Protocols

Industrial IoT

Software and Services Group
IoT Developer Relations, Intel

IIoT Connectivity

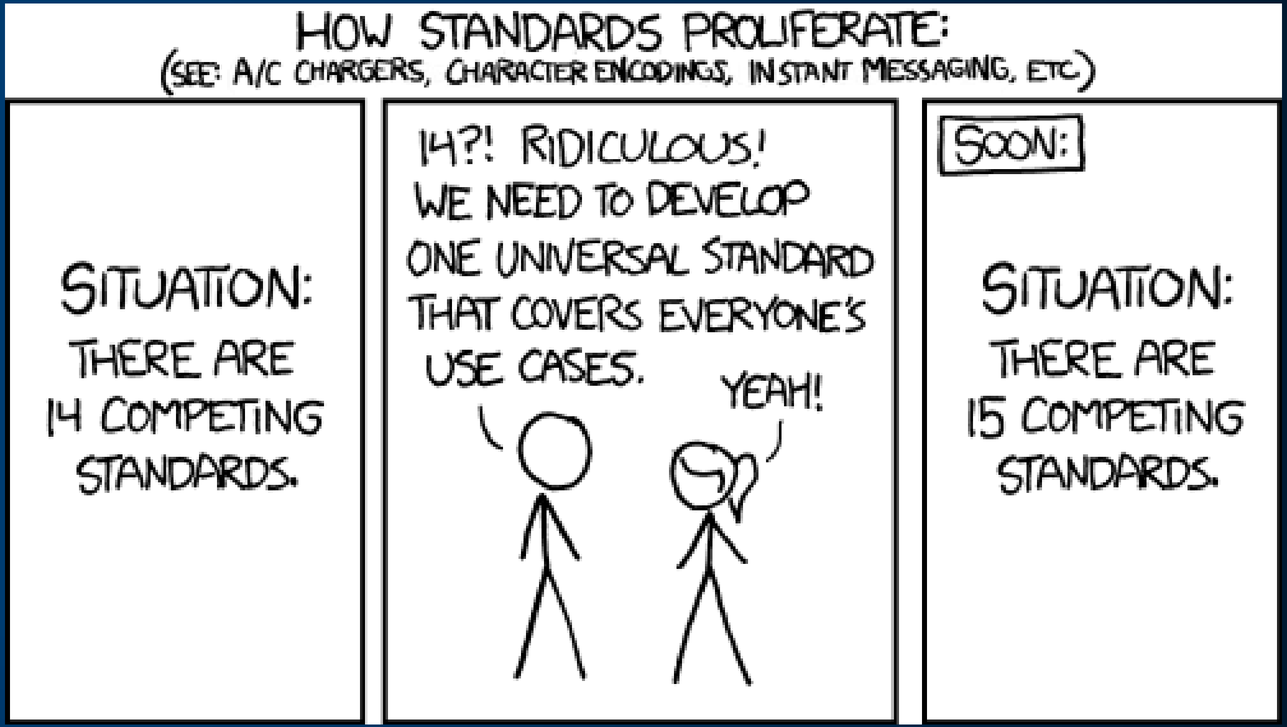
Goal: Seamless Information Sharing

Challenge: Myriad of domain specific connection technologies, Brownfield connectivity

IIoT systems usually integrate with **brownfield** technologies to preserve the capital investments, and **greenfield** technologies to spur innovation.

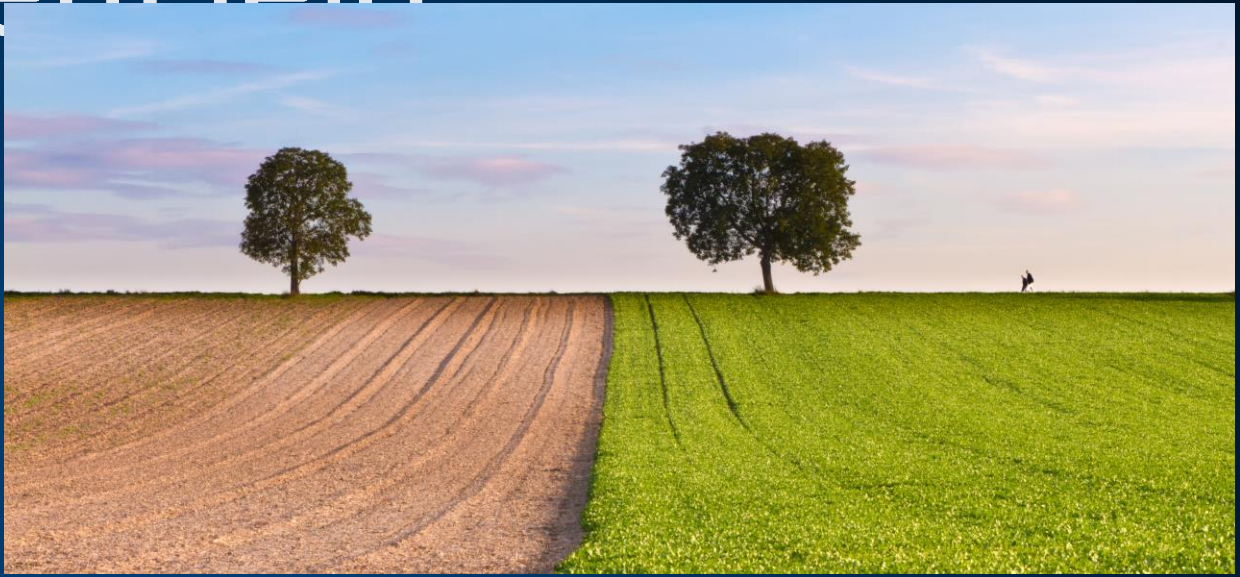
Industrial Protocols

- Industrial Ethernet: PROFINET*, EtherNet/IP*, EtherCAT*
- Automation: Modbus* , DeviceNET*, ZigBee*
- Control Systems: OPC/UA*, DDS*
- Vehicle Automation, Power Systems, Meter Reading, Etc.


