Industrial IoT and Digital Twins for a Smart Factory: An open source toolkit for application design and benchmarking

Jeff Morgan
National University of Ireland, Galway
16 PUBLICATIONS **301** CITATIONS

Muhammad Intizar Ali
Dublin City University
110 PUBLICATIONS **2,117** CITATIONS

Industrial IoT and Digital Twins for a Smart Factory
An open source toolkit for application design and benchmarking

Vignesh Kamath
SFI Confirm Centre for Smart Manufacturing
Data Science Institute, NUI Galway
vignesh.kamath@nuigalway.ie

Jeff Morgan
SFI Confirm Centre for Smart Manufacturing
Data Science Institute, NUI Galway
jeff.morgan@nuigalway.ie

Muhammad Intizar Ali
SFI Confirm Centre for Smart Manufacturing
Data Science Institute, NUI Galway
ali.intizar@nuigalway.ie

Abstract—**The rapid evolution of digital technology and designed intelligence, such as the Internet of Things (IoT), Big data analytics, Artificial Intelligence (AI), Cyber Physical Systems (CPS), has been a catalyst for the 4th industrial revolution (known as industry 4.0). Among other, the two key state-of-the-art concepts in Industry 4.0, are Industrial IoT (IIoT) and digital twins. IIoT facilitates real-time data acquisition, processing and analytics over large amount of sensor data streams produced by sensors installed within a smart factory, while the 'digital twin' concept aims to enable smart factories via the digital replication or representation of physical machines, processes, people in cyber-space. This paper explores the capability of present-state open-source platforms to collectively achieve digital twin capabilities, including IoT real-time data acquisition, virtual representation, analytics, and visualisation. The aim of this work is to 'close the gap' between research and implementation, through a collective open source IoT and Digital Twin architecture. The performance of the open-source architecture in this work, is demonstrated in a use-case utilising industry 'open data', and is bench-marked with universal testing tools.**

Keywords-**Industry 4.0; IoT; Digital Twins; Open-Source; Benchmark;**

I. Introduction

Digital twin refers to a digital replica of potential and actual physical assets (physical twin), such as processes, people, places, systems and devices [1]. The digital representation provides both the elements and the dynamics of how the physical twin operates and lives throughout its life-cycle. These digital twins are based on semantic data models which are updated by real-time sensor data, and have the potential to incorporate simulation, enabling prognostic assessment [2]. Digital Twins are interlinked with other Industry 4.0 technologies, including Cyber Physical Systems (CPS), and the Internet of Things (IoT) [1], forming an evolutionary paradigm for smart manufacturing in smart factories.

IoT technologies are gaining momentum with every passing day and continue to support a variety of applications facilitating real-time monitoring, analytics and decision support systems. Industrial IoT is one of the key pillars for the realization of the true vision of Industry 4.0 and one of the major underlying technology for digital twins. A large number of off-the-shelf IoT middleware solutions are available, providing the businesses to opt for either for an open source solutions or licensed enterprise solutions depending on their requirements. Similarly, both open source and proprietary solutions are available for digital twins such as Eclipse Ditto (open source) and Microsoft Azure Digital Twin (proprietary).

Open-Source software exposes source code for public inspecting, utilisation, and enhancement [3]. Open-source software promotes collaboration and leaning, with the advantages of being royalty free, unlocking lower software development costs, reduced time to market, with abundant support, and the ability to scale and consolidate. The open source tools are often the first choice for academia due to easy access and flexibility for further extension. However, industry is often left to make a hard choice between open source tools (steep learning curve) and propriety solutions (vendor lock-in).

Presently, a challenge for industry and academia is how to 'close the gap' between research and implementation. The value in utilising digital twins for smart manufacturing is clear [4], however the technology stack implementation possibilities are boundless in today's digital landscape, with a high risk of non-standardisation, and cross-platform incompatibility. Academically, international researchers are striving to achieve Digital Twin capabilities http://www.maya-euproject.com/index.php/project, however the technology platforms established can be "closed source" and not available to the public. Commercially, enabling digital twin technology is in-part present in high-end technology providers [5], however cross platform unification requires high levels of investment. As such, there is an opportunity for academic and industry collaboration, with accelerated development and support present in open-source platforms for IoT and Digital Twins.

