*Journal of Information and Data Management, 2023, 14:1,* doi: 10.5753/jidm.2023.3189

# `Hurricane`: a Dataflow-oriented Data Service for Smart Cities Applications

**Maicon Banni** [ **Universidade Federal Fluminense** | ***maiconbanni@id.uff.br*** ]
**Maria Luiza Falci** [ **Universidade Federal Fluminense** | ***marialuizafalci@id.uff.br*** ]
**Isabel Rosseti** [ **Universidade Federal Fluminense** | ***rosseti@ic.uff.br*** ]
**Daniel de Oliveira** [ **Universidade Federal Fluminense** | ***danielcmo@ic.uff.br*** ]

*Institute of Computing, Universidade Federal Fluminense, Av. Gal. Milton Tavares de Souza, s/n, São Domingos, Niterói, RJ, 24210-590, Brazil.*

**Received:** 27 February 2023 • **Published:** 20 October 2023

**Abstract** The concept of *Smart Cities* has gained relevance, especially in the last decade, due to the availability of data associated with cities, *e.g.*, car traffic, public transportation, crime data, *etc*. The purpose of using these data is to improve the services offered to the citizens. Most of these applications manipulate spatiotemporal data. These data are processed in a dataflow that starts with the collection, integration, and aggregation and ends with visualization. This way, specialized data services for smart city applications are most welcome. However, many of the existing data services in this context, are either specific to a particular application/domain or do not consider the entire data life cycle. In this article, we present `Hurricane`, a dataflow-oriented data service for smart city applications. `Hurricane` executes multiple dataflows to gather, pre-process, integrate, and public data. `Hurricane` was evaluated with an application in the area of public security and results reinforced the importance of this type of data service.

**Keywords:** Smart Cities, Data Management, Data Integration.

## 1 Introduction

The concept of *smart Cities* has gained more relevance in the past decade [Bilal *et al.*, 2020]. Many initiatives have been proposed in many areas of public administration, *e.g.*, public security [Chen *et al.*, 2017; Lourenço *et al.*, 2018]. Regardless of the goal, a smart city aims at managing and using the available infrastructure and services efficiently and effectively to improve the citizens' well-being [Bilal *et al.*, 2020]. A smart city depends on open data to plan public policies. In fact, in the past few years, the smart cities infrastructure has been able to generate fine-grained data available and in unprecedented space-time scales (*e.g.*, sensors data, social networks data, *etc.*) [Nandury and Begum, 2016]. Thereby, the existence of a smart city depends on how well the organizations can analyze and extract useful knowledge from the available data. However, the access and processing of these data may not be trivial, because it requires integrating data in different formats, volumes, and granularities.

Even though multiple smart city open data sources were available in the past years [Pisco and Marques-Neto, 2021], many smart city applications consume data previously integrated at a certain level. However, most available data on open data sources have not been integrated before (*i.e.*, raw data). One example is an application for crime analysis on urban centers, where diverse data types can be used in an integrated analysis, such as (i) police reports; (ii) spatial distribution of homeless population; (iii) statistics regarding drug users, *etc*. While data from police reports can be acquired from Department of Public Security websites (*e.g.*, SSP-SP[1]), the statistics regarding homeless populations and drug users can be collected on the IPEA website[2]. Integrating those data, organizing them, and making them available to third parties requires the user to perform a manual, tedious, and error-prone task. That is an open, yet important, problem in the smart cities context [Raghavan *et al.*, 2019]. Thus, solutions that can automatically or semiautomatically support integrated management of those data are still necessary.

Several approaches aim at supporting data management for smart city applications. However, the existing approaches are related to a specific domain [Zhou *et al.*, 2021; Bellini *et al.*, 2021; Garcia-Font, 2020] or they are focused in a topic such as data transfer optimization [Nandury and Begum, 2016] or enriching semantic queries [Silva *et al.*, 2021]. Even the approaches that are proposed as generic data management frameworks [Liu *et al.*, 2017; Jindal *et al.*, 2020] do not take into account data integration during its complete life cycle, *i.e.*, it is not possible to discover which data were derived from a specific input raw data, as well as which step or activity performed these transformations, *i.e.*, provenance [Freire *et al.*, 2008]. In fact, Ribeiro and Braghetto [2021] discuss the importance of this issue and define a conceptual architecture that includes all the necessary services to manage data in the smart cities context. According to Ribeiro and Braghetto [2021], a data management architecture for smart cities has to consider the following stages: (i) ingestion; (ii) metadata and provenance management; (iii) data processing and (iv) data queries [Ribeiro *et al.*, 2020].

This article proposes `Hurricane`, a data service that of-

[1] `https://www.ssp.sp.gov.br`
[2] `https://www.ipea.gov.br`

fers the functionalities described by Ribeiro and Braghetto [2021]. Hurricane is an extensible and configurable data management service designed for the smart cities context. Hurricane supports (i) upload of heterogeneous data from external sources, (ii) data pre-processing (*e.g.*, remove duplicated data), (iii) aggregate and integrate data in multiple spatiotemporal dimensions to ease visualization and data analysis. This aggregation step is important since numerous smart cities applications execute multiple aggregations on space and time and have to return results with the minimum possible latency, *e.g.*, below 0.5 seconds [Liu and Heer, 2014], (iv) integrate data between different datasets through mappings defined by the user (*i.e.*, semiautomatic integration), (v) capture metadata, including provenance data, (vi) anonymize raw and pre-processed data, (vi) make data available to developers/consumers through APIs.

Different from other data management approaches in the smart cities context, Hurricane follows the dataflow abstraction [Silva *et al.*, 2017a; de Oliveira *et al.*, 2019b] for data management, since a dataflow represents an evolution of the data transformations and follows the data propagation through the applications in a fine-grain way (*e.g.*, multiple related data files). This way, Hurricane does not store only processed data, but also intermediate data and raw data that were loaded from external data sources in a Data Lake [Nargesian *et al.*, 2019]. It allows the dataflow to be registered, and unplanned analysis to be performed in the future. Furthermore, Hurricane considers data privacy issues, as sensitive data can be within the data files acquired from external sources (*e.g.*, personal data in a police report). Hurricane was evaluated in quantitative and qualitative experiments with an application in the public security domain and has shown promising results.

This article is an extension of work originally reported in the Proceedings of the Brazilian Symposium on Databases [Banni *et al.*, 2022] held in Búzios, RJ - Brazil, on September 2022. In this extended version, we enriched the experimental evaluation with more analyses. We have also improved the background and the related work sections. The remainder of this article is structured as follows. This article is organized into five sections besides the Introduction. Section 2 discusses background knowledge. Section 3 brings related work. Section 4 presents Hurricane details. Section 5 presents Hurricane evaluation, and, finally, Section 6 concludes the present article and presents future work.

## 2 Background Knowledge

This section discusses two important concepts regarding this article: (i) Data Integration and (ii) Dataflows.

### 2.1 A Brief Explanation on Data Integration

Integrating heterogeneous data is a challenge independently of the application scenario. In the smart city context, the integration is also challenging due to multiple data formats, lack of metadata, and different granularity levels. The process of integrating heterogeneous data is commonly classified into two different categories: (i) virtual [Chawathe *et al.*, 1994], and (ii) materialized strategies [Widom, 1995]. In the former strategy, data are stored in their sources and integrated using a mediator, which receives query requests, submits queries to multiple data sources, and consolidates the results. This type of integration strategy is straightforward when data is structured (*e.g.*, using a relational database), but is complex when we have to deal with unstructured data (which is the context of smart cities applications) since an extractor is required for fetching the data in a structured form before processing the query [Chawathe *et al.*, 1994]. On the other hand, in the materialized strategy, the user must include in the software stack a component that materializes the information from multiple and heterogeneous sources as well as a monitor component that periodically accesses the original data sources and fetches data updates. Historically, the aggregated and materialized data are consolidated and stored in a centralized database, usually a Data Warehouse [Kimball and Ross, 2002].

Data warehouses follow the multidimensional modeling paradigm [Kimball and Ross, 2002]. This paradigm is commonly based on different types of schema (*e.g.*, star and snowflake) which are composed of fact and dimensions tables that store data in Relational DBMS, for instance. Although data warehouses represent a step forward, they present some limitations to be used in the smart city context: (i) Data warehouses are designed to optimize data insertion and retrieval, disregarding the aspect of updates (which may be common in smart cities context due to the addition of multiple sources), (ii) Data warehouses heavily rely on Extract, Transform, and Load (ETL) pipelines, which may be complex to be modeled *a priori* in the smart city context, and (iii) Data persistence is likely to exceed the capacity of traditional tools such as traditional DBMS, which may cause clogging for smart city data workloads [Syed, 2020].

