Performance Evaluation of Open Source IoT Platforms

Ahmed A.Ismail
Information Technology Department
Faculty of Computers and Information
Cairo University, Cairo, Egypt
a.alaa@fci-cu.edu.eg

Haitham S.Hamza
Information Technology Department
Faculty of Computers and Information
Cairo University, Cairo, Egypt
hhamza@fci-cu.edu.eg

Amira M.Kotb
Information Technology Department
Faculty of Computers and Information
Cairo University, Cairo, Egypt
a.kotb@fci-cu.edu.eg

***Abstract*—IoT platforms play a central role in IoT architecture; as they enable the development of services and applications by end users. More than 300 IoT platforms were reported in the literature. IoT platforms vary in their capabilities and features. Accordingly, Selecting between the various platforms can be troublesome. This is particularly true given the fact that no benchmark or systematic evaluation criteria currently exist to evaluate and compare various platforms. To this end, this paper attempts to develop a performance evaluation for open source IoT platforms to reduce the complexity of the selection process. In particular, the paper evaluates the scalability (in terms of throughput and average response time) and stability (in terms of resource utilization and robustness) of the ThingsBoard and SiteWhere platforms. The two platforms are evaluated under heavy load of sensor data readings.**

***Index Terms*—Internet of Things, IoT Platform, ThingsBoard, SiteWhere, Performance Evaluation**

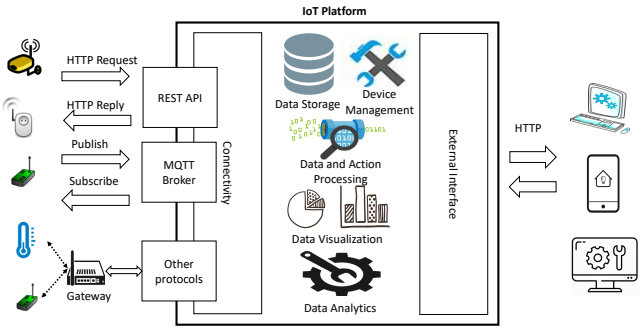

Fig. 1. IoT platforms architecture.

I. Introduction

Recent advances on the Internet of Things(IoT) promise to enable the development of innovative applications in almost all verticals that impact our daily life [1]. A key component in IoT architecture is the so-called IoT platforms (also known as Application Enablement Platform - AEP). Figure 1 shows an abstract architecture of the IoT platforms. The typical platform architecture consists of various components needed to connect, collect, monitor, manage and analyze things data. These functions include connectivity protocols, data storage, device management, data and action processing, data visualization and data analytics.

Due to the importance of the IoT platforms, a wide-range of platforms were proposed over the last few years. In fact, it is estimated that more than 300 platforms with different capabilities and features are available today at both commercial and open source levels [2]. Moreover, it is expected that the investment in IoT platforms will continue to grow for the next 5 to 10 years with a revenue of 1.6 B$ by 2021 [3], [4].

Despite the value of having several IoT platforms to choose from, however; selecting the appropriate platform can be troublesome. This is particularly true given the fact that no benchmark or systematic evaluation criteria currently exist to evaluate and compare various platforms. To this end, this paper attempts to develop a performance evaluation for open source IoT platforms to reduce the complexity of the selection process. In particular, the paper evaluates the throughput and stability in terms of CPU and memory utilization, and robustness as a message drop rate of two well-known open source platforms, namely ThingsBoard [5] and SiteWhere [6]. The two platforms are evaluated under a heavy load of sensors data readings. Two protocols are evaluated in these platforms HTTP REST(Representational state transfer) and MQTT (Message Queuing Telemetry Transport). The same evaluation approach can be used to evaluate any platform that has an open REST and MQTT APIs.

The remainder of this paper is organized as follows, Section II presents the related work to evaluating and testing IoT platforms. Section III introduces the evaluation metrics and experiment setup. Section IV presents an overview and comparison between ThingsBoard and SiteWhere platforms. Section V presents an analysis of results; Section VI presents the conclusions.

