ISSN: 2230-9926

Available online at http://www.journalijdr.com

IJDR

International Journal of Development Research
Vol. 10, Issue, 09, pp. 40875-40882, September, 2020
https://doi.org/10.37118/ijdr.20071.09.2020

RESEARCH ARTICLE OPEN ACCESS

PROPOSAL FOR DECENTRALIZED COMPUTATIONAL STRUCTURE FOR SELECTION OF DISPERSED AND NON-STRUCTURED DATA

[1,2]Santos, L. C., [1,3]Silva, M. L.P. and [3]Santos Filho, S. G.

[1]Escola Politécnica da Universidade de São Paulo
[2]Faculdade de Tecnologia da Zona Leste
[3] Faculdade de Tecnologia de São Paulo

ARTICLE INFO

Article History:
Received 20th June 2020
Received in revised form
16th July 2020
Accepted 11th August 2020
Published online 30th September 2020

Key Words:
Enterprise Service Bus, Fog Computing, Non Structured Data, Service Oriented Architecture.

***Corresponding author: Santos, L. C**

ABSTRACT

The consolidation of new technologies, based on Electrical and Electronic Engineering, such as the application of sensors like nodes in the internet of things, the distribution of processing systems, in networks, such as cloud computing, among others, resulted in the generation of a quantity data, which, in the last two years, corresponds to almost all the data generated by humanity. Data storage and processing in a cloud environment can bring some intrinsic limitations, such as high latency and low availability events. This work presents a proposal for the use of Enterprise Service Bus (ESB) as a fog computing application, with the perspective of increasing availability and decreasing latency when acquiring data. Experimental research uses proofs of principle and proofs of concept as methods for carrying out experiments. A proof of principle and three proofs of concept, using large masses of data and the need for processing, such as Machine Learning, were carried out, demonstrating that the initial expectation is ratified.

Citation: ***Santos, L. C., Silva, M. L.P. and Santos Filho, S. G.*** "Proposal for decentralized computational structure for selection of dispersed and non-structured data", *International Journal of Development Research*, 10, (09), 40875-40882.

INTRODUCTION

The evolution and changes brought about by the electronic sector is indisputable especially due to Microelectronics that allowed the proliferation of small, low-cost instruments. More important, recently these instruments turned into systems with easy interconnection, analysis tools and the possibility of insert data into a structure, namely Cloud of Things (CoT). These abilities are much due to the also astonishing development of sensors. Sensors, produced in the most different ways and for the most unexpected uses, became of fundamental importance in health, environmental and industrial applications. Examples not only of this huge variety of uses but also on the sensors development can be seen in recent reviews on the topic and it is fundamental to establish the requirement engineering. From a macro point of view, the mechanical approach rules; it has the control of machining, vibrations and the forces applied in this process, Postel *et al* (2019) and also large structures can be inspected, such as rail infrastructure, and even important decisions, such as preventive maintenance can be taken (FALAMARZI *et al*, 2019).

On a meso dimension approach, the possibility of flexible sensors is relevant (NAG, MUKHOPADHYAY and KOSEL, 2019), but it is on micro dimensions that sensors are spread on. Above all, they are low cost devices that can be mass produced, normally using high tech and low environmental impact technologies (ARACHCHIGE *et al*, 2019&NAZEMI *et al*, 2019&ZHU *et al*, 2020). The direct consequence of easiness on acquisition and use is a huge mass of data to analyze. Thus, the last decade has seen the emergence of terms like data mining and big data. Nonetheless, although useful, these current research tools require computational efforts not always accessible on daily bases. Moreover, mobility is also demanded for these data analysis technology, i.e., the possibility of manipulating and presenting such data on standard consumer electronic devices (CEDs), such as tablets and mobiles. Thereby, a decade ago, Preechaburana *et al* (2012) advocated the use of CEDs for obtaining data in the environmental and health area, simply by adding new sensing platforms on scanners, DVDs, RFIDs, compact cameras and, more important, cell phones.

According to the authors, cell phones are the ultimate answer because they constantly evolve and already have a wide range of features, mainly to be adapted to measurements on chemical field. Furthermore, Sutherland *et al* (2019) stated that, for global environmental conservation goals, the use of cell phones for environmental assessment is, and will continue to be, extensive. In addition to the above, Agência Fapesp[1] (2020) explains that, in a moment like the COVID-19 pandemic, it is necessary to group as much data as possible and make it available for researchers to have access and manage to improve their research and propose solutions. This research opted for Service Oriented Architecture (SOA) technology due to the search for efficiency and the possibility of scaling specific functions. Villamizar (2015) pointed out monolithic applications are difficult to scale, since some few parts are more required than others and, with this approach, the entire application must be resized, consuming resources even when they are not used. Moreover, the services are created using REST technology because they are less boilerplate and also independent of language, the distribution of services is in FaaS format and using a structure of corporate service buses with Fog Computing. This scheme was chosen to create independence for large servers. Therefore, the following definitions and theoretical aspects are crucial to understand the Fog Computing development made on this paper.

