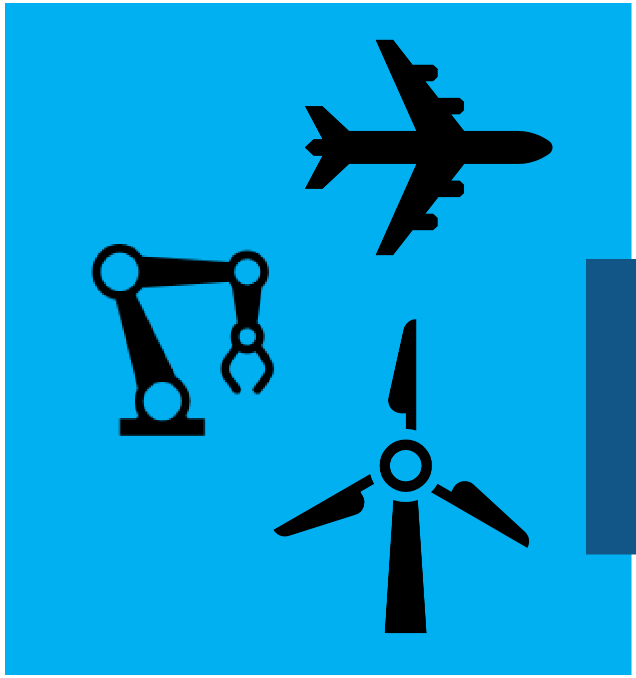
MATLAB EXPO 2018

Deploying AI for Near Real-Time Manufacturing Decisions

Arvind Hosagrahara
Heather Gorr, PhD

The Need for Large-Scale Streaming



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

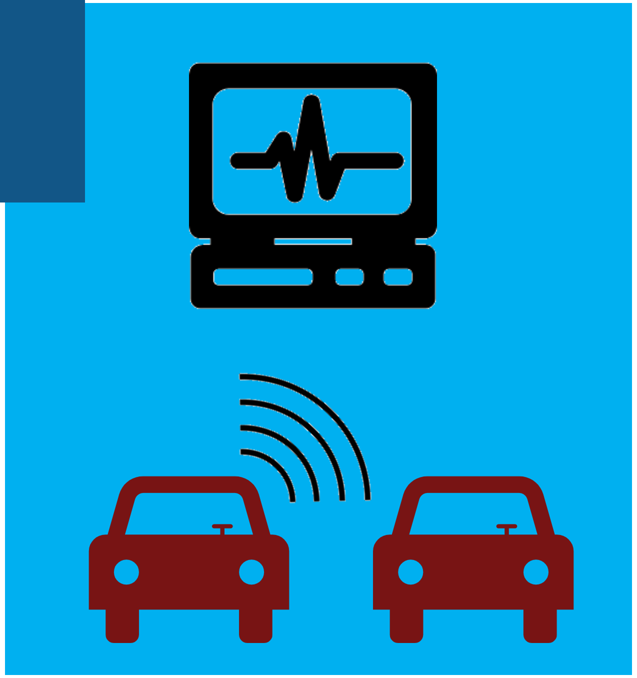
Patient Safety

Better Treatment Outcomes

Connected Cars

Safety, Maintenance

Advanced Driving Features



Car: ~25 GB per hour

Example Problem: Develop and operationalize a machine learning model to predict failures in industrial pumps

The current system requires the operator to manually monitor the operational metrics for anomalous conditions. It is dependent on experience and expertise to detect and take preventative action

Process Engineer

Develops machine learning model in MATLAB

System Architect

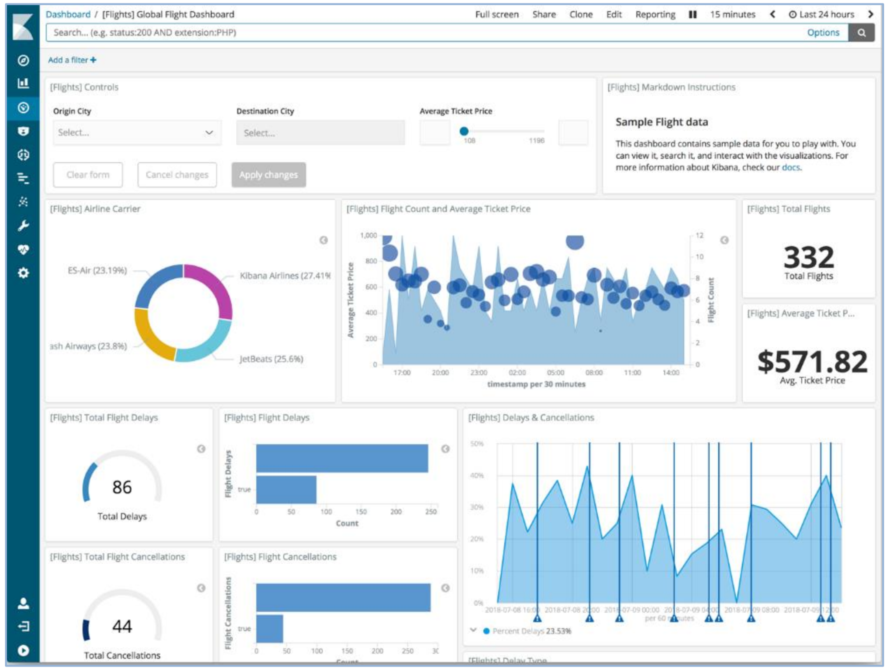
Deploy and operationalize model on Azure cloud

Azure

Operator

Makes operational decisions based on model output

kibana




5"
12,000 PSI

Baker Hughes Develops Predictive Maintenance Software for Gas and Oil Extraction Equipment Using Data Analytics and Machine Learning

Truck with positive displacement pump.

Challenge

Develop a predictive maintenance system to reduce pump equipment costs and downtime

Solution

Use MATLAB to analyze nearly one terabyte of data and create a neural network that can predict machine failures before they occur

Results

- Savings of more than $10 million projected
- Development time reduced tenfold
- Multiple types of data easily accessed

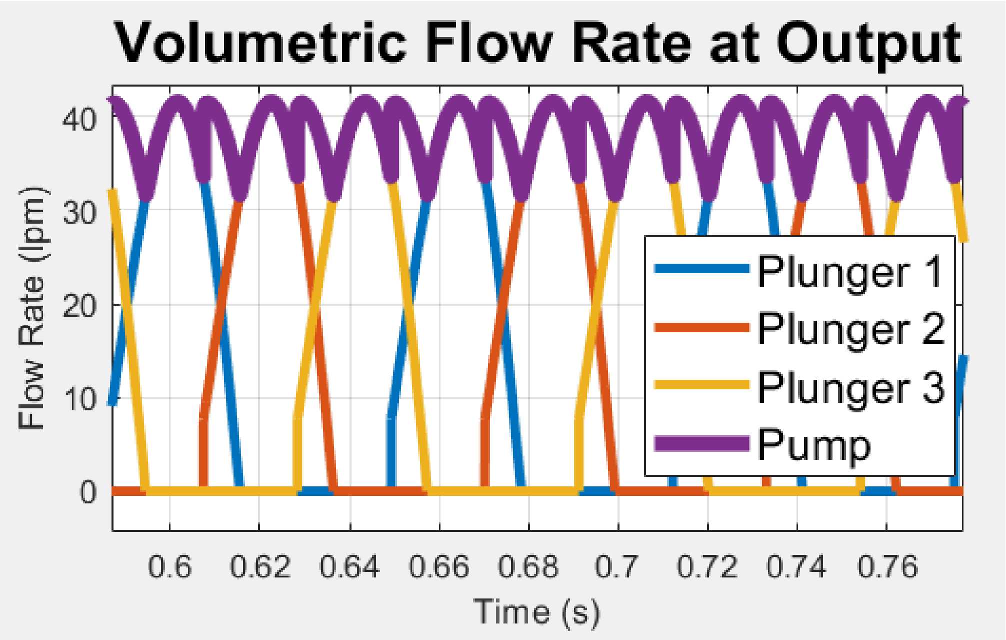
"MATLAB gave us the ability to convert previously unreadable data into a usable format; automate filtering, spectral analysis, and transform steps for multiple trucks and regions; and ultimately, apply machine learning techniques in real time to predict the ideal time to perform maintenance."

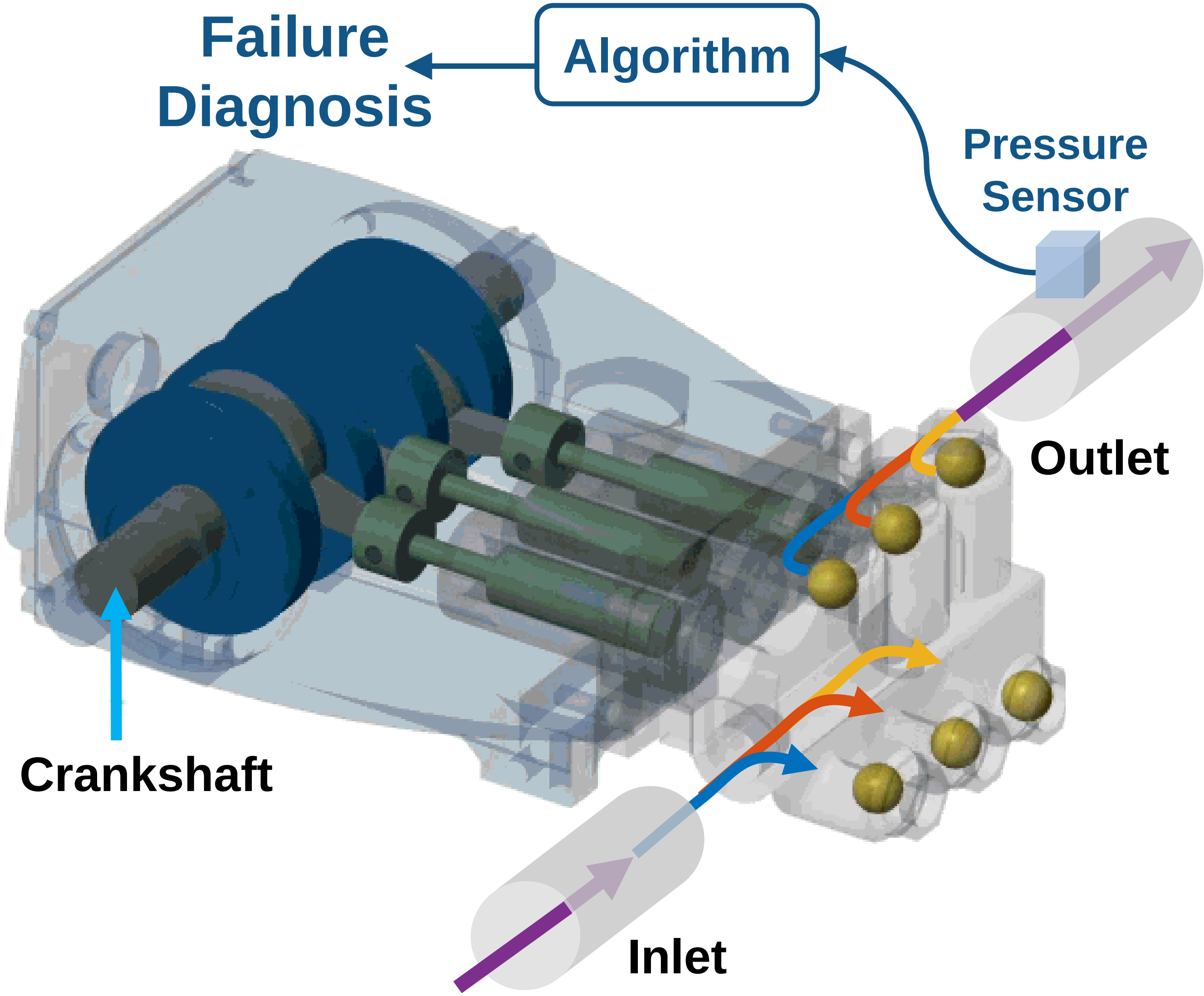
- Gulshan Singh, Baker Hughes

[Link to user story]

