Open Source Based Industrial IoT Platforms for Smart Factory: Concept, Comparison and Challenges

Myungsoo Kim, Jaehyeong Lee, and Jongpil Jeong(✉)

Department of Smart Factory Convergence, Sungkyunkwan University, Suwon, Gyeonggi-do 16419, Republic of Korea
{sioals,objective,jpjeong}@skku.edu

Abstract. As Internet of Things (IoT) and Services are applied to manufacturing industries, Industry 4.0 is evolving. For Industry 4.0 to be realized in the factory, it is essential to implement horizontal integration of cross enterprise value networks, vertical integration within the factory. As interest in IoT has increased, number of platforms that are designed to support it has increased greatly. Because of the different approaches, standards, use cases, and the types of Industrial IoT platforms are very complex and diverse. Therefore, it is difficult to understand, choose and apply the appropriate platform. In this study, we examine the function, effectiveness and cautions for open source based platforms that can be applied to IoT based smart manufacturing. In addition, the architecture of five open source IIoT platforms are analyzed by our reference architecture and clean up all in Table. Finally, we present the use cases of Industrial IoT environment and analyze its effects.

Keywords: Smart factory · IIoT platform · Open source · Edge computing

1 Introduction

Most of initial implementation of IoT was a special project rather than a standard solution so system integrators or service providers had to designate scope of implementation, design and manage it. As industrial IoT is becoming more common, employees in industries expect that business system will perform completely and automated tasks. In other words, it is expected that IoT should be available to existing systems and that it can be connected to the Internet and generate business value using data collected and analyzed from manufacturing facilities. Based on these demands, a number of suppliers have developed components that enable them to deploy industrial IoT platforms or applications, often relying on the ecosystem of partners to provide solutions. In most cases, partners provide IoT devices, cloud storage, edge computing, enterprise applications, overall project management or system integration. Industrial IoT, which

S. Misra et al. (Eds.): ICCSA 2019, LNCS 11624, pp. 105–120, 2019.
https://doi.org/10.1007/978-3-030-24311-1_8

is based on automated robots and data, can help businesses innovate so that businesses are fully computerized, highly automated, and autonomously operate in certain areas [1,2].

IoT nowadays promotes the adoption of different open source technologies, standards and protocols that help devices communicate with one another. The following are the drivers that prompt organisations to adopt open source technologies for IoT.

- Cost: Adoption of open source IoT frameworks involves no costs at all, as these are free for use. This encourages the community and organisations to implement IoT without any hesitation.
- Innovation: Open source code from the community helps in building newer applications, leading to more innovation and agility. The developers are able to build different products, which will be interoperable across different OSs such as Android, Windows, iOS (iPhone OS) and Linux.
- Open APIs (Application Programming Interface): Use of open source APIs for the IoT framework offers a common gateway for different software, hardware and the systems to communicate with one another.
- Libraries: An open source IoT framework offers a wide range of libraries, SDKs (Software Development Kit) and open source hardware like Raspberry Pi and Arduino, ensuring that companies remain on the cutting edge of technology by using different open sourced tools to customise IoT platforms.
- Security: Open source software can protect individual data by implementing really strong encryption for the use of the general public, and hence supply the building blocks for mobile security and the protection of data.
- Interoperability: Adoption of open source solves the problem of interoperability.

Today's global market has at least 49 IoT platforms to meet the needs of diverse user and application groups such as enterprise, government, farmer, healthcare, and manufacturing. However, because of the lack of overall knowledge of the IoT platform, researchers and enthusiasts can hardly choose a specific IoT platform when they are in the product or solution development phase that leverages IoT support technology [3]. Savaglio et al. [4] Investigated and compared the most relevant autonomic and cognitive structure for the IoT. They investigated architectural style management and other applications with self management and cognitive abilities. These architectures strive to minimize human intervention and protect the heterogeneity of the device, which is an interesting research topic and deserves further study in the future. Palade et al. [5] studied the evaluation criteria of IoT middleware by calculating the weights of other standards using Analytic Hierarchy Process (AHP) method. They analyzed and compared middleware performance in service registration and service configuration. Finally, they compared the application development process of four middleware through four different scenarios. Kim et al. [6] examined various IoT applications and abstracted the general platform model. They introduce the concept of things, which is closely related to the devices presented in this

paper. The gateway provides connectivity to the platform when things can not communicate directly with the platform. Service users as well as services and software providers are connected to the platform by RESTful APIs. If complex data processing is not required on the platform, service usage may be directly connected to the device to collect metrology data. Thus, the most important thing in implementing a smart factory is data.