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

BrownField / GreenField

- Brownfield Integration: Preserve capital investments, lower cost
- Greenfield Integration: Spur innovation, increase efficiency, higher cost
- Further complications of interoperability



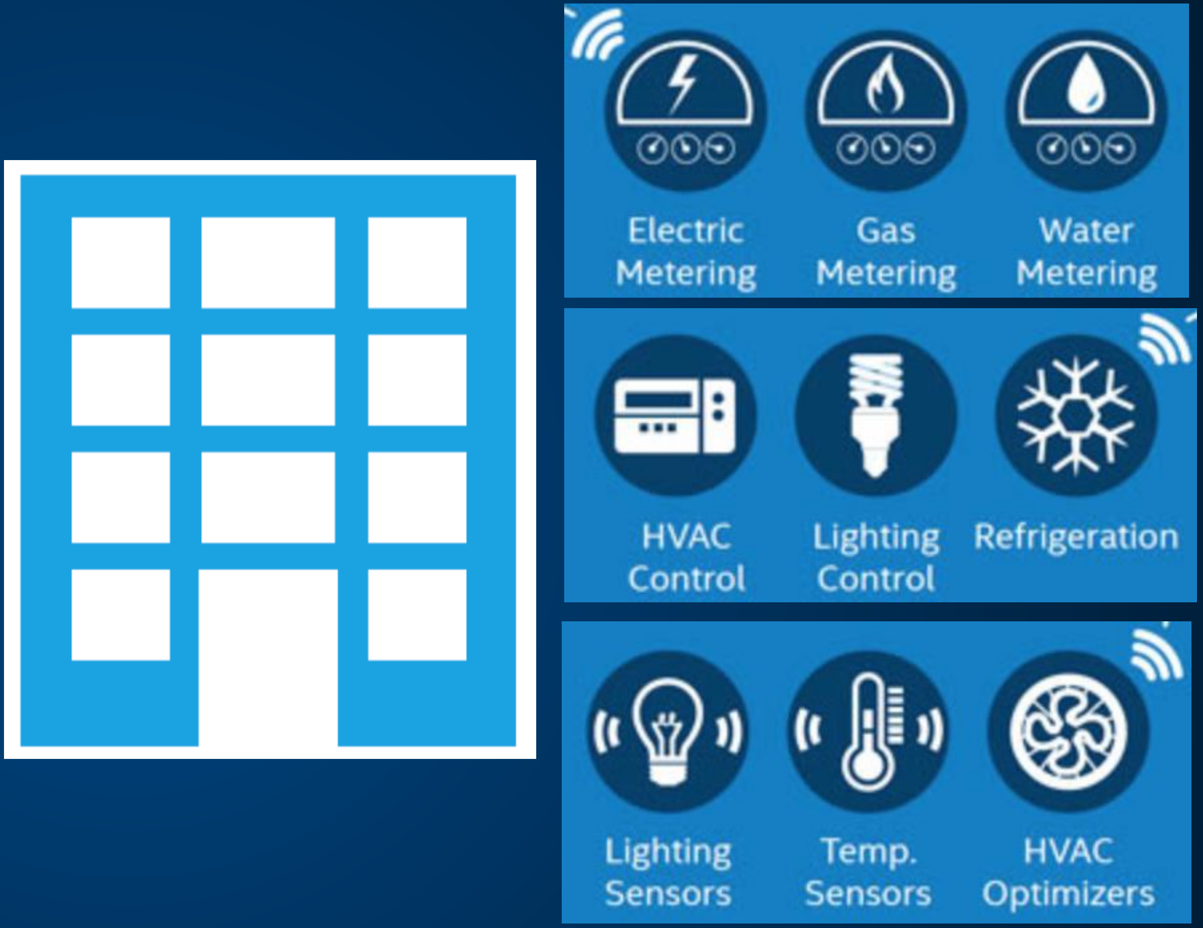
Case Study: Rudin, NANTUM, Prescriptive Data

Goal

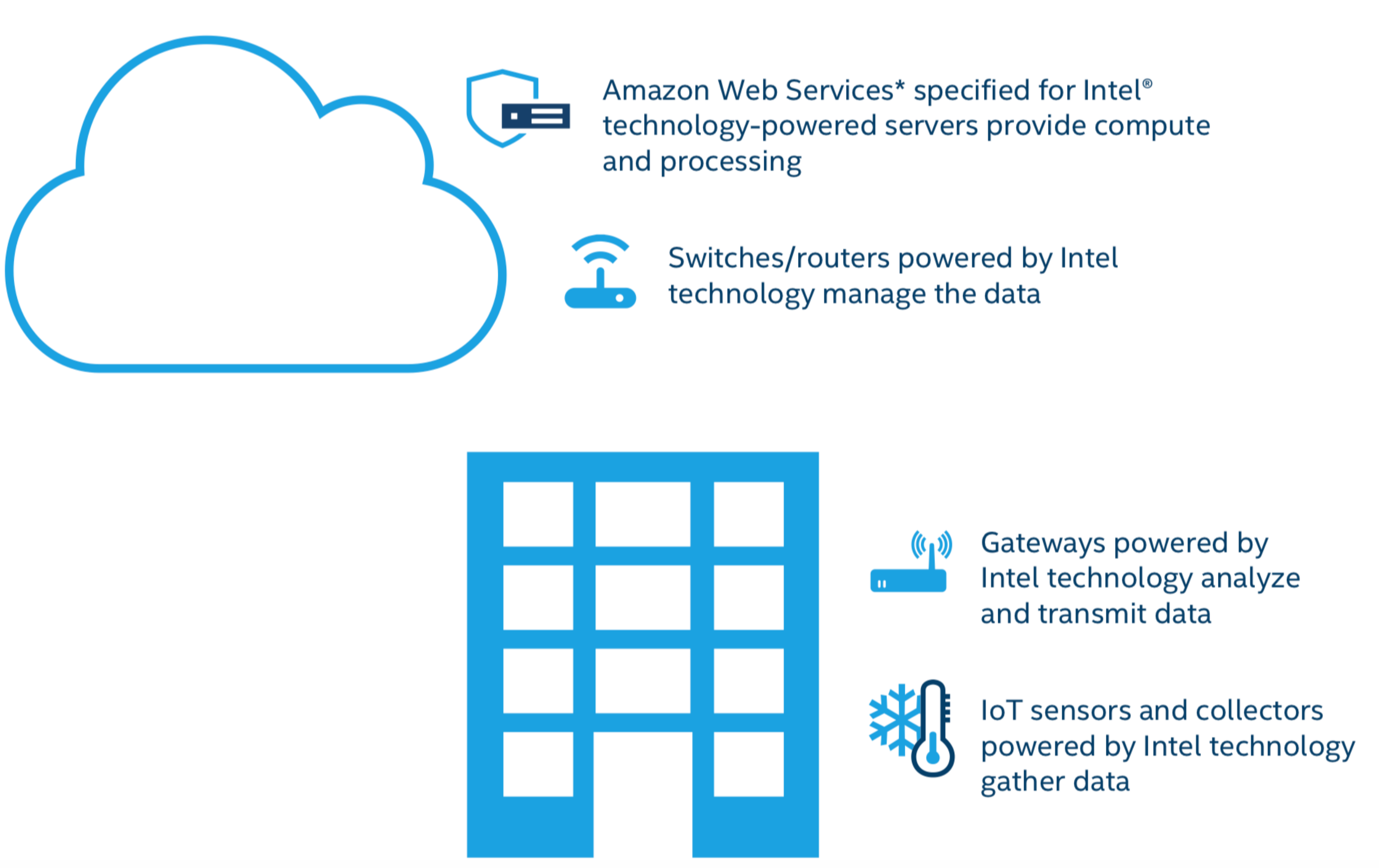
- Rudin*: Real-estate Management Company in NYC – 15M sq/ft
- Improve comfort, increase sustainability, lower operational costs
- Build smart building management system

