Integration of Robot and IIoT over the OPC Unified Architecture

Hsien-I Lin† and Yu-Che Hwang

Graduate Institute of Automation Technology, National Taipei University of Technology
Taipei, Taiwan
sofin@ntut.edu.tw†

Abstract: Cloud technology uses the main function of IoT (Internet of Things) devices to quickly transfer huge amounts of data because data must be transmitted efficiently and efficiently among equipments. IoT enables smart devices to collect and share data without manual assistance. Compared to IoT, IIoT (Industrial IoT) technology is used for industrial purposes, but common industrial communication protocols such as Modbus TCP and Profinet only provide real-time data and cannot provide safe and reliable historical data. The OPC unified architecture (OPC UA) overcomes the above shortcomings and changes the entire industrial communication protocol and resolves the issue of object-oriented concepts. Other IoT protocols such as RTI DDS and MQTT do not emphasize the integration of industrial equipments. In this paper, we provide a solution enabling the smart robotic arm (Staubli) to convert its original communication protocol (SOAP/WSDL) into the role of an OPC UA server. This solution is successfully integrated in ERP/MES manufacturing production systems. The proposed method can help basic IoT robots be seamlessly embedded in ERP/MES manufacturing systems. We demonstrated how the OPC UA cloud computing platform worked. When a PCB color image was uploaded from the client, the OPC UA server processed it by the Gaussian blur and Canny filters and then returned the processed image to the client in a gray-scale image.

Keywords: IoT, Industrial IoT, OPC UA, robot.

1. INTRODUCTION

The German Institute of Electrical, Electronics and Information Technology has also released the pioneering Industry 4.0[1] road map to create a fully digitalized smart manufacturing production system with large number of network technologies, software technologies, the Internet of Things, cloud computing, and big data. Although the overall structure of Industry 4.0 is still in the process of exploration, if it can be applied in various fields in the future, it will construct a new intelligent industrial system, through analysis of various huge amounts of data, to generate customers that can satisfy customers. Thus, we all know that the keys to Industry 4.0 are artificial intelligence and big data over cloud systems where the information must be transmitted efficiently to play a role. With the revolution of cloud technology in Industry 4.0, the operation of the manufacturing industry has also undergone tremendous changes. Research has pointed out that the cloud has many advantages over old IT systems, allowing the industry to easily review the quality of a factory or a single device. Industry 4.0 represents the combination of cyber-physical systems (CPS), industrial Internet of Things (IIoT), and cloud computing. In short, machines are connected to the network and belong to a smart factory that can visualize the entire production system.

This paper is supported in part by the Ministry of Science and Technology Grant MOST 105-2221-E-027-064-MY3. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Ministry of Science and Technology

1.1 Cloud system

Cloud applications bring a number of benefits to manufacturers, including the ability to audit the quality of a single device, the quality of all products across the globe, the replacement of individual plant processing systems with a cloud system, and the monitoring of all plant products through a single screen to provide control of supplier operations. Cloud products can increase processing efficiency and help product quality to become a competitive advantage. In addition, software-as-a-service (SaaS) can reduce manufacturer capital expenditures (CAPEX) into lower and predictable operating expenses, thereby reducing software cost. The community of cloud systems believes that once the cloud system is launched, the obstacles associated with cost, training, and security will be easily deducted by efficiency and consistent manufacturing quality. Thus, digital revolution is already happening, and the companies that adapt with these technologies will benefit the most. After the emergence of Cloud Computing in 2008, it has become the focus of the information industry. Major mainstream manufacturers are all involved in the field of cloud computing. Major software, hardware and systems vendors in Taiwan have also invested a lot of manpower and capital into the cloud computing market. Besides, Google, Amazon, IBM and Microsoft are constantly expanding and strengthening their cloud computing capabilities to attract different industries and vendors into the cloud world. However, the cloud services provided by them are different and use differently, which makes many software engineers inconvenient.

The most challenging technicians of cloud are (1) artificial intelligence and (2) massive data transmission tech-

nology, namely IoT[2] technology. Cloud computing actually is not a small web application. On the contrary, the idea of cloud computing is very different from the one of general applications. If the developer of the application cannot understand the difference between the cloud and the general application, it is easy to design a general application that uses only a little bit of resources in the cloud. The cloud environment is no longer just a server or two, but thousands of virtual machines. The network environment is a complex VLAN or multi-segment system in the data center. The database may be in a different network segment, and the storage environment is a decentralized architecture. The VM executed by the application may be different due to the collection of the virtual machine or the switching of the machine. The original session architecture cannot be operated in the cloud because the session on the VM environment will disappear if the VM is moved. Thus, the data cannot exist locally, but must exist in the remote storage such as the SQL server database.

