Available online at www.sciencedirect.com

Manufacturing Letters

Manufacturing Letters 33 (2022) 66–73

50th SME North American Manufacturing Research Conference (NAMRC 50, 2022)

Streaming Machine Generated Data via the MQTT Sparkplug B Protocol for Smart Factory Operations

Pavel Koprov[a], Ashwin Ramachandran[a], Yuan-Shin Lee[a], Paul Cohen[a], Binil Starly[a,*]

[a]*Edward P. Fitts Department of Industrial and Systems Engineering, North Carolina State University, Raleigh, NC 27695, USA*

* Corresponding author. Tel.: +919-515-1815. *E-mail address:* bstarly@ncsu.edu

Abstract

The implementation of smart manufacturing relies on back-and-forth industrial communications between the machine assets on the shop floor to the cloud-level information systems. The Purdue Model of Computer Integrated Manufacturing has been a standard for industrial control system architecture guidance. However, in IIoT, data flow is not hierarchical since intelligence is added to the machine asset level. This paper describes the implementation and test run of the MQTT Sparkplug-B protocol with conventional machining assets, with its data being streamed from the machine controllers to a shop floor network and further on to a cloud-based platform. The publish/subscribe pattern of communication between the machining asset, and the cloud level is established, further demonstrating the ease with which a unified namespace can be achieved between various information and Operational Technology networks seamlessly and with relative ease.

Keywords: Digital Factory; IIoT; Industrial Internet of Things; Digital Twin; Legacy Machines; IT/OT integration

1. Introduction

The Industrial Internet of Things (IIoT) can be described as an interconnection of various sensors, actuators, devices, computers, and machines that communicate, share data, and collaborate [1]. Data is envisioned to make physical machine assets on the production floor intelligent and self-aware, leading to the realization of Smart Factories. Factories that leverage data as an asset through flexible communication between machine assets on the floor and higher order information technology (IT) systems enable the factory to be more agile and responsive to production needs. Machine assets on the production floor can send a prodigious amount of data that can be used for many useful applications from machine monitoring, continuous control, real-time visibility, and machine-machine coordination. IIoT enables the implementation of artificial intelligence (AI) based solutions on machine assets, mobile robots, inspection cameras, material handling robots, etc. [2]. IIoT enables Smart manufacturing and is bolstered by several other technologies such as 5G, RFID (radio frequency identification), cloud and distributed computing, and many others. IIoT enabled machine asset devices conventionally do not have enough computing power [3] and must often share the data to other devices with a specific purpose such as Supervisory Control and Data Acquisition (SCADA), Historians, or Analytical systems for Machine Learning (ML) application. The machine assets must have a high data transfer rate with very low overhead. It is easy to imagine how congested the network can be if the factory floor that consists of thousands of devices begins transmitting data from the Operational Technology (OT) side of the factory over to the Information Technology (IT) systems that power the factory floor enterprise operations.

Conventional methods of industrial communication such as OPC-UA (Object Linking and Embedding for Process Control - Unified Architecture) mostly use a poll/response approach, which may not always be reliable, particularly in scenarios when poor network connectivity through wireless connections and a non-deterministic number of device assets come on and off the network [4]. On the other hand, publish/subscribe (pub/sub) type messaging architecture can alleviate the concerns of a poll-response approach by decentralizing data flow. In MQTT, every machine asset can talk to any other machine asset within the factory floor network via a broker. Recently, the OPC Foundation has released a pub/sub model specification to implement the same data structure through other application layer protocols such as Advanced Message Queuing Protocol (AMQP) or Message Queuing Telemetry Transport (MQTT) [5]. AMQP is heavier and more information rich than MQTT, with benefits and drawbacks (AMQP has its specification and can send bulk messages of indefinite size, while the overhead of a message is much greater) [6]. MQTT was designed for industrial devices with very small computing capabilities and inconsistent network connections. It allows a theoretically indefinite number of devices to exchange information without having a stable connection.

In addition, MQTT consumes traffic only when needed (Report-by-Exception) and does require only regular pinging (default is once per minute). A typical MQTT network includes at least two nodes: a client and a broker. A broker is a de-facto server whose role is to retrieve messages published by clients, store for a short time, queue and forward them to the clients who are subscribed to the specific topics. The header of the MQTT message is only 2 bytes. IIoT enabled by MQTT utilizes the pub/sub architecture that secures devices from accessing through network ports. Any authenticated device can subscribe and publish to a topic. Topics themselves maintain a hierarchy that organizes data payload under a hierarchical tree structure. Due to the pub/sub patterns of communications, smarter manufacturing operational services can be delivered over the cloud, enabling more effective management of manufacturing resources for shop floor management. Manufacturing Execution Systems or Manufacturing Operations Management, as defined by the ISA-95 standards, enable its users to track asset utilization, product recipe, statistical process control with data synchronized from the plant floor, through business systems to the executive management. The pub/sub pattern enables a unified namespace of operations that connects various disparate IT systems and OT systems, thereby easing the path to interoperability concerns while maintaining distinct separation of concerns between each IT system.

