Decentralized Computational Structure for the Selection of Dispersed and Non - Structured Data Proposal

Shonal Rath[1], Hardik Agrawal[2]

BA Economics (Hons), University of Delhi, New Delhi – 110052, India
Mail ID*: shonalrath2000[at]gmail.com*

B. Tech Electrical, Electronics and Communications Engineering, Birla Institute of Applied Sciences
Mail ID: *hardikagrawal46[at]gmail.com*

Abstract: ***The convergence of new technologies based on Electrical and Electronic Engineering, such as the use of sensors as nodes in the internet of things, the distribution of processing systems in networks, and cloud computing, among others, has resulted in the generation of a volume of data that, in the last two years, has accounted for nearly all of humanity's data. In a cloud environment, data storage and processing might have certain inherent constraints, such as high latency and poor availability events. This paper proposes using Enterprise Service Bus (ESB) as a fog computing application, with the goal of enhancing availability and lowering latency.***

Keywords: Enterprise Service Bus (ESB), Cloud Environment, Cloud Computing, IoT

1. Introduction

The electrical sector's progress and changes are undeniable, especially thanks to Microelectronics, which enabled the spread of tiny, low - cost devices. More importantly, these instruments have lately evolved into Cloud of Things systems with easy connectivity, analytical tools, and the ability to enter data into a structure (CoT). These skills owe a lot to the incredible advancements in sensor technology. Sensors, which are made in a variety of methods and for a variety of purposes, have become critical in health, environmental, and industrial applications. Examples of this wide range of applications, as well as sensor development, may be seen here. It is critical to develop the requirement engineering, as demonstrated in recent studies on the subject. From a macro perspective, the mechanical approach reigns supreme; it has control over machining, vibrations, and the forces applied during the process, according to Postel et al (2019), and it can also inspect large structures, such as rail infrastructure, and make important decisions, such as preventive maintenance (FALAMARZI et al, 2019).

The possibility of flexible sensors is significant on a meso dimension approach (NAG, MUKHOPADHYAY, and KOSEL, 2019), yet sensors are dispersed on micro dimensions. Above all, they are low - cost, mass - produced products that are typically made with high - tech, low - impact technology (ARACHCHIGE et al, 2019&NAZEMI et al, 2019&ZHU et al, 2020). The result of ease of gathering and usage is a massive amount of data to evaluate. As a result, phrases like data mining and big data have emerged in the recent decade. Nonetheless, these existing research methods, while beneficial, need computing resources that are not always available on a daily basis.

Furthermore, these data analysis technologies require mobility, i.e., the ability to manipulate and show data on typical consumer electronic devices (CEDs), such as tablets and smartphones. Preechaburana et al (2012) proposed for the use of CEDs for data collection in the environmental and health fields a decade ago, simply by adding new sensing platforms to scanners, DVDs, RFIDs, small cameras, and, most importantly, mobile phones.

Cell phones, according to the authors, are the ideal solution since they are continually evolving and already have a wide variety of characteristics, including the ability to adjust to chemical field data. Furthermore, Sutherland et al (2019) said that the usage of mobile phones for environmental evaluation is and will continue to be widespread in order to achieve global environmental conservation goals. In addition to the foregoing, Agência Fapesp1 (2020) argues that, in a situation like as the COVID - 19 pandemic, it is critical to collect as much data as possible and make it accessible to academics so that they may enhance their study and provide answers. The Service Oriented Architecture was chosen for this study.

Monolithic apps, according to Villamizar (2015), are difficult to scale since certain portions are more important than others, and with this technique, the entire programme must be scaled, using resources even when they are not utilised. Furthermore, the services are built utilising REST technology since it is less boilerplate and language neutral; the services are distributed in FaaS format and using a corporate service bus topology using Fog Computing. This method was chosen to provide big servers with independence. As a result, the terminology and theoretical features that follow are critical to comprehending the Fog Computing evolution presented in this work.

SOA, SOAP, and REST are acronyms for Service Oriented Architecture, Simple Object Access Protocol, and Representational State Transfer (REST) SOA is a paradigm in which logical units are split into tiny units with specified and independent functions that make up a big system's composition. This technique improves the software development process since it encourages software reuse because, rather than rewriting the same business model many times, each unit (service) may be referred just once and as many times as needed (MOHAMMADI and MUKHTAR, 2011). The low computational weight of processing, the weak coupling, and the ability of intercommunication via m are its primary features.

