smart cities

MDPI

Article

A Unified Knowledge Model for Managing Smart City/IoT Platform Entities for Multitenant Scenarios

Pierfrancesco Bellini, Daniele Bologna, Paolo Nesi * and Gianni Pantaleo

DISIT Lab, DINFO Department, University of Florence, 50121 Firenze, FI, Italy;
pierfrancesco.bellini@unifi.it (P.B.); daniele.bologna@unifi.it (D.B.); gianni.pantaleo@unifi.it (G.P.)
* Correspondence: paolo.nesi@unifi.it

Highlights:

What are the main findings?

- Early identification of causes for problems and dysfunctions at their inception in large multitenant smart city infrastructures.
- Maintain references to data, processes, and APIs to add/develop new scenarios in the infrastructure, minimizing effort in complex applications.

What are the implications of the main finding?

- Increased scalability of large smart city infrastructures.
- Increment of maintainability of multiple applications that are sharing the same data pool.

Abstract: Smart city/IoT frameworks are becoming more complex for the needs regarding multitenancy, data streams, real-time event-driven processing, data, and visual analytics. The infrastructures also need to support multiple organizations and optimizations in terms of data, processes/services, and tools cross-exploited by multiple applications and developers. In this paper, we addressed these needs to provide platform operators and developers effective models and tools to: (i) identify the causes of problems and dysfunctions at their inception; (ii) identify references to data, processes, and APIs to add/develop new scenarios in the infrastructure, minimizing effort; (iii) monitor resources and the work performed by developers to exploit the complex multi-application platform. To this end, we developed a semantic unified knowledge model, UKM, and a number of tools for its implementation and exploitation. The UKM, with its inferences, allows to browse and extract information from complex relationships among entities. The proposed solution has been designed, implemented, and validated in the context of the open source Snap4City.org platform and applied in many geographical areas with 18 organizations, 40 cities, thousands of operators and developers, and free trials to keep platform complexity under control, as in the interconnected scenarios of the Herit-Data Interreg Project, which is a lighthouse project of the European Commission.

Keywords: knowledge and data engineering tools and techniques; data and knowledge visualization; knowledge management applications; knowledge model; smart city; microservice architectures; formal models; artificial intelligence

Citation: Bellini, P.; Bologna, D.; Nesi, P.; Pantaleo, G. A Unified Knowledge Model for Managing Smart City/IoT Platform Entities for Multitenant Scenarios. *Smart Cities* **2024**, *7*, 2339–2365. https://doi.org/10.3390/smartcities7050092

Academic Editor: Pierluigi Siano

Received: 26 June 2024
Revised: 21 August 2024
Accepted: 24 August 2024
Published: 27 August 2024

1. Introduction

Recently, smart cities are becoming more and more oriented on exploiting technical solutions as fundamental tools for decision makers to: (i) enable real-time control on what is happening in the city; (ii) make short, medium, and long-term forecasts for planning interventions; (iii) perform simulations for studying possible improvements and plans. New trends in smart city technology include the introduction of integrated infrastructure and services [1]. Smart cities are open to citizens to exploit on demand personalized smart services (via mobile apps, specific devices, web apps, etc.), allowing them to become active participants in the improvement and evolution of their city via Living Labs