Complications

- “Islands of Information” – mix of equipment, vendors, protocols, interfaces
- ~2.5 months to integrate all systems for single building
- System run PC inside building – Databases and Learnings not easily shared, limited I/O



Solution



Amazon Web Services* specified for Intel® technology-powered servers provide compute and processing

Switches/routers powered by Intel technology manage the data

Gateways powered by Intel technology analyze and transmit data

IoT sensors and collectors powered by Intel technology gather data

- Rudin OS
- Intel Gateway & Sensors
- Intel Switches/Routers
- AWS Cloud Technology Integration

Results

- Nantum OS – Scalable, automation API Library, Enterprise Grade Security
- 17 Buildings online in a single year
- Energy Use down 9%, Hot/Cold Calls down 70%, $1M lower operational costs in first year
- Rudin created Prescriptive Data – joined Intel IoT Soultions Alliance, qualified as an Intel IoT Market Ready Solution, expected savings $.55 USD / FT^2

beco
cisco
ARROW ELECTRONICS
AUTOMATEDLOGIC
United Technologies
Delta CONTROLS
ECORITHM
FIRECOM
cisco Meraki
LUTRON
Honeywell
intel experience what's inside
Johnson Controls
LENEL
lōtik
SENSOURCE
VERDIGRIS
OBVIUS
Schindler
Schneider Electric
Andover Continuum
SIEMENS
SQUARE D by Schneider Electric

Cross Platform Sensor Support: Mraa

MRAA - An I/O Library for the Internet of Things

Provides I/O abstraction across both Intel and non-Intel (community added) MCU boards, UNIX boards and IoT Gateways.

- X86
- Minnowboard
- NUC
- UP2 Board
- Intel® NUC
- UP* and UP Squared*
- Arduino* and Genuino* 101

ARM

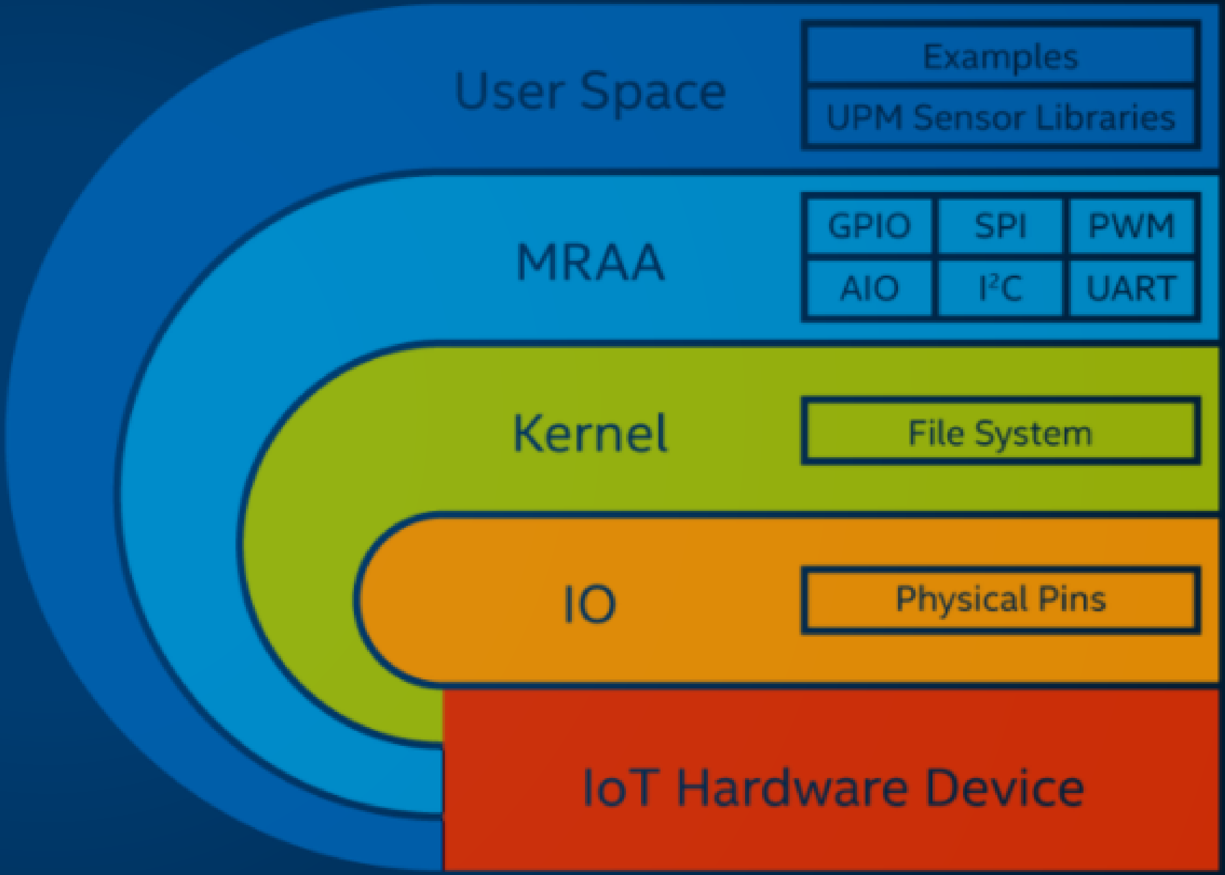
Raspberry Pi

Banana Pi

Beaglebone Black

phyBOARD-Wega

96Boards



https://github.com/intel-iot-devkit/mraa

Making Sensors and Actuators Accessible

Support for Multiple Operating Systems

Zephyr · yocto PROJECT · WIND · ubuntu · ubilinux · archlinux · openSUSE · androidthings