Web - based services are the most frequent designs, with REST and Simple Object Access Protocol (SOAP) being the most used technologies (LI and SVRD, 2010). SOAP is a three - part framework that includes a service provider, a service consumer, and a service registration. The service provider holds the logical unit and its network reference, which allows requests to be made and data to be returned. The service consumer is a programme that has various purposes, one of which is to submit a request to a service, with or without data, and receive a response. The service registry (MUMBAIKAR) is a logical model that includes the addresses of accessible services.

The message model employed in this structure is regarded heavy for transmission, and it is not considered an appropriate structure for applications that require a lot of data interchange or network changes, such as mobile apps. Instead, the RESTFul architecture is widely utilised (WAGH and THOOL, 2012). Nonetheless, Muehlen et al. (2005) clarify that SOAP and RESTFul are not diametrically opposed architectures, but rather systems with distinct goals, because RESTFul is based on a structural style, but SOAP may be utilised in many designs. The REST design, unlike the SOAP model, is considerably less boilerplate and based on verbs and names; moreover, it considers verbs from the HTTP protocol on communication bets. Muehlen et al. (2005), on the other hand, proved that there are more objects, which are represented by URIs, and fewer methods, which are the verbs defined in the HTTP protocol, such as GET, POST, PUT, and DELETE, among others. Figure 2 shows how a GET operation searches for a sequence of data in some format, a POST action transfers an object format to a resource, a PUT operation builds or modifies a resource, and a DELETE activity deletes an existing resource (FENG et al, 2009).

Cloud Computing is the most popular computing paradigm, thanks to service models like Software as a Service and Platform as a Service, among others. Fog Computing is comparable to Cloud Computing, but with a smaller scope and a focus on low - latency applications. Cloud Computing offers ubiquity, real - time network connectivity, and access to a large number of resources, both physical and logical, such as services, servers, storage units, and applications. The Cloud Computing architecture must be built in such a way that its resources are available anytime there is a need for them, without the requirement for contact with any service provider (MELL, 2011).

Wang (2010) proposes a more heuristic definition, stating that cloud computing is a collection of network services with scalability, quality assurance, and service, which is typically personalised, low cost, and on - demand computational structure, and which can be accessed in a simple and ubiquitous manner. Mell (2011) cites the US National Institute of Standardization and Technology's (NIST) five fundamental features of a cloud computing paradigm, which are self - service on demand, broad network access, resource pooling, fast elasticity, and quantifiable service. Hardware as a Service (HaaS), Software as a Service (SaaS), and Infrastructure as a Service (IaaS) are cloud models, according to Mell (2011), Wang (2008), and Spillner (2017). The internet of things (IoT), according to Atzori et al. (2010), is the dispersed presence of online components, people, or equipment, with identity and interconnected from a network. This provides a lot of flexibility, but it also has limits, such as processing capacity and heterogeneity issues, which may be reduced utilising the Cloud of Things idea, as illustrated by Son and Lee (2018). Combining virtualized IoT elements in a cloud structure that provides processing power on demand, as well as the large processing and storage capacity of this structure, could solve several problems, the first of which is to design a centralised structure for receiving and distributing data.

Fog computing, also known as Edge Computing, Cloudlets, Multi - access Edge Computing, or Mobile Edge Computing, is a cloud computing technology that aims to subdivide part of the cloud structure, with part of the processing or storage placed closer to the nodes, which are at the top of the fog structure (MEBREK, 2017). It's thought to be the migration of the cloud environment to the structure's edge (GARCIA et al, 2018). Computation Offloading is a Cloud Computing application, according to Chamas (2018). Processing is transferred from a device, usually a mobile device, to another processing environment due to inadequate processing performance.

However, according to Ye et al. (2016), this processing creates a lot of traffic on computer networks, which increases delay in operations. As a result of being close to the nodes, the data travels a shorter distance and might be fragmented, fog computing is a solution for activities that need low latency. The enterprise service bus is used in this study as a fog computing technology. It combines Service Oriented Architecture with an event - driven approach to enable direct communication between diverse services, as well as translations when appropriate, because the services contained in the bus can be accessed by consumer applications or by the bus itself.

In such a scenario, the integration of technologies, the expansion of computer processing powers, and, on top of that, the storage of these technologies have resulted in a massive creation of unstructured data that is completely scattered in storage. As a result, the goal of this project is to provide a simple tool for acquiring, analysing, and modifying data, using a fog computing system. The following is a breakdown of the work: The Methods utilised to accomplish these goals are explained first, followed by the Results and Conclusions.

2. Methodology

Applied research is the focus of this project. The previous documentary study is intended to define accessible technologies, however the majority of the stages are exploratory, since when requirements engineering is established, an experimental method is required to validate the software's functioning. As a result, not only does qualitative and quantitative work establish whether certain premises are appropriate, but it also defines the performance of the programme under development. The following are the primary approaches utilised in this study.