In order to meet the scalable operational requirements of the cloud server, the application must support at least the following capabilities: (1) The application must support the capabilities of load balancing, especially the state management function. When the user is assigned to a different server by load balancer, the user status information can still be correctly obtained. The process will not be interrupted due to the different servers required; (2) the application must store the data in remote storage instead of the local machine to avoid data loss during server switching; (3) globalization, the source of the cloud application can be global, and the application can be freely deployed in the data center built by the cloud provider. If the application needs to support globalization. According to the needs of the enterprise, the cloud application must be able to quickly generate the application execution environment in different data centers and serve the local or the user in the shortest time.

1.2 IoT

The concept of IoT is used to collect and share data without human assistance. IoT devices include from smart thermostats of vehicle systems to biochip transponders of farm animals. Even people can have sensors connected to the Internet. IoT is not only beneficial to consumers, but also to industries. Thus, IIoT (Industrial Internet of Things) is coined for industrial purposes such as gas pumps to HVAC systems and factory machinery. The information exchange in industry is not brand new since they have been doing this for years. However, IIoT takes it to a whole new level as it provides systems and standards for universal interconnects over IP (Internet Protocol). IP is a protocol for exchanging information over the Internet, regardless of the type of device being exchanged. Many of networked industrial equipments are closed systems. Even though they communicate with other devices on their network and speak the same language over the same protocol and architecture, the interconnect potential of existing network infrastructure is greatly expanded with IP-enabled devices.

IIoT is more than just simple devices that can be turned on and off remotely. IoT enable devices to have their own local logic and processing, exchange information with each other, and send and receive messages and commands over the network. IIoT brings the potential to leverage machine learning and big data technologies to bring automation to a new level of independent performance. Consider sensors that have collected data such as pressure, temperature, vibration, flow, etc., but now data can be combined with data from other systems and collected, analyzed and exchanged in new ways. IIoT applications include basic system monitoring and control, as well as analysis to change system and service operations. IIoT experts indicated that these technologies will greatly improve quality control, sustainable maintenance, and green practices, and significantly improve supply chain management and efficiency.

In this paper, we provide a solution to enable different robots to hook on cloud systems through IIoT. Then, robots can be integrated in ERP systems seamlessly. We implemented this solution on a Staubli robotic arm that uses (SOAP/WSDL)communication protocol and made it exchange information with a cloud system through OPC UA. The paper is structured as follows. We begin at reviewing Microsoft Windows Azure could platform and OPC UA and SOAP protocols in Section II and the implementation of the proposed solution is then presented in Section III. Section IV concludes the paper.

2. OPC UA/SOAP IIOT PROTOCOL

2.1 OPC UA

OPC UA generally has security mechanisms and uses encryption to specifically build or reside in so-called "no-man areas" where data must cross firewalls, platforms, and security barriers. The basic approach is that the data is encrypted directly from the source and is therefore protected as it passes through the network, making it easier for other devices to share messages because they already protect the data. Specifically speaking, security moves to edge devices rather than relying on network devices. Compared to common industrial communication protocols such as Modbus, TCP and, Profinet that only provide real-time data and cannot provide safe and reliable historical data, OPC UA [3] overcomes the above shortcomings and changes the entire industrial communication. The OPC UA allows for easy modeling of objects, which can be viewed as a piece of data, a process, a system or an entire plant. Objects consist of variables and methods, so clients accessing objects can choose to get the data they need and then provide an extremely rich set of useful information for a variety of applications. Table 1 shows the function comparison between OPC UA and Modbus

TCP/ProfiBus.

Table 1 Function comparison between OPC UA and Modbus TCP/ProfiBus.

