Revista Brasileira de Computação Aplicada, July, 2023

DOI: 10.5335/rbca.v15i2.14234
Vol. 15, Nº 2, pp. 105–119
Homepage: seer.upf.br/index.php/rbca/index

ORIGINAL PAPER

DASTData: a Fog-Cloud model for distributed storage and traceability of IoT data in Smart Cities

Daniel Lopes Ferreira [1], Rodrigo da Rosa Righi [1], Vinicius Facco Rodrigues [1]

[1]Programa de Pós-Graduação em Computação Aplicada - Universidade do Vale do Rio dos Sinos (UNISINOS)

*danielferreira17@edu.unisinos.br; rrrighi@unisinos.br; vfrodrigues@unisinos.br

Received: 2023-01-10. **Revised:** 2023-05-04. **Accepted:** 2023-07-11.

Abstract

In this work, a solution for the distributed storage of data in Smart Cities is presented. An Edge-Fog-Cloud architecture that partitions the data through the Sharding technique proposes a hierarchical model that manipulates IoT data generated by Smart Cities. The problem is related to the approaches used to promote an integrated environment. Related works tend to use cloud-focused approaches that generate high latency rates, and those that use Fog Computing only use the layer as *middleware*, not exploring greater possibilities for use. In this context, this work presents the DASTData model that aims to enable lower latency rates, more data security, fault tolerance, high availability, and concurrent queries to promote a better experience in data management and availability in smart cities. In addition, our contribution to the literature is related to the proposition of an architecture focused on enabling the traceability of users who have mobile behavior in the city, providing the ability to analyze patterns and occurrences through the consolidation of data from one or more individuals. In the results obtained through the tests carried out in this work, we observed that in queries DASTData is up to 73% more efficient.

Keywords: Cloud Computing; Data Sharding; Data Traceability; Distributed Storage; Fog Computing.

Resumo

Neste trabalho é apresentada uma solução para armazenamento distribuído de dados em Smart Cities. Através de uma arquitetura Edge-Fog-Cloud que particiona os dados através da técnica Sharding, é proposto um modelo hierárquico que manipula dados IoT gerados por Smart Cities. O problema está relacionado com as abordagens utilizadas para promover um ambiente integrado. Trabalhos relacionados tendem a utilizar abordagens focadas em nuvem que geram altas taxas de latência, e os que utilizam Fog Computing utilizam apenas a camada como *middleware*, não explorando maiores possibilidades de uso. Nesse contexto, este trabalho apresenta o modelo DASTData que visa possibilitar menores taxas de latência, mais segurança de dados, tolerância a falhas, alta disponibilidade e consultas simultâneas para promover uma melhor experiência no gerenciamento e disponibilidade de dados em cidades inteligentes. Além disso, nossa contribuição para a literatura, está relacionada à proposição de uma arquitetura focada em possibilitar a rastreabilidade de usuários que possuem um comportamento móvel na cidade, proporcionando a capacidade de analisar padrões e ocorrências através da consolidação de dados de um ou mais indivíduos. Nos resultados obtidos através dos testes realizados neste trabalho, observa-se que em consultas o DASTData é até 73% mais eficiente.

Palavras-Chave: Armazenamento Distribuído; Computação em Neblina; Computação em Nuvem; Fragmentação de Dados; Rastreabilidade de Dados.

1 Introduction

With the urban population growth in the great centers, *Smart Cities* become an ever closer reality. The possibility

of integration between the various areas of a city brings the expectation of improvements in public management in all its attributions, such as mobility, health, education, and infrastructure (Lai et al., 2020). Integrating the generated data is also a desire of the users, who can keep active control of their health, schedule, and home to improve their productivity and have more time and quality of life. However, the tasks of storing and querying this data need to be as performant and transparent as possible.

Data management is the critical factor for the success of an intelligent ecosystem (Benhamida et al., 2022). This is because smart cities, from their sensors, generate a massive amount of data (Sasubilli et al., 2020) that needs to be stored efficiently to guarantee better performance in the processing and analysis of this data. In this context, it is noted that dealing with data at the edge of the network reduces latency, increases security, and reduces the load on the network when compared to a distant model such as the one based only on Cloud Computing (Vilela et al., 2020). One way to implement this manipulation closer to the data is through clusters (or nodes) of Fog Computing (Naeem et al., 2019) that are distributed in the city and are more efficient in the data processing.

Furthermore, Relational Database Management Systems (RDBMS) are complicated or add more complexity in this application context, requiring greater flexibility. On the other hand, Non-Relational Databases (NoSQL) have flexibility in the data schema as their native characteristic. They were born to be easily distributed and scaled from techniques such as Sharding (Abdelhafiz and Elhadef, 2021) that partitions data horizontally among several database instances.

This work is linked to the Minha História Digital (My Digital History) project. It aims mainly at the proposition and implementation of a storage architecture in hierarchical nodes in the Edge-Fog-Cloud model for smart cities, focusing on the storage and traceability of vital data generated through IoT sensors. This architecture should promote geographic proximity to clients, reduced latency, increased security, reduced data flow, fault tolerance, concurrent queries, and easy scalability of resources. Some architectural propositions were analyzed to support the concepts and approaches already studied, seeking to find possible gaps in the literature.

After analyzing the related work (Kudo, 2018; Sinaeepourfard et al., 2018; Shwe et al., 2016; Abreu et al., 2016; Lomotey et al., 2018; Zhang, 2020; Benhamida et al., 2022), it was observed the need to propose an architecture that meets all the aforementioned requirements, seeking to fill gaps in the literature related to the ability to trace user data dispersed in distributed architectures and the reduction of connection latency between services and IoT devices. In this context, this article presents the DASTData model, an acronym for *Distributed Architecture to Store and Trace Data*, which mainly addresses how to store and trace data in complex and distributed architectures such as those of Smart Cities through the joint use of technologies such as Sharding and concepts like Fog and Cloud Computing. With the proportions that a smart city architecture can take and observe the distributed storage problems that need to be solved in this context, the present work aims to develop a data storage and traceability model for smart cities. Our contribution can be summarizes as follows:

i. Proposal of a fog-cloud model that uses the sharding technique for efficient distribution and traceability of IoT data in smart cities.

This work is organized into seven sections. In Section 1, the work's problem and justifications are contextualized, as well as the objectives of the present research. Section 2 defines the concepts relevant to the model's proposition. In Section 3, we expose the methodology for selecting the works that are directly related to this research. We also detail relevant aspects of each one to check for gaps in the propositions. In Section 4, we quote the design decisions taken in light of the related works' positive and negative points. We also define and present the proposed architecture to solve the verified problems. Moving on to Section 5, we discuss how the model evaluation methodology was defined to prove its efficiency. In Section 6 we present some preliminary results obtained through the implementation of the prototype. Finally, in Section 7, we present a summary of what was exposed in Section 4, we also discuss the expected contributions of the proposed architecture and comment on future works.

2 Fundamentals

This section addresses the concepts cited in this work to clarify and support the research developed. From the description of Cloud Computing, Fog Computing, the Internet of Things, Smart Cities, and Data Fragmentation, it will be possible to advance in the research by keeping the concepts updated with previous studies that define these subjects.

2.1 Internet of Things

Internet of Things is a concept directly related to the transparent interaction between the devices of our daily lives in an M2M[1] model (Patel et al., 2016). Each of these devices is located at a level called *Edge* of the network as they are very close to end users. They generate a massive amount of data at all times through small sensors, wearable devices, smart cameras, or even devices that mediate communications such as *gateways* and *middlewares*, among others, depending on the high-level architecture shown in Fig. 1.

An application of IoT technologies can be observed in the smart bracelets that are increasingly popular today. They have smart sensors capable of monitoring vital signs such as heart rate, blood pressure, and blood oxygenation and send this data to the owner's phone. Later, some smartphone applications can perform analysis and trigger alerts on the device, or it can automatically send this data to an external service that will run Artificial Intelligence

[1]Machine to Machine (M2M): a concept that describes the communication between machines, sensors, and devices in general with no or minimal human intervention.

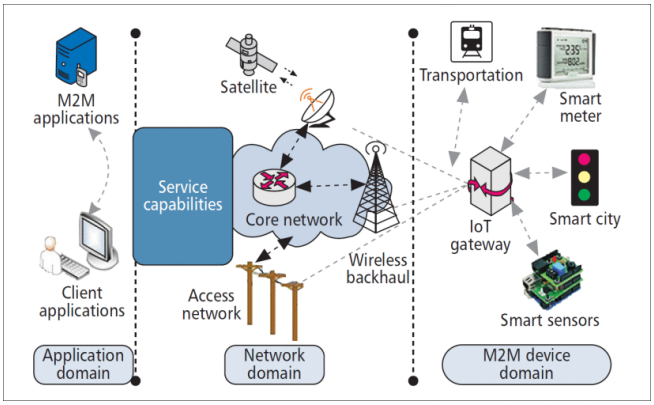

Figure 1: High level IoT architecture.