The MQTT protocol however does not define a standard for how the body of the message is structured, the type of meta-data to be transferred, nor does it define the topic level hierarchy that delineates the organization of data. A recently released version of the MQTT Sparkplug B (SpB) specification addresses some of these concerns, particularly amenable IIoT-based operations [7]. In comparison, the OPC-UA pub/sub model has a much more complicated specification of the message and topics structure than the SpB (on average 25 times heavier), as it is still a client/server specification legacy [5], [8]. Since the memory overhead on OPC-UA is quite a bit higher than MQTT, the power consumption on the edge device is consequently higher. In addition, OPC-UA is open only for the members of the OPC foundation. Combined with industry communication standards such as MTConnect, MQTT SpB can enable the smoother transition and implementation of IIoT within the factory flow without a restrictive licensing model. Once the architecture is laid out in the factory, this data can be permitted to flow out of the factory floor to power cloud-based applications. These cloud-based apps can be extremely useful to small and medium-scale manufacturers who can rapidly scale the implementation of Smart Manufacturing applications through profile reuse and synergies in laying down a common platform to integrate various apps through standard interfaces.

In collaboration with industrial partners, the Clean Energy Smart Manufacturing Innovation Institute (CESMII) has created the Smart Manufacturing Innovation Platform (SMIP) to enable frictionless movement of information - raw and contextualized data - between real-time operations, people, and the systems that create value across manufacturing processes. The Smart Manufacturing Innovation Platform is an interoperability solution to deliver Industrial 'Plug and Play' applicability to discrete, hybrid, and process-oriented industries. The platform provides secure connectivity to the manufacturer's equipment and processes and adds valuable context so that applications can access information intelligently and through automated mechanisms.

The key technology in the platform is the concept of Profiles, that CESMII is creating to contextually describe sensors, equipment, and processes and generate semantics for data, showing how variables relate to each other. Combining these new concepts in context will provide the ability to define new systems without the need for extensive middleware configuration or infrastructure maintenance and management [9]. The SMIP is a collection of technologies for simplifying access to manufacturing data by normalizing across protocols, enforcing a reusable object model, and guaranteeing an interface contract for application development.

2. Project Goal

The main goal of this project is to demonstrate the ability to stream data from machine assets in a unified namespace, particularly when hundreds of machine assets are connected on a network within a digital factory. This work showcases how the MQTT Sparkplug™ B specification can enable a unified namespace that allows IT/OT technology integration while still enabling interoperability, transparency, and operations security. Real-time data streaming from the machine is stored in local databases and then streamed to the CESMII SMIP, a cloud-based solution that can plug into an extensible collection of apps. The two objectives of this project are as follows:

- The addition of low-cost systems-on-chip devices, such as the Raspberry PI device, acts as an edge node that publishes data reported by the computer numerical control (CNC) unit. The objective is to demonstrate a system architecture that connects machine assets to a factory-wide network of connected machine assets while utilizing available standards such as the MTConnect protocol to structure data obtained from CNC machines semantically.

- Demonstrate how data is streamed locally to data historians directly connected to the factory network and utilize available protocols to stream data to the cloud. We have showcased this project via a local data dashboard that updates its values based on streaming data from the asset. The second scenario demonstrates how data is streamed to the cloud platform for verification and analysis. As data is streamed to the cloud, a semantic layer maps data from the asset (CNC machines) to specific machine profiles based on the MTConnect standard.

This project aims to study the use of new information architecture to ease the integration of IT/OT technologies for Industry 4.0 initiatives. As data is more prevalent and AI/ML applications run on the cloud, this enables scenarios where ML models directly update models embedded within the machine assets on the factory floor. In such cases these machine assets must be in constant communication with cloud-based systems for analytics and intelligent predictive control. Current Factory floor information architectures do not necessarily ease the communication from assets on the factory floor to cloud based systems.

3. Literature Review

Most modern manufacturing processes follow the Reference Model for Computer Integrated Manufacturing (CIM), also known as the "Purdue model" [10]. This framework was proposed in the early 90s and has been de-facto a standard for IT/OT systems implemented for the CIM and led to the development of the ISA-95 standard [11]. The IIoT architecture does not fit this model, especially with Big Data and ML applications. The Purdue model prescribes five layers that separate OT from IT. Levels from 0 to 3 are located on the factory floor and are not connected to the internet. The demilitarized zone, or level 3.5, performs the cybersecurity function and air-gaps the OT from IT. Enterprise levels (4-5) are retrieving only siloed data and cannot connect directly to assets on the factory floor. A five-layered model limits data flow in the context of Smart Manufacturing and specifically in IT/OT integration in an Industrial IoT setting. For example, machine learning models are constantly being updated in the cloud. Device assets on the factory floor would need to access the most recently updated model. Simultaneously, device assets may also need to share specific data generated during operations to update the ML model. The Purdue reference model also does not allow quick equipment addition and operation. This model does not fit the IIoT paradigm since device assets on the shop floor will need to access cloud-level resources. MQTT SpB can be applied to the existing Purdue model and allow access to the data from the IT level and send additional data from ML predictive systems to the OT level.