II. Related Work

Recently, several papers have been published to evaluate IoT platforms and their technical features. A framework is proposed in [7] to evaluate high-level features and design of IoT platform. Another more detailed framework [8] defines four factors to simplify the selection process of IoT platforms. These factors are technical functional offered, the strategy of provider, market presence, and recommendations. Julien Mineraud et al. [2] present a comprehensive gap analysis of 39

978-1-5386-8509-9/18/$31.00 ©2018 IEEE

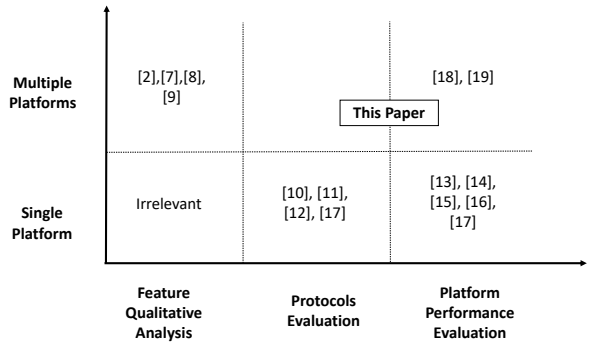

Fig. 2. Related literature of IoT platforms evaluation.

platforms and define gaps and recommendations that providers should follow to enhance the platform performance. Guth et al. [9] make a comparison of IoT platforms architectures and introduce a reference architecture of IoT platforms. One of the included platforms in the analysis is SiteWhere.

Dinesh Thangavel et al. [10] make a middleware that uses MQTT and CoAP and starts testing their performance. Another interesting performance test in [11] done in the cloud to evaluate publish/subscriber protocols using different scenarios in the context of the smart city. They use data from social weather service, smart car sharing, and traffic monitoring and test performance of protocols in each case. Alejandro Talaminos et al. [12] introduce a software framework called Distributed Computing Framework (DFC) that used to benchmarking machine to machine (M2M) protocols in healthcare applications.

Konstantinos Vandikas et al. [13] present the first performance evaluation of a platform called IoT-Framework. In their experiment, they use a load generator tool called Tsung to make a different load to their platform. The same approach used in [14] to evaluate IoTivity Cloud. Alexey et al. [15] evaluate the performance of OpenIoT platform according to its data ingestion and storage capability. Recent work [16], [17] are conducted to evaluate the performance of Fiware. In [17] two protocols of Fiware is evaluated MQTT and LWM2M. To this end, previous IoT platform evaluations can be considered a baseline performance of one platform using a synthetic model to test platforms under different loads. In [18] a performance evaluation is done between Fiware and ETSI M2M. Mauro da Cruz et al. [19] provide a performance evaluation of 5 open source IoT platforms using Jmeter [20]. They indicate that SiteWhere has a better performance than others. They evaluate REST API and measure error percentage and average response time of different load rate with different size of the packet.

Figure 2 depicts the landscape of the related work for IoT platform evaluation and analysis. As shown in the figure platform evaluation can be divided into three main categorize; namely, technical feature analysis, IoT protocols evaluation, and IoT platform performance evaluation. These three categorize can be done on a single platform or multiple platforms. In our paper, we evaluate the performance of two open source IoT platforms and evaluate two protocols HTTP REST and MQTT using open source tools.

III. PERFORMANCE EVALUATION

We present the evaluation approach and metrics that are used to evaluate the performance of the platforms. We follow the same evaluation approach and metrics in [13], [17]. The scenario we used is a smart campus that produces sensors reading from different devices and store it in platforms for further processing. Data are random and their format and type are customized according to the CityPulse dataset [21]. Furthermore, we explain the experiment setup and configuration.