In this paper, we present an open source toolkit designed by knitting together a few hand-picked open source tools for each data processing layer of IoT and digital twin reference architectures. We test the performance of the designed toolkit using a benchmark dataset.

II. Related Work

Open-source software has been recognised to provide key software features for enabling industry 4.0 technology [3] , such as data aggregation, security, device management, event management and analysis, and digital twin management. Presently however, there are more than a 1000 IoT open source platforms available [6], which possesses both opportunities and challenges. Open source IoT platform examples [7] include: Eclipse Kuksa, Eclipse Hono, Eclipse Hawkbit, Eclipse Lesham, Eclipse Keti, Eclipse Chi, Grafana. The authors in [7] explore the utilisation of these solutions for data transmission, visualisation and the device management services. The authors in [3] provide a comprehensive review of open source tools, libraries, data representations, date exchange, data analytics, and derived a high-level micro-services architecture, consisting of Apache Kafka, RabbitMQ, logstash, InfluxDB, Grafana, Elacticsearch, kibana, hadoop, tensorflow. Presently, Leveraging of open source technology by industry is becoming a reality, with information technology (IT) and operational technology (OT) solutions being developed collaboratively, as seen in Bosch IoT Suite [3] https://www.bosch-iot-suite.com/.

This paper, explores the collective capability of 5 open-source platforms, namely Eclipse Hono[1], Eclipse Ditto[2], Apache Kafka[3], Influx DB[4] and Grafana[5]. The collective functionally of these platforms is designed to enable digital twin capabilities, including IoT real-time data acquisition, virtual representation and management, real-time analytics, and visualisation. The open source architecture is demonstrated with industry 'open data' made available by Microsoft, and bench-marked with universal testing tools, such as 'Apache jmeter', to understand scaling performance capacity.

III. Open Source Architecture

In this section, we present architecture of our open source toolkit for industrial IoT and digital twin. Below we give a brief overview of the 5 selected tools;

Eclipse Hono provides remote service interfaces for connecting large numbers of IoT devices and interacting with them in a uniform way regardless of the device communication protocol. Hono supports devices communicating via common IoT protocols, as standard, such as HTTP, MQTT and AMQP. It is a micro service-based architecture which utilises a reactive programming model, allowing for horizontal scaling out. As, such communication is easy via a common API regardless of the device protocol.

[1]http://www.eclipse.org/hono/
[2]https://www.eclipse.org/ditto/
[3]http://kafka.apache.org/
[4]https://www.influxdata.com/
[5]https://grafana.com/

Eclipse Ditto is a framework that supports IoT Digital twins software pattern implementation. Ditto's capabilities includes: mirrors physical assets/devices, acts as a "single source of truth" for a physical asset, provides aspects and services around devices, and keeps real and digital worlds in sync. Ditto's digital twin framework provides web API's to interact with digital twins, ensures access can only be done with authorisation, enables one-to-one or one-to-many Digital twin interactions, and integrated into other back-end infrastructures, such as brokers and messaging systems.

Apache Kafka enables the building of data pipelines for real-time streaming applications. Kafka is seamlessly horizontally scalabe, false tolerant and high speed. The three main capabilities of Kafka is to, 1. publish and subscribe to stream of records, 2. store these records in fault tolerant way, and 3. process them as they occur.

Influx DB is a real-time series database, which is simple to setup and scale. A time-series database is used to store log, multiple types of data (e.g. sensor data), over a period of time. Key functionally of Influx DB, is data is written in real-time and read in real-time, with a potential of a million writes per second.

Grafana is an analytics and monitoring solution which provides plugin data source models and support for many of the most popular time series databases, such as: Graphite, Prometheus, Elastic search and Influx DB. Grafana enables users to query, visualize and alert on metrics and logs from multiple stored locations stored.

The combination of these 5 open source platforms provide a universal architecture for the Digtal Twin, as seen in Figure 1. This architecture provides real-time IoT connectivity and data acquisition (Hono), virtual representation and management (Ditto), horizontal data pipeline connectivity (Kafka), time series data storage (Influx DB), data analytics and visualisation (Grafana).