Triplex Pump

- Crankshaft drives three plungers
  - Each 120 degrees out of phase
  - One chamber always discharging
  - Smoother flow than single or duplex piston pumps





Project statement: Develop a full end-to-end predictive maintenance system and demo with the plant operator in one 3-4 week sprint

Plant Operator

1. I would like to monitor the ***flow***, ***pressure***, and ***current*** of each pump in my plant so that I always know their ***operational state***
2. I need to be ***alerted*** when any pump's operational parameters drift outside of an acceptable range so that I can take ***immediate corrective action***
3. I would like to get a continuous estimate of each pump's ***remaining useful life (RUL)*** so that I can ***schedule maintenance or replacement*** of the asset

Project constraints and solutions

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

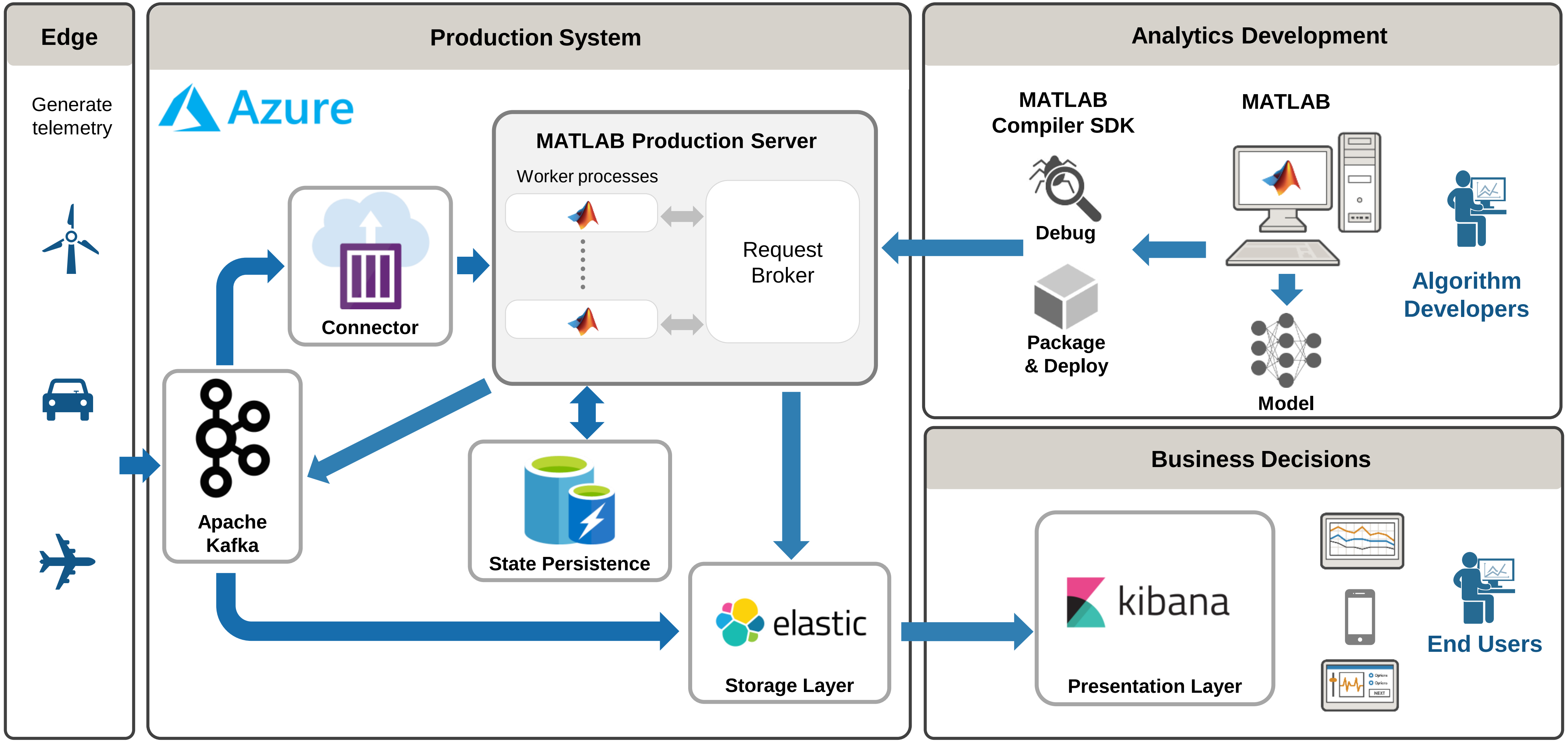
Solution: Use an accurate physics-based software model for the pump to develop synthetic training sets

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 a cloud platform that enables us to quickly configure and provision the services we need to build the solution, while minimizing lock-in to a particular provider

Predictive Maintenance Architecture on Azure



Agenda

1 Access and Explore Data
- Files
- Databases
- Sensors

2 Preprocess Data
- Working with Messy Data
- Data Reduction/ Transformation
- Feature Extraction

3 Develop Predictive Models
- Model Creation e.g. Machine Learning
- Parameter Optimization
- Model Validation

4 Integrate with Production Systems
- Desktop Apps
- Enterprise Scale Systems (kafka, AWS Kinesis, OSIsoft.)
- Embedded Devices and Hardware

5 Visualize Results
- 3rd party dashboards (Spotfire TIBCO Software, tableau, Qlik, Power BI)
- Web apps

Review model requirements

- Requirements from the Operator
  - Continuous predictions of:
    - Fault values
      - Leak cylinder area
      - Block in factor
      - Bearing friction
    - Type of fault (“Blocking”, “Leaking”, “All”)
    - Remaining Useful Life (RUL)

Operator

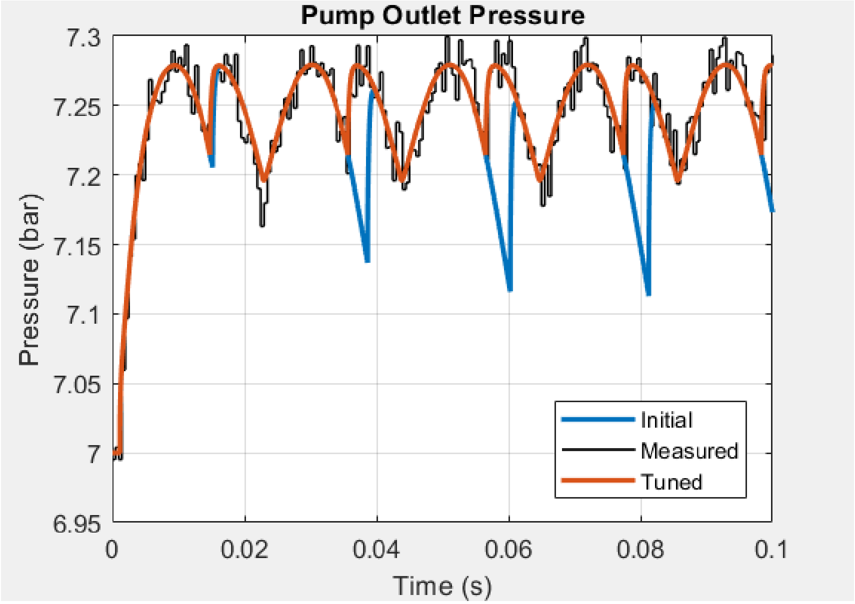
- Requirements from System Architect
  - Define window for streaming
  - Define format of results, intermediate values
  - Scalable code
  - Test code