This paper consists of the following: In Sect. 2, we define four key functions of the IIoT platform and derived reference alignment through them. In Sect. 3, we compare and analyze features of the five representative open source platforms based on reference architecture of IIoT platform and clean up all in Table. Finally, In Sect. 4, we present use case of industrial environment and analyze its effects.

2 Concept of Open Source Based IIoT Platforms

IoT platform of users can remotely collect data and manage all IoT devices from a single system [7]. Although there are many IoT platforms that are available online, all of them are based on IoT platform host and quality of support. As a result, IoT platform performs a major function of converting its factories to Smart Factories. Overall, the IoT platform should have the following functions: Extract data from equipment, sensors, and devices, Connect and analyze edge devices, Store large amounts of data, analyze, and lastly Control data in real time [8].

2.1 Support for Device Management and Integration

Device management is one of the most important functions that can be expected from any IoT software platform. The IoT platform should maintain a list of connected devices, track their operational status, be able to handle configuration, firmware (or other software) updates, and provide device level error reporting and processing. Ultimately, device users should be able to obtain individual device level statistics. Integrated support is another important function that can be expected from IoT software platforms. The API should provide access to critical operations and data that should be exposed on the IoT platform. It is common to use REST APIs to achieve this objective.

2.2 Securing Information

Information security measures that are needed to operate IoT software platforms are much higher than normal software applications and services. Millions of devices that connect to IoT platforms need to predict a proportional number of vulnerabilities. Generally, network connectivity between IoT devices and IoT software platforms needs to be encrypted with a strong encryption mechanism to avoid the possibility of eavesdropping. However, most low cost, low power devices included in modern IoT software platforms cannot support these

advanced access control measures. Thus, the IoT software platform itself needs to implement an alternative means of dealing with these device level problems. For example, the level can be improved by how to separate IoT traffic into a private network, secure strong information at the cloud application level, and updating the firmware through authentication [9].

2.3 Data Acquisition Protocol

Another important aspect to note is the type of protocol used to communicate data between components of the IoT software platform. IoT platform needs to be expanded to millions or billions of devices (nodes). Lightweight communication protocols should be used to enable low energy use and low network bandwidth functions. The protocols used for data collection can be classified according to several categories, such as applications, payload containers, messaging, and legacy protocols [10].

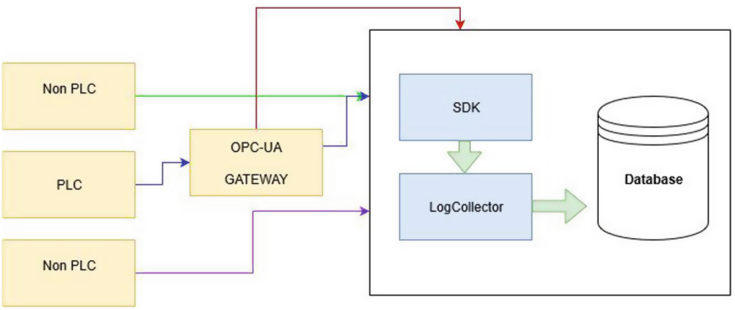

Fig. 1. Edge data flow

In the meantime, the manufacturing industry has been able to achieve productivity enhancement through process specific automation based on Programmable Logic Controller (PLC) and embedded PC. This automation consists of vertically integrated optimization systems through PLC vendors' specialized industrial Ethernet protocols EtherNet/IP, Profinet, CC-Link, POWERLINK and EtherCAT. In Fig. 1, data from various protocols generated by PLC Devices are directed to SDK via OPC UA(Open platform Communications Unified Architecture) protocol. Data will then be stored in the database via LogCollector. The database can be divided as five data types. This can be largely divided into PLC device data, Data from Non PLC devices (through SDK), Data from Non PLC devices (through TCP), Security Data from OPC UA gateways, log data. This data becomes target data for machine learning techniques for efficient process control.

2.4 Data Analysis

Data collected from sensors connected to IoT platforms should be analyzed intelligently in order to gain meaningful insight. There are four main types of analysis available for IoT data: realtime, batch, prediction and interactive analysis. Realtime analysis performs an online analysis of streaming data. Batch analysis executes actions against accumulated data sets. Therefore, the deployment task runs at a scheduled time and can last for hours or days. Predictive analysis focuses on predictions based on various statistics and machine learning techniques. Interactive analytics perform multiple navigation analyses on streaming and batch data. Lastly, this is a realtime analysis that is weighted on all IoT software platforms.