as in [2]. Modern smart city infrastructures must support numerous data providers and consumers, various data exchange modalities, multiple data transformations, machine learning/artificial intelligence processes, and thus services executed within a single framework. The data may be very complex from all points of view (formats, protocols, subject matters and coverage, open and/or private data by respecting specific licensing and access rules, etc.). These facts increase the demand for controllability of services and solutions in the framework in order to enhance the quality and availability of services for decision makers and final users. To this end, in order to simplify the implementation of processes, service-oriented and microservices approaches are progressively adopted [3,4]. The heterogeneity of data types and models to be managed and their cross-exploitation among services are additional levels of complexity to be addressed. The complexity of data models and types goes beyond the concept of the Digital Twin for which the physical entities are going to have a synchronized counterpart in the digital world [5]. The decision-makers need, in most cases, higher level models and integrated views augmenting the digital representations that can be obtained from the single entity representation. Thus, they are pushing towards the Global Digital Twin concept, in which the integration of local digital twins taking into account the integration aspects is addressed [6]. The different data sources may include IoT/IoE (Internet of Things/Everything) networks and data, open data portals, social media, private and/or public data, GIS (Geographic Information System), census data, BIM (Building Information Modeling), etc. These data and derived data may be needed as those of exploited and/or generated by city utilities, such as industry plants, mobile apps, mobile cellular data, and origin destination matrices [6–8]. Thus, several smart and control applications need to exploit the same cross-domain data and services coexisting on the same smart city infrastructure. This approach allows to improve smart city sustainability by loading data once, distributing management and operative costs, and avoiding the implementation of multiple and redundant solutions replicating data gathering flows, processes, storage, services, dashboards, etc. The coexisting smart services may benefit from accessing/sharing each other: (i) data sources in the big data view (for example, the production of reliable parking predictions could take into account weather forecasting and real-time traffic data [9,10]); (ii) computing processes (for example, solutions for computing heatmaps and servers for distributing them in tiles over maps to the consumers). The smart city framework should provide flexible real-time services to support interconnected processes for data ingestion, storage management, data transformation and management, data analytics, data visualization/interaction services, and also simulations for implementing what-if analysis associated with the Digital Twins, local and global [11,12]. The efficient and sustainable management of an integrated set of services leads to the need to create multitenant smart city infrastructures in which data, processes, and tools may be reused for multiple purposes. This approach implies at the same time the need to manage a wider complexity in development, management, and maintenance. The network of services, a collection of processes and data flows, has to be finally deployed on some distributed architecture, created, exploited, and maintained by different actors and their technical operators.

In addition to the above-mentioned aspects, smart city solutions, including IoT systems and services, have to take into account the interconnections among common data, processes, and users/actors involved in the development and maintenance of smart city/IoT solutions, especially when multiple actors are working on the same framework. An actual smart city may present a large number of data-driven, distributed, complex multiservice solutions. Thus, the work of technical operators is not trivial and may be largely simplified if the smart city infrastructure provides an integrated framework for navigating and controlling data, processes (microservices), and users' activities in the multitenant infrastructure, taking into account all the reciprocal relationships, as described in this paper.

1.1. Paper Aims and Structure

Smart city operators of multitenant smart city infrastructures have to cope with solutions that share the same data and, thus, solutions in which processes/services and tools are cross-exploited by multiple applications and developers. It is quite common that several cities belonging to the same region/area would need to access the same data. For example, for regional mobility infrastructure (they need to know the bus stops in their location but also where the buses are going to arrive), from the regional telecom operator data as touristic and commuter flows represented as origin destination data, from the same weather forecasts and quality of air data, from the same health data (triage hospital conditions), from the regional Lidar data for terrain and land conditions/evolution, etc. These are the typical data collected and managed at regional/area level. In most cases, smart city processes need access to those data to download them and process them in local for their purpose. On the other hand, regional operators may even not be aware of their usage since they are distributed as open data. An integrated multitenant solution would reduce the costs and maintain the data access clear for all, for producers and consumers, reducing duplications, and making the update and distribution more efficient. The adoption of multitenant approach for smart city leads to make possible the usage of "smart city as a service" approach, thus reducing the initial investment and save money in: (a) developing data connectors, and adopting new solutions (since the same solution could be in place in other cities as well), (b) reducing data duplications and collection of new data, (c) reducing the risks for vendor lock-in on sensors and verticals (since the application layer and hardware layers could be separated), (d) sharing costs for maintenance, exploiting high experts for smart city applications, (e) sharing costs in the control room support in which services need to be high available for early warning, (f) sharing costs for planning tools, which are typically quite high, since they are only used sporadically, while with a central solution the costs will be shared among different cities/operators.

