Common Industrial Cloud Interface – Uses Cases and Technical Requirements for Data Transfer

Stephen C. Briant
Rockwell Automation

Tom Whitehill
Schneider Electric

February 22, 2017
Agenda

- Introduction
- Reference Architecture
- Cloud Technologies
- Guiding Principles
- Information Exchange Patterns
- Use Cases
- Proof-Of-Concept
- Conclusion – Next Steps
Introduction

- The Industrial Internet of Things is here, bringing new technologies, challenges and opportunities to industrial automation
- Companies are looking to the internet and cloud computing to provide new ways to improve operations and increase productivity as well as addressing concerns of an aging workforces
- Cloud computing offers many advantages that were previously unavailable, starting with the ability to connect to devices across an enterprise or a machine type across multiple enterprises. In addition, the ability to scale computing power and storage are enabling new possibilities for analyzing data streams.
- Acquiring data from devices is the primary focus in the market today, but there are definitely opportunities to do more
- In April, 2016, ODVA announced the formation of a new Special Interest Group (SIG) for the Common Industrial Cloud Interface (CICI) to address these opportunities.
- This new SIG intends to leverage technologies available in cloud platforms and “connect” them with the rich information defined in CIP Devices in a simple and secure manner.
Cloud Technologies

- **AMQP**
  Advanced Message Queuing Protocol, is the open standard and has emerged as a very popular protocol for sending messages to and receiving messages from Cloud-based systems. In addition to being Open and Standard, AMQP was designed with these characteristics Security, Reliability, Interoperability.

- **MQTT**
  Message Queue Telemetry Transport, is an ISO standard (ISO/IEC PRF 200922), publish/subscribe, lightweight messaging protocol used for Cloud-connectivity for limited network bandwidth, remote applications.

- **JSON**
  JavaScript Object Notation, is a terse, readable, structured data format. It is very popular as a payload format for Device-to-Cloud and Cloud-to-Device messaging. A benefit to using JSON is that many stream processing applications are built to natively consume JSON structures efficiently and cost-effectively. Below is a very basic JSON message:

```json
{
    "name"="CICI",
    "message"="Hello World!"
}
```
• There are fundamental differences in Cloud-based vs CIP-network based application development … a few are summarized here:
  – Distribution: Cloud-based applications are intrinsically distributed, combining resources from multiple compute, storage and service platforms to achieve function.
  – Real-Time: due to its distributed nature and platform dependencies, the notion of real-time is an uncommon concept in public Cloud computing.
  – Protocols and Payloads: Message payloads that are easily programmatically digestible and extensible
  – Cloud communication protocols are generally widely used, open standards which are not industry specific and which may be replaced at any time
  – CIP communications, while an open standard, is industry specific and slow to change.
  – Application scope and lifecycle: Cloud-based applications can have very broad scope, virtually no limitations and can be modified or updated very rapidly
Guiding Principles

• Security
  – all communication must be initiated from the device or gateway.

• Performance
  – Communication must be performant and scalable across the Reference Architecture, therefore CIP must “stay home” or stay on premise

• Four V’s of Big Data
  – Volume (must handle the scale of data)
  – Variety (must handle different forms of data)
  – Velocity (must handle the speed required for analysis of the data)
  – Veracity (must handle the uncertainty or quality of the data)
Information Exchange Patterns

- **Telemetry**
  - Device > Cloud
  - One Way

- **Inquiry**
  - Device > Cloud w/Response
  - Two Way

- **Notification**
  - Cloud > Device
  - One Way

- **Command**
  - Cloud > Device w/Response
  - Two Way
Use Cases – Device Lifecycle

• Commissioning
  – Out-of-the-box definitions (Inquiry, Command)
  – Cloud Registration / backend business setup (Inquiry, Command)
  – On-boarding/Provisioning (Inquiry, Command)
  – Context of device in application (Inquiry, Command)
  – Control/Application loading (Inquiry, Notification, Command)

• Operating
  – Monitoring (Telemetry)
  – Maintenance (Telemetry, Command)
  – Calibration (Inquiry, Command)
  – Diagnosis (Telemetry)
  – Enable/Disable (Command)
  – Optimization / Changing Parameters / Programs (Telemetry, Command)
  – Software updates (Inquiry, Notification, Command)
  – Device Replacement (Inquiry, Command)