- This study is divided into many case studies since it incorporates data from a variety of sources.
- Due to the nature of Software Engineering, the following project development stages were implemented (Pichi Junior, 2011):
- A comparison of similar goods on the market (see bibliometric analysis); • A comparison of use (see bibliometric analysis);
- Functional analysis
- Analysis results
- Final structure and idea description
- In terms of requirement engineering (Pichi Junior, 2011), as given in this paper's Results: Product engineering, representation language selection; Process engineering, which entails simulating a process.

A proof - of - principle was carried performed in the first phase, utilising medical data, as detailed in a previous paper (Santos et al, 2019). Following that, a variety of issues were addressed to test various hypotheses, including tiny amounts of data, sensor control, and pattern recognition. Finally, in an unstructured interview, an expert assessed the acquired results. The qualities investigated in the conceptual proofs of this work are the creation of services, interconnected in a bus, to make treatment of scattered and non - structured data, data availability in real time, and analysis of datasets trained to operate in machine learning operations. There hasn't been a final evaluation made yet.

The conversation began with a demonstration of how an Enterprise Service Bus (ESB) may be used to create a Fog Computing framework technologically. The interviewee agreed to try it out, recommending that a series of tests be run with the primary activities taking place in a cloud environment with a lot of data and/or latency issues. Following that, the first proof - of - concept was presented, which involved the use of services on a bus to filter, process, and make available a significant amount of unstructured data. The respondent approved of the test and offered suggestions for how it may be used in future proof - of - concept scenarios.

3. Results and Discussion

The results are provided as bibliometric analysis, function analysis, case studies, and final evaluation are cited in section II. Bibliometric Analysis: Môro (2018) has said that 90 percent of all data created by mankind was generated in the preceding two years, emphasizing the importance of the future. In 2020, 1.7 megabytes will be created per person each second. Because data is created by various equipment and/or programmes, it is stored in a variety of forms in files, databases with various settings, and, in some cases, solely through HTML codes, making access difficult.

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

Key Terms	Results	Type
REST Ful	2.240.000	Independent Terms
BigData	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 Off loading"	4	

Table 2: Measure of Proof - of - Principles

Test	Methodology	Results
1	Query, 5times, directly on the Mongo DB server, a Linux command, with 2 Parameters as a filter, returningapproximately4300 JSON documents	Mean43 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	Mean13 seconds (First execution takes 62 seconds to initialize Apache Spark®)

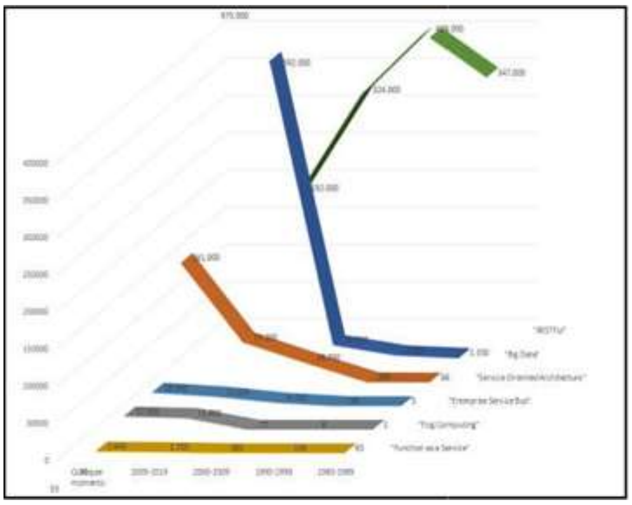


Figure 1: Behavior of the development of technologies for software production, estimated through the number of citations in Google® Scholar

As seen in Figure 1, Google Scholar exhibits a wave - like pattern, with certain terms being replaced by others. As previously stated in the introduction, this phenomena is inherent in the electronic sector, with abrupt shifts being much more prevalent in the field of information technology. The exception is the word "RESTful, " which was already established before the turn of the millennium, but the other phrases have seen a significant growth in attention only in the recent decade, which is compatible with the development of mobile applications. Initially, important phrases were searched separately, and then searches combining the terms were conducted.