Therefore, large infrastructures for smart cities need an integrated framework providing models and tools to (i) identify the causes of problems and dysfunctions that may occur in the smart city infrastructure since their inception, (ii) provide support to the developers operating in the same infrastructure, reusing data and processes/services, minimizing the development costs (e.g., providing references to data, processes/services, and related APIs when they develop new scenarios exploiting the data and processes that are already present, and (iii) monitor the work performed and resource exploited by the developers in a large context in which the reuse of data and processes is the normality.

As an example of point (i), when a problem arises in visualizing some data in a dashboard (graphic user interface), the framework should support the operators to understand in a short time the origin of the problem, be it on data, services, results/services of other users, or a combination. More specifically, the problem(s) could be due to missing data, changes in data message formats arriving, faults in algorithms, communications, operating systems, authentication, data licensing, actions of other users, etc., in multiple services developed by multiple users.

In the case of point (ii), at a given step of a new application development, data sources need to be actually available, services and data results have to be accessible, communications and APIs should be responding, and storage has to be ready to obtainthe requested data and flows. The applications would be developed by accessing the data/resources and computing the results, which could be consequently displayed by some visualization tools.

Both (i) and (ii) aspects must be integrated in the same model and framework to allow managing the full set of processes/data and allowing to navigate among the several relationships they have with each other in a short time. With the aim of (a) reducing the time to identify and solve the problems and (b) developing by exploiting data and processes in place. Point (iii) implies that the framework model for these complex infrastructures has to be scalable and provides support for expanding new applications, data, and tools in connection to those developed by other operators. The possibility of adding new functionalities without reloading data is related to efficiency of development and cost savings.

As described in the related work section, the state-of-the-art frameworks for smart city IoT development do not support developers providing suitable tools for solving the identified problems. As described in the next section on related work, most of them are focused on developing single tenant and vertical applications; others permit the development of multiple applications without supporting the multitenant and without providing an underlined model and framework covering all aspects and relationships among data, processes, and rendering tools.

Therefore, this paper proposes a unified knowledge model, UKM, and framework for semantic reasoning and managing data and processes in a cross-network of microservice applications that can be created in many different IoT/IoE-enabled smart city scenarios, exploiting the same data and tools as in multitenant multiuser smart city infrastructures. To this end, the UKM provides a specific model entity and reasoner to facilitate the retrieval of information for monitoring, troubleshooting, and diagnosing their correct behavior and/or dysfunctions.

The UKM model and tools have been designed to cope with problems emerging from the growing number of interconnected scenarios and the introduction of the Digital Twin approach. UKM allows developers to shorten the development process and the platform operators to the early identification of problems for the production of suggestions to developers and to keep under control the evolving set of applications hosted in the platform. They can be provided by professionals as well as autonomously produced by developers of stakeholders exploiting the infrastructure.

The UKM has never been presented in other papers, despite its implementation and validation in the context of the Snap4City framework (https://www.snap4city.org (accessed on 26 June 2024)). UKM has been developed as a core part of Snap4City, which in turn is grounded on Km4City ontology for data modeling and management (spatial, temporal, and relational aspects among city entities) without addressing the modeling of relationships among processes, tools, dashboards, etc. [13]. Snap4City is applied in different smart cities and areas: Italy (Firenze, Pisa, Livorno, Prato, Lonato del Garda, Modena, Merano, Cuneo, etc.) and in Europe (Antwerp, Santiago De Compostela, Valencia, Pont Du Gard, Occitanie, Dubrovnik, Mostar, Rhodes, Varna, Bisevo, and West Greece, Malta, etc.), and their surrounding geographical area (such as in Italy the regions of Tuscany, Sardinia, and Lombardia, and in Belgium and Finland). The biggest installation of the framework is a multi-tenant solution managing advanced smart city IoT/IoE applications with 18 organizations, 40 cities, and thousands of operators and developers, including an open, free trial and test that increases complexity and the needs of control. The UKM solution proposed in this paper has been developed and validated in the context of multiple interconnected scenarios of Herit-Data (https://herit-data.interreg-med.eu/, accessed on 26 June 2024), which addressed the exploitation of big data for tourism management by means of a set of applications.