One good-known way of accessing the data of metal cutting machine tools is via MTConnect. It is a (read-only) royalty-free communication protocol that uses XML and HTTP to request and deliver data from MTConnect-enabled devices [12]. The benefit of this protocol is that it does not require any proprietary software or deep knowledge to access the information from the equipment. One needs to open the browser, type the IP address and the port of the polled asset, followed by one of the four possible requests: probe, current, sample, or asset. The biggest drawback is that someone needs to create MTConnect Agent and Adapters, which most likely will need to be performed by the original equipment manufacturers (OEM). Ben Edrington et al. [13] implemented the machine monitoring system based on MTConnect and DMG MORI SEIKI assets. Their case utilizes Mori Net and Messenger software developed by DMG MORI and uses the MTConnect protocol. Unfortunately, not all equipment has built-in MTConnect agents, and even those with it might not give all the information needed (Haas MTConnect [14]).

Object Linking and Embedding for Process Control - Unified Architecture (OPC-UA) is a machine-to-machine protocol that the OPC foundation designed specifically for industrial applications [15], [16]. Unlike MQTT, OPC-UA uses server/client architecture to poll the client to get the data [17]. Every device type has its proprietary specification that must be followed to work correctly with the OPC-UA server. Classic OT/IT infrastructure includes various devices connected to the OPC-UA servers that communicate with IT structures. All OT participants can also communicate through OPC-UA servers [16]. The data types that can be transferred through those protocols are also predefined by specifications and must completely correspond to the ISA-95 standard [18]. It was shown that there is an issue when one is trying to communicate between different OPC-UA servers or even non-OPC servers [19]. There are several attempts to simplify the use of UA without specific proprietary software [20], [21], but they require substantially more computational power than the MQTT protocol.

Due to the specific needs of the IIoT architecture, the Eclipse foundation developed a special specification called SparkPlug™ [7]. This document specifies the structure of topics, message payload and communication parameters. The generic topic looks as follows:

namespace/group_id/meassage_type/edge_node_id/[device_id]

A namespace is either "spAv1.0" or "spBv1.0", depending on which specification version is implemented. Group_id can be any UTF-8 alphanumeric string. Message_type indicates how to handle the message payload. Below are listed possible message types:

- NBIRTH – Birth Certificate for MQTT Edge of Nodes
- NDEATH – Death certificate for MQTT EoN.
- DBIRTH – Birth certificate for Devices.
- DDEATH – Death certificate for Devices.
- NDATA – Node data message.
- DDATA – Device data message.
- NCMD – Node command message.
- DCMD – Device command message.
- STATE – Critical application state message.

Some of the devices available at the production floor exploit proprietary protocols or even send analog signals. This necessitates the need for EoN (edge of nodes) – devices that convert proprietary protocol data and analog signals to MQTT Sparkplug data type. Legacy machines that do not communicate through modern protocols can be enabled using EoN devices. On the other hand, those devices labeled as Sparkplug-enabled can publish directly to the device_id topic. SpB specification is critical for IIoT as it defines standardized topic namespace and payload definition. It enables auto-discovery of devices and continuous session awareness. New devices will be easy to find in the network, and their payload structure is deterministic.

4. Methods

4.1. System Architecture of Pub/Sub Model within Factory

A high-level system diagram demonstrating the concept of pub/sub system architecture is shown in Figure 1. Various machine assets on the shop floor can publish data to an MQTT broker via a topic hierarchy specified by the SpB specification. Different subscriber node types listening to specific topic namespaces pick up the messages and process them as needed. For example, a subscriber listening to messages published by Node N1 is processed and inserted into a plant historian for storage and archival. In another example, subscriber Node S3 might receive a notification that the robot has stopped operation due to a malfunction. Once data enters a historian, this data can be distributed to other client applications, such as visualization dashboards, performing data analysis, etc. The subscriber nodes themselves are not directly communicating with the publisher nodes. The communication happens only through the MQTT Broker topic channels. If industry standards are available, such as the MTConnect standard for CNC machines, this standard can structure the data sent by the publisher. Devices that might be part of a production cell can individually communicate with the broker on specific topics or collectively communicate to the MQTT broker via a production cell-specific topic channel. It is also noted that in the recent MQTT SparkPlug™ B version, subscriber nodes can also publish to specific topic channels at the MQTT broker for system state information (hence double sided arrows from subscriber nodes in Fig 1.).