Service Oriented Architecture (SOA), Simple Object Access Protocol (SOAP) and Representational State Transfer (REST)

SOA is a model whose logical units are divided into small units, with defined and independent functions, which will be the composition of a large system. This approach allows the improvement of the software development process because this model assists the software reuse since, instead of rewriting the same business model several times, each unit (service) can be only referenced and as many times as necessary (MOHAMMADI and MUKHTAR, 2011). Its main characteristics are the low computational weight of processing, the weak coupling, and the possibility of intercommunication by messages. These characteristics make the services flexible and scalable, being substituted or modified to adapt to new needs at any time. The most common architectures are in the form of web-based services, with REST and Simple Object Access Protocol (SOAP) being the most used technologies (LI and SVÄRD, 2010). SOAP is a structure composed of 3 elements, service provider, consumer of services and the registration of services. The service provider has the logical unit and its reference in a network that allows the entry of requests and the eventual return of data. The service consumer is an application that performs several functions, one of which is to send a request, with or without data, to a service and receive its return. The service registry is a logical model that contains the addressing of available services (MUMBAIKAR *et al*, 2013). Using a protocol, such as the Hypertext Transport Protocol (HTTP), the consumer application makes a request to the service using an XML document as a message, in a structure that contains a header with the URL information (MUMBAIKAR *et al*, 2013). The message model used in this structure is considered heavy for transport and, for applications with great needs for data exchange or network changes, such as mobile applications, it is not considered an adequate structure, being commonly replaced by the RESTFul architecture (WAGH and THOOL, 2012). Even so, Muehlen *et al*. (2005) explains that SOAP and RESTFul are not opposite architectures, but with different purposes, since RESTFul is based on a structural style and SOAP can be used as an element in multiple architectures. Unlike the SOAP model, the REST architecture is much less boilerplate and based verbs and names; moreover, it considers verbs from the HTTP protocol on communication between consumer applications and services, instead of messages in XML documents (MUMBAIKAR *et al*, 2013). However, Muehlen *et al*. (2005) demonstrated that there are more objects, represented by identifiers called URI, and few methods, which are the verbs described in the HTTP protocol, such as GET, POST, PUT and DELETE, among others. The communication between service and consumption, Figure 2, is stipulated as a GET operation to search for a sequence of data in some format, POST transfers some object format to a resource, PUT creates or updates a resource and DELETE removes an existing resource (FENG *et al*, 2009).

On Computing Models, Cloud Computing is the most prominent, due to service models, such as Software as a Service and Platform as a Service, among others. Fog Computing is similar to Cloud Computing, but has a reduced scope and it is aimed at applications that require low latency.

On Cloud Computing is provided ubiquity, access to the network in real time and to a number not necessarily counted of resources, which can be physical or logical, such as services, servers, storage units or applications. The Cloud Computing structure must be designed in such a way that its resources are available whenever there is demand, without the need for communication with any service provider (MELL, 2011). Proposing a more heuristic definition, Wang (2010) explains that cloud computing is a set of network services, with scalability, quality assurance and service, usually personalized, with low cost and on-demand computational structure, which can be accessed in a simple and pervasive way. Mell (2011) lists the five essential characteristics of a cloud computing model, as established by the US National Institute of Standardization and Technology (NIST), which are mainly self-service on demand, broad network access, resource pooling, rapid elasticity and measurable service. According to Mell (2011), Wang (2008) and Spillner (2017) are cloud models: Hardware as a Service (HaaS), Software as a Service (Saas), Platform as a Service (PaaS), Infastructure as a Service (IaaS), Function as a Service (FaaS), among others.

According to Atzori *et al*. (2010), internet of things (IoT) is the disseminated presence of internet elements, people or equipment, with identification and interconnected from a network. This brings great versatility but also has limitations, such as the processing capacity and heterogeneity problems that, as demonstrated by Son and Lee (2018), could be mitigated using the concept of Cloud of Things. Combining virtualized IoT elements in a cloud structure, which provides the processing power, on due time, the large processing and storage capacity of this structure could solve several problems, the first of which is to design a centralized structure for receiving and distributing data (RUI and DANPENG, 2015). As a technology associated with cloud computing, fog computing, also called as Edge Computing, Cloudlets, Multi-access Edge Computing or Mobile Edge Computing, it is a definition proposed by CISCO in 2012 that aims to subdivide part of the cloud structure, with part of the processing or