Table 1 lists the keywords and the total number of times each phrase appears on Google Scholar. Along with a remarkably low Google appearance of "Mobile - Sensing Technology, " the evolving landscape leads to an even lower quantity if the phrase "Computation Offloading" is connected by the conjunction AND. That is, however, the case if data management is carried out with low - cost instruments such as mobile phones and platforms such as fog computing. It's worth noting that these four occurrences occurred in the previous five years, implying that there has been little effort put towards such growth.
This is owing to the mobile phone's limitations in data processing, which necessitates the usage of cloud computing. On the other hand, it is essential to interact with the device and the data processed in a different context, necessitating the development of low - cost software solutions that are quick and in the form of applications, i. e., as closely as possible to the expected human - computer interaction practises. Content analysis was done because only four publications dealt with computational processing in mobile utilising the principles that works unravel. This study was created utilising SWAT concepts, which means that an attempt was made to check not only the advantages and limits of each technique on these works, but also the effectiveness of each approach.

Li (2018) proposes an IoT system for evaluating sensors, however it does not allow data search that has already been aggregated in databases. Chang (2017) makes a review analysis of systems already proposed in the literature and based on BPMN model (neutral information regarding SWOT analysis). Despite the fact that the author provides useful information, the current study is unique in that it is based on the creation of computational structure. To improve speed, Liyanage (2018) proposed a Platform as a Service solution for portable devices. This project is similar to the one described in this paper. Nonetheless, Liyanage's proposal differs from ours in that ours emphasises the usage of Fog Computing architecture to improve performance as well as the use of Function as a Service. As a result, the current study is supported not only by the small number of computer systems already given, but also by the dispersion of the current systems' methods and modes.

Functional Analysis (Architecture):
The major goal is to establish an Enterprise Service Bus that would act like Fog Computing and interconnect tiny services (that execute specific activities, Function as a Service). A medical dataset was utilised for the proof - of - principles, which was compiled from various files and online sites. Due to the various forms and presentations of the data, two data formatting services were created with the goal of standardising the data in JSON format, which is a character string that separates the fields and can be read by any application. Figure 2 shows how the first service reads the files, determines the format, and then proceeds with the development of the proof - of - principles.

The second service scans the HTML page and scrapes the material that is needed. A service that constantly consulting and verifying whether there is fresh JSON to enter in a non - structured database is connected to the two data formatting services. The variety of formats and potential fields in the data source justify the use of a non - structured database in this situation. A final service connected to the database, which can be used by any third - party programme and comes with performance - enhancing processing tools, allows for filtered database queries.

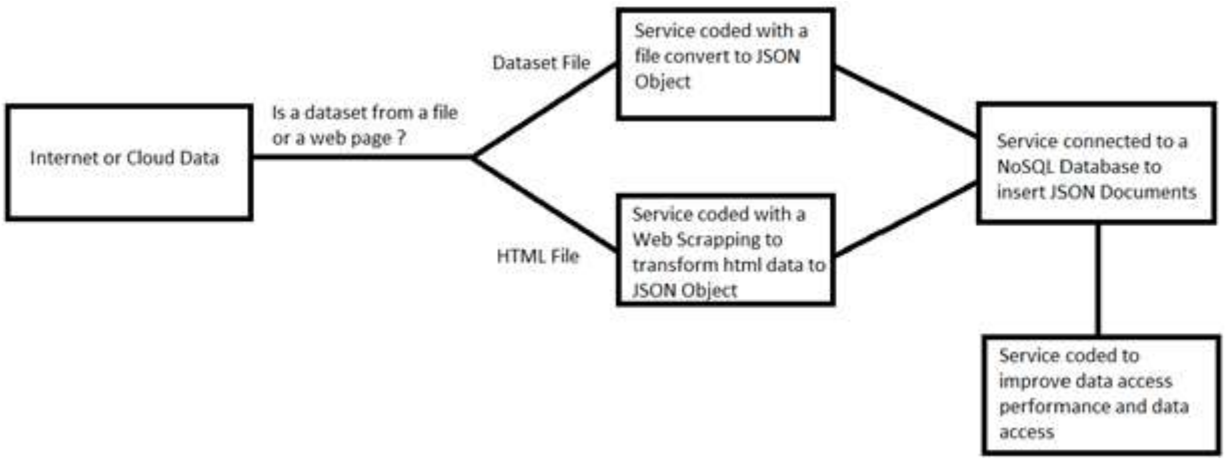

Internet or Cloud Data
Is a dataset from a file or a web page ?
Dataset File
HTML File
Service coded with a file convert to JSON Object
Service coded with a Web Scrapping to transform html data to JSON Object
Service connected to a NoSQL Database to insert JSON Documents
Service coded to improve data access performance and data access