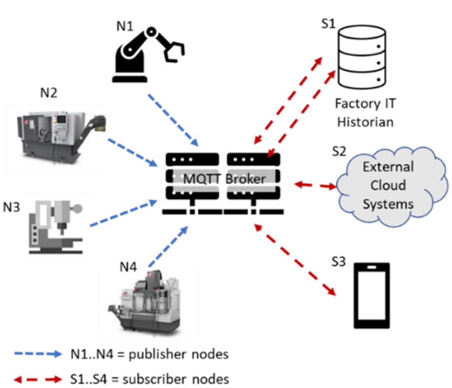

Fig. 1. Industrial implementation of MQTT SparkPlug™ B protocol

This unified namespace architecture ensures that machine assets do not need to communicate with higher-order information systems (enterprise resource planning, content management system) through specified controlled layers. Rather a unified namespace is established that allows decentralized machine-machine or machine-cloud communication architecture without the need for centralized nodes that force communication between various levels. MQTT SpB IIoT architecture allows adding any number of new devices to the nodes and any number of nodes to the existing system. It needs only a table from DB to be created before Historian starts to make records for this device or node. Such an approach enables the vast majority of opportunities for Industry 4.0 technologies to be implemented on a production floor.

4.2. Hardware and Software Setup

In this study, four Haas CNC machines were used (2 Haas VF-2 and 2 Haas ST-10), each with the New Generation Control™ (NGC): VF-2 operates on software version 100.19.000.1402, ST-10 runs on software version 100.20.000.1011. There are three possible ways for machine data collection: HaasConnect™, MTConnect, and MDC - Ethernet Q Commands [14]. MDC - Ethernet Q Commands (Q-codes) use either Ethernet or wireless connection to be established. An operator must first establish a telnet connection with a specified IP address and port in this mode. After this procedure, one can send a specific Q-code, and the NGC will return the tag and the value as a text string. This mode was selected to get the data from the CNC machines. Relying on the information retrieved by the Q-codes provided far more data than the in-built MTConnect agent running within the machines. The machine assets, brokers, and subscribers were connected on a virtual private network. These machines were not connected to the broader university-wide infrastructure due to the insecure Telnet protocol. A high-level system architecture is depicted in Figure 2.

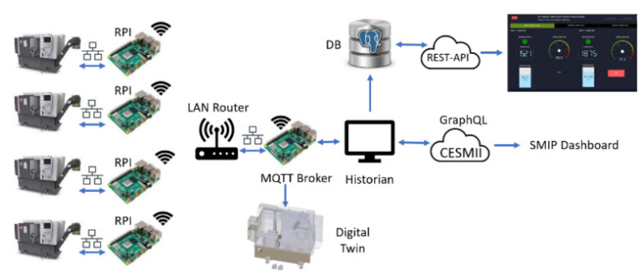

Fig. 2. The architecture of machine data collection from Haas machines

Each machine asset was connected via Ethernet to a Raspberry PI 4B (RPI) that runs an MQTT Publisher node. Arrows between RPI and CNC machines represent 2-way communication between EoN and the asset. RPIs are connected to the local area network (LAN) through a wireless connection. LAN makes sure that no one can externally access the network, and Ethernet connection assures that only one device is connected to the CNC. Every RPI contains a 'publisher' daemon that starts whenever the RPI is powered on. Topics and data are structured following the SpB specification v2.2 [7]. MQTT broker is installed on another RPI within the network that is running 24/7. The Historian continuously collects data from the assets and stores them in a PostgreSQL database (DB). Dashboard reads the data from the DB through the REST server. The REST interface provides a middleware layer to collect data requested by client applications (browsers, dashboards, etc.). Important to note that any client in MQTT network is required to know the URI (uniform resource identifier) of the broker. This can either be an IP address or an URL. The amount of devices connected to the network is limited by the network setup. In our prototype setup, the network could connect up to 253 devices in our subnet and 255 devices for the next 254 subnets (more than 65K devices). This can however be further scaled by network address translation (NAT).

The Haas machines' Q-codes list contains thousands of various parameters and macro variables [22]. Some of these parameters are generic, and some are specific to the model of the CNC machine. Q-codes that retrieved specific CNC machines' timestamps, status/mode coordinates, cutting conditions, and vibration information were utilized for this study. There is also a limit to the number of Q-codes sent through a Telnet protocol before they are read.

4.3. Pub/Sub Model via MQTT- SpB Specification

Table 1. Publisher Client pseudocode

Publisher Client pseudocode

```
read connection parameters from "Pub_config.txt"
read the list of Q-codes from "DB Table columns.csv"
set LWT "OFFLINE" message to NDEATH topic
connect to the MQTT broker
if connection established:
    publish to the topic NBIRTH
    publish to the topic NDATA
else:
    reconnect
if telnet connection is available:
    publish to the topic DBIRTH
    set "running" flag up
else:
    reconnect
while True:
    if STATE topic contains "ONLINE":
        send Q-codes to NGC
        parse data
    if "running" flag is up:
        publish parsed data to the broker
    else:
        publish previous data with flag "STALE"
        set "running" flag down
        publish to the DDEATH topic
```

Publisher code was written in Python, including a paho-mqtt library. Configuration files contain the CNC machine's IP address (Ethernet), the IP address of the MQTT broker (for wireless transfer), and the client's name. The CNC IP address is needed for RPI to send Q-codes, and the broker's IP address is needed to publish. A list of broker IP addresses can contain many values to supply sustainability of the system. It allows changing the CNC machine and broker addresses without interrupting the main code. The publisher runs in an infinite loop, polling the list of Q-codes as necessary. Data is published to the broker only if there are changes in any of the machine parameters that must be reported. Publisher pseudocode is depicted in Table 1.

