TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 19, No. 4, August 2021, pp. 1226~1233
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v19i4.19012

An adaptive IoT architecture using combination of concept-drift and dynamic software product line engineering

I Made Murwantara, Pujianto Yugopuspito
Informatics Department, Universitas Pelita Harapan, Indonesia

Article Info

Article history:

Received Aug 26, 2020
Revised Jan 20, 2021
Accepted Jan 31, 2021

Keywords:

Adaptive
Concept drift
Internet of things
Software architecture
Software product line

ABSTRACT

Internet of things (IoT) architecture needs to adapt autonomously to the environment and operational to maintain their supreme services. One common problem in the IoT architecture is to manage the reliability of data services, such as sensors' data, that only sending data to the collector via gateway. If there is a disruption of services, then it is not easy to manage the system reliability. To this, an adaptive environment which is based on software reconfiguration creates a great challenge to provide better services. In this work, the software product line engineering (SPLE) reconfigures the edge devices via rules and software architecture. To identify disruption of data services which can be detected based on anomaly and truncated data. Our work makes use of concept drift to provide a recommendation to the system manager. This is important to avoid misconfiguration in the system We demonstrate our method using an open-source internet of things portal system that integrated to a cluster of sensors which is attached to specific gateway before the data are collected into a cloud storage for further processes. In identifying drifting data, the adaptive sliding window (ADWIN) method outperforms the Page-Hinkley (PH) with more selective identification and sensitive reading.

Corresponding Author:

Pujianto Yugopuspito
Department of Informatics
Universitas Pelita Harapan
M. H Thamrin Boulevard St. 1101, Lippo Village, Tangerang, Banten, 15811 Indonesia
Email: yugopuspito@uph.edu

1. INTRODUCTION

In recent years, Internet of things (IoT) has become a common technology to support human daily activities for almost any living sectors. In the year 2019, at least, 4.81 billion of IoT units had been installed, globally [1]. And it is predicted more than $389 billions [1] will be spent for IoT system within three years. The huge demand of IoT has also increased, since smart technology for transportation, vehicle and residence, mostly, use IoT technology to provide high end services. IoT services has also become a primary demand to have better improvement for public services such as health system [2], intelligent transportation [3], traffic management [4], civil works [5] and many other sectors. Where in their operation, IoT produces vast amount of data that retrieved from a high number of sensors and devices attached to the system. The incoming data are transferred from many sensors into, mostly, cloud data collector before analysed for further use.

In general, the architecture of IoT has capability to provide services to convey one on making their decision based on data reading from sensors and other inputs for specific purpose with the help of machine learning and big data technology. Machine learning processes that using real-time input data should have stable stream as it has different behaviour compares to static data analysis. To this, an unstable flow of data may lead into wrong decision which will impact to end-users. As an example, flooding emergency system depends on the water flow measurement that obtained from groups of sensors installed along the monitoring river. During rainy season, some sensors have problem to send their information, the emergency warning system may generate wrong recommendation to the operator. Another critical example is the application of IoT for healthcare system, such as intensive care unit (ICU) that rely on accurate measurement. If the reading of combination of sensors, such as cardiac monitoring and air ventilator, has slightly intermittent data reading, then an alarm of fatality might be raised by the system. This will cause error on health critical analysis. As a result, a mechanism to identify the data behaviour should be developed to address the aforementioned problems.

Despite of IoT has been widely implemented in many sectors, some hesitations on how accurate and significant the data that will be processed, especially, using machine learning still questionable. This work focuses on how to build an adaptive IoT architecture that concern on data drifting. This is important, since sensors [6] may have distortion regarding of their data because of their wrong installation, positioning or malfunction. For example, sensor temperature manages to read the change of temperature on normal condition. However, the application of sensor with small distances to a heat generator such as water boiler will lead into wrong information. Adaptive system provides a kind of autonomous capability to self-adjustment on differences in order for a system to have a reliable and good performance [7]. An adaptive system such as in cloud computing [8] can be realized as an attenuation to specific condition. As an example, in Murwantara *et al.* [8] make use of transition model to explicate how the change should be going on based on certain condition.