[1] Enfrentamento da pandemia exige infraestrutura para compartilhamento de dados abertos, Agência Fapesp (2020), https://agencia.fapesp.br/enfrentamento-da-pandemia-exige-infraestrutura-para-compartilhamento-de-dados-abertos/33869/

storage placed closer to the nodes, which are the top of the fog structure (MEBREK, 2017). It is considered as a movement of the cloud environment to the edge of the structure (GARCIA *et al*, 2018). As defined by Chamas (2018), one of the Cloud Computing applications is the Computation Offloading. Defined as the transfer of processing from a device, mostly mobile, to another processing environment, due to low resource capacity (processing, energy, storage, etc.), however, Ye *et al*. (2016) explains that this processing generates traffic, on computer networks, of very heavy objects, which end up increasing latency in operations. therefore, fog computing is a solution for operations that need low latency because it is close to the nodes, the data travels for a shorter distance or can be fragmented. This research uses, as technology for fog computing, the enterprise service bus. It combines Service Oriented Architecture with an event-driven approach to create a possibility of direct communication, and when necessary to make translations, between heterogeneous services, since the services included in the bus can be accessed by consumer applications or by other services (MARÉCHAUX, 2006). In addition to communication, since it is mostly implemented in Web Services (REST or SOAP), the buses must describe the destinations in a neutral way, only using the identifiers. Service management and access security should be priorities in the implementation of ESB as they will also communicate with critical applications. On such scenario, the integration of technologies, the increase of computational capacities for processing and, on top of all this, the storage of them have provided a great generation of data, unstructured and of totally dispersed in storage. Thus, this work aims the production of a simple tool for obtaining, analyzing and manipulating data; for that matter, a fog computing system is used. This work is organized as follows: first, the Methods used to achieve these objectives are discussed, followed by Results and Conclusions.

MATERIAL AND METHODS

The nature of this work is applied research. The previous documentary research is meant as means for definition of available technologies, but, most of the steps taken are exploratory, since, after requirements engineering is determined, an experimental approach is compulsory in order to verify the functionality of the software. Therefore, it is qualitative and quantitative work that determines not only whether certain premises are adequate, but also the performance of the software under development, respectively. The main methods used on this research are as follows:

- Research bases: since this work uses multiple data from several distinct sources, it is multiple case studies;
- Considering its nature, Software Engineering, the following project development stages were carried out (Pichi Junior, 2011):
  - Analysis of similar products found in the market (see bibliometric analysis);
  - Analysis in relation to use (see bibliometric analysis);
  - Functional analysis;
  - Results obtained from the analysis
  - Final structuring and Description of the concept

Regarding requirement engineering (Pichi Junior, 2011), as presented in Results, on this paper:

- Product engineering, choose of representation language;
- Process engineering - the simulation of the process, and its respective optimization;
- Project management - monitoring the steps
- Interface evaluation

In an initial step a proof-of-principle was carried out using medical data, as described in a former work (Santos *et al*, 2019). After that, several distinct problems were addressed to test different propositions, such as small amount of data, sensors control and pattern recognition. Finally, the obtained results were evaluated by an expert in an unstructured interview. Creation of services, interconnected in a bus, to make treatment of dispersed and non-structured data, availability of data in real time, analysis of datasets trained to work in machine learning operations are characteristics explored in the conceptual proofs of this work. Until the present moment, no final assessment has been made with professionals or possible users, however, an assessment of the idea, with a recognized professional in the area, for her Ph.D. degree, was made using an unstructured interview. The interview started demonstrating the idea of using an Enterprise Service Bus (ESB) as a technological possibility of implementing a Fog Computing structure. The interviewee approved the possibility, suggesting that a series of tests be carried out with the main activities carried out in a cloud environment, which has a large amount of data and / or latency problems. Following, the first proof-of-principle was presented, with the application of services on a bus that contains filtering, treatment and availability of a large mass of unstructured data. The interviewee approved the test and suggested possibilities to be applied in other possible proofs of concept. Finally, the interviewee was asked whether, with her experience in the area, it was understood that this would be a disruptive possibility with wide applicability, which was answered yes and listed sub-areas of knowledge that could apply the technology.

RESULTS

The results are presented as referenced on section II, Bibliometric Analysis, Function Analysis, Case Studies and Final Evaluation

Bibliometric Analysis: Recently Môro (2018) pointed out that 90% of the data generated by all humanity was generated in the previous 2 years, in addition emphasized the expectative of 1.7 Megabytes generated per person per second in 2020. As the data are generated by different equipment and/or applications, they are also storaged in files of the most diverse formats, in databases with different configurations and, sometimes, available only by HTML codes, that make access difficult; thus, this situation demands a lot of time and effort to create a consistent bases to do analysis, generate knowledge and make decisions. In order to evaluate the current available technology for data management, a bibliometric survey was carried out. Initially, all viable technology was listed and its frequency on the research works was estimate using Google Academic. The survey of viable technology used two different approaches: first, keywords were selected using oriented interviews with experts, and then, these keywords were crossed among each other. The surveyed keywords, when compared among themselves using number of appearances on

Table 1 - Number of appearances, in Google Scholar®, of key terms related to data search

