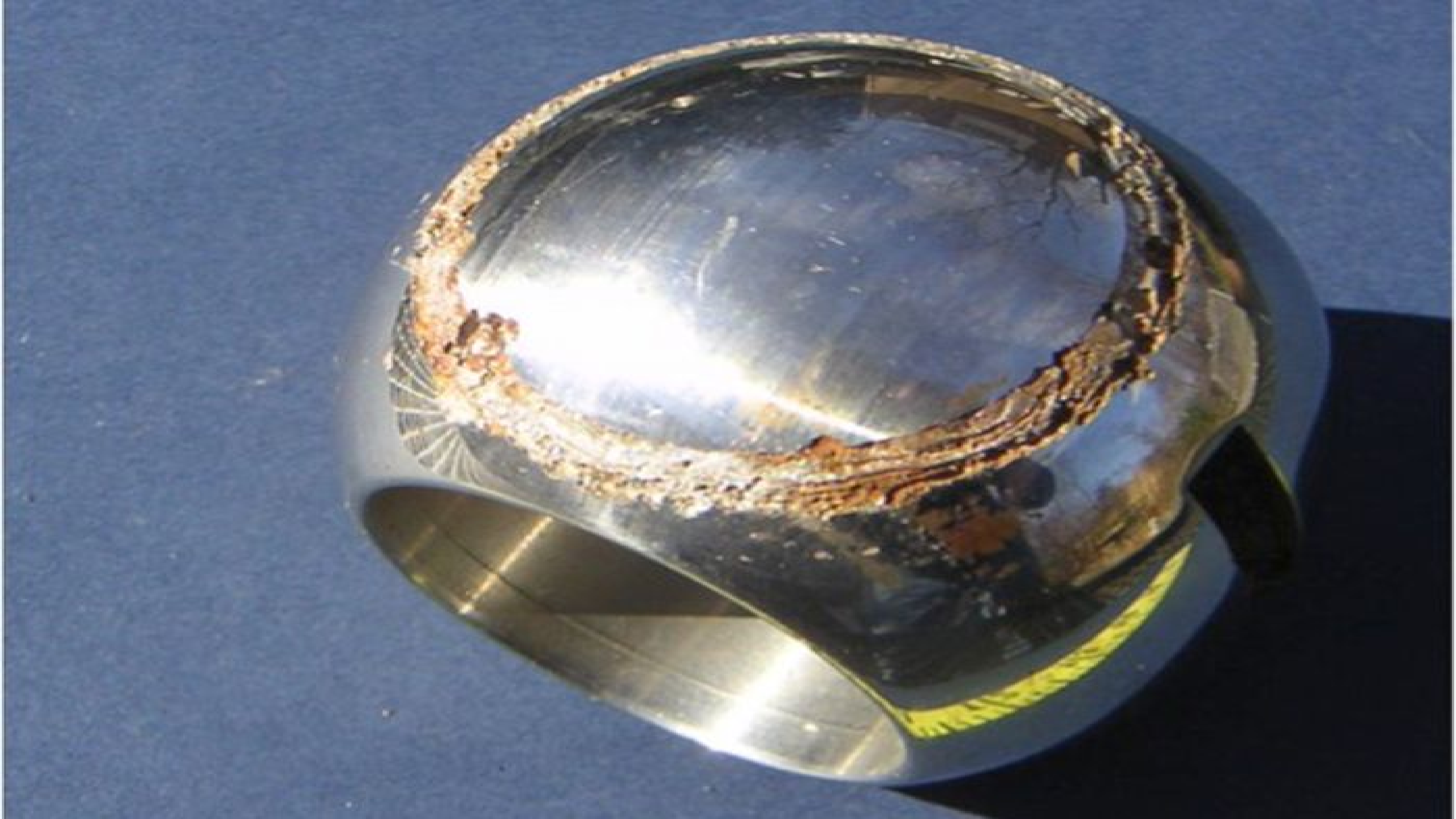
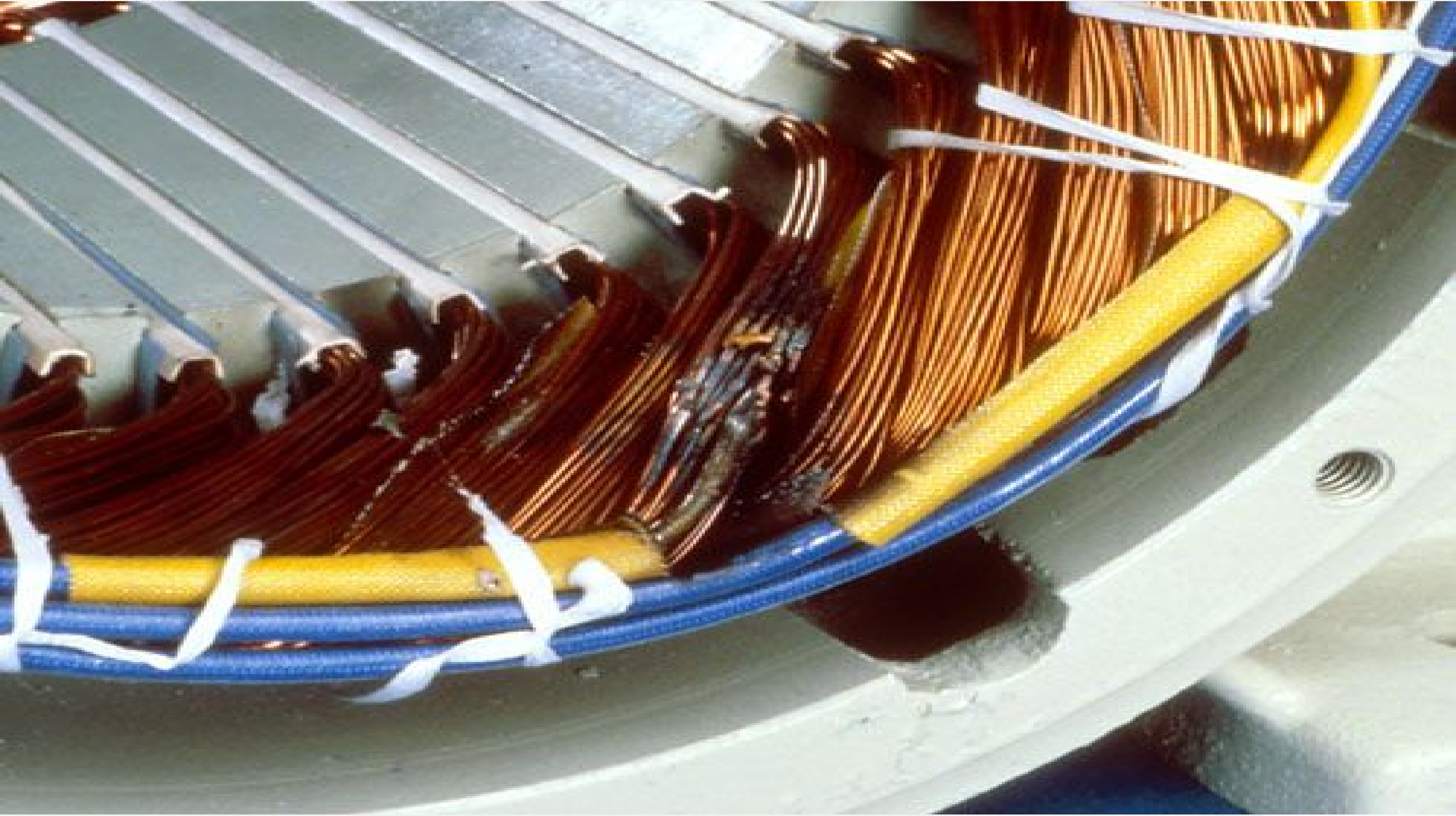
RED HAT
SUMMIT

IoT in Oil and Gas
Architectures and Experiences

Glenn West - Principal Engineer, Red Hat
Michael Costello - Senior Middleware Consultant, Red Hat
John Archer - Senior Energy AppDev Solution Architect, Red Hat
Anne Joseph - CEO, Rutledge Omni Services Group