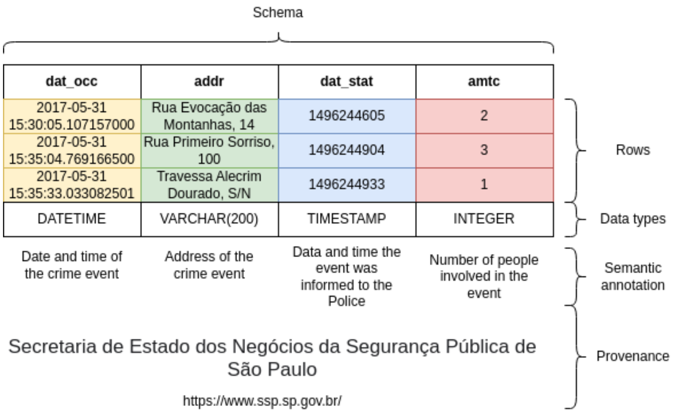
An alternative for processing and integrating the huge volume of smart city data is using Data Lakes, which can be defined as centralized repositories for users to insert structured, semi-structured and unstructured data [Miller, 2018]. In recent work, Brito [2018] highlighted the main differences between data warehouses and data lakes. The main difference is that data lakes store data in their raw format, *i.e.*, they do not require a pre-defined schema to store data. Also, data lakes do not require a previously defined ETL process [Nargesian *et al.*, 2019], *i.e.* data are loaded in their raw format and the user defines *a posteriori* and on demand how to access and query data. The data life-cycle in data lakes is defined by Ciobanu *et al.* [2019] as:

1. Data Ingestion: Data gathered from multiple sources are stored in a centralized repository in its raw format;
2. Data Storage: Data are stored in their original format (structured or unstructured data) by taking full advantage of distributed file systems, such as Hadoop Distributed File System (HDFS) and AWS Simple Storage Service (S3);
3. Metadata Extraction: Data lakes heavily rely on multiple levels and types of metadata for categorize and search for data. Metadata plays a fundamental role in data lakes for retrieving information and commonly involves the data labeling using users-defined tags of in-

terest, and;

4. Exploration and Visualization: Data lakes convert raw data into a format that eases the extraction of information, whereas *ad-hoc* external processes usually carry out data exploitation.

The metadata in data lakes can be categorized as (i) structure/schema, (ii) semantics, and (iii) provenance data. The first type of metadata identifies, describes, and locates data fields available in raw data for querying and analyses, while semantic metadata describes the concepts associated with each data attribute in raw data. Finally, provenance metadata provides the origin and derivation path for stored data. Figure 1 shows an excerpt of a crime dataset that registers crime events on a specific date and a specific address. This dataset has associated metadata to be stored in a data lake. Schema metadata defines the attributes' data types (*e.g.*, `INTEGER` and `VARCHAR`), while semantic metadata includes annotations associated with each attribute. For instance, the "Date and time of the crime event" annotation is associated with the attribute `dat_occ` in the example of Figure 1. Likewise, the provenance metadata in Figure 1 store the data origin and derivation path.



| dat_occ | addr | dat_stat | amtc |
|---|---|---|---|
| 2017-05-31<br>15:30:05.107157000 | Rua Evocação das Montanhas, 14 | 1496244605 | 2 |
| 2017-05-31<br>15:35:04.769166500 | Rua Primeiro Sorriso, 100 | 1496244904 | 3 |
| 2017-05-31<br>15:35:33.033082501 | Travessa Alecrim Dourado, S/N | 1496244933 | 1 |
| DATETIME | VARCHAR(200) | TIMESTAMP | INTEGER |
| Date and time of the crime event | Address of the crime event | Data and time the event was informed to the Police | Number of people involved in the event |

Schema
Rows
Data types
Semantic annotation
Provenance
Secretaria de Estado dos Negócios da Segurança Pública de São Paulo
https://www.ssp.sp.gov.br/

**Figure 1.** Example of the three types of Data Lake metadata.

## 2.2 Dataflow Concept

The data management in a smart city application is usually composed of multiple steps (*e.g.*, data download, aggregation, *etc.*), and can be seen as a dataflow [de Oliveira *et al.*, 2019b]. This dataflow reflects the processing steps that a certain portion of data will undergo when subjected to the smart city application. Thus, a dataflow abstraction is a natural solution to model and capture metadata from the integration process of a smart city application. We can formalize the notation of dataflows according to the main definitions presented by Ikeda *et al.* [2013] and Silva *et al.* [2017b]. A data element $e = \{v_1, v_2, \ldots, v_n\}, v_i \in \mathbb{V}$ is a sequence of values $v_i \in \mathbb{V}$. A data collection $c$ is a collection of data elements. A dataset $s = \langle A, C \rangle$ is a tuple with a sequence of attributes $A = \{a \mid a \in \mathbb{A}\}$, where domain $\mathbb{A}$ is the pair $\langle$name, type$\rangle$, and $C$ is a set of data collections in which $\forall\ c \in C \wedge e \in c,\ |e| = |A|$. A data transformation $t(I),\ t : I \rightarrow O$ is a function that maps input dataset $I$ to output dataset $O$. A data dependency $\phi = \langle s, t, t' \rangle$ is a triple composed of a dataset $s$ and two data transformations $t$ e $t'$, where $s \subseteq t(s) \wedge s \subseteq t'(t(s))$. Therefore, a dataflow $D = \langle T, S, \Phi \rangle$ is a triple with data transformations $T$ of a dataset $S$, and a set of data dependencies $\Phi$. Given a specific data transformation $t(I) = O$, an instance of a data transformation $t(I') = O'$ is the result of mapping any subset of data collections from dataset $I' = \cup_{k=1}^{|I|} C'_k,\ C'_k \subseteq C_k \in I$ to any data collections in the output dataset $O' = \cup_{k=1}^{|t(I)|} C'_k,\ C'_k \subseteq C_k \in t(I)$.

# 3 Related Work

As most of the human population will live in cities in the next few decades, there is a great effort to improve the quality of these cities by turning them into smart cities. Ahmad *et al.* [2022] discuss a series of challenges to create a smart city and human-centered city, and one of the most critical challenges is associated with *data management*. As aforementioned, there are some approaches found in the literature that propose solutions related to data management for smart city applications. In this section, we have conducted a simplified systematic mapping following the recommendations of Petersen *et al.* [2015]. The systematic mapping may be defined as a method for creating a classification and structuring a particular topic of interest. In the context of this article, the topic of interest is "Data Management for Smart Cities". As recommended by Petersen *et al.* [2015], we have defined the following research question: **RQ1:** Which are the existing approaches for "Data Management for Smart Cities?".

One possible way to search for the publications is using the keywords extracted from the research question RQ1 (*e.g.*, "data management", "smart city") to find relevant approaches in the domain. In this article, we have chosen the snowballing strategy [Wohlin, 2014]. This strategy considers a seed set of papers as input. This seed set comprises seminal papers of an area and papers. These papers have their references analyzed, and the cited papers that meet specific criteria are considered in a new snowballing round (*i.e.*, they also have their references analyzed). This process may repeat for several rounds or until no new papers are included in the next snowballing round. We used Google Scholar to get the seed set by searching "Data Management for Smart Cities". As inclusion criteria, we considered peer-reviewed papers, papers published after 2010 (a 10-year window), and papers published in English that cover the data management in smart cities topic. Any paper that does not meet these criteria was excluded from the snowballing process. We obtained 13 papers [Consoli *et al.*, 2015; Silva *et al.*, 2021; Liu *et al.*, 2017; Jindal *et al.*, 2020; Ribeiro and Braghetto, 2021; Costa and Santos, 2017; Mehmood *et al.*, 2019; Garcia-Font, 2020; Zhou *et al.*, 2021; Bellini *et al.*, 2021; Nandury and Begum, 2016; Bohli *et al.*, 2015; Ribeiro and R. Braghetto, 2022]. Following, these approaches are discussed.

Consoli *et al.* [2015] propose a framework for data integration on Smart Cities using the concept of Linked Open Data. The approach proposed by Consoli *et al.* [2015] represents data as RDF triples, and ontologies are associated with each dataset imported into the proposed framework. Since the focus of the approach proposed by Consoli *et al.* [2015]

is enriching the semantics, the authors do not take into account important issues such as data ingestion, aggregation, and processing.

Silva *et al.* [2021] propose a framework named Aquedücte to provide semantic data integration in the context of smart city applications. Aquedücte provides data loading features from JSON files or Shapefiles. One advantage of Aquedücte is that it is based on the NGSI-LD protocol (Next Generation Service Interfaces - Linked Data). However, besides requiring the use of the protocol NGSI-LD (which is not supported by many applications), Aquedücte requires a data conversion step from the original format to NGSI-LD, following the semantic model defined, which generates an additional (and non-negligible) overhead.

Liu *et al.* [2017] and Jindal *et al.* [2020] propose frameworks for data management that are independent of the domain application. The proposed frameworks consider multiple requirements listed by Ribeiro and Braghetto [2021] as data ingestion, integration, and analytical queries. However, these frameworks do not take into account the importance of metadata management during the data life cycle and also do not consider privacy issues. Thus, both frameworks do not consider storing raw data through the data management process, which can compromise auditing and *post-mortem* analysis.

