Software Monitoring of an IoT Chain Communicating over LoRaWAN

Follow-up on Planned Data Collection

Mohamed El Kharroubi
Algorithmic, Complexity and Logic Laboratory
UPEC University
Creteil, France
email: mohamed.el-kharroubi@u-pec.fr

Fabrice Mourlin
Algorithmic, Complexity and Logic Laboratory
UPEC University
Creteil, France
email: fabrice.mourlin@u-pec.fr

***Abstract* — In this work, we are interested in the problem of authentication badge usage and its control through a low energy message exchange protocol. This problem is interesting because the monitoring of these badges provides incident prevention and reporting on usage. Past work has only signaled the loss of a professional badge by a ringing tone, which is not consistent with current usage. Some authentication phases require the badge to be read over a long period. Our solution is based on sending a small number of messages on a specific protocol for each change of state of the badge. Our main result is the construction of a report on the status of all the badges in use and especially the detection of real loss of badge with notification to their owner. The gain is appreciable compared to the false detections that were previously reported.**

Keywords-loose coupled component; Big Data; LoRaWan protocol; NoSQL Database.

I. INTRODUCTION

Monitoring an Internet of Things (IoT) system necessarily involves a data collection phase. It is often complex because it depends on the deployment of the IT system: storage sites, communication nodes, choice of sensors, signal strength and of course security. This last feature brings the ultimate facet of technicality that transforms the IoT system into a hard-to-observe one.

Today's applications offer uses where personal information is obvious and the understanding of information collection is a key point. Even more so, the planning of data collection is itself monitored. Malicious interventions not only harm application results but also introduce biases that affect the overall application usage. If this duration is initially associated with the duration of deployment of the application, we observe more and more that the use of access to local or remote clouds prolong the impact of these biases and harm the observations on the long term.

These accesses to persistence systems are generally done by using the classic http or https protocol. It seems natural today to favor local storage over remote storage to ensure a good reactivity to incidents. Our past studies have also shown the contribution to data security. On the other hand, imposing data schemas remains too strong a constraint to respect when collecting data on a widely distributed system. The use of log message schema is a step forward to apply aggregation operations on messages of different origins. It is also an action to simplify the understanding of data collection. The typing is very simple and underlines the time stamp of the message and the locality of its production. There are different syntaxes for timestamp format and time zones. The use of data formats customizes the analysis of timestamps.

New information is taken into account such as the signature of the device or the language or the country. In relation to the current execution, useful information is given about the current process, or even the name of the current thread, the name of the lock acquired or returned, the name of the signal sent or processed. In short, the operating system aspects are taken into account, but there still has to be one, which is not necessarily the case in the context of IoT chains.

This information often represents functional states. Thus, a control device connected to the network can read the battery status, mains voltage, storage device status, etc. In addition, it can also collect and transmit statistics about the storage capacity used. Hence, the need for richer data schema to explain what is involved in data collection. The limit of this situation remains the consideration of business data in software monitoring with the evolution that this implies throughout the life of the software. We present our work on the collection of data from IoT chains for the application of Big Data processing.

The rest of this paper is organized as follows. In Section 2, we detail the work related to the project we are addressing, emphasizing their contributions first and the features we wish to push back. Section 3 discusses our use case and the software architecture we are implementing. Section 4 describes our Big Data processing and the provision of information related to the previous collections. Section 5 discusses the current analysis results and the impact on the NoSQL database storage. Finally, Section 6 describes the future work to achieve a better understanding of the computations. The acknowledgement and conclusions close the article.

II. RELATED WORK

In the Internet of Things (IoT) domain, each use case relies on rich and sometimes complex electronic devices for which it is important to design an ad-hoc software approach. The following studies show that monitoring is an integral part of the IT system, like mascarpone in tiramisu.

 ISBN: 978-1-68558-041-4

H. Lv et al. [1] designed a monitoring system for an urban environment that relies on a dedicated communication network to complement on-the-fly data collection. In addition to the monitoring network, there is a remote station for collection and simulation control. In this context, an observer station is identified to ensure live monitoring with the possibility to react in case of unexpected events, but this remains precarious especially in the case of long duration simulation. Events can occur in a cascade mode and it could be difficult to have a suited reaction.