The paper is structured as follows. Section 2 reports an analysis of the related work. In Section 3, the main concepts for unifying the structure for data/IoT management in smart cities are reported, also addressing the main problems and requirements. In the same section, two scenarios have been reported/described to put in evidence of complexity when multiple applications insist on the same data and processes. Section 4 describes the Snap4City smart city architecture as a reference for the discussion. In Section 5, the formal model of the UKM and its properties are presented. Section 6 presents a validation of the UKM and tools through the presentation and analysis of a real-case scenario, highlighting of the complexity that can be handled, visual browsing of the semantic model, and semantic queries that can be performed according to the objective of the framework management and fault assessment. Section 7 reports an overview about how UKM supports global assessment and related results applied on the Snap4City.org deploy. Conclusions are drawn in Section 8.

2. Related Work

In the context of the above-described complex environments, integrated supports have to be capable of managing multiple organizations, multiple applications and services, big data storages, multiple flows, and several data sources (internal and external). Therefore, the related work areas are in the context of data ingestion and warehouse, data transformation, semantic modeling, micro-service applications, and development/data lifecycle. These are the main components that presently cannot be uniformly managed by smart city frameworks at the state of the art.

Big data warehouse (BDW) architectures are designed to provide support to decision makers and data analysts to store, manage, classify, retrieve, and explore heterogeneous data. Numerous platforms for the management of sensor networks (IoT/IoE), data storage systems, and infrastructures have been developed and proposed. Most of them are capable of creating single processes for data ingestion and thus are not capable of modeling the relationships from data, processes, and solutions [14]. Specific solutions are applied in specific scenarios of a smart city (e.g., energy on smart buildings [15], road accidents [16], power usages [17]), instead of managing the state of the smart city in its entire complexity. Such platforms are usually realized through open-source tools, also to respect the guidelines of both municipalities and European comities, and therefore they have found wide use of NoSQL (e.g., Hbase, Hadoop, MapReduce, MongoDB, and Cassandra [16,18,19]). In most cases, these solutions are not capable of coping with semantic modeling and relationships among smart city entities being focused on modeling a limited number of vertical applications such as for managing parking, waste, water, energy, etc.

The data ingestion phase is often carried out either mainly through data transformation tools such as Extracting-Transforming and Loading (ETL/ELT) processes [20–22] and/or by the implementation and management of IoT processes collecting data from sensor systems (IoT/IoE) [23]. More difficult is the usage of tools that are capable of managing in a scalable manner data processing in push and pull towards big data stores. Some platforms also propose the use of Apache NiFi [20] to manage data flows and data categorization in the ingestion phases through semantic webs and ontologies. However, semantic modeling aspects are often described as future development [14,15,21], or applied only to some thematic areas not currently in use on large data piers. In order to address the several challenges imposed by the collection, management, and exploitation of huge amounts of heterogeneous data with several processes in smart cities and IoT big data contexts, the definition of well-formalized relationships among processes and data is a crucial aspect [24,25]. To this purpose, semantic modeling of IoT, smart city data, and entities has been proposed by means of specific ontological models, as described in [26,27], among which the W3C SSN (Semantic Sensor Network) and SOSA (Sensor, Observation, Sample, and Actuator) [28], which are a set of ontologies describing sensors, actuators, as well as their observations and actuations. Other examples are the CMTS (Connectivity Management Tool Semantics) ontology [29] and IoTSAS for real-time semantic annotation and interpretation of IoT sensor stream data [30]. Recently, Knowledge Graphs (KG) [31] have attracted the interest of researchers to be proposed as semantic data models in the IoT context [32], in which elements of the graph elements are represented by ontology concepts. Semantic frameworks and ontology are also increasingly adopted in IoT-enabled smart city applications [33,34].

