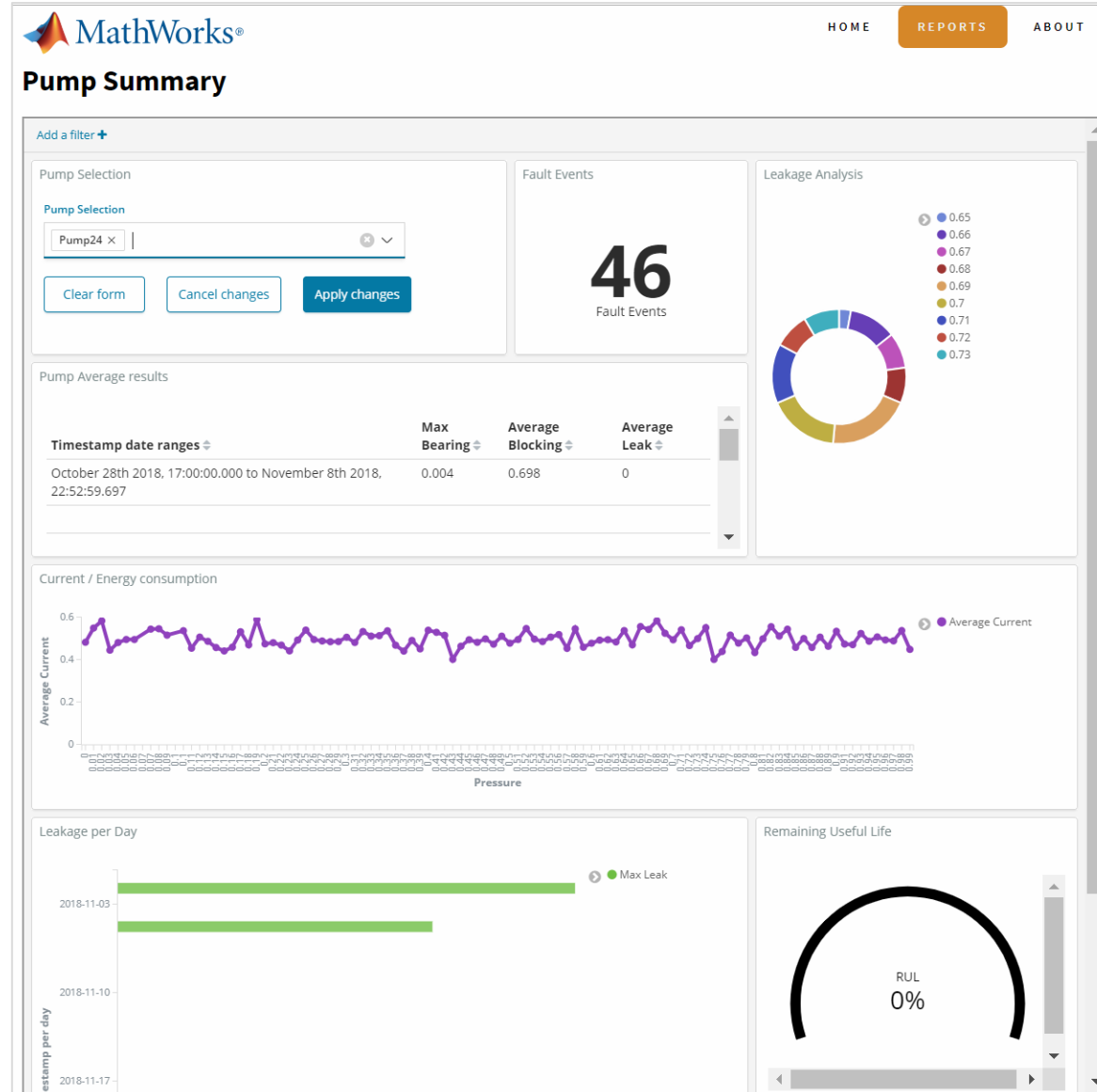


5

Visualize Results

Plant  
Operator

# Complete Your Application



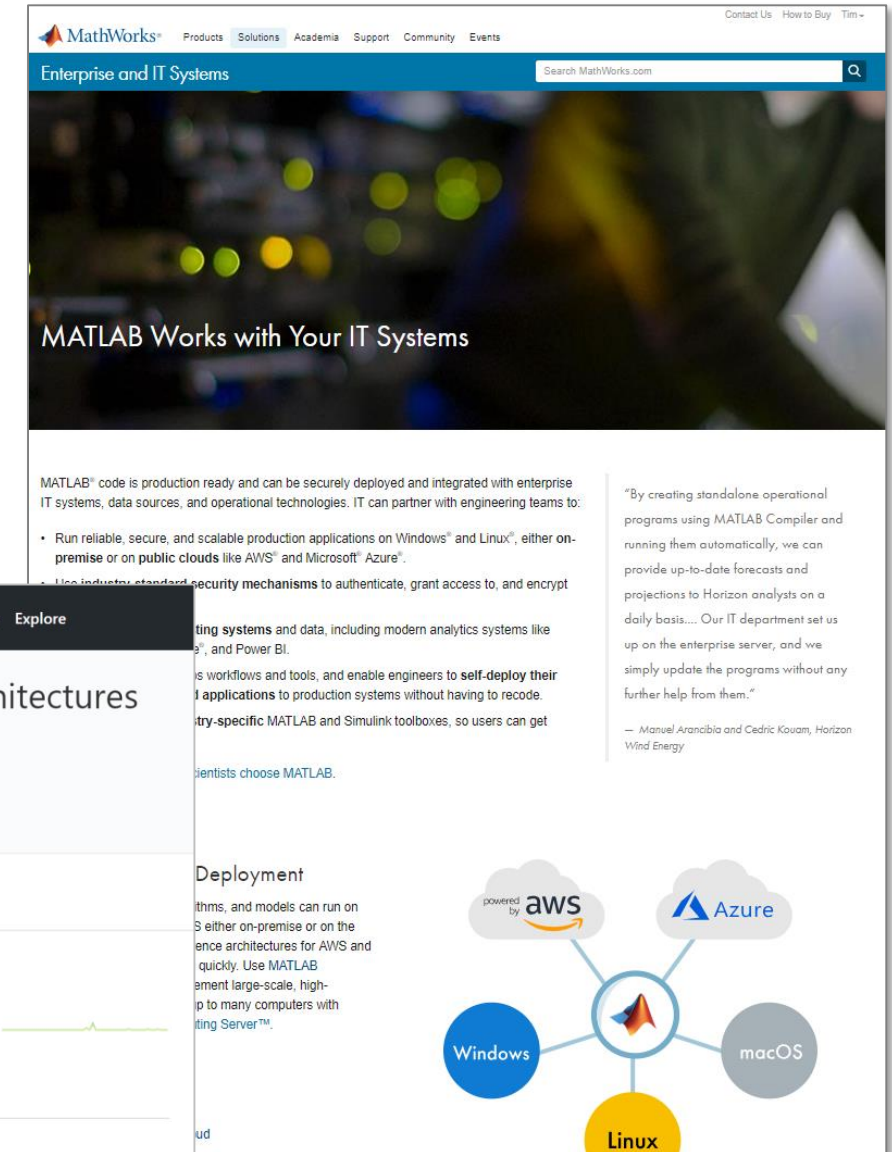


## Team Retrospective

- Completed demo of full system in 3 week sprint
- Successfully used digital twin to generate faults and train models
- Fast prototyping of physical and AI models with MATLAB and Simulink.  
Easy integration with OSS
- Cloud platform enabled faster IT setup
- Next steps:
  - Make model adjustments
  - Test against real pump
  - Customize dashboard for Operator's needs

# Resources to learn and get started

- [GitHub: MathWorks Reference Architectures](#)
- [Working with Enterprise IT Systems](#)
- [Data Analytics with MATLAB](#)
- [Simulink](#)



The image shows two overlapping screenshots. The top screenshot is from the MathWorks website, specifically the 'Enterprise and IT Systems' section. It features a dark background with yellow and green bokeh lights. The text 'MATLAB Works with Your IT Systems' is prominently displayed. Below this, there is a paragraph about MATLAB's production readiness and a list of bullet points. The bottom screenshot is from a GitHub repository titled 'MathWorks Reference Architectures'. It shows the repository's overview page with a search bar, filters for repositories, people, and projects, and a list