ATLANTIS PRESS
Advances in Computer Science Research (ACRS), volume 54
International Conference on Computer Networks and Communication Technology (CNCT2016)

Industrial Big Data Platform Based on Open Source Software

Wen YANG[1,2], Syed Naeem Haider[1], Jian-hong ZOU [1] and Qian-chuan ZHAO[1,*]

[1]Center for Intelligent and Networked Systems, Department of Automation, Tsinghua University, Beijing, 100084, China

[2]Key Laboratory of Space Launching Site Reliability Technology, Haikou 570100, China

whutyw@126.com, haider.gillani12@gmail.com, zoujh13@mails.tsinghua.edu.cn, zhaoqc@tsinghua.edu.cn

Keywords: Industrial Big Data, Open source, Industry 4.0, Industrial Internet.

Abstract. Deep integration of industrial system and information technology triggered the fourth industrial revolution-Industry 4.0 which based on cyber physical system and Industrial Big Data. Although many researchers have discussed the basic concept of industry 4.0 and Industrial Big Data, as we known no literatures about how to design and develop an Industrial Big Data platform. Business solutions are generally not open to the public, so little is known about how to achieve it. Open source big data tools have widely used in the Internet field, but it is not clear how these tools are used in the industrial field. We focus on how to use open source big data tools to build a big data platform for industrial systems and a systematically designed framework is proposed including data acquisition, transmission, processing, storage and visualization.

Introduction

The information technology as key drivers of productivity, has been gradually penetrating and changing our society in every way since the birth of the Internet. In recent years, Internet is gradually integrated with traditional manufacturing and industrial control to achieve intelligent manufacturing and individual character manufacturing. This triggered the birth of industry 4.0 [1,2] which refers to the fourth industrial revolution that was first announced by Germany as one of the key initiatives of its high-tech strategy in 2011 and aim to meet the challenges of the production process complexity and uncertainty and eventually realize intelligent factory. Industry 4.0 can be defined as a data driven intelligent manufacturing which draws together Cyber-Physical Systems [3], the Internet of Things [4]. In order to provide useful insight to the factory management and gain correct content, data has to be processed with advanced tools (analytics and algorithms) to generate meaningful information. In 2013, GE first proposed Industrial Internet which is the result of the integration of the global industrial system and advanced computing, analysis, induction and Internet [5]. Industrial Big Data generally refers to a large amount of diversified time series generated at a high speed by industrial devices [6], and is the core of both industry 4.0 and industrial Internet. The term emerged along with the concept of industrial Internet and Industry 4.0, and diverges from Big Data, which is more popular in information technology. Industrial Big Data created by industrial devices might hold more potential business values [7]. A research conducted by Accenture and General Electric forecasted that the values created by Industrial Internet of Things and Industrial Big Data could be worth $500 billion by 2020 [8].

Industrial Big Data has captured the interest of the research community due to its widespread application value. In [9], authors proposed a unified 5-level architecture as a guideline for implementation of CPS which is architecture for Industry 4.0 based manufacturing system. The service innovation and smart analytics for the big data environment are discussed in [10]. From the IT-perspective, the requirements of Industry 4.0 are analyzed in [11] and it shows how the requirements can be matched to the capabilities of Big Data software solutions. Big data software and process of content analysis were proposed, the specific software tools and platforms, however have not been discussed in depth. [12] presented a survey of the open source technologies that support big data processing in a real-time/near real-time fashion, including their system architectures and platforms. In [13], authors provided a systematic evaluation of various big data platforms based on characteristics that are pertinent to big data analytics in order to aid the users with better understanding about the suitability of these platforms for different problem scenarios. At present, the literatures are concerned with the architecture and framework, however as it has no explanations to the realization of these architecture, and no one knows how these architectures are implemented. Some business solutions for Industrial Big Data have been presented. Since GE released the industrial internet, the industrial Internet platform Predix [14] is presented and allows developers to connect with the user, and it can be used to develop customized data analysis and solutions in accordance with the needs of users on the platform. IMS and NI have co-developed the Watchdog Agent [15] based on LabView. The user can solve the problem of industrial application based on this platform. However, these systems and platforms are provided by the Business Companies and unable to understand the technical details.

