IEEJ TRANSACTIONS ON ELECTRICAL AND ELECTRONIC ENGINEERING
IEEJ Trans 2018
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI:10.1002/tee.22684

Letter

Maintenance of Business Machines with Edge and Cloud Collaboration

Yoji Yamato[a], Non-member
Hiroki Kumazaki, Non-member

Recently, internet of things (IoT) technologies have progressed, but IoT maintenance applications are not widespread in Japan yet because of insufficient analyses of real-time situations and the high costs of configuring failure detection rules and collecting sensing data. In this paper, using lambda architecture concept, we propose a maintenance platform on which edge nodes analyze sensing data, detect anomalies and extract a new detection rule in real time, and a cloud orders maintenance automatically.

Keywords: lambda architecture; predictive maintenance; Jubatus; IoT; cloud computing; Industrie 4.0

Received 4 November 2017; Revised 8 October 2017

1. Introduction

Recently, IoT [1,2] and cloud [3–8] technologies have progressed. Manufacturing and maintenance are hot areas for IoT applications, and IoT platforms have also been created to develop and operate IoT applications effectively. However, existing IoT platforms mainly target the visualization of statuses, and these platforms are insufficient to accelerate the maintenance field.

We target business machine maintenance in factories. To resolve existing problems, we propose a maintenance platform with lambda architecture [9] in which edge nodes analyze sensor data, detect anomalies and extract a new detection rule in real time, and a cloud orders maintenance automatically; it also analyzes whole data collected by batch process in detail and updates learning model of edge nodes to improve accuracy.

2. Problems of Existing Technologies

NTT DOCOMO and GE released an IoT solution in 2015 that provides GE's wireless router Orbit with NTT DOCOMO's communication module. Users can collect operation statuses of facilities by setting Orbit. AWS IoT is a platform to integrate Amazon Web Services for IoT applications.

However, there are three problems regarding maintenance.

The first is that the site situation is not analyzed sufficiently in real time. Docomo's IoT applications mainly visualize applications of collected data by batch processing, and real-time actions such as repair parts orders are not considered.

The second is that the cost of configuring rules to detect failures is high. To detect failures from IoT data, most analysis applications need rules or thresholds. A data scientist extracts these rules, but this task needs much technical skills.

The third is that the cost of collecting sensor data is high. In AWS IoT, to analyze IoT data, users need to collect all data in a cloud and require a network for machines in multiple regions.

3. Maintenance Platform Using Lambda Architecture

To resolve existing problems, we propose a maintenance platform using the lambda architecture concept. Lambda architecture was architecture [9] proposed by Marz that provides results of both the speed layer and batch layer for users.

[a] Correspondence to: Yoji Yamato. E-mail: yamato.yoji@lab.ntt.co.jp; yoji.yamato.wa@hco.ntt.co.jp

NTT Software Innovation Center, NTT Corporation, 3-9-11 Midori-cho, Musashino-shi Tokyo 180-8585, Japan

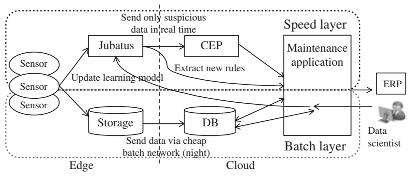

Fig. 1. Lambda architecture adoption for maintenance

Figure 1 shows our adoption image of lambda architecture. Storage for data and Jubatus are deployed on edges such as factories. Unlike the batch processing of Hadoop, Jubatus is a machine learning framework suitable for sequential processing of stream data [10]. A Complex Event Processing (CEP) engine and DB are deployed on a cloud.

In the speed layer, Jubatus in edge nodes analyzes sensor stream data, detects anomaly specious data and then sends the data to CEP in a cloud. Jubatus also extracts a new anomaly detection rule by a machine learning algorithm and sends it to a cloud. A maintenance application is called and receives suspicious data by CEP, and it predicts failures.

In the batch layer, the raw data of edges are sent to a cloud and stored in DB through low-cost methods such as night transfer. A maintenance application predicts failure to analyze data in detail, which does not require prompt actions. Moreover, a data scientist may analyze raw data to extract an improved machine learning model and distribute it to edges periodically.