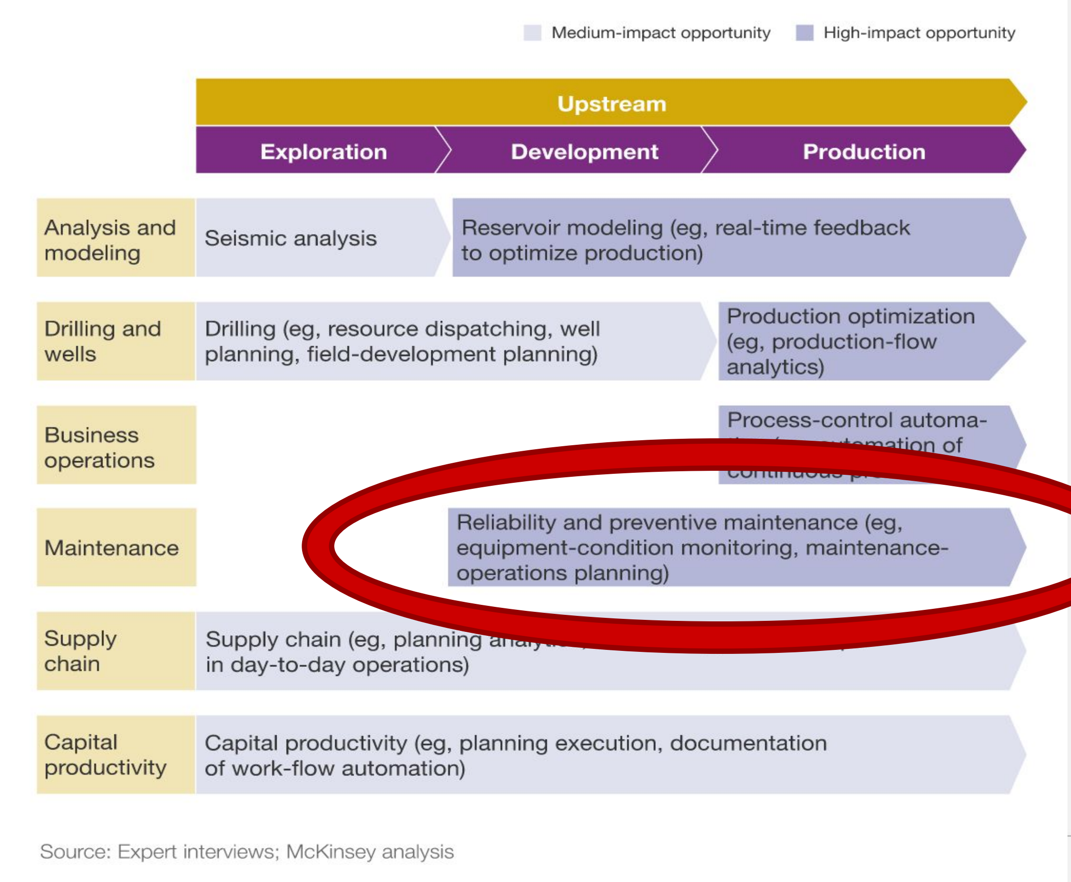




Oil and Gas Common IIoT Use Cases

- Asset Surveillance
  - ESP (Electrical Submersible Pump)
    - Heat, Vibration, Voltage, Flow Rate
  - BOP (Blowout Preventers)
  - Seismic Vessel Streamers
    - Twisting, UV and Salt Corrosion
  - Pipeline and Valves
    - Flow Rate, Corrosion, Pig Status, Leak Detection
- Remote Production Monitoring
- Remote Drilling Monitoring
- HSE - Field Worker
  - Wearables, Location, Certifications, Environmentals

Prioritizing Upstream Use Cases



▫ Medium-impact opportunity ■ High-impact opportunity

Upstream

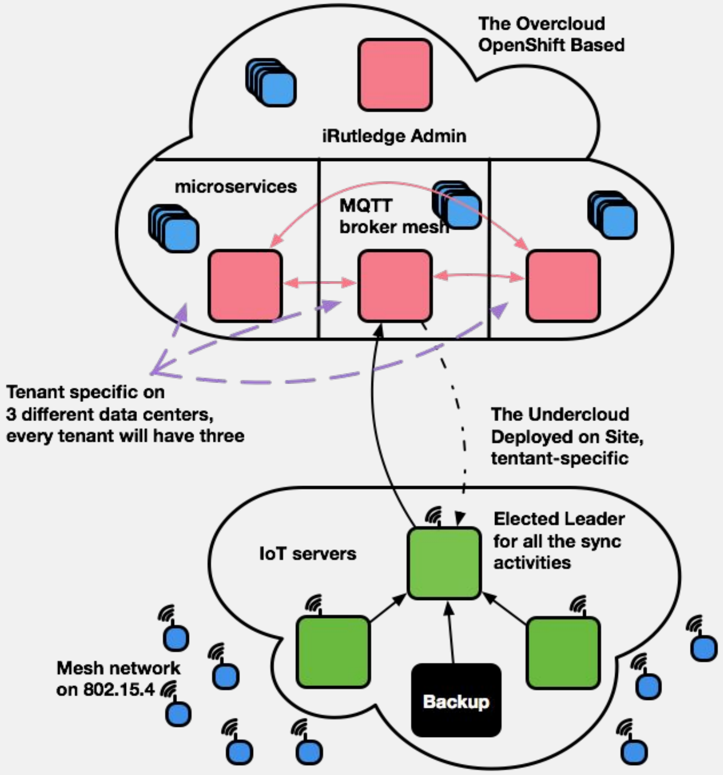
	Exploration	Development	Production
Analysis and modeling	Seismic analysis	Reservoir modeling (eg, real-time feedback to optimize production)	
Drilling and wells	Drilling (eg, resource dispatching, well planning, field-development planning)		Production optimization (eg, production-flow analytics)
Business operations			Process-control automa- tion (eg, automation of continuous p
Maintenance		Reliability and preventive maintenance (eg, equipment-condition monitoring, maintenance-operations planning)	
Supply chain	Supply chain (eg, planning analyti , in day-to-day operations)		
Capital productivity	Capital productivity (eg, planning execution, documentation of work-flow automation)		

Source: Expert interviews; McKinsey analysis

The cost of Non-Productive Time pe asset during drill to completion is $500K - $1M per day, and post completion is $40K-$300K per day or average

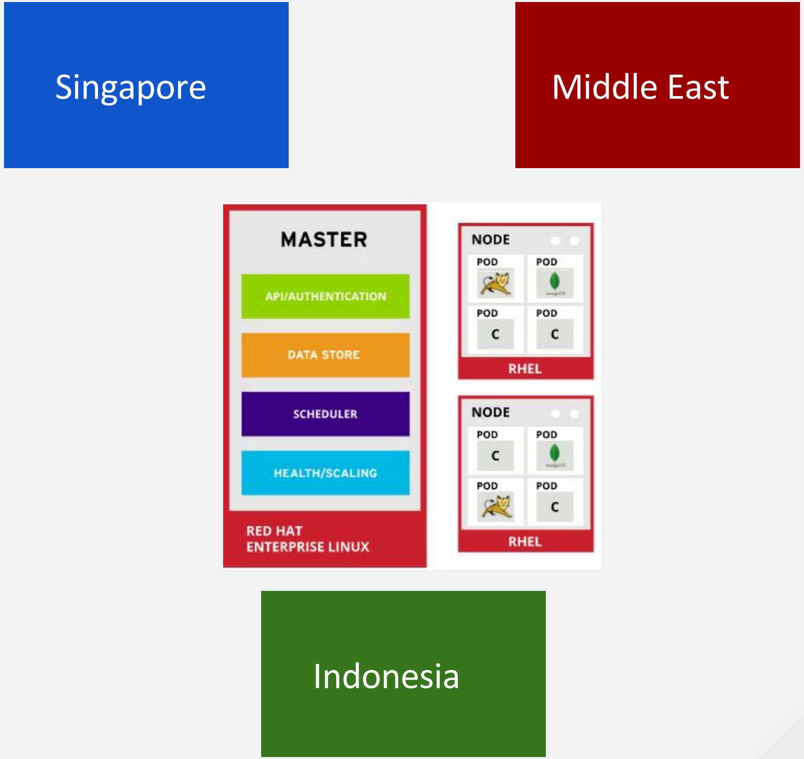
Rutledge Overview

- Three Tier Architecture
  - Overcloud
  - Undercloud
  - IOT



Overcloud

- Three Datacenters
  - Openshift on Bare Metal
  - High Performance Storage
  - RHEV for Dev/Test Needs
  - Hyper-converged Hardware
- Reasons:
  - Ping Time
  - Data Sovereignty
  - Redundancy