The following mode of operation is followed: CNC machine does not send data after it is turned on or before it is turned off. The RPI also boots faster than the CNC machine. It publishes node birth certificate and pings telnet port; after receiving the first non-empty response, it is converted to a start-up message, RPI publishes a device birth certificate and sets a flag of running CNC machine; if telnet connection fails or there are any other reasons RPI loses connectivity with CNC machine, it publishes last known data with flag 'STALE' and device death certificate; when the machine is powered-off RPI also turns down and is not able to publish anything to a broker. It makes the Last Will and Testament message (LWT) published to a node death certificate topic. Most edge devices run continuously and do not need to be powered on/off frequently.

Table 2. Subscriber (Historian) pseudocode

Subscriber Client pseudocode

```
read connection parameters from "Sub_config.txt"
connect to the database
set LWT message "OFFLINE" to STATE topic
connect to the MQTT broker
publish message "ONLINE" to STATE topic
subscribe to all device and node topics
while True:
    if a new message in any topic:
        print message "Received message in «topic»"
        if node or device name is not in the list of DB tables:
            print("Incorrect topic name")
        else if message is from NBIRTH or DBIRTH:
            make a record in DB that node or device is powered on
        else if message is from topic NDATA or DDATA:
            make a record in DB with node or device data
        else if message is from topic NDEATH or DDEATH:
            make a record in DB that node or device is powered off
```

The main subscriber associated with the Historian uses configuration files to connect to the DB. When the historian code starts first, it publishes 'ready' to the STATE topic after connecting to the DB. It is used by the publishers subscribed to this topic to begin publishing. On the other hand, Historian is also subscribed to birth/death and data topics. The topics' names are compared to the tables available in DB, and if it matches one of the tables, the payload is stored to the DB. Historian also reads the birth and death certificates and forms respective rows recorded to the DB. Timestamp from Historian is used as a unified source of time. "Keep-alive" time is set for every publisher before connection. MQTT uses a 1.5 times period to reconnect the client. If the client does not respond within this period broker disconnect client and publishes the LWT message. The Subscriber client pseudocode is depicted in Table 2.

The unified namespace decentralized architecture enables the seamless addition of new device assets to the machine or any number of machine assets on the production floor. The pub/sub model also allows easier integration of the IT/OT layers in future factories.

4.4. Machine Profiles on CESMII SMIP

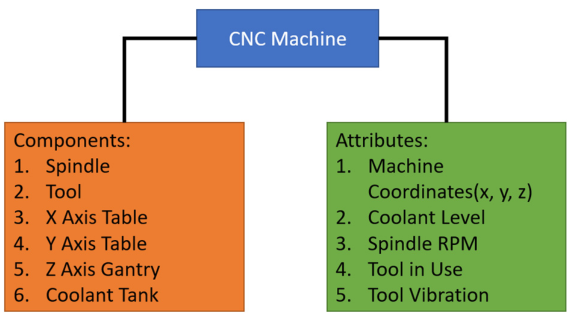

Fig. 3. Generic CNC Profile defined within the CESMII SMIP

Machine profiles provide a semantic taxonomy of tags that help define the characteristic features of physical machine assets. These profiles enable a unified format of type definitions that multiple users can reuse. For example, an original equipment manufacturer (OEM) sells a piece of equipment and includes a Smart Manufacturing Platform profile (SM Profile) that describes the characteristics of the equipment and the run-time data available within it. This SM Profile, or Type definition, can be used to build information systems as features for a machine learning algorithm or to communicate energy consumption information to potential customers [23]. A profile is a generic model of a machine that contains all the working components of the device. Figure 5 gives a sample profile of a CNC machine. As shown in Figure 5, any equipment will have a common set of base components and attributes. This will be true for all devices of the same type used worldwide. Thus, creating a generic profile enables manufacturers to build their equipment type based on a generic model instead of building from scratch, making the process simpler and easier to adopt. The concept of the parent-child relationship also exists within the idea of Machine Profiles.