These ideas can resolve existing problems.

Jubatus in the speed layer can detect anomalies in real time, and prompt actions such as repair parts orders can be carried out. Because Jubatus in the speed layer can use an anomaly detection (e.g., LOF [Local Outlier Factor]) algorithm that detects differences from usual operation values, we can extract new rules based on site environment data and reflect these data to a cloud maintenance application with low cost. Rules from the detailed analysis in the batch layer can be applied to sites to improve the analysis accuracy of each site. Because our platform sends only suspicious data to a cloud in the speed layer, we can reduce network costs. Whole data is sent by lower-cost methods, such as night transfer or bike transfer.

Figure 2 shows proposed platform architecture.

(i) A sensor is attached to a business machine, and sensing data is stored in an edge. Jubatus in an edge node analyzes stream data through an anomaly detection algorithm. When stream data are different from usual operation, Jubatus detects anomalies; then, only anomaly data are sent to a cloud. Jubatus also can extract new anomaly rules by machine learning, and then, those

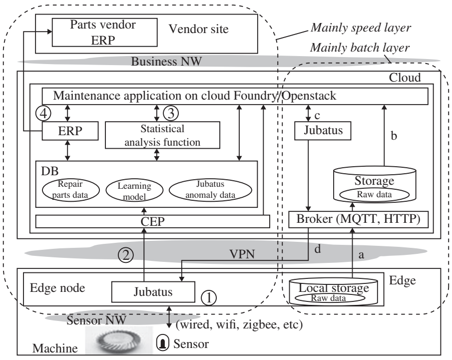

Fig. 2. Proposed maintenance platform architecture

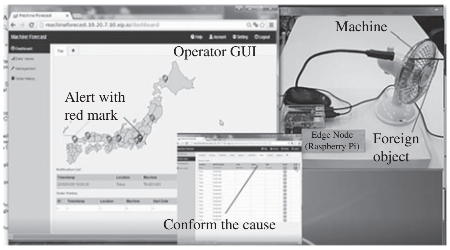

Fig. 3. An example application for maintenance

rules are also sent to a cloud. (ii) Data are sent to a cloud by MQTT or HTTP. A CEP engine receives anomaly data and calls a maintenance application and stores data to DB. (iii) A maintenance application predicts a failure, such as what part is failed and how much failure probability is within a certain period using statistical analysis function. (iv) Based on predicted failures, a maintenance application orders ERP for repair parts arrangement using coordination technologies [11–18].

In parallel with speed layer operations, learning models in edges and rules in a cloud are updated bidirectionally. Rules extracted from Jubatus in step 1 are merged with failure detection rules in a maintenance application. In the batch layer, (i) raw data are collected to a cloud in batch processes. (ii) A maintenance application analyzes these data for two objectives. One is for failure prediction, the same as step 3, to analyze data in detail, which need no prompt action. The other is to update the machine learning model of Jubatus. (iii) A data scientist may analyze raw data and update a model using Jubatus in a cloud to improve analysis accuracy. (iv) An updated model is distributed to edges by a cloud [19–21]. These bidirectional updates improve the analysis accuracy both in edges and the cloud.

Ref. [2] is our previous work, and it targeted only the speed layer. It cannot improve analysis accuracy because it cannot detect problems through detailed analysis in the batch layer, and it cannot update the Jubatus models from the batch layer results.

4. Implemented Example Application

Here, we demonstrate a sample application on the platform. It is a simple use case that detects fan failures by analyzing the sound data of microphones attached to factory machines.

Figure 3 (right side) shows an edge in a factory. The edge node is Raspberry Pi, and Jubatus is installed on it. Machine behavior data are collected by microphone, and Jubatus analyzes the stream frequency data sequentially. When we insert a foreign object into a fan in Fig. 3 (right side), Jubatus detects an anomaly of strange frequency in the speed layer. Raspberry Pi sends anomaly data, including judge results, to a cloud. A cloud maintenance application analyzes anomaly data in detail, predicts a failure and shows an alert with a red mark in Fig. 3 (left side). When an operator clicks the red mark, a page with predicted failure causes pops up. An operator can also issue a maintenance order.