In this work, we focused on data as the sensing elements whether an architecture should have new reconfiguration or not. Some methods using combination of dynamic software product line engineering (DSPLE) [9], [10] and applied data science [11] has convey some research to use data as the feedback for autonomous system. In concept and data drifting technique [12]-[14] the change of pattern and data has also provided the change of information in real-time. This paper has three contributions as follow,

- In detecting anomaly of data transaction, we need to monitor all data transaction in real-time and asses it on the fly with current existing data to find any drifting or change pattern, which may indicate anomaly information. When comparing data in real-time some computation may increase the cost of CPU operation. To this we need to find a method that has capability to predict whether the data may drift and when it will occur. In this work, we make use of concept-data drift method that uses machine learning approach to have better understanding of data pattern.
- Recommendation of which software package should be included or excluded to have better operation based on the change of data pattern is quite a challenging work. As a result, we need to figure out of how to automate [15] the software reconfiguration mechanism. We concern that the implementation of IoT may have hundreds or thousands of sensors and IoT gateway, which not easy to manage the reconfiguration in a manual way. This work takes advantage of common automation in Software Engineering environment for and in Cloud computing system where an automation tool, such as Ansible [16], helps multiple remote system management in a massive scale.
- To manage change of an IoT architecture, we need to monitor and identify simultaneously not only the Thinger condition but also the data pattern that stream from group of sensors as a reference to have better system performance. As a result, an adaptive architecture has its importance to be included as a success factor. In this work, we adopted the dynamic software product line engineering approach to have an adaptive IoT architecture from the software engineering point of view. A mechanism to support the dynamic change of software components configuration should be explicated in this paper using concept-drift approach as the sensing system services & monitoring.

In an adaptive system, a real-time adjustment is the most critical point of work. To change software architecture in a real-time condition, we adopted the transition model introduced in [17] which give us benefit to manage the reconfiguration.

2. BACKGROUND

2.1. Concept drift

Concept drift is a statistical approach using supervised online learning to identify anomaly on data stream by analyzing between input and output data [14] over time. In a dynamic environment, data simultaneously change overtime in an expected or unexpected, and usually occurs with environmental events [18]. Data drift degrades machine learning model accuracy over time. In handling concept drift, mining and analysis of streaming data are the most critical. One of the methods to cope with this problem is the adaptive sliding window (ADWIN) algorithm [19]. This method has efficient variation via dynamic changing to the length of a data window which has good performance in predicting stream data. ADWIN approach monitoring the stream data using sliding windows with different sizes and width, rather than statistically observed. Another method to handle concept drift is Hoeffding tree (HT) [20] to construct decision tree that performs well using constant memory and time per instance. In [21] has enhanced the Hoeffding Tree capability by analyzing the consistency of error rate. Another monitoring change detection is Page-Hinkley (PH) [22] test using sequential analysis technique that has efficient detection that is designed to detect change in a Gaussian signal approach.

2.2. Dynamic software product line engineering

DSPLE augments the software product line engineering (SPLE) to have adaptive behavior to cope with the existing concurrent or real-time problems. Where in SPLE, the member of product line harvests new software products using existing artefacts as core reference by configuring software components in high quality, short time and less cost [23]. For further knowledge on SPLE, we can read from this article [24]. The advantage of reusability method in SPLE has also been implemented in IoT development [25], [26]. In this work, we focus on managing a DSPLE approach to engage with dynamic software architecture for IoT.