2.5 Reference Architecture of IIoT Platform

Based on the concept of the above four IoT platforms, we have derived reference architecture. In the case of Fig. 2, the function of the IIoT platform is drawn in reference architecture focusing on data processing. First, sensors and actuators are connected to IoT devices. In the IoT gateway and IoT devices, the data flow is not significantly preprocessed, and in the cloud, there is a way to move up to the cloud, or to pre the data before it is stored in the cloud through edge computing, or to extract only the necessary data from the edge device, and store it in the cloud. In other words, devices have processors and storage capacities that can run software and connect to IoT integrated middleware. If the device cannot connect directly to the additional system, it is connected to the gateway. A gateway provides the skills and mechanisms necessary to transform different protocols, communication technologies, and payload formats. IoT Integration Middleware serves as an integrated layer for various types of sensors, actuators, IoT devices and applications. IoT integration middleware is not limited to functions described above. With numerous IIoT open source

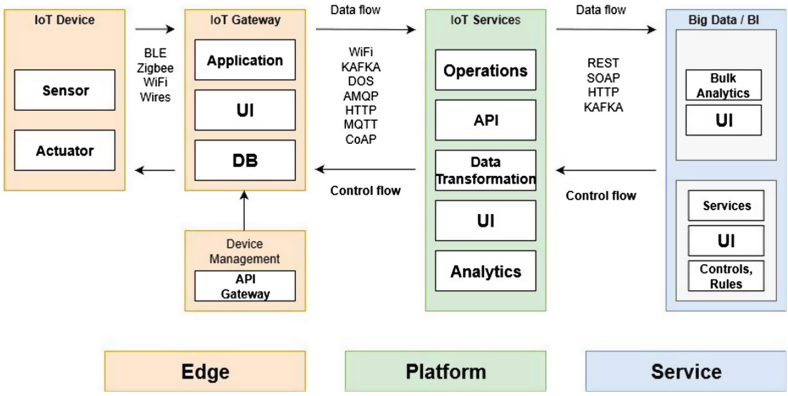

Fig. 2. Reference architecture of IIoT platform

platforms, tools and open stack to add functions, the combination method is endless. Application refers to software that uses IoT Integration Middleware to gain insight into the physical environment and to manipulate the physical environment.

3 Comparison of Open Source Based IIoT Platforms

In this section, five open source based IoT platforms that can be used at industrial sites are compared and analyzed based on their respective open structure. The open source platforms analyzed here are Kaa, Sitewere, DeviceHive, and Fiware. Describes the strengths and weaknesses of each platform, focusing on the data flow of the IIoT reference architecture, and compares the functions that are considered important at the industrial site. Lastly, clean up all in Table 1.

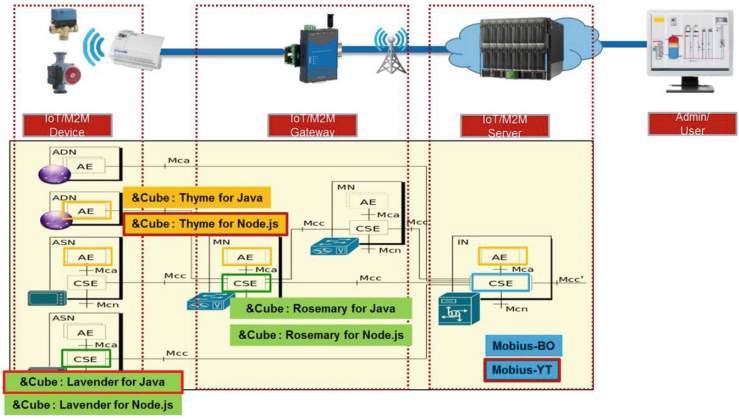

Fig. 3. Mobius IoT platform structure [11]

3.1 Mobius

Figure 3 shows the architecture of Mobius with essential and interaction of functions. With the reference of the architecture of this study, Fig. 3 illustrates a map of servers and devices. The lower half is the configuration of oneM2M (Standards for Machine to Machine and the Internet of Things) system, and the components are IN (Infrastructure Node), MN (Middle Node), ASN (Application Service Node) and ADN (Application Dedicated Node). As described, Mobius is an open source implementation of IN, a server side oneM2M entity. It gathers data from the device and provides data to applications and other gateways or devices. The advantage of Mobius is that it faithfully follows oneM2M. The IoT application communicates with the field domain IoT gateway/device

via Mobius. To connect to Cube through TAS (Ting Adaptation Software) to activate the Internet, Cube communicates with Mobius via oneM2M standard API. In addition, IoT applications use the oneM2M standard API.