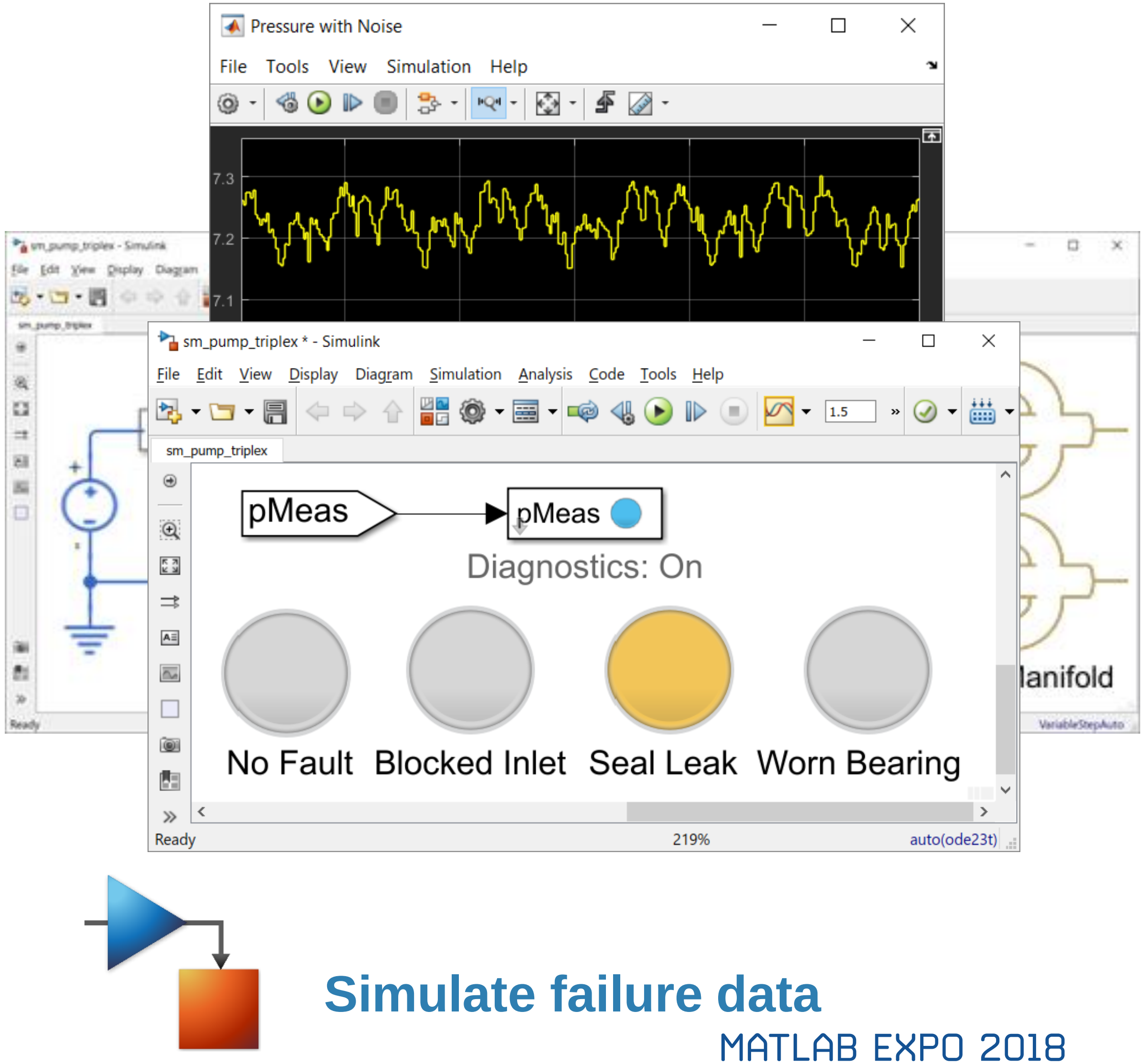
System Architect

1

Access and Explore Data

Use sensor data from pump to identify levels of failure



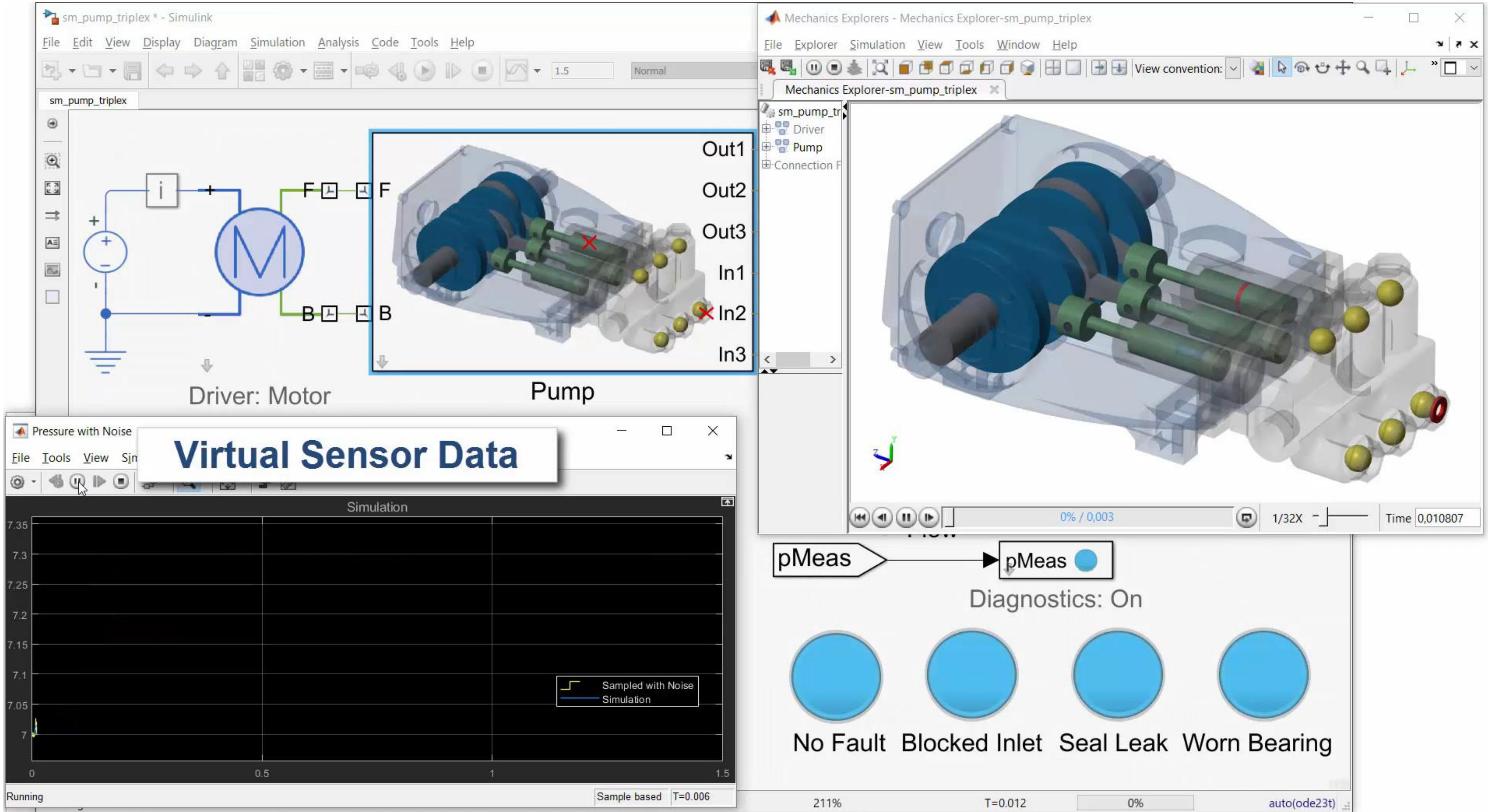


Pump sensor data

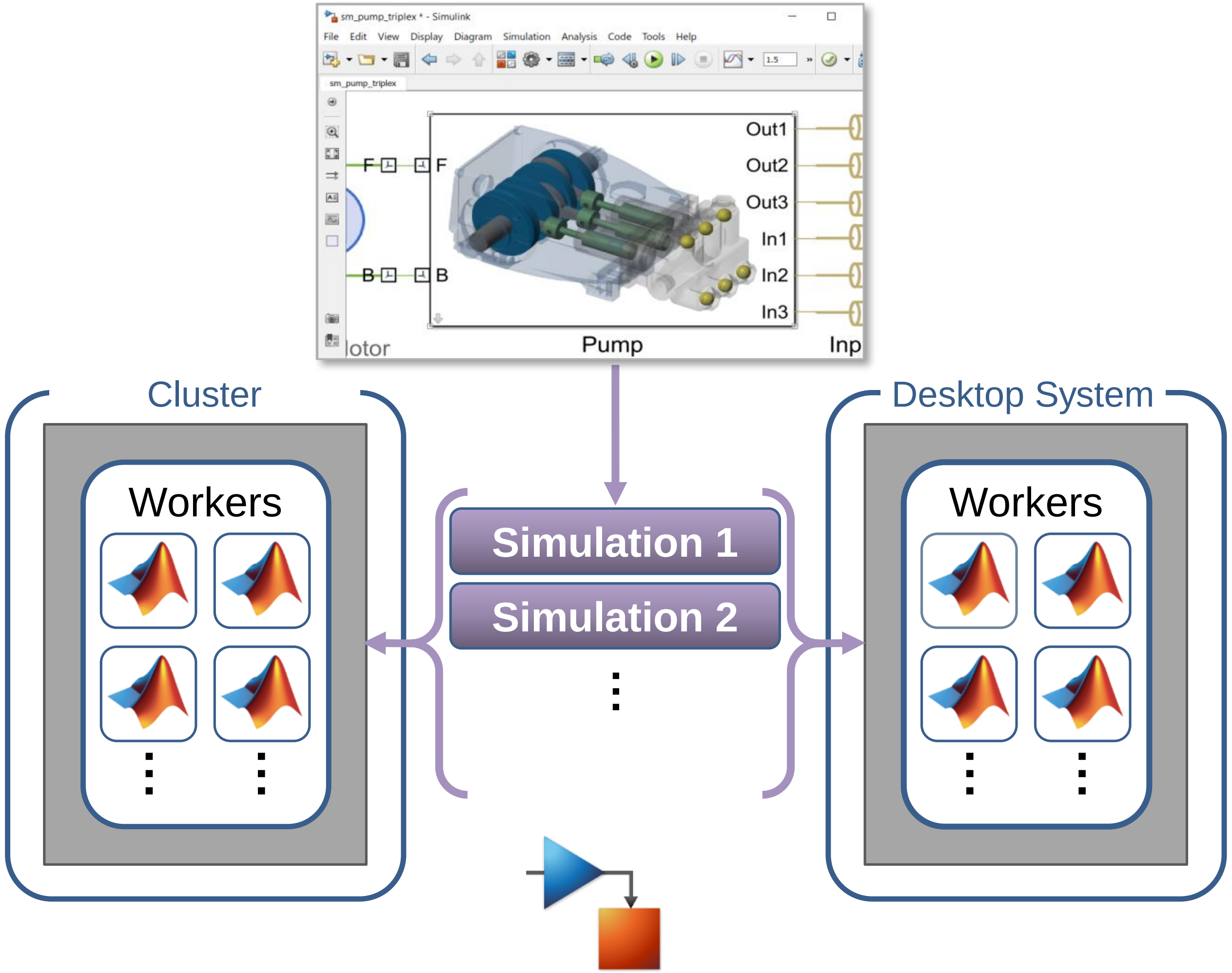
Simulate failure data

1 Access and Explore Data

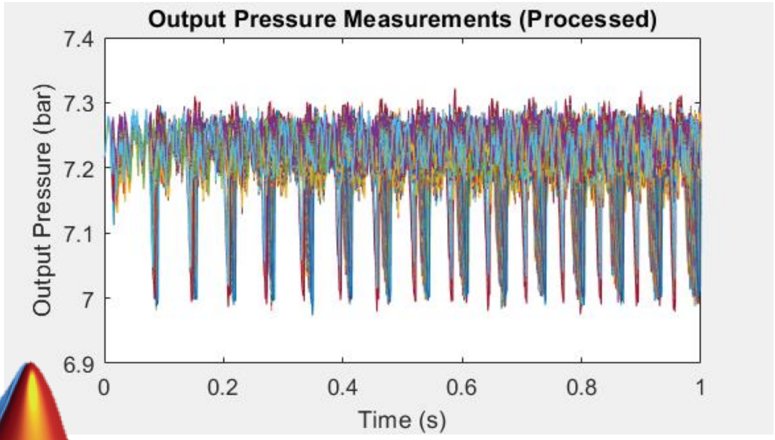
Build digital twin and generate sensor data