3. RESEARCH METHOD

This work has adopted the ThingerIO [27] portal system to have the cloud services to collect and provides an application programming interface (API) as a gateway for further analytical process. As shown in Figure 1, we extend the IoT framework with SPLE [28] approach to have a kind of adaptive mechanism. Our framework has five layers with specific functionalities as follow,

- The lowest layer, edge devices, manages the very basic architecture with sensors and devices to collect data via sensor and execute command using actuator. In this work, we have used Raspberry Pi 4 as the main controller in this layer.
- The next is edge layer that provides gateway for groups of Thinger (devices) to send data into the highest hierarchy of this architecture. As shown in Figure 2, this layer only passing through the data without any detection or modification, and its main service is to maintain connectivity between edge devices layer to cloud services.

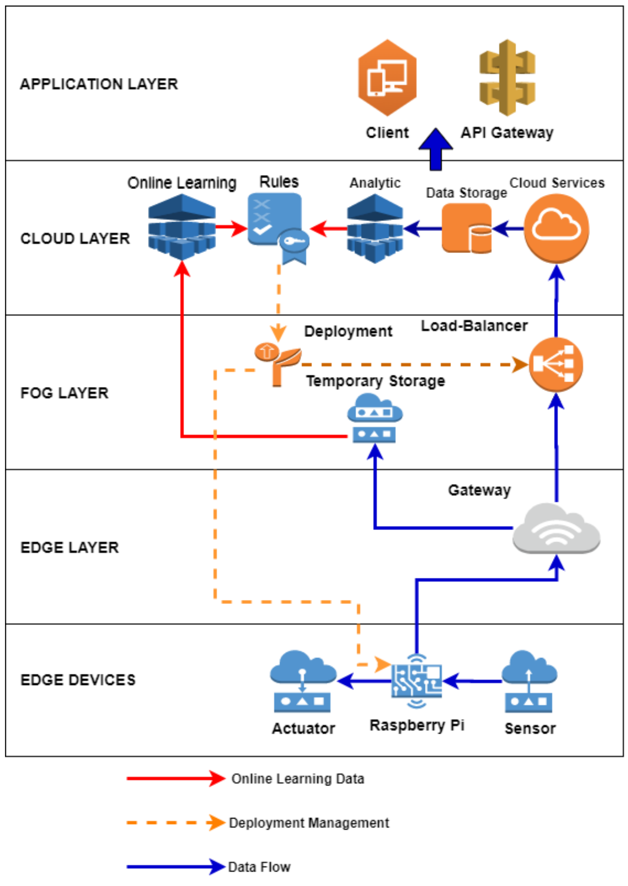

Figure 1. Adaptive IoT architecture with DSPLE

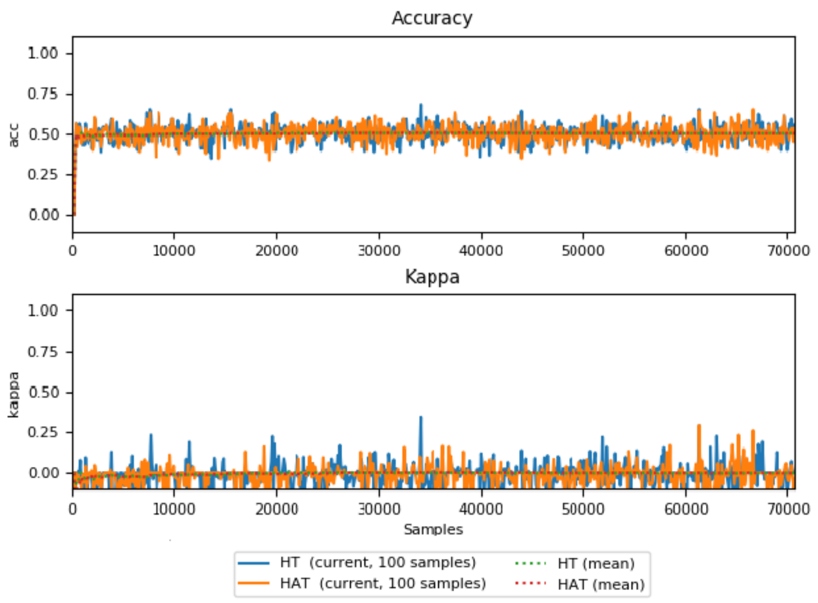

Figure 2. Hoeffding tree (HT) and hoeffding adaptive tree (HAT) results

- In fog layer, there are three functionalities. First, it manages the load-balancer to reduce intermittent in collecting data and relaying through cloud services. Second, it maintains temporary storage as input for online learning analysis in the highest layer. And, the last one, this layer also working as the reconfiguration manager via deployment manager to manage dynamic software configuration to the load-balancer, in the same layer, and Raspberry Pi in the edge devices layer.
- Cloud layer is the most critical operation in this framework. It manages the cloud services and storage. And, it also performs analytic operation to maintain the reliability and availability of data. This layer accomplishes two analytic computation to have batch and online processing. In batch processing, the data have been accumulated in the data storage which consists of sensors data. In online analytical processing, this layer read and predict the data pattern and behaviour to make sure its reliability. To have a recommendation whether to change the software configuration or not, this layer reading the batch data to know the data consistency and latency to observe communication and traffic which may have a result to reconfigure the load-balancer. Another information from online learning is to identify the problem that may occurs because of edge devices layer failures. We make use of Hoeffding tree (HT) and Hoeffding adaptive tree (HAT) to have information on data pattern and the performance of this approach. To predict the data drifting, we implement the ADWIN and Page-Hinkley model which give us estimation on when the data have problem, how bad is the data inconsistency, and also to know when the data transmission will get back to become normal. This information important because we need to make a decision whether this problem should be addressed into execution of software reconfiguration to specific devices.