In software field, open source technology develops rapidly and benefited from extensive technical support provided by technical experts and organization. The non-profit Apache Software Foundation (ASF) is a group of developers that have created and developed many famous open source projects, such as the Apache Web server and the Axis Web services framework. Open source Big Data tools have played a central role in Internet field, and have become the foundation and core of all big data platform. Due to the maturity and efficiency of the open source big data tools, we believe they are also applicable to sensor data, and used to build big data platforms for Industrial system.

The major contributions of this paper is that a real time data acquisition scheme based on publish/subscribe pattern and message architecture for large-scale sensor nodes is proposed. Then on the base of these, an Industrial Big Data platform is brought forward base on open source software, and its system architecture is shown in Fig. 1. Compared with the traditional industrial monitoring system, it has advantages in data transmission, storage, processing and visualization which are shown in Table 1.

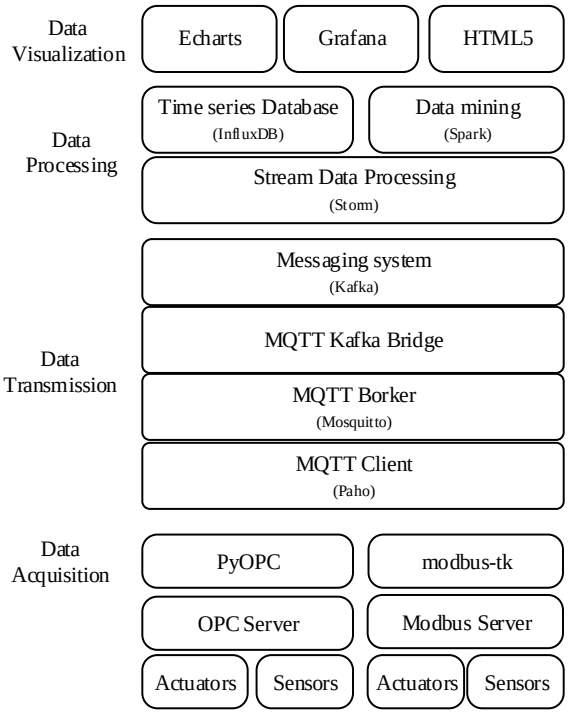

Figure 1. Architecture of Industrial Big Data platform based on the open source software.

Table 1. Comparison between our platform and the existing monitoring platform

Feature	Existing industrial monitoring system	Our industry big data platform
Data acquisition mode	pull data	publish-subscribe
QoS	No	QoS 0: At most once delivery QoS 1: At least once delivery QoS 2: Exactly once delivery
Acquisition and processing	synchronization	asynchronous
datagram structure	Byte array	JSON
storage data type	numerical values	numerical value, text, image
scalability of storage	Scale up	Scale out
reliability	difficult	Distributed fault-tolerant
Data mining	No	Integrated mainstream machine learning algorithm
Visualization	2D	2D/3D, Map, hierarchical display

The remainder of the paper is organized as follows. In section 2 data acquisition is described. Data transmission program based on publish/subscribe models proposed in section 3. In section 4 the others parts of platform are discussed which are pre-processing module, data storage module, data mining analysis module and data visualization. Section 5 concludes with discussion and implication to future work.

Data Acquisition

Data Acquisition Interface

For some technology and business reasons, devices produced by different manufacturers have different data interface. Among them the most common are OPC, Modbus and DNP3. Although these data interfaces are detailed in the open specification, it is not easy to develop the software that supports these data interfaces, especially in the case of ensuring security and reliable.