Simulate data with many failure conditions



Run parallel simulations

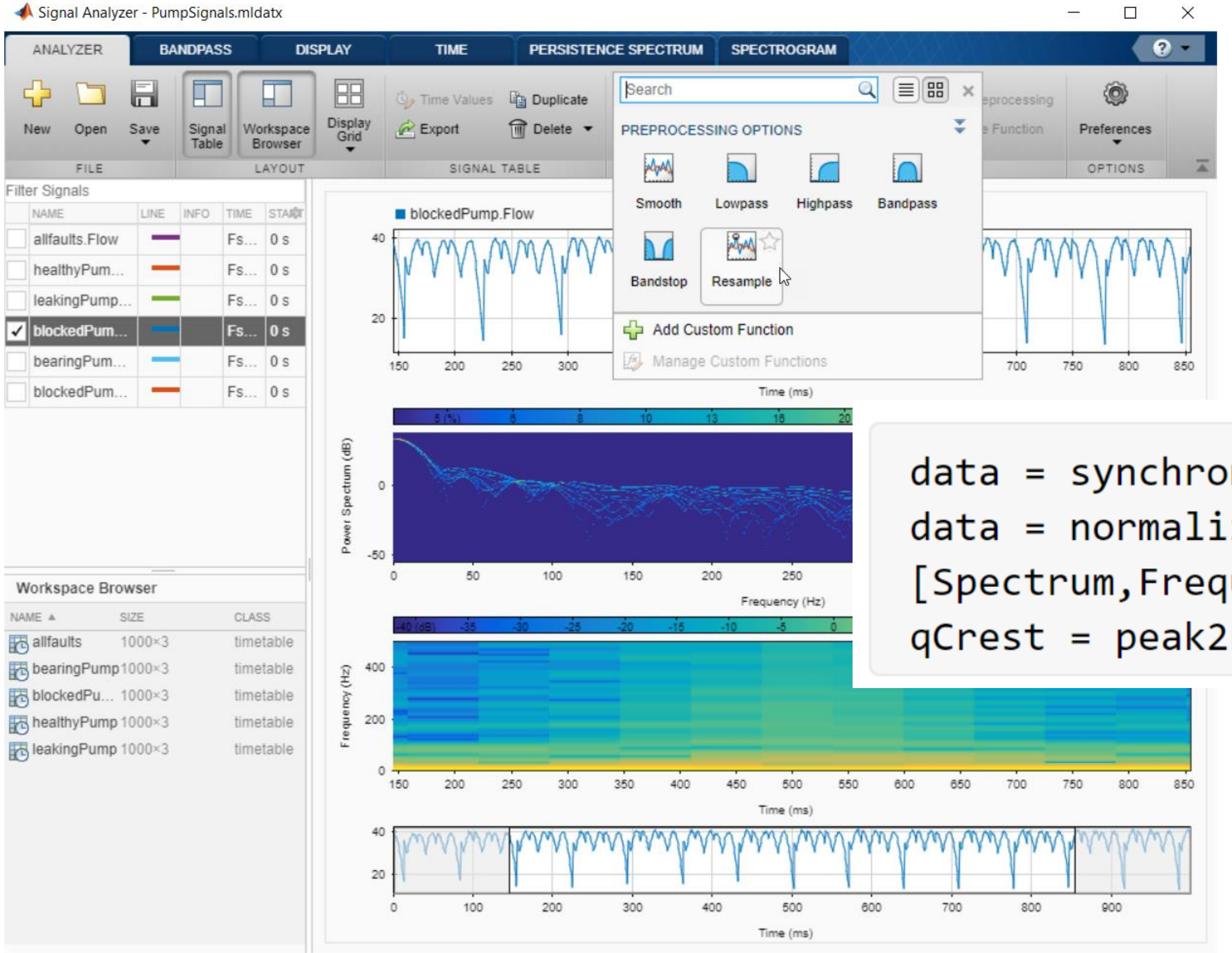


Store data on HDFS and leverage MATLAB + Spark for machine learning

MathWorks

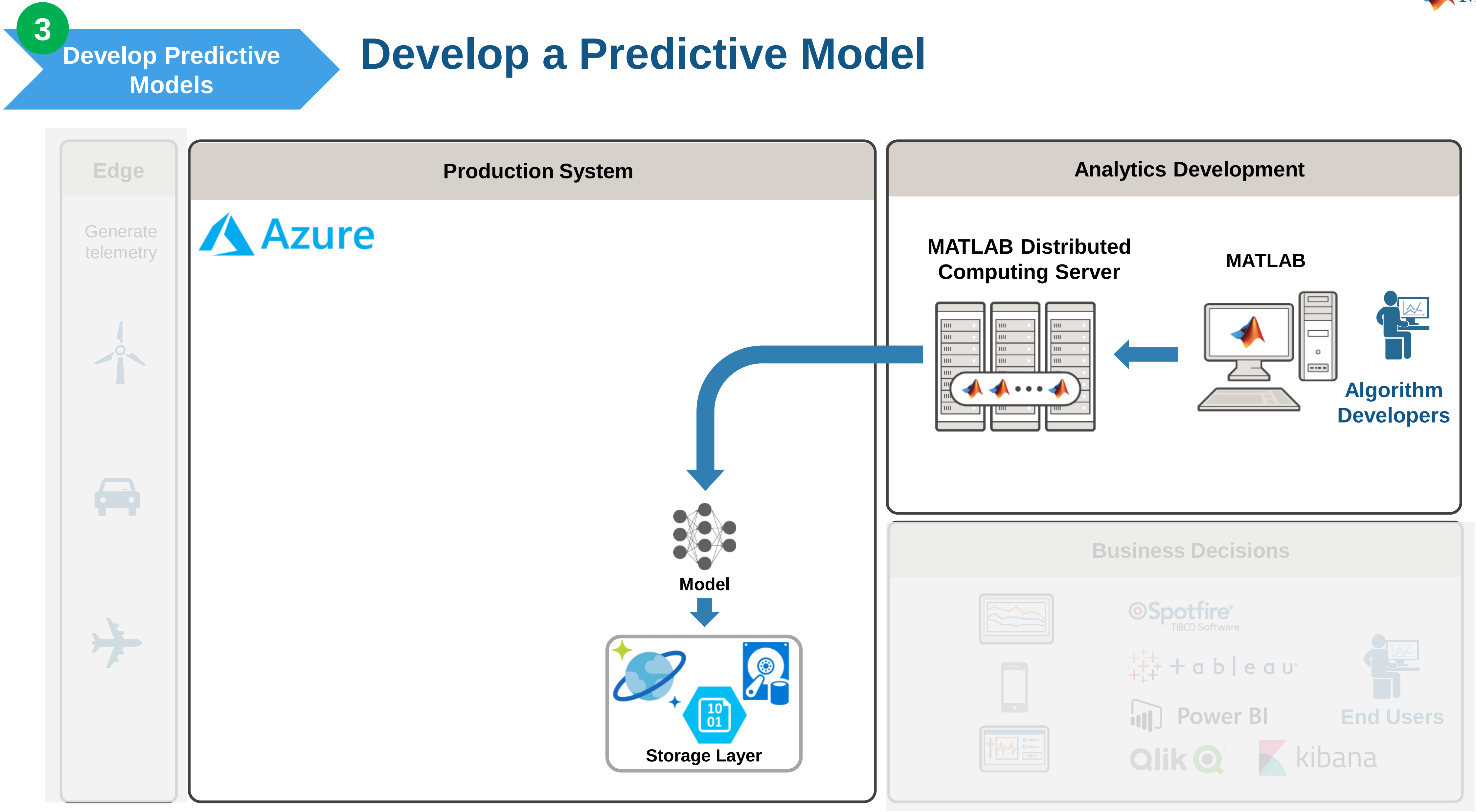
2 Preprocess Data

Preprocess data and represent signal information




```
data = synchronize(Flow,Pressure,t,'linear');
data = normalize(data,'center');
[Spectrum,Frequencies] = pspectrum(data.Flow);
qCrest = peak2rms(data.Flow);
```

Develop a Predictive Model



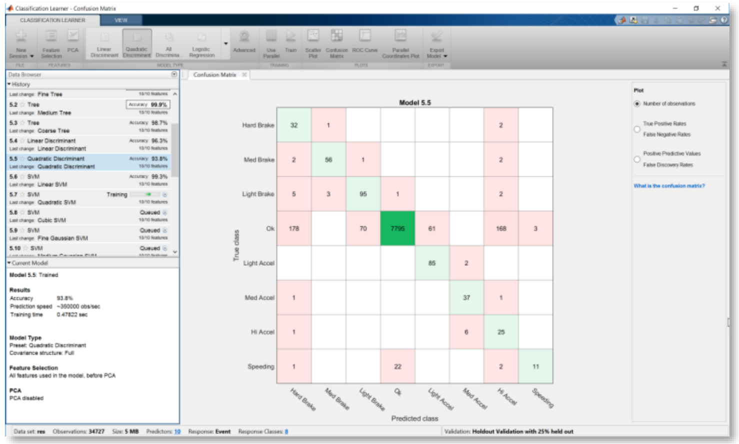
3 Develop Predictive Models

Develop a Predictive Model in MATLAB

time	1 Event	2 SpeedGPS	3 AccelerationSensorXAxis	4 AccelerationSensorYAxis	5 AccelerationSensorZAxi
Mon May 11 04:03:15 UTC 2015	Hard Brake	10.8360	-0.6996	0.6014	0.205
Wed May 06 19:09:48 UTC 2015	Hard Brake	27.8280	0.1419	0.9035	-0.526
Sun May 17 17:09:19 UTC 2015	Hard Brake	6.5520	0.9986	-0.0761	-0.004
Fri Jan 16 20:38:37 UTC 2015	Hard Brake	39.6128	0.0999	0.8000	0.367
Sat May 02 14:00:37 UTC 2015	Hard Brake	61.1280	0.4006	-0.4022	0.663
Mon Apr 27 17:54:27 UTC 2015	Fast Accel	37.7640	0.1527	0.4666	0.857
Sun May 03 21:00:42 UTC 2015	Fast Accel	17.2440	1.0235	0.0815	0.304
Mon May 04 11:30:33 UTC 2015	Fast Accel	19.6560	0.1336	0.8932	-0.578

Label Faults

Represent Signals



Train Model

Validate Model

```
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
```

Scale up

```
tt = tall(data);  % test tall array
model = TreeBagger(50,tt,'Event');
```

Scale to out of memory data

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

Scale Up

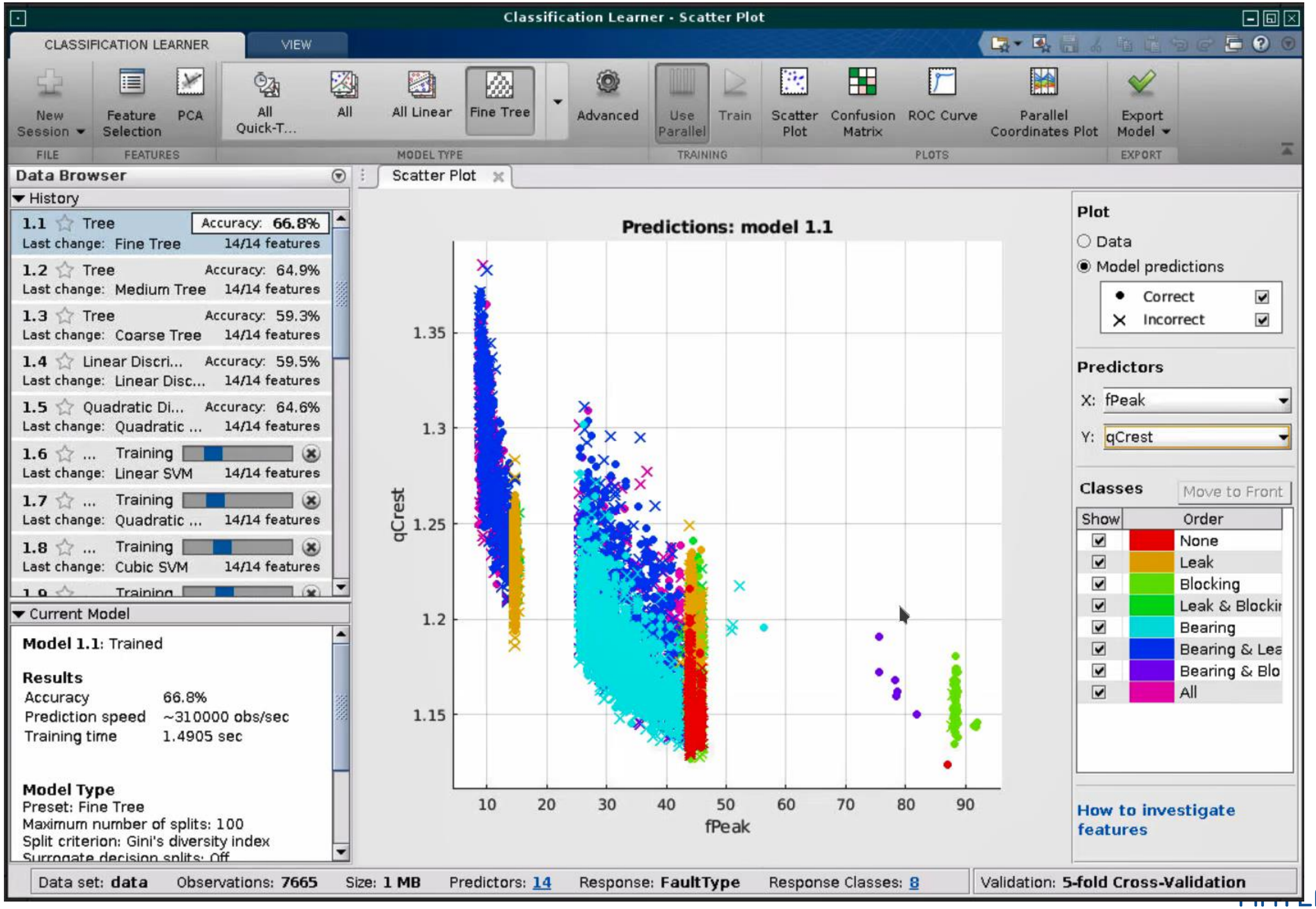
3 Develop Predictive Models

Develop a Predictive Model in MATLAB

- Predict fault values (Regression)
- Predict type of fault (Classification)
- Predict remaining useful life (Exponential degradation)
- Use 1 second of data for predictions