Key Terms	Results	Type
RESTFul	2.240.000	Independent Terms
Big Data	675.000	
“Service Oriented Architecture”	161.000	
“Enterprise Service Bus”	13.300	
“Fog Computing”	17.900	
“Function as a Service”	2440	
“Mobile-Sensing Technology”	280	
“Enterprise Service Bus” “Service Oriented Architecture” “Fog Computing”	70	Combined Terms
“Enterprise Service Bus” “Service Oriented Architecture” “Fog Computing” “Big Data”	53	
“Mobile-Sensing Technology” “Computation Offloading”	4	

Table 2. Measure of Proof-of-Principles

Test	Methodology	Results
1	Query, 5 times, directly on the MongoDB server, a Linux command, with 2 parameters as a filter, returning approximately 4300 JSON documents	Mean 43 seconds
2	Using service improving the database queries, call service 5 times, passing 2 parameters as a filter (the same as in Test1), returning approximately 4300 JSON documents	Mean 13 seconds (First execution takes 62 seconds to initialize Apache Spark®)

Google Scholar, as can be seen in Figure 1, shows a behavior that resembles waves, with some words replaced by other ones. This phenomenon is inherent to the electronic sector, as previously reported on the introduction, with sudden changes being even more common in the area of information technology. The exception is “RESTful” term, such technology was already consolidated at the beginning of the millennium, whereas, for the other terms, there has been a huge increase in interest only in the last decade, which is consistent with the development of apps for mobile phones as well. Initially, the key terms search was made individually and, subsequently, searches were made combining the terms, according to the combinations of technologies provided throughout the project.

Table 1 shows the keywords and the respective total number of appearances of these terms on Google Scholar. The emerging scenery points to, along with a surprisingly low of Google appearance of “Mobile-Sensing Technology”, an even much lower number if the term "Computation Offloading" is linked by the conjunction AND. However, that is the exact situation if data management is to be done using low cost instruments, such as mobiles, and platforms, such as fog computing. It is worth noting these only four appearances occurred on the last 5 five years, in other words, there is low efforts on such development. This is due to the limitation of the cell phone in processing data, which leads to the use of cloud computing. On the other hand, it is necessary to interact with the device and the data analyzed in another environment, hence the importance of developing inexpensive software solutions, fast and in the form of apps, that is, as much as possible according to practices expected on human-computer interaction. Since only 4 papers dealt with computational processing in mobile using the ideas that works unravel, content analysis was performed. This analysis was designed using SWAT ideas, i.e., an effort was done in verify not only advantages and limitations of each approach on these works but also in understanding the conceptual developed and possible lack of feasibility. Therefore, Zamfir (2016) proposes a computer system for communication and sensor data storage that does not include search of data already consolidated in databases. Li (2018) suggests a computer system for analyzing sensors, which is IoT system, however, again, does not include data search already consolidated in bases. Chang (2017) makes a review analysis of systems already proposed in the literature and based on BPMN model (neutral information regarding SWOT analysis). Although the author hands over important information, the present research is quite different and based on the development of structure computational. Liyanage (2018) put forward a Platform as a Service implementation for portable devices to achieve better performance. This work has some similarities with the one presented in this paper. Nonetheless, Liyanage´s proposition is a little different from ours since ours focuses on Fog Computing architecture for increase performance in addition to the use of Function as a Service. Thus, the present research is justified not only by the low number of computer systems already presented, but also the dispersion of the means and modes of these current systems.

Functional Analysis (Architecture): The main objective is to create an Enterprise Service Bus to interconnect small services (that perform specific functions, Function as a Service) and will behave like Fog Computing. For the proof-of-principles, a medical dataset was used, which come from files of different formats or from web pages. As the data are in different formats, and presentations, 2 data formatting services were designed, with the purpose of standardizing, in JSON format, which is a character string that separates the fields, easily read by any application. Figure 2, with the diagram of the proof-of-principles creation illustrate that the first service reads the files, identifies the format and proceeds with the conversion. The second service reads the HTML page and scrapes, extracting the desired content. Connected to the 2 data formatting services, a service that keeps consulting and checking if there are new JSON to insert in a non-structured database. In this case, the choice for a non-structured database is justified by the diversity of formats and possible fields in the data source. A last service also interconnected to the database, which can be consulted by any third-party software, coupled with processing tools for performance gain, offers the possibility of filtered queries to the database.

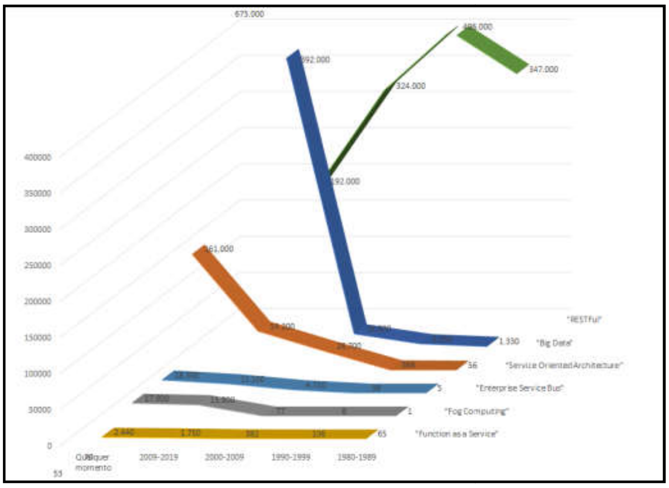

Figure 1. Behavior of the development of technologies for software production, estimated through the number of citations in Google® Scholar