IV. Use Case

The following use-case demonstrates the real-time capability and application of the open source Digital Twin architecture, with industrial 'open data' provided as part of Microsoft challenges in 'predictive maintenance' category which is found in the following link shorturl.at/efhoV [6]. The telemetry time series machine data is defined by 4 variables: voltage, rotation, pressure, and vibration. These measurements were collected from 100 industrial machines thought the year 2015. In order to replicate a real-time scenario for this use-case, a cronjob is created, which acts a scheduler, to take data of all 100 machines from the data set every minute. Additionally, a tenant, e.g. a logical entity which groups together a set of devices, is created which registered and grouped the 100 'virtual' machines with

[6]https://github.com/ashishpatel26/Predictive_Maintenance_using_Machine-Learning_Microsoft_Casestudy/tree/master/data

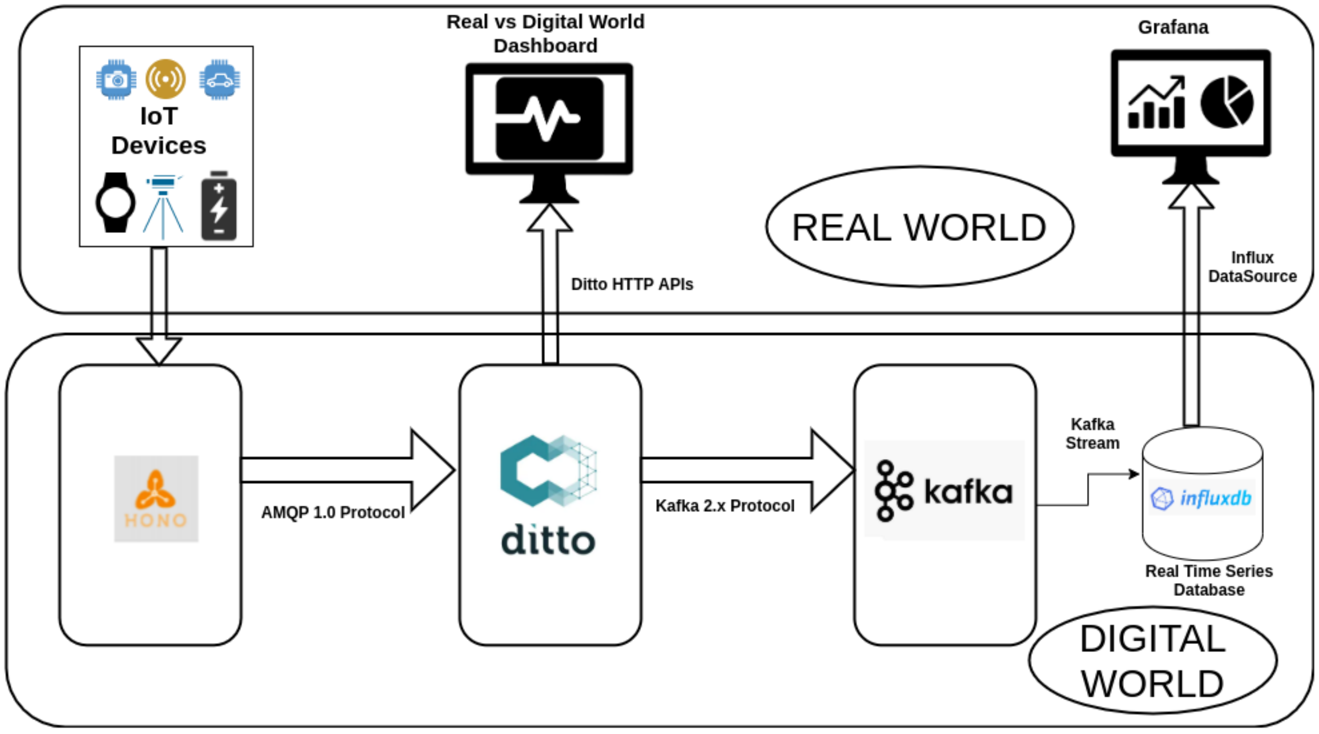

Figure 1. Building digital twin applications using open source tools

unique ID and password authentications. Once activated, the cronjob sends data to the register tenant every minute, simulating the real-time data acquisition activity of the 100 ’virtual’ IoT machines.