Costa and Santos [2017] proposes an approach to deploy data warehouses for smart cities, alongside a storage mechanism that keeps the input data in its raw format. The approach proposed by Costa and Santos [2017] is based on well-known tools and libraries, such as Talend and HDFS (Hadoop Distributed File System). The generated data warehouse can be queried using SQL via Presto. The approach proposed by Mehmood *et al.* [2019] is similar to the one proposed by Costa and Santos [2017], but the major difference is that it has its interface to query and visualization. However, the approach proposed by Costa and Santos [2017] and also the one proposed by Mehmood *et al.* [2019] do not take into account provenance data and do not capture other metadata. In addition, these approaches do not concern with data privacy issues.

Garcia-Font [2020] proposes a data management architecture focused on the user to communications in the context of smart cities. The approach proposed by Garcia-Font [2020] defines data to be managed in a decentralized way to reduce service providers' dependency but focuses only on communication applications. Zhou *et al.* [2021] propose an architecture for data management on smart cities in the health field. The approach proposed by Zhou *et al.* [2021] considers only medical records and exams, and it is not extensible to other domains. Similarly, Bellini *et al.* [2021] and Nandury and Begum [2016] also focus on a specific domain to propose their framework, *i.e.*, public transportation. The difference is that the approach proposed by Nandury and Begum [2016] focuses on data loading and transferring steps, and does not provide solutions for data integration and metadata capturing.

Liu *et al.* [2017] propose a framework for smart city data management, which takes into account data gathering, cleaning, and anonymization. The approach proposed by Liu *et al.* [2017] defines what type of anonymization must be performed according to the data classification (*i.e.*, sensitive, quasi-sensitive, and open/public). Although the approach proposed by Liu *et al.* [2017] considers privacy issues, it does not consider provenance nor considers the dataflow abstraction to support the data management.

Bohli *et al.* [2015] propose a data management framework named SMARTIE. SMARTIE is a distributed framework for IoT-based applications. It allows for the application to store, share, and query data gathered from heterogeneous data sources. Similarly to `Hurricane`, SMARTIE follows a dataflow paradigm, which is designed to offer scalable and secure information for smart city applications. Although SMARTIE represents a step forward, it does not take into account metadata management (including provenance) or privacy issues.

Ribeiro and R. Braghetto [2022] propose a microservice architecture for data integration for smart cities. This approach is an extension of the reference architecture presented in Ribeiro and Braghetto [2021]. The architecture proposed by Ribeiro and R. Braghetto [2022] supports data ingestion and integration features, but with the unique feature that is a single point of access to microservices by external applications and a centralizing interface for the services. The approach proposed by Ribeiro and R. Braghetto [2022] also collects some level of metadata, but they do not consider the concept of dataflows either collecting provenance data or using a data lake to store final and intermediate data. Table 1 summarizes the related work characteristics and compares the approaches found in the literature with `Hurricane`.

In the state of practice, Oracle [2023] also provides services that help users to store, query and process large volumes of data. Their framework also combines AI and ML features within the services, which is desirable to implement smart city applications. However, Oracle services do not support application-specific configurations, which are possible in `HURRICANE`.

# 4 Proposed Approach: `Hurricane`

`Hurricane` is a configurable and extensible data service for smart city applications. Figure 2 presents `Hurricane` architecture, where the gray components are contributions of this article. As mentioned in Section 1, `Hurricane` is designed on top of the dataflow abstraction for data management and processing, *i.e.*, the entire processing is executed after the instantiation of multiple dataflows, where each step of the data management is monitored and has its data and metadata captured. Each dataflow has a specific goal and it is automatically instantiated on Apache Airflow framework[3] based on previously defined configurations set by the user. Apache Airflow provides several fundamental features such as parallel and distributed processing that can be applied depending on the data volume that needs to be processed by `Hurricane`.

Data in `Hurricane` present a well-defined life cycle, that starts with data ingestion from external data sources. The optimization of the ingestion process is not the main focus of this article, as many solutions can be used to efficiently access and load data into `Hurricane` such as PFTP

[3] `https://airflow.apache.org`

**Table 1.** Related Work (√ = Supported, × = Not Supported, ± = Partially Supported, ⋆ = Not Available).

| Approaches | Features | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Data Warehouse | Data Lake | Semantic Support | Provenance | Anonymization | Domain Specific | Dataflow Oriented | Distributed Execution | Integration Support | Load Support | Open Source | Practical (not conceptual) |
| Consoli *et al.* [2015] | ⋆ | ⋆ | √ | × | × | × | × | × | ± | × | √ | √ |
| Bohli *et al.* [2015] | × | × | × | × | × | √ | × | × | √ | √ | √ | √ |
| Nandury and Begum [2016] | ⋆ | ± | × | × | × | √ | × | ⋆ | √ | √ | √ | √ |
| Costa and Santos [2017] | √ | √ | × | × | √ | × | × | √ | √ | √ | √ | √ |
| Liu *et al.* [2017] | √ | ± | × | × | √ | × | × | √ | √ | √ | √ | √ |
| Mehmood *et al.* [2019] | ⋆ | √ | × | × | × | × | × | √ | √ | √ | √ | √ |
| Garcia-Font [2020] | ⋆ | ± | × | × | × | √ | × | √ | √ | √ | √ | √ |
| Jindal *et al.* [2020] | √ | ± | × | × | × | × | × | √ | √ | √ | √ | √ |
| Bellini *et al.* [2021] | ± | √ | √ | × | × | √ | × | ⋆ | ± | √ | √ | √ |
| Ribeiro and Braghetto [2021] | ± | √ | × | √ | × | × | × | √ | √ | √ | √ | × |
| Silva *et al.* [2021] | ± | ± | √ | × | × | × | × | ± | √ | √ | √ | √ |
| Zhou *et al.* [2021] | ⋆ | ± | √ | × | × | √ | × | ⋆ | √ | √ | √ | √ |
| Ribeiro and R. Braghetto [2022] | ± | √ | × | √ | × | × | × | √ | √ | √ | √ | √ |
| `Hurricane` | √ | √ | × | √ | √ | × | √ | √ | √ | √ | √ | √ |

[Bhardwaj and Kumar, 2005] or GridFTP [Radic *et al.*, 2007]. `Hurricane` imports data from different sources, whether they are structured or not. The requirement is an available API that can be used to access external data. The only exception to this requirement is when importing topological data from cities (most smart city applications require this type of data as data are georefenced). Thus, data associated with cities topology are directly imported from Open Street Map (OSM)[4], a collaborative mapping project that aims at creating an editable map of the Earth, using a specific ETL component to collect data from Open Street Map API (Step 1 on Figure 2), called `ETL[OSM]`. `ETL[OSM]` extends OSMnx library [Boeing, 2017], which enables cities' streets network modeling and many other operations based on geospatial geometry available on OSM. `ETL[OSM]` downloads this information in JSON format and stores it on a `Hurricane` Data Lake (Step 2). Once the JSON is loaded, a graph that represents the city map is created. This graph is presented in two dataframes that contain nodes and edges. The edges are partitioned into segments with approximately 100 meters of distance. Besides the graph, the street metadata are also collected, *e.g.*, *oneway*, which indicates that the street goes just one way and *highway* which defines the Street/Highway type. Figure 3 presents an example of a graph generated for a small region around the Assis Chateaubriand Art Museum of São Paulo (MASP) located on Paulista Avenue in the city of São Paulo, Brazil. Algorithm 1 presents the procedure executed to create the segments of streets with data downloaded using `ETL[OSM]`. The algorithm interpolates the points in the downloaded city geometry to create segments of street of approximately 100 meters.

The `ETL[OSM]` component creates a dataflow dynamically to extract necessary information from OSM to identify the streets of a city, *i.e.*, its topology. `ETL[OSM]` instantiates the dataflow based on a configuration file similar to the fragment presented on the Listing 1. The *workflow_type* tag determines the dataflow type that will be created dynamically, in this case, "model", because it generates a city topology as output. The *datalake_client* tag defines the connectors to storage, in this case *hdfs*, the standard distributed file system of the Hadoop stack. The *datalake_workdir* tag defines the working directory where all data are going to be generated on the data lake. The *retries* tag defines the number of attempts that have to be considered when an error occurs, while *owner* represents the identifier of the user responsible for the data loading. The *metadata_url, schema* and *tablespace* tags are the parameters used to connect with the metadata database. Finally, the city that has to be considered to generate the street topology is defined by the tags *city, state* and *country* on *places* tag. The other data from external sources are accessed by their respective `External APIs` (Step 3) by the `Data Ingestor` (Step 4). This component receives data from the APIs and loads them onto the data lake on the specific area of each data type on the `Distributed Storage`.