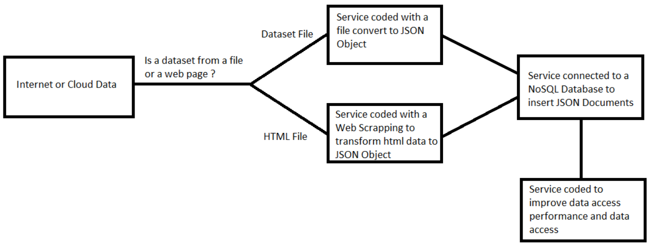

Figure 2. Diagram of proof-of-principles

The first proof of concept is made to be coupled to the bus, providing services for the sensing, as shown in figure 3. A service was developed to receive data from sensors that were measuring the temperature in a room, which needed a controlled temperature. Five sensors were distributed throughout the room to provide temperature values and sent data to the service. Another service was consulting the previous one to check if there was new data to send to a structured database. To check if the room temperature was suffering from external variations, a service that scrapes a website that monitors the temperature in several cities (including the city where the room is located), was developed, and was also consulted by the service that inserts the data in the database. To work with the data, 2 services were developed, also coupled to the bus. One that offers data that can be consumed by third-party applications and another that monitors values and sends an alert to the person in charge of the room in case of inadequacy.

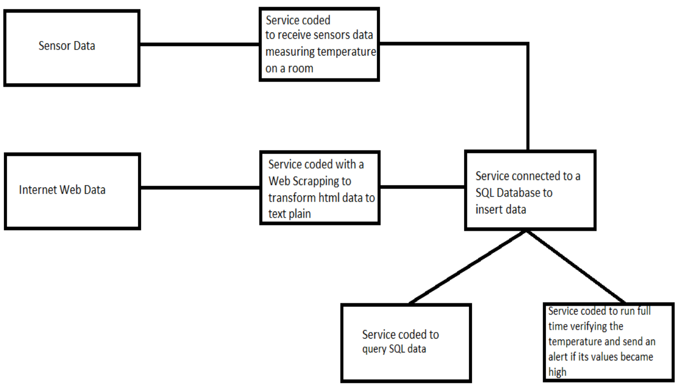

Figure 3. Diagram of First Proof of Concept

The second proof of concept tries to validate the solution in the higher computational processing. A machine learning training model to detecting elements in images was deployed in a cloud computing environment. Figure 4 presents that a service was coupled to the ESB to do the acquisition of this training model (it is planned, later, to make a service that returns data to cloud to improve the trained model). A service that receives images and applies the machine learning model to check if it is possible to identify elements in that image. A third service, which communicates with the previous one, looking for a JSON object with the elements identified in the image and offering it to third party software is also on the bus. The third proof of concept is similar to the proof of principles, except for the data source that is structured. A source of Brazilian data on people who have been tested for contamination with Sars-Cov-2 has been identified. This source is structured in CSV format, therefore, it is sufficient to create a service that analyzes the file and filters the data of interest. A service as presented in Figure 5, connected to a structured database is requesting data from the service mentioned above and sends it to the database. Finally, a service that sends filtered data from the database to third-party applications was also coupled to the bus.

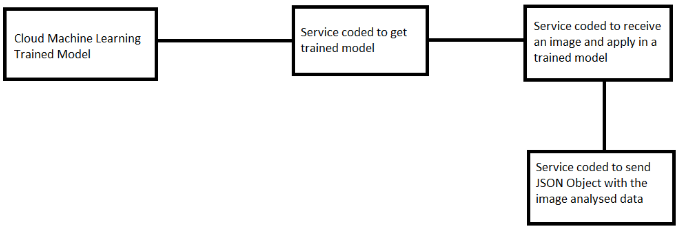

Figure 4. Diagram of Second Proof of Concept

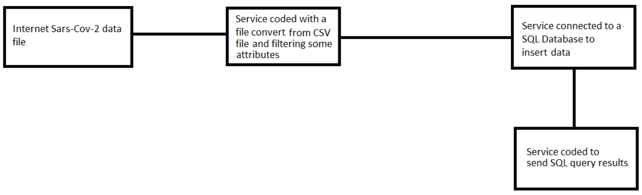

Figure 5. Diagram of Third Proof of Concept

Case Studies: The first step is software language, structure and architecture definitions. Since it is Fog Computing, due to the operation closer to the edge of the network, the first step was to define the language\structure of the system: Service Oriented Architecture Defined on a Service Bus Corporation. After that, it is required to describe what types of services should be modeled\created, what integration between them and the possible applications that would consume the services (FaaS), and how the data would persist. The services chosen were Data Search, Data Decoding, Data Transformation, Receiving data from devices, such as sensors, and Data Persistence. Data search service is mandatory, to filter data dispersed all around on the same universe. Decoding and data transformation services is required because the data is available in different formats and, for better performance, the definition of a single database structure is the solution. Persistence of data means that the received and transformed data will be preserved. Devices, such as sensors, reading is interesting because, once the universe has been defined, masses of data can be brought directly to the system database.
Persisting data is carried out in NoSQL MongoDB, since in this format storage resources are optimized (Oliveira *et al.*, 2018). For very large masses of data, this choice makes a lot of