And then, MobiusYT is a middleware server platform that connects various IoT devices through physical communication media and creates virtual representation (oneM2M resources) for each IoT device, enabling communication between devices and IoT applications. In this way, MobiusYT provides an open environment and API for users to build their IoT ecosystem by interconnecting their devices and developing userspecific IoT services. Mobius implements IN-CSE, a cloud server in the infrastructure domain. It also supports protocol bindings include HTTP (HyperText Transfer Protocol), CoAP (Constrained Application Protocol), MQTT (Message Queuing Telemetry Transport), and Web sockets.

3.2 Kaa

Figure 4 shows the architecture of Kaa essential and interaction of functions. With the reference of the architecture of this study, the SDK collects data end points, communicates configuration profiles, and enables messaging at the end points. It supports REST communication with the server and can distribute SDK to devices. It is a middleware platform for building IoT end-to-end applications that can be used as a gateway or application server. Supports device management, device interaction, remote device configuration and distributed remote device firmware updates, cloud service creation, data collection and analysis, user behavior analysis, targeted event notification, and big data-based data storage. As IIoT platform of Kaa, It has quite many function as follows: Remote factory floor monitoring, Unified factory wide interconnectivity, Predictive maintenance, Gateway apps and edge analytics, cloud based data storage and analytics. The disadvantage of Kaa (personal server deployment) is that it is not possible to look up data stored through REST APIs on the server, which means that users must develop other develop other applications for this function [12].

Fig. 4. Kaa IoT platform structure [13]

3.3 SiteWhere

Figure 5 shows the architecture of SiteWhere with essential and interaction of functions. With the reference of the architecture of this study, data from devices and commands to device components is organized by device components. Moreover, they also represent sensor and actuator components. This is because they are not explicitly described within an architecture. Users can connect devices with MQTT, AMQP (Advanced Message Queuing Protocol), Stomp, and other devices via self registration, REST service, or batch. Because devices can communicate through platforms and various protocols, a gateway concept exists between devices and platforms, but is not depicted as a separate component. It supports MQTT, AMQP, and REST communication with the server.

It is an open source middleware platform created and maintained licensed under common public attributes license. The main functions of the platform are provided by the SiteWhereTenant engine, including the device management and communication engine. Therefore, these components consist of IoT integration middleware in the reference architecture. The REST APIs and Integration component enables additional applications to be connected to the platform. It aggregates with third party integration frameworks such as Module AnyPoint.

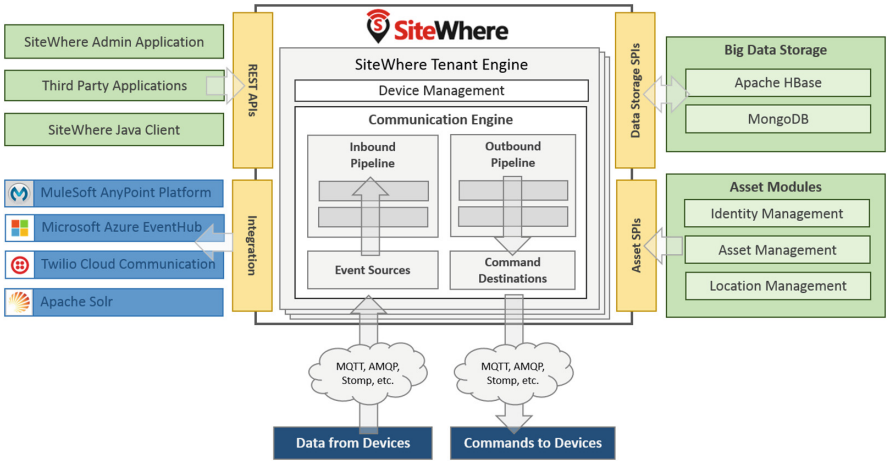

Fig. 5. SiteWhere IoT platform structure [14]

3.4 DeviceHive

Figure 6 shows the architecture of DeviceHive with essential and interaction of functions. With the reference of the architecture of this study, devices communicates and passes it to the gateway. DeviceHive is a micro service based system, built with high scalability and availability in mind. Looking at the architecture

for subsequent data processing, the RESTful and Websocket APIs were provided to enhance user convenience. Finally, various plugin services are left optically to preserve the scalability features of open source platform.