- In application layer, it provides the client integration to the system to have data and prediction via web services. Two services supported in this framework via application programming interface (API) gateway, where user seamlessly accessing the data. And, client connectivity via direct client-server connection.

4. RESULTS AND ANALYSIS

Our adaptive IoT architecture, as shown in Figure 1, comprises of five layers with adaptive management exist on two layers, which are cloud layer and fog layer. In identifying data drifting, we make use of concept drift to observe the incident. So that, we can predict and create an action plan to adjust the system into normal operation. It is worth to note that our work concern on IoT software architecture, not the hardware system. In our experiment, the Raspberry Pi connected to sensors and actuators which sends data to Cloud data collector based on ThingerIO IoT architecture [27]. We measure and collect the data, then implement the data drifting identification model, on the fly, as online learning to estimate which action should be done according to our transition model. We have collected the data for one month, which gathered more than 100,000 records of sensors data.

In this work, concept drift is used to create a recommendation of action plan that enable an adaptive mechanism in an IoT software architecture. To this, the concept drift via online learning corroborates in conveying the software reconfiguration. In predicting the possibility of data drifting, the ADWIN method performs with high sensitivity on stream change pattern with distribution monitoring, and Page-Hinkley algorithm using sequential analysis [29]. As shown in Table 1, the result of ADWIN as detector, and Table 2 using Page-Hinkley to detect data drifting. In using ADWIN model, detector has identified more data which can be seen on the interval index. Page-Hinkley model identified a smaller number of index interval that convey to less data drifting.

To evaluate the recommendation model, we make use Hoeffding Tree and Hoeffding Adaptive Model to gain more information, as depicted in Figure 2. In accuracy, both model HT and HAT show 0.4993 and 0.4983, slightly similar value, where HAT has capability to identify drifting while adapting to the change of workload [30]. Where Kappa, as shown in Figure 2, is a sensitive measure to quantify the predictive performance which related to the streaming classifier because the number of instances may change in the data stream. It compares between the detected accuracy with the estimated accuracy. In performance comparison, kappa results for HAT slightly lower and more stable than HT, which have spikes in several time frame. For further information about Kappa can be read in [31].