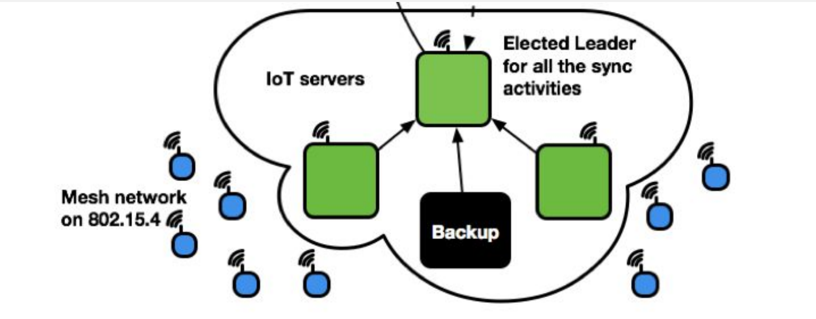
Singapore

Middle East

Indonesia

Undercloud

- Extreme Environment
- Cloud Access – 0 to Infinite
- Mission Critical Local Apps
- Must have local servers
- Redundancy is mandatory
- Blackbox function is needed
- Special Hardware



Undercloud Server

- Project started with concept of traditional IOT Gateway
- During version one, we saw developer things of this more of an Access Point
- We found the power and performance not up to software needs
- Moved to IOT Server Concept

- IOT Server – 3 + 1 on Site
  - Xeon-D
  - NVMe Solid State Storage
  - Dual 10Gig Ethernet
  - Dual IOT Radio
  - Dual Wifi Radio
  - 12 Slots M.2
  - RHEL Atomic

Software Environment

- Projects Started with DevOps and Microservices
- Openshift Confirmed as Overcloud – Containers on Top
- How to do Microservices Everywhere?
- Due to real-time nature of undercloud, needed static environment
- RHEL ATOMIC To Rescue

- Final System is “Microservice” Everywhere”
- A Microservice in the Rutledge system = 1 Container
- Development Environment
  - NodeJS
- Database?

What to do for Database in Scale Out

- Version 1, the database was not specified, any database any microservice
- We found that this resulted in a more traditional approach, not microservice
- Recovery and Reliability took second seat
- Realtime nature and syncing of overcloud and undercloud got forgotten.

- Version 2, Database tightly specified
- Database and Microservice were connected using a framework.
- Rockmsvc framework born
- Solved syncing between onsite IOT Servers and under/over cloud.

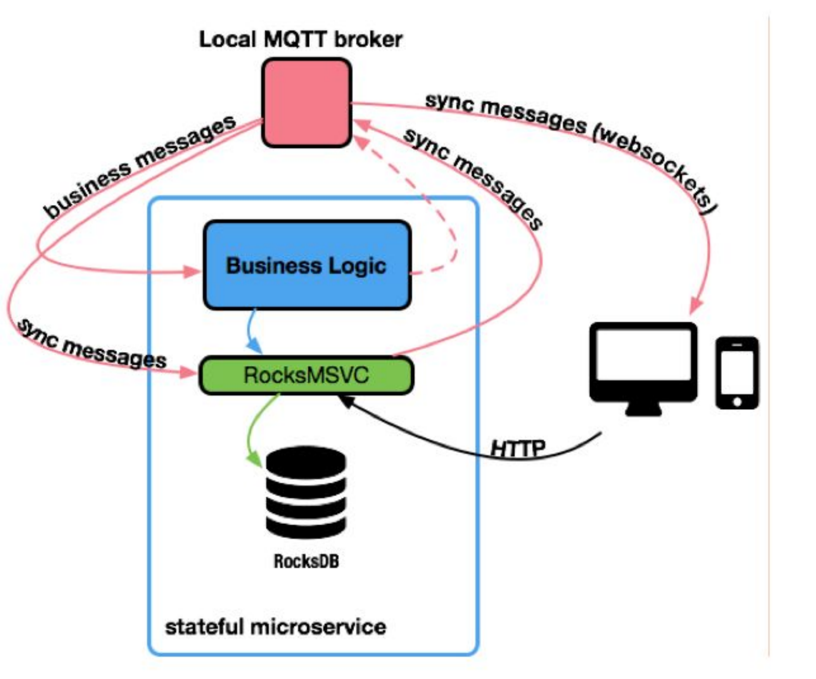
Microservices and Rocks – Version 2

- Architecture rule of 100 Lines of Code per Microservice
- Rockmsvc provides:
  - Rocksdb
  - Rest API
  - Schema Support
  - Support for business Logic
  - Horizontal and Vertical Sync

- Why Rocksdb
  - Designed for SSD and in-memory
  - Compression
  - Low Overhead
- Framework provides:
  - Multi-Table
  - Multi-Key
  - Application Sync over MQTT
  - Local Storage / No SAN Needed
  - Runs on Openshift and RHEL ATOMIC

Microservices

- Takes messages from AMQ, and performs business logic.
- Looks up data in Rocks
- Writes Data In Rocks
- Send message(s) to AMQ
- Rockmsvc will also send update messages on deletes and writes so the 3+1 local copies can stay up to date



IOT Overview

- Fixed Devices
    - Room Sensor
        - Provides a relay point for mesh, and a PIR Sensor to detect Humans
    - Fixed Gas Detectors
        - Provides level of various Poisonous Gas
- Mobile Devices
    - Badge

- IOT Devices
    - Inherently Safe Battery Power
    - Solar Power Top Up
    - Inductive Charging
- Full Bidirection IP V6 Network Device
- MQTT over IPV6

Lessons Learned

- Make sure your architect eats and breaths Microservices
- Buy-in at every Level for Microservice Architecture
- Biggest issues are often not technical – It's people resisting change
- Power Management is King in IOT
- Management of Network Bandwidth is important in IOT Applications – Bandwidth and Power Connected
- Clean Division of Display Logic and Business Logic offers flexibility

Global Midstream Oil & Gas Company

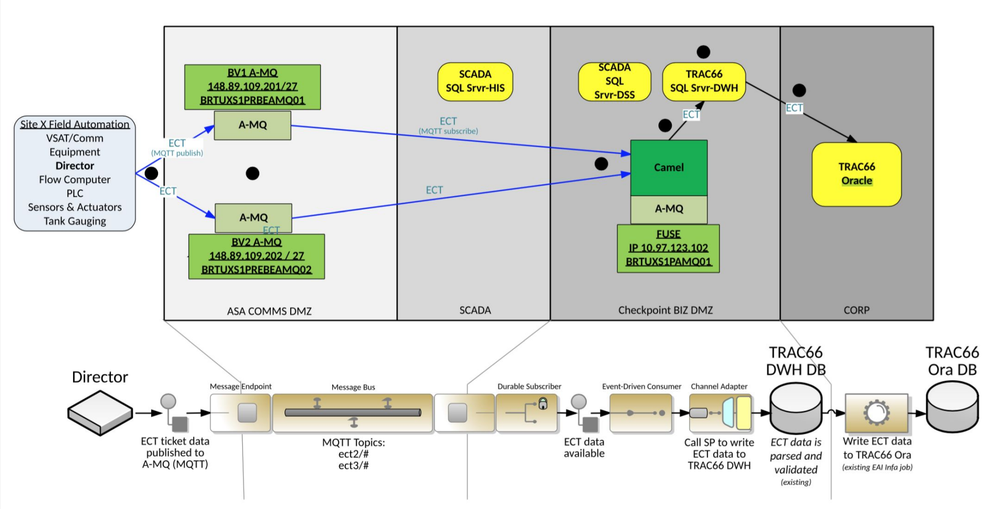
Red Hat JBoss Fuse & A-MQ in SCADA system to monitor flow of oil/gas in pipelines

Partnered with Red Hat to adopt a software defined approach to taking sensor data off the pipelines, and ingest it into their SCADA system over the IoT protocol, MQTT.