**Algorithm 1** Street Segmentation

```
$edges = getEdges()$                                  ▷ Get available edges via OSMnx.
$segments = \varnothing$
while $!(isEmpty(edges))$ do
    $vertices = redistributeVertices(edge.geometry, 100)$
    while $!(isEmpty(vertices))$ do
        $startVertex = vertices.getCurrentPoint()$
        $endVertex = vertices.getNextPoint()$
        $distance = getDistance(startVertex, endVertex)$
        $addresss = vertice.getName()$
        $type = vertice.getHigWay()$
        $oneWay = vertice.getOneWay()$
        $s = createSegment(startVertex, endVertex,$
         $address, type, oneWay, distance)$
        $segments.add(s)$
    end while
end while
```

Once the raw data and cities' segment graph are stored on the data lake, the dataflow to process and integrate data can be enacted (Step 5). The data processing on `Hurricane` is organized in four main layers: (i) `ETL Raw`, (ii) `ETL Bronze`,

[4] `https://www.openstreetmap.org`

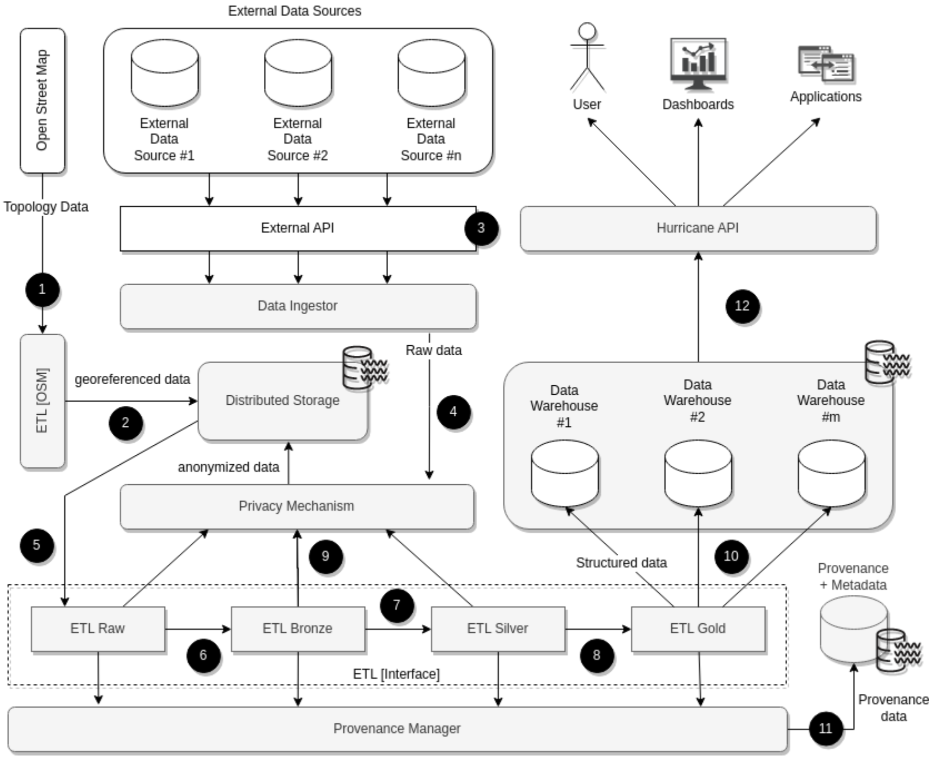


**Figure 2.** Hurricane Architecture.

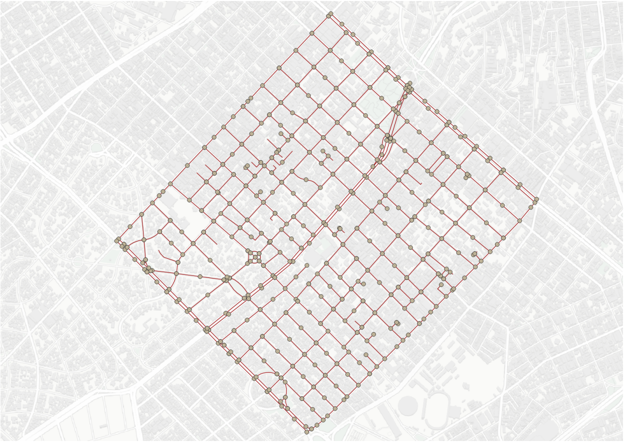


**Figure 3.** Graph representing the region around the Assis Chateaubriand Art Museum of São Paulo (MASP) located on Avenida Paulista - Adapted from Cunha Sá *et al.* [2022].

(iii) ETL Silver, and (iv) ETL Gold. In each layer, the data goes through a series of data transformations performed by specific dataflows, one for each layer. These dataflows are instantiated with a configuration file similar to the fragment presented on the Listing 2. The *workflow_type* tag defines the dataflow type that has to be created dynamically, in this case, "general". The *dag_id* tag represents the dataflow unique identifier. The *related_dag_id* tag represents the dataflow responsible for creating the city graph with street segments and that is going to be integrated into the other downloaded data. The *schedule_interval* tag is a configuration used by Apache Airflow scheduler to monitor all the dataflows tasks that are being executed. The *max_concurrency* tag defines the number of threads that can be executed in parallel, respecting its data dependencies. The *raw_interfaces* subgroup defines the interfaces that execute the data mappings contained on raw files stored in the data lake with the topology data obtained from OSM.

Each interface has a unique identifier called *name*, a directory where the raw data can be gathered on the data lake represented on the directory defined in *input_path* tag. Furthermore, the *rules_columns* tag defines the input file data mapping with the data extracted from OSM. This mapping is automatically executed using global variables *date*, *period*, *latitude* and *longitude*, that are mapped to the attributes contained on the raw data files, *i.e.*, the user has to map the global variables to the fields of the raw data file. On the fragment presented on the Listing 2, *date* is mapped to the attribute DATA_OCORRENCIA, *period* to PERIODO_OCORRENCIA, *latitude* and *longitude* to attributes with the same names. It is worth noticing that Hurricane performs semiautomatic data integration since the user has to make the mapping manually. Although in its current version Hurricane does not provide automatic integration components, the architecture could be extended to consider approaches such as the ones proposed by Salvadores *et al.* [2009] and Pinkel *et al.* [2015]. On *feature_columns* tag additional characteristics that should be considered are defined, *e.g.*, *duplicated_key*, that defines the key attribute that has to be considered to discard duplicated tuples in the dataset.

When the input parameters are defined, Hurricane starts to instantiate the dataflows. The dataflow associated with the ETL Raw layer is responsible for identifying duplicated registers (according to the configurations informed on the configuration file) and defining a key attribute in the raw data. Afterward, the dataflow associated with the ETL Bronze is instantiated (Step 6). At this point, summarizations and data aggregation are executed. By default, these aggregations are executed using the values of latitude/longitude and date (*i.e.*,

```
    "workflow_type"     : "model",
    "dag_id"            : "Generate-City-Model",
    "datalake_client"   : "hdfs",
    "datalake_workdir"  : "/datalake/model/SP",
    "retries"           : 5,
    "owner"             : "Anonimyzed",
    "metadata_url"      : "127.0.0.1:5432/cdbase",
    "schema"            : "public",
    "tablespace"        : "pg_default",
    "places"            : [
        {
            "city"      : "Sao Paulo",
            "state"     : "Sao Paulo",
            "country"   : "Brazil"
        }
    ]
```

**Listing 1:** A fragment of the configuration file of ETL[OSM] component.

space and time aggregations). It is worth noticing that the aggregation function used aggregation has to be configured on the interface (on Listing 2 the function COUNT was the one defined), and the user can define other dimensions to aggregate on the configuration file (*e.g.*, on the field ATT_DIMENSION). As soon as the aggregations are finished, the dataflow associated with the ETL Silver is instantiated (Step 7). The ETL Silver associates aggregated data with the topology data extracted from OSM, *i.e.*, the graph containing streets' segments. It is important to mention that not always the latitude and longitude informed on the aggregated data represent a point in one of the street segments defined by the OSM data. Thus, the ETL Silver identifies which street segment is the closest to the point in question. Finally, the dataflow associated with the ETL Gold (Step 8) is instantiated and creates a data warehouse (DW) for each smart city domain. By default some tables are created: a Fact Table and multiple Dimension Tables, which are related to SEGMENT, VERTEX, and TIME, to represent the spatial and temporal components. However, in case other attributes have been informed on ATT_DIMENSION, new dimension tables are created according to these attributes' domain. Figure 4 presents the template to create the data warehouse in Hurricane with the Fact Table, the TIME dimension and the spatial dimensions SEGMENT, VERTEX, NEIGHBORHOOD and DISTRICT and ZONE to represent the location. The tables DIMENSION #1, DIMENSION #2 and DIMENSION #d represent the possible $d$ dimensions that can be created by Hurricane.