	OPC /OPC UA	Modbus TCP /ProfiBus
Variable TagName / register definition	TagName	Register
Object oriented function	Yes	No
Realtime data acquisition	Yes	Yes
Alarm event log	Yes	No
Historian data log	Yes	No
Security	Yes	No

After briefing of the existing capabilities of OPC UA, we discuss how these features make it a perfect fit for IIoT standard[4]. Even though many related organizations have developed a complete communication architecture based on the characteristics of IoT, such as WSDL/SOAP, OPCUA, DDS[5], MQTT[6], ROS, etc., with the strong involvement of governments in the development of Industry 4.0, the possibility of other IIoT technology development has been suppressed. Thus, OPC UA fills this gap by acting as an integrator, making it very easy to move data between these systems.

2.2 SOAP

SOAP (simple object access protocol) is a communication protocol and encoding format based on XML[7] format for communication between applications. The original idea was Microsoft and user-level software, which has been evolved over several generations. The current specification is SOAP 1.2, but the version SOAP 1.1 is more common. SOAP is widely regarded as the backbone of a new generation of cross-platform, cross-language distributed computing applications and called Web services. Axis[8] is essentially a SOAP engine constructing SOAP processors such as clients, servers, gateways, etc. The current version of Axis is a framework written in Java, and the C/C++ client is already developed. But Axis is not just a SOAP engine and it also includes a simple stand alone server, a server can be plugged into a servlet engine such as Tomcat, a web services description language (WSDL), and a tool for monitoring TCP/IP packets. Axis is the third generation of Apache SOAP (starting with IBM as "SOAP4J"). At the end of 2000, Apache's SOAP V2 committers began discussing how to make the engine more flexible, configurable, and capable of handling both SOAP and XML forthcoming protocol specifications from the W3C.

2.3 gSOAP

gSOAP[9] is a C and C++ software development kit for SOAP/XML Web services and XML for universal data binding. The gSOAP tool supports integration of C/C++ code, embedded systems, and real-time software to share computing resources and information with other SOAP applications. SOAP / XML Applications can be deployed on different platforms, programming languages, and completely different organizations behind a firewall. The gSOAP tool is also very popular for implementing C and C++ integration of XML data. This means that the application native data structure can be automatically encoded in XML without the need to write a conversion code. These tools also generate XML schemas for data binding, so external applications can consume XML data based on the schema. Other features of gSOAP are integrated high-speed, architecture, concrete XML parsing and validation, HTTP basic and digital authentication, NTLM authentication, proxy authentication, XML compression, internationalization/localization support, etc.

2.4 CLR

The common language runtime (CLR) [10]is the name chosen by Microsoft for their .NET[11] virtual machines. It is Microsoft's implementation of the common language architecture (CLI), which defines an environment in which code is executed. The CLR implements a bytecode called the common intermediate language. The CLR is implemented on Microsoft's Windows operating system. A list of implementation versions of this specification can be found by examining the common language architecture. Some of these versions are implemented in non-Windows operating systems. The main functions of the CLR are basic category library support, memory management, thread management, garbage collection, security, type checker, exception management, debug management, middle code (MSIL) to machine code (Native) compilation, class loading, etc. Developers write programs in higher-level programming languages. The compiler then compiles the code into Microsoft's relay language (MSIL). When executed, the CLR converts the MSIL code into the native code of the operating system. Figure 1 shows the cross-platform soap client/server development environment.

2.5 OPC UA vs. DDS

RTI Data Distribution Service (DDS)[12] is a high performance network middleware application. Its publish-subscribe communication model allows distributed units to share data without having to consider the actual physical location or architecture. It is multi-platform and used in combination with more than 70 hardware, operating systems and compilers. Table 2 shows the function comparison among OPC UA, ROS, DDS, and MQTT.