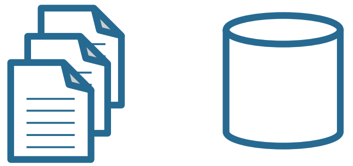
3 Develop Predictive Models

Develop a Predictive Model in MATLAB

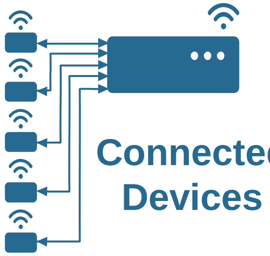


Develop a Stream Processing Function

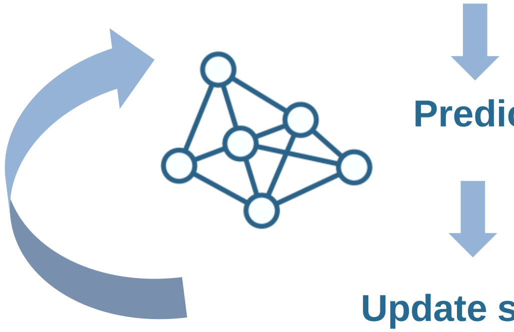
- Model is built and tested on historic data



- Model will be used on streaming data



- Retain model state and update



4 Integrate with Production Systems

Develop a Stream Processing Function in MATLAB

Streaming Function

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

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

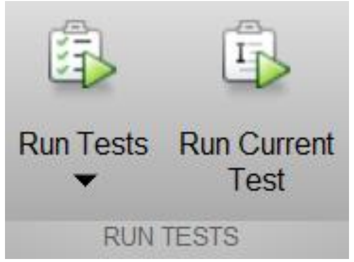
```
writeResults(Leak,Blocking,Bearing,FaultType,RUL,Model)
end
```

Process each window of data as it arrives

Previous state

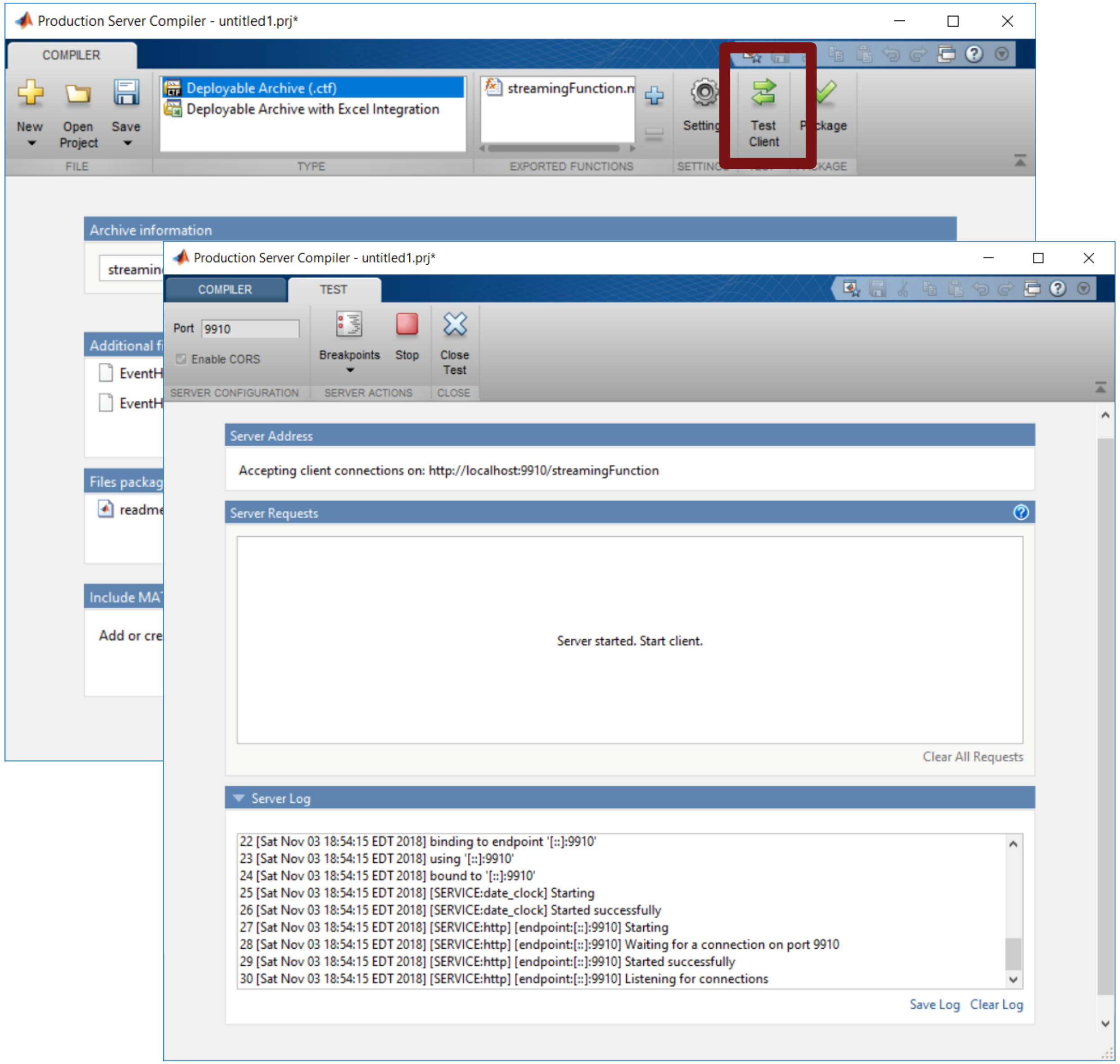
Current window of data to be processed

Test Stream Processing Function



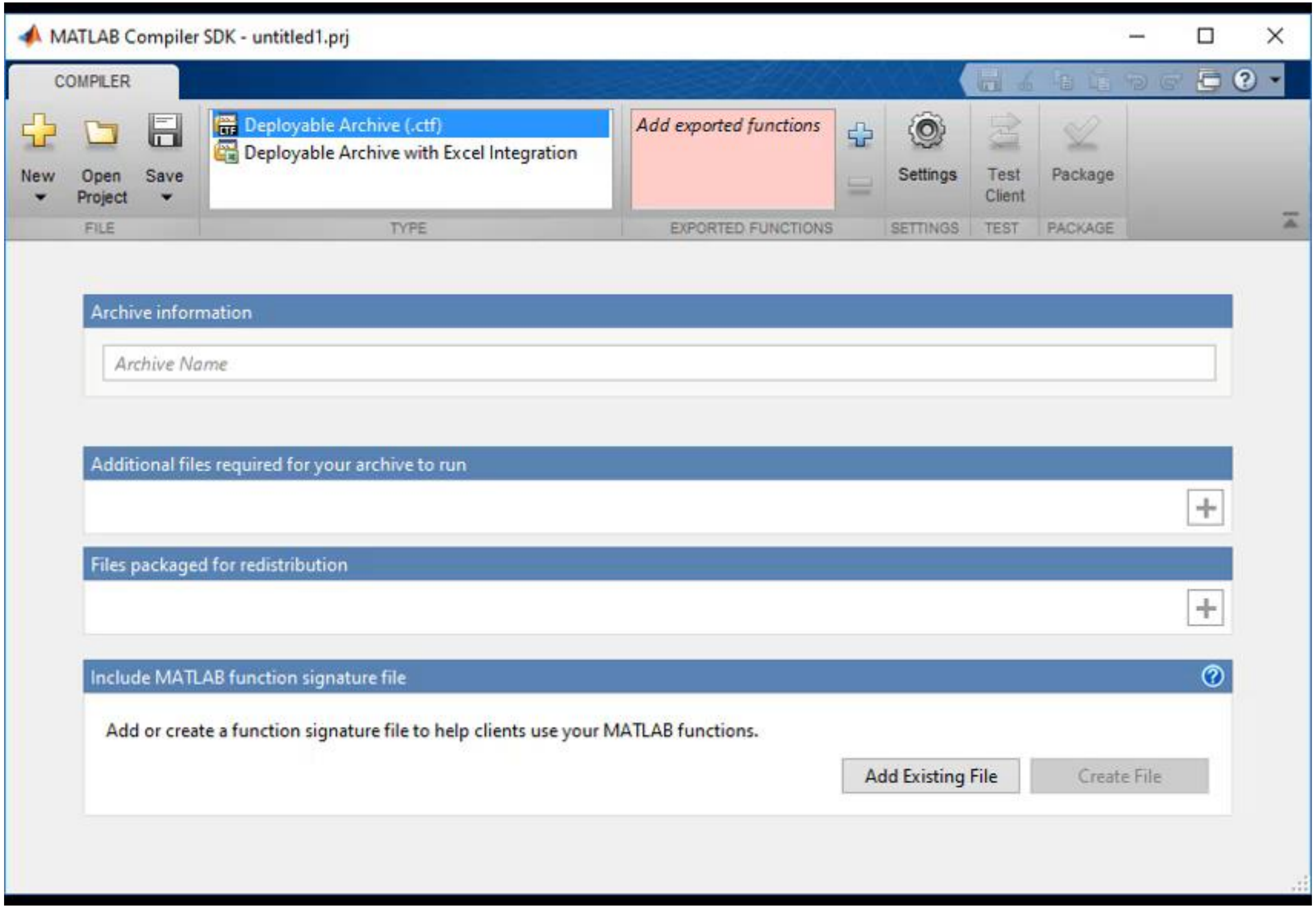

```
results = runtests('predictFaults_tests')
```

```
Running predictFaults_tests
....
Done predictFaults_tests
__________

results =
  1×4 TestResult array with properties:

    Name
    Passed
    Failed
    Incomplete
    Duration
    Details
Totals:
   4 Passed, 0 Failed, 0 Incomplete.
   0.01614 seconds testing time.
```