5. Conclusion

This paper proposed a maintenance platform to accelerate the automatic maintenance of business machines using the lambda architecture concept and demonstrated an example application.

References

(1) Yamato Y, Fukumoto Y, Kumazaki H. Proposal of shoplifting prevention service using image analysis and ERP check. *IEEJ Transactions on Electrical and Electronic Engineering* 2017; **12**(S1):141–145.

(2) Yamato Y, Fukumoto Y, Kumazaki H. Predictive maintenance platform with sound stream analysis in edges. *Journal of Information Processing* 2017; **25**:317–320.

(3) Yamato Y, Nishizawa Y, Muroi M, Tanaka K. Development of resource management server for production IaaS services based on OpenStack. *Journal of Information Processing* 2015; **23**:58–66.

(4) Yamato Y. Optimum application deployment technology for heterogeneous IaaS cloud. *Journal of Information Processing* 2017; **25**:56–58.

(5) Yamato Y. Cloud storage application area of HDD-SSD hybrid storage, distributed storage and HDD storage. *IEEJ Transactions on Electrical and Electronic Engineering* 2016; **11**:674–675.

(6) Yamato Y. OpenStack hypervisor, container and baremetal servers performance comparison. *IEICE ComEX* 2015; **4**:228–232.

(7) Yamato Y, Shigematsu N, Miura N. Evaluation of agile software development method for carrier cloud service platform development. *IEICE Transactions on Information and Systems* 2014; **E97-D**:2959–2962.

(8) Yamato Y, Katsuragi S, Nagao S, Miura N. Software maintenance evaluation of agile software development method based on OpenStack. *IEICE Transactions on Information and Systems* 2015; **E98-D**:1377–1380.

(9) Marz N, Warren J. *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*. Manning Publications Co.: New York; 2015.

(10) Hido S, Tokui S, Oda S. Jubatus. *NIPS 2013 Workshop on Big Learning*, 2013.

(11) Sunaga H, Takemoto M, Yamato Y, Yokohata Y, Nakano Y, Hamada M. Ubiquitous life creation through service composition technologies. *WTC2006*, 2006.

(12) Yamato Y, Nakano Y, Sunaga H. *Context-Aware Service Composition and Change-Over Using BPEL Engine and Semantic Web*. InTech Open Access Publisher: Rijeka, Croatia; 2010.

(13) Yamato Y, Ohnishi H, Sunaga H. Development of service control server for web-telecom coordination service. *IEEE ICWS 2008*, 2008; 600–607.

(14) Yokohata Y, Yamato Y, Takemoto M, Tanaka E, Nishiki K, Okuda T, Teranishi Y. Context-aware content-provision service for shopping malls based on ubiquitous service-oriented network framework and authentication and access control agent framework. *IEEE CCNC 2006*, 2006.

(15) Yamato Y, Nakano Y, Sunaga H. Study and evaluation of context-aware service composition and change-over using BPEL engine and semantic web techniques. *IEEE CCNC 2008*, 2008; 863–867.

(16) Yokohata Y, Yamato Y, Takemoto M, Sunaga H. Service composition architecture for programmability and flexibility in ubiquitous communication networks. *IEEE SAINTW'06*, 2006; 145–148.

(17) Yamato Y, Ohnishi H, Sunaga H. Study of service processing agent for context-aware service coordination. *IEEE SCC 2008*, 2008; 275–282.

(18) Sunaga H, Yamato Y, Ohnishi H, Kaneko M, Iio M, Hirano M. Service delivery platform architecture for the next-generation network. *ICIN2008*, Session 9-A, October 2008.

(19) Yamato Y. Performance-aware server architecture recommendation and automatic performance verification technology on IaaS cloud. In *Service Oriented Computing and Applications*. Springer: Heidelberg, Germany; 2016.

(20) Yamato Y. Server selection, configuration and reconfiguration technology for IaaS cloud with multiple server types. *Journal of Network and Systems Management* 2017; **26**:339–360.

(21) Yamato Y. Automatic system test technology of user virtual machine software patch on IaaS cloud. *IEEJ Transactions on Electrical and Electronic Engineering* 2015; **10**:165–167.