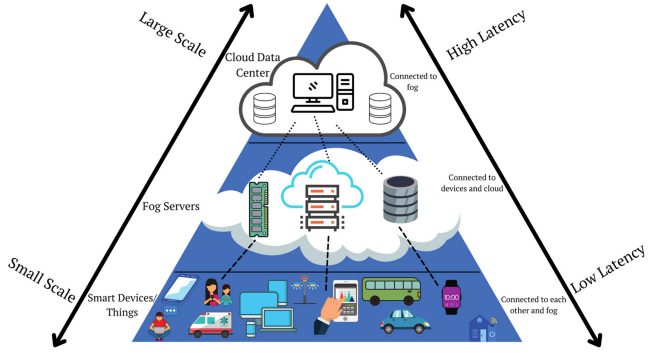

Figure 2: Hierarchical Edge-Fog-Cloud architecture that exemplifies device scale and latency relationships depending on proximity to users at each tier.

algorithms and make predictions about the user's health, for example.

There are significant challenges when we talk about *IoT* devices integrated into people's daily lives in smart ecosystems (Chen et al., 2014). Smart cars can interact with smart semaphores to improve traffic control. Instant payments can be made by walking into an establishment, picking up a product, and walking out the door, all integrated with a digital wallet. Finally, the applications are diverse, such as cities, hospitals, homes, supermarkets, and other intelligent services that interact with each other through the internet to facilitate people's daily lives.

2.2 Cloud and Fog Computing

Cloud Computing (Mell and Grance, 2011), is a concept that applies to network resources, servers, storage, applications, and on-demand services made available through the internet to different customers with little management effort and reduced cost.

As a new paradigm, it brought some innovations to computing. Providers can share resources with different users who pay only for what is used (*pay-as-you-go* system), not generating wasted resources and allowing the ideal service to be acquired for each need. Furthermore, resources are not necessarily close to users and other services but relatively centralized in large *data centers*. This feature can be beneficial in terms of costs and maintenance.

Although it is a very beneficial technology for many cases, there are some open problems regarding smart cities and *IoT* devices. The centralization of *data centers* is a problem for applications sensitive to high demand (Vilela et al., 2020), as network congestion or large geographic distances can generate critical latency for these systems.

Therefore, with the growing number of *IoT* devices, there was a need for an intermediate layer to communicate with services in the Cloud (Naeem et al., 2019). From this, the concept of Fog Computing emerges as a layer that allows processing prior to *Cloud*, closer to edge devices. With particular functions, it provides better response times, lower infrastructure costs, more performance, better scalability, and more security over data.

Fog is an emerging concept that seeks to complement the use of the Cloud. In other words, the purpose is not to be a replacement but a new layer, a complement (Bonomi et al., 2012). Fig. 2 shows that Fog is closer to users connecting with Edge devices and Cloud data centers.

This proximity allows for lower latencies in interactions between clients and servers, as well as reducing the amount of data sent to the Cloud, as Fog performs trivial processing for the devices, and only essential requests are sent to the applications in the Cloud (for consolidation of information or heavier processing, for example). In addition, a Fog approach improves performance in responding to devices (Vilela et al., 2020), as the workload per server is significantly reduced, which reduces the chances of overloading a single point in the network.

2.3 Smart Cities

Due to the accelerated population growth of urban environments, we must have more cities with efficient locations and services to serve citizens. The concept of Smart Cities arises from this need, which is complex systems, often called "systems of systems" (Khatoun and Zeadally, 2016), because they deal with integrating different devices and services. As illustrated in Fig. 3, most intelligent city models have six components: government, economy, mobility, environment, housing, and people.

However, there are some latent challenges when dealing with smart cities, and most of them are linked to scalability and QoS[2] of the model, as they deal with integrations between critical and high-priority systems, such as, for example, transit traffic (mobility component) and public health management (housing component).

The treatment and storage of data are one of these challenges since, in a smart city, data integration is the key to more efficient management. In addition, the amount of data generated in Smart Cities (Chang, 2021) is enormous. Understanding, storing, and retrieving this data in the

[2] *Quality of Service (QoS)*: concept applied in networks that prioritize high-performance applications and that concern them. Suitable parameters related to bandwidth, latency, packet losses, and *jitter* (latency variation).

best possible way requires a solid, scalable, and secure architecture. One of the data sources of smart cities is IoT devices (Section 2.1). The key to a robust architecture that can meet it can be found in Fig. 2 from the union between these devices (on the Edge) with Cloud and Fog Computing (Section 2.2).

2.4 Data Sharding

In a relational database (RDBMS) data is usually semantically organized into tables, columns, and rows. Rows are the stored records, columns are the attributes of those records, and tables are the groupings of that data into a standard set. This robust structure has met several academic and market solutions for years. However, there is a particular difficulty in dealing with this architecture regarding Big Data, distributed storage, high performance, and concurrency.

For this purpose, non-relational databases (NoSQL) emerged, which were conceived with the idea of having flexible schemas, being scalable and prepared to meet the architecture that requires high availability and storage performance (Al Jawarneh et al., 2021). Furthermore, when we talk about these types of databases, a concept that comes to the fore is Data Fragmentation or *Sharding*. Sharding is not a new term, nor is it unique to NoSQL, but it has become popular because most NoSQL databases bring this feature and apply it natively in a simplified way.

The concept deals with the technique of dividing, based on an established criterion, a group of data into different subgroups (Corbellini et al., 2017) located on nodes of the database network. There are different ways to shard the data using Sharding. An example would be the division using an equality criterion of item quantity per node, always trying to keep the same quantity in each to provide the load balance between them. Another possible application would be the definition of a division criterion from one or more fields of the stored data, and ranges or exact values are defined per node.

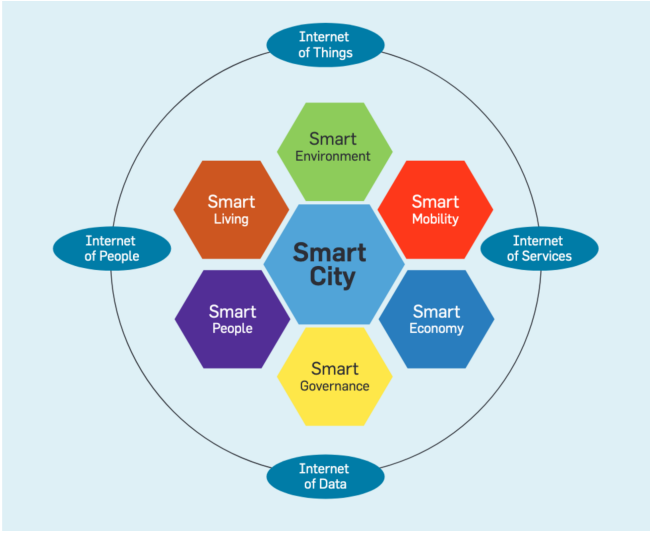

Figure 3: Components of Smart Cities

Due to their dynamic structure, NoSQL databases usually have the Sharding feature. Some of them even offer more robust solutions for application in a distributed storage architecture that bring numerous advantages, such as the possibility of concurrency to solve a query, proximity between clients and persisted data, data replication to improve architectural fault tolerance, and even sharded cache per split unit.

3 Related Work

This section details the related work linked with the research we are conducting. The selection of articles was made by searching reliable databases such as IEEE Xplore[3], SpringerLink[4], ScienceDirect[5], MDPI[6] and ACM Digital Library[7] and aimed primarily at finding related articles that focused on distributed storage architectures, layered hierarchical architectures, and traceability of data distributed in smart cities that have an architecture composed of Fog and/or Cloud. The search and analysis resulted in 7 articles, according to Table 1, considered adherent to one or more relevant aspects of this research and that directly contribute to the basis of the work to be developed.