A. Evaluation Approach and Tools

We study the performance of the two platforms under a different load of requests received from virtual devices that are directly connected to IoT platforms. We simulate many virtual sensors or publishers (10, 100, 200, 500, 800, 1000) using JMeter [20] as a load generator. We add to it an MQTT plugin [22] to support the MQTT protocol. Virtual sensors send requests or messages with different rates 100 and 200 requests or messages per second to platforms, thus the total number of requests arrive at the platform form 1000 publishers with 100 rates are 10^5 requests. After the finish of our experiment, we collect measurements using Prometheus [23], an open source monitoring platform that has two components Prometheus server and node exporter. Prometheus server is the main component that can manage various hosts. Node exporter monitors the performance of the Linux kernel of the IoT platform host and sends it to Prometheus server. It collects metrics such as CPU and memory usage, etc. We test each case five times and take the average of the test results.

B. Evaluation Metrics

In the following, we briefly define the selected metrics and how to measure them.

1) Scalability: IoT platform should support handling thousand and millions of devices with minimum delay to support time-critical applications. Throughput and response time are selected to measure scalability as a high throughput value indicates a good performance and it should scale linearly as a number of devices increase. Throughput calculated as a number of messages received to the platform in a second. Low response time indicates fast response and processing of platforms.

2) Stability: the stability of the system measured using several factors such as CPU utilization, active memory, and robustness in terms of drop rate. CPU utilization is reported from Linux Kernel as the average usage of all CPUs in the system. Active memory presents instructions and pages in Megabyte (MB) stored in memory in a time of the test.

C. Experiment Setup

Table I shows the specification of two laptops and software used in experiments.

TABLE I
SPECIFICATION OF TWO PLATFORMS USED IN EXPERIMENT.

Specification	Publisher	IoT Platform
Type	HP G62	Lenovo IdeaPad 310
CPU	Intel Core I3 2.27 GHz	Intel Core I7-7500U 2.7 GHz
Memory	6 GB	8 GB
OS	Windows 10 64 bit	Ubuntu 16.04
Installed Softwares	Jmeter 3.3 and Prometheus server	ThingsBoard + Cassandra DB, Sitewhere + MongoDB, Node exporter, and Eclipse Mosquitto

IV. OVERVIEW OF SELECTED IOT PLATFORMS

This section reviews the technical features of the two IoT platforms as summarized in Table II.

A. ThingsBoard Platform

ThingsBoard is a platform for data collection, processing, visualization and device management.

Connectivity: ThingsBoard supports standard IoT protocols such as MQTT, CoAP, and HTTP.

Device Management: ThingsBoard gives the user the ability to register, manage and monitor different devices. Moreover, it provides API for server-side applications to send commands to devices and vice-versa.

Data storage and visualization: ThingsBoard supports databases such as HSQLDB, PostgreSQL, and Cassandra. In our case we deploy Cassandra. It offers customized dashboards to users to monitor data in real time with many configurable widgets.

Data Analytics: ThingsBoard has its role engine for primary analysis of incoming messages and it can be integrated with Kafka and Apache Spark for more complex processing.

B. SiteWhere Platform

SiteWhere is a customizable IoT middleware platform that provides integration with many services.

Connectivity: SiteWhere supports various IoT protocols such as Restful, MQTT, AMQP, Stomp, CoAP, Socket, and Web Socket.

Device management: SiteWhere provides device specifications, device groups, asset assignment and comprehensive administration GUI.

Data storage and visualization: SiteWhere support various databases such as MongoDB, HBase, and InfluxDB. In our case we deploy MongoDB. SiteWhere doesn't provide data visualization, but it can be integrated with InfluxDB and Grafana.

Data Analytics: can be obtained by integrating SiteWhere with a Siddhi software engine for Complex Event Processing (CEP) and Apache Spark.