It is important to emphasize that during the execution of the ETL Gold, all integration between data loaded from external sources and topological data was previously executed by the ETL Silver layer. So, the only responsibility of the ETL Gold layer is to consume the data on the data lake to generate the final vision of the data that are going to be consumed by the end user. The intermediate data produced by each dataflow are stored on the data lake in a way that they can be used in the future (Step 9) and later synchronized by a full overwrite on the data warehouse modeled on PostgreSQL (Step 10). These data can optionally be anonymized by Pseudonymization or by differential privacy mechanisms [Dwork and Lei, 2009] using the Privacy Mechanism. Although most anonymization mechanisms were initially proposed in the context of interactive queries, there is also the use of anonymization approaches for the publication of anonymized datasets, when the anonymization is applied to the dataset before publishing it and queries are performed on the already anonymized data [de Oliveira *et al.*, 2019a; Bertelli *et al.*, 2022].

```
{
   "workflow_type"     : "general",
   "dag_id"            : "Hurricane-PM-Crimes",
   "related_dag_id"    : "Generate-City-Model",
   "datalake_client"   : "local",
   "datalake_workdir"  : "/datalake/model/SP",
   "max_concurrency"   : 4,
   "fact_tablename"    : "CRIME",
   "aggregation"       : "COUNT",
   "raw_interfaces"    : [
       {
           "name": "vehicles_robbery",
           "input_path": "/raw/vehicles_robbery/",
           "header": 0,
           "rules_columns": {
               "date"       : "DATA_OCORRENCIA" ,
               "period"     : "PERIODO_OCORRENCIA",
               "latitude"   : "LATITUDE",
               "longitude"  : "LONGITUDE"
           },
           "duplicated_key" : ["ANO_BO", "NUM_BO"],
           "att_dimension": [{"name":"TIPO_CRIME",...]
       } ...
```

**Listing 2:** A fragment of the ETL Configuration Template for ETL Raw, ETL Bronze, ETL Silver and ETL Gold.

Finally, all provenance data are collected by the Provenance Manager and stored in a specific database (Step 11) that contains all the data derivation paths. In Figure 4 all tables in yellow represent the provenance data. One can note that in all data transformations, the user that executed the transformation is registered in the database. It is worth mentioning that existing solutions for capturing provenance can be used in this context. In the current version of Hurricane, the DfAnalyzer [Silva *et al.*, 2020] is used to capture provenance data. Finally, an application or the end user can consume the integrated and aggregated data via Hurricane API (Step 12). Hurricane is being open-sourced and its source code is available at https://github.com/UFFeScience/Hurricane.

# 5 Experimental Evaluation

In this section, we present Hurricane evaluation, both quantitative and qualitative. Firstly, we discuss the chosen case study and the environment setup and then the evaluation results.

## 5.1 Case Study

The application chosen as the case study is associated with the analysis of crime Hot Spots [Kikuchi *et al.*, 2012], a cluster of crime events distributed across space. Analyzing crime hot spots is an important task to identify places with high crime rates and promote predictive policing strategies [Lourenço *et al.*, 2018]. The idea behind predictive policing

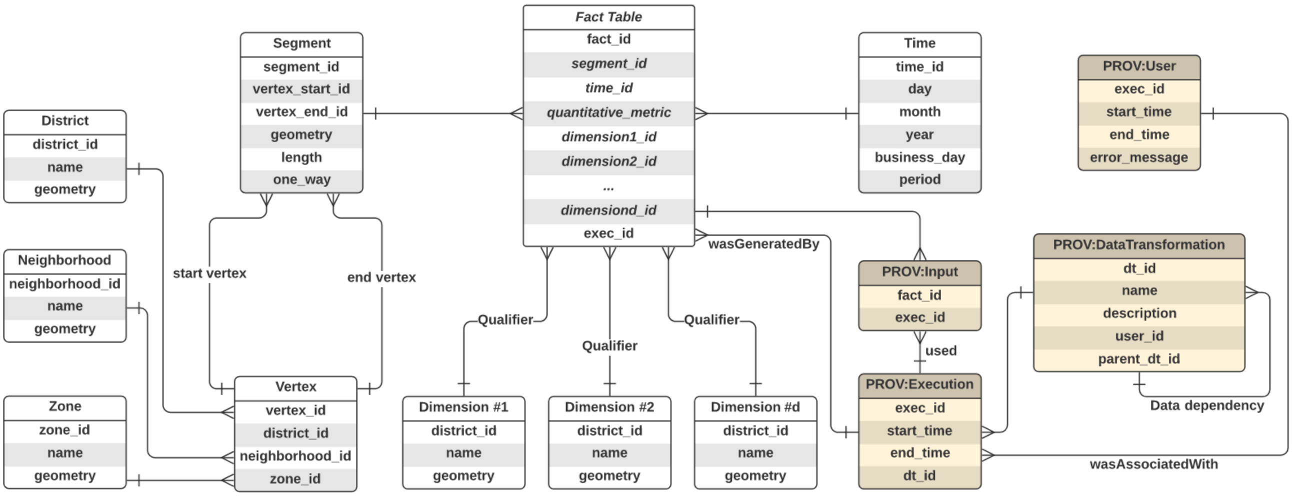


**Figure 4.** *Data Warehouse* Creation *Template* on Hurricane.

is to make the police present to the citizens, either through the police officers' presence at strategic points in the city or with police patrols. However, in large cities such as Rio de Janeiro and São Paulo, analyzing the crime hots spots is far from trivial.

To illustrate the presence of crime hot spots, let us take as an example two regions of the city of São Paulo: "Alto da Mooca" and "Itaim Paulista". Figure 5 presents the heat map of the streets to identify the hot spots, where the color scale varies from light yellow to dark red, and the darker the color, the higher the crime rate. In Figure 5(a), most of the streets from "Alto da Mooca" do not show any crime event in the selected period (November/2019). On the other hand, in Figure 5(b), most of the streets from the "Itaim Paulista" region present a high crime rate in the same period, characterizing a hot spot. To identify the hot spots, the crime data have to be aggregated in space and time, *i.e.*, by street segments on the map and by the period of the year. We used Hurricane to manage and integrate those data in our experiments.

To perform the experiments presented in this section, the crime data were downloaded from São Paulo's SSP-SP website (`https://www.ssp.sp.gov.br/`). We considered the crimes that occurred in 2019. It is worth noticing that not all crime types that can be accessed on the website were downloaded. In addition, all data provided by SSP-SP contains all attributes of a police report, *i.e.*, private and sensitive information such as name, name of the mother, *etc*. We have just considered eight crime types in the experiments, *i.e.*, femicide, car theft, car robbery, cellphone theft, cellphone robbery, theft, armed robbery, and homicide. Furthermore, some crime events are not georeferenced (*i.e.*, with latitude e longitude), in a way that enables the integration of the crime data with the map segments generated by Hurricane. Thus, Figure 6 presents the total number (in blue) and the total number of validated crime events (in red) after being processed by the Hurricane. It is worth noticing that the y-axis is in a log scale.

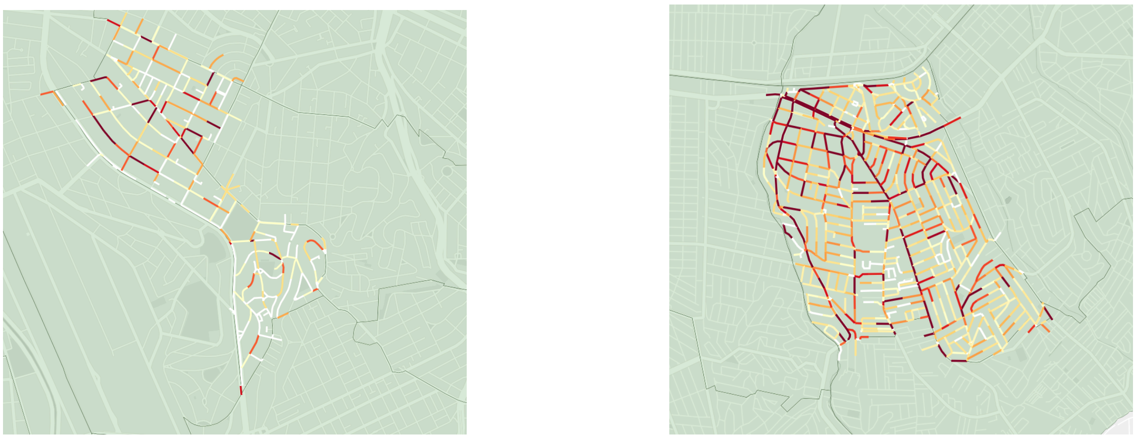
## 5.2 Environment Setup

For the experiments executed in this article, we have deployed Hurricane on top of the Amazon AWS cloud. Amazon AWS is one of the most popular cloud computing environments and provides different types of virtual machines to be deployed on demand. Although there are more than 100 types of virtual machines, in the experiments presented in this article we have considered the `a1.xlarge` virtual machine. A1 virtual machines have AWS Graviton Processors. The virtual machine has 4 vCPUs and 8 GiB RAM. The virtual machine was configured to be accessed using SSH without password checking (although this is not recommended due to security issues).

## 5.3 Quantitative Evaluation

In the quantitative evaluation, we follow the evaluation strategy proposed by Ribeiro and R. Braghetto [2022]. The idea is to measure and analyze the performance for the functionalities of Hurricane under both normal and above-normal workload conditions. Firstly, we have measured the performance of the `Data Ingestor` component. The idea is to measure the latency to load data from external sources to Hurricane distributed storage. Thus, we executed an experiment that submits concurrent requests to Hurricane to download the crime dataset explained in Subsection 5.1, but for all years from 2003 to 2019. We varied the number of concurrent requests from 1 to 200 requests per second. Figure 7 shows the download time degradation (in seconds) as the number of concurrent requests is increased. It is worth noticing that the download times do not take into account the OSM data, which is performed by a different component in the architecture, *i.e.*, `ETL[OSM]`.