Table 1: Selection of Related Works

Database	Work	Title
IEEE	Kudo (2018)	Fog computing with distributed database
IEEE	Sinaeepourfard et al. (2018)	Data preservation through fog-to-cloud (F2C) data management in smart cities
IEEE	Shwe et al. (2016)	An IoT-oriented data storage framework in smart city applications
Springer Link	Abreu et al. (2016)	A resilient Internet of Things architecture for smart cities
Springer Link	Lomotey et al. (2018)	Traceability and visual analytics for the Internet-of-Things (IoT) architecture
Science Direct	Zhang (2020)	Design and application of fog computing and Internet of Things service platform for smart city
MDPI	Benhamida et al. (2022)	Dynamic architecture for collaborative distributed storage of collected data in fog environments

In (Kudo, 2018) the authors proposed an architecture for storing data generated by *IoT* devices. Due to the large amount of information that these sensors can generate, it is proposed that this data be pre-stored at the *Fog* level in non-relational Databases due to its high scalability capacity and the dynamic data structures supported by this model. Furthermore, these databases work very well in a distributed way, which for a *Smart City*, for example, is highly positive, as it allows users to be geographically

[3]IEEE Xplore: https://ieeexplore.ieee.org/
[4]SpringerLink: https://link.springer.com/
[5]ScienceDirect: https://www.sciencedirect.com/
[6]MDPI: https://www.mdpi.com/
[7]ACM Digital Library: https://dl.acm.org/

close. In this work, MongoDB was used through the GridFS feature, which allows storing documents more significant than 16 MB (MongoDB document size limit) and access to data portions without needing to load the entire registry. This feature allows for storing large amounts of data because instead of storing all the data in a single document, it divides it into parts.

Also, in this article, the author proposes a storage model based on three levels. The first stores the original sensor data in *Fog*. The second and third levels are located in the Cloud. At the second level, data extracted from primary processing is stored. Moreover, the third level stores the results of the extracted data analysis. This architecture provides data access to be performed both through the Cloud and directly on the Fog nodes but is not concerned with user mobility.

In (Sinaeepourfard et al., 2018), the work's concern with data lifecycle management is notable, which links it directly to this research. The proposed model was successfully validated in Barcelona, Spain, and is concerned with three main phases covering the entire data lifecycle: acquisition, processing, and preservation. These phases were proposed for a Smart City that uses a three-layer architecture: Fog layers 1 and 2 and the Cloud layer. The first layer, Fog 1 or Edge, is the layer closest to the users who interface the users and their devices with Fog 2. The second layer, Fog 2, is an intermediate layer between the Cloud and Fog 1 and is responsible for processing and storing data with a higher access rate. The third and most superior layer, the Cloud, is the point furthest from the users, which generates more latency. It is responsible for storing historical data and performing heavier processing.

The work (Shwe et al., 2016) proposes a distributed storage framework that is structured in two layers. The first layer is a mesh backbone network that stores real-time system data in a distributed manner. The second layer is responsible for central storage. Both layers were designed for the Cloud, but the work does not exclude the possibility of implementing some services, such as Fog, in the first layer (mesh). Each node (or Access Point) in the first layer has its own storage network and contributes to the formation of a mesh network where each node has access to the data of the others. Recurring access and latency-sensitive data are stored in the first tier, and historical and long-term data is sent to the upper tier through aggregation routines running in the background.

In (Abreu et al., 2016) the main objective is the concern with the resilience and fault tolerance of an IoT architecture for Smart Cities. This research was applied to the city of Lisbon in Portugal and implements services capable of providing dashboards that assist in decision-making by government institutions and allow, through data analysis, the prediction and performance of routines in the city without human intervention. The proposed model is divided into three layers (IoT, Fog, and Cloud), and in each layer, multiple instances of microservices are implemented. In this way, the unavailability of a single service does not disable the others, and the redundancy of instances and load balancing ensures greater availability of the executed applications. The work reaffirms the importance of the Fog-Cloud union for workload distribution and latency reduction and places this structure as a critical factor for a resilient architecture in intelligent cities.

In (Lomotey et al., 2018), the authors proposed a multilayer IoT architecture model that facilitates traceability through cloud-based storage that records device information and metadata of requests made. On-demand, the metadata repository must be requested to respond to the tracking of specific data between devices on the network. In addition, through visualization tools, the proposed methodologies allow for determining the link between IoT devices in order to understand the flow of data on the network. This architecture seeks to solve the problem of correlation between different data from the same users since the mobility in the Smart City network and the different IoT devices generate massive data. However, as it is a Cloud-based model, recording and writing latency is challenging.

In (Zhang, 2020), the advantages of the collaborative use of Fog and Cloud Computing to handle IoT data in Smart Cities were emphasized. The proposed architecture is based on a layered model that handles the data coming from IoT devices in the Fog Computing layer and later in the Cloud Computing layer. Each node in Fog has some virtualized services that perform aggregation and consolidation operations. When there is overhead (or identification of possible overhead), the data in the Fog is forwarded to the Cloud. Although this strategy increases, latency can benefit the model as a whole. The proposal was made with cities in China in mind, a country with an accelerated urbanization process. In future works, the concern with optimizing the allocation of computational resources for the Fog has been brought up, as this layer has limited resources by hardware.

Finally, in (Benhamida et al., 2022) the authors proposed a dynamic architecture for distributed storage of information collected in Fog environments. Dynamicity is related to the inherent mobility of Fog and IoT devices and distribution with data storage in Fog nodes. This research also addresses a collaborative model between Fog and Cloud to minimize access to the Cloud and, consequently, reduce latency. The architecture is hierarchically divided into three layers: Fog Edge (with Fog Nodes), Fog Server (with Fog Servers), and Cloud. In short, the model is responsible for locating the records through local storage on each Fog Server periodically updated in "assignment tables" that link the IoT sensor to the Fog Node. In addition, authorization requests are synchronous, and data and assignment tables are replicated at each tier.

The works were analyzed among themselves through nine crucial competencies for the context of a distributed architecture of data storage in Fog and Cloud Computing in Smart Cities in order to allow the traceability of information and users, have a resilient system, and achieve low levels of latency (according to Table 2). The characteristics analyzed are:

- Traceability (C1) — data is easily traceable from requests made to any node in the network.
- Interoperability (C2) — the different architecture services are easily integrated between different layers and nodes to make communication standardized and transparent.

Table 2: Comparison between selected articles.

Work	C1	C2	C3	C4	C5	C6	C7	C8	C9
Kudo (2018)	N	Y	Y	Y	N	N	N	Yes, data in the Fog is later extracted to the Cloud	Yes, parses data pre-processed by Fog
Sinaeepourfard et al. (2018)	N	Y	Y	N	Y	N	Y	Yes, data in Fog is data that needs real-time analysis, or that can withstand little delay	Yes, it does massive processing and complex analysis
Shwe et al. (2016)	N	Y	Y	N	N	Y	N	Yes, mesh network that distributively stores real-time data	Yes, central storage of historical and long-term data
Abreu et al. (2016)	N	Y	Y	N	Y	Y	Y	Yes, pre-process data to handle heterogeneity and be processed in Cloud services (or some in Fog)	Yes, services layer and IoT middleware, however, both layers can take part in Fog and Cloud to handle specific demands
Lomotey et al. (2018)	Y	Y	N	N	N	N	N	No, it focuses on traceability between IoT devices and the Cloud	Yes, data analytics services are in the Cloud
Zhang (2020)	N	Y	Y	N	N	N	N	Yes, Fog is in a single horizontal layer that handles critical latency demands	Yes, the Cloud processes more complex data that was not able to be processed by Fog
Benhamida et al. (2022)	Y	Y	Y	N	N	Y	N	Yes, it has two layers of Fog: one closer to the user at the edge of the architecture and another one further up that performs the processing. Data is replicated at each tier, and there is a local assignment table at each node	Yes, the Cloud performs the most demanding and historical data processing. It also has a copy of the attribution table and also has the data of the lower layers replicated

- Hierarchical (C3) — the proposed architecture has a hierarchy or some division into layers to better divide responsibilities and load, facilitating the management of services.
- Has Data Fragmentation (C4) — data is stored in a fragmented way allowing for distributed storage and concurrent queries to different portions of one or more data.
- Has Cache (C5) — has some caching system that improves response time and avoids unnecessary I/O operations for previously queried information.
- Has Replication (C6) — has some form of redundancy of the proposed services and stored data to be a fault-tolerant architecture and have a better QoS.
- Validated in some real city (C7) — the proposed architecture has already been validated in a real city or simulation that reproduces it.
- Has Fog (C8) — the architecture encompasses Fog Computing nodes close to the devices and/or at the edge of the network.
- Has Cloud (C9) — the architecture encompasses communication with cloud services that are not necessarily close to the devices.

Analyzing Table 2, it is possible to see that all jobs are processed in Cloud Computing and that, in most cases, this layer is responsible for processing long-term data that is extracted from the nodes further down, but that it also responds for immediate requests, which brings higher latencies to the model since there is no geographic proximity in this case. As for Fog Computing, most works have a Fog layer that, in some cases, is optional or with unexplored potential, being mentioned only as a possibility and not addressed as the main application of the model. Herefore, it has many advantages. The use of intermediate processing closer to the users is still not unanimous in all the proposed architectures.