V. RESULTS ANALYSIS

A. Scalability

Figure 3 and 4 show the results of the throughput and average response time in case of REST; respectively. We use different rate of requests or messages 100 and 200. As shown in these figures, ThingsBoard has a better performance in case of REST reach a maximum at 1010.7 and 696.3 msg/sec at both rates with 1000 and 800 publishers. In SiteWhere throughput is increasing till reach 359.2 msg/sec in 100 requests and 320 msg/sec in 200 requests with connected 500 publishers, then it starts to decrease when the number of connected devices increases in case of 800 and 1000 publishers. SiteWhere takes much time than Thingsboard to process requests recording 422 and 1237 msec versus 93 and 342 msec in Thingsboard with 1000 connected publishers. Doubling the number of requests cause more messages that need to be processed in the platform so increase processing time and decrease throughput.

For MQTT in Figure 5, it is a publish/subscribe message protocol used for power-constrained devices, so it is so light, simple and faster than HTTP. MQTT has three quality of service(QoS) levels. We use QoS level 0 in our experiment. It just sends a message once without ensuring of its delivery; thus, the communication is very fast and has a high throughput, but it is not reliable as it can drop some messages. SiteWhere uses external MQTT broker to support MQTT. In our case, we used Eclipse Mosquitto [24]. ThingsBoard on other side has its own MQTT broker to map messages. The Throughput of Sitewhere is better than Thingsboard but have some error or drop rate. The maximum throughput recorded by SiteWhere MQTT is 2928.3 and 4451.8 msg/sec with 1000 connected publishers and has an average error rate equal to 20.97% and 32.50% in both cases. Thingsboard records 994 and 1928 msg/sec with 1000 connected publishers but with no error rate so it is more reliable. The average response time of MQTT test is a very small approach to zero as it is a very fast protocol.

In Figure 6 we try to change the size of packet send to platforms [19] by sending 10, 50 and 100 parameters or fields instead of 1 parameter in the message at a fixed rate of 100 requests per second. Again ThingsBoard is better in REST recording 84.4 msg/sec in case of 10 parameters then it begins to decrease when a number of parameter increase and size of message increase. In Sitewhere REST record 82 msg/sec and begin to decrease. For MQTT SiteWhere has a poor performance and has a high error rate increases above 90% so we don't include it. Thingsboard has a stable performance in MQTT with an average throughput of 58.1 msg/sec. The maximum average response time of REST API in SiteWhere is 917 msec and 511 msec in Thingsboard.

B. Stability

Figure 7 shows CPU utilization at the evaluation time of platforms with rate 100 msg/sec in REST and MQTT. In REST both platforms consume more CPU cycles to handle multi-threading that accept the connections. ThingSboard consumes more CPU to handle requests reach 77.1% with 1000 publishers. In the other side, SiteWhere reaches maximum CPU usage in case of 500 publishers recording 62.9% then it begins to decrease. In MQTT the reverse case happens SiteWhere takes more CPU cycles to reach 67.1% with 1000 publishers.

TABLE II
FEATURE COMPARISON BETWEEN THINGSBOARD AND SITEWHERE PLATFORMS.

Platforms	**ThingsBoard**	**SiteWhere**
Programming language	Java	Java spring framework
Supported SDKs	No, use available IoT protocols	Android and iOS
Security	Use link encryption (SSL) and access token for devices	Use link encryption (SSL)and Spring Security
Supported IoT protocols	MQTT, CoAP, HTTP, and ThingsBoard Gateway	HTTP, MQTT, AMQP, Stomp, CoAP, and Web Socket
Device management	Yes	Yes
Data visualization	Yes	No
Data Analytics	Yes	Yes
Supported databases	PostgreSQL, HSQLDB, Cassandra	MongoDB, HBase, InfluxDB.

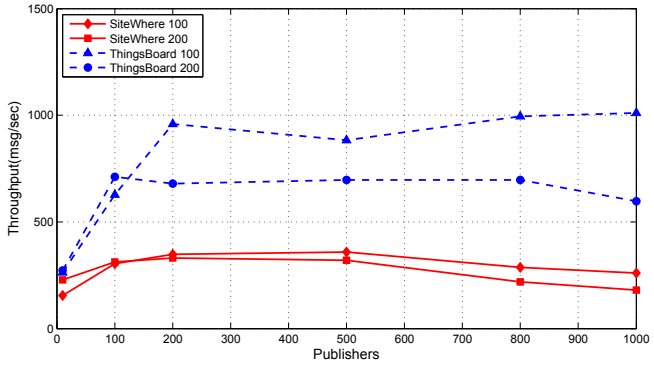

Fig. 3. The throughput of ThingsBoard and SiteWhere REST APIs with a rate of requests 100 and 200 requests per second.