To sum it up, DeviceHive is an open source IoT platform that is rich in functions distributed under Apache 2.0 license. DeviceHive is free to use and change. Even though it is an open source, online versions can be used as PaaS, free trial versions can be provided, or they can be expanded to paid versions. Further function is that Docker and Kubenets placement options are provided. Users can download and use both public and private clouds, and can scale a single virtual machine to an enterprise class cluster. Users can connect to a device or hacker board via REST API, WebSockets, or MQTT. To successfully deploy the solution, users must install PostgreSQL, Apache Kafka, and Java 8 or later. Devicehive's disadvantage is that measurement data on the device (when deploying a personal server) is cached. This means that if user restarts the server or run out of memory, user loses all of your data. If users want this function, they should create additional connectors or modify the back end logic [17].

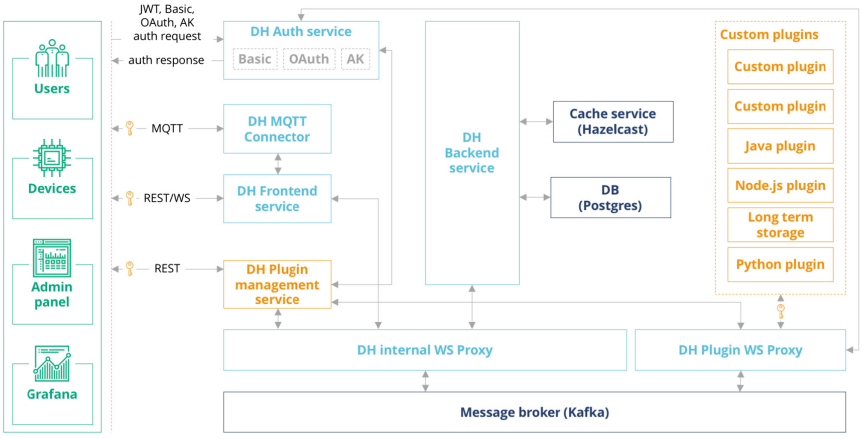

Fig. 6. DeviceHive IoT platform structure [15]

3.5 FIWARE

Figure 7 shows the architecture of FIWARE with essential and interaction of functions. With the reference of the architecture of this study, the implementation of FIWARE IoT architecture ranges from simple scenarios that connect several devices to a data field backend Context Broker using standard IoT communication protocols to scenarios that distribute to large IoT networks and platforms to provide advanced configuration. [16] FIWARE is funded by the

European Union and the European Commission. The FIWARE catalog contains a Generic Enabler (GE) that represents a rich library of components. The architecture in Fig. 6 shows only GE in the IoT segment. FIWARE distinguishes only devices from NGSI13 devices. Since the FIWARE manual explains that the device may have integrated sensors and actuators, all device components consist of device, sensor, and actuator components. Devices can communicate directly with IoT backend or through gateways located within IoT Edge. Both IoT gateway and IoT NGSI gateway will activate and manage device communication with IoT backend.

Thus, IoT Edge represents the gateway of the IoT reference architecture. FIWARE is a Pub/Sub Implementationation of NGSI-9 and NGSI-10 Open RESTful API specifications and only support REST communication with the server. The IoT BackEnd and Data Context Broker provide the main functions of FIWARE, so they are encapsulated by IoT Integration Middleware. FIWARE's documentation describes how additional applications are connected to the platform through a data context broker. It provides platform and standardized API aggregation for variable fields. Application components are not shown in the FIWARE architecture diagram, but they are placed on top of the data context broker as described. Offers an enhanced OpenStack based cloud where features and catalogs are hosted. Advanced Web based User Interface: Interface with Geographic Information and Interactive 3D Chart. Components and tools for creating mashups and application store based services and data distribution that enable data visualization.