There are several OPC frameworks are available that introduce simple building of client and server applications. However, most of these frameworks are not freely available or are based on Microsoft's .Net framework and are therefore platform dependent. Due to these limitations, the PyOPC framework [16] was developed, which fully implements the OPC XML-DA standard, and provides the features such as open source, Multi-platform capable (Microsoft Windows, Linux, Mac OS X and others), extensible and reusable, enabling developers to build OPC XML-DA based applications in an easy way. In literally less than 4 lines of code we have the foundation for a complete read data from an OPC server.

```
//import the OpenOPC module
1. import OpenOPC
//create OpenOPC instance ()
2. opc = OpenOPC.client()
//connect to OPC Server
3. opc.connect('opc server name')
//reading a single item
4. opc.read('Tag name')
```

The Modbus.org site offers links to third-party sites for Modbus users' and developers' convenience. The developer's main purpose in making these tools available is to make the development of testing tools as easy as possible. There are several open source library for Modbus, including pymodbus, MinimalModbus and Modbus-tk. They are all based on the python language which is flexible and easy to learn.

Data Structure

Sensors data is a typical time series data, which have onto many points, one for each discrete sample of the metric. In order to distinguish and describe the measurement information of different sensors, the following information should be considered carefully.

a) Measurement. It is used to describe a sensor's measurement object or physical process, such as a room.

b) Tags. Tags are an optional part of data structure and they are useful for storing commonly-queried metadata.

c) Fields. The key-value pair data structure that records metadata and the actual data value. Field keys are strings and they store metadata. Field values are the actual data; they can be strings, floats, integers, or Booleans.

d) Time. The date and time associated with a point. The consistence of the data acquisition time is important for data analysis.

In order to describe the collected data, we define a data structure for sensor data based on JSON (JavaScript Object Notation) which is a lightweight data-interchange format. JSON is easy for humans to read and write and easy for machines to parse and generate simultaneously. It is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON ideal data-interchange language.

The measurement data of a temperature sensor T1 can be described as follows and this data structure is compatible with time series database InfluxDB that will be discussed in Section 4.1. This means the data can be writing to InfluxDB directly.

```
[
   {
"measurement": "room temperature",
"time": "2016-10-10T23:00:00Z”,
"tags": { “sensor id”: “IoT://city/building1/floor1/room2/T1”},
"fields": {"value": 0.64}
   }
]
```

Data Transmission

As the characteristics of the industrial system itself, industrial data transmission needs to meet several requirements.

1) *Real time*. In industrial field, data transmission delay may cause hidden problems cannot be found in time, so it is necessary to ensure the low latency of data transmission.

2) *Data distribution*. According to different application needs, there are one or more data consumers and it requires that the same data can be transmitted to different recipients.

3) *QoS*. As data packet transmission may be delayed or drop out, the transmission mechanism should provide QoS service.

4) *Security*. There are a number of threats for data transmission, such as (a) device could be compromised, (b) communication could be intercepted, altered, re-routed or disclosed (c) denial of service attacks. So security mechanisms such as authentication and authorization of sender and receiver, integrity and privacy of data packet should be considered.

5) *Lightweight communication protocol*. Due to industrial device calculation and communication resources are limited; the lightweight communication protocol is needed.

Transmission Protocol

Data generated by the mass sensors and devices should be transferred to the storage and analytics systems. General Internet protocols like HTTP and REST are often too heavy weights for the sensor, in terms of overheads, processing and memory requirements. At present, a variety of communication protocols have been proposed for the application of Internet of things, such as CoAP, AMQP, MQTT and XMPP. MQTT and CoAP are specifically designed with the above requirements in mind. Both of them use a client-server model, run on IP networks, provide asynchronous (none blocking) mode of operation, and have support for QoS and security. MQTT is a messaging protocol that was introduced by IBM in 2013, which aims at connecting sensors and actuators with remote server. It utilizes the publish/subscribe pattern to provide transition flexibility and simplicity of implementation. MQTT is a very light protocol; its minimum length is only two bytes and suitable for resource constrained devices that use unreliable or low bandwidth links. It is built on top of the TCP and deliver messages with three levels QoS including at most once delivery, at least once delivery and exactly once delivery. As a messages system, MQTT simply consist of three components, subscriber, publisher and broker. Device such as sensors or physical objects acts as a publisher of information. A data processing or storage server would register as a subscriber for specific topics in order for it to be informed by the broker when

publishers publish topics of interest. After that, the publisher transmits the information to the interested entities (subscribers) through the broker. Fig. 2 shows the architecture of MQTT. MQTT has strong open source support: brokers (e.g., Mosquitto), clients (e.g., Paho) and client tools are all available in a wide variety of programming languages that are shown in Table 2.