Regarding the validation in an actual city (C7) it is possible to see that only two works validated its architecture in a real environment or applied it to a simulation of the real scope. According to Sinaeepourfard et al. (2018), validation was carried out in the city of Barcelona in Spain and (Abreu et al., 2016) in the city of Lisbon in Portugal. In addition, we see another extreme in terms of the Traceability (C1), Data Fragmentation (C4), Cache (C5), and Replication (C6) competencies, which together make a massive contribution to a distributed model and which in no case were mentioned in the same architectural proposition.

Therefore, it is possible to observe some gaps in the propositions of the related works. First, we do not have a model that integrates several essential characteristics for data traceability in a distributed storage architecture. That is, more propositions need to be addressing efficient tracking of stored data. In addition, many approaches perform excess replication of metadata and/or records of IoT devices in several layers of the model, which generates an *overload* in the network that could be avoided using some centralized mechanism of metadata storage that can locate the data we store locally on each Fog Node. Finally, a few approaches specify which type and/or database system will be used in the architecture, which is a significant point for model implementation decisions.

4 Proposed Model

This section describes the DASTData model (an acronym for *Distributed Architecture to Store and Trace Data*). Based on the Theoretical Foundation (Section 2) and Related Work (Section 3), it was possible to define essential aspects to design a storage model for Smart Cities. In this way, the model's focus was defined as the proposition of a three-layer architecture (Edge, Fog, and Cloud) for distributed storage in smart cities. This architecture must include traceability and concurrent access to data from Edge layer devices stored in the Fog.

To detail each aspect, in the Section 4.1 we quote and justify the design decisions that were taken in elaborating the architecture. Then, in Section 4.2 we describe in detail the characteristics of the architecture in order to explain and exemplify its objectives. Finally, in the Section 4.3, we conclude with the definition of the services and routines that will be initially proposed.

4.1 Design Decisions

As seen in Section 3 regarding related work, especially from the analysis performed and summarized in Table 2, there are some essential competencies in building a model for a storage architecture. Therefore, some design decisions were made regarding the definition of what the model must have and meet, arriving at the following scenario:

- We will have three layers in the architecture called Edge, Fog, and Cloud. They will communicate through blocking and non-blocking interactions. They must also work in an integrated way to deal with the data running through their services in the architecture.
- NoSQL database will be used due to its advantages in a distributed context and data with dynamic structure, as mentioned before.
- Layers must be hierarchical so that each sub-layer deals with data specific to its competence to be defined.
- Device data storage must be partitioned via Sharding and distributed across Fog nodes to enable concurrent queries and geographic proximity to users.
- The architecture must be able to handle data from mobile users in order to allow the traceability of these users between the various nodes that compose it.
- The database instances on each node of the Fog will be interconnected through the network formed by the use of Sharding. This approach makes load-balancing actions between nodes less costly, as there is no need for prior data transmission for a new node to be responsible for the demand.
- In order to ensure greater data availability and redundancy in case of failure, databases must have replication. This approach is transparent to the user as data access is centralized, and the DBMS takes care of replication management.

4.2 Architecture

The architecture proposed from the design decisions mentioned in the previous subsection is divided into three layers: Edge, Fog, and Cloud. In this context, we have that the model's focus is on the distributed storage in the Fog and on the traceability of the data. In order to concentrate services sensitive to latency, the Fog Computing nodes will be placed geographically close to the clients at the Edge. Regarding the Cloud, this layer will usually be further away and can easily be in another state or country, depending on the service provider. This structure is aimed at a Smart City for generic use but could be used in different contexts, such as public health management. In this case, vital signs data generated at the Edge from IoT sensors such as smart bracelets communicate with a Fog Node that stores these records locally.

In short, the Edge layer is responsible for generating the data for the model. Fog for storing this data in a distributed way and processing critical data that need a prompt response and low latency. Finally, the Cloud is responsible for heavier computations that demand more time and for being the input channel for actors interested in Smart City management, such as public managers, institutions, governments, etc. Fig. 4 illustrates the proposed architecture and the arrangement of storage clusters between the different Fog nodes. In addition, the following subsections detail each level and aspect of the architecture, detailing the Edge, Fog, and Cloud layers, Data Fragmentation, and Replication.

4.2.1 *Edge*

The edge layer is composed of intelligent devices and environments such as IoT gateways, smartphones, wristbands, cars, cameras, traffic lights, smart homes, hospitals, and all those IoT devices that citizens can own. *Edge Devices* connect to Fog primarily through a session log. This procedure enables the device to communicate with a specific Fog Node and use its available services. In addition, devices on the Edge can have a mobile behavior; at some point, they are connected to a Fog Node, and later they move around the city and end up connecting to another Fog Node. This routine for the Edge Device is identical to the first connection to any Fog Node, making the procedures performed by the model transparent to the end user to enable mobility and subsequent data tracking.

4.2.2 *Fog*

The Fog layer is located between the Edge and the Cloud architecture. It is responsible for performing the most latency-sensitive processing for Edge Devices, such as vital signs analysis or traffic management. This layer has the services that help operate the Smart City, some being implemented in the architecture presented in this work and others with the possibility of future implementation. There are possibilities, for example, Edge Device connection control services (Section 4.3.1), saving services (Section 4.3.2), data search, analysis and consolidation, load balancing, among others. Within Fog, there is also a process of ranking the nodes so that nodes closer to the Cloud store more consolidated data than at lower levels in their local database. This approach also makes it possible for services to be scaled on a *bottom-up* model so that, when deemed necessary, a service can transfer an operation to the parent node.

Although the data is geographically isolated and close to the users through the *Sharding* technique, this feature allows all instances to access each other's data. In this way, load balancing algorithms, for example, when deciding that a particular task should be performed on the parent node, do not need to overload the network by previously transferring data from one database to another. As the nodes are all interconnected, the data needed to perform the task in the node with more free resources is accessible for processing, leaving the responsibility of loading the data necessary for the requested service. In addition, since Fog nodes are independent and handle a smaller set of data, consolidation tasks that require information from more than one Fog Node can be performed concurrently.

In this work, we consider that the Fog nodes are arranged in the city according to future implantation decisions. It is assumed that the database interconnections are between all Fog Nodes, but the scale routines are *bottom-up* as mentioned before. Therefore, the scope of the Fog Nodes is in zones so that a neighborhood can have one or more responsible Fog Nodes depending on

the analysis of the traffic in that region of the city. Another possible implementation would be arranged so that several adjacent neighborhoods with less traffic form a region with only one Fog Node. At this point, the architecture is flexible about these implementations.

In order to meet the objectives of this work related to the storage and traceability capabilities of Edge data for Fog, within the scope of this research, two Fog services will be implemented, which will be addressed in the Section 4.3. Although it is not part of and not addressed by the proposed model, a relevant point to be mentioned is the Fog Node selection algorithm. As it is an architecture prepared to deal with the mobility of devices at the Edge, there is the premise that there will be a service at the Edge and/or Fog responsible for controlling when to request the session registration on a new Fog Node.

4.2.3 *Cloud*

The Retrieve Data service and the user tracking metadata storage service (OTS - Object Tracking Service) are available in the Cloud. In this model, the Cloud is the furthest layer from the users and concentrates services that tolerate a longer response time. The distance of the Cloud concerning other services can reach large proportions on the scale of different states and countries, which can generate high latency rates and is a problem for applications that need a prompt response. In this architecture, two Cloud services are proposed that will help consolidate data and tolerate longer response times.

The first is the OTS, responsible for storing historical data on the displacement of users between Fog nodes. The second service is the data query service, *Retrieve Data*, which communicates with the OTS and Fog. To compose a traceability record of mobile devices in the city, *Retrieve Data* obtains the history contained in the OTS and, from this information, consults the data stored in the Fog Nodes. The idea is that from the Cloud, actors interested in city data have access to consolidated graphs, dashboards, and reports. In Section 4.3 we describe in more detail the data structures, responsibilities, objectives, and functioning of these services.