Table 1. Excerpt on ADWIN Model

Index	Data
95	22.41
127	26.46
159	20
191	37.19
223	41.80
255	44.56
287	49.08
319	51.26
351	54.55
383	54.78

Table 2. Excerpt on Page-Hinkley model

Index	Data
64	19.48
107	23.56
144	28.62
177	34.95
216	40.45
262	45.13
303	49.70
353	54.35
1521	26.05
1562	30.12

In handling the transition change between conditions and recommendation, we adopted the transition diagram to convey us on building a DSPLE mechanism. As shown in Figure 3, the configuration of our IoT architecture is divided into three (3) set. The first is high feature when all sensors are sending data in normal data transaction. In medium feature, only half numbers of sensors data are sent by the Raspberry Pi and disable the sensor with readability problem. The Low Feature only have minimal number of sensors included in the configuration.

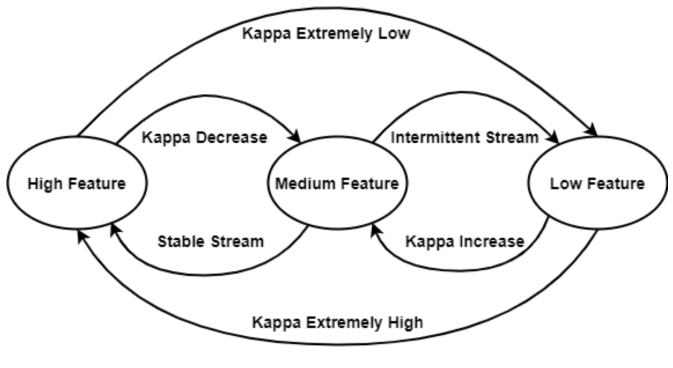

Figure 3. Transition Diagram

To manage the reconfiguration process, we developed a distributed automated IoT configuration model as shown in Figure 4. During normal operation, the analytic engine always checking the data pattern using batch and real-time processing to identify any disruption or failure based on the data pattern. This is important because we cannot put more workload into the edge devices to avoid problems on data transmission. Gateway provides interaction between rule/policy manager to the edge devices with minimal data engagement. However, when the analytic engine provides a recommendation to reconfigure the edge devices, the rule/policy manager will send an action plan to change the device configuration (Raspberry Pi Sensor configuration). It follows by the deployment manager (reconfiguration/ansible) to call the impacted edge devices to receive an execution order. The order is sent via SSH protocol by Ansible mechanism which has been prepared to receive order via deployment manager. As shown in Algorithm 1, the deployment manager send a request to disable/enable the GPIO port in Raspberry Pi (as edge devices). With this concept in mind, we can reconfigure the edge devices in any location as long as the system has connection to the deployment manager.

In this work, we still doing the improvement of the adaptive mechanism, especially the online learning that assess the data pattern. A stable interconnection is a must to have fully automation. We have tried to run on low bandwidth which is the typical of IoT environment. However, some enhancement should be made more precisely to avoid a runaway configuration, which usually occurs when the communication failed during the reconfiguration process. For example, in reconfiguring an edge devices, an acknowledgement message which indicate the stage of reconfiguration has been done by the devices should be reported to the deployment manager. If there is a problem with communication, then the deployment manager will not have any information regarding the reconfiguration process. Our proposed framework has leading us into knowledge on building an adaptive system for IoT system, can have an alternative, using DSPLE approach with concept-drift as control mechanism. It also has enlightened us on how important to concern about data drifting in an IoT architecture.