Support for Multiple Languages

C · C++ · Java · Lua · python™ · JS

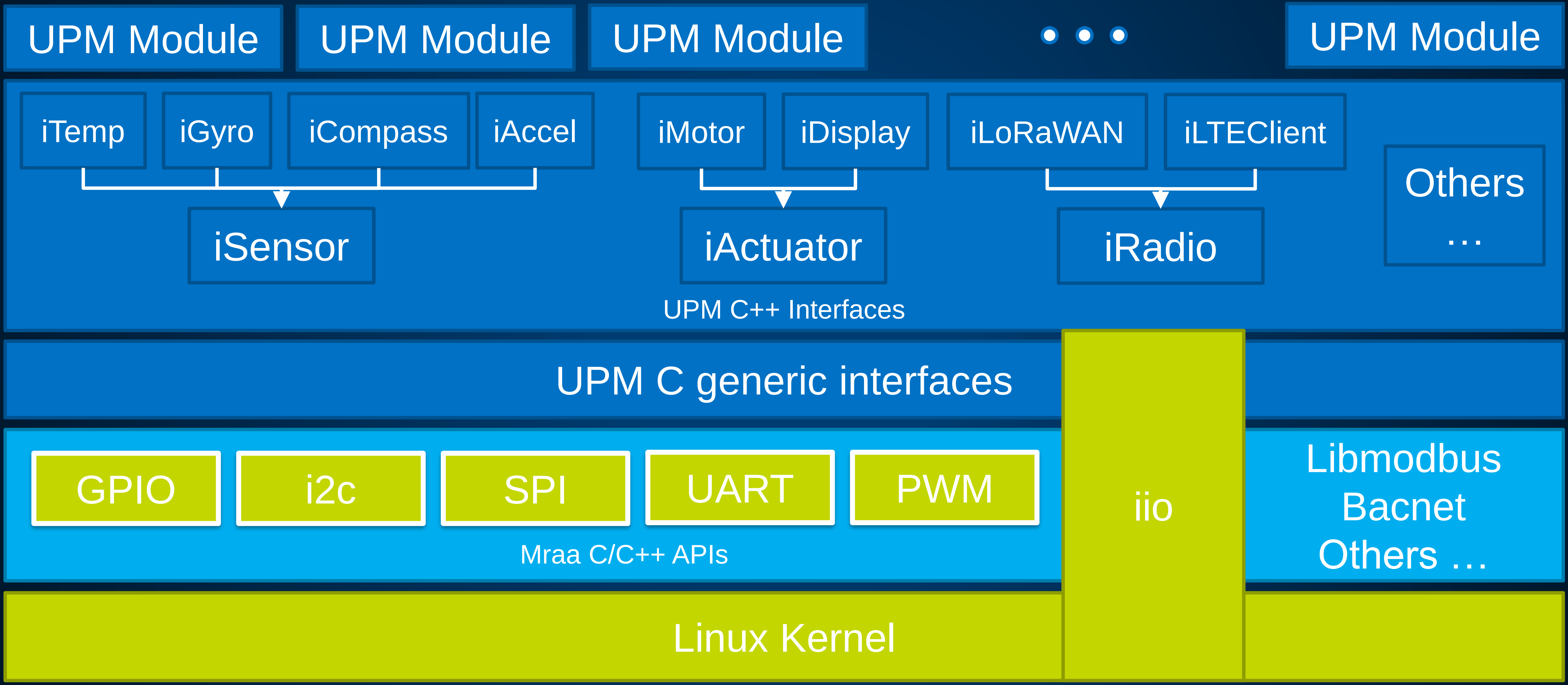
Prototype Sensors

DFROBOT DRIVE THE FUTURE · seeed Grow the Difference · adafruit · sparkfun ELECTRONICS

Industrial Sensors

AEOTEC BY AEON LABS · BOSCH · ™ · OMEGA® · OMET · ST · Honeywell

MRAA & UPM – Architecture



UPM Module

UPM Module

UPM Module

• • •

UPM Module

iTemp

iGyro

iCompass

iAccel

iSensor

iMotor

iDisplay

iActuator

iLoRaWAN

iLTEClient

iRadio

Others …

UPM C++ Interfaces

UPM C generic interfaces

GPIO

i2c

SPI

UART

PWM

Mraa C/C++ APIs

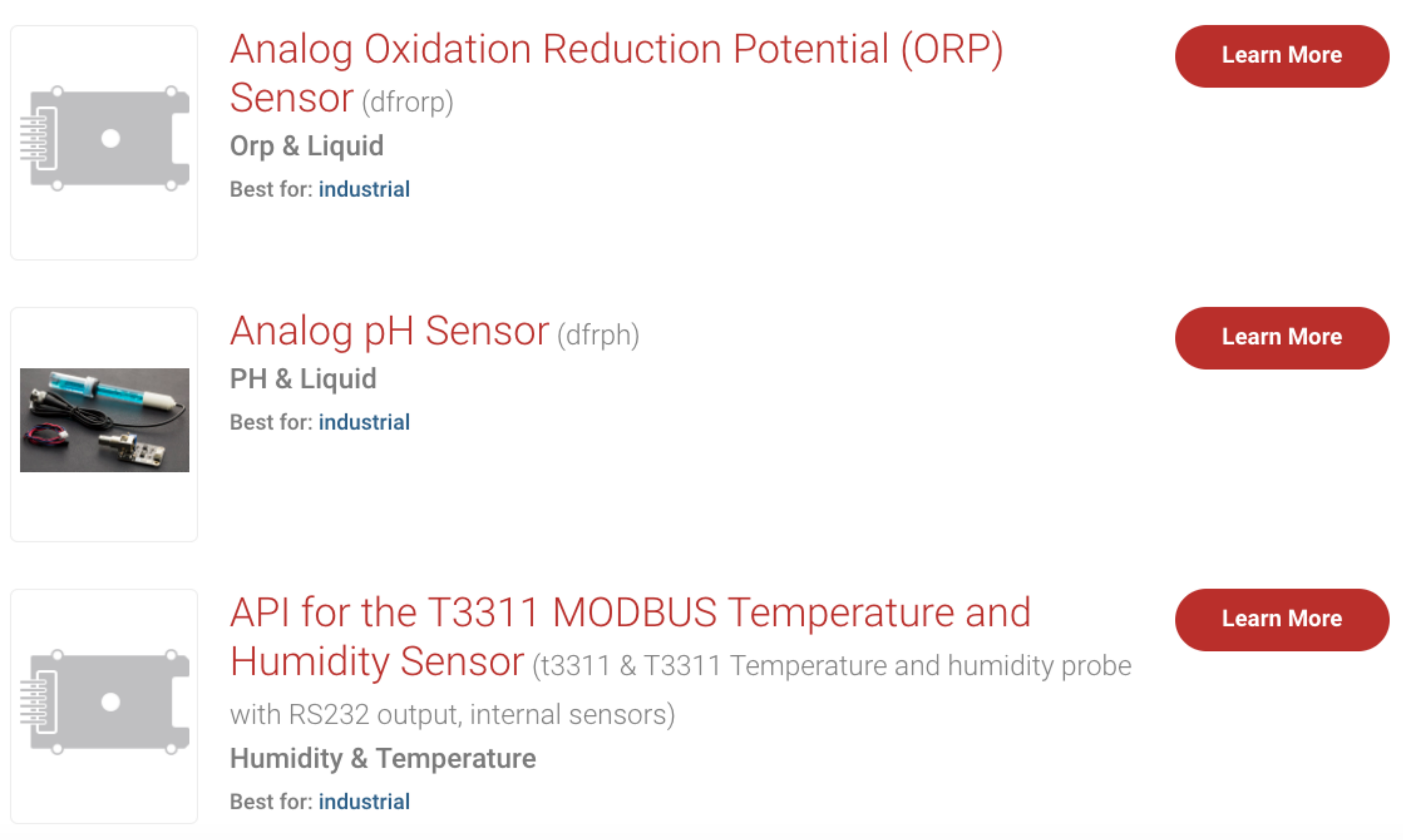
iio

Libmodbus
Bacnet
Others …

Linux Kernel

Using MRAA with UPM

https://upm.mraa.io



Analog Oxidation Reduction Potential (ORP) Sensor (dfrorp)