W. T. Hartman et al. [2] work aims to build, test, and implement a low-cost energy monitoring and control system using IoT devices. They have designed a complete system from input to output that includes a mobile app, cloud database, application-programming interface, and hardware development. Equipped with these tools, they propose developments for each energy consuming equipment and thus observe the uses to limit the losses. If the main principles implemented are clearly explained, it remains that the development of monitoring, as an additional layer to a device already in place, is delicate here and it is almost impossible to provide business information.

S. Nocheski et al. [3] were interested in monitoring to maintain the sustainable habitat environment for certain fish species inside fishing ponds through distributed machine-to-machine communication. Their goal is to reduce the chain of responsibility for some basic actions. Their IoT system consists of sensors that measure crucial water quality factors, such as temperature, light intensity, or water level, and an on-board computer that processes the data and sends audio and visual notifications to the fish farm manager. They lead to an experimental development where their action/reaction concept is implemented. The Wivity modem allows the user to communicate with the IoT system through Wi-Fi, cellular, Long Range Wide Area Network (Lora WAN) or satellite communication, all in one product. In this context, a protocol is dedicated to the monitoring and communication of minimal data. Each message is defined by a basic grammar.

G. Yang et al. [4] presented their work on the detection of behavioral signs of pain. Their goal is to evaluate this pain at work in real situation rather than using self-reporting which can only be practiced in delayed mode. They propose a wearable device with a bio sensing face mask to monitor a patient's pain intensity using facial surface electromyogram (sEMG). The wearable device is integrated with the Internet for remote pain monitoring. It transmits data to the server via an http gateway. The cloud-accessible persistence system manages the wireless communication between the server and the web application. In this context, the data security aspects remain a concern but among the assets of the system, the patient is free to move and his mobility is taken into account.

The work of K. Sabanci et al. [5] relies on an electronic device (a camera network) to identify people in a store and count them by calculating their path through an enclosed space. Mobile clients running on a nomadic terminal such as a phone can read in a shared cloud the paths of these people to monitor their theme of interest. The authors want to distinguish between stationary and mobile people. In addition, they apply a background subtraction technique to isolate moving people. This counter will inform users of the areas left empty and then make this information available in a cloud. Users have a mobile application to track changes in the meter. The collection of information from cameras is the source of information, while the sharing is obtained with a shared persistence system. Specific clients come to consult the evolutions of this data.

The use of data schema is a more recent topic, as it introduces difficulties about data representation. J. Wang et al. [6] were interested in the control of the greenhouse environment. The acquisition and control parameters, network protocols are different for various greenhouse feedback. These factors are the keys to the abilities to communicate effectively and transfer meaningful data in IoT infrastructures. In order to realize the adaptation of data communication between the gateway and server in a greenhouse IoT system, an XML-based data encapsulation method was designed to enable data interoperability in a distributed greenhouse IoT system. A multi-agent system is used to fuse heterogeneous information and responses for data synchronization in the greenhouse IoT system. The results show the importance of patterns when communicating data over the network and this independently of the chosen protocol. From BlueTooth, RFID to HTTP, the principle is to format the data flow to better communicate with an external station.

T. M. Bandara et al. [7] make a new contribution with a large area data collection where wireless sensors are accurately distributed over an agricultural area. Thus, the data returned are all localized and this allows representing the parameters with reinforced semantics. Examples of parameters are soil moisture, temperature sensors, water volume sensors, etc. The notion of data schema is introduced in a format specific to the project, but it allows the unification of representations. The use of schema brings the validation of data before its use. Thus, during the analysis, we can discard data that does not respect the associated schema. Moreover, data extraction is of better quality when it is applied to validated data. A schema describes not only the order of the tokens but also the accuracy of the data. On the other hand, data communication process is based on energy-intensive telecommunication technologies.

In the same field of agricultural monitoring, D. Davcev et al. [8] uses a low-energy cost Long Range Wide Area Network (LoRa WAN) protocol for data communication. Of course, this introduces an additional risk because new properties appear such that the message body is encrypted, which means an additional cost for the processing but an increased security against network intrusions. Furthermore, the range of the antennas and their placement play a crucial role in the software architecture of the proposed prototype. The reading of these works underlines the difficulty of adopting the same approach.