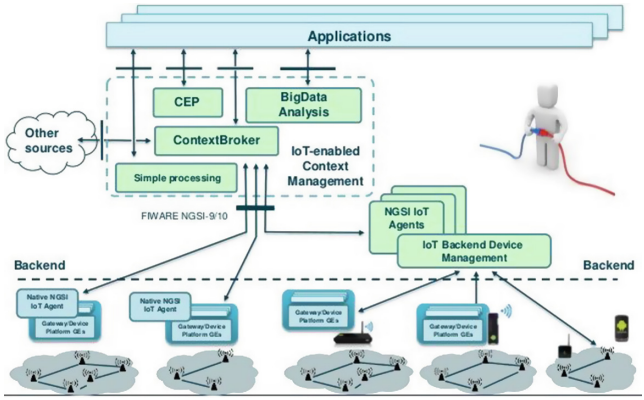

Fig. 7. FIWARE IoT platform structure [16]

Table 1. Comparison of Opensource based IoT platform

Open source platform	Kaa	FIWARE	Device Hive	Sitewhere	Mobius
Protocol	MQTT, CoAP, XMPP, HTTP	HTTP, MQTT	REST API, WebSockets, MQTT	AMQP, Stomp, MQTT, WebSocket	MQTT, CoAP, HTTP, WebSocket
Encryption	RSA, AES, SSL	Basic Authentication	JWT	SSL, Spring Security	Basic Authentication
GUI/Dash Board	✓	✓	✓	✓	✓
Database	MogoDB, Casandra, Hadoop, NoSQL	MySQL	PogreSQL, SAP Hana	MongoDB, HBase, InfluxDB	MySQL
Data analytics	Apache Cassandra, Apache Zappelin	Apache Spark, Apache Flink	Apache Spark	Apache Spark	Apache Spark
Language	C, C++ Java	C, C++ Java, Python	Java script	C++	node.js
Integration	Rest APIs	Rest APIs	Rest APIs MQTT APIs	Rest APIs Any point, and more	Rest APIs

4 Challenges of Open Source Based IIoT Platforms

So far, we analyze and research of the IIoT platform concepts, and application method with practical guide. Also, we compare five open source IoT perform through reference architecture for actual site application. Based on the above research, this section presents IIoT platform use case and analyzes its effects.

4.1 Right Choosing of IIoT Platform for Smart Factory

The first priority is to select suppliers who provide all three previously described IoT functions (application execution, data aggregation and storage and connection management). It is important that these elements are placed in place and fully integrated. When a company purchases or applies a platform that does not have one or more of these layers, it creates unnecessary complexity and costs because the platform must be imported from another layer. Adjacent functions, such as analysis and machine learning, are also becoming increasingly important in realizing the value of IoT. Second, the extent to which the platform is coordinated with the developer's technology should be reviewed. Businesses should

ensure that the core IoT platform complies with the technology of the development team and keep in mind that implementation work is required even if it is a comprehensive platform product. The IoT platform may need to be proficient in certain programming languages. If a developer is a professional Java programmer but needs to use Python in a new IoT platform, difference in technologies will slow down and implementation will be delayed. Finally, openness and ease of integration should be considered. Businesses should choose IoT platforms that meet business requirements and allow distribution while minimizing impact on current systems. Purchaser should first consider whether IoT platform features openness and ease of integration. It is also important to see whether IoT platforms can converge properly between open frameworks (open source). An open framework is modularized using easy to use APIs, and is easily integrated and fits seamlessly into existing IT architectures. This is particularly important for companies with enterprise service buses or complex event handling architectures. This is because a link with the IT environment in the field can seamlessly integrate the new framework with existing systems. In particular, companies should look for vendors that provide a comprehensive platform that meets risks and business models for the IoT. And the platform provider must have a clear understanding of how it differs from its competitors.

4.2 Design of Open Source Based IIoT Platform for Smart Factory

Smart factory is to implement a manufacturing system consisting of intelligent autonomous systems that respond immediately to productivity, efficiency, energy savings, increase in efficiency and energy. Data communication between the process systems and interworking with the higher systems are essential. To this end, the OPC Foundation standardizes OPC UA, an industrial protocol for data integration, and implements standard extensions for realtime data communication and security (Fig. 8).