Orp & Liquid

Best for: industrial

Analog pH Sensor (dfrph)

PH & Liquid

Best for: industrial

API for the T3311 MODBUS Temperature and Humidity Sensor (t3311 & T3311 Temperature and humidity probe with RS232 output, internal sensors)

Humidity & Temperature

Best for: industrial

Node.js Code for Temperature Sensor

```
// Load Grove module
var groveSensor = require('jsupm_grove');

// Create the temperature sensor object using AIO pin 0
var temp = new groveSensor.GroveTemp(0);
console.log(temp.name());

// Read the temperature ten times, printing both the Celsius and
// equivalent Fahrenheit temperature, waiting one second between readings
var i = 0; var waiting = setInterval(function() {
      var celsius = temp.value();
      var fahrenheit = celsius * 9.0/5.0 + 32.0;
      console.log(celsius + " degrees Celsius, " );
      console.log(Math.round(fahrenheit) + " degrees Fahrenheit, " );
      i++;
      if (i == 10) clearInterval(waiting);
}, 1000);
```

Unified API architecture: EdgeX Foundry

Edgex Foudnry

- Industrial IoT Edge / Fog Architecture
- Micro-Service Based
- Docker Based
- REST API - Consul
- JSON Device & Data Definitions
- Platform Agnostic
- Open Source – Linux Foundation

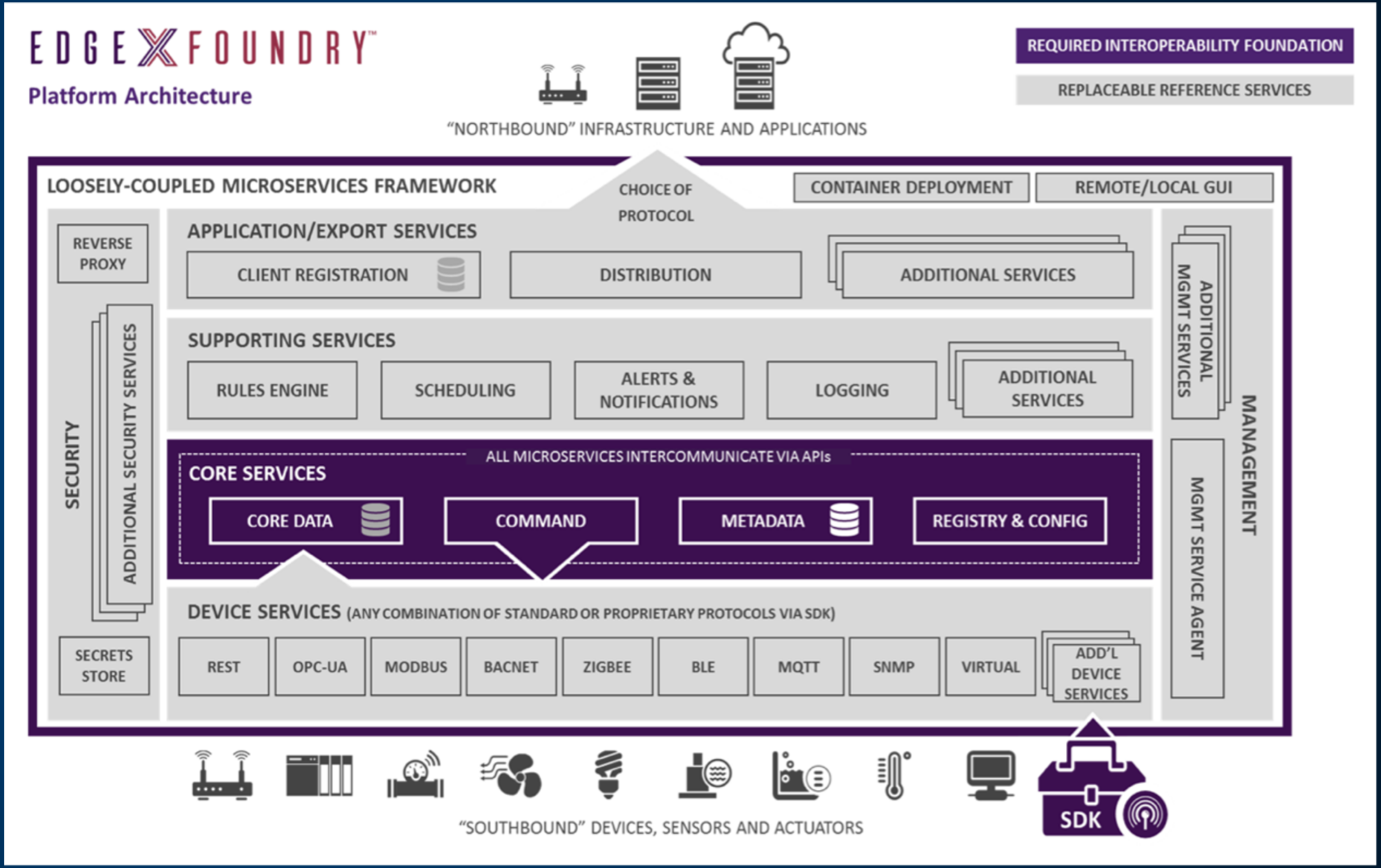
```
docker-compose pull


POST to
http://localhost:48080/api/v1/event


BODY:
{"device":"countcamera1","readings":[{"name":"humancount","value":"5"}…
```

Unified APIs

- Loosely-coupled microservices bound by **common APIs** established through vendor - neutral collaboration in Linux Foundation

- **Polyglot**: microservices can be written in any programing language (e.g. Java, Python, Go Lang, C) and deployed in containers or VMs

- **Modular:** Once key APIs are established, entire subsections can be replaced, combined, etc. with proprietary, differentiated "EdgeX-compliant" offerings, even Core Services