 ISBN: 978-1-68558-041-4

TABLE I. CONTRIBUTION AND LIMITATION OF EXISTING SOLUTIONS

Contribution	Limitations
[1] point authentication of user	authentication is missing for a certain period of time
[2] monitoring mixed with business code	need for structured monitoring broken down into monitoring layers
[3] the Wivity modem does not allow the event specific monitoring	there is a need to configure the monitoring by event family
[4] the collected data are stored in a cloud and there is safety problem	there is a need to store data locally at the monitoring system without exposing it to the outside of the system
[5] the persistence system is shared ans tha data access is shared	shared access introduces security issues that must be managed or prohibited.
[6] it brings the use of XML data schema	this concept is to be propagated to all data collection phases, JSON, YAML, etc
[7] the scope od data transmission	the use of specific data transmission is an asset for the separation of concepts

The work presented in this Section represents a significant excerpt that shows the importance of data collection and its difficulties. Among other things, it highlights problems of explanation in the case of non-compliant or erroneous data collection. How monitoring can provide answers to these questions. We are also confronted with this need for explanation in our work. There are many reasons for this. On the one hand, it can come from the collection process itself, but the works presented also show sensor failures, data transfer problems, lack of energy to allow the signal to reach its objective. This list is not exhaustive, and it is important that these incidents are time-stamped and recorded.

III. SOFTWARE ARCHITECTURE

A. *Use Case*

Our case study is based on the use of authentication cards that all the people of the university have in order to enter the different premises of our site. To be more precise, these cards are nominative and are associated with accounts configured in such a way that certain premises are accessible and not others. These cards are also useful when using the classic printers and 3D printers present in different rooms. Thus, a print initiated in an office by a given person does not require the person to choose the destination printer but only the type of printer. After moving to an available printer, authentication with the business card triggers the previously initiated output.

The authentication card can be used in two ways: a simple contact with a sensor or the insertion of the card in a dedicated reader. This is necessary especially on 3D printers or large format printers whose working time can exceed one minute and leads to a new authentication request at regular period. In these cases, it is frequent that a user forgets his card when taking the documents out of the printer. Many users discover that their card is missing the next time they use it. But in this case, where did it stay? For this purpose, we asked for the design of a LoRa WAN cardholder. We chose to use the LoRa Wan protocol because on the one hand, it is a low-cost radio protocol that we master in the laboratory, and on the other hand, it is a protocol that ensures that the body of the messages is encrypted and thus ensures better security of the collected data.

This cardholder has a sensor to detect the presence or absence of card. Moreover, when the card is absent, it emits a LoRa WAN message to indicate the absent card and the identifier of the place of emission. This message is kept in the memory of the cardholder and will be sent again one hour later with an increment. In the end, the transmissions will stop only once the card is back in its holder. The use of such a protocol is explained by the autonomy of the cardholder in energy, but also by the fact that we have a LoRa WAN antenna on the site and an IP/LoRa WAN gateway per floor.

The messages collected by the network monitoring show the loss of a card. They allow the application of a script in order to send a message to the person concerned. In order to continue the validation phase of this device, we have planned several phases of monitoring data collection. From these data, a Big Data batch allows to extract the useful information to insert it in a NoSQL database. A report on the activity of this database is then published for the security department.

B. *System Architecture*

The authentication system is already deployed in the university's buildings, so we are considering the structure of a typical floor, such as those in our own building. We represent one device per device family and the users each have a cardholder containing initially a business card. The entrance and exit of the laboratory being controlled by the use of these badges, each door has two readers that we assume are placed on each side of the door. Each area requiring particular rights is thus considered as a laboratory, i.e., with a minimum entry/exit door. For reasons of simplicity, these zones can be nested one inside the other, but there is no door that allows you to leave two zones simultaneously, you will need two successive doors for that.