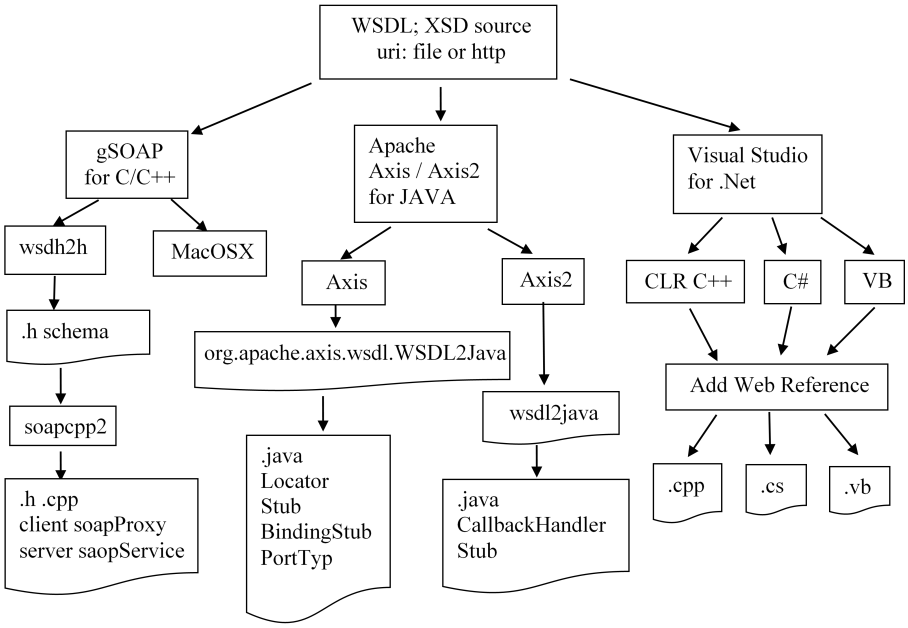

Fig. 1 Cross-platform soap client/server development environment.

Table 2 Function comparison among OPC UA, ROS, DDS, and MQTT.

	OPC UA	ROS	DDS	MQTT
Communication	TCP	UDP TCP	UDP, TCP, SHM	TCP
Patterns	RPC, Pub/Sub	RPC, Pub/Sub	(RPC), Pub/Sub	Pub/Sub
QoS	No	No	Yes	Yes
Authentication	User, PKI	(Mac)	PKI	User, PKI
Encryption	Yes	No	Yes	Yes
Std. API	Yes	No	Yes	No
Semantic Data	Yes	No	No	No

3. IMPLEMENTATION

3.1 Windows Azure Web Application for Smart Manufacturing

This example illustrates that all the arm process parameters or track paths of the intelligent robot in the R&D center or the production site can be uploaded to the cloud platform. The cloud application can be viewed and downloaded to the production site arm controller through the cloud application. The utilization rate and any production debugging and adjustment parameters can also be uploaded to the cloud platform as the management logistics data to form the infrastructure of IoT in Industry 4.0. The ultimate goal is to reach the realm of the unmanned (light-off) factory. Figure 2 shows the process of uploading the robot trajectory and related parameters to the cloud system. The web page on the cloud is ready for previewing and downloading the robot parameters. Figure 3 shows the webpage on the cloud system.

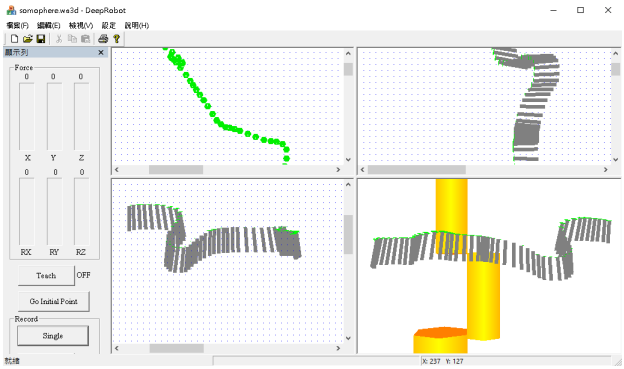

Fig. 2 Robot trajectory and related parameters are uploaded to the cloud system.

3.2 Robot over OPC UA

We used FreeOPC UA C++ as the basis for the implementation. FreeOPCUA contains Class descriptions: server class, client class, high level functions and node class, and subscription class. The server class is to create the namespace, populate your server address space using use the get_root() or get_objects() to get node objects and get_event_object() to fire events, and start the server. The client class is to connect and browse address space, define server address space using use the get_root() or get_objects() to get node objects, and get_event_object() to fire events. The high level functions and node class

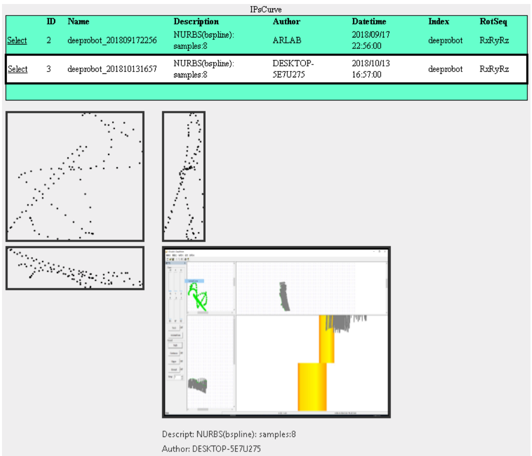

Fig. 3 Cloud application on the webpage for previewing and downloading the robot parameters.

are to access node attribute, browse, and populate address space. The subscription class is to subscribe object returned by server or client objects. The object represents a subscription to an opc ua server. The operation procedure is shown in Fig. 4.