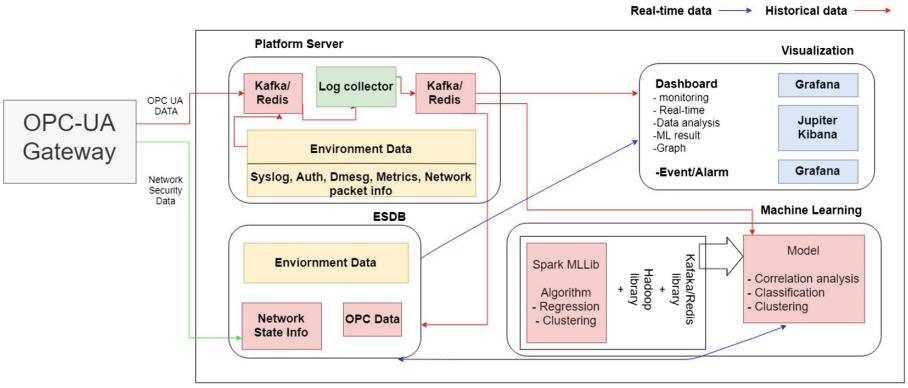

Fig. 8. Open source based IIoT platform for smart factory

Figure 7 describes the Industrial IoT platform architecture proposed in this paper, which shows the overall flow from the edge mentioned in the section above to the platform server and to the area of service where final machine learning techniques are utilized and visualized. It supports a standard protocol (OPC UA) for communication with various industrial equipment operating independently and on a limited basis at industrial sites, and supports real time event storage and streaming at the same time. In addition to PLC, it provides flexible data pipelines for communication with IoT devices such as embedded devices and schema sensors, and has various database interlocking functions for data exchange and storage with external systems. It also supports the analysis of forecast through big data and machine learning and the visualization of real time day sites and analysis data such as tea trees, graphs, tables, and reactive SVG animations, enabling easy implementation of factory specific smart factory in a short period of time. As shown here, the essential part is to integrate data collected from the IIoT platform, process and analyze meaningful information from the data, and provide services that can be used by operators and managers in decision making [17].

4.3 Use Case and Implementation of IIoT Platform

With the introduction of smart factory into manufacturing processes, the optimal manufacturing process can be derived to suit the environment of each demand manufacturing company through the editor of the existing manufacturing process that is difficult to change organically. In addition, it provides sample processes for each type of manufacturer for smart factory learning and supports hands on practices for understanding processes and optimizing process design. By creating these OPC UA training kits, workers at industrial sites help integrate protocols and use control through Opc UA. In addition, it can help understand the flow of data and eventually create OPC UA Gateway for Industrial IoT Platforms [18]. The detailed functions are as follows (Fig. 9).

Fig. 9. Use case and implementation of IIoT platform

- Modeling through OPC UA protocol.
- It Reflects into real time address space after modeling
- GUI makes modeling easy and simple.
- Automatically correct references for easy modeling editing.
- XML extraction allows manual insertion of any OPC server.
- It can be modeled by drag and drop without the need to remodel nodes already registered in the address space.

OPC UA has limitations in systems that require quick response or systems that require high-speed control. High-speed data communication is supported, but we cannot guarantee high-speed responsiveness. TimeSensitive Network (TSN) has emerged as an alternative to the high speed control limits of OPC UA. TSN is a feature set that is added to standard Ethernet to support applications that require deterministic characteristics for data transmission of data. It is similar to real-time communication by reserving traffic lanes for communication packets to eliminate latency for critical data. TSN, which is an extension standard for Ethernet, is under the leadership of CISCO. In addition, in older plants that do not support OPC UA, separate devices that support OPC UA should be additionally installed to work with existing facilities. However, despite the restrictions, 450 companies worldwide are working with OPC Foundation, among them with industrial protocol companies related to smart factory, and standardization is underway. Smart Factory is sometimes called a data factory. This means that the acquisition and integration of data are the most important features in smart factory construction. Applications are being carried out in various research and industries and are challenging in smart manufacturing as well as smooth construction of IIoT platform.

5 Conclusion

IoT shows huge potential that can affect many industries such as automotive, agricultural, manufacturing, and Smart City. According to Ericsson, more than 18 billion IoT related devices are expected to be used worldwide by 2022. In addition, device growth is accelerated by three factors: the emergence of applications for IoT, new business models, and device costs [19]. Smart factory is at the center of the rapid increase in the use of IoT equipment. When establishing industrial IoT platform in order to implement Smart Factory, data processing is the most difficult part for those involved. Therefore, many IoT platforms are making a lot of investments and strategies to secure technical skills in data processing. In terms of the IIoT platform, data processing is involved in all processes with a heart like role, and it has huge competitive edge if it is clear that data processing is done with IoT platform. It is very difficult to select one of these complex and multi functional IIoT platforms with the functions and resources users want. Thus, to solve these problems, the structure of each open source IIoT platform was replaced by a reference architecture. This effect helps people quickly understand structure through communication and data flow

