The Need for Large-Scale Streaming

**Predictive Maintenance**
*Increase Operational Efficiency*
*Reduce Unplanned Downtime*

**Medical Devices**
*Patient Safety*
*Better Treatment Outcomes*

**Connected Cars**
*Safety, Maintenance*
*Advanced Driving Features*

**Finance**
*High Frequency Trading*
*Sentiment Analysis*
Example Problem: develop and operationalize a machine learning model to predict failures in industrial pumps

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

- **Process Engineer**
  - Develops models in MATLAB and Simulink

- **System Architect**
  - Deploys and operationalizes model on Azure cloud
  - Azure

- **Operator**
  - Makes operational decisions based on model output
  - Kibana
Project statement: develop end-to-end predictive maintenance system and demo in one 3-4 week sprint

Monitor **flow**, **pressure**, and **current** of each pump so I always know their operational state

Need **alert** when fault parameters drift outside an acceptable range so I can take **immediate action**

Continuous estimate of pump’s **remaining useful life (RUL)** & **classification** ➔ schedule maintenance or replace the asset
Project statement: constraints & solution

Process Engineer: I have few or zero failure data

Generate realistic synthetic data / use Machine Learning models

Architect IT: I have a limited budget, and don’t know the adjusted platform

Leverage cloud platform to quickly configure it

Process Engineer: We need multiple tools for multidisciplinary problems

Use MATLAB and integrate with other environments
Predictive Maintenance Architecture on Azure
Review model requirements

Operator
- Type of fault
- RUL

System Architect
- Time-windowing
- Out-of-order delivery
- Test code
- Scalable code

Process Engineer
A complete end-to-end workflow

Access Data

Preprocess

Identify Features

Predictive Analytics

Deploy & Integrate

Files

Multiple formats

Data Transformation

Model Creation

Enterprise Scale System

Database & Cloud

Messy Data

Feature Extraction

Machine Learning

Visualization

Sensors

Arrange data

Feature Selection

Parameter Optimization

Model Validation

Web Apps

Files

Multiple formats

Data Transformation

Model Creation

Enterprise Scale System

Database & Cloud

Messy Data

Feature Extraction

Parameter Optimization

Model Validation

Sensors

Arrange data

Feature Selection

Web Apps
Access/Generate data

Crankshaft drives three plungers ➔ Three types of failures

- Outlet Algorithm
- Pressure Sensor
- Failure Diagnosis
- Crankshaft Bearing Friction
- Inlet Blocking Fault
- Leak Area

Process Engineer
Access/Generate data

Digital Twin

Bearing Friction
Blocking Fault
Leak Area

Data & Failures
Simulation

Parallel Computing

Run many parallel simulations

simulationEnsembleDatastore

Process Engineer
Preprocessing data

data = synchronize(Flow, Pressure, Current, t, 'linear');
data = normalize(data, 'center');
Identify Condition Indicators

Feature Diagnostic Designer

Visualize data
Extract features
Select the most useful features

Process Engineer
Predictive Analytics: regression

= Remaining Useful Life

Process Engineer
Predictive Analytics: classification

Type of fault

Classification Learner App

Process Engineer
Integrate with Production Systems

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

- **Continuous Data**: Pump Sensor Data
- **Messaging Service**
- **Streaming Function**: $f(x)$
- **Update State**
- **Make Decisions**

*Process Engineer*

*System Architect*
Develop a 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
Package Stream Processing Function easily
Review System Requirements

Operator

- Alerts
- Type of fault

Engineer

- High frequency
- Big Data
- Scalability

System Architect
Integrate Analytics with Production Systems

**Edge**

**Production System**

- **Azure**
- **MATLAB Production Server**
  - Worker processes
  - Request Broker
- **kafka**
- **State Persistence**
  - **Storage Layer**
  - **elastic**

**Analytics Development**

- **Compiler SDK**
- **MATLAB**
- **Package & Deploy**
- **Model**
- **kibana**
  - **Presentation Layer**