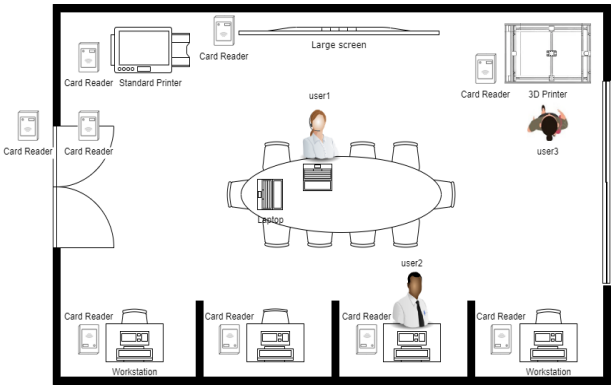

Figure 1. Physical deployment diagram of authentication system

A typical physical deployment diagram is shown in Figure 1. It helps to understand the entities that come into play in our scenario. All devices used in the lab are attached to a card reader including video conferencing equipment. This allows, among other things, to have additional information in case of an incident. The main element of this diagram is the user: three instances are present in Figure 1. Each of them carries a badge

reader containing their professional badge. In order to be connected, each of them inserted their professional badge in the reader embedded in the laptop for user1 or in the external reader at the workstation for user2 and user3. This means that they have taken their badge out of the cardholder and now it is empty. For users1 and user2, authentication is one-time and they can put away their badge as soon as the operation is finished. On the other hand, for user3 who uses the 3D printer, security constraints require his presence during the printing process. In addition, she cannot put away her badge because her presence is periodically evaluated during the printing process. This means that her badge will remain empty during the printing process and that warning messages will be issued.

A diagram of components Figure 2 applies to our first diagram and includes the software part, which produces the data, the collections and especially triggers the extraction of data towards a database. The management of badges is done in parallel with the use of badges, if they are used for the laboratory equipment, our software is monitoring badges is deployed on several materials: the cardholder, the LoRa/IP gateway and a Big Data component (Apache Spark) for data processing. Finally, actions are performed following the events of the NoSQL database.

Two distinct applications are represented in this diagram, Figure 2. The gray components belong to the user authentication application within the laboratory, while the yellow components belong to the badge monitoring application. The latter includes a data collection phase, followed by an information routing using the Kafka framework [9] and a Big Data analysis component based on the Apache Spark framework [10]. These two frameworks are considered standards in the world of Big Data for their ability to partition data in memory and especially to distribute calculations in relation to the data.

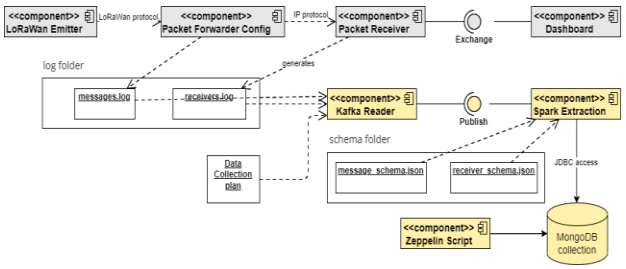

Figure 2. Component Diagram Structure

A log file directory receives all files generated by the packet forwarder or by the packet receiver. These files contain text in JSON format which is a standard text-based format for representing structured data. That is why we also have a JSON schema directory. This allows us to validate the data read by the Kafka component. On the other hand, it is then possible to search for information by using the JSONPath query language [11]. This library is used by the Spark SQL component. The management of the collections is parameterized via a configuration file used by the Kafka component, which manages the data sampling.

IV. BIG DATA WORKFLOW

The design of our Big Data workflow is based on sampling data, extracting information, and then enriching this data before saving it in a persistence system. Thus, it will be easy to build models or reports on data movements.

Figure 2 shows a breakdown into three components where the Kafka component is the entry point of the workflow; it has a large parameterization and initiates the data distribution.

A. *Kafka Reader Component*

Apache Kafka is an open-source framework for streaming messages to other components. It is a must-have tool because its outbound data rate is very high, and its low computational resource consumption is much better than competing tools. The objective of this component is to deliver, analyze and index millions of log messages from professional badge monitoring activity over LoRaWan protocol. The Kafka framework is designed for this type of high-volume, distributed message flow. It ensures that our component named Kafka Reader can process every event without being overwhelmed and without putting pressure on the log message producers.

Log message producers send events to Kafka whose content respects a data schema, which divides them evenly into partitions. We have defined a message schema for each sending source. This makes it easier to perform searches and transformations. Kafka organizes messages into topics, which are themselves divided into partitions. Each partition is an ordered queue of events assigned to our component named Spark Extractor. In our case, the more partitions there are the higher the throughput, at the expense of memory. Of course, this number of partitions has a strong impact on the distribution of computation that is done by Spark Extractor.