of repositories including 'mdcs-on-azure' and 'mps-on-aws'. To the right of the GitHub screenshot is a diagram titled 'Deployment' showing a central MATLAB logo connected to five nodes: 'powered by aws', 'Azure', 'macOS', 'Linux', and 'Windows'.

**MathWorks Reference Architectures**

Reference Architectures  
https://mathworks.com/cloud Verified

Repositories 6 People 1 Projects 0

Find a repository... Type: All Language: All

**mdcs-on-azure**  
Stand up a MATLAB Distributed Computing Server cluster using Azure Deployment  
PowerShell ★ 2 Updated 25 days ago

**mps-on-aws**  
Stand up a MATLAB Production Server using CloudFormation

**Deployment**

Algorithms, and models can run on S either on-premise or on the cloud. Use MATLAB to create reference architectures for AWS and Azure. Quickly. Use MATLAB to implement large-scale, high-performance computing across many computers with MATLAB Server™.

powered by aws Azure macOS Linux Windows

“By creating standalone operational programs using MATLAB Compiler and running them automatically, we can provide up-to-date forecasts and projections to Horizon analysts on a daily basis.... Our IT department set us up on the enterprise server, and we simply update the programs without any further help from them.”  
— Manuel Arancibia and Cedric Kouam, Horizon Wind Energy