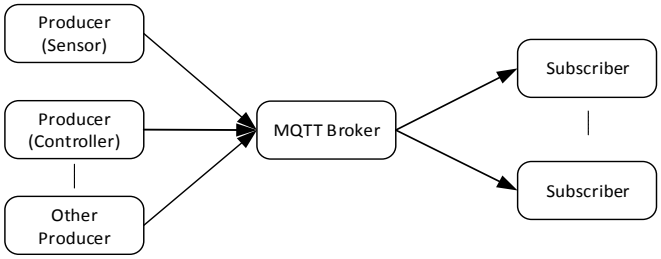

Figure 2. The architecture of MQTT

Table 2. Open source class library for MQTT

Broker	Language
Mosquitto	C, C++, Python
Apache Apollo	Java, python
RabbitMQ	Java, .NET, Ruby, Python, PHP, JavaScript
ActiveMQ	Java, C, C++, C#, Ruby, Perl, Python, PHP
mosca	JavaScript
emqttd	Erlang

Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol versions 3.1 and 3.1.1. Discarding more complex solutions, especially with regard to the installation and configuration, we can build a broker just execute a command.

linux:mosquitto -c mosquitto.conf

Windows: mosquitto.exe

Mosquitto provides SSL support for encrypted network connections and authentication. Secure messaging using SSL or TLS is supported.

The Eclipse Paho project provides open-source client implementations of MQTT and support Java, C/C++, Python, JavaScript, etc. The Paho Python Client provides a client class with support for both MQTT v3.1 and v3.1.1 on Python 2.7 or 3.x. It also provides some helper functions to make publishing messages to an MQTT server very straightforward. It is easy to write a client that connects to the broker and send/receives data. That is all the sender need to do to load the class library and then execute the send data command.

import paho.mqtt.publish as publish

publish.single("topic", "msg", hostname="test")

As a producer, Paho can be easily integrated with PyOPC or Modbus-tk, and send sensor data to the remote server through MQTT protocol.

Message Architecture

MQTT does not provide any buffering mechanism and it does not scale very well for large system. An intermediate messaging system like RabbitMQ or Apache Kafka can overcome this problem. Add such a messaging system between the MQTT broker and the analytics system can improve the overall system performance as well as provides easy scalability.

Apache Kafka already has MQTT support. The MqttKafkaBridge [17] which consumes MQTT messages and republishes them on Kafka on the same topic and makes integration effortless.

Data Processing and Visualization

The data received from the Kafka may need further processing, like data storage, data processing and visualization.

Data Storage

Comparing with the Internet Big Data, besides some common "4V" characters, Industrial Big Data is a typical time series data and have the characteristics of time series data. A series of sensor data is identified by a source name or ID (for example: host ID) and a metric name and consists of a sequence of {timestamp, value} measurements ordered by timestamp, where the timestamp is probably a high-precision UNIX timestamp and the value is a floating-point number. Therefore, the database system used in the Industrial Big Data should consider meet the requirements of both Internet Big Data and time series data.

1) *Scalability*. As the rapid development of the Internet of things and machine run in 7 × 24 hour. The time series data generated by machines (sensors) will be much more than the data generated by the human. So the database should have the ability of scale out to meet the needs of large-scale data storage.

2) *Real time*. Massive streams of sensor data generate by sensors (tens of thousands per minute or even hundreds of thousands of records per second) which must be written to database with high speed and be queried immediately.

Obviously, the traditional relational database has been unable to meet the above requirements. Luckily, there are some open source time series databases that are suitable for Industrial Big Data storage.