After analyzing the time required to download data into the Hurricane architecture, we need to analyze the time required for the service to execute each of the ETL layers mentioned in Section 4. Figure 8 presents the average processing time (*i.e.*, $\overline{x}$) and the standard deviation (*i.e.*, $\sigma$) of 10 execution of each Hurricane dataflow in seconds. Analyzing Figure 8 it is possible to observe that even when a dataset is considered with only one year of data (*i.e.*, the year of 2019), the larger execution times are on the processing steps of cities' data, on integration step and data summarization. In special,



**(a)** "Alto da Mooca" region.

**(b)** "Itaim Paulista" region.

**Figure 5.** Crime occurrences on "Alto da Mooca" and "Itaim Paulista" in November/2019.

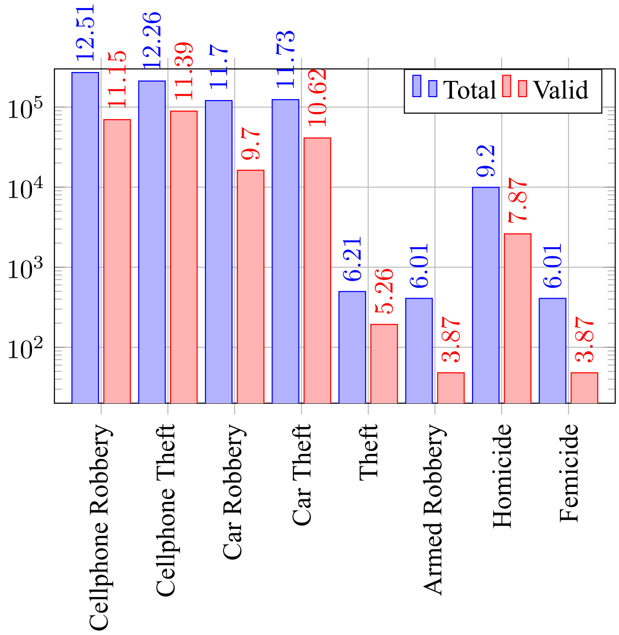


**Figure 6.** Number of crime events in the downloaded dataset

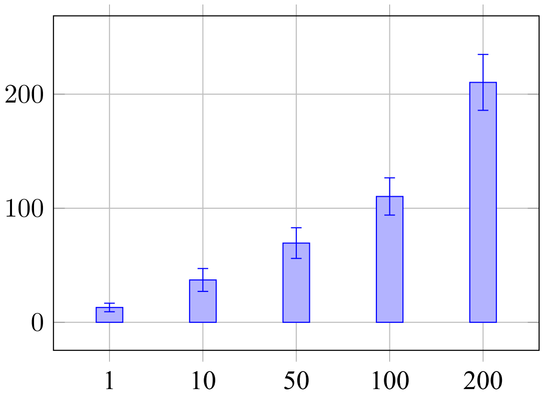


**Figure 7.** Download time of crime dataset in Hurricane.

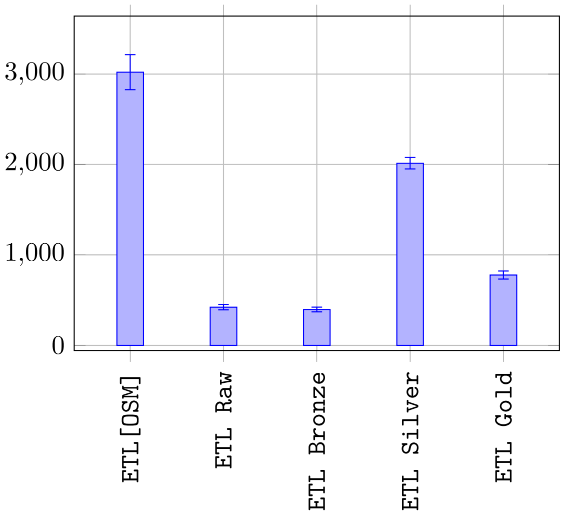


**Figure 8.** Execution time for the ETL steps in Hurricane.

processing the city topology and generating the street graph need to obtain data from external sources the metadata from neighborhoods and city zones, that are not available automatically by OSMnx. However, these times are acceptable considering the volume of data.

## 5.4 Privacy Mechanism Evaluation

In order to evaluate the privacy mechanism of HURRICANE, the original dataset downloaded from SSP-SP was considered. In this subsection, we present the results obtained by applying the differential privacy technique. Differential Privacy [Dwork *et al.*, 2006a] is a mathematical model that allows for statistical analysis of a dataset without compromising the privacy of individuals. With strong privacy guarantees, differential privacy is based on a mechanism that introduces random noise into a query response. By using differential privacy mechanisms, the user has to measure the tradeoff between utility and privacy and calibrate the mechanism. In this article, we specifically used the mechanisms of Laplace and Gaussian. We will briefly discuss each of these mechanisms.

The Laplace mechanism [Dwork *et al.*, 2006b] provides differential privacy for queries that return numerical values and is commonly applied to statistical queries. This mech-

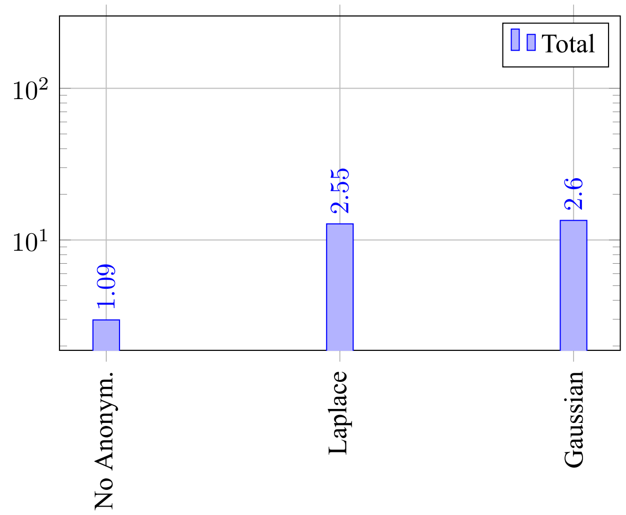


**Figure 9.** Elapsed time for anonymizing the raw data in HURRICANE (in seconds).

anism introduces random noise into the original response based on the Laplace distribution, which is determined by the parameters $\epsilon$ and $\Delta f$. The probability density function of the Laplace distribution is given by $Lap(z|b) = \frac{1}{2b}\exp\left(-\frac{|z|}{b}\right)$. The Gaussian mechanism [Dwork *et al.*, 2014] involves adding random noise following the Gaussian distribution. But, differently from the Laplace mechanism, the Gaussian mechanism does not satisfy $\epsilon$-differential privacy but rather $(\epsilon, \delta)$-differential privacy, where $\delta$ is a relaxation factor applied. The Gaussian distribution is given by $f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$, where $\mu$ is the mean, $\sigma$ is the standard deviation, and $\sigma^2$ is the variance. Thus, for any arbitrary $d$-dimensional function, let $f : \mathbb{N}^{|x|} \rightarrow \mathbb{R}^d$, the Gaussian mechanism [Dwork *et al.*, 2014] is defined as $M_G(x, f, \epsilon) = f(x) + (Y_1...Y_k)$, where $Y_i \sim Gau\left(\frac{c\Delta_2(f)}{\epsilon}\right)$, i.i.d.

Following the protocol defined by Bertelli *et al.* [2022], in this evaluation, we explore the following values for $\epsilon = \{0.01, 0.05, 0.1, 0.25, 0.5, 1\}$ for both Laplace and Gaussian mechanisms. Both the Laplace and Gaussian mechanisms were centered around a mean of $\mu = 0$. We also defined a query that encompasses the necessary crime aggregation per street segment. Thus, the query used in this experiment is "What is the total number of femicide, car theft, car robbery, cellphone theft, cellphone robbery, theft, armed robbery, and homicide per street segment?" This query counts the crimes in each police report per street segment.

Figure 9 presents the average execution time (10 executions), in seconds, of the aforementioned query without anonymization and with anonymization for each mechanism. This allows for the analysis of the overhead introduced by the Privacy Mechanism in HURRICANE. These values consider that the $\Delta f$ has been calculated *a priori*. One can conclude that with the pre-calculated $\Delta f$, the overhead for the Laplace and Gaussian mechanisms is acceptable. Similarly to the results presented by Bertelli *et al.* [2022], the Laplace mechanism showed a slightly lower execution time than the Gaussian mechanism.

Let us now analyze the utility of the data after anonymization. We use Relative Error metric in this analysis. This value is directly linked to the distribution of the data in the downloaded dataset, *i.e.*, how widely the data is distributed, and how much its absence impacts the query result. This information is measured based on the value of $\Delta f$. By analyzing Table 2, one can observe how the relative error behaves with the variation of $\epsilon$ values in the range $[0.01, 0.05, 0.1, 0.25, 0.5, 1.0]$ for the aforementioned query. Low values of $\epsilon$, although ensuring more privacy for individuals in the downloaded dataset, imply reduced utility. Note that as the value of $\epsilon$ increases, the relative error decreases, and this is independent of the mechanism used. Thus, an expert user is required to analyze the tradeoff between utility and privacy according to the context and requirements of the application. Analyzing Table 2, one can state that the Relative Error for all values of $\epsilon$ presents acceptable values that preserve the utility of the data. It is worth noticing that for $\epsilon = 0.01$, the relative error decreases an order of magnitude compared to the next value of $\epsilon = 0.05$. Table 2, referring to the Gaussian mechanism, also presents high relative error values for the query. In this specific case, we can see that values from $\epsilon = 0.1$ already yield results with a certain level of utility. This way, the Privacy Mechanism of HURRICANE can anonymize data in a timely manner while maintaining the utility of data.