Figure 2: Diagram of proof - of - principles

As illustrated in figure 3, the initial proof of concept is constructed to be linked to the bus, offering sensing services. A service was created to accept data from sensors that measured and regulated the temperature in a room. Five sensors were placed throughout the area to give temperature readings and transmit data to the service. Another service was checking to see whether there was any new data to transmit to a structured database from the prior one. A service that scrapes a website that monitors the temperature in many cities was used to see if the room temperature was affected by external changes. (as well as the city in which the room is located)

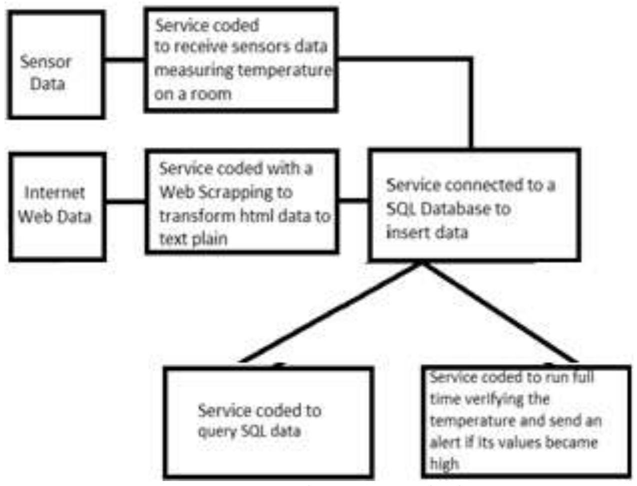

Sensor Data
Service coded to receive sensors data measuring temperature on a room
Internet Web Data
Service coded with a Web Scrapping to transform html data to text plain
Service connected to a SQL Database to insert data
Service coded to query SQL data
Service coded to run full time verifying the temperature and send an alert if its values became high

Figure 3: Diagram of First Proof of Concept

The answer is validated in greater computational processing in the second proof of concept. In a cloud computing environment, a machine learning training model for identifying components in pictures was implemented. Figure 4 shows how a service was connected to the ESB in order to get 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 pictures and uses a machine learning model to see if elements in the image can be identified.

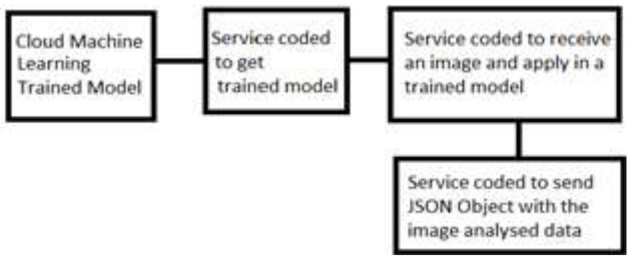

Cloud Machine Learning Trained Model
Service coded to get trained model
Service coded to receive an image and apply in a trained model
Service coded to send JSON Object with the image analysed data

Figure 4: Diagram of Second Proof of Concept

A service that receives pictures and uses a machine learning model to see if elements in the image can be identified. A third service is also on the bus, and it connects with the previous one, searching for a JSON object containing the components specified in the picture and providing them to third - party applications. The third proof of concept is identical to the proof of principles, but with the exception of the structured data source. A source of Brazilian data on persons who have been tested for Sars - Cov - 2 exposure has been discovered. Because this source is in CSV format, it is simple to construct a service that analyses the file and filters the relevant data. A service connected to a structured database, as shown in Figure 5, requests data from the above - mentioned service and provides it to the database. Finally, the bus was connected to a service that distributes filtered data from the database to third - party apps.

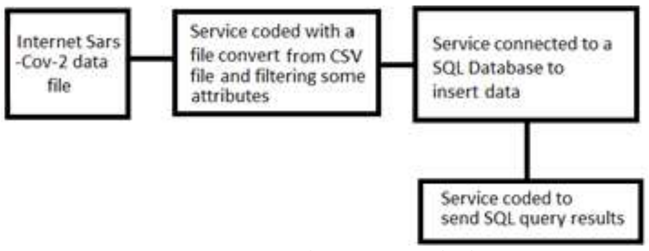

Internet Sars -Cov-2 data file
Service coded with a file convert from CSV file and filtering some attributes
Service connected to a SQL Database to insert data
Service coded to send SQL query results

Figure 5: Diagram of Third Proof of Concept

4. Case Study

Software language, structure, and architectural definitions are the initial stage. Due of the database, it's called Fog Computing. Two services, both linked to the bus, were created to work with the data. One that provides data that may be ingested by third - party apps, and another that monitors values and notifies the person in control of the room if there is a problem.