The OneM2M Base ontology [35], the Thing Description (TD) ontology [36], and the NGSI-LD by ETSI (Next Generation Service Interfaces) [37] provide semantic schemas enhancing interoperability among IoT protocols and applications. However, often the description schemas and classification fields are generic (such as in the case of SSN), and thus they cannot be used to annotate data with specific domain knowledge, which is pivotal to model and implement real smart city scenarios, services, and applications [38]. Furthermore, most of the models proposed in literature are not capable of providing the necessary semantic expressiveness and interoperability to cover all the possible smart city

use cases and scenarios. For this reason, works in progress are reported, as in [39], which aims at integrating and aligning some of these resources.

Nevertheless, the semantic interoperability requires the integration and solution of several challenges, such as the lack of a standard vocabulary for IoT and smart city data, the management of syntactical differences in data representation formats (e.g., datetime and geospatial standards, value units, etc.), and the management of missing data, redundancies, and errors [40].

Integration among a number of ontologies has been proposed in Km4City [13] (where the following vocabularies have been used, DCTERMS: metadata Dublin Core Metadata Initiative; FOAF: friends of a friends; Good Relation: entities relationships; iotlite: IOT Vocabulary; OTN: ontology of Transportation Networks; OWL-Time: time reasoning; SAREF Smart Appliances REFerence extension for building devices available at https://saref.etsi.org/saref4bldg/ (accessed on 20 August 2024); Schema.org for people and organizations; SSN: Semantic Sensor Network ontology (see https://www.w3.org/TR/vocab-ssn/, accessed on 20 August 2024); WGS84 Datum of GeoObjects; GTFS, General Transit Feed Specification, and Transmodel, for public transport infrastructures: lines/rides time schedules, real-time records, paths, etc.). In [41], OGC SensorThings API [42], CityGML [43], and IndoorGML [44] have been integrated in the same ontology to cope with IoT aspects and BIM for indoor building representation modeling. In [26,45], an overview of ontology applications in smart cities have been reported, in which it is evident that the semantic modeling has been mainly used for modeling city entities, city challenges, and planning and not the structures of the smart city applications. Km4City performance has been assessed in [46]. A solution to provide high performance for Km4City stores for smart cities has been to perform a horizontal scalability by federating the knowledge bases (RDF stores) of the different smart cities [47].

Many big companies are proposing solutions to make cities smarter, such as IBM [48] on services for citizens, business, transport, communication, water, and energy; governmental, educational, e-health, safety, energy, transport, and utilities; CISCO on people, things, and data [49], etc. On the industry domain, standard IoT architectures such as the ETSI IoT standard [50], the ITUT IoT Reference Model [51], the IoTA Reference Model, Intel's IoT Architecture [52], and IWF, IoT World Forum Architecture [53], and many others are taken into account in [54] to identify the most relevant architecture for IoT. Most of the architectural solutions are based on multitier architectures ranging from 3 to 6 layers [55]. The number of tiers is not relevant for microservice-based solutions and to create/manage in an efficient manner innovative and effective services while exploiting city data and information [1,56]. On the other hand, the work and solution proposed in this paper are agnostic regarding the architecture, proposing a semantic model for reasoning about the relationships among data, processes, rendering, and solutions.

To manage the high variety of devices and smart applications, service-oriented frameworks and microservice architectures have been increasingly adopted in recent IoT-based solutions for smart cities [4,57]. In fact, the microservices architecture involves the development of a large set of loosely coupled services [58], enhancing scalability and availability, facilitating maintenance, and fast isolated testing. These aspects were incorporated with the aim of simplifying the complexity of traditional service-oriented architectures (SOA) [59], more oriented to functions providing also a status in the protocol. Moreover, several data life cycle models have been proposed for smart city big data, IoT; however, they are in most cases limited to specific scenarios and different fields, as reviewed in [60,61]. Efforts in describing a unified model are provided in [60], in which the authors proposed a comprehensive scenario agnostic data life cycle (COSA-DLC) model for big data, describing also a smart city use case in Barcelona. On the other hand, the above-mentioned state-of-the art data life cycle models do not model and neither take into account the complexity of multitenant solutions in terms of the many and multiple cross-relationships among processes, data flows, and users.