4 Integrate with Production Systems

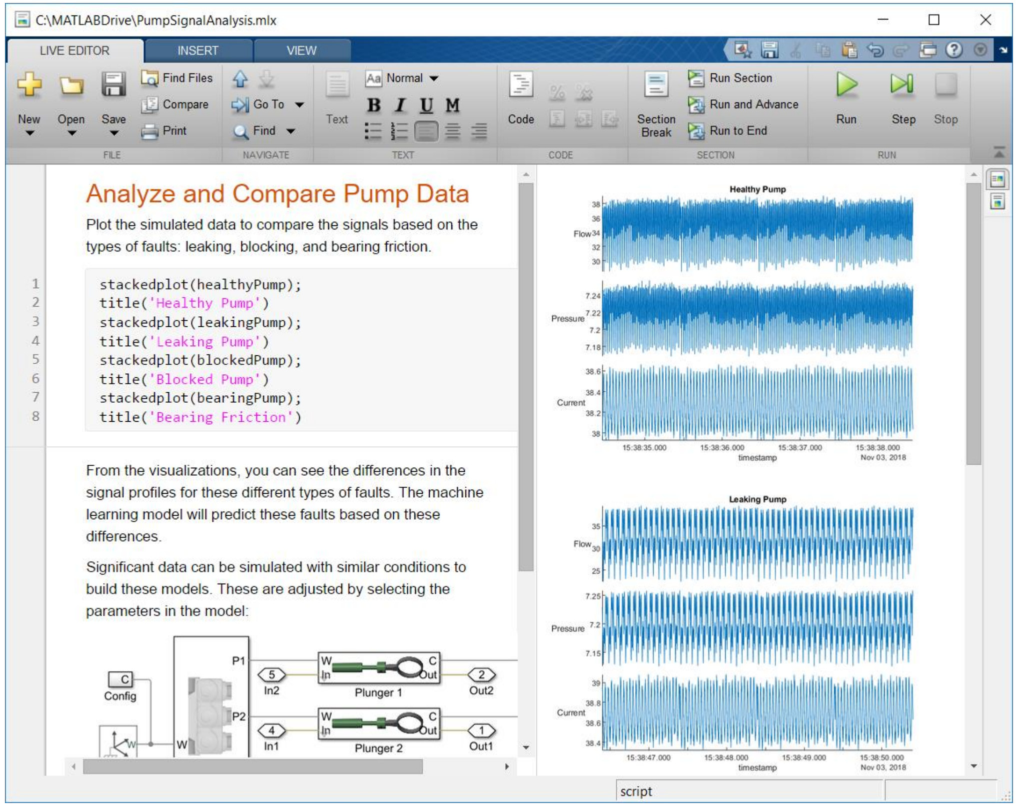
Package Stream Processing Function



4 Integrate with Production Systems

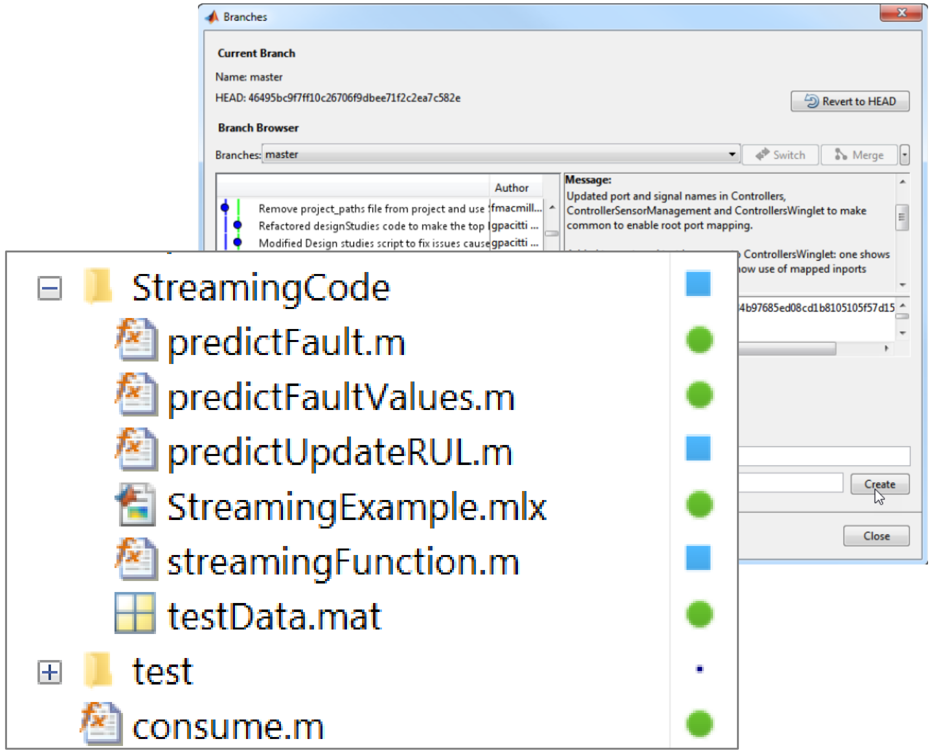
Share with the team

Review results with Operator



pdf, html, LaTeX

Share code with System Architect



Source Control

Review system requirements

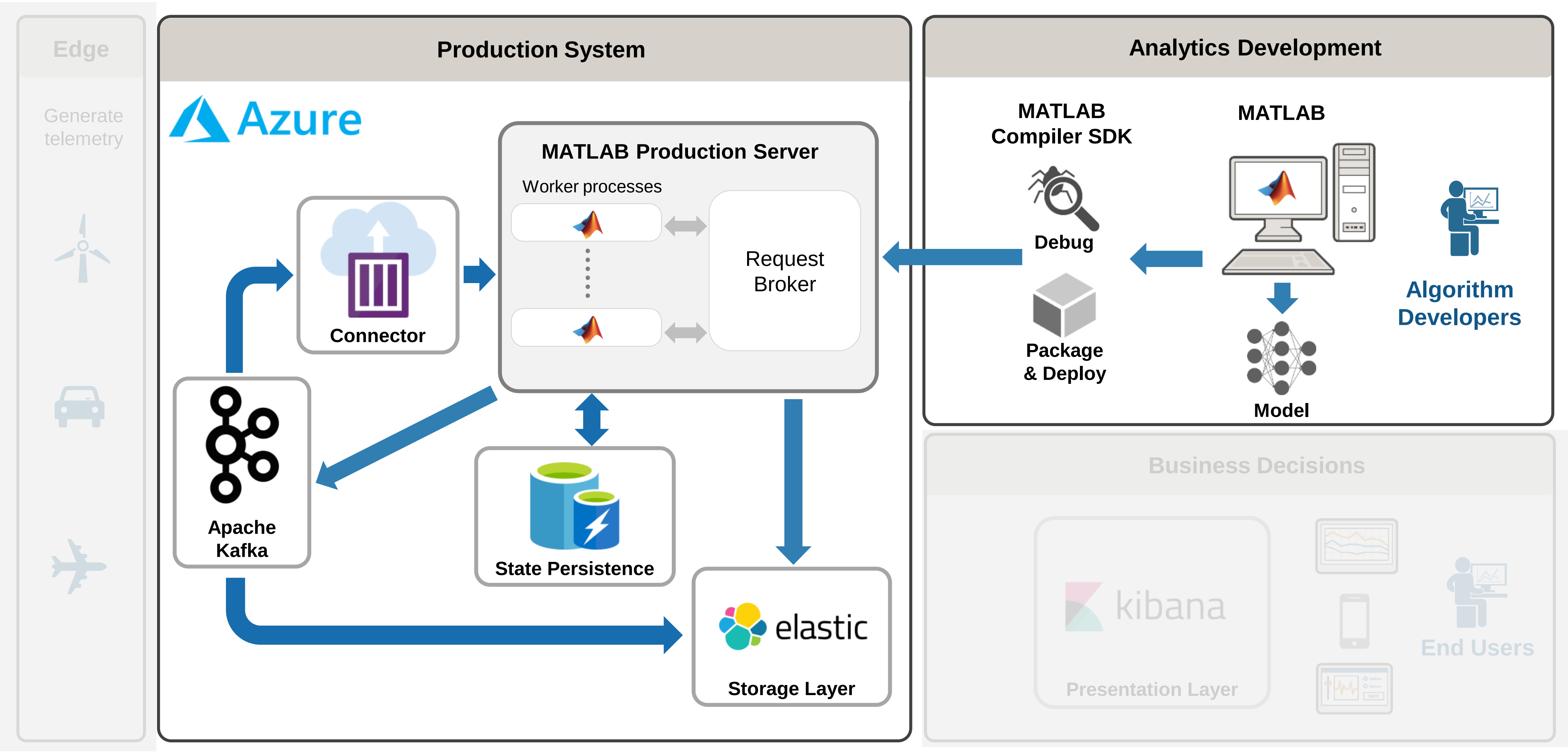
- Requirements from the Process Engineer
  – Every millisecond, each pump generates a time-stamped record of flow, pressure, and current
  – Model expects 1 sec. window of data per pump
  – Initially, 1’s – 10’s of devices, but quickly scale to 100’s

Process Engineer

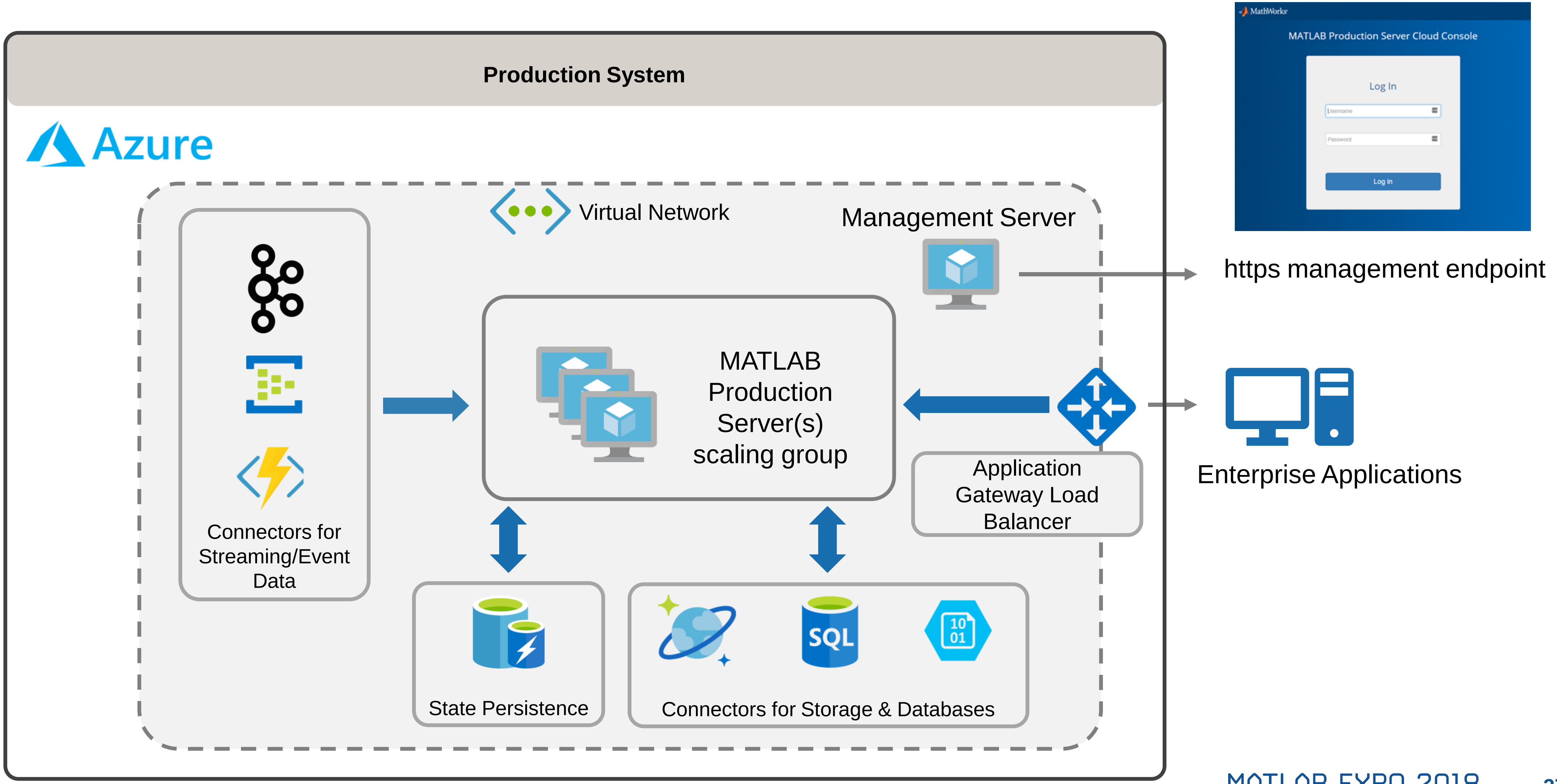
- Requirements from the operator
  – Visual description of the device output for each pump
  – Alerts when parameters drift outside the expected ranges
  – Continuous estimating of RUL for each pump