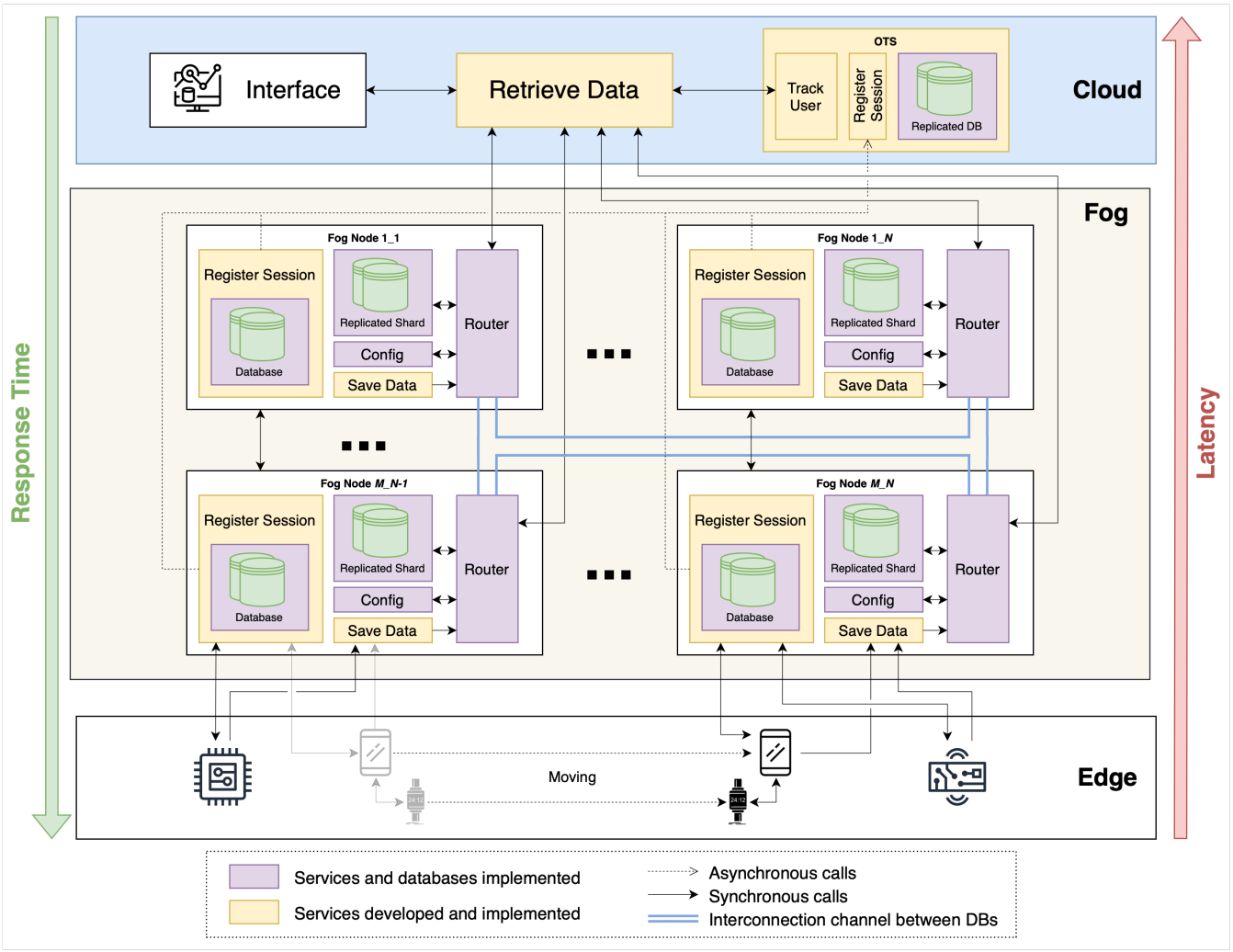

Figure 4: Distributed storage architecture proposal for Smart Cities.

4.2.4 *Data Fragmentation and Replication*

Data from Edge devices will be stored in Fog databases through the Sharding technique, performing the fragmentation from the identifier code of the respective Fog Node. In each Fog Node, there is a NoSQL database which is a partition of the global mass of data. This database has a service linked to it called *Router*, which is responsible for accessing the database, and defining the best execution plan for the requested read or write operation. As described in Section 2.4 it is necessary to define a key field as the fragmentation criterion and one of the records indexers. Therefore, as there is a data-saving service that consistently links the Fog Node identifier in the registry, the *shard_server* field was defined as a good option for the key to divide the data geographically.

Finally, each database instance on the Fog Nodes must be replicated to two more instances. Replication is essential to increase availability and add a layer of fault tolerance to the model. The replication will follow the *master/slave* model. When forming the replication, the DBMS elects an instance to be the main one that will receive all requests from the Router. A new election defines the primary instance if any unavailability occurs. This procedure does not generate losses, as the data is replicated in each instance. Another advantage of replication is the response time for read operations. As the data is replicated across the instances, it is possible to configure settings for multiple instances to respond to read operations, so that load balancing occurs and parallelism can process these requests.

4.3 Operation

The functioning of this model depends on some critical services. The following subsections describe the four primary services that enable distributed storage and user traceability.

4.3.1 *Fog Register Session*

The Fog Nodes *Register Session* module is responsible for receiving initial connection requests from devices on the Edge, generating a session identifier, making this information available to the requesting device, and asynchronously persisting this data in the OTS. Fig. 5 presents the sequence diagram of the session registration operation performed from an Edge Device to a new Fog Node. This procedure must be performed every time a Fog Node is responsible for performing tasks from an Edge Device. The flow starts with the registration request made by the Edge Device. The Fog Node receives this request and generates an identifier UUID version 1[8] to represent the new session called *session_id*, stores this record in a local database of the service and returns the identifier to be used later by the Edge Device in subsequent requests.

In addition, *Register Session* makes an asynchronous call to the OTS Cloud service that registers this session and, if there is a previous connection on another Fog Node, asynchronously requests that the old session be expired. The way these asynchronous calls will be performed and handled does not concern the proposed model, so this decision is open for future implementations. However, it is suggested that some fault-tolerant method be used to guarantee that the procedures will be executed successfully, for example, with the use of queue services or with the *publish/subscribe* message exchange pattern.

4.3.2 *Fog Save Data*

Save Data is the name of the service responsible for receiving requests that store data from Edge devices on Fog Nodes. It operates in conjunction with the replicated and partitioned database via Sharding. In addition, it acts as a *wrapper*, adding more information to the data that will be stored, such as Fog Node, session, and user identifiers. Its communication with the database is through the *Router* contained in each Fog Node and the storage metadata service called *Config*.

Fig. 6 shows two examples of data stored in Fog Nodes and how the records of the same user moving between two Fog Nodes would look like. The *session_id* field varies only when there is a context change. The *id* field is a unique identifier with the format that best suits the implementation's database. It should be noted that the *data* field has a flexible structure, as it contains the data sent by the device or service to be stored.

In addition, Fig. 7 shows the sequence diagram of operations performed in the data-saving flow from Edge to Fog. The main flow consists of the Edge Device sending the request to save data in the Fog Node passing a valid *session_id*. Fog Node validates the informed session and saves the record. The validations regarding the session by the Fog Node are to verify if a session identifier was informed and if the value informed is valid or not.

4.3.3 *Cloud OTS*

This service is located in the Cloud and is responsible for storing the global tracking metadata of users in the city. In Fig. 8 an example of the base structure of the data stored in the OTS is presented, and in the sequence diagram of Fig. 5 it was previously shown how the data would reach the OTS through of asynchronous requests. The field *shard_server* will be the fragmentation key. That is, from this field, the stored data will be linked to the corresponding partition of the database. The field *session_id* is the session identifier and one of the data indices. It is used to group data from the same session, linking the OTS records with the Fog Nodes data.

From these OTS historical records, it will be possible to calculate the time spent on each node through the connection time, perform the tracking and extract useful information from these stored data. Different services can use the traceability returned by the OTS for different purposes. One of these applications is the search for data in Fog Nodes based on a user identifier and a period presented in the following subsection with the *Retrieve Data* service. However, the application of OTS history is not limited to the search for data in Fog Nodes. There may be "brother" services to *Retrieve Data* that manipulate this data to obtain other information.

[8]UUID v1: it is a universally unique identifier generated from a *timestamp* and the MAC address of the computer that is generating it.