The initial stage in bringing the operation closer to the network's edge was to establish the system's linguistic structure: Service Oriented Architecture Defined on a Service Bus Corporation. Following that, it's necessary to explain what sorts of services should be modelled and built, as well as the connection between them and the potential applications that would use the services (FaaS) and how the data would be preserved.

Data Search, Data Decoding, Data Transformation, Receiving data from devices such as sensors, and Data Persistence were the services chosen. A data search service is required to filter data that is spread throughout the cosmos. Because the data is accessible in a variety of formats, decoding and data transformation services are necessary. For improved efficiency, the answer is to define a single database structure. The term "data persistence" refers to the preservation of both incoming and modified data. Reading from devices such as sensors is intriguing because, once the universe has been established, massive amounts of data may be immediately sent to the system database.

Data is stored in NoSQL MongoDB because storage resources are optimised in this format (Oliveira et al., 2018). This option makes a big impact when dealing with enormous amounts of data. SQL is more efficient when it comes to querying. A service was also offered that reads data sources and inserts them without the requirement for human involvement, reducing the complexity of data entry. The elsewhere described proof - of - principle Santos et al (2019) took this method, and as a result, a service bus was created. Because it is built on the FaaS idea, the parameters must be defined in order to provide its particular functionality.

The data search service takes a logical address containing files, determines which files have relevant data, and provides a data structure in list format with absolute addressing for each file and its format. The data decoding service takes this list, reads each file, and converts the information into plain text.

The data transformation service takes a large amount of unformatted text and attempts to convert it to JSON, the NoSQL database's preferred format. The data persistence service may 1) accept several JSON documents and store the data in a NoSQL database, delivering just one HTTP code message. The success of the procedure is indicated by this message. Data extraction from public healthcare files was used as a proof - of - principle. These files contained data from nearly a 13 - year span, were in various formats, and were organised according to the consolidated period. As a consequence, there were more than 30 million JSON documents in the database, each having an average of 25 characteristics. After going through the services mentioned in the functional analysis, Table 2 shows the results of the consultations to the database in the ESB using (a) without the service that improves the consultations and (b) with the service that improves the consultations. The next three case studies, as said, are based on the suggested design, which offers numerous advantages. The enterprise service bus was designed, and proof - of - concept tests were conducted to evaluate its key features. The initial proof of concept was completed using a service that analyses unstructured medical data files with sensitive material. Sensors were put in a room in the initial proof of concept, with the requirement for temperature monitoring and comparison with the outside temperature. The bus features a service that gets data from sensors via the network in real time, as well as a service that connects with a Web Service that is likewise updated in real time with the city's temperature. When one or more sensors detect temperature changes, a communication system is activated.

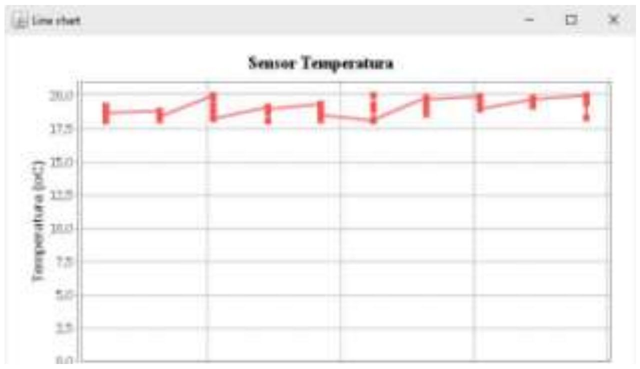


Figure 6 (a): Third party Java SE application

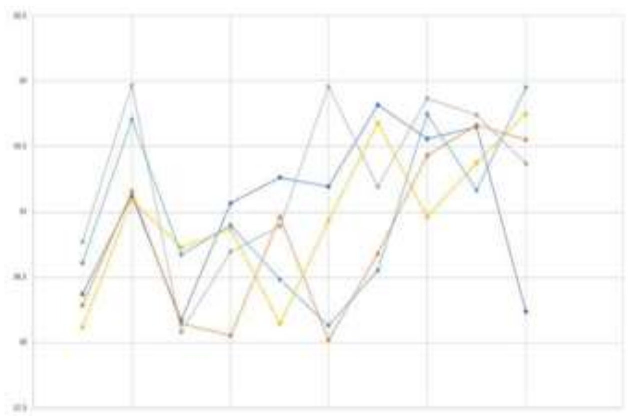


Figure 6 (b): MS Excel® consuming data

Figure 6 shows third - party programmes that use sensor data to produce graphs, (a) in a JavaSE application and (b) in a Microsoft Excel® application. The second proof of concept developed, features a machine learning mechanism where a dataset housed in the cloud creates machine learning for the identification of daily components in photos.