InfluxDB [18] is an open source time series database designed for high-performance writes and compact disk storage. It built from the ground up to handle high write and query loads. It is written entirely in Go and compiles into a single binary with no external dependencies that make it a great choice for working with time series data. InfluxDB provide simple, high performing write and query HTTP(S) APIs and plug-in support for other data ingestion protocols such as Graphite. OpenTSDB [19] is another distributed, scalable time series database written on top of HBase. The data stored in OpenTSDB is metric as a unit, and metric is a monitor such as a sensor monitoring point. OpenTSDB supports data storage forever, that is, the saved data will not be deleted automatically, and the original data will be saved (some of the monitoring system will be stored longer before the data aggregation). OpenTSDB use AsyncHbase instead of HTable which HBase owned and use thread safe, non-blocking, asynchronous, multi-thread concurrent API of HBase, in the high concurrency and high throughput, it can get better results.

Data Processing

Many industrial applications are depend on real time processing such as fault detection and early warning. How to quickly find out the abnormal data from massive real time data is the difficulty and challenge faced by the Industrial Big Data processing.

Apache Storm is a free and open source distributed, reliable, and fault-tolerant real-time computation system for processing streams of large volumes of data. Storm

makes it easy to reliably process unbounded streams of data, doing for real time processing what Hadoop did for batch processing. Storm has many use cases: real time analytics, online machine learning, continuous computation, distributed RPC, ETL, and more. Spout and bolt are two components that work together for streaming data processing. Spout is a source of stream and it passes the data to a component called bolt.

Storm and Kafka naturally complement each other, and their powerful cooperation enables real-time streaming analytics for Industrial Big Data. Storm-Kafka is an open source extended component of Storm and provides core Storm and Trident spout implementations for consuming data from Apache Kafka 0.8.x [20]. For both Trident and core Storm spout implementations, it use a BrokerHost interface that tracks Kafka broker host to partition mapping and kafka config that controls some Kafka related parameters.

Data Preprocessing Bolt

Some reasons lead to the raw data does not meet the requirements of the data analysis. For example, sensors crash occasionally and network disruption can lead to data loss. Environmental interference can raise outliers. Before data analysis, the data should be preprocessed to improve the quality of data. These preprocessing are deployed a signal preprocessing bolt including data integrity test, format standardization, abnormal data removal and scale transformation.

Device Monitoring and Alarm Management Bolt

In order to ensure the safety of the operation of industrial devices, the health status of the device should be evaluated in real time. A device monitoring bolt can be designed for it, and the same type of device is detected by a bolt. False negatives and false positives are difficulties and challenges faced by industrial monitoring and alarm management. Comprehensive industrial system structure model and data association analysis can help reduce the incidence of false positives and false negatives. In the Storm, several bolts can be developed and specially used for alarm analysis.

Writing Database Bolt

The industrial data needs to be written to the database for historical archival purposes as well as for avoiding unnecessary repeat computations. Instead of writing data directly to a database through an MQTT client, an alternative way is to write it through Storm. Data are written to database with an independent bolt facilitates the isolation of data streams and optimal management. InfluxDB-java [21] is a Java Client library which provide the interface of the operation InfluxDB and can be integrated with Storm easily.

Visualization

Outstanding data visualization can help people to understand the meaning of data. Industrial Big Data visualization mainly has two requirements. The first is data visualization which is to communicate information clearly and efficiently to users via the statistical graphics, plots, information graphics charts selected. The second is topography structure visualization which reflects the position and connection between the devices and the sensors.

Data Visualization

Grafana [22] is a leading open source application for visualizing large-scale measurement data, and is commonly used for visualizing time series data including industrial sensors, home automation, weather, and process control. It has many

outstanding characteristics including rich graphing, graph styling, various dashboards and provide rich support for InfluxDB and OpenTSDB.

ThingSpeak [23] is a versatile open source platform and API that allows developers and app designers to gather data from sensors and other sources, analyze and visualize it. Freeboard is a purpose-built visualization tool for the Internet of Things. It allows you to create a dashboard full of different widgets and immediately share it with anyone.