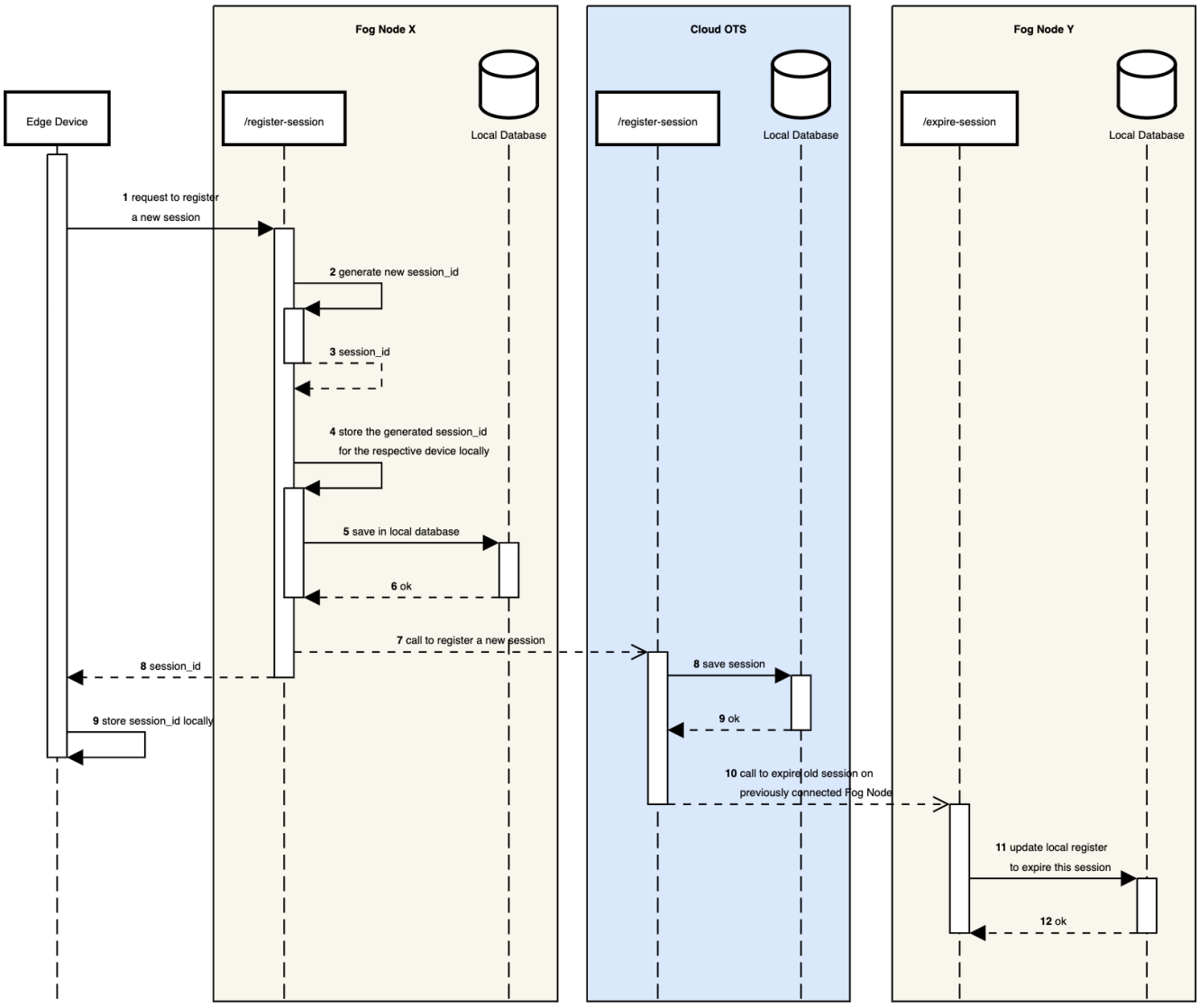

Figure 5: Sequence diagram of the synchronous and asynchronous operations performed to register a new session in a Fog Node.

Figure 6: Example of a data structure for the same user stored in two different Fog Nodes. Side (a) represents data from different sessions for the same user, while side (b) represents data from the same session for the same user.

4.3.4 *Cloud Retrieve Data*

This service is an example of the integration between the OTS and the data stored in the Fog Nodes to unite the distributed storage with the traceability of the users. Fig. 9 shows a user's data search operation sequence diagram in a given period. A generic interface requests the data for the service *Retrieve Data* that forwards a query to the OTS and, after receiving the return, decides the best route to request the data, taking into account variables such as the current availability of *Routers*. Once the best routes are defined, concurrent queries are performed directly for each Fog Node that owns the records. This approach provides high performance for the model, fetching data stored in a distributed fashion. After the end of the queries, the data is aggregated in JSON format to be returned to the interface that made the request.

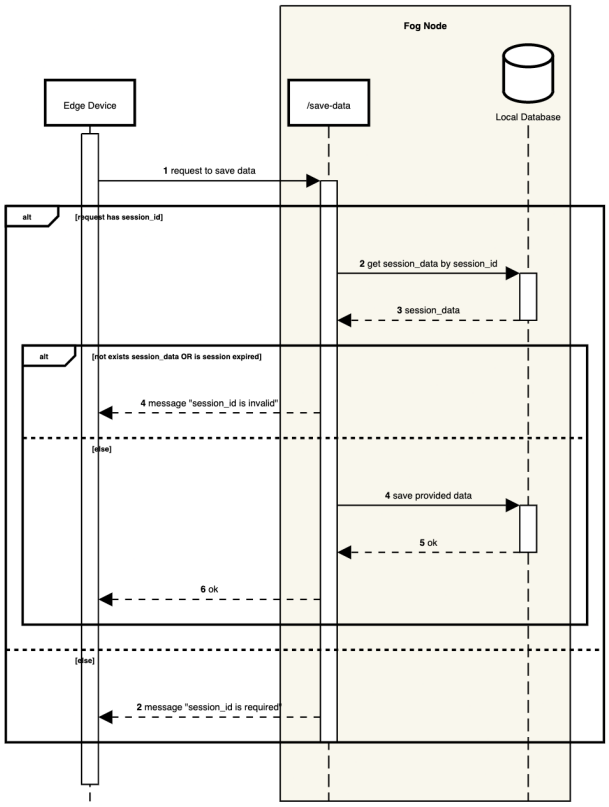

Figure 7: Sequence Diagram of the *Save Data* service of the Fog Nodes.

```
[
  {
    "id": "<unique-id>",
    "shard_server": 1002,
    "router_address": "layer-10-node-02.fog.com.br",
    "connected_at": "2022-06-01T08:00:00.000Z",
    "user_id": "d9e4b8cc-b2b0-4ee8-955c-4511f63a7c22",
    "session_id": "48a88774-e599-11ec-8fea-0242ac120002"
  },
  {
    "id": "<unique-id>",
    "shard_server": 1003,
    "router_address": "layer-10-node-03.fog.com.br",
    "connected_at": "2022-06-01T11:45:34.000Z",
    "user_id": "d9e4b8cc-b2b0-4ee8-955c-4511f63a7c22",
    "session_id": "9fcd4526-beda-43ff-92ff-0242ac120003"
  },
  {
    "id": "<unique-id>",
    "shard_server": 1002,
    "router_address": "layer-10-node-02.fog.com.br",
    "connected_at": "2022-06-01T18:30:15.000Z",
    "user_id": "d9e4b8cc-b2b0-4ee8-955c-4511f63a7c22",
    "session_id": "e1c97d5f-e599-11ec-8fea-0242ac120002"
  }
]
```

Figure 8: Example of structure of data stored in OTS.

5 Evaluation Methodology

We implemented a prototype that simulates the organization of a Smart City and allows the installation of several database instances to obtain evaluation metrics that allow the analysis of the results of the proposed model. Therefore, in this section, the validation process will be described. The Section 5.1 describes how the model prototyping and emulation will be done. In the Section 5.2, the test infrastructure set up to evaluate the architecture and which devices will be used in the emulation is presented. Then, the variants considered for the tests will be described in the Section 5.3. Finally, the metrics used to evaluate the model are defined in the Section 5.4.

5.1 Prototype and Data Emulation

The prototype created to enable the emulation contains the implementation of the Edge, Fog, and Cloud layers, according to Section 4.2. To meet the objectives of the proposed model, the selected database was *MongoDB v4.2*[9]. Being NoSQL and document-oriented, this database fits the model, as it natively has flexible data structures and sharding resources by zones and replication. The programming language chosen for the prototype services was *JavaScript* in association with *runtime* for server-side code execution *Node.js v16*[10]. The ease of integration between the programming language and this database through ODM[11] Mongoose[12] was a critical factor in its choice.

The data generated from the Edge were simulated from the generation of datasets using the library Faker.js[13]. In the repository available at `https://github.com/daniellferreira/fog-sharded-storage`, the infrastructure of a Fog Node and its services *Register Session* and *Save Data* were implemented. Cloud services *OTS* and *Retrieve Data* were implemented in the repositories `https://github.com/daniellferreira/ots` and `https://github.com/daniellferreira/ots-retrieve-data` respectively. The emulation sought to reproduce read and write operations in the databases of the Fog Nodes of a Smart City, both for the single manipulation of the records in the Fog Nodes and from the search for the traceability of a given user from the Cloud.