Lowered their operational IT spend for MQ

Adopted open standards

- Easier to maintain
- Faster innovation
- More secure

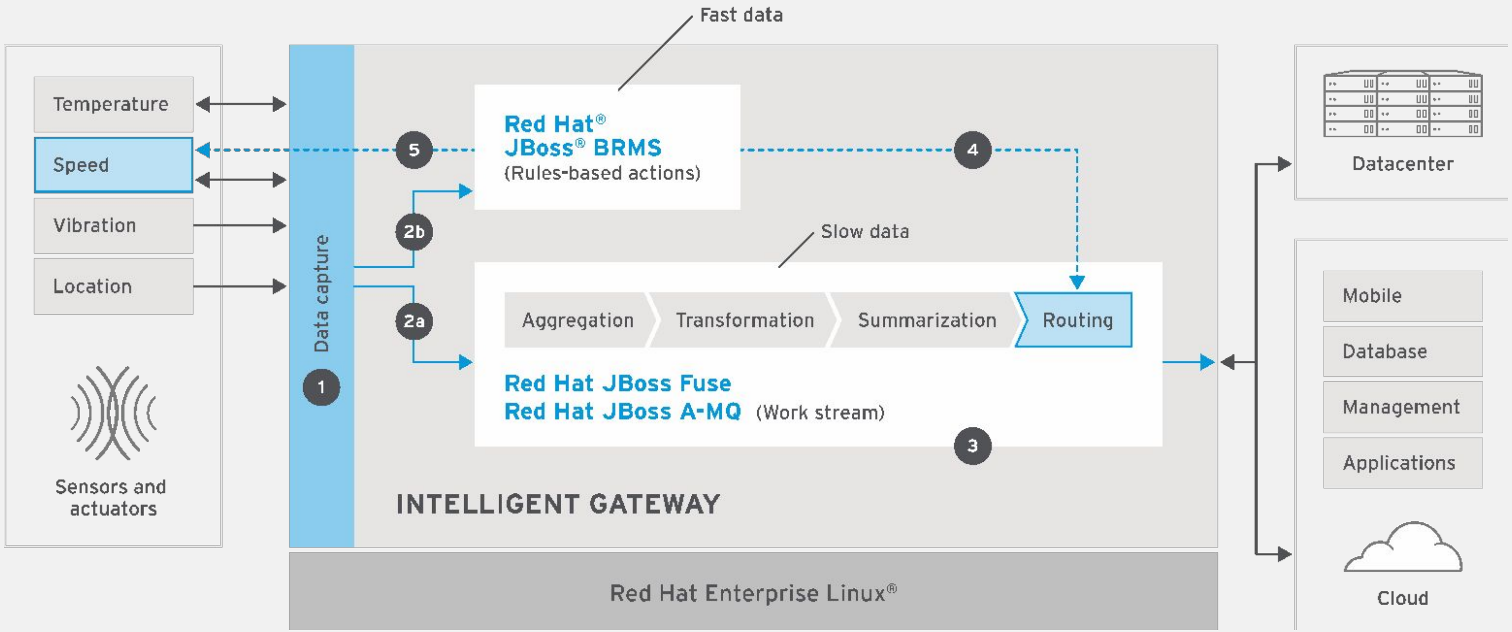


How to Future Proof the IT OT gap

- Prevent new Silos - Single Backplane
- Validating Custom vs Package
- General Protocol Handling
- Message Formats
- New Sensors
- New Use Cases
- Intelligent Bandwidth Utilization

Intelligent Gateway Architecture

Transforming device data into actionable information



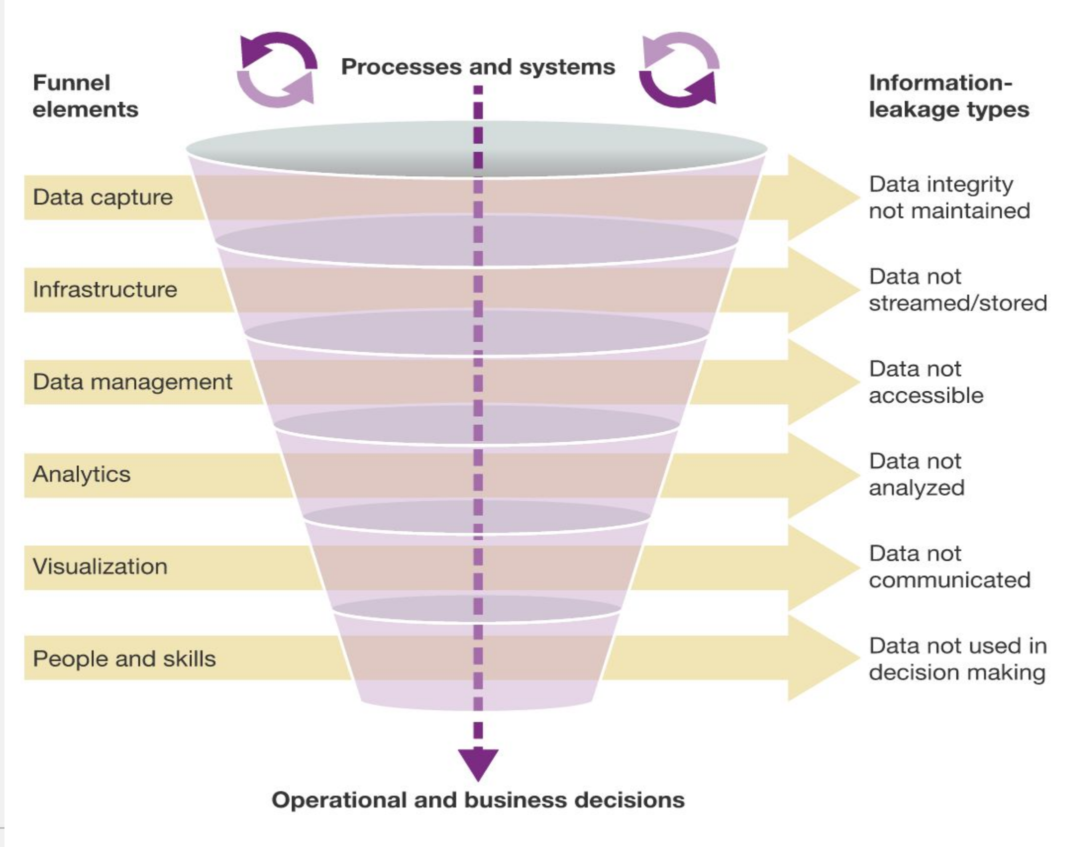
Red Hat and Open Source for IoT

eclipse
Drools
Apache Camel
fedora
ActiveMQ
kura
Kapua

Our open source solutions

- Free you from proprietary lock-in and cost escalation
- Capture community innovation
- Provide the enterprise-level security, reliability, scalability and support required by the IoT
- Bring the internet of things to life quickly, cost-effectively, and with lower risk.