**Business Decisions**
Configure MATLAB Production Server in the cloud

Production System

Azure

Virtual Network

Management Server

MATLAB Production Server(s) scaling group

Application Gateway Load Balancer

State Persistence

Connectors for Storage & Databases

Connectors for Streaming/Event Data

https://github.com/mathworks-ref-arch
Zoom on Kafka connector to MPS

Production System

MATLAB Production Server
- Worker processes
- Request Broker

Connector
- Kafka
- Pump fleet
- Pump results
- Timetable

System Architect
Streaming data is treated as an unbounded Timetable

<table>
<thead>
<tr>
<th>Event Time</th>
<th>Pump Id</th>
<th>Flow</th>
<th>Pressure</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>18:01:10</td>
<td>Pump1</td>
<td>1975</td>
<td>100</td>
<td>110</td>
</tr>
<tr>
<td>18:10:30</td>
<td>Pump3</td>
<td>2000</td>
<td>109</td>
<td>115</td>
</tr>
<tr>
<td>18:05:20</td>
<td>Pump1</td>
<td>1980</td>
<td>105</td>
<td>105</td>
</tr>
<tr>
<td>18:10:45</td>
<td>Pump2</td>
<td>2100</td>
<td>110</td>
<td>100</td>
</tr>
<tr>
<td>18:30:10</td>
<td>Pump4</td>
<td>2000</td>
<td>100</td>
<td>110</td>
</tr>
<tr>
<td>18:35:20</td>
<td>Pump4</td>
<td>1960</td>
<td>103</td>
<td>105</td>
</tr>
<tr>
<td>18:20:40</td>
<td>Pump3</td>
<td>1970</td>
<td>112</td>
<td>104</td>
</tr>
<tr>
<td>18:39:30</td>
<td>Pump4</td>
<td>2100</td>
<td>105</td>
<td>110</td>
</tr>
<tr>
<td>18:30:00</td>
<td>Pump3</td>
<td>1980</td>
<td>110</td>
<td>113</td>
</tr>
<tr>
<td>18:30:50</td>
<td>Pump3</td>
<td>2000</td>
<td>100</td>
<td>110</td>
</tr>
</tbody>
</table>

**MATLAB Function**

**Input Stream**

<table>
<thead>
<tr>
<th>Time window</th>
<th>Pump Id</th>
<th>Bearing Friction</th>
</tr>
</thead>
<tbody>
<tr>
<td>18:00:00</td>
<td>Pump1</td>
<td>5</td>
</tr>
<tr>
<td>18:10:00</td>
<td>Pump3</td>
<td>...</td>
</tr>
<tr>
<td>18:10:00</td>
<td>Pump4</td>
<td>...</td>
</tr>
<tr>
<td>18:20:00</td>
<td>Pump1</td>
<td>...</td>
</tr>
<tr>
<td>18:20:00</td>
<td>Pump3</td>
<td>4</td>
</tr>
<tr>
<td>18:30:00</td>
<td>Pump5</td>
<td>...</td>
</tr>
<tr>
<td>18:30:00</td>
<td>Pump3</td>
<td>5</td>
</tr>
<tr>
<td>18:30:00</td>
<td>Pump4</td>
<td>9</td>
</tr>
</tbody>
</table>

**Output Stream**
Debug your streaming function on live data
Complete your application

Edge Analytics Development

Analytics Development

Compiler SDK

MATLAB

Package & Deploy

Model

Business Decisions

Presentation Layer

kibana

Operator
Access Data
Build Machine Learning models
Deployment
Web apps
MATLAB®
Baker Hughes Develops Predictive Maintenance Software for Gas and Oil Extraction Equipment Using Data Analytics and Machine Learning

By Gulshan Singh, Engineer Manager

Challenge:
Reduce pump equipment costs & downtime

Solution:
Use MATLAB to analyze 1 TB of data and create a neural network to predict machine failures

Results:

Savings of more than $10 million projected
Development time reduced tenfold
Ease of use Multiple types of data

We saw three advantages in using MATLAB [...]. The first is speed; development in C or other language would have taken longer. The second is automation. The third is the wide variety of technologies.

Follow this link to read the complete user story of Gulshan Singh