Algorithm 1. An Excerpt to Enable/Disable GPIO in Raspberry Pi via Ansible

```
 - name: enable/disable remote GPIO
command: "raspi-config nonint do_rgpio {{0 if raspi-config.enable_rgpio else 1}}"
when: "'enable_rgpio' in raspi-config and raspi-config. enable_rgpio!=raspi_rgpio_enabled"
tags:
- raspi
```

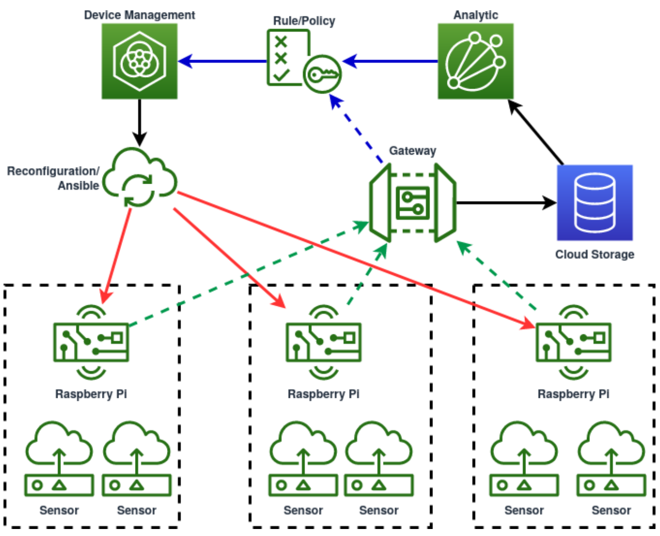

Figure 4. Adaptive IoT 2

5. CONCLUSION

We have proposed an adaptive IoT architecture that concern to the change of stream data behavior to manage a dynamic IoT system. This work adopted the dynamic software product line (DSPLE) approach to handle the change of architecture configuration. In sensing the data drifting from a sensor or feature, our technique makes use of concept drift to detect anomaly and misbehavior of data pattern. Despite of bandwidth limitation in most IoT infrastructure. We manage to create a dynamic IoT architecture which, autonomously, reconfigure the edge devices to have its condition of work into normal stage. Our work also makes light on alternative adaptive IoT architecture that is not rely only on devices monitoring but based on data pattern as indicator of problem within the ecosystem. Our results show that the adaptive sliding window (ADWIN) method outperforms the Page-Hinkley with more selective identification and sensitive on data reading. For future works, we would like to add a mechanism to be able to handle failure detection during a reconfiguration process with minimal supervision by the deployment manager. This is important to reduce data traffic inside the IoT ecosystem.

ACKNOWLEDGEMENTS

This work has been funded by the Research and Community Services (LPPM) Universitas Pelita Harapan grant No. P-011-FIK/I/2020. The infrastructure of this research is supported by research contract No 25/E1/Kpt/2020, 225/SP2H/LT/DRPM/2019, 100.ADD/LL3/PG/2020 Minister of Education and Culture of the Republic of Indonesia.

REFERENCES

[1] L. Goasduff, “Gartner Says 5.8 Billion Enterprise and Automotive IoT Endpoints Will Be in Use in 2020,” *Gartner Report 2019*, vol. 21, no. 1, pp. 1-9, 2020.

[2] A. Mahajan, G. Pottie, and W. Kaiser, “Transformation in Healthcare by Wearable Devices for Diagnostics and Guidance of Treatment,” *{ACM} Trans. Comput. Healthc.*, vol. 1, no. 1, pp. 1-12, Mar. 2020, doi: 10.1145/3361561.

[3] A. A. Brincat, F. Pacifici, S. Martinaglia, and F. Mazzola, “The Internet of Things for Intelligent Transportation Systems in Real Smart Cities Scenarios,” *2019 {IEEE} 5th World Forum on Internet of Things ({WF}-{IoT})*, Apr. 2019, doi: 10.1109/wf-iot.2019.8767247.

[4] D. Goel, S. Chaudhury, and H. Ghosh, “An {IoT} approach for context-aware smart traffic management using ontology,” *Proceedings of the International Conference on Web Intelligence*, Aug. 2017, doi: 10.1145/3106426.3106499.

[5] S. Jeong and K. Law, “An IoT Platform for Civil Infrastructure Monitoring,” *2018 {IEEE} 42nd Annual Computer Software and Applications Conference ({COMPSAC})*, Jul. 2018, doi: 10.1109/compsac.2018.00111.