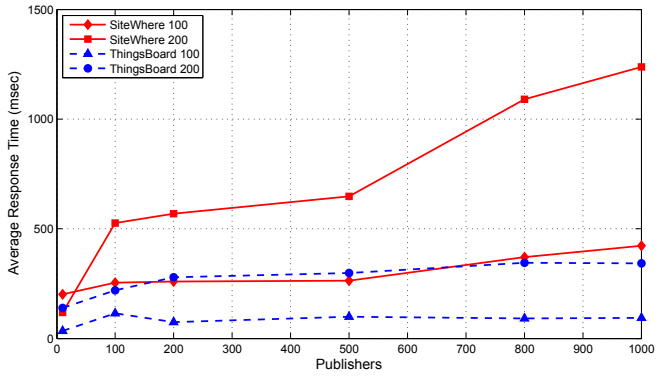

Fig. 4. Average response time REST APIs with a rate of requests 100 and 200 requests per second.

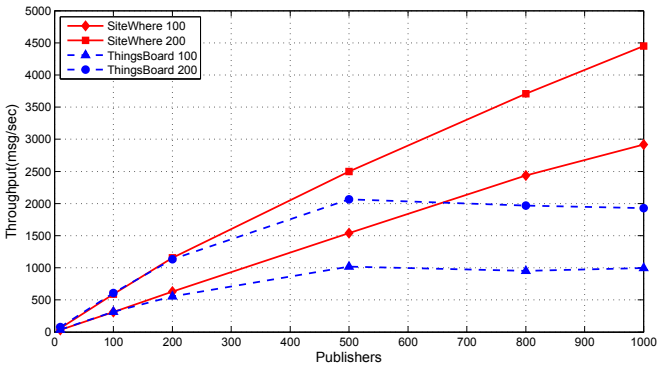

Fig. 5. The throughput of ThingsBoard and SiteWhere MQTT API with a rate of requests 100 and 200 requests per second.

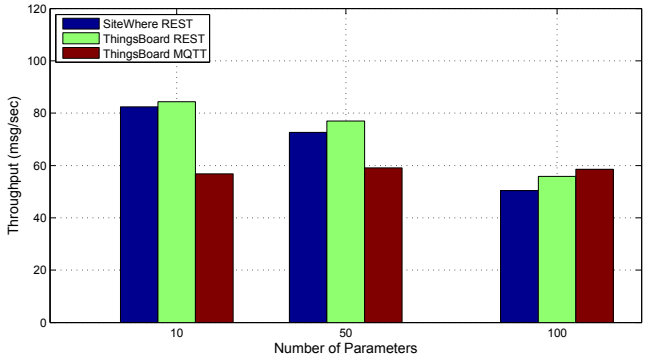

Fig. 6. The throughput of ThingsBoard and SiteWhere REST API and MQTT API with a fixed rate of requests 100 requests per second and variable size of message.

ThingsBoard begins with a high CPU rate reaches 40.8% at 500 publishers then begin to slow down.

Figure 8 that shows the average active memory in both cases of REST and MQTT with a fixed rate of 100 msg/sec. We can see that REST API consumes more memory than MQTT on the two platforms. SiteWhere has a high memory consumption than ThingsBoard in the two protocols. The maximum memory consumption of SiteWhere is 2548 MB≈2.4 GB with 1000 publishers in REST and 730 MB with 800 publishers in MQTT. The maximum memory consumption of ThingsBoard, on the other hand, is 1900MB≈1.85GB with 800 publishers in REST and 408.27MB with 500 publishers in MQTT.