EDGE X FOUNDRY™

Platform Architecture

REQUIRED INTEROPERABILITY FOUNDATION

REPLACEABLE REFERENCE SERVICES

"NORTHBOUND" INFRASTRUCTURE AND APPLICATIONS

LOOSELY-COUPLED MICROSERVICES FRAMEWORK

CHOICE OF PROTOCOL

CONTAINER DEPLOYMENT

REMOTE/LOCAL GUI

REVERSE PROXY

SECURITY

ADDITIONAL SECURITY SERVICES

SECRETS STORE

APPLICATION/EXPORT SERVICES

CLIENT REGISTRATION

DISTRIBUTION

ADDITIONAL SERVICES

SUPPORTING SERVICES

RULES ENGINE

SCHEDULING

ALERTS & NOTIFICATIONS

LOGGING

ADDITIONAL SERVICES

ALL MICROSERVICES INTERCOMMUNICATE VIA APIs

CORE SERVICES

CORE DATA

COMMAND

METADATA

REGISTRY & CONFIG

DEVICE SERVICES (ANY COMBINATION OF STANDARD OR PROPRIETARY PROTOCOLS VIA SDK)

REST

OPC-UA

MODBUS

BACNET

ZIGBEE

BLE

MQTT

SNMP

VIRTUAL

ADD'L DEVICE SERVICES

ADDITIONAL MGMT SERVICES

MGMT SERVICE AGENT

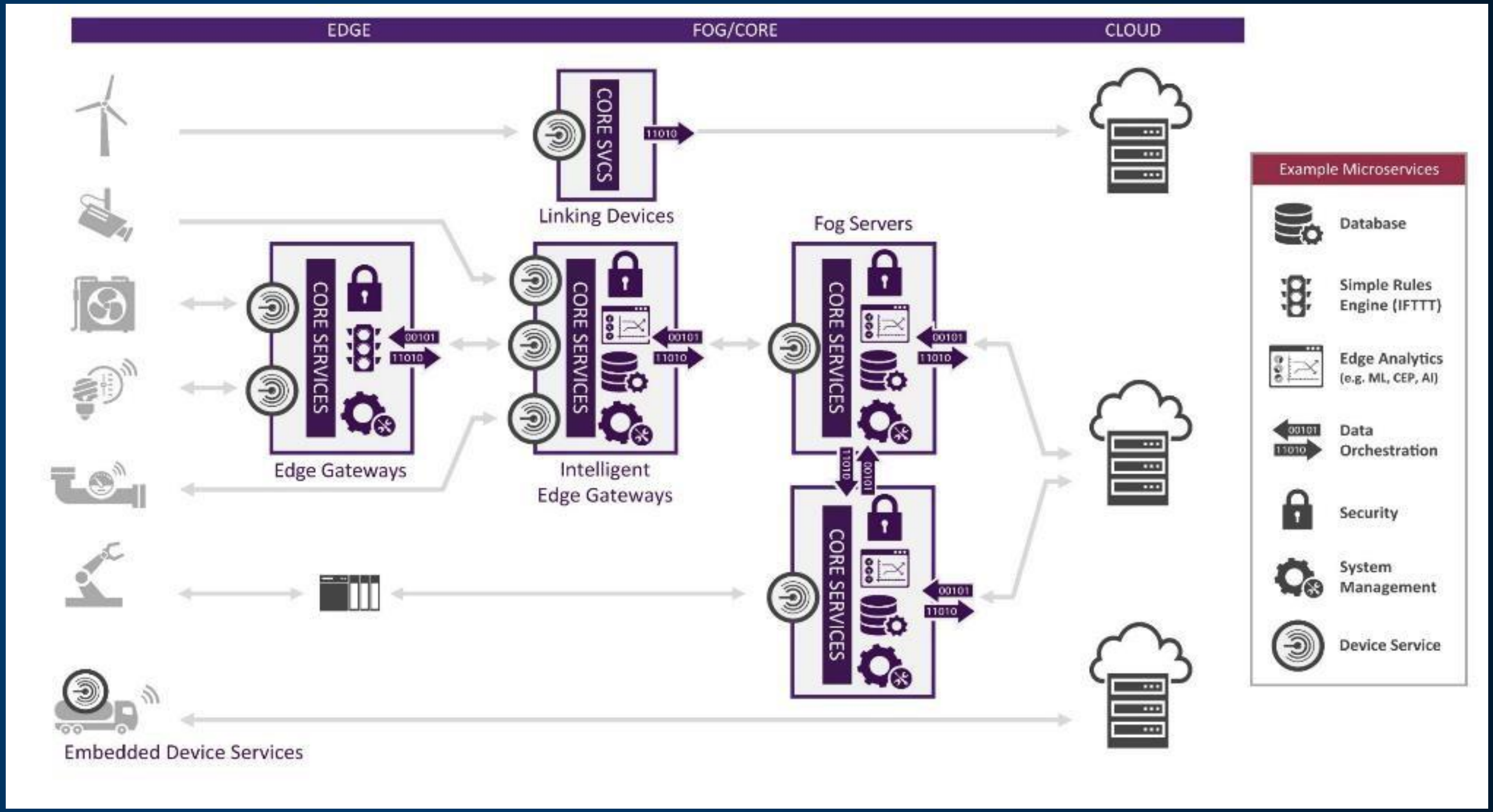
MANAGEMENT

SDK

"SOUTHBOUND" DEVICES, SENSORS AND ACTUATORS

Cloud Scalability

- **Highly scalable:** microservices can be scaled up or down depending on need
- **Distributed:** Entire platform can run on one node, such as a gateway, or be distributed across many nodes
- **Cross Compatible:** Restful API ensure compatibility across most cloud services or existing on prem deployments.



Core Services

Configuration and Registry microservice

- Provides centralized management
- Configuration and operating parameters for all microservices
- Location and status of micro services

Metadata

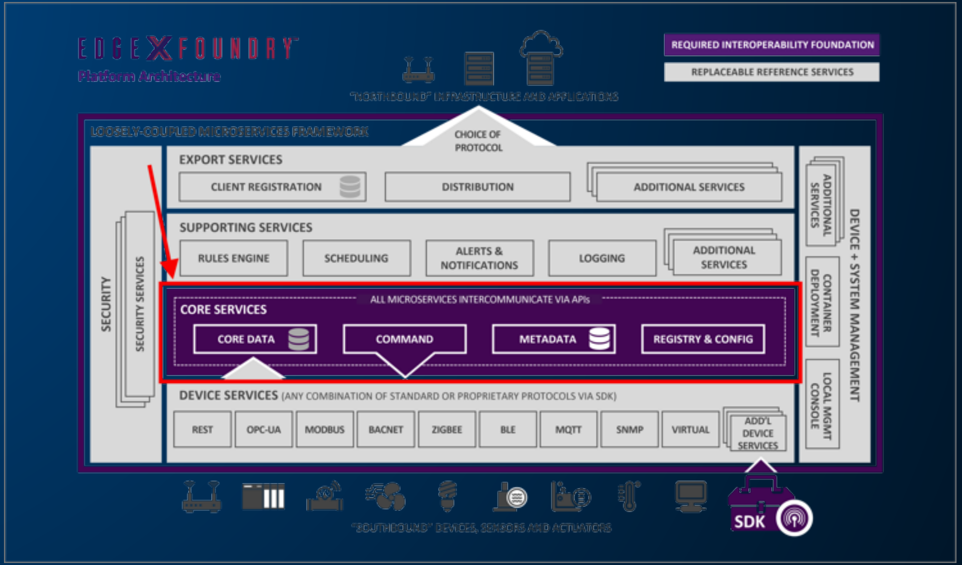
- Manages information about the devices
- Knows the type, and organization of data reported by the devices and sensors
- Knows how to command the devices and sensors