Data Centricity - Oil and Gas Data Opportunities



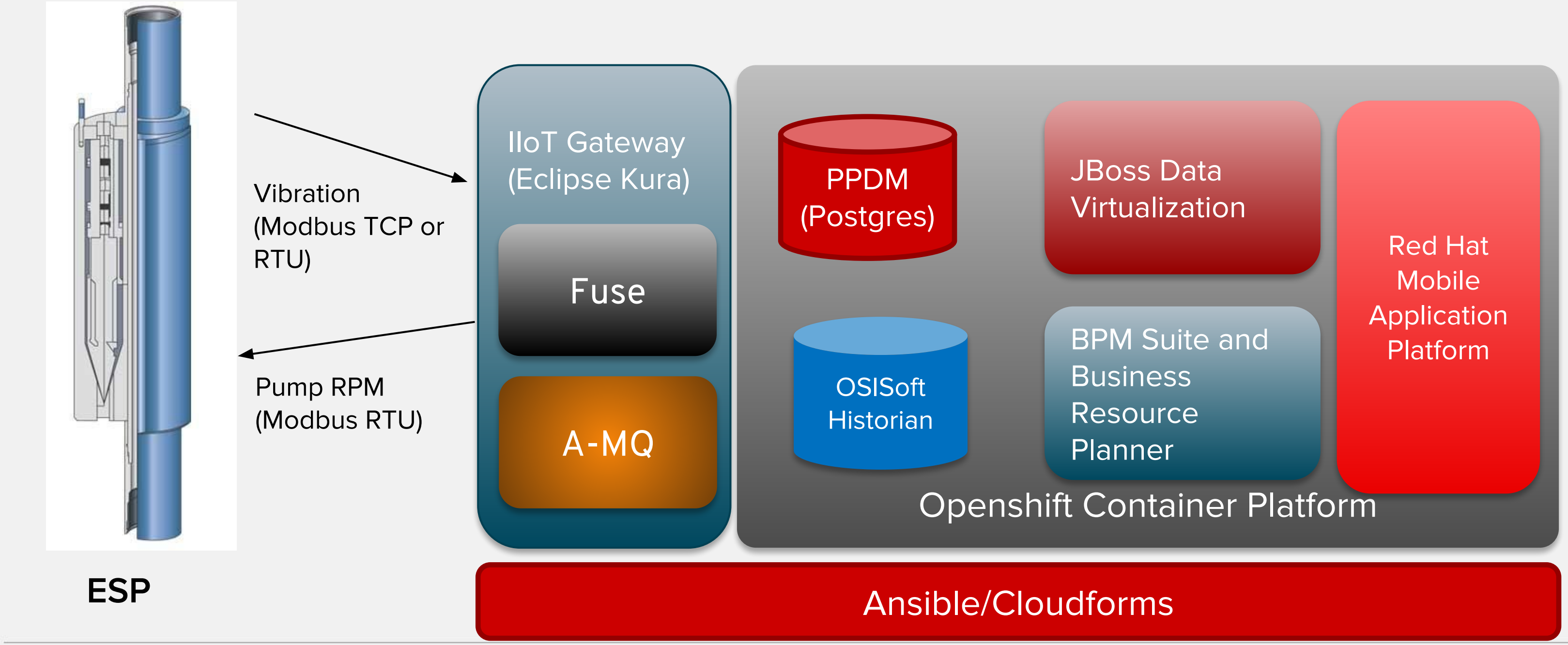
Application Focus
Challenges the package applications to be maintained and provide net new value of the support cost – data locked away

Data Focus
Services can be leveraged and composed to create net new value to build applications rapidly to deliver immediately. Reuse enhanced with DaaS.

Electrical Submersible Pump Machine Health

- **Challenges**
  - Deferred production
    - Lost oil due to ESP downtime
    - Underperforming ESP
  - Inefficient pump operation
  - Faster equipment deterioration due to misoperation
- **Solution**
  - Effective ESP Machine Health will increase the average run-life of an ESP and decrease downtime.
  - Optimize pump flow rate
  - Minimize operational risk (e.g. operation under low flow conditions)
  - Reduce number of stops and starts
  - Optimize pump replacement

Machine Health Demo Architecture



Vibration (Modbus TCP or RTU)

Pump RPM (Modbus RTU)

ESP

IIoT Gateway (Eclipse Kura)

Fuse

A-MQ

PPDM (Postgres)