Each partition has several replicas that mirror the given partition as closely as possible. In case of a damaged partition, Kafka automatically switches to a replica, which provides a fault tolerance system without us having to implement it. The analysis of processing times highlights the use of these replicas. The synchronization of this distributed component relies on Apache Zookeeper, which allows the construction of an Apache Kafka node cluster [12]. Our configuration file also includes information about the automatic restart in case of connection loss, as well as data about the multiple user components of the Kafka nodes.
The configuration choices are strongly influenced by the work of B.R Hiraman [13].

B. *Spark Extractor Component*

Supplying data from a Kafka topic means that the Kafka Reader component controls the data sampling: from the duration of the reading window to the overlap time between two consecutive windows, including the number of data partitions to be consumed by the Spark component. Depending on the consumed topic, the latter determines the data schema to apply in order to validate the data. Then, the use of JSON Path queries allows us to extract the data we want to keep in a document inserted in MongoDB. An example of a message coming from the packet forwarder is given in Table 2.

 ISBN: 978-1-68558-041-4

TABLE II. SIMPLE ERROR MESSAGE FROM PACKET FORWARDER

Content of the body part

<pre>{
"timestamp": "2022-12-23T12:34:56Z",
"level": "error",
"message": "There was an error forwarding the packet",
"request_id": "4402574329",
"user_id": "pflrw1"
}</pre>

The role of the data schema is essential because it is used for our indexing process of the log messages before insertion. Moreover, if the data schema evolves over time, its externalization to the Spark Extractor component ensures an adaptation of the treatments performed on the analyzed log messages.

We chose the MongoDB server because it is very elastic and allows us to combine and store multivariate data without compromising indexing options, data access and validation rules. We use the plugin named mongo-spark-connector in order to establish a connection with the MongoDB server as in [14][15]. We provide the connection URI to MongoDB in the SparkConf object. From the data frames built in memory with Spark Engine, we save each of them via the mongo-spark-connector. Because the value of the message key is rich. We use our data schemas to parse the value associated with the message key [Table II] and thus enrich the JSON document and add new keys/values not present in the original document. This is made possible by the use of dynamic schema in MongoDB.

As an example, table 3 details a log message from the Packet Receiver. Each event from the badge holder is transmitted via a packet forwarder to our Packet Receiver.

The log messages from this component are essential to understand the actions that are triggered.

TABLE III. SIMPLE MESSAGE FROM PACKET RECEIVER

Content of the body part

<pre>{
"timestamp": "2022-12-20T10:31:51Z",
"level": "success",
"message": "badge out of the badgeholder at 2022-12-20T10:25:22Z located at the reader_id r10f1b4",
"request_id": "4402171112",
"user_id": "pflrw1"
}</pre>

Many messages are issued to track the monitoring behavior of business badges. The collected information is aggregated to provide richer messages for the observer. Thus, a first message notifies the exit of the badge from the badge holder. Then, a second message is sent when the badge is inserted in the badge reader. Finally, the badge reading event for authentication. Of course, these messages do not have the same component as their origin, but they offer a follow-up of the badges uses overall laboratory. The configurations shown in Figure 2 allow controlling the format and the details of the transmitted messages.

From our component, Spark Streaming engine defines micro-batches that initially guarantee a latency of around 100 ms at execution and a unique processing of Spark events. With Spark 2.3 came the continuous processing model with a reduced latency of 1ms, Spark events can nevertheless be duplicated. The first step is to create a Spark Session. We then retrieve our streaming data from Kafka topics where the messages are already partitioned. The *startingOffsets* option is set to *latest*, forcing us to restart the data stream in Kafka when the Spark application is expecting data. We select only the *value* column, containing our data in string form. The other columns contain metadata that might be useful in a production environment.

The MongoDB logs show the events associated with our insertions. They contribute to a first control during our test phase.

V. ANALYSIS RESULTS

A. Data Visualization

We wrote SQL and scala scripts with the Zeppelin tool to access the MongoDB database. The SQL queries are primarily useful to extract information and then use the Zeppelin graphical library to visualize the results. Examples of queries allow knowing which badges are currently in use, which is to say currently in use by a badge reader. Figure 3 shows badge usage over a 4-week period. On the x-axis is the number of operations while on the y-axis is the time of use of these badges in reading.