difference. Query is better performed in SQL. A service that reads data sources and inserts them without need for human interaction was also provided, in this way, difficulty in performance of data entry is diminished. The proof-of-principle described, elsewhere Santos *et al* (2019), used this approach; therefore, a service bus was developed. Since it is based on FaaS concept, the definition of the parameters is essential to create its specific functionality. The data search service receives a logical address that contains files, identifies which files have relevant data and returns a data structure in the list format with absolute addressing of each file and its format. The data decoding service receives this list, reads each file and transforms the data into plain text format.

The data transformation service receives a mass of unformatted text and tries to convert it into JSON format, the format expected by the NoSQL database. The data persistence service can 1) receive several documents in JSON format and persist the data in the NoSQL database, returning only one message in HTTP code. This message indicates whether the operation was successful; 2) receiving query parameters to return data sets for analysis or archetypes.

The proof-of-principle was data extraction from healthcare public files. These files provided data from almost 13-year period, were in different formats and structured according to the period that has been consolidated. The result was a database greater than 30 million JSON documents with an average of 25 attributes each. Table 2 shows the results of the consultations to the database in the ESB using, after going through the services described in the functional analysis, (a) without the service that improves the consultations and (b) with the service improving the consultations. Since the proposed architecture has several advantages, the posterior three cases studies, as pointed out in Method item, were performed. The enterprise service bus was created, and proof of concepts were made to assess its main characteristics. The first proof of concept carried out has a service that reads unstructured medical data files, the content of which is confidential. The first proof of concept performed has sensors installed in a room with the need for temperature monitoring and a comparison with the external temperature. A service that receives data from the sensors over the network, full time, has been associated with the bus, as well as a service that communicates with a Web Service, also updated in real time with the temperature of the city. If one or more sensors detect temperature variation, a communication service sends a message to the person in charge. Figure 6 presents third party applications consuming de sensors data to generate graphs, (a) in third party JavaSE application and (b) MS Excel® application. The second proof of concept performed, has a machine learning mechanism where a dataset contained in the cloud generates machine learning for the identification and everyday elements in images.

The training model is brought as a service that detects images and, if it brings a positive result, sends new data to increase the dataset in the cloud and improve training, increasing accuracy. Figure 7, extracted from internet, presents an internal machine learning processing and a service out.

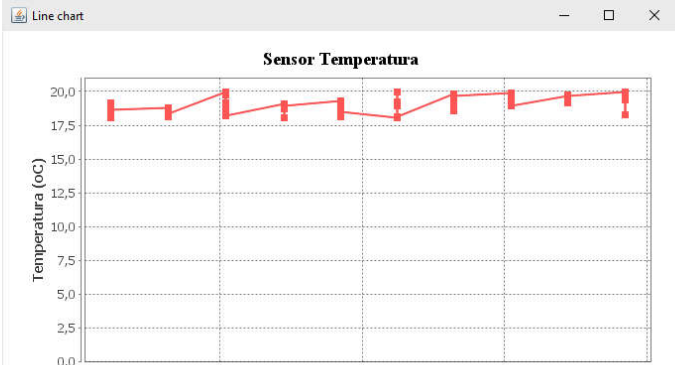

Figure 6a. Third party JavaSE application

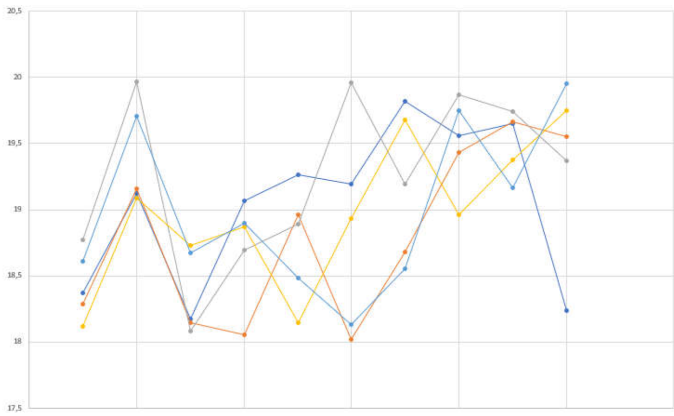

Figure 6b. MS Excel® consuming data

Despite generating the image, as shown in figure 7, the service converts the image into a JSON format output with the image contents. The image generated the JSON:

{"bus":"1","bicycle":"1","car":"1","person":"8","truck":"2"}

The third proof of concept performed, has a dataset of approximately 3 million tests of people for the detection of contamination by Sars-Cov-2.