5.2 Testing Infrastructure

The infrastructure used to evaluate the model was composed of the following items by layers:

- Edge – was used a computer with an Intel Core i7-

[9] MongoDB: `https://www.mongodb.com/docs/manual/`

[10] Node.js: `https://nodejs.org/`

[11] Object Document Mapper (ODM): an application that maps and abstracts documents (records) in the database to structures known by programming languages .

[12] Mongoose: `https://mongoosejs.com/`

[13] Fakerz. js: https://fakerjs.dev/

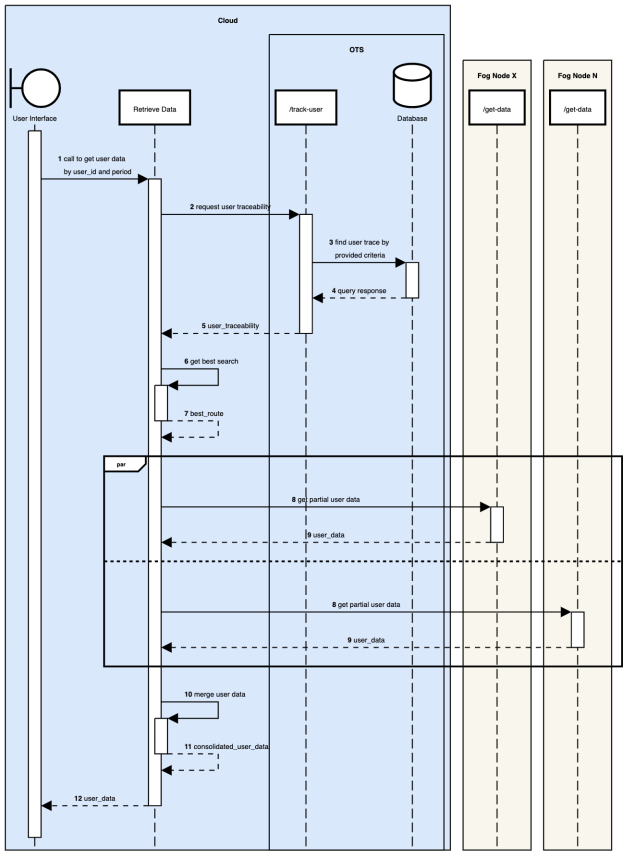

Figure 9: Sequence diagram of the *Retrieve Data* service.

9750H Hexa-core processor with 2.6 GHz of clock, 16 GB of DDR4 RAM with a minimum frequency of 2667 MHz, and operating system macOS version 11.6.5. Locally, multiple instances of Edge Devices were simulated through the Apache JMeter[14] application that made calls to Fog services.

- Fog – a computer was used to represent two Fog Nodes by running independent Docker containers. Fog's *Save Data*, *Config* and *Router* services and the MongoDB database with sharding and replication in two instances were implemented. Regarding the configuration of this computer, it is a quad-core processor Intel Core i5-8265U with 1.6 GHz of clock, 20 GB of DDR4 RAM with a minimum frequency of 2666 MHz, and an operating system Linux distribution Linux Mint 20.3. In addition, the ngrok[15] service was used to proxy the Fog Nodes, exposing applications running locally to the internet and access to the Cloud.
- Cloud – the Retrieve Data and OTS services were deployed on the Heroku platform[16], and the OTS database was deployed on the MongoDB Atlas platform[17]. Both platforms are Cloud abstractions for servers and databases and use the resources of *Amazon Web Services*[18] (AWS) to host your services. In this way, the same AWS *data center* in the US (*us-east-1*) was selected, so it is possible to reduce the latency of communication between Cloud services and databases.

5.3 Scenarios and parameters

In order to evaluate the model in this research, the following two scenarios were implemented:

i. Scenario A – in the first scenario, the response time for reading and writing operations from the Edge in the database of a single Fog Node was evaluated. As a parameter, the number of Edge Devices was varied, progressing the number of devices from 1 to 30. In addition, ten calls were made per Edge Device, so in the last iteration, 30 simultaneous requests were made and a total of 300 requests. A random delay of a maximum of 100ms was also used so that the sending of requests had a slightly more natural behavior. This scenario allowed the reproduction of the operation of the Smart City regarding the interaction of Edge Devices with Fog Nodes.

ii. Scenario B – the second scenario simulated the traceability of users performed from the Cloud service *Retrieve Data*. The number of Fog Nodes (from 1 to 4) or a centralized approach were used as parameters. In each Fog Node, a dataset with 1000 records was inserted, from which approximately 100 records should be returned in the query performed. Thus, for the centralized approach, the mass of data ranged from 1000 to 4000 records and should return approximately 100 to 400.

This scenario varied in 3 approaches and was executed ten times for each. The first is proposed by the DASTData model: data distributed in Fog Nodes and the query via the Cloud performed concurrently. The second approach also used distributed storage, but querying the Fog Nodes was changed in the *Register Session* service to be performed sequentially. Finally, the third approach used centralized storage in Fog to simulate approaches that do not distribute data across Fog Nodes. Each of these approaches represents a possible implementation for the traceability of distributed data. This scenario could, for example, be querying user data through a cloud dashboard.

5.4 Evaluation Metrics

The evaluation metrics used in the first scenario evaluate the model's performance. Therefore, the mean, median, 90th percentile (P90), 95th percentile (P95), 99th percentile, and minimum and maximum latency values in milliseconds were used. In the second scenario, as the evaluation is comparative between different approaches, the average latency in milliseconds for the executions was compared.

[14] Apache JMeter: https://jmeter.apache.org/
[15] ngrok: https://ngrok.com/
[16] Heroku: https://www.heroku.com/
[17] MongoDB Atlas: https://www.mongodb.com/atlas/database
[18] Amazon Web Services (AWS): https://aws.amazon.com/

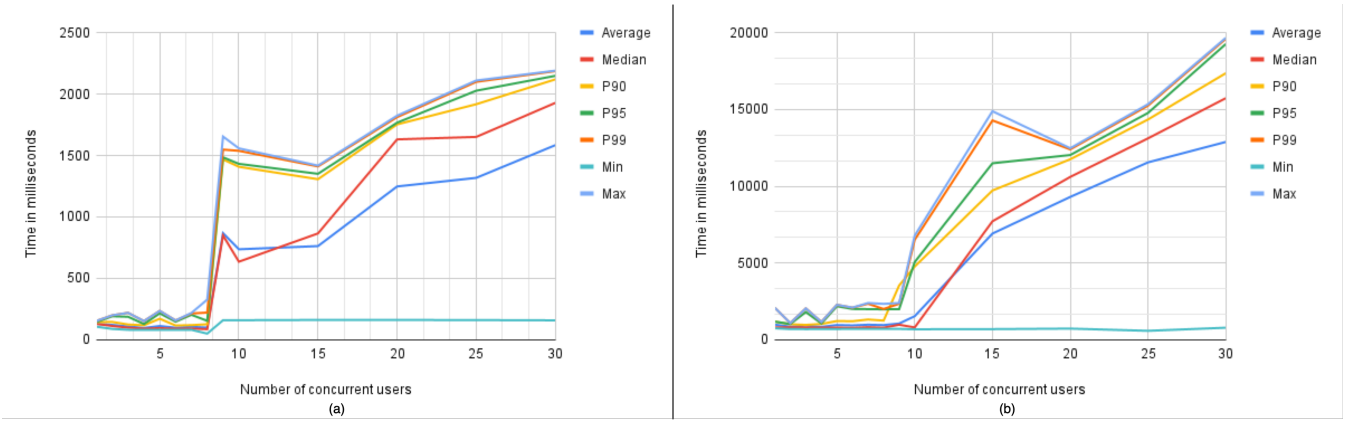

Figure 10: Scenario A results for read (side a) and write (side b) operations on databases distributed across Fog Nodes.

6 Results

This section will present the results obtained through the evaluation methodology presented in Section 5. In addition, the following subsections directly reference the scenarios described earlier.

6.1 Scenario A

The results shown in Fig. 10 demonstrate that reading operations are faster than writing operations. Latency averages, for example, show that reading can be up to 80% faster than writing. This is because write operations replicate this information in each database instance contained in the Fog Node at the time of including the information. Furthermore, as read operations do not compromise data integrity, they are performed concurrently by each database instance. Another point observed in the figure is related to the latency increase when eight simultaneous users are exceeded. This happens because, as described in Section 5.2, the processor of the Fog Nodes in this emulation is a quad-core with four physical and four logical cores.