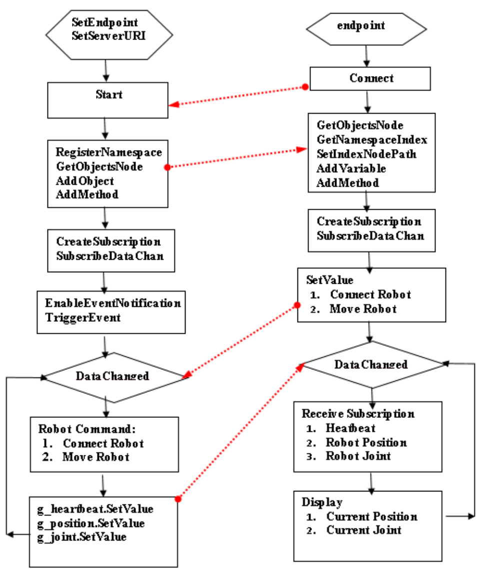

Fig. 4 Operation procedure of the free OPC UA.

The robot system was an Staubli TX60 that uses WSDL/SOAP. We implemented an OPC UA server/client integration program that was responsible for converting the Staubli TX40 communication WSDL/SOAP format to OPC UA. Then, the manufacturing MES system issued the instructions to command the robot the move. Our system converted the instruction into CNC G-Code like command as follows

CONNECT STAUBLI IP: 6.12.24.166
Move,200,0,000,0,180,0
Move,500,0,000,0,180,0
STOP

The program experiment was run over Staubli CS8 emulator version 7.3. After the emulator was executed, it was set to the automatic mode with a speed of 25%. At the beginning, we started the OPC UA server and then the OPC UA client. Figures 5 and 6 show the OPC UA server and client programs, respectively. When the instruction "m_move.SetValue(Variant("M,200,0,000,0,180,0"))" was set, the robot in the emulator was moved to the specified position.

```
[2019-08-28 13:53:33.451] [server] [info] Root node is: Node(ns=0;i=84;)
[2019-08-28 13:53:33.454] [server] [info] Children are:
[2019-08-28 13:53:33.477] [server] [info]    Node(ns=0;i=85;)
[2019-08-28 13:53:33.480] [server] [info]    Node(ns=0;i=86;)
[2019-08-28 13:53:33.481] [server] [info]    Node(ns=0;i=87;)
[2019-08-28 13:53:33.483] [server] [info] Ctrl-C to exit
```