The detailed setup of the architecture and use-case is defined as such:

- Standard installation of all 5 open source platforms per their official websites.
- Establishing the connection between Eclipse Hono and Eclipse Ditto
  - Creating a Tenant in Hono - The information registered for a tenant is used to determine if devices belonging to the tenant are allowed to connect to a certain protocol adapter, or if devices are required to authenticate. The Tenant API is defined by means of AMQP 1.0 message exchanges, i.e. a client needs to connect to Hono using an AMQP 1.0 client in order to invoke operations of the API as described in the following sections.
  - Registering the device in the Hono Tenant – Any number of devices can be registered in a Tenant. The Device Registration API is used by Hono’s protocol adapters to get information about the connected devices.
  - Authenticating the registered Devices in Hono – Each device in a tenant will have unique authentication ID password, which will be set during this phase.
  - Creating a Thing (Ditto) – This will act as a digital twin of the registered devices in Hono once the AMQP connection is established. In this use-case 100 devices are registered, as such 100 things are created, which represent 100 digital twins of the devices.
  - Creating an AMQP communication connection between Hono and Ditto for a particular Tenant. - Send data from the registered devices by making use of the Telemetry API of Eclipse Hono. The digital twin with the ’Thing’ ID name-space:demo-device-id will receive the updated value which is also reflected at the twin’s HTTP endpoint, for example: http://yourserverIP/api/2/things/namespace:demo-device-id.
- Establish the connection between Eclipse Ditto and Influx DB via Kafka
  - Ditto will store only the latest state sensor values of a digital twin. As such, a connection is made between Ditto and influx DB, via kafka, in order to keep a track on the historical data. This connection is made through the Kafka 2.x protocol and influx API.
- Visualization
  - Grafana provides inherent data source plugin API’s