[6] A. Gaddam, T. Wilkin, M. Angelova, and J. Gaddam, “Detecting sensor faults, anomalies and outliers in the internet of things: A survey on the challenges and solutions,” *Electron.*, vol. 9, no. 3, pp. 1–15, Mar. 2020, doi: 10.3390/electronics9030511.

[7] I. M. Murwantara, H. Tjahyadi, P. Yugopuspito, A. Aribowo, and I. A. Lazarusli, “Towards adaptive sensor-cloud for Internet of Things,” *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 16, no. 6, pp. 2771-2781, Dec. 2018, doi: 10.12928/TELKOMNIKA.v16i6.11557.

[8] I. M. Murwantara, B. Bordbar, and J. B. F. Filho, “A self-adaptive architecture with energy management in virtualized environments,” *Proceedings-2017 International Conference on Soft Computing, Intelligent System and Information Technology: Building Intelligence Through IOT and Big Data, ICSIIT 2017*. Sep. 2017, doi: 10.1109/ICSIIT.2017.18.

[9] M. Hinchey, S. Park, and K. Schmid, “Building dynamic software product lines,” *Computer (Long. Beach. Calif).*, vol. 45, no. 10, pp. 22-26, Oct. 2012, doi: 10.1109/MC.2012.332.

[10] A. M. Sharifloo, A. Metzger, and C. Quinton, “Learning and Evolution in Dynamic Software Product Lines,” *Proceeding of 11th International Symposium on Software Engineering for Adaptive and Self-Managing System*, May 2016.

[11] G. G. Eds and G. Goos, *Search-Based Software Engineering*. 2019.

[12] I. Žliobaitė, “Learning under Concept Drift: an Overview,” pp. 1-36, Jan. 2010, [Online]. Available: arxiv.org, /abs/1010.4784

[13] S. Wang, L. L. Minku, and X. Yao, “A systematic study of online class imbalance learning with concept drift,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, no. 10, pp. 4802-4821, 2018, doi: 10.1109/TNNLS.2017.2771290.

[14] M. Harel, K. Crammer, R. El-Yaniv, and S. Mannor, “Concept drift detection through resampling,” *31st Int. Conf. Mach. Learn. ICML 2014*, vol. 3, Jan. 2014, pp. 2682-2694.

[15] L. Mainetti, V. Mighali, L. Patrono, and P. Rametta, “A novel Rule-based Semantic Architecture for IoT Building Automation Systems,” *23rd International Conference on Software, Computer and Telecommunication Networks*, Sep. 2015, doi: 10.1109/SOFTCOM.2015.7314063.

[16] L. Hochstein, “Ansible: Up and Running,” *Fantastic intro to*, *First Edit. O'Reilly*, 2015.

[17] N. Bencomo, S. Götz, and H. Song, “Models@run.time: a guided tour of the state of the art and research challenges,” *Softw. Syst. Model.*, vol. 18, no. 5, pp. 3049-3082, Oct. 2019, doi: 10.1007/s10270-018-00712-x.

[18] N. Marz and J. Warren, “Big Data: Principles and best practices of scalable realtime data systems,” 2015.

[19] A. Bifet and R. Gavaldà, “Learning from time-changing data with adaptive windowing,” *Proc. 7th SIAM Int. Conf. Data Min.*, Apr. 2007, pp. 443-448, doi: 10.1137/1.9781611972771.42.

[20] P. Domingos and G. Hulten, “Mining high-speed data streams,” *Proceeding Sixth ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, Nov. 2000, pp. 71-80, doi: 10.1145/347090.347107.

[21] M. M. W. Yan, “Accurate detecting concept drift in evolving data streams,” *ICT Express*, vol. 6, no. 4, pp. 332-338, doi: 10.1016/j.icte.2020.05.011.

[22] J. Gama, R. Sebastião, and P. P. Rodrigues, “On evaluating stream learning algorithms,” *Mach. Learn.*, vol. 90, no. 3, pp. 317-346, Oct. 2013, doi: 10.1007/s10994-012-5320-9.