• Decommissioning
  – Removing a device (Telemetry, Command)
Proof-Of-Concept

- **Telemetry Use Case**
  - A cloud application needs a list of CIP Devices
  - A cloud application needs a list of product names of CIP Devices
  - A cloud application needs the firmware version of a CIP Devices

- **Goal**
  - Set up a simple application that accomplishes the Telemetry use case
  - Leverage open standards were possible
  - Leverage existing code and content to expedite efforts

- **Results**
  - Used a prototype CIP Stack and CIP Scanner in Node.JS
  - Used an open source OPC UA Server in Node.JS
  - Used Microsoft’s open source C# application to read OPC UA Server, connect to Azure IoT Hub via AMQP
  - Defined an Azure IoT Device on Azure IoT Hub
  - Used Device explorer on Azure to view AMQP message stream
Reference Architecture: Proof-Of-Concept

Data Flow:
- Protocol = AMQP
- Gateway Publishes Telemetry Identity Object

Example:
- Protocol = AMQP
- Gateway Publishes Telemetry Identity Object
CIP Stack/CIP Scanner

- Load application into Node.JS

```javascript
C:\temp\OPCUA\node-opcua\node MyServer.js
initialized
{name:"cip_enip_udp", "hostname":"NAUSSEW6MXRG12", "pid":3000, "level":50, "msg":"Socket failed. Error: bind EADDRINUSE"}, "time":"2017-02-10T08:38:36.033Z", "t":0}
Server is now listening...
press CTRL+C to stop
disc: []
```

- Print output of discovered CIP Devices are loaded into OPC UA Server

```json
[ { "identity": { vendor_id: 1, device_type: 12, product_code: 166, major_revision: 10, minor_revision: 6, status: 46, serial_number: "10896684", product_name: "1756-EN2T/DC", ip_address: "10.88.53.1" } },
 { "identity": { vendor_id: 1, device_type: 14, product_code: 190, major_revision: 29, minor_revision: 11, status: 1034, serial_number: "1263990", product_name: "1756-L85E/8", ip_address: "10.88.53.3" } },
 { "identity": { vendor_id: 1, device_type: 14, product_code: 2192, major_revision: 12, minor_revision: 2, status: 97, serial_number: "54102646", product_name: "PowerFlex 755", ip_address: "10.88.53.9" } } ]
```
Reference Architecture: Proof-Of-Concept

Example:
- Protocol = AMQP
- Gateway Publishes Telemetry Identity Object

Device 1
Device 2
... 
Device n

On Premise

CICI Gateway

CIP Interface + CIP Identity Reader
OPC UA Server + OPC UA Client to AMQP

Data Flow

Azure Cloud

Ingress Queue

Backend Cloud Applications and Presentation

Inspection Here

CICI Scope Inspection

CIP Network

Device 2

Device n

CICI Scope

Azure IoT Hub AMQP Endpoint

CIP Network

Device 1
Setup OPC UA Client to OPC UA Server

- OPC UA Server address
- Node to start data transfer
Set up OPC UA Client to Azure IoT Hub

- Set up path to Azure
- Name of endpoint device
- Name for event hub
- Selection of transport
Proof-Of-Concept

Reference Architecture: Proof-Of-Concept

Data Flow:
- Example: Protocol = AMQP
  - Gateway Publishes Telemetry Identity Object

On Premise
- CIP Network
  - Device 1
  - Device 2
  - ... Device n

CICI Scope
- CICI Gateway
- CIP Interface + CIP Identity Reader
- OPC UA Server + OPC UA Client to AMQP

Azure Cloud
- Azure IoT Hub AMQP Endpoint
- Ingress Queue
- Inspection Here

Backend Cloud Applications and Presentation
Observe messages on Azure IoT Hub
Conclusion – Next Steps

• The Common Industrial Cloud Interface SIG was formed to enable cloud applications to have access to the valuable information available in CIP Devices
• There are many opportunities or use cases; however, the most obvious set of use cases are grouped around managing the lifecycle of CIP devices
• The Common Industrial Cloud Interface SIG will use guiding principles and Information Exchange Patterns to flush out functionality needed for these use cases
• The next step will be map needed functionality to available technologies and standards, including those shown in the Proof-Of-Concept
• If you would like more information or want to contribute, please consider joining the Common Industrial Cloud Interface SIG!