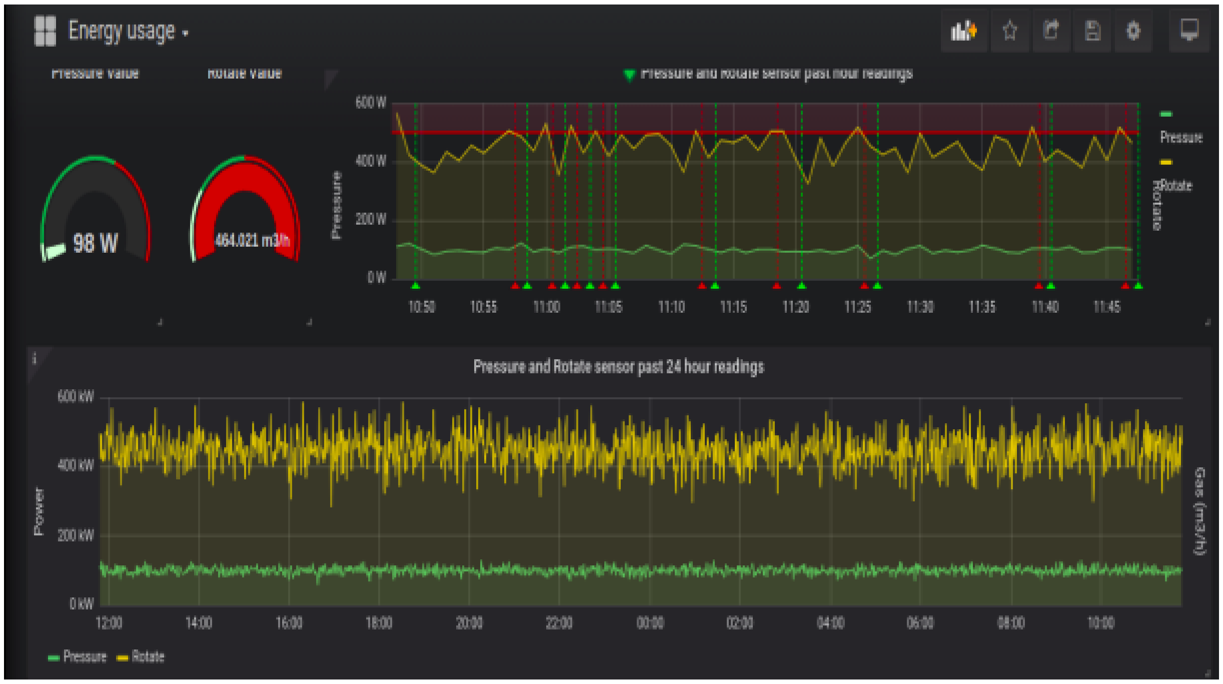

Figure 2. Grafana output of use case

for the connection to influx DB, enabling the historical values, e.g. past hour/day/week, to be interpolated and visualised. Grafana is scalable and can visualise the data of all 100 machines in the use-case.

– In addition to the Grafana based visualization, a direct DITTO communication JavaScript based management dashboard was created, to connect the real world and the virtual world, as seen in Figure 2. This dashboard is updated in real time via the HTTP API's. Using this management dashboard users can easily access, create, edit or delete digital twins, in a user-friendly way.

The visual results of the architecture can be seen in Figure 2. Data is seamlessly streamed in real-time throughout the open source platforms. The insight into the physical process is provided digitally in current value meters (x-value), and historical value graphs (y-time). The solution will enable a maintenance engineer to monitor the performance of 100 machines, with high / low limits, and enable the create of events for sensor readings exceeding thresholds.

V. Performance Testing

For the adoption of open source technology in academic and commercial use, the capability of the technology must be understood in regards to function, scale, and performance. As such, the performance of the open source architecture is evaluated under different load scenarios. The universal testing tool 'Apache Jmeter' is utilised for the evaluation (Test 1 and 2), as well as custom python scripting (Test 3). Testing is evaluated based on scaling digital twins (machines), identifying client and device execution time, server response time, and the presence of errors.

The testbed computer hardware/software specification is as follows: LAN: Ditto server is installed in a Linux Server with 64 GB RAM and Intel(R) Xeon(R) CPU E5606 @2.13 GHz clock speed. LAN: Python test scripts run from a Linux machine (x2) with 8GB memory and Intel (R) Core(TM) I7-8650U CPU@1.9GHz clock speed. WAN: Google virtual machines (x2) are utilised through 'dotcom-monitor' (https://userauth.dotcom-monitor.com) WAN: Apache Jmeter is utilised through the online testing tool dotcom-monitor (https://userauth.dotcom-monitor.com)

A. *Test 1 - Producer WAN*

The 'Producer WAN' performance test explores the interaction of 'users' directly with Ditto, in a WAN configuration, leveraging Google virtual machines and Apache Jmeter. This test linearly increases the number of virtual users, 1 every 3 seconds, until the limit of 100 users is achieved, equating to 100 users x 3 sec = 5 minutes. An additional 2 minutes is added to the test, maintaining the 100 users, equating to a total test time of 7 minutes. Virtual users interact with Ditto by sending HTTP 'POST' requests to change digital twin sensor values, which is 4 values per POST request. The peak of this test is when the 100 virtual users is reached.

The results are seen in Figure 3, performance is equated from server response time of Ditto. The first plot in the figure compares virtual users count over time, the second