## 5.5 Qualitative Evaluation

In the qualitative evaluation we performed a viability study with experts in public security to evaluate the following Research Question: (RQ1) "Does Hurricane support users and developers of Smart Cities' applications to manage and integrate data?". The subjects of the study were selected according to their occupation area, *i.e.*, public security experts. In total, five experts were chosen to participate. The main idea was that the participants should evaluate the Hurricane data management to support hot spot analysis. Based on the answers, it can be observed that all the participants have an undergraduate degree. The experts' degrees are in the areas of Mathematics, Statistics, and Social Sciences. Among the specialists, 60% worked in the area for more than 10 years and 40% between 5 and 10 years.

The idea is that the subjects load the dataset into Hurricane and submit queries to identify crime hot spots after data integration. Firstly, the subjects were trained to analyze the data. The training respected the same roadmap for all subjects to avoid biases. After the use of Hurricane, a questionnaire was available to evaluate the proposed data service. The questionnaire is based on the Technology Acceptance Model (TAM) [Davis, 1989], which was extensively used in similar contexts [de Souza *et al.*, 2015]. Inspired by the results of de Souza *et al.* [2015], we used the TAM approach to capture the user perception of Hurricane regarding its utility and ease-of-use. Table 3 shows the questions used for the TAM evaluation of Hurricane. Each question within the questionnaires follows a Likert scale [de Souza *et al.*, 2015], and the user must select only one option between (a) Very Low, (b) Low, (c) Medium, (d) High, and (e) Very High.

Table 3 presents the overall results of TAM questionnaires, in which most of the surveyed experts (40% high and 20% very high) indicate Hurricane applies to their daily rou-

**Table 2.** Relative error of the Laplace and Gaussian mechanisms for different values of $\epsilon$.

| | | $\epsilon$ | | | | | |
|---|---|---|---|---|---|---|---|
| Mechanism | $\Delta f$ | 0.01 | 0.05 | 0.1 | 0.25 | 0.5 | 1.0 |
| Laplace | 0.040 | 1.380 | 0.070 | 0.010 | 0.003 | 0.001 | 0.001 |
| Gaussian | 0.040 | 1.930 | 1.120 | 0.320 | 0.040 | 0.010 | 0.005 |

**Table 3.** `Hurricane` evaluation with TAM questionnaire.

| **Question** | **Very Low** | **Low** | **Medium** | **High** | **Very High** |
|---|---|---|---|---|---|
| Which is the applicability of `Hurricane` in your daily duties? | 0.00% | 20.00% | 20.00% | **40.00%** | 20.00% |
| Which would be the data-driven performance enhancement if you adopted `Hurricane`? | 0.00% | 20.00% | 0.00% | **80.00%** | 0.00% |
| Which is the information quality level presented by `Hurricane`? | 0.00% | 0.00% | 0.00% | **80.00%** | 20.00% |
| How easy was it to use `Hurricane`? | 0.00% | **60.00%** | 20.00% | 20.00% | 0.00% |

tines. However, more than 60% of them also indicated using, identifying, and fixing data errors in `Hurricane` is not simple. This indicates that `Hurricane` requires refactoring to improve usability. Notice that `Hurricane` was first implemented as a proof of concept rather than a final product. All users highlighted that would be interesting the development of a dashboard integrated to `Hurricane` to have the partial visualization of data during the processing (*i.e.*, human-in-the-loop for data analysis). In this qualitative evaluation, some threats to validity were identified. Firstly, the data update control. On the current version of `Hurricane`, there is data update control, *i.e.*, `Hurricane` depends on the user to upload new data. Besides, the capacity of supporting a big quantity of simultaneous accesses was not analyzed on `Hurricane`.

# 6 Conclusions

This article presents a data service for smart city applications called `Hurricane`. `Hurricane` is able to import data from multiple data sources and integrate them according to a mapping provided by the users. `Hurricane` is derived from applied research in a multidisciplinary project that includes public security experts from Military Police from Rio de Janeiro and also computer science experts from the Universidade Federal Fluminense (UFF).

Both quantitative and qualitative evaluations were conducted on the public security domain with domain experts to analyze if `Hurricane` in fact offers support on the capture, storage, aggregation, and query to an application of crime hot spots analysis. The subjects of the study reinforced the importance of services such as `Hurricane`, which are able to obtain data from different sources, process and integrate them and make them available in an efficient and agile manner. Even though the `Hurricane` deployment represents an important step, the users requested improvements in usability and the inclusion of new features.

Thus, as future work, we plan the integration of a dashboard on the service to visualize the generated data, and the inclusion of semantic support, so `Hurricane` will be able to identify new relationships on the data and implicit knowledge through inference. Additionally, evaluations on other application domains are already planned, in special on rainfall data analysis in big urban centers. Finally, other mapping types are possible but have not been explored. New file formats will be also considered as JSON, Parquet, and ORC, together with the use of Apache Spark as distributed processing framework.

## Funding

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. This research was also funded by CNPq and FAPERJ.

## Authors' Contributions

Maicon Banni, Isabel Rosseti, and Daniel de Oliveira contributed to the conception of this study. Maicon Banni, Maria Falci, and Daniel de Oliveira performed the experiments. Maicon Banni is the main contributor and writer of this manuscript. All authors read and revised the final manuscript.

## Competing interests

The authors declare that they have no competing interests.

## Availability of data and materials

The datasets generated and/or analyzed during the current study are available in `https://osf.io/mxrgu/` and `https://github.com/UFFeScience/Hurricane`.

# References

Ahmad, K., Maabreh, M., Ghaly, M., Khan, K., Qadir, J., and Al-Fuqaha, A. (2022). Developing future human-centered smart cities: Critical analysis of smart city security, data management, and ethical challenges. *Computer Science Review*, 43:100452.

Banni, M., Rosseti, I., and de Oliveira, D. (2022). Hurricane: um serviço para gerência de dados de aplicações de cidades inteligentes. In *Anais do XXXVII Simpósio Brasileiro de Bancos de Dados*, pages 151–163, Porto Alegre, RS, Brasil. SBC.

Bellini, E., Bellini, P., Cenni, D., Nesi, P., Pantaleo, G., Paoli, I., and Paolucci, M. (2021). An ioe and big multimedia data approach for urban transport system resilience management in smart cities. *Sensors*, 21:435. DOI: 10.3390/s21020435.

Bertelli, L., Ströele, V., Machado, J. C., and de Oliveira, D. (2022). Privacidade diferencial em sistemas polystore: uma abordagem prática. In *2022: Proceedings of the 37th Brazilian Symposium on Databases, SBBD 2022, Buzios, Brazil, September 19 -23, 2022*, pages 279–291. SBC. DOI: 10.5753/sbbd.2022.224305.

Bhardwaj, D. and Kumar, R. (2005). A parallel file transfer protocol for clusters and grid systems. In *First International Conference on e-Science and Grid Computing (e-Science'05)*, pages 7 pp.–254.

Bilal, M., Usmani, R. S. A., Tayyab, M., Mahmoud, A. A., Abdalla, R. M., Marjani, M., Pillai, T. R., and Targio Hashem, I. A. (2020). *Smart Cities Data: Framework, Applications, and Challenges*, pages 1–29. Springer International Publishing, Cham.

Boeing, G. (2017). Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Comp., Env. and Urban Sys.*, 65:126–139. DOI: https://doi.org/10.1016/j.compenvurbsys.2017.05.004.

Bohli, J.-M., Skarmeta, A., Victoria Moreno, M., García, D., and Langendörfer, P. (2015). Smartie project: Secure iot data management for smart cities. In *2015 International Conference on Recent Advances in Internet of Things (RIoT)*, pages 1–6.

Brito, J. J. (2018). *Data Warehouses in the era of Big Data: efficient processing of Star Joins in Hadoop*. Computer science and computational mathematics, ICMC-USP.

Chawathe, S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman, J., and Widom, J. (1994). The tsimmis project: Integration of heterogenous information sources. In *Information Processing Society of Japan*.

Chen, H., Cheng, T., and Wise, S. (2017). Developing an online cooperative police patrol routing strategy. *Computers, Environment and Urban Systems*, 62:19–29. DOI: https://doi.org/10.1016/j.compenvurbsys.2016.10.013.

Ciobanu, M. G., Fasano, F., Martinelli, F., Mercaldo, F., and Santone, A. (2019). A data life cycle modeling proposal by means of formal methods. In *Proceedings of the Asia Conference on Computer and Communications Security*, page 670, New York, NY, USA. Association for Computing Machinery.

Consoli, S., Mongiovì, M., Nuzzolese, A. G., Peroni, S., Presutti, V., Recupero, D. R., and Spampinato, D. (2015). A smart city data model based on semantics best practice and principles. In *WWW 2015*, pages 1395–1400. ACM. DOI: 10.1145/2740908.2742133.