JBoss Data Virtualization

OSISoft Historian

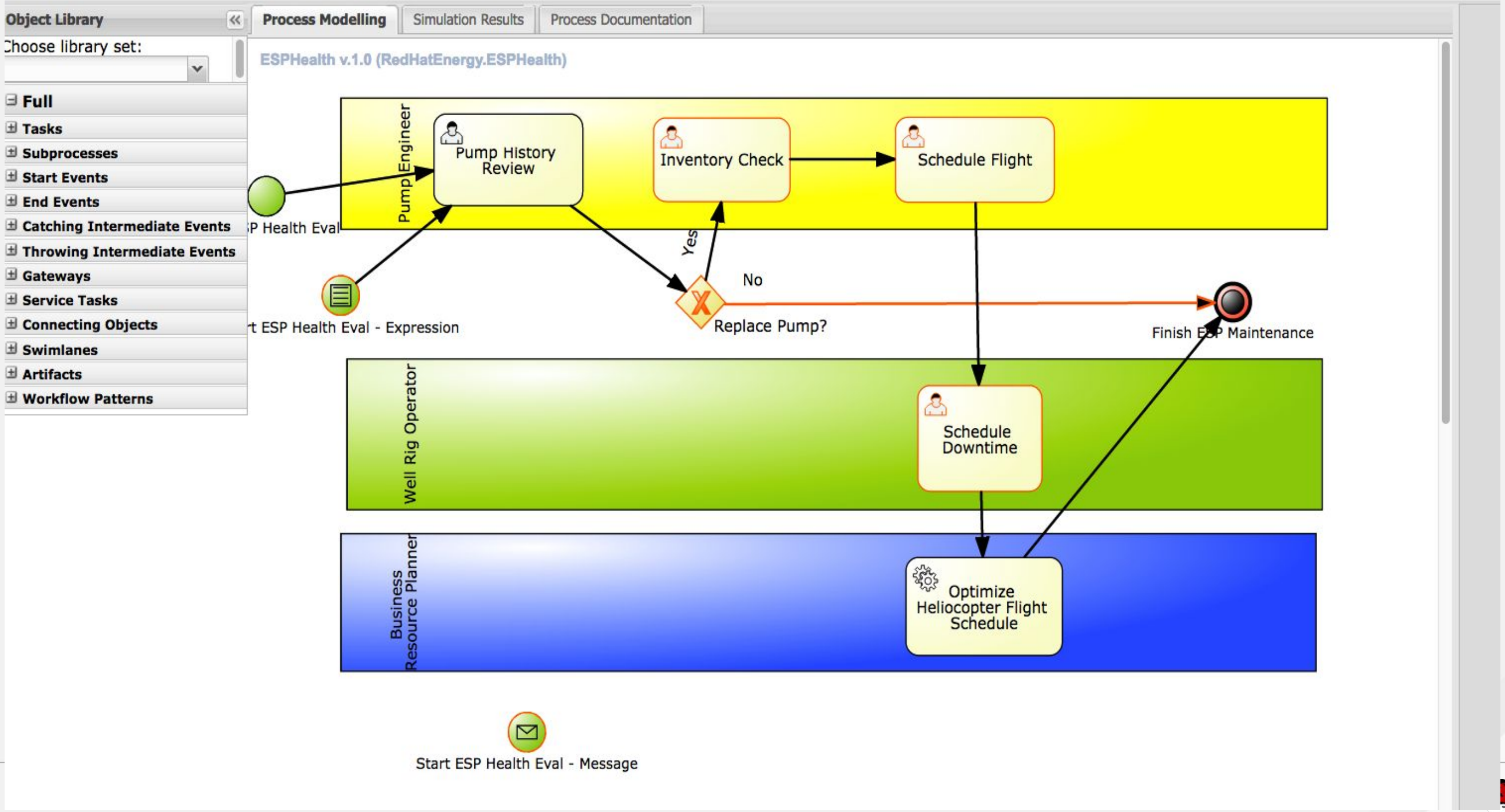
BPM Suite and Business Resource Planner

Red Hat Mobile Application Platform

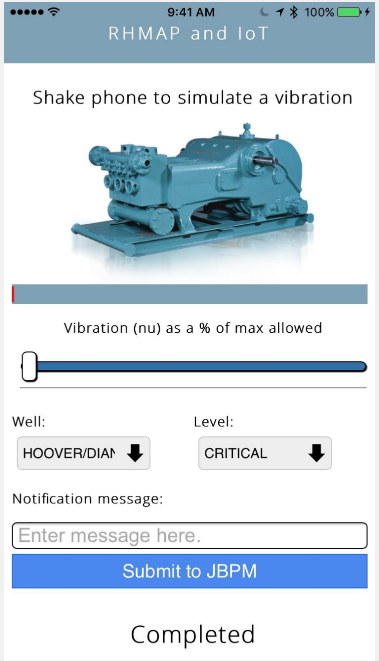
Openshift Container Platform

Ansible/Cloudforms

ESP Replacement Business Optimization - BPMN



ESP Demo Walkthrough



9:41 AM 100%

RHMAP and IoT

Shake phone to simulate a vibration

Vibration (nu) as a % of max allowed

Well: HOOVER/DIAN

Level: CRITICAL

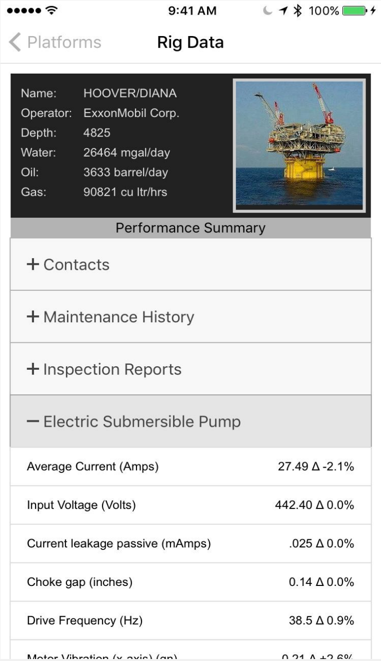
Notification message:

Enter message here.

Submit to JBPM

Completed

ESP Vibration Simulation



9:41 AM 100%

Platforms | Rig Data

Name:	HOOVER/DIANA
Operator:	ExxonMobil Corp.
Depth:	4825
Water:	26464 mgal/day
Oil:	3633 barrel/day
Gas:	90821 cu ltr/hrs

Performance Summary

+ Contacts

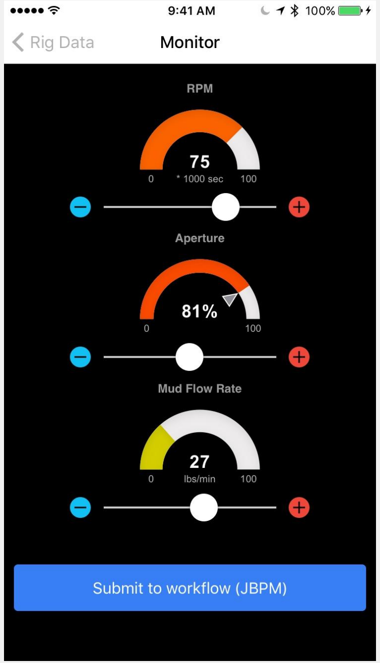
+ Maintenance History

+ Inspection Reports

– Electric Submersible Pump

Average Current (Amps)	27.49 Δ -2.1%
Input Voltage (Volts)	442.40 Δ 0.0%
Current leakage passive (mAmps)	.025 Δ 0.0%
Choke gap (inches)	0.14 Δ 0.0%
Drive Frequency (Hz)	38.5 Δ 0.9%
Motor Vibration (x-axis) (gn)	0.21 Δ +2.6%

ESP Remote Metrics



9:41 AM 100%

Rig Data | Monitor

RPM: 75 (* 1000 sec; 0–100)

Aperture: 81% (0–100)

Mud Flow Rate: 27 lbs/min (0–100)

Submit to workflow (JBPM)

ESP Remote Control

Global visibility to Production KPIs

Demo ESP Machine Health Scenario

- Non Productive Time (NPT) and Invisible Lost Time (ILT) involved with production sites and personnel
- Just in Time synchronization for equipment and field workers
- Enable Remote Operators and SMEs
- End-to-end visibility of Maintenance Cycle
- Consume real-time machine health information
- Continuous Deployment and Improvement

RED HAT SUMMIT

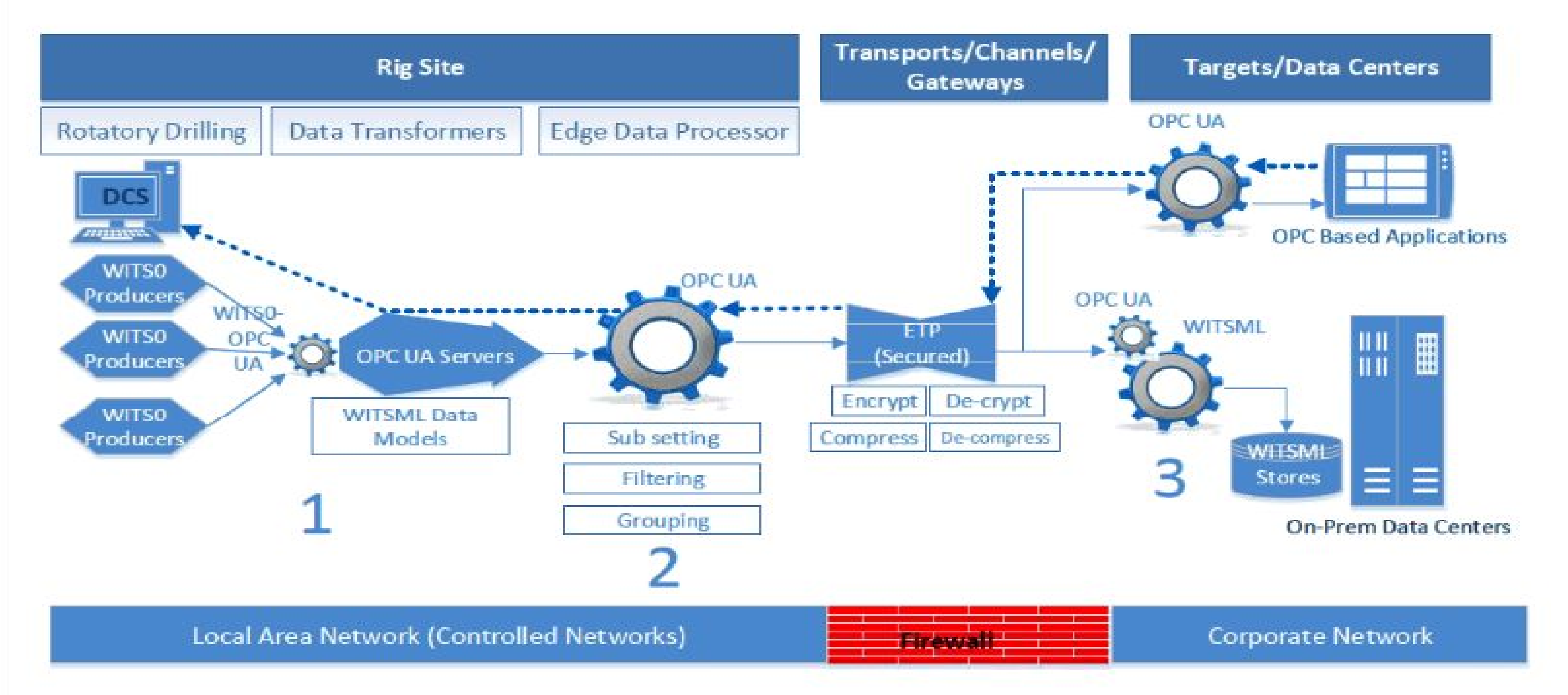
THANK YOU

- plus.google.com/+RedHat
- linkedin.com/company/red-hat
- youtube.com/user/RedHatVideos
- facebook.com/redhatinc
- twitter.com/RedHatNews