Regarding the applications and processes' points of view, ontologies modeling processes and their relationships have been defined in IoT scenarios, such as the Process-aware IIoT Knowledge Graph [62]. The IIoT Knowledge Graph does not address the problems related to managing multiple processes and solutions by multiple users on the same platform, exploiting the same data. Large-scale process data are increasingly addressed, especially associated with industrial big data contexts, leading to the conceptualization and semantic description of Big Process [63]. However, these works have not yet been scaled up on real case studies and do not address the temporal queries related to IoT and time series. From the service perspective, the Service ontology Pattern Language (SOPL) has been presented in [64] and [65], providing the conceptualization of services as a network of interconnected ontology modeling patterns, allowing to build service ontologies in specific domains. On the other hand, SOPL is more focused on the concept of service among generic service providers and target customers (persons and organizations), without addressing entities and relationships related to the IoT and smart data models context. In [66], a lifecycle approach has been proposed to model IoT-enabled smart city services as System-of-Systems (SoS) in order to overcome the vertical development of such systems into locked single domains. To this aim, the Service Lifecycle Management (SLM) concept has been proposed and formalized through a Lifecycle Modeling Language (LML), which unfortunately does not provide an effective implementation and solutions to solve the above-mentioned problems due to the lack of formal management of multiple processes and solutions by multiple users on the same platform, exploiting the same data.

3. Concepts and Requirements Analysis

In the context of advanced multitenant smart city frameworks supporting multiple applications and solutions at the same time, the requirement analysis has to cope with a number of aspects: data ingestion, business logic, data analytics, storage, data rendering, and user interface.

In Figure 1, a conceptual functional architecture is presented that abstracts from the technical multitier solutions mentioned in Section 1. In Figure 1, the most relevant data flows among the main components are highlighted. Each module/block of the figure represents a functional area and not a single tool. For example, the data ingestion includes the set of tools, processes, and services for data ingestion, and the same complexity may be present in the business logic, data analytics, storage, and front-end. The functional model also abstracts from the authentication and authorization aspects and from aspects of security, scalability, etc.

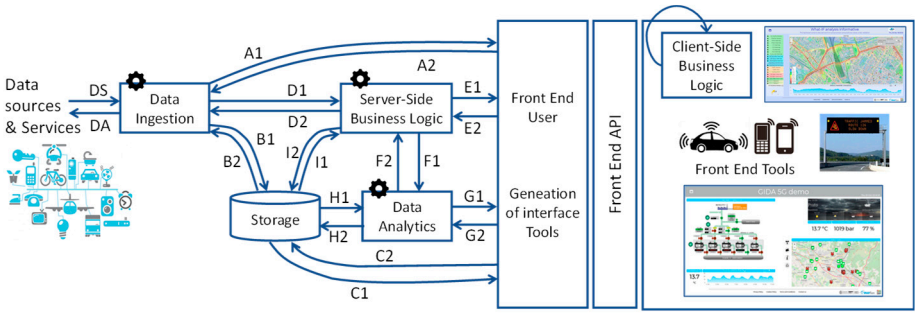

Figure 1. Conceptual architecture, from simple to advanced smart city solutions. Letters are used to identify the different connections, numbers are used to identify the direction of the communication.

The early generations of smart city applications have been implemented by tools for open data management as well as by GIS solutions. In those cases, the data services managed data as Data Sensors (DS) and Data Actuators (DA) to store them in some storage (via B1, which may be indexed and retrieved somehow) and rendered them asynchronously (via C1). Typically, the data ingestion/transformation is performed with the so called ETL/ELT batch processing [20,21]. The arrival of IoT/IoE devices constrained them to

be more reactive (data-driven, managing real-time data) in connecting DS/DA and data services, thus producing the needs of having direct connections with the front-end (as A1), possibly data-driven. In that context, data ingestion tools have passed to be implemented by IoT brokers to manage IoT devices via bidirectional protocols (C1/C2 and B1/B2) [4,27]. At the same time, for acting on the field direction from the user interface (e.g., A2/A1), direct connection has been activated to allow sending data and events from the user interface to IoT brokers and devices as signals/messages in event-driven (for example, with web sockets).