ECharts [24] is another free, powerful charting and visualization library offering an easy way of adding intuitive, interactive, and highly customizable charts to your commercial products. It is written in pure JavaScript and based on zrender, which is a whole new lightweight canvas library.

Topology Visualization

HTML5 introduces elements and attributes that reflect typical usage on modern websites. In addition to specifying markup, HTML5 specifies scripting application programming interfaces (APIs) that can be used with JavaScript. There are also new APIs such as canvas, editable content, drag-and-drop, web messaging and geo location, etc. So it is an ideal visualization tool of topology structure for complex industrial system.

Gephi [25] is the leading visualization and exploration software for all kinds of graphs and networks. Gephi is open source and free, and is widely used in link analysis, social network analysis and biological network analysis.

Conclusions

How to mine the valuable Industrial Big Data and utilizing it is the key to the successful implementation of the industry 4.0 and industrial Internet. However, industrial data platform and tool is the precondition and means of analyzing and using Industrial Big Data. This paper analyzes the characteristics of Industrial Big Data and features of the current Internet Big Data tools and platforms. On this basis, an industry big data platform based on open source software is put forward. It has advantages of open source, reliability, flexibility and meets the requirements of Industrial Big Data acquisition, transmission, processing, storage and visualization. For future work, we will build a prototype system based on the proposed scheme.

Acknowledgement

This work is supported in part by open project of Key Laboratory of Space Launching Site Reliability Technology, NSFC Grant. No. 61425027 and special fund of Suzhou-Tsinghua innovation Leading Action.

References

[1] M. Herman, T. Pentek, and B. Otto, “Design Principles for Industrie 4.0,” Dortm. TU, 2015.

[2] J. Lee, B. Bagheri, and H.-A. Kao, “Recent Advances and Trends of Cyber-Physical Systems and Big Data Analytics in Industrial Informatics,” in Int. Conference on Industrial Informatics (INDIN) 2014.

[3] E. A. Lee, "Cyber physical systems: Design challenges," in Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on, 2008(363-369).

[4] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," IEEE Commun. Surv. Tutor., vol. 17, no. 4, (2015)2347-2376.

[5] P. C. Evans and M. Annunziata, "Industrial internet: Pushing the boundaries of minds and machines," Gen. Electr., p.21, 2012.

[6] G. I. Platforms, The Rise of Industrial Big Data. Whitepaper, 2012.

[7] J. Kelly, The industrial internet and big data analytics: Opportunities and challenges. 2013.

[8] Information on http://www.computerweekly.com/opinion/Big-data-to-unlock-value-from-the-Industrial-Internet-of-Things

[9] H.-A. Kao, W. Jin, D. Siegel, and J. Lee, "A Cyber Physical Interface for Automation Systems—Methodology and Examples," Machines, vol. 3, no. 2, (2015) 93-106.

[10] J. Lee, H.-A. Kao, and S. Yang, "Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment," Procedia CIRP, vol. 16, (2014)3-8.

[11] P. Gölzer, P. Cato, and M. Amberg, "Data Processing Requirements of Industry 4.0-Use Cases for Big Data Applications," Data Process., 2015.

[12] X. Liu, N. Iftikhar, and X. Xie, "Survey of Real-time Processing Systems for Big Data," in Proceedings of the 18th International Database Engineering & Applications Symposium, New York, NY, USA, (2014)356-361.

[13] D. Singh and C. K. Reddy, "A survey on platforms for big data analytics," J. Big Data, vol. 2, no. 1, (2014)1-20.

[14] Information on https://www.predix.io/

[15] Information on http://www.imscenter.net/

[16] Information on http://pyopc.sourceforge.net

[17] Information on https://github.com/jacklund/mqttKafkaBridge

[18] Information on https://influxdata.com/

[19] Information on http://opentsdb.net/

[20] Information on https://github.com/apache/storm/tree/master/external/storm-kafka.

[21] Information on https://github.com/influxdata/influxdb-java.

[22] Information on http://grafana.org/

[23] Information on https://thingspeak.com/

[24] Information on http://echarts.baidu.com/

[25] Information on https://gephi.org/