between each element. Section 3 research major functions of using IoT platform and presents the necessary guide from the user's perspective. Based on all of the above, Sect. 4 introduced five open source structures, which are widely used in the field, based on Fig. 1, and researched various features. And then, Table 1, which features five open source IIoT platforms, will be of great help to IoT developers and entrepreneurs who wish to apply them. Finally, we presents Industrial IoT platform Architecture that is proposed and explains that data flow, integration, information processing and service are implemented from edge area to IIoT platform to service area that becomes visualization. Through this paper, users of the IIoT platform, IT developers and researcher can get help when choosing the appropriate IIoT platform and broaden their understanding of the IIoT platform. In the following research, it is possible to implement services and evaluate performance by utilizing two or more open source platforms that are appropriate for the IIoT platform presented above.

Acknowledgement. This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2019-2018-0-01417) supervised by the IITP (Institute for Information & communications Technology Promotion).

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2016R1D1A1B03933828).

This work has supported by the Gyeonggi Techno Park grant funded by the Gyeonggi-Do government (No. Y181802).

References

1. Chen, B., et al.: Smart factory of industry 4.0: key technologies, application case, and challenges. IEEE Access **3**(6), 6505–6519 (2017)
2. Atzori, L., et al.: The internet of things: a survey. Comput. Netw. **54**, 2787–2805 (2010)
3. Top 49 tools internet of things. https://www.profitbricks.com/. Accessed 4 May 2019
4. Savaglio, C., Fortino, G.: Autonomic and cognitive architectures for the internet of things. In: Di Fatta, G., Fortino, G., Li, W., Pathan, M., Stahl, F., Guerrieri, A. (eds.) IDCS 2015. LNCS, vol. 9258, pp. 39–47. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23237-9_5
5. Abdur Razzaque, M., et al.: Middleware for internet of things: a survey. IEEE Internet Things J. **3**(1), 70–95 (2016)
6. Kim, J., et al.: M2M service platforms: survey, issues, and enabling technologies. IEEE Commun. Surv. Tutorials **16**(1), 61–76 (2013)
7. Ngu, A., et al.: IoT middleware: a survey on issues and enabling technologies. IEEE Internet Things J. **3**(4), 1–20 (2017)
8. Cruz, D., et al.: A reference model for internet of things middleware. IEEE Internet Things J. **5**(2), 871–883 (2018)
9. Singh, M., et al.: Secure MQTT for internet of things. In: Tomar, G. (ed.) 5th International Conference on Communication Systems and Network Technologies, pp. 746–751. IEEE, Gwalior (2015). https://doi.org/10.1109/CSNT.2015.16

10. Zhu, Q., et al.: IoT gateway: bridging wireless sensor networks into Internet of things. In: IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, pp. 347–352. IEEE, Hong kong (2010). https://doi.org/10.1109/EUC.2010.58
11. Mobius Architecture. http://developers.iotocean.org/archives/module/mobius. Accessed 4 May 2019
12. Key Capability of the Kaa IoT platform. https://www.kaaproject.org/platform/. Accessed 4 May 2019
13. Kaa Architecture. https://kaaproject.github.io/kaa/docs/v0.10.0/Architecture-overview/. Accessed 4 May 2019
14. Sitewhere Architecture. https://github.com/sitewhere/sitewhere. Accessed 4 May 2019
15. DeviceHive Architecture. https://devicehive.com/. Accessed 4 May 2019
16. Fiware Architecture. https://www.fiware.org/. Accessed 4 May 2019
17. Um, C., et al.: Industrial device monitoring and control system based on oneM2M for edge computing. In: Pal, N.R. (ed.) 2018 IEEE Symposium Series on Computational Intelligence, pp. 1528–1533. IEEE, Bangalore (2018). https://doi.org/10.1109/SSCI.2018.8628736
18. Cho, H., et al.: Implementation and performance analysis of power and cost-reduced OPC UA gateway for industrial IoT platforms. In: 2018 28th International Telecommunication Networks and Applications Conference (ITNAC), pp. 1–3. IEEE, Sydney (2018). https://doi.org/10.1109/atnac.2018.8615377
19. Ericsson, A.B.: The Internet of Things forecast. https://www.ericsson.com/en/mobilityreport/internet-of-things-forecast/. Accessed 15 Mar 2019