Slightly more complex solutions could be those that need to perform some computing on stored data, which can be performed only asynchronously due to their complexity (e.g., periodic processes). This problem can be solved by activating data analytical processes and services and exploiting connections from them to the storage (H1/H2) to read historical data and write results back (for example for computing some: machine learning, artificial intelligence, simulation, predictions, early warning, anomaly detection, heatmaps, origin destination matrices, optimization, etc.). The results of generic data analytics can be made accessible on the user interface once saved in the storage, also storing eventual scenarios collected from the user interface (via C2/C1), for example, for implementing some simulation and/or what-if analysis. Otherwise, data analytic results can be directly consumed on the fly without saving values (assuming that the results are just a temporary exploration) (e.g., via G1/G2).

Typical smart city applications and services need to compute predictions/simulations on the basis of historical and contextual data. The computation could be performed periodically or on demand. The results are usually saved on storage, and they can be accessed by user interface devices (web and mobile apps, dashboards, digital signages, etc.) via communication and services, such as H1/H2 connections. Examples of smart applications and services that exploit computing tools are smart irrigation (deciding the best irrigation time), smart parking (providing free parking slot predictions), optimizing routing for waste collection, multimodal routing for final users, and computing origin destination matrices, heatmaps, anomaly detections as early warning, etc.

Recently, in the context of smart cities, a number of complex smart city applications are demanded by decision makers for city management and control rooms (with the aim of implementing workflow management, simulation, and what-if analysis). This may imply developing complex workflows in a short time, with frequent updates and modifications of data analytics parameters, data flow, workflow, and business logic behind the functionalities of the smart application user interface. To this aim, most of the smart city solutions integrated the possibility of implementing a business logic module with workflows/dataflows (exploiting E1/2, F1/2, and I1/2 connections), thus including multiple connections for real-time data-driven flows (D1/2, E1/2), as in Figure 1. To this end, most of the development environments are based on using traditional programming languages for their programming (such as AWS, MS Azure IoT, etc.), while in some other cases visual languages and microservices are used (e.g., Node-Red by JS foundation [4]). Thus, the capability of the solution highly depends, in practice, on the flexibility of the business logic/workflow, which in turn exploits API and microservices, and thus on the integration of the user interface to create smart applications with the development environment and data flow. These smart applications can be regarded as Business Intelligence tools. In some cases, they may have some business logic on the client side, directly implemented on the end client, for example, on the web pages loaded on the browser [67].

The implementation of solutions can be modeled and realized by using microservices provided by the functional areas of Figure 1 and taking into account nonfunctional aspects, e.g., [10]. The development framework has to enable developers to create advanced applications and, at the same time, support operators to keep the infrastructure under control, despite the complexity generated by multiple applications that may share the same platform, data, data-driven processes, data analytics, and users' tools at the same time. The city operators do not necessarily know all the details of each specific service/application

that has been developed by different third-party actors/organizations, and thus neither how they have been implemented in terms of data flows, processes, business logic at the user interface, etc.

In the following subsections, two scenarios are provided. The first scenario presents two solutions exploiting the same data process. The second example provides evidence about the complexity of managing and monitor five solutions, while actual smart city platforms may need to run at the same time tens of them for each tenant.

3.1. Scenario 1

The first scenario presents two solutions exploiting the same data processes (see Figure 2). The operator is interested in getting predictions about free slots for parking (exploiting traffic flow predictions and other data) and about traffic flow status. In this case, two different Solutions (1) and (2) are managed to produce the informative dashboard and info-services for the mobile app. Lowercase letters in the figure represent the different steps involved in each Solution.