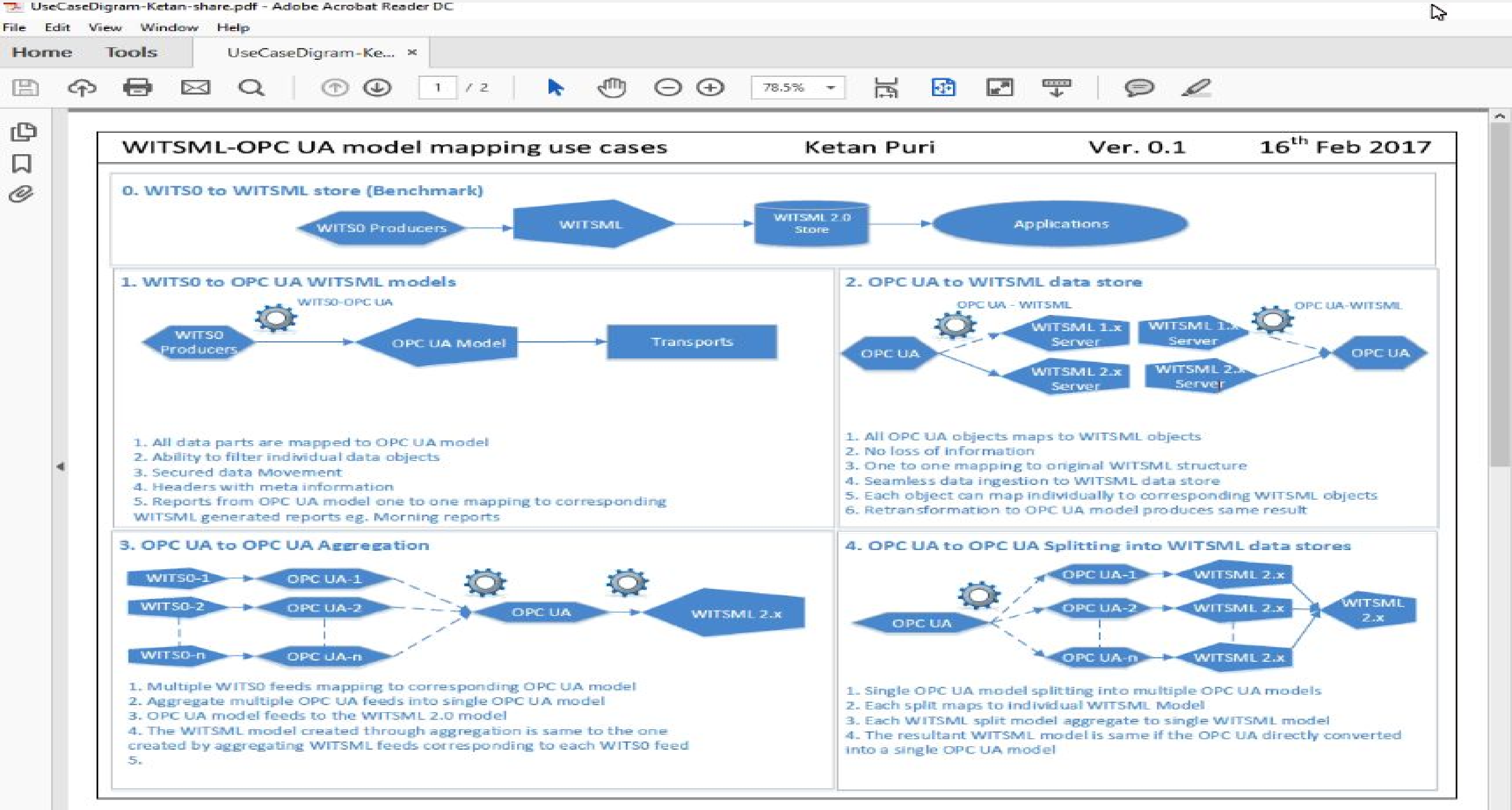
RED HAT
SUMMIT

LEARN. NETWORK.
EXPERIENCE
OPEN SOURCE.

Energistics WITSML – OPC-UA Reference Architecture



Energistics WITSML – OPC-UA Use Cases



WITSML-OPC UA model mapping use cases Ketan Puri Ver. 0.1 16th Feb 2017

0. WITS0 to WITSML store (Benchmark)

1. WITS0 to OPC UA WITSML models

1. All data parts are mapped to OPC UA model
2. Ability to filter individual data objects
3. Secured data Movement
4. Headers with meta information
5. Reports from OPC UA model one to one mapping to corresponding WITSML generated reports eg. Morning reports

2. OPC UA to WITSML data store

1. All OPC UA objects maps to WITSML objects
2. No loss of information
3. One to one mapping to original WITSML structure
4. Seamless data ingestion to WITSML data store
5. Each object can map individually to corresponding WITSML objects
6. Retransformation to OPC UA model produces same result

3. OPC UA to OPC UA Aggregation

1. Multiple WITS0 feeds mapping to corresponding OPC UA model
2. Aggregate multiple OPC UA feeds into single OPC UA model
3. OPC UA model feeds to the WITSML 2.0 model
4. The WITSML model created through aggregation is same to the one created by aggregating WITSML feeds corresponding to each WITS0 feed
5.

4. OPC UA to OPC UA Splitting into WITSML data stores

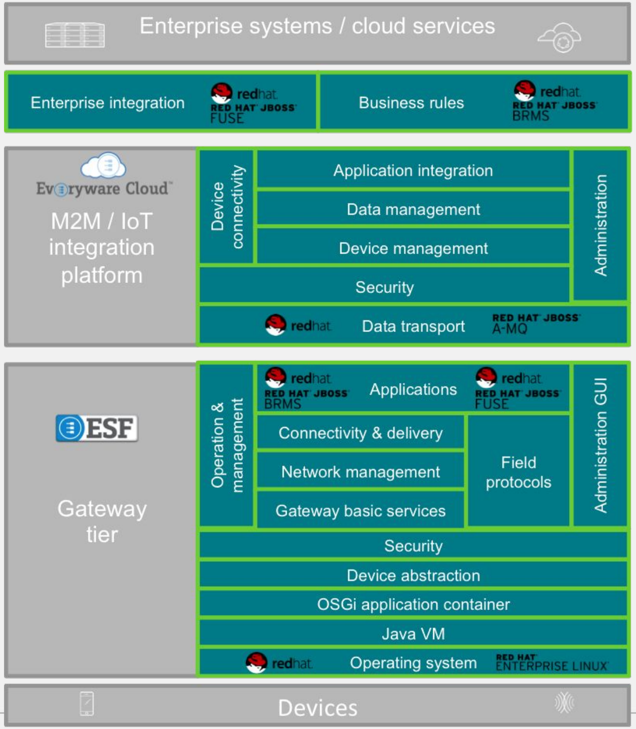
1. Single OPC UA model splitting into multiple OPC UA models
2. Each split maps to individual WITSML Model
3. Each WITSML split model aggregate to single WITSML model
4. The resultant WITSML model is same if the OPC UA directly converted into a single OPC UA model

Intelligent Gateway & IoT Cloud

Data acquisition, integration, & rules activation, providing dynamic intelligence at the edge

Bridge between IT and OT by streamlining the many data formats and velocities

Process and act on data at scale
Reduce latency and bandwidth
Apply real-time decisions locally
Transform IoT data and connect with enterprise systems
Control and manage millions of IoT devices



(IIOT) Beyond the Gateway

- As a device messaging protocol, Modbus is ubiquitous; however, its limitations make it **non-ideal for highly available messaging needs**
- Other TCP based messaging standards have evolved to carry modbus payloads
  - MQTT
  - STOMP
  - OPC (highly used in many IIOT use cases)
  - AMQP
  - Others