Fig. 5 OPC UA server program.

```
• client.Connect(endpoint);
[2019-08-28 13:34:15.811] [client] [info] Connecting to : opc.tcp://127.0.0.1:4840/freeopcua/server/
[2019-08-28 13:34:15.957] [client] [info] ua_client        | creating session ...
[2019-08-28 13:34:15.961] [client] [info] ua_client        | create session OK
[2019-08-28 13:34:15.961] [client] [info] ua_client        | activating session ...
[2019-08-28 13:34:15.964] [client] [info] ua_client        | activate session OK
[2019-08-28 13:34:15.966] [client] [info] keep_alive_thread    | starting
[2019-08-28 13:34:15.966] [client] [info] Root node is : Node(ns=0;i=84;)
[2019-08-28 13:34:15.967] [client] [info] Child of objects node are :
[2019-08-28 13:34:15.974] [client] [info]    Node(ns=0;i=2253;)
[2019-08-28 13:34:15.974] [client] [info]    Node(ns=2;i=99;)
[2019-08-28 13:34:15.975] [client] [info] NamespaceArray is :
[2019-08-28 13:34:15.978] [client] [info]    http://opcfoundation.org/UA/
[2019-08-28 13:34:15.978] [client] [info]    http://freeopcua.github.io
[2019-08-28 13:34:15.978] [client] [info]    deeprobotics IIOT
[2019-08-28 13:34:15.998] [client] [info] got node : Node(ns=0;i=2258;)
• m_workshop.SetValue(Variant("CONNECT STAUBLI 6.12.24.166"));
• m_move.SetValue(Variant("M,200,0,000,0,180,0"));
Node:Node(ns=2;i=2002;), Value:158
Node:Node(ns=2;i=2004;), Value X:500, Y:1.06977e-14, Z:1.38778e-14, RX:-1.18302e-15, RY:-180, RZ:-5.17114e-23
Node:Node(ns=2;i=2005;), Value J1:-2.29244, J2:47.9978, J3:65.5917, J4:-2.18628e-07, J5:66.4105, J6:-2.29244
Node:Node(ns=2;i=2002;), Value:159
Node:Node(ns=2;i=2004;), Value X:495.56, Y:-0.000232811, Z:0.00100907, RX:-2.23264e-12, RY:-180, RZ:1.91907e-12
Node:Node(ns=2;i=2005;), Value J1:-2.31302, J2:47.1216, J3:67.1294, J4:-2.21724e-07, J5:65.749, J6:-2.31302
Node:Node(ns=2;i=2004;), Value X:456.773, Y:-0.00245883, Z:0.0107396, RX:-2.95856e-11, RY:-180, RZ:1.97854e-11
Node:Node(ns=2;i=2005;), Value J1:-2.50983, J2:39.97, J3:79.4426, J4:-2.51826e-07, J5:60.5874, J6:-2.50983
Node:Node(ns=2;i=2002;), Value:160
Node:Node(ns=2;i=2004;), Value X:406.781, Y:-0.00591146, Z:0.0259288, RX:2.3194e-09, RY:-180, RZ:4.45398e-11
Node:Node(ns=2;i=2005;), Value J1:-2.819, J2:31.5698, J3:93.2604, J4:-3.00171e-07, J5:55.1699, J6:-2.819
Node:Node(ns=2;i=2004;), Value X:303.184, Y:-0.00903749, Z:0.0392296, RX:-6.61672e-11, RY:-180, RZ:4.41079e-11
Node:Node(ns=2;i=2005;), Value J1:-3.78406, J2:14.9186, J3:117.77, J4:-4.49801e-07, J5:47.311, J6:-3.78406
Node:Node(ns=2;i=2004;), Value X:251.019, Y:-0.00565701, Z:0.0241143, RX:-1.00065e-08, RY:-180, RZ:1.33913e-11
Node:Node(ns=2;i=2005;), Value J1:-4.5712, J2:5.98046, J3:128.809, J4:-5.62519e-07, J5:45.2102, J6:-4.5712
Node:Node(ns=2;i=2004;), Value X:207.81, Y:-0.00103556, Z:0.00436822, RX:-9.66949e-09, RY:-180, RZ:1.49072e-13
Node:Node(ns=2;i=2005;), Value J1:-5.52308, J2:-2.33281, J3:137.477, J4:-6.83512e-07, J5:44.8555, J6:-5.52308
Node:Node(ns=2;i=2002;), Value:161
Node:Node(ns=2;i=2004;), Value X:200.026, Y:-1.02411e-06, Z:4.20059e-06, RX:1.18413e-10, RY:-180, RZ:-3.97052e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73842, J2:-3.97009, J3:138.996, J4:-7.08608e-07, J5:44.9736, J6:-5.73842
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2002;), Value:162
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2002;), Value:163
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
Node:Node(ns=2;i=2004;), Value X:200, Y:-2.31277e-15, Z:-2.77556e-14, RX:-4.73133e-16, RY:-180, RZ:1.59028e-15
Node:Node(ns=2;i=2005;), Value J1:-5.73917, J2:-3.97567, J3:139.002, J4:-7.08694e-07, J5:44.9741, J6:-5.73917
```

Fig. 6 OPC UA client program.

3.3 OPC UA Cloud Server Computing

The cloud system also provides some computing capabilities. A robot is on the client side and can register the cloud transport method and then bring in parameters (arguments) for the server to perform operations through the OPC UA protocol. For example, a PCB image is captured by the camera on the robot, and it is sent to the OPC UA server to remove high frequency noises. The cloud operations are Gaussian blur and Canny filters. After that, the post-processed gray-scale image is sent back to the robot client side. Figure 7 shows the cloud computing by

Gaussian blur and Canny filters. The program procedure is shown in Fig. 8.