[23] J. M. Horcas, M. Pinto, and L. Fuentes, “Software product line engineering: A practical experience,” *ACM Int. Conf. Proceeding Ser.*, Sep. 2019, doi: 10.1145/3336294.3336304.

[24] R. T. Geraldi, S. Reinehr, and A. Malucelli, “Software Product Line Applied to the Internet of Things : A Systematic Literature,” *Inf. Softw. Technol.*, vol. 124, Aug. 2020, doi: 10.1016/j.infsof.2020.106293.

[25] R. Pastor-Vargas, L. Tobarra, A. Robles-Gómez, S. Martin, R. Hernández, and J. Cano, “A wot platform for supporting full-cycle iot solutions from edge to cloud infrastructures: A practical case,” *Sensors (Switzerland)*, vol. 20, no. 13, pp. 1-22, 2020, doi: 10.3390/s20133770.

[26] V. Tzeremes and H. Gomaa, “A software product line approach to designing end user applications for the internet of things,” *ICSOFT 2018-Proc. 13th Int. Conf. Softw. Technol.*, no. Icsoft, 2019, pp. 656-663, doi: 10.5220/0006904906560663.
[27] A. L. Bustamante, M. A. Patricio, and J. M. Molina, “Thinger.io: An open source platform for deploying data fusion applications in IoT environments,” *Sensors (Switzerland)*, vol. 19, no. 5, Mar. 2019, doi: 10.3390/s19051044.
[28] F. Anon, V. Navarathinarasah, and C. Lung, “Building a framework for internet of things and cloud computing,” Sep. 2014, doi: 10.1109/iThings.2014.28.
[29] R. Pears, S. Sakthithasan, and Y. S. Koh, “Detecting concept change in dynamic data streams: A sequential approach based on reservoir sampling,” *Mach. Learn.*, vol. 97, no. 3, pp. 259-293, Jan. 2014, doi: 10.1007/s10994-013-54339.
[30] U. Adhikari, T. H. Morris, and S. Pan, “Applying Hoeffding Adaptive Trees for Real-Time Cyber-Power Event and Intrusion Classification,” *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4049-4060, Jan. 2018, doi: 10.1109/TSG.2017.2647778.
[31] J. Cohen, “A Coefficient of Agreement for Nominal Scales,” *Educ. Psychol. Meas.*, vol. 20, no. 1, pp. 37-46, Apr. 1960, doi: 10.1177/001316446002000104.

BIOGRAPHIES OF AUTHORS

I Made Murwantara earned his PhD degree from the School of Computer Science, University of Birmingham, UK, in 2016, and Master degree, in 2002 from the Faculty of Computer Science, University of Indonesia. Now, he is a senior lecturer at the Informatics Department, Universitas Pelita Harapan, Indonesia. His research interest, primarily, on Software Engineering for and in Cloud, Internet of Things, Data Science and Automation in Software Engineering. Most of his works and publications concern about energy usage in the Cloud Computing environment and adaptive management to reduce energy consumption in Virtualized environment. Recently, he developed a method using Software Product Line Engineering to have an adaptive system that can be applied to Robotic and Internet of Things (IoT) software architecture.

Pujianto Yugopuspito received the Engineer degree in Mechanical Engineering from Universitas Gadjah Mada, Indonesia in 1991, the Master of Science degree in Software Techniques for Computer Aided Engineering from Cranfield University, UK in 1996, and the Doctor of Engineering in Computer Science and Communication Engineering from Kyushu University, Japan in 2001. He is a senior lecturer at the Informatics Department, Universitas Pelita Harapan, Indonesia. Before joining UPH, he was an IT practitioner at the Indonesian Aerospace Industries. His research area includes software engineering, high performance computing, and distributed system. He received several research grants from the Minister of Education of the Republic Indonesia. He is a member of IEEE since 2009.