(IIOT) MQTT

“**MQTT** is an ISO standard (ISO/IEC PRF 20922) machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an ***extremely lightweight publish/subscribe messaging transport***. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium. For example, it has been used in sensors communicating to a broker via satellite link, over occasional dial-up connections with healthcare providers, and in a range of home automation and small device scenarios. It is also ideal for mobile applications because of its small size, low power usage, minimised data packets, and efficient distribution of information to one or many receivers”

source : mqtt.org

(IIOT) MQTT Characteristics

- MQTT provides a push based **publish subscribe** protocol based on its set of protocol level control packets

- This architecture occurs on top of TCP

- MQTT's control packets are kept as small as possible **enabling high latency, low bandwidth networks**

- MQTT carries a **data agnostic payload** (it doesn't have to be modbus…it could be any type of serializable binary)

- MQTT's QoS, Session Persistence, Last Will and Testament, and Retained Message Flags provide capabilities uniquely suited to devices

(IIOT) MQTT
Provides a “Stateless Session” Scope

- With the use of retained messages and LWT feature, **MQTT** as a protocol allows other clients to bind into a sort of **“stateless session scope”**
  - MQTT subscribers may event based on LWT messages
    - IIOT Example: A monitoring application receives a LWT message from a critical system that it is “offline”
- Retained message flag allows publishers to avail subscribers of their values without having to maintain state of whether or not subscribers were interested in the publisher's last relevant message
- Due to the use of the MQTT keepalive feature, clients become aware of the state of publishers in combination with their LWT

(IIOT) MQTT Best Practices

- **Report by Exception** – all of the data all of the time may not be valuable
  - *IIOT Example: Temperature fluctuations*

- **Just Good Enough Delivery - The broker must prevail**
  - Sacrifice message durability for broker performance
    - *IIOT Example: Durable chirps may cause a broker to deliver millions (or more) of message in order at once*
- Be wary of durable subscriptions, make sure to protect the broker against them!

(IIOT) MQTT ActiveMQ

- ActiveMQ provides an **MQTT based TCP transport** for IOT Messaging:

```
<transportConnector
name="mqtt+nio"uri="mqtt+nio+ssl://localhost:1883"/>
```

- Transport has available options for:
  - Keep Alive
  - Subscription Strategy
  - Message prefetch

(IIOT) AMQ Security Survival Guide

- AccessControlLists’S are critical

```
<authorizationEntries>
    <authorizationEntry topic="field.device.1.>"
read="field-device-consumers" write="field-device-producers"
admin="admin"/>
    <authorizationEntry topic="field.device.1.iotdata>"
read="iot-consumer" write="iot-producer" admin="admin"/>
</authorizationEntries>
```

- Restrict publishers and subscribers to only what they need
  - *Don’t allow #,> on high level topic namespaces*

(IIOT) AMQ Security Survival Guide

PROTECT THE BROKER with destination policies

```
<policyEntry topic="CRITICAL.IOT.DATA.>">
    <pendingMessageLimitStrategy>
        <constantPendingMessageLimitStrategy limit="10"/>
     </pendingMessageLimitStrategy>
     <subscriptionRecoveryPolicy>
          <timedSubscriptionRecoveryPolicy recoverDuration="10000"/>
     </subscriptionRecoveryPolicy>
</policyEntry>
```

(IIOT) AMQ Security Survival Guide (continued)

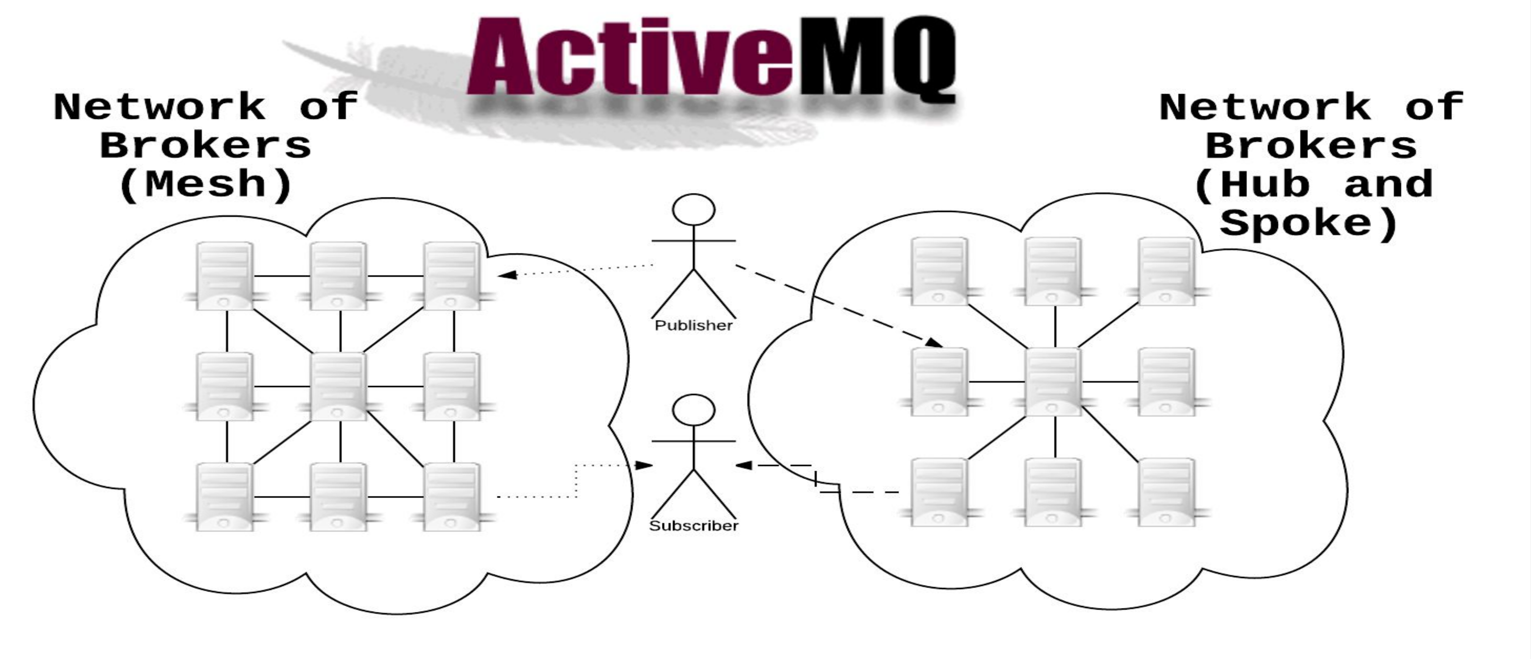
- **Secure The Transport**
  - **SSL/TLS** – ensure the wire is encrypted
  - Narrow the zone of trust (ensure survival of the edge)
  - This can help assure prevention of device and broker intrusion
- **Message Level Authorization**
  - Interceptors that apply custom message based authorization to messages (inspect payloads, headers, etc.)

```
<messageAuthorizationPolicy>
    <bean class="com.redhat.iiot.FactoryAuthorizationPolicy"
xmlns="com.redhat.iiot"/>
</messageAuthorizationPolicy>
```

(IIOT) Highly Available AMQ Based Broker Topologies Enable IIOT

- ***Elastically scale*** to device load
  - HA topologies in use with PAAS appliances such as OpenShift allow for message brokers to elastically scale to pub/sub needs
- Topologies ensure ***high availability*** and distribution of load
- AMQ has various capabilities with durability of its journal that allows most MQTT QoS to flourish at ***incredibly high volumes***
  - KahaDB
  - Memory Persistence Adapter – Message simply dispatch in memory

(IIOT) Highly Available Broker Topologies


ActiveMQ
Network of Brokers (Mesh)
Network of Brokers (Hub and Spoke)
Publisher
Subscriber