4.5. Streaming Data to CESMII SMIP Cloud

To facilitate data streaming from the machine asset to the CESMII SMIP Cloud, an Edge Gateway is installed at the facility to facilitate secure high-speed ingress of data into the SMIP. The Gateway (also known as the SM Edge, SM Edge Gateway, or simply the Connector) connects to multiple plant floor (or more genetically OT) data sources to securely transmit data to the core Platform. The Gateway uses an extensible approach to protocol adaptation using Connector Adapters. Connectors can be used to adapt a variety of data sources to the Gateway, which provides store-and-forward and secure data transmission. Though connectors are available for historians such as Wonderware [24] and OSI [25], CESMII SMIP can also accept data directly from MQTT topics through GraphQL. The machine data collected by the edge nodes is written into a PostgresSQL database as time-series records. These records can now be accessed for multiple purposes. Dashboards running locally access the database for live data plotting.

The data is also accessed through Python scripts running locally to utilize GraphQL to write the incoming values to the corresponding tag IDs on SMIP. The process involves a call to retrieve all tag IDs and the related variable names. The names are then matched to the column names/keys in the Python dictionary pulled from the DB. All tags with non-null values in the table are written as records to SMIP using GraphQL command run from within the Python script. The records update occurs as and when the records are written to the DB, thus reducing delay.

5. Results

5.1. Cloud and Local Factory Monitoring Dashboard

The purpose of interconnectedness on the factory floor is to make processes more efficient and inform critical decision-making based on key metrics from the process. The best way to visualize processes in real-time is to use dashboards. The largest advantages of using MQTT SpB are the following:

- Disconnection from the main database and restricted access.
- The broker can have multiple subscribers.

Q-codes from Haas NGC support the requirement of most of the parameters that are needed to implement a local factory SCADA system. MQTT's capability to seamlessly add new assets makes it a perfect fit for the Industry 4.0 IIoT. The factory network traffic is not clogged with useless poll/response requests, and any data can be transferred starting from a byte and ending with video files. Dashboards have a minimum latency, retrieve real-time data directly from the MQTT broker, and access historical data from the REST server. In this application, the same data that the CESMII SMIP is subscribing to is also subscribed to by a dashboard application. A sample dashboard is shown in Figure 4.

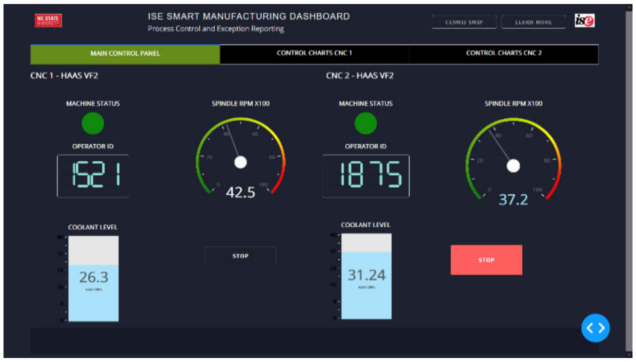

Fig. 4. Dashboard of key metrics transferred from two CNC machines in real-time

5.2. Powering Digital Twins (3D Models)

Digital twins (DT) [26] can get the information update from the Historian and supply enough data to keep up to date with the physical asset. Haas NGC Q-codes yield enough data to make a visual DT of the CNC machine, implemented within an Autodesk Fusion 360 CAD/CAM environment. In this approach, data published by RPI from one of the VF-2 machines (Figure 5) was utilized to demonstrate the DT of the machine within a CAD/CAM package. We have written Python scripts using Autodesk Fusion 360 API and paho-mqtt library. A 3D model of the Haas VF-2 was downloaded and adapted to reflect the actual machine asset. The plugin script is executed, which enables the Fusion 360-MQTT client to subscribe to the DDATA topic of VF-2 machine and performs movements in 3D space identical to movements of the physical asset. The DT uses the same data source as the Historian and the Dashboard, showing a good example of the unified namespace since it is subscribed to the same topic channels as the Historian and the Dashboard.

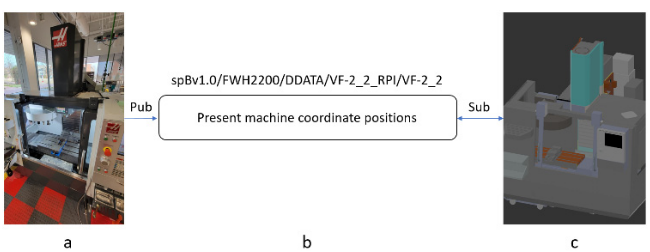

Fig. 5. Example of DT (c) consuming data published by the physical asset (a) through subscribing to the device DDATA topic (b)

6. Conclusion

This project demonstrates the application of the MQTT SpB specification to stream data from machine assets on a shop floor through a publish/subscribe model to clients that can be situated both within and outside the factory. This type of communication architecture collapses the Purdue Model by removing structured layers to enable a more node-to-node communication of assets. It allows interoperability between various machine assets and stronger IT/OT networks integration to boost factory-floor visibility from the enterprise level down to the machine asset. The use of profiles within cloud platforms can enable the direct exchange of profiles by small and medium scale manufacturers who can leverage the semantic definition of various machine assets for faster integration of data from the factory floor to a wider expanse of manufacturing apps that can deliver smart manufacturing services and enable smart factories.