Command

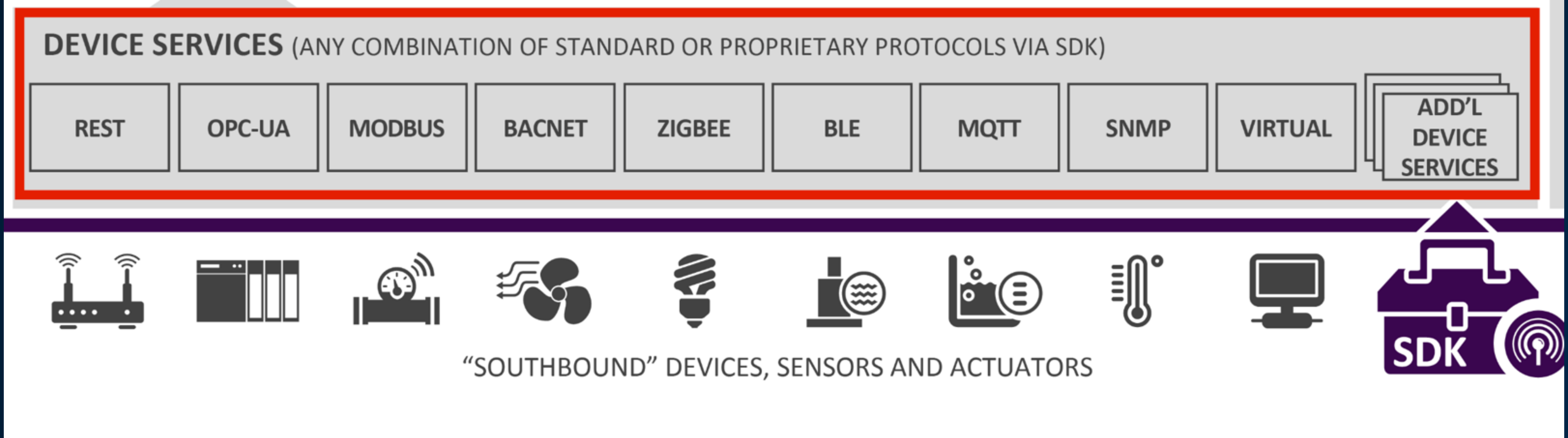
- Enables the issuance of commands or actions to devices
- Translator of command or action requests from rules engine or export services

Core Data

- Centralized persistence facility for data readings
- Store the device and sensor data on the edge system
- Uses a REST API for moving data into and out of the local storage



Device Services


DEVICE SERVICES (ANY COMBINATION OF STANDARD OR PROPRIETARY PROTOCOLS VIA SDK)
REST
OPC-UA
MODBUS
BACNET
ZIGBEE
BLE
MQTT
SNMP
VIRTUAL
ADD'L DEVICE SERVICES
"SOUTHBOUND" DEVICES, SENSORS AND ACTUATORS
SDK

- Communicate with the devices, sensors, actuators, and other IoT objects through protocols native to the IoT object.
- Converts the data produced and communicated by the IoT object, into a common EdgeX Foundry data structure
- Sends that converted data into the Core Services layer, and to other microservices in other layers of EdgeX Foundry.
- Provisioning and Adding a device is through different Rest API calls.

- To add new devices
- It is written in JAVA.
- Eclipse IDE is supported.
- Go and C Language support as device sdk is work in progress.
- Other languages will be available in future.

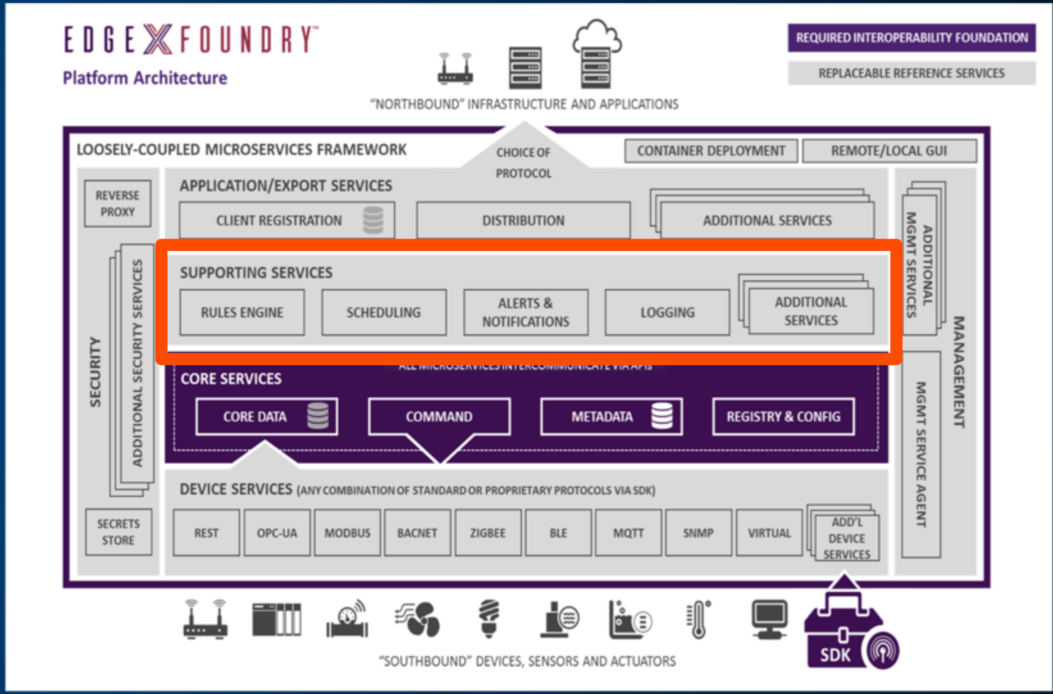
Supporting Services

Alerts and Notification – Notifies another system or person of something discovered on the Node by another microservice.

Logging – Monitor and understand the tasks carried by system and the communication between microservices.

Scheduling – Cleans up the event and reading the data exported to Gateway.

Rules Engine – Provides Edge event triggering mechanism.