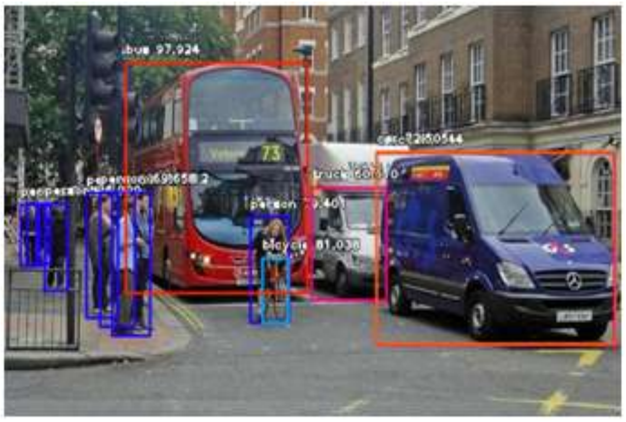


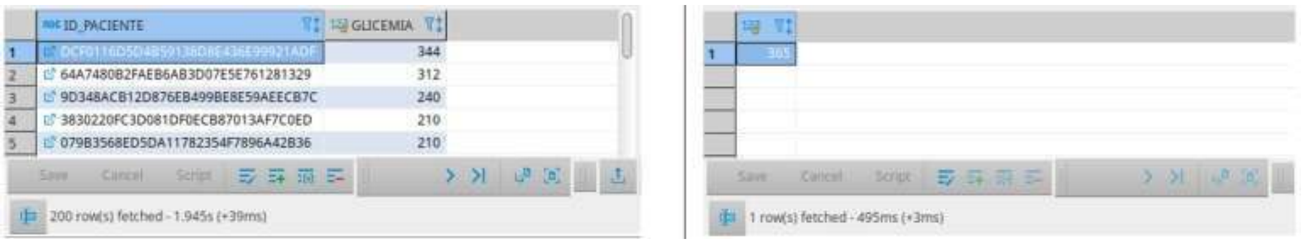
Figure 7: Internal machine learning processing with service data generated by third party request

The training model is delivered as a service that identifies pictures and, if a positive result is obtained, transmits additional data to the cloud to expand the dataset and enhance training, therefore boosting accuracy.

Figure 7 shows an internal machine learning processing and a service out, both of which were taken from the internet. Despite the fact that the picture is generated, as seen in Figure 7, the service turns it into a JSON format output including the image contents. The following JSON was generated by the image:

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

The final proof of concept used a dataset of roughly 3 million people's tests for contamination detection using Sars - Cov - 2.



	ID_PACIENTE	GLICEMIA
1	DCF0116D5D4B59136D8E436E99921ADF	344
2	64A7480B2FAEB6AB3D07E5E761281329	312
3	9D348ACB12D876EB499BE8E59AEECB7C	240
4	3830220FC3D081DF0ECB87013AF7C0ED	210
5	079B3568ED5DA11782354F7896A42B36	210

200 row(s) fetched - 1.945s (+39ms)

1	365

1 row(s) fetched - 495ms (+3ms)

Figure 8: Service response to a third – party

The data is stored in the cloud and is structured, but it is only accessible internally in an unusual format. A data collection service linked to a relational database was prepared and brought the data into a database that had previously been filtered to meet the analyst's requirements. A database filtering service enabled a one - dimensional examination of the data, allowing inferences about contamination linked to previously established comorbidities to be reached. On the investigated base, the findings are achieved in seconds or fractions of seconds. A third - party application that uses the service is shown in Figure 8.

Final Assessment
In the first proof of concept, it is feasible to demonstrate the capabilities of a Fog Computing environment applied to Enterprise Service Bus to function in real time, decreasing time in real - time monitoring operations, because equipment can burn if the sensors fail. The second proof of concept shows how the fog environment may communicate directly with the cloud environment, allowing for easier access to data and response to the cloud data.

5. Conclusion

This research proposes the use of low - cost, high - resource technology to develop a system that allows for the search, decoding, and storage of linked data that is dispersed and in many forms. This method includes a novel computational framework that permits decentralised and non - structured data to be captured for subsequent processing. This recovered data, which may be structured in a network or made available by sensors such as those found on smartphones or in microsensors, can be accessed with little resource usage and delay. Other services that the system can provide include data analysis, treatment, and transformation.

Fog Computing is used in the system, which is a future technology frontier. Some of the qualities expected in a fog computing environment are real - time access from processing data, intercommunication with cloud computing, and low latency dealing with huge amounts of data, as seen in the three case studies addressed in this paper.