Operator

Integrate Analytics with Production Systems



MATLAB Production Server on Azure

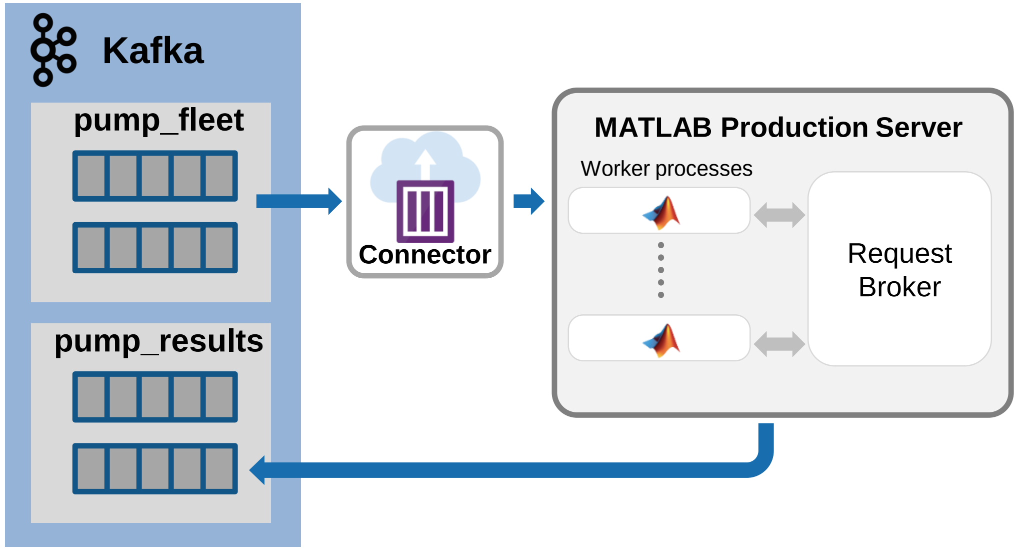


Connecting MATLAB Production Server to Kafka

Goals:

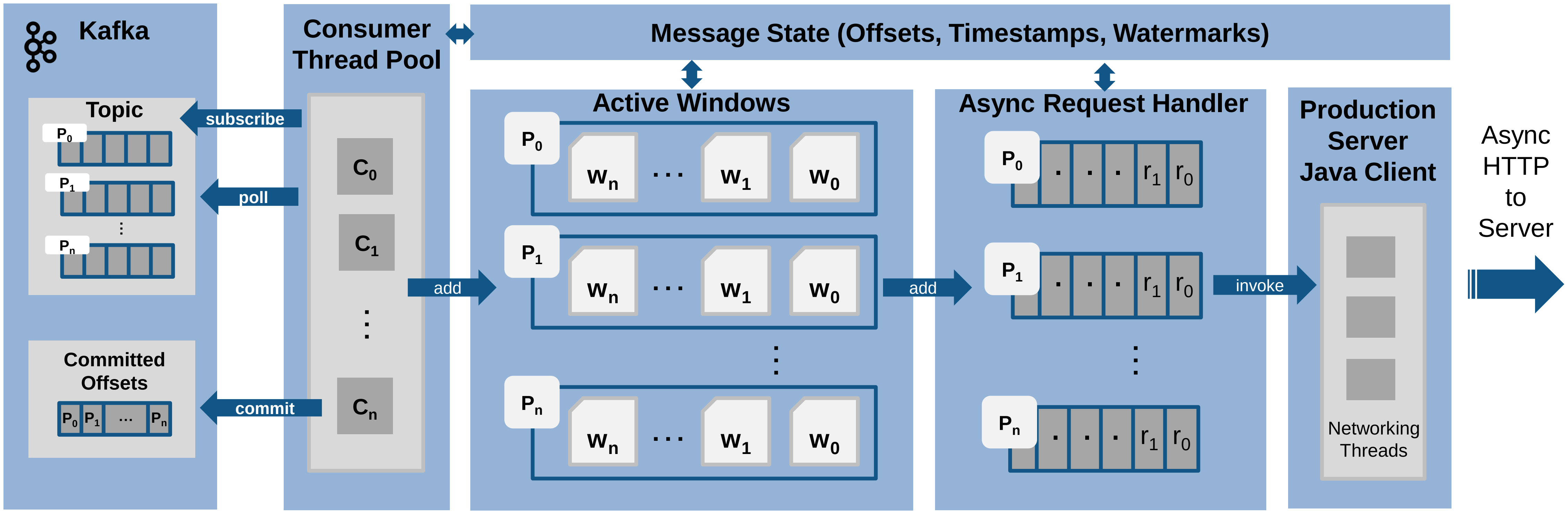
- Develop a micro-service that feeds a single Kafka topic to a MATLAB function
- Develop a publisher library for MATLAB for writing output to a results stream

- Basic Features:
  - Group data into time windows and pass to MATLAB as a timetable
  - Time-stamps are in "event time" and may arrive out of order
  - Execute streaming function asynchronously and exploit parallel execution of Kafka partitions
  - Use Kafka's check-pointing (i.e. at-least-once)



- Deployment model: Docker container suitable for Azure Container Instance

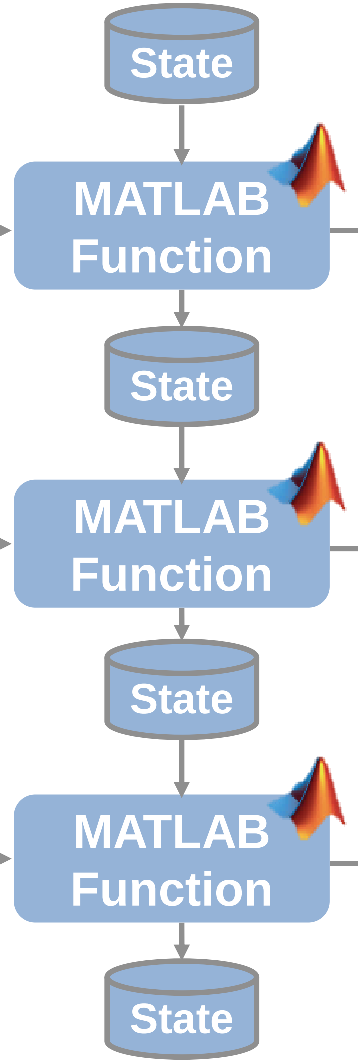
Kafka connector architecture



Streaming data is treated as an unbounded Timetable

Input Table

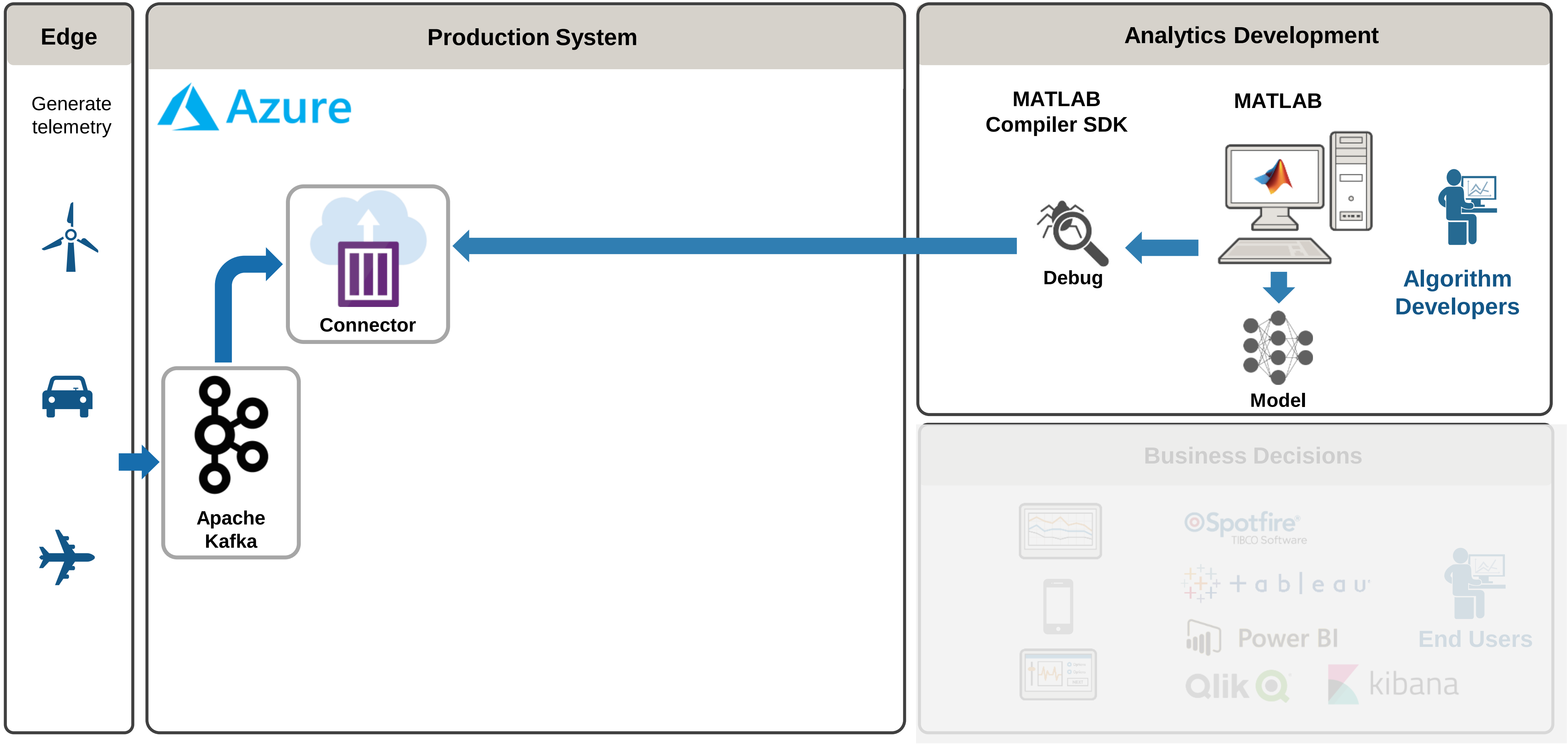
Event Time	Pump Id	Flow	Pressure	Current
18:01:10	Pump1	1975	100	110
18:10:30	Pump3	2000	109	115
18:05:20	Pump1	1980	105	105
18:10:45	Pump2	2100	110	100
18:30:10	Pump4	2000	100	110
18:35:20	Pump4	1960	103	105
18:20:40	Pump3	1970	112	104
18:39:30	Pump4	2100	105	110
18:30:00	Pump3	1980	110	113
18:30:50	Pump3	2000	100	110
…	…	…	…	…