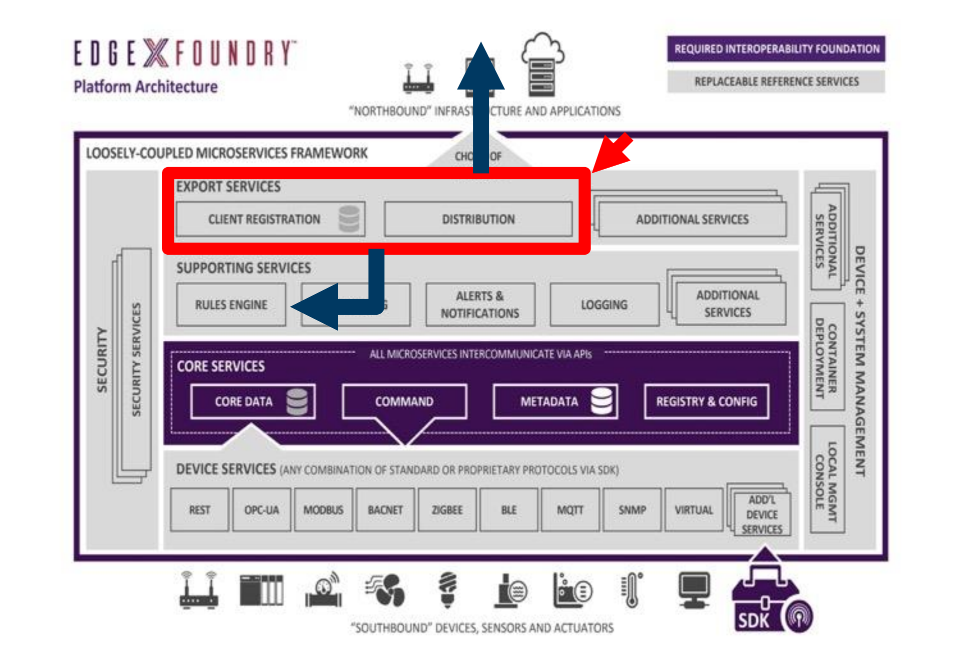
Export Services

Export Services = Export Client microservice + Export Distribution microservice

- Provides ability to get EdgeX sensor/device data to other external systems or other EdgeX services
- External systems like Azure IoT Hub, Google IoT Cloud, etc.
- Other EdgeX services include the Rules Engine microservice or other “analytics” systems/agents in the future

- **Export Client** – allows for internal or external clients to
  - Register for sensor/device data of interest
  - Specify the way they want it delivered (format, filters, endpoint of delivery, etc.)

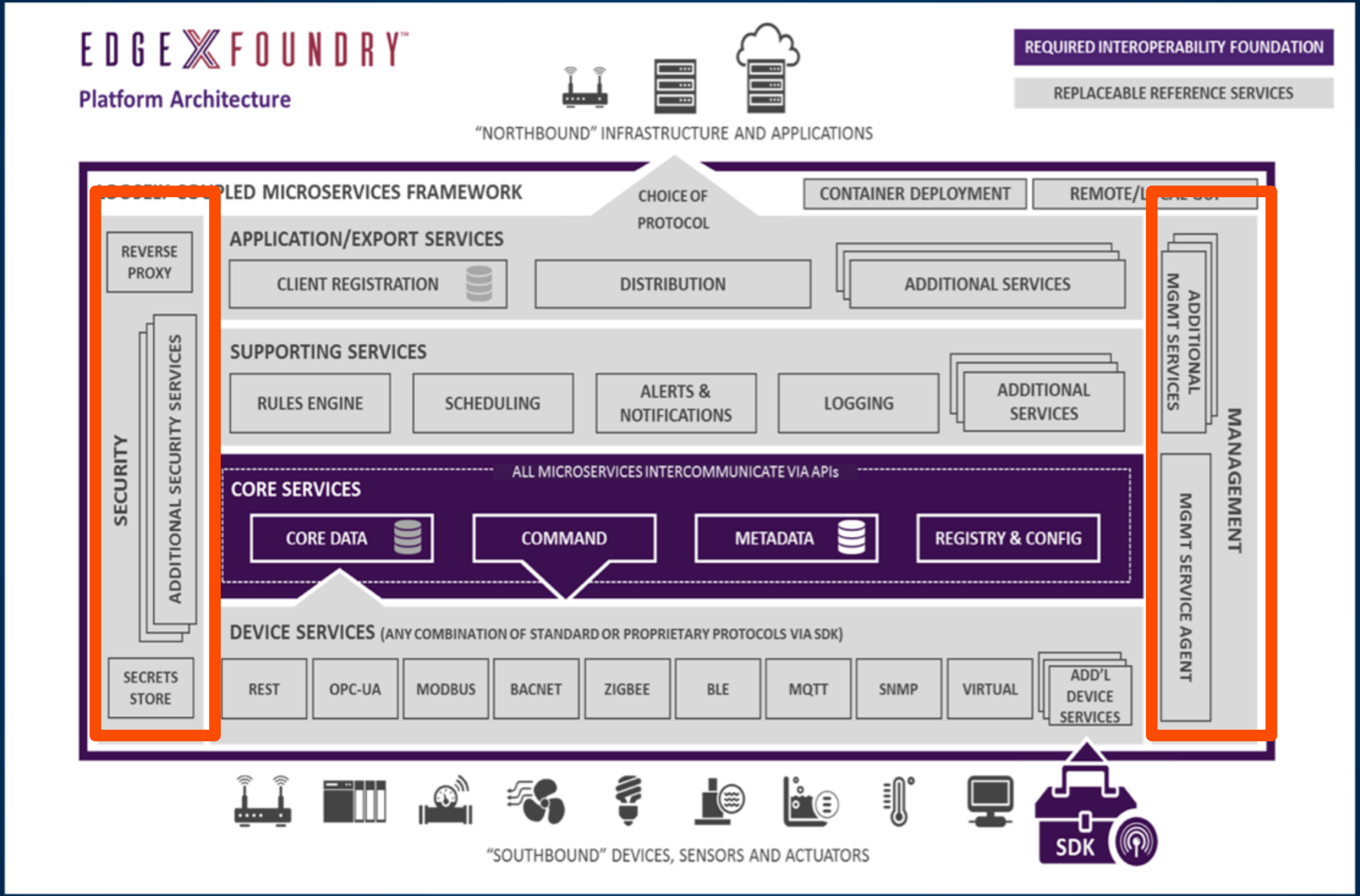
- **Export Distribution** – performs the act of delivering the data to registered clients
  - Receives all the sensor/device data from Core Data
  - Performs the necessary transformations, filters, etc. on the data before sending it to the registered client endpoints



Management and Security

Security

Security elements both inside and outside of EdgeX Foundry protect the data and command of devices, sensors, and other IoT objects managed by EdgeX Foundry.



System Management

- Start, stop, restart the micro services
- Obtain various metrics so that the operation and performance of the services can be monitored
- Receive the EdgeX micro service configuration