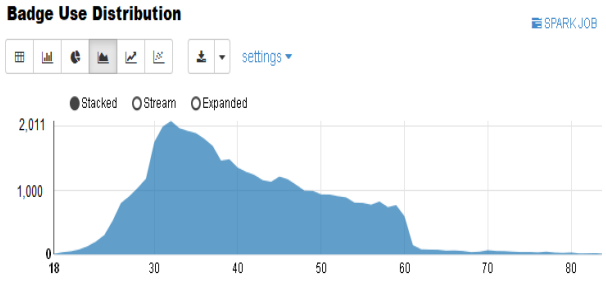

Figure 3. Representation of the time of use of the badges compared to the number of operations made

The NoteBook concept is particularly interesting in the case of simple data visualization. On the one hand, it allows validating the validity of a representation with respect to the expected users, and on the other hand, it ensures that the query language has sufficient expressive power to quickly build this representation.

Figure 4 shows a query to calculate the number of badges that are not in their badge holder.

Once the query is validated, it is easy to include it in a Scala script. The interest is major because the Scala script is compiled, which means that the query is not interpreted for

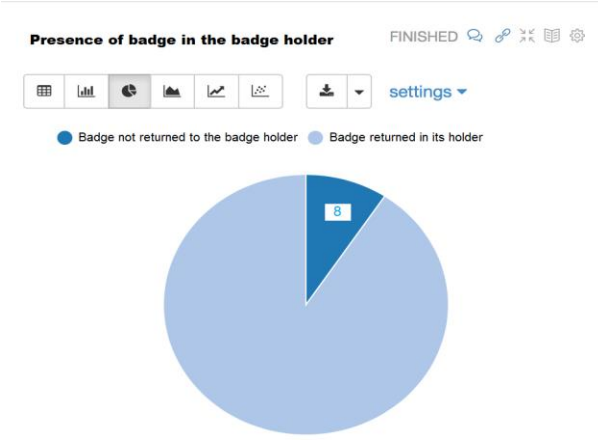

Figure 4. Calculation of the number of badges not returned to the badge holder over a period

each data set but its evaluation is reused for all the date sets, i.e., for each sampling window. We use the Zookeeper tool in conjunction with Spark because it allows more flexibility between data engineers and developers. Several works have shown the interest of such a tool for experience sharing [16]. It is also possible to install additional processors and this is what we did for JSON querying directly on MongoDB. The JsonPath interpreter is not installed by default and there is no such tool in the Zookeeper marketplace. That is why we customized our installation with an ad-hoc development based on the JsonPath library in Scala.

B. Streaming Configuration

Our experiments allowed us to externalize in the configuration of the Spark Extractor component several useful coefficients to manage data sampling. We can cite:

- The duration of a sampling window (ms),
- The overlap time of two consecutive windows (ms),
- The size of the data collected during a window (MB),
- The quota of data not respecting a data format compared to the read data (%).

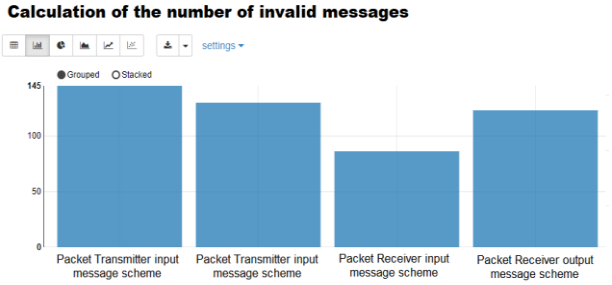

Figure 5. Use of the streaming parameters

An adapted configuration allows a better exploitation of the computing resources, in particular on clusters where the computing time is paying. Even if it was not our case, we share this cluster with other users, and it is advisable not to overload a resource that has become crucial for other simulations. Finally, from our results, we can illustrate these parameters comparisons between several types of configurations (see Figure 5).

VI. Conclusion and Future Work

We presented our work around the planning of data collection and analysis. We illustrated our work with the analysis of log messages from LoRa WAN transmission tools. We have shown that the use of data schema is an asset to have more accurate collection results. Moreover, we have created highly configurable Big Data analysis components. By precisely defining these parameters, we are able to make the best use of our computational resources and, above all, we have the means to search for the best-defined set of micro-batches.