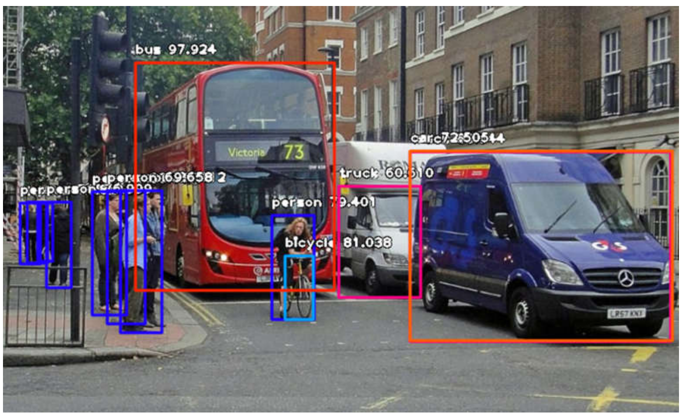

Figure 7. Internal machine learning processing with service data generated by third party request

Figure 8. Response from service to a third-party

The data is in a cloud environment, structured, but made available in an internal, unconventional format. A data acquisition service, associated with a relational database, was formatted and bringing the data to form a database, already filtered according to the analyst's need. A database filtering service made possible to make a one-dimensional analysis of the data, allowing conclusions to be drawn regarding contamination related to previously existing comorbidities. The results are obtained in seconds or fractions of seconds on the studied base. Figure 8 presents a third-party application consuming the service.

Final Evaluation: In first proof of concept, is possible to demonstrate the ability of a Fog Computing environment applied on Enterprise Service Bus, to work in real time, reducing time in real-time monitoring activities, since, if the sensors fail, equipment can burn. The second proof of concept demonstrate that the fog environment can be directly in communication with the cloud environment, facilitating access to the necessary data and feedback to the cloud data. In third proof of concept, other analyses are still under study to determine other possibilities and was possible to demonstrate, as in the proof-of-principle, a reduction in the latency in obtaining the data, the possibility of creating a data environment that allows a specific analysis.

Conclusion

This work suggests the use of technologies that presents low financial and computational cost and extracts the maximum resources to create a system that allows searching, decoding and persisting related data, which are spread out and in varied formats. This approach encompasses a new computational structure that allows capture decentralized and non-structured data for further analysis. This retrieved data, which can be arranged in a network or made available by sensors, such as those contained on smartphones or in microsensors, may be consulted with low resource consumption and low latency. Furthermore, the system can also make other services, such as data analysis, treatment and transformation, to centralize it in a database. The system makes use of Fog Computing, a technology frontier to be faced in the future. The three case studies discussed in this work shows the real time access from the processing data, the intercommunication with the cloud computing and the low latency working with large amount of data, some of the characteristics expected in a fog computing environment.

REFERENCES

Arachchige W., Jayathilaka D. M., Qi K., Qin Y., Chinnappan A., Serrano García W., Baskar C., Wang H., He J, Cui S., Thomas S. W., Ramakrishna S. 2019. Significance of Nanomaterials in Wearables: A Review on Wearable Actuators and Sensors, Advanced Materials, Volume31, Issue7 1805921- 1805942 https://doi.org/10.1002/adma.201805921

Atzori, L., Iera, A., Morabito, G. 2010. The internet of things: A survey. *Computer networks, 54*(15), 2787-2805.

Chamas, C. L. (2018). *Consumo de energia em dispositivos móveis Android: análise das estratégias de comunicação utilizadas em Computation Offloading.*(Master Dissertation) São Paulo: USP.

Chang, C., Srirama, S. N., & Buyya, R. 2017. Mobile Cloud Business Process Management System for the Internet of Things: A Survey. *ACM Computing Surveys (CSUR), 49*(4).

Falamarzi A., Moridpour S., Nazem M. 2019. A Review on Existing Sensors and Devices for Inspecting Railway Infrastructure. *Jurnal Kejuruteraan, 31*(1), 1-10.

Feng, X., Shen, J., Fan, Y. 2009. An alternative to RPC for Web services architecture. *2009 First International Conference on Future Information Networks*, 7-10.

Garcia, J., Simó, E., Masip-Bruin, X., Marín-Tordera, E., & Sànchez-López, S. 2018. Do We Really Need Cloud? Estimating the Fog Computing Capacities in the City of Barcelona. *IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, 290-295.

Li, W., Svärd, P. 2010. REST-based SOA application in the cloud: a text correction service case study. *2010 6th World Congress on Services*, 84-90.

LI, W., Wang, B., Sheng, J., Dong, K., Li, Z., & Hu, Y. 2018. A Resource Service Model in the Industrial IoT System Based on Transparent Computing. *Sensors*.