Output Table

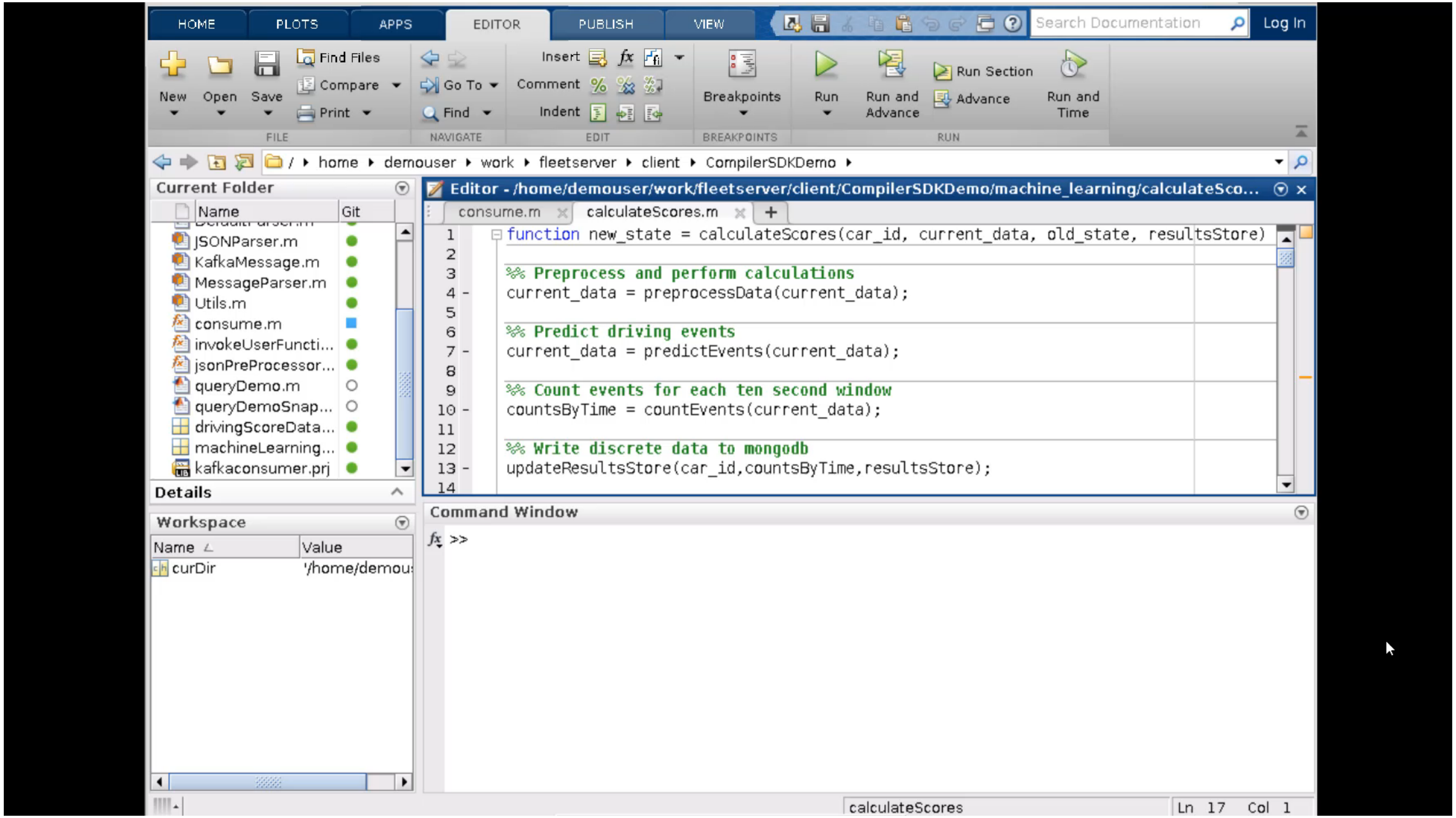
Time window		Pump Id	Bearing Friction
…		…	…
18:00:00	18:10:00	Pump1	5
		Pump3	…
		Pump4	…
18:10:00	18:20:00	Pump2	7
		Pump3	3
		Pump4	…
18:20:00	18:30:00	Pump1	…
		Pump3	4
		Pump4	…
18:30:00	18:40:00	Pump5	…
		Pump3	5
		Pump4	8

Debug a Stream Processing Function in MATLAB



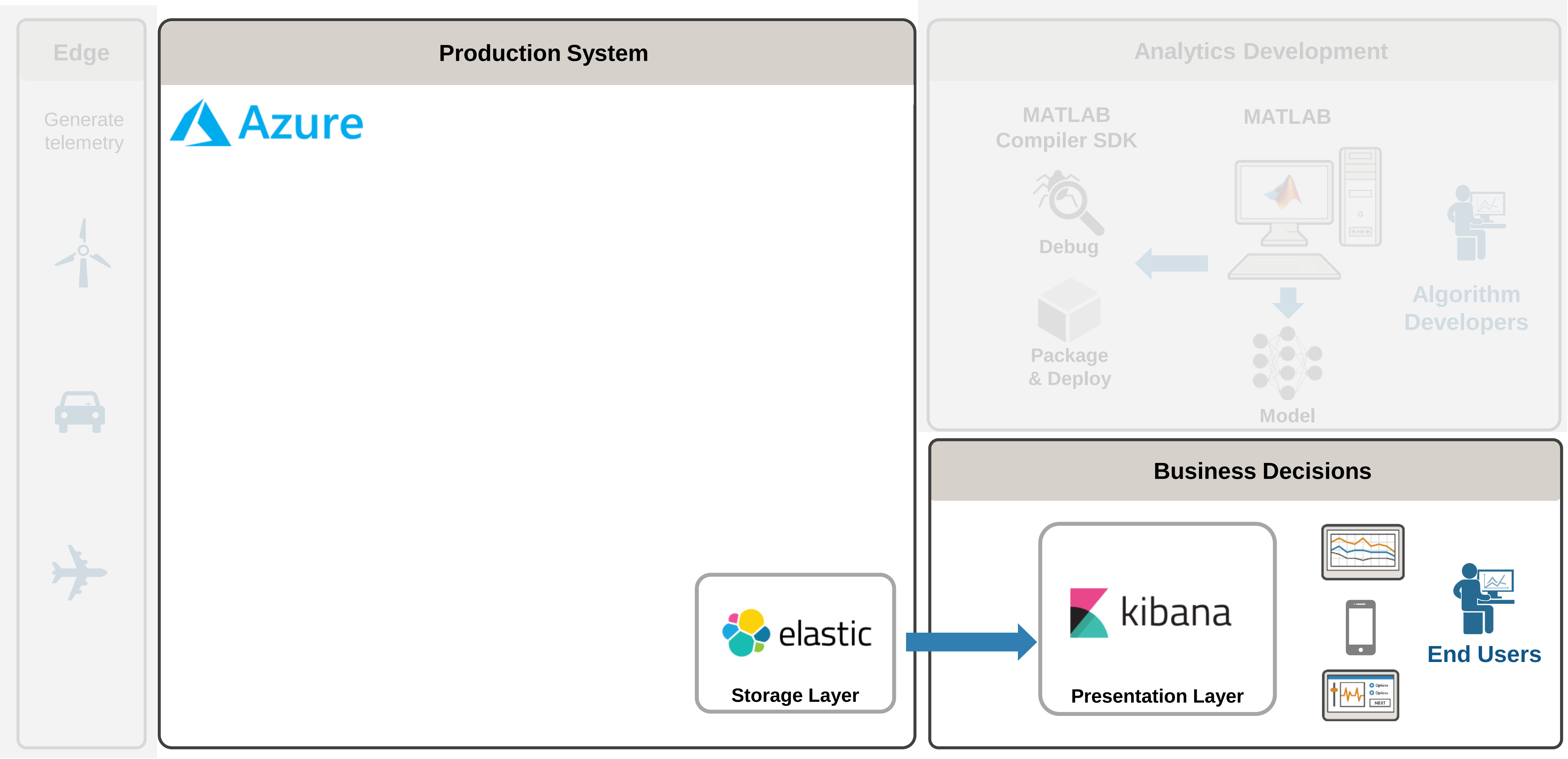
4 Integrate with Production Systems

Debug a Stream Processing Function in MATLAB



5 Visualize Results

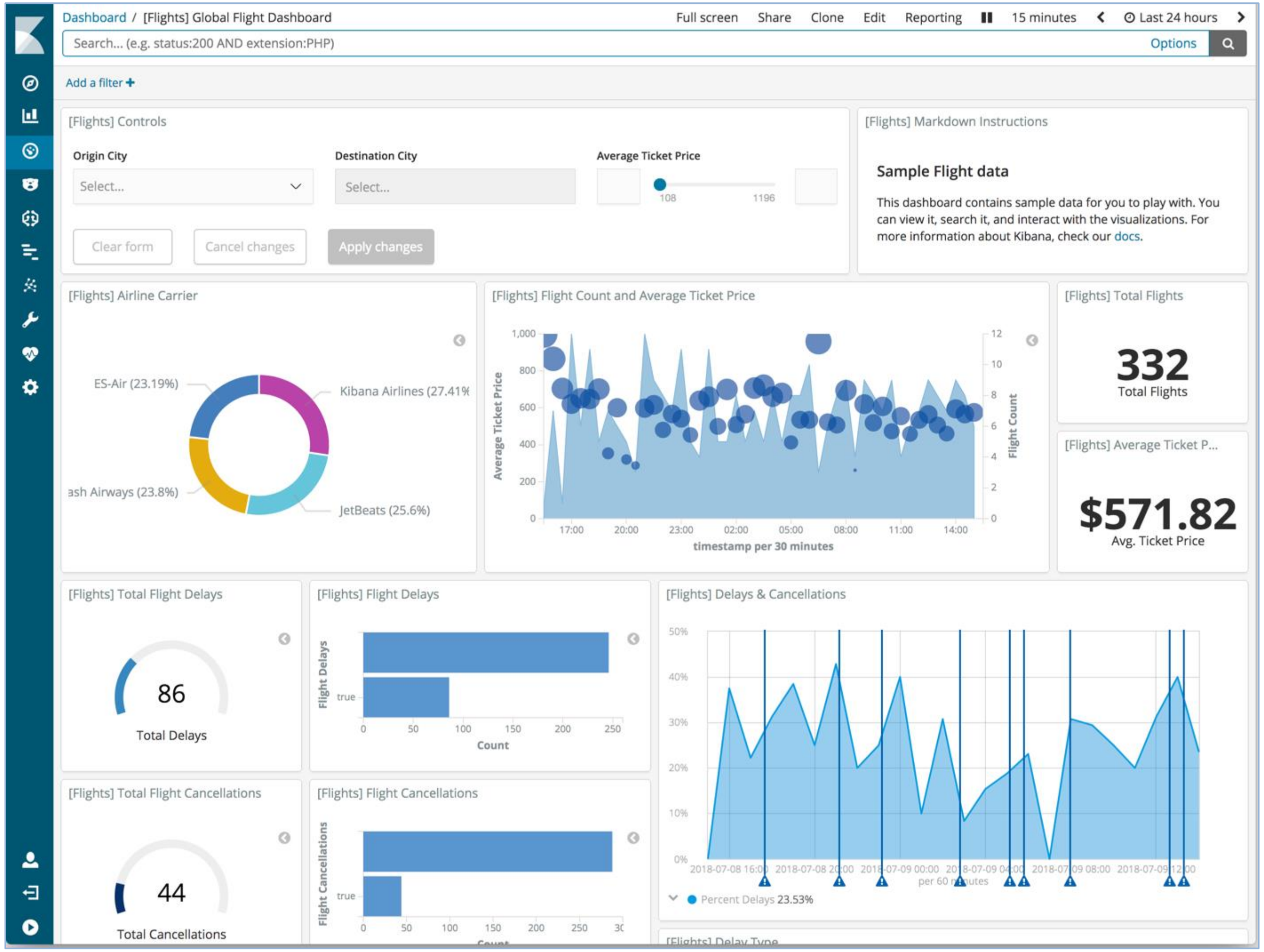
Complete Your Application



5

Visualize Results

Complete Your Application



Key Takeaways

- Rapidly import, generate, and preprocess your data with MATLAB and Simulink
- Spend less time preprocessing your data with pre-built MATLAB functions
- Focus on selecting and tuning your AI models with MATLAB apps instead of dealing with the mechanics of training
- Deploy your predictive models to the cloud, and integrate with third-party storage, applications, and visualization dashboards with MATLAB Production Server

Resources to learn and get started

- [Data Analytics with MATLAB]
- [Working with Enterprise IT Systems]
- [MATLAB Production Server]
- [MATLAB Compiler SDK]
- [Statistics and Machine Learning Toolbox]
- [Predictive Maintenance Toolbox]
- [Simulink (related to pump model)]