Among the future work ahead of us, we plan to continue collecting our LoRa WAN message use case with there being other post processing that needs better monitoring. Thus, the use of these business cards are developing and if some sides are humanly delicate to accept, others like the understanding of the uses of resources of the laboratory are important. Indeed, in a policy of maintenance or renewal of materials, not all are equivalent. To prioritize certain actions, it is necessary to understand that all share some tools while others are more specialized. To illustrate this point, we will return to the use of 3D printers, for which we were able to calculate the time of use and thus show a high level of use by a large number of users. This underlines that the need for new functionality, such as a reduction in printing time, is a positive feature for all.

References

[1] Z. Lv, B. Hu and H. Lv, "Infrastructure monitoring and operation for smart cities based on IoT system. IEEE Transactions on Industrial Informatics", vol. 16 no. 3, pp. 1957-1962, 2019.

[2] W. T. Hartman, A. Hansen, E. Vasquez, S. El-Tawab, and K. Altaii, "Energy monitoring and control using Internet of Things (IoT) system. In 2018 Systems and Information Engineering Design Symposium (SIEDS)", pp. 13-18, IEEE, 2018.

[3] S. Nocheski and A. Naumoski. "Water monitoring iot system for fish farming ponds. Industry 4.0", 2018.

[4] G. Yang, M. Jiang, W. Ouyang, G. Ji, H. Xie, A. M. Rahmani and H. Tenhunen, "IoT-based remote pain monitoring system: From device to cloud platform. IEEE journal of biomedical and health informatics", vol. 22 no. 6, pp. 1711-1719, 2017.

[5] K. Sabancı, E. Yigit, D. Üstün, A. Toktaş and Y. Çelik, "Thingspeak based monitoring IoT system for counting people in a library. In 2018 International Conference on Artificial Intelligence and Data Processing (IDAP)", pp. 1-6, IEEE, 2018.

[6] J. Wang, M. Chen, J. Zhou and P. Li, "Data communication mechanism for greenhouse environment monitoring and control: An agent-based IoT system. Information Processing in Agriculture", vol. 7 no. 3, pp. 444-455, 2020.

[7] T. M. Bandara, W. Mudiyanselage and M. Raza, "Smart farm and monitoring system for measuring the Environmental condition using wireless sensor network-IOT Technology in farming. In 2020 5th International Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA) ", pp. 1-7, IEEE, 2020.

[8] D. Davcev, K. Mitreski, S. Trajkovic, V. Nikolovski and N. Koteli, "IoT agriculture system based on LoRaWAN. In 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS) ", pp. 1-4, IEEE, 2018.

[9] B. Leang, S. Ean, G. A. Ryu and K. H. Yoo, "Improvement of Kafka streaming using partition and multi-threading in big data environment. Sensors", vol.19 no. 1, pp. 134, 2019.

[10] Y. Lou and F. Ye, "Research on data query optimization based on SparkSQL and MongoDB. In 2018 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES) ", pp. 144-147, IEEE, 2018

[11] P. Bourhis, J. L. Reutter and D. Vrgoč, "JSON: Data model and query languages. Information Systems", pp. 89, 101478, 2020.

[12] T. Dunning and E. Friedman, "Streaming architecture: new designs using Apache Kafka and MapR streams. O'Reilly Media, Inc.", 2016.

[13] B. R. Hiraman, "A study of apache kafka in big data stream processing. In 2018 International Conference on Information, Communication, Engineering and Technology (ICICET) ", pp. 1-3, IEEE, 2018.

[14] M. Armbrust, T. Das, J. Torres, B. Yavuz, S. Zhu, R. Xin and M. Zaharia, "Structured streaming: A declarative api for real-time applications in apache spark. In Proceedings of the 2018 International Conference on Management of Data", pp. 601-613, 2018.

[15] P. Sangat, M. Indrawan-Santiago, and D. Taniar, “Sensor data management in the cloud: Data storage, data ingestion, and data retrieval. Concurrency and Computation: Practice and Experience", vol. 30, no. 1, e4354, 2018.

[16] A. MadhaviLatha and G. V. Kumar, "Streaming data analysis using apache cassandra and zeppelin. IJISET-International Journal of Innovative Science, Engineering and Technology", vol. 3 no. 10, 2016.

 ISBN: 978-1-68558-041-4