References

[1] Arachchige W., Jayathilaka D. M., Qi K., Qin Y., Chinnappan A., SerranoGarca W., Baskar C., Wang H., He J., Cui S., Thomas S. W., Ramakrishna S., SerranoGarca W., Baskar C., Wang H., He J., Cui S., Thomas S. W., Ramak 2019. Advanced Materials, Volume 31, Issue 7, 1805921 - 1805942 https: //doi. org/10.1002/ adma.201805921 Nanomaterials in Wearables: A Review on Wearable Actuators and Sensors
[2] L. Atzori, A. Iera, and G. Morabito.2010. A survey on the internet of things. Computer networks, vol.54, no.15, pp.2787 - 2805.
[3] C. L. Chamas, C. L. Chamas, C. L. Chamas (2018). Energy consumption in Android mobile devices: an analysis of communication strategies used in Computation Offloading. (Dissertation Master's) USP, So Paulo.
[4] C. Chang, S. N. Srirama, and R. Buyya. A Survey of Mobile Cloud Business Process Management Systems for the Internet of Things.49th Annual ACM Computing Surveys (CSUR) (4).
[5] A. Falamarzi, S. Moridpour, and M. Nazem. A Review of Existing Railway Infrastructure Inspection Sensors and Devices 1 - 10 in Jurnal Kejuruteraan, 31 (1).
[6] X. Feng, J. Shen, and Y. Fan.2009. A Web services architecture alternative to RPC. The First International Conference on Future Information Networks was held from July 7 to 10, 2009.
[7] Garcia, J., Simó, E., Masip - Bruin, X., Marn - Tordera, E., and Sànchez - López, S.2018. Garcia, J., Simó, E., Masip - Bruin, X., Marn - Tordera, E., & Sànchez - López, S. Is Cloud Really Necessary? Estimating the City of Barcelona's Fog Computing Capabilities International Conference on Utilities (IEEE/ACM)
[8] REST - based SOA application in the cloud: a text correction service case study, Li, W., and Svärd, P.84 - 90 in the 6th World Congress on Services in 2010.
[9] W. LI, B. Wang, J. Sheng, K. Dong, Z. Li, and Y. Hu. ‘A Transparent Computing - based Resource Service Model for the Industrial IoT System. Sensors.
[10] M. Liyanage, C. Chang, and S. N. Srirama. Mist computing in the Internet of Things requires an adaptive mobile Web server framework.14 (3/4), 247 - 267 in International Journal of Pervasive Computing and Communications.
[11] J. L. Maréchaux, J. L. Maréchaux, J. L. Maréchaux, J. L. Maréchaux, J. L. Maréchaux, J. L. Maréchaux, J. L. Maréch 1269 - 1275
[12] P. Mell and T. Grance.2011. Cloud computing as defined by the National Institute of Standards and Technology (NIST).
[13] Mukhtar, M., and Mohammadi, M.2011. Supply Chain Management SOA - based business procedure.213 - 216 in 2011 Malaysian Conference on Software Engineering.
[14] D. K. Môro, D. K. Môro, D. K. Môr (2018). Entidades Nomadas em Documentos de Lngua Portuguesa Reconhecimento (Ed. T. d. Curso) Universidade Federal de Santa Catarina, Araranguá.
[15] M. Z. Muehlen, J. V. Nickerson, and K. D. Swenson (2005). The case of REST vs. SOAP in developing web services choreographic standards. Decision Support Systems, vol.40, no.1, pp.9 - 29.
[16] Web services based on soap and rest concepts, S. Mumbaikar and P. Padiya, 2013.3 (5) of the International Journal of Scientific and Research Publications
[17] Nag, A., Mukhopadhyay, S. C., and Kosel, J.2019. Nag, A., Mukhopadhyay, S. C., and Kosel, J. Fabrication, Characterization, and Implementation of Printed Flexible Sensors Springer.
[18] H. Nazemi, A. Joseph, J. Park, and A. Emadi. Advanced Micro - and NanotechnologiesSensors, 19, 1285 https: //doi. org/10.3390/s19061285 Gas Sensor Technology: A Review
[19] M. S. Oliveira, N. F. Melo, and L. C. Santos.2018. Banco de Dados SQL vs. No - SQL Banco de Dados Journal of the South American Development Society, 4 (11), pp.298 - 320.
[20] W. Pichi Junior, Pichi Junior, Pichi Junior, Pichi Junior, Pi (2011). Construço de Protótipo para Ensino Tecnológica: Cromatografia em Estudo de Caso (Dissertation Master's) Paula Souza Centro, So Paulo.