Liyanage, M., Chang, C., & Srirama, S. N. 2018. Adaptive mobile Web server framework for Mist computing in the Internet of Things. *International Journal of Pervasive Computing and Communications, 14*(3/4), 247-267.

Maréchaux, J.L. 2006. Combining service-oriented architecture and event-driven architecture using an enterprise service bus. *IBM developer works*, 1269-1275.

Mebrek, A., Merghem-Boulahia, L., Esseghir, M. 2017. Efficient green solution for a balanced energy consumption and delay in the IoT-Fog-Cloud computing. *IEEE 16th International Symposium on Network Computing and Applications (NCA)*, 1-4.

Mell, P., Grance, T. 2011. *The NIST definition of cloud computing.*

Mohammadi, M., Mukhtar, M. 2011. SOA-based business process for Supply Chain Management. *2011 Malaysian Conference in Software Engineering*, 213-216.

Môro, D. K. (2018). *Reconhecimento de Entidades Nomeadas em Documentos de Língua Portuguesa.* (T. d. Curso, Ed.) Araranguá: Universidade Federal de Santa Catarina.

Muehlen, M. Z., Nickerson, J. V., & Swenson, K. D. (2005). Developing web services choreography standards—the case of REST vs. SOAP. *Decision Support Systems, 40*(1), 9-29.

Mumbaikar, S., Padiya, P. 2013. Web services based on soap and rest principles. *International Journal of Scientific and Research Publications, 3*(5), 1-4.

Nag, A., Mukhopadhyay, S. C., Kosel, J. 2019. *Printed Flexible Sensors: Fabrication, Characterization and Implementation.*Springer.

Nazemi, H., Joseph, A., Park, J., Emadi, A. 2019. Advanced Micro- and Nano-

Gas Sensor Technology: A Review. Sensors, 19, 1285 https://doi.org/10.3390/s19061285

Oliveira, M. S., Melo, N.F., Santos, L.C. 2018. Banco de Dados No-SQL X Banco De Dados SQL. *South American Development Society Journal, 4*(11), 298-320.

Pichi Junior, W. (2011). *Construção de Protótipo para Ensino na Área Tecnológica: Cromatografia como Estudo de Caso.*(Master Dissertation) São Paulo:Centro Paula Souza.

Postel, M., Aslan, D., Wegener, K., Altintas Y. 2019. Monitoring of vibrations and cutting forces with spindle mounted vibration sensors. *CIRP Annals*. *68*(1), 413-416.

Preechaburana, P., Gonzalez, M. C., Suska, A., Filippini, D. 2012. Surface Plasmon Resonance Chemical Sensing on Cell Phones. *Angew. Chem. Int. Ed*, 11585-11588.

Rui, J., Danpeng, S. 2015. Architecture design of the Internet of Things based on cloud computing. *Seventh International Conference on Measuring Technology and Mechatronics Automation*, 206-209.

Santos, L.C., Simões, E. W., Santos Filho, S.G., Silva, M.L.P. 2019. Proposta de

estrutura computacional decentralizada, de fácil uso, para aplicação em seleção de dados ambientais dispersos e não estruturados. *Anais do XXI Engema,* http://engemausp.submissao.com.br/21/anais/arquivos/101.pdf, 1-16, ISSN 2359-1048

Son, Y. H., Lee, K.C. 2018. Cloud of things based on linked data. *2018 International Conference on Information Networking (ICOIN)*, 447-449.

Spillner, J. (2017). Snafu: Function-as-a-service (faas) runtime design and implementation. *arXiv preprint arXiv:1703.07562*.

Villamizar, M., Garcés, O., Castro, H., Verano, M., Salamanca, L., Casallas, R., Gil, S. 2015. Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud. *10th Computing Colombian Conference (10CCC)*, 583-590.

Wagh, K., Thool, R. 2012. A comparative study of soap vs rest web services provisioning techniques for mobile host. *Journal of Information Engineering and Applications, 2*(5), 12-16.

Wang, L., Tao, J., Kunze, M., Castellanos, A. C., Kramer, D., Karl, W. 2008. Scientific cloud computing: Early definition and experience. *10th ieee international conference on high performance computing and communications*, 825-830.

Ye, D., Wu, M., Tang, S., & Yu, R. 2016. Scalable fog computing with service offloading in bus networks. *IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud)*, 247-251.

Zamfir, M., Florian, V., Stanciu, A., Neagu, G., Preda, Ş., & Militaru, G. 2016. Towards a Platform for Prototyping IoT Health Monitoring Services. *Borangiu T., Dragoicea M., Nóvoa H. (eds) Exploring Services Science, 247*.

Zhu, J., Liu, X., Shi, Q., He, T., Sun, Z., Guo, X., Liu, W., Sulaiman, O.B.,

Dong, B., Lee, C. 2020. Development Trends and Perspectives of Future Sensors and MEMS/NEMS. Micromachines, 11, 7,. https://doi.org/10.3390/mi11010007

****