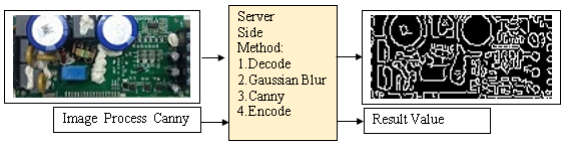

Fig. 7 Cloud computing by Gaussian blur and Canny filters.

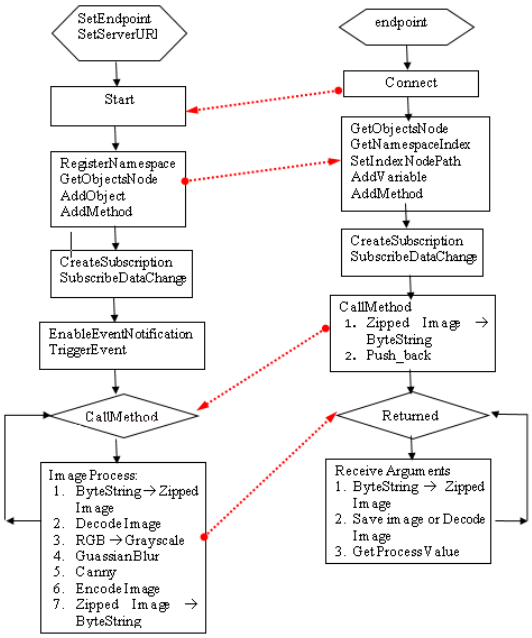

Fig. 8 Program procedure of Cloud computing.

4. CONCLUSIONS

Clouds are used between artificial intelligence and huge amounts of data, and data must be transmitted efficiently to function. Thus, IIoT brings the potential to leverage machine learning and big data technologies to bring automation to a new level of independent performance. Compared with common industrial communication protocols such as Modbus TCP and Profinet, OPC UA has more advantages. In this paper, we implemented an IIoT over a robotic arm through the OPC UA communication architecture. In the work, we converted the robot communication protocol SOAP into OPC UA and then the robot could be instructed through the OPC UA server/client. Finally, we demonstrated an example which showed how the OPC UA cloud computing platform worked. By using the cloud platform, we were able to easily process a PCB image by the Gaussian blur and Canny filters.

REFERENCES

[1] S. Wang, J. Wan, D. Li, and C. Zhang, "Implementing smart factory of industrie 4.0: an outlook," *International Journal of Distributed Sensor Networks*, vol. 12, no. 1, p. 3159805, 2016.

[2] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, vol. 1, no. 2011, pp. 1–11, 2011.

[3] A. Veichtlbauer, M. Ortmayer, and T. Heistracher, "Opc ua integration for field devices," in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*. IEEE, 2017, pp. 419–424.

[4] 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)*. IEEE, 2017, pp. 1–6.

[5] P. Bellavista, A. Corradi, L. Foschini, and A. Pernafini, "Data distribution service (dds): A performance comparison of opensplice and rti implementations," in *2013 IEEE symposium on computers and communications (ISCC)*. IEEE, 2013, pp. 000 377–000 383.

[6] T. Yokotani and Y. Sasaki, "Comparison with http and mqtt on required network resources for iot," in *2016 international conference on control, electronics, renewable energy and communications (ICCEREC)*. IEEE, 2016, pp. 1–6.

[7] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, "Extensible markup language (xml) 1.0," 2000.

[8] M. Barbos and E. Pop, "Web services for ubiquitous mobile device applications," in *International Conferences on Advanced Service Computing*, 2010.

[9] R. A. Van Engelen and K. A. Gallivan, "The gsoap toolkit for web services and peer-to-peer computing networks," in *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*. IEEE, 2002, pp. 128–128.

[10] E. Meijer and J. Gough, "Technical overview of the common language runtime," *language*, vol. 29, p. 7, 2001.

[11] J. Galloway, P. Haack, B. Wilson, and K. S. Allen, *Professional ASP. NET MVC 4*. John Wiley & Sons, 2012.

[12] T. White, M. N. Johnstone, and M. Peacock, "An investigation into some security issues in the dds messaging protocol," 2017.