Costa, C. and Santos, M. Y. (2017). The suscity big data warehousing approach for smart cities. IDEAS 2017, page 264–273, New York, NY, USA. ACM. DOI: 10.1145/3105831.3105841.

Cunha Sá, B., Muller, G., Banni, M., Santos, W., Lage, M., Rosseti, I., Frota, Y., and de Oliveira, D. (2022). Polroutedds: a crime dataset for optimization-based police patrol routing. *Journal of Information and Data Management*, 13(1).

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*, pages 319–340.

de Oliveira, D., Rodrigues, E., Costa, S., Amora, P. R. P., Caldas, A., Horta, M., de Fillippis, A. M., Ocaña, K. A. C. S., Vidal, V. M. P., and Machado, J. C. (2019a). Um estudo comparativo de mecanismos de privacidade diferencial sobre um dataset de ocorrências do ZIKV no brasil. In *XXXIV Simpósio Brasileiro de Banco de Dados, SBBD 2019, Fortaleza, CE, Brazil, October 7-10, 2019*, pages 253–258. SBC. DOI: 10.5753/sbbd.2019.8832.

de Oliveira, D. C. M., Liu, J., and Pacitti, E. (2019b). *Data-Intensive Workflow Management: For Clouds and Data-Intensive and Scalable Computing Environments*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers. DOI: 10.2200/S00915ED1V01Y201904DTM060.

de Souza, I. E., Oliveira, P. H. L., Bispo, E. L., Inocencio, A. C. G., and Parreira, P. A. (2015). TESE - an information system for management of experimental software engineering projects. In Siqueira, S. W. M. and Carvalho, S. T., editors, *Proceedings of the Brazilian Symposium on Information Systems*, pages 563–570. ACM.

Dwork, C. and Lei, J. (2009). Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 371–380.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006a). Calibrating noise to sensitivity in private data analysis. In Halevi, S. and Rabin, T., editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006b). Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer.

Dwork, C., Roth, A., *et al.* (2014). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407.

Freire, J., Koop, D., Santos, E., and Silva, C. T. (2008). Provenance for Computational Tasks: A Survey. *Computing in Science & Engineering*, pages 20–30.

Garcia-Font, V. (2020). Socialblock: An architecture for decentralized user-centric data management applications for communications in smart cities. *JPDC*, 145:13–23. DOI: 10.1016/j.jpdc.2020.06.004.

Ikeda, R., Sarma, A. D., and Widom, J. (2013). Logical provenance in data-oriented workflows? In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 877–888. IEEE.

Jindal, A., Kumar, N., and Singh, M. (2020). A unified framework for big data acquisition, storage, and analytics for demand response management in smart cities. *FGCS*, 108:921–934. DOI: 10.1016/j.future.2018.02.039.

Kikuchi, G., Amemiya, M., and Shimada, T. (2012). An analysis of crime hot spots using GPS tracking data of children and agent-based simulation modeling. *Ann. GIS*, 18(3):207–223. DOI: 10.1080/19475683.2012.691902.

Kimball, R. and Ross, M. (2002). *The data warehouse* *toolkit: the complete guide to dimensional modeling, 2nd Edition*. Wiley.

Liu, X., Heller, A., and Nielsen, P. S. (2017). Citiesdata: a smart city data management framework. *Knowl. Inf. Syst.*, 53:699–722. DOI: 10.1007/s10115-017-1051-3.

Liu, Z. and Heer, J. (2014). The effects of interactive latency on exploratory visual analysis. *IEEE transactions on visualization and computer graphics*, 20:2122–2131.

Lourenço, V., Mann, P., Guimaraes, A., Paes, A., and de Oliveira, D. (2018). Towards safer (smart) cities: Discovering urban crime patterns using logic-based relational machine learning. In *2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8-13, 2018*, pages 1–8. IEEE. DOI: 10.1109/IJCNN.2018.8489374.

Mehmood, H., Gilman, E., Cortes, M., Kostakos, P., Byrne, A., Valta, K., Tekes, S., and Riekki, J. (2019). Implementing big data lake for heterogeneous data sources. In *ICDEW 2019*, pages 37–44. DOI: 10.1109/ICDEW.2019.00-37.

Miller, R. J. (2018). Open data integration. *Proc. VLDB Endow.*, 11(12):2130–2139.

Nandury, S. V. and Begum, B. A. (2016). Strategies to handle big data for traffic management in smart cities. In *ICACCI 2016, India*, pages 356–364. IEEE. DOI: 10.1109/ICACCI.2016.7732072.

Nargesian, F., Zhu, E., Miller, R. J., Pu, K. Q., and Arocena, P. C. (2019). Data lake management: Challenges and opportunities. *Proc. VLDB Endow.*, 12:1986–1989. DOI: 10.14778/3352063.3352116.

Oracle (2023). Oracle Smart Cities. `https://www.oracle.com/au/government/smart-cities/`. Accessed: July 10, 2023.

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information & Software Technology*, 64:1–18.

Pinkel, C., Binnig, C., Jiménez-Ruiz, E., May, W., Ritze, D., Skjæveland, M. G., Solimando, A., and Kharlamov, E. (2015). RODI: A benchmark for automatic mapping generation in relational-to-ontology data integration. In Gandon, F., Sabou, M., Sack, H., d'Amato, C., Cudré-Mauroux, P., and Zimmermann, A., editors, *The Semantic Web. Latest Advances and New Domains - 12th European Semantic Web Conference, ESWC 2015, Portoroz, Slovenia, May 31 - June 4, 2015. Proceedings*, volume 9088 of *Lecture Notes in Computer Science*, pages 21–37. Springer.

Pisco, V. G. and Marques-Neto, H. T. (2021). iwalk: Uma solução para medição e análise da caminhabilidade de cidades com portais de dados abertos. In *Anais do V Workshop de Computação Urbana*, pages 84–97. SBC.

Radic, B., Kajic, V., and Imamagic, E. (2007). Optimization of data transfer for grid using gridftp. In *2007 29th International Conference on Information Technology Interfaces*, pages 709–715.

Raghavan, S., Boung Yew, S. L., Lee, Y. L., Tan, W., and Kee, K. K. (2019). *Data Integration for Smart Cities: Opportunities and Challenges*, pages 393–403. DOI: 10.1007/978-981-15-0058-9$_{3}$8.

Ribeiro, M. and R. Braghetto, K. (2022). A scalable data integration architecture for smart cities: Implementation and evaluation. *Journal of Information and Data Management*, 13(2).

Ribeiro, M. B. and Braghetto, K. R. (2021). A data integration architecture for smart cities. In *SBBD 2021, Rio de Janeiro, Brazil*, pages 205–216. SBC. DOI: 10.5753/sbbd.2021.17878.

Ribeiro, M. W. M., Lima, A. A. B., and de Oliveira, D. (2020). OLAP parallel query processing in clouds with c-pargres. *Concurr. Comput. Pract. Exp.*, 32(7). DOI: 10.1002/cpe.5590.

Salvadores, M., Correndo, G., Rodriguez-Castro, B., Gibbins, N., Darlington, J., and Shadbolt, N. R. (2009). Linksb2n: Automatic data integration for the semantic web. In Meersman, R., Dillon, T. S., and Herrero, P., editors, *OTM 2009, Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009, Vilamoura, Portugal, 2009*, volume 5871 of *Lecture Notes in Computer Science*, pages 1121–1138. Springer.

Silva, J., Almeida, J. G., Batista, T., and Cavalcante, E. (2021). Aquedücte: A data integration service for smart cities. WebMedia '21, page 177–180, NY, USA. ACM. DOI: 10.1145/3470482.3479631.

Silva, V., Campos, V., Guedes, T., Camata, J. J., de Oliveira, D., Coutinho, A. L. G. A., Valduriez, P., and Mattoso, M. (2020). Dfanalyzer: Runtime dataflow analysis tool for computational science and engineering applications. *SoftwareX*, 12:100592.

Silva, V., Leite, J., Camata, J. J., de Oliveira, D., Coutinho, A. L. G. A., Valduriez, P., and Mattoso, M. (2017a). Raw data queries during data-intensive parallel workflow execution. *FGCS*, 75:402–422. DOI: 10.1016/j.future.2017.01.016.

Silva, V., Leite, J., Camata, J. J., De Oliveira, D., Coutinho, A. L. G. A., Valduriez, P., and Mattoso, M. (2017b). Raw data queries during data-intensive parallel workflow execution. *FGCS*, 75:402–422.

Syed, A. (2020). The challenge of building effective, enterprise-scale data lakes. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, SIGMOD '20, page 803, New York, NY, USA. Association for Computing Machinery.

Widom, J. (1995). Research problems in data warehousing. In *CIKM'95*, CIKM '95, pages 25–30, New York, NY, USA. ACM.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, EASE '14. ACM.

Zhou, R., Zhang, X., Wang, X., Yang, G., Guizani, N., and Du, X. (2021). Efficient and traceable patient health data search system for hospital management in smart cities. *IEEE Internet Things J.*, 8(8):6425–6436. DOI: 10.1109/JIOT.2020.3028598.