VI. CONCLUSION

In this paper, we study the performance of two well-known open source platforms, namely ThingsBoard and SiteWhere. First, we compare the technical features of two platforms. Then, we evaluate the performance of the two protocols used in platforms HTTP and MQTT under a heavy load of data readings to study their performance in data ingestion.

Results show that ThingsBoard has a better performance than SiteWhere in REST. The average throughput of ThingsBoard in two rates of requests are 789.6 and 608.8 msg/sec while SiteWhere reaches 285.5 and 265.1 msg/sec. For MQTT protocol, SiteWhere has better performance in MQTT but

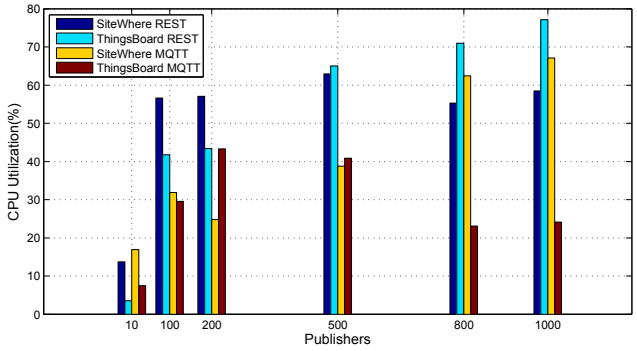

Fig. 7. CPU Utilization of REST and MQTT APIs with a rate 100 msg/sec.

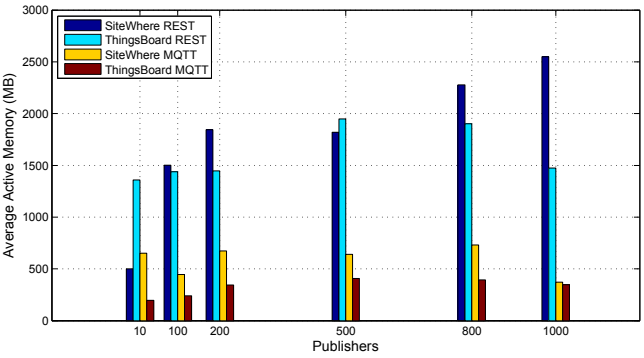

Fig. 8. Average Active memory in MB for REST and MQTT APIs with a rate 100 msg/sec.

with a high error rate. The average throughput of SiteWhere is 1311.2 and 2076.6 msg/sec while ThingsBoard is 645 and 1296.8 msg/sec. Although, ThingsBoard has throughout lower than SiteWhere in MQTT. It has a stable performance with minimum error rate. Moreover, ThingsBoard showing better performance when message size increase opposite to SiteWhere. The two platforms need more resources CPU and memory to handle many requests especially Sitewhere.

Our next step is to evaluate more open source IoT platforms and consider other use cases. We will also attempt to make the evaluation on a powerful server or on a cloud to generate a high load and to evaluate horizontal and vertical scaling of the platform.

ACKNOWLEDGMENT

This work is supported in part under the research project Campus as Mash-up Platform for IoT Experimentation(CAMPIE) funded by the National Telecommunications Regulatory Authority (NTRA) grant CFP5/2015.

REFERENCES

[1] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4):2347–2376, 2015.

[2] Julien Mineraud, Oleksiy Mazhelis, Xiang Su, and Sasu Tarkoma. A gap analysis of internet-of-things platforms. *Computer Communications*, 89:5–16, 2016.

[3] Amy Forni and Rob Meulen. Gartner Hype Cycle 2016. http://www.gartner.com/newsroom/id/3412017, 2016. [Online; accessed 28-August-2017].

[4] Padraig Scully and Knud Lueth. IOT PLATFORMS: MARKET REPORT 2015- 2021. Technical report, IoT Analytics, 2016.

[5] ThingsBoard. https://thingsboard.io/, 2017. [Online; accessed 28-August-2017].

[6] SiteWhere platform. http://www.sitewhere.org/, 2017. [Online; accessed 28-August-2017].