This work demonstrated how the MQTT SpB works with Haas CNC machines. Work is still required to translate machine specific codes to gather data and have it transported to subscriber endpoint. This means an information model is required to enable interoperability between various device assets and machine vendors. While information models like MTConnect exists for CNC machines, not all machine types and their associated assets have means to standardize information coding to enable interoperability. This model required manual selection of parameters and their Q-codes. It can be easily seen that every other vendor has its own proprietary way of accessing the machine data and protocols being used. Other assets will require other approaches to perform data collection.

Lightweight MQTT protocol with SpB specification surpasses OPC-UA and MTConnect protocols in reliability, scalability and the fraction of the useful network traffic consumption. Nevertheless, the absence of wide implementation of this specification in IIoT impedes the development of the plug and play solutions. MTConnect and OPC foundation have been working on finding the optimal solution for the pub/sub IIoT model [27], [28]. Some software solution to use MTConnect and the MQTT protocol also exist on the market [29]. We expect that in the future a companion spec for MTConnect with MQTT is required for further enhancing the use IIoT in factories that have CNC metal cutting machines.

The implementation of our approach can be used in retrofitting of the equipment opposing the work of Etz et al. [30]. The future work orients toward increasing the nomenclature of the equipment type and vendors: CNC machines, robots, conveyor stations, sensor suites, 3D printers, autonomous guided vehicles that are typically used in a production environment. Existing IT systems do not have functionality of direct access to the MQTT broker, thus it has to retrieve the data from DB, which violates the idea of the unified namespace. Future work in this direction is needed to ensure IT software vendors have the ability to communicate (subscribe) directly with the MQTT broker. Sparkplug B has only been released in 2019 and no further revisions are available now. We expect that as adoption and industry specific use cases arise, more revisions and additions will be made to be more applicable in industrial settings.

Acknowledgments

The work conducted in this project was performed with partial support from CESMII – Clean Energy Smart

Manufacturing Innovation Institute through its regional smart manufacturing innovation center funding program.

References

[1] T. Primya, G. Kanagaraj, and G. Subashini, "An Overview with Current Advances in Industrial Internet of Things (IIoT)," in *Proceedings of International Conference on Communication, Circuits, and Systems*, Singapore, 2021, pp. 89–97. doi: 10.1007/978-981-33-4866-0_12.

[2] D.-S. Kim and H. Tran-Dang, "An Overview on Industrial Internet of Things," in *Industrial Sensors and Controls in Communication Networks: From Wired Technologies to Cloud Computing and the Internet of Things*, D.-S. Kim and H. Tran-Dang, Eds. Cham: Springer International Publishing, 2019, pp. 207–216. doi: 10.1007/978-3-030-04927-0_16.

[3] L. Nguyen and A. Kortun, "Real-time Optimisation for Industrial Internet of Things (IIoT): Overview, Challenges and Opportunities," *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, vol. 7, no. 25, p. 167654, Jan. 2021, doi: 10.4108/eai.16-12-2020.167654.

[4] "Is the Purdue Model Still Relevant?," *Automation World*, May 12, 2020. https://www.automationworld.com/factory/iiot/article/21132891/is-the-purdue-model-still-relevant (accessed Nov. 27, 2021).

[5] OPCFoundation, "OPC 10000-14 Unified Architecture Part 14 Pub Sub," *OPC UA Online Reference*. https://reference.opcfoundation.org/v104/Core/docs/Part14/ (accessed Mar. 03, 2022).

[6] S. Vinoski, "Advanced Message Queuing Protocol," *IEEE Internet Comput.*, vol. 10, no. 6, pp. 87–89, Nov. 2006, doi: 10.1109/MIC.2006.116.

[7] Cirrus Link, "Sparkplug Topic Namespace and State ManagementV2.2-with appendix B format - Eclipse.pdf." Eclipse Foundation, Inc, Oct. 11, 2019. Accessed: Nov. 02, 2021. [Online]. Available: https://www.eclipse.org/tahu/spec/Sparkplug%20Topic%20Namespace%20and%20State%20ManagementV2.2-with%20appendix%20B%20format%20-%20Eclipse.pdf

[8] "Comparison MQTT vs OPC-UA," *Muutech Monitoring Solutions*, Aug. 20, 2019. https://www.muutech.com/en/comparison-mqtt-vs-opc-ua/ (accessed Mar. 03, 2022).

[9] "Technology - SM Innovation Platform," *CESMII – The Smart Manufacturing Institute*. https://www.cesmii.org/technology-smip/ (accessed Nov. 26, 2021).

[10] T. J. Williams, Ed., *A Reference model for computer integrated manufacturing (CIM]: a description from the viewpoint of industrial automation*. Research Triangle Park, N.C: Instrument Society of America, 1989.