6.2 Scenario B

Fig. 11 presents the results of the test scenario that simulated how the search for user traceability would be in the Cloud (*Retrieve Data* for OTS) and later the search for data in Fog Nodes. Analyzing this graph, it is possible to conclude that using distributed storage with concurrent queries is more efficient than centralized storage. In addition, there is also the advantage of using concurrency in the task of querying distributed data, as in this approach, there is no need to wait for a response from a server to request data from the next one. Therefore, through the results presented in this scenario, it is observed that the approach of the model proposed in this research can be approximately up to 74% more efficient than in a centralized model and up to 53% more efficient compared to a distributed sequential model.

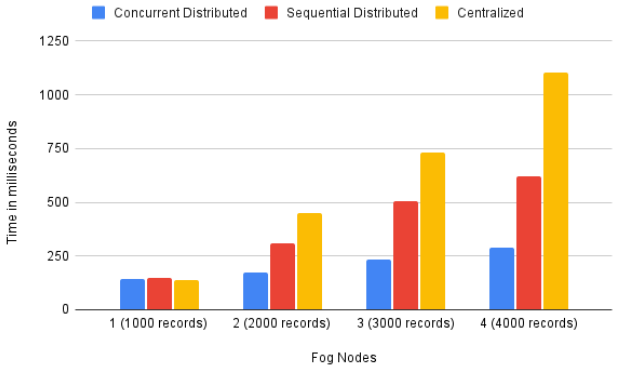

Figure 11: Scenario B results for fetching a user's data in the model.

6.3 Discussion

From the results presented, it is possible to discuss some points of the architecture related to the scalability of Fog Nodes. Although access is decentralized, there may still be problems related to population concentration in specific Fog Nodes to the detriment of others. The high number of concurrent users showed in Scenario A that concentrating access to just one database could be detrimental to the model's response latency from a certain point onwards. For this reason, a solution could be related to using on-demand subshardings in each Fog Node to provide horizontal scalability internally. In this case, more databases could be implemented that equally share the workload to resolve to write and read operations within a single Fog Node.

Another relevant point is the implementation of MongoDB as the main engine for the evaluation. Some of the results may be biased due to using a single DBMS. Therefore, it may be relevant to study and implement the DASTData model in other Non-Relational Databases with replication support in order to determine whether the chosen resource's algorithms are not also contributing to the observed improvement points.

7 Conclusion

In this work, the DASTData model was presented, an architecture proposal for distributed storage that enables the traceability of Smart Cities users. Through high-performance approaches such as concurrency, replication, and sharding, this research demonstrated the advantages of DASTData concerning centralized models and/or models that do not use concurrent operations. It is expected that this model will contribute to the academic literature as a possible storage architecture to be implemented in Smart Cities.

Although the model meets the objectives of the work, the results showed some limitations related to the high demand of several Edge Devices to the Fog Nodes concomitantly. Furthermore, the model needs to cover some points about implementing asynchrony and switching decision algorithms between Fog Nodes. There was also no detail on performing load balancing between Fog Nodes. Therefore, there are some future works to propose improvements to the limitations mentioned and also related to the emulation environment. It would be highly relevant to carry out tests in the proportions of a city in order to obtain more authentic results than those of a real operation.

References

Abdelhafiz, B. M. and Elhadef, M. (2021). Sharding database for fault tolerance and scalability of data, *2021 2nd International Conference on Computation, Automation and Knowledge Management (ICCAKM)*, IEEE, Dubai, United Arab Emirates, pp. 17–24. Available at https://doi.org/10.1109/iccakm50778.2021.9357711.

Abreu, D. P., Velasquez, K., Curado, M. and Monteiro, E. (2016). A resilient internet of things architecture for smart cities, *Annals of Telecommunications* **72**(1-2): 19–30. https://doi.org/10.1007/s12243-016-0530-y.

Al Jawarneh, I. M., Bellavista, P., Corradi, A., Foschini, L. and Montanari, R. (2021). Efficient QoS-aware spatial join processing for scalable NoSQL storage frameworks, *IEEE Transactions on Network and Service Management* **18**(2): 2437–2449. https://doi.org/10.1007/s12243-016-0530-y.

Benhamida, N., Bouallouche-Medjkoune, L., Aïssani, D. and Kouahla, Z. (2022). Dynamic architecture for collaborative distributed storage of collected data in fog environments, *Wireless Personal Communications* **123**(4): 3511–3537. https://doi.org/10.1007/s11277-021-09301-6.

Bonomi, F., Milito, R., Zhu, J. and Addepalli, S. (2012). Fog computing and its role in the internet of things, *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, MCC '12, ACM, New York, NY, USA, pp. 13–16. Available at https://doi.org/10.1145/2342509.2342513.

Chang, V. (2021). An ethical framework for big data and smart cities, *Technological Forecasting and Social Change* **165**: 120559. https://doi.org/10.1016/j.techfore.2020.120559.

Chen, S., Xu, H., Liu, D., Hu, B. and Wang, H. (2014). A vision of IoT: Applications, challenges, and opportunities with china perspective, *IEEE Internet of Things Journal* **1**(4): 349–359. https://doi.org/10.1109/jiot.2014.2337336.

Corbellini, A., Mateos, C., Zunino, A., Godoy, D. and Schiaffino, S. (2017). Persisting big-data: The NoSQL landscape, *Information Systems* **63**: 1–23. https://doi.org/10.1016/j.is.2016.07.009.

Khatoun, R. and Zeadally, S. (2016). Smart cities: Concepts, architectures, research opportunities, *Communications of the ACM* **59**(8): 46–57. https://doi.org/10.1145/2858789.

Kudo, T. (2018). Fog computing with distributed database, *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*, IEEE, Krakow, Poland, pp. 623–630. https://doi.org/10.1109/aina.2018.00096.

Lai, C. S., Jia, Y., Dong, Z., Wang, D., Tao, Y., Lai, Q. H., Wong, R. T. K., Zobaa, A. F., Wu, R. and Lai, L. L. (2020). A review of technical standards for smart cities, *Clean Technologies* **2**(3): 290–310. https://doi.org/10.3390/cleantechnol2030019.

Lomotey, R. K., Pry, J. C. and Chai, C. (2018). Traceability and visual analytics for the Internet-of-Things (IoT) architecture, *World Wide Web* **21**(1): 7–32. https://doi.org/10.1007/s11280-017-0461-1.

Mell, P. and Grance, T. (2011). The NIST definition of cloud computing, *Technical report*, Gaithersburg, MD. Available at https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf.

Naeem, R. Z., Bashir, S., Amjad, M. F., Abbas, H. and Afzal, H. (2019). Fog computing in internet of things: Practical applications and future directions, *Peer-to-Peer Networking and Applications* **12**(5): 1236–1262. https://doi.org/10.1007/s12083-019-00728-0.

Patel, K., Patel, S., Scholar, P. and Salazar, C. (2016). Internet of things-iot: definition, characteristics, architecture, enabling technologies, application & future challenges, *International Journal of Engineering Science and Computing* **6**(5): 6122–6131. https://ijesc.org/upload/8e9af2eca2e1119b895544fd60c3b857.Internet%20of%20Things-IOT%20Definition,%20Characteristics,%20Architecture,%20Enabling%20Technologies,%20Application%20&%20Future%20Challenges.pdf.

Sasubilli, S. M., Kumar, A. and Dutt, V. (2020). Improving health care by help of internet of things and bigdata analytics and cloud computing, *2020 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, IEEE, Las Vegas, NV, USA, pp. 1–4. Available at https://doi.org/10.1109/ICACCE49060.2020.9155042.

Shwe, H. Y., Jet, T. K. and Chong, P. H. J. (2016). An IoT-oriented data storage framework in smart city applications, *2016 International Conference on Information and Communication Technology Convergence (ICTC)*, IEEE, Jeju, South Korea, pp. 106–108. Available at https://doi.org/10.1109/ictc.2016.7763446.

Sinaeepourfard, A., Garcia, J., Masip-Bruin, X. and Marin-Tordera, E. (2018). Data preservation through fog-to-cloud (F2C) data management in smart cities, *2018 IEEE 2nd International Conference on Fog and Edge Computing (ICFEC*, IEEE, Washington, DC, USA, pp. 1–9. Available at https://doi.org/10.1109/cfec.2018.8358732.

Vilela, P. H., Rodrigues, J. J. P. C., Righi, R. d. R., Kozlov, S. and Rodrigues, V. F. (2020). Looking at fog computing for E-Health through the lens of deployment challenges and applications, *Sensors* **20**(9): 2553. https://doi.org/10.3390/s20092553.

Zhang, C. (2020). Design and application of fog computing and internet of things service platform for smart city, *Future Generation Computer Systems* **112**: 630–640. https://doi.org/10.1016/j.future.2020.06.016.