[7] Oleksiy Mazhelis and Pasi Tyrvainen. A framework for evaluating internet-of-things platforms: Application provider viewpoint. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 147–152. IEEE, 2014.

[8] Pankaj Ganguly. Selecting the right iot cloud platform. In *Internet of Things and Applications (IOTA), International Conference on*, pages 316–320. IEEE, 2016.

[9] Jasmin Guth, Uwe Breitenbücher, Michael Falkenthal, Frank Leymann, and Lukas Reinfurt. Comparison of iot platform architectures: A field study based on a reference architecture. In *Cloudification of the Internet of Things (CIoT)*, pages 1–6. IEEE, 2016.

[10] Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee-Xian Tan, and Colin Keng-Yan Tan. Performance evaluation of mqtt and coap via a common middleware. In *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2014 IEEE Ninth International Conference on*, pages 1–6. IEEE, 2014.

[11] Daniel Happ, Niels Karowski, Thomas Menzel, Vlado Handziski, and Adam Wolisz. Meeting iot platform requirements with open pub/sub solutions. *Annals of Telecommunications*, 72(1-2):41–52, 2017.

[12] Alejandro Talaminos-Barroso, Miguel A Estudillo-Valderrama, Laura M Roa, Javier Reina-Tosina, and Francisco Ortega-Ruiz. A machine-to-machine protocol benchmark for ehealth applications–use case: Respiratory rehabilitation. *Computer methods and programs in biomedicine*, 129:1–11, 2016.

[13] Konstantinos Vandikas and Vlasios Tsiatsis. Performance evaluation of an iot platform. In *Next Generation Mobile Apps, Services and Technologies (NGMAST), 2014 Eighth International Conference on*, pages 141–146. IEEE, 2014.

[14] Thien-Binh Dang, Manh-Hung Tran, Duc-Tai Le, and Hyunseung Choo. On evaluating iotivity cloud platform. In *International Conference on Computational Science and Its Applications*, pages 137–147. Springer, 2017.

[15] Alexey Medvedev, Alireza Hassani, Arkady Zaslavsky, Prem Prakash Jayaraman, Maria Indrawan-Santiago, Pari Delir Haghighi, and Sea Ling. Data ingestion and storage performance of iot platforms: Study of openiot. In *International Workshop on Interoperability and Open-Source Solutions*, pages 141–157. Springer, 2016.

[16] Ramón Martínez, Juan Ángel Pastor, Bárbara Álvarez, and Andrés Iborra. A testbed to evaluate the fiware-based iot platform in the domain of precision agriculture. *Sensors*, 16(11):1979, 2016.

[17] Victor Estuardo. Performance evaluation of scalable and distributed iot platforms for smart regions. Master's thesis, Lulea University of Technology, sweden, 2017.

[18] João Cardoso, Carlos Pereira, Ana Aguiar, and Ricardo Morla. Benchmarking iot middleware platforms. In *A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2017 IEEE 18th International Symposium on*, pages 1–7. IEEE, 2017.

[19] Mauro AA da Cruz, Joel JPC Rodrigues, Arun Kumar Sangaiah, Jalal Al-Muhtadi, and Valery Korotaev. Performance evaluation of iot middleware. *Journal of Network and Computer Applications*, 109:53–65, 2018.

[20] Jmeter load generator. https://jmeter.apache.org/, 2018. [Online; accessed 23-March-2018].

[21] CityPulse weather dataset. http://iot.ee.surrey.ac.uk:8080/datasets.html#weather, 2018. [Online; accessed 1-July-2018].

[22] Jmeter MQTT Plugin. https://github.com/emqtt/mqtt-jmeter, 2018. [Online; accessed 23-March-2018].

[23] Prometheus Monitoring Tool. https://prometheus.io/, 2018. [Online; accessed 23-March-2018].

[24] Eclipse Mosquitto MQTT broker. https://projects.eclipse.org/projects/technology.mosquitto, 2018. [Online; accessed 23-March-2018].