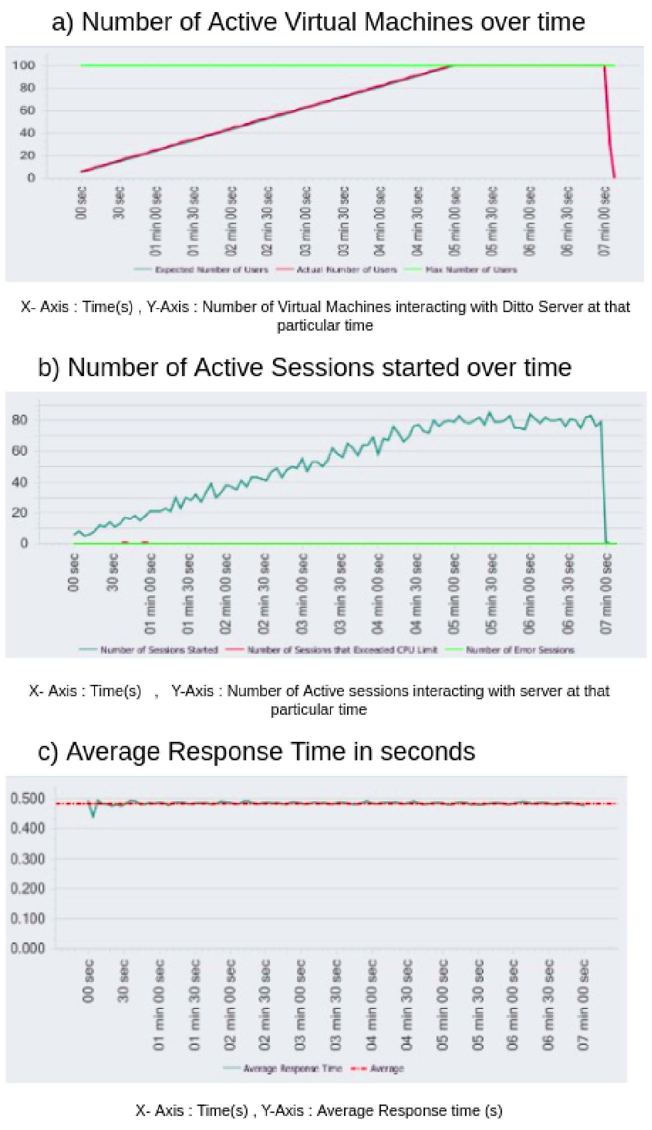

Figure 3. WAN test Results

plot compares active open sessions over time, while the third plot shows response times for users communicating with Ditto. The results identify that the average response time of requests is 0.49 sec with no increased latency with increased users. As such, the maximum number of sessions opened sequentially is bench-marked at 2 requests per second. In total, 5578 HTTP Post requests were sent to Ditto across the 7 minute testing time, equating to on average 13 sessions active per second, with a maximum 85 active sessions recorded at one time. No errors were recorded by any of the users throughout the test.

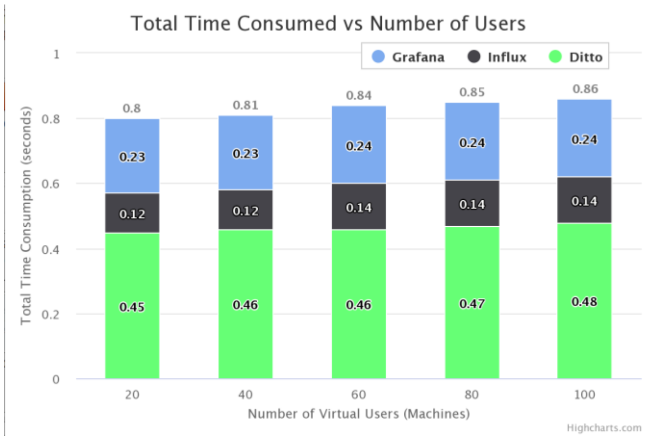

Figure 4. Time Consumed vs Total Number of users

B. Test 2 - Architecture Flow LAN

The ’architecture flow LAN’ performance test identifies the time taken for data to transfer through the open source architecture. From Physical device (data) to HONO, to DITTO (digital twin), to Influx DB (storage), to Grafana (visualisation). In this test scenario, Digital twins / Devices are scaled from 20-100, each operate in sequence updating 4 sensor values in each digital twin. The average time per test scenario is calculated and presented in Figure 4.

The results identify a fast (sub-second) flow of data throughout the architecture. The highest time consumer within the architecture is transferring data from the device into DITTO. Once present in DITTO, data is more rapidly dispersed to storage and visualisaiton. The scaling of digital twins (20+ increments) has a average 1.5 percent increase effect on total time, which is shared amongst architecture components.

C. Test 3 - Generation

The ’Generation’ performance test explores the speed of dynamically generating digital twins through batch scripting in a LAN configuration. This test measures the execution time of a single script to create a set amount of digital twins. As shown in Table I, a total of 2500 devices were registered in HONO, for test purposes. Each twin has the same composition of 4 sensor values. In the first case scenario - 1 computer is running 1 script to update a total of 2500 digital twins. In the second scenario - 2 computers are running 1 script each to update a total of 2500 digital twins, which is 1250 each, in parallel.