[11] "MQTT Enables IIoT Security Best Practices within the Purdue Model," *IIoT World*, Aug. 02, 2021. https://www.iiot-world.com/industrial-iot/connected-industry/mqtt-enables-iiot-security-best-practices-within-the-purdue-model/ (accessed Nov. 27, 2021).

[12] "MTConnect_Part_1-0-Overview_and_Fundamentals_1-8-0.pdf." https://docs.mtconnect.org/MTConnect_Part_1-0-Overview_and_Fundamentals_1-8-0.pdf (accessed Nov. 28, 2021).

[13] "Machine Monitoring System Based on MTConnect Technology | Elsevier Enhanced Reader." https://reader.elsevier.com/reader/sd/pii/S2212827114009664?token=294385BA44229E959F4F4A6C18734D418B36FBA2B25E797B4DB78046F33EB1C95844075768BFAB8E32312A0D81632A8C&originRegion=us-east-1&originCreation=20211128194733 (accessed Nov. 28, 2021).

[14] "Machine Data Collection - NGC." https://www.haascnc.com/service/troubleshooting-and-how-to/how-to/machine-data-collection---ngc.html (accessed Nov. 02, 2021).

[15] "Unified Architecture," *OPC Foundation*. https://opcfoundation.org/about/opc-technologies/opc-ua/ (accessed Nov. 28, 2021).

[16] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC Unified Architecture*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. doi: 10.1007/978-3-540-68899-0.

[17] A. Helmbrecht-Schaar, "IIoT Protocols: OPC UA vs. MQTT Sparkplug." https://www.hivemq.com/iiot-protocols-opc-ua-mqtt-sparkplug-comparison/ (accessed Nov. 28, 2021).

[18] "OPC Foundation, 'OPC UA for ISA-95 Common Object Model,'" *OPC Foundation*. https://opcfoundation.org/members/ (accessed Nov. 28, 2021).

[19] I. Seilonen, T. Tuovinen, J. Elovaara, I. Tuomi, and T. Oksanen, "Aggregating OPC UA servers for monitoring manufacturing systems and mobile work machines," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2016, pp. 1–4. doi: 10.1109/ETFA.2016.7733739.

[20] S. Cavalieri, D. Di Stefano, M. G. Salafia, and M. S. Scroppo, "A web-based platform for OPC UA integration in IIoT environment," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2017, pp. 1–6. doi: 10.1109/ETFA.2017.8247713.

[21] R. Schiekofer, A. Scholz, and M. Weyrich, "REST based OPC UA for the IIoT," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2018, vol. 1, pp. 274–281. doi: 10.1109/ETFA.2018.8502516.

[22] Haas Automation Inc., *Mill Operator's Manual*, Revision F. Oxnard, CA 93030-8933: Haas Automation Inc., 2017. [Online]. Available: https://www.haascnc.com/content/dam/haascnc/en/service/manual/operator/english---mill-ngc---operator's-manual---2017.pdf

[23] *Smart Manufacturing Profiles*. CESMII - The Smart Manufacturing Institute, 2021. Accessed: Nov. 28, 2021. [Online]. Available: https://github.com/cesmii/SMProfiles

[24] A. Vaclavova and M. Kebisek, "Design of Virtual Model of Production Line Using Wonderware ArchestrA," in *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, Jun. 2018, pp. 000425–000430. doi: 10.1109/INES.2018.8523998.

[25] "PI System - Connecting data, operations & people | OSIsoft." https://www.osisoft.com/pi-system (accessed Nov. 28, 2021).

[26] G. Shao, "Use Case Scenarios for Digital Twin Implementation Based on ISO 23247," National Institute of Standards and Technology, May 2021. doi: 10.6028/NIST.AMS.400-2.

[27] "A Cyber-Physical Machine Tools Platform using OPC UA and MTConnect | Elsevier Enhanced Reader." https://reader.elsevier.com/reader/sd/pii/S0278612518301110?token=9237AA1F177489FE62864C144CB3108DD552C40AC4AEDDF9C31351C8F5807FD22C736CB069438FE9A081221A837470CF&originRegion=us-east-1&originCreation=20220305194621 (accessed Mar. 05, 2022).

[28] "MTConnect," *OPC Foundation*. https://opcfoundation.org/markets-collaboration/mtconnect/ (accessed Mar. 05, 2022).

[29] "How to Access MTConnect Data Via MQTT | Industrial Internet of Things Data Platform |," *Open Automation Software*, Jul. 06, 2021. https://openautomationsoftware.com/knowledge-base/how-to-mtconnect-mqtt/ (accessed Mar. 05, 2022).

[30] D. Etz, H. Brantner, and W. Kastner, "Smart Manufacturing Retrofit for Brownfield Systems," *Procedia Manufacturing*, vol. 42, pp. 327–332, Jan. 2020, doi: 10.1016/j.promfg.2020.02.085.