The results identify, for x1 computer, the generation of 100-2500, results in a speed range of 33-23 digital twins per sec. For x2 computers, the generation of 100-2500, results in a speed range of 69-46 digital twins per sec. For x2 computers, the total execution time across two computers is halved, enabling twins to be created faster in

No. Digital Twins	Script time (sec) x 1 computer	Script time (sec) x 2 computer
100	3	1.45
1000	33	16
1500	54	29
2000	80	42
2500	106	54

Table I

parallel. Additionally, a linear decrease in generation speed, is observed in both scenarios of 30 percent, with an increase in total digital twins from 100-2500.

VI. Conclusion

This paper explored the capability of present-state open source platforms to collectively form a open source architecture for smart factory solutions. The functional capability of the architecture was demonstrated with a use case leveraging industrial open data, which included: IoT real-time data acquisition, virtual representation and management, real-time analytics, and visualisation. Furthermore, performance testing was achieved with various WAN and LAN scenarios, including the scaling of multiple clients and devices.

Performance bench marking was established in regards to execution and response time, for data throughput among architecture components, and dynamic digital twin generation. The results identify a low-medium speed [K2] (2Hz per user WAN/LAN), robust (zero errors, low latency), highly dynamic (0-2500 digital twins, 33-23 gen-per-sec), scalable (client and device, 0-100 users), open-source architecture for IoT smart factory solutions.

In conclusion, this work was made possible by open source communities and universal collaboration, and the pursuit advanced science and knowledge in industry 4.0. The open source architecture is available for both academic and industrial use. Further 'closing the gap' between academia and industry, is possible through new use-cases, user-friendly tools for operation and development, the inclusion new features and services embedded within the architecture or in collaboration. Any changes in architecture composition can be benchmarked and compared with the findings presented in this paper.

In future, we intend to use this work for deploying different solutions for Industry 4.0 particularly application of AI and semantic Web techniques to develop intelligent applications for smart manufacturing [8]. Our proposed framework is flexible and extensible allowing even cross domain applications on top of our digital twin infrastructure [9].

Acknowledgment

This publication has emanated from research supported by research grants from Science Foundation Ireland (SFI) under Grant Number SFI/16/RC/3918 (Confirm) and SFI/12/RC/2289_P2 (Insight), co-funded by the European Regional Development Fund.

References

[1] Yuqian Lu et al. "Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues". In: *Robotics and Computer-Integrated Manufacturing* 61 (2020), p. 101837. ISSN: 0736-5845.

[2] Elisa Negri, Luca Fumagalli, and Marco Macchi. "A Review of the Roles of Digital Twin in CPS-based Production Systems". In: *Procedia Manufacturing* 11 (2017). 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy, pp. 939–948. ISSN: 2351-9789.

[3] Violeta Damjanovic-Behrendt and Wernher Behrendt. "An open source approach to the design and implementation of Digital Twins for Smart Manufacturing". In: *International Journal of Computer Integrated Manufacturing* 32.4-5 (2019), pp. 366–384.

[4] Peter Augustine. "Chapter Four - The industry use cases for the Digital Twin idea". In: *The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases*. Vol. 117. 1. Elsevier, 2020, pp. 79–105.

[5] Qinglin Qi et al. "Enabling technologies and tools for digital twin". In: *Journal of Manufacturing Systems* (2019). ISSN: 0278-6125.

[6] Robert Höttger et al. "Combining Eclipse IoT Technologies for a RPI3-Rover along with Eclipse Kuksa." In: *Software Engineering (Workshops)*. 2018, pp. 101–106.

[7] P. Zehnder and D. Riemer. "Representing Industrial Data Streams in Digital Twins using Semantic Labeling". In: *2018 IEEE International Conference on Big Data (Big Data)*. Dec. 2018, pp. 4223–4226.

[8] Pankesh Patel, Muhammad Intizar Ali, and Amit Sheth. "From raw data to smart manufacturing: AI and semantic web of things for industry 4.0". In: *IEEE Intelligent Systems* 33.4 (2018), pp. 79–86.

[9] Muhammad Intizar et al. "Multi-layer cross domain reasoning over distributed autonomous IoT applications". In: (2017).