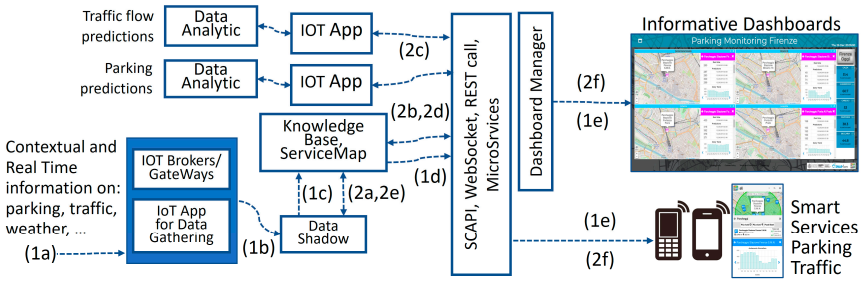

Figure 2. Scenario 1 with two solutions—Smart Parking.

The Scenario 1–Solution #1 represents the process of collecting contextual, historical, and real-time data about traffic flow and parking status; moreover, weather forecast values are also collected from external services (1a). Real-time data can be gathered from different sources (for instance, by IoT devices and processed via the IoT app for data transformation through Node-RED, ETL, and FIWARE IoT agents [68]), and collected in the data-shadow storage (1b). The data are also indexed on the Knowledge Base, KB (1c), with the support of the ServiceMap tool, which allows to perform queries on the KB and to visualize and browse results on the map. Data are also made accessible through the smart city API, SCAPI (1d), and they can be finally visualized on dashboards and mobile apps (1e) as contextual information, such as traffic and parking status. The Scenario 1–Solution #2 exploits data collected by Solution #1 (2a); historical and real-time data are used (2b) by an IoT app and/or by data analytics processes to estimate predictions (2c) (on traffic flow and parking). Parking predictions are more precise by knowing traffic flow and weather forecasts [10,69]. The predictions (2c) are stored via (2d) in the KB index and in storage (2e), and they can be visualized by informative dashboards and mobile apps (2f).

3.2. Scenario 2

This second scenario deals with the production of an interactive application for smart city management, involving what-if analysis and decision support for city administrators and operators. The output of this scenario is characterized by the production of informative and interactive dashboards to interact with smart city devices and to communicate administrative messages in real-time (for example on digital signage panels), as well as to perform simulation and what-if analysis by using dynamic traffic routing. The scenario is illustrated in Figure 3.

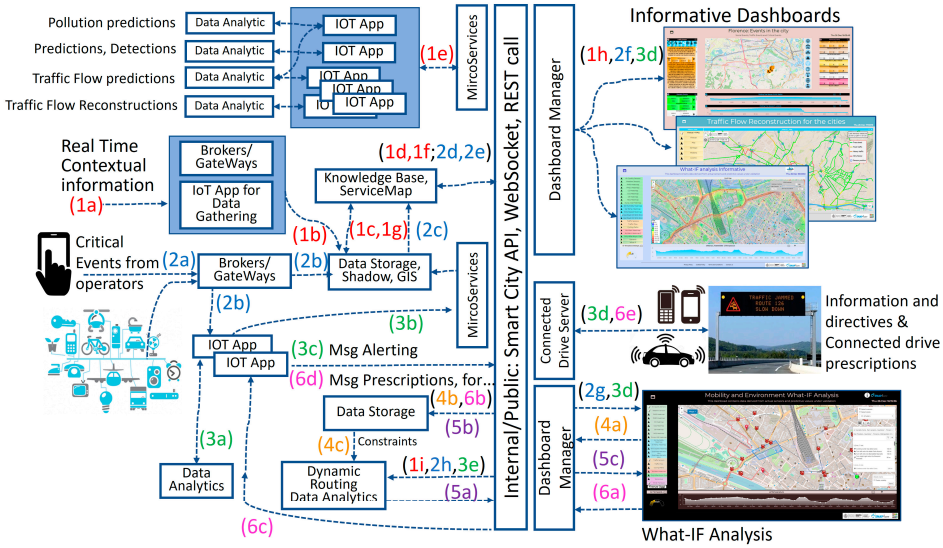

Figure 3. Scenario 2: what-if analysis for dynamic routing scenario. Colors and numbers on labels are used to simply the identification of the subscenarios.

Scenario 2–Solution #1 exploits contextual information as historical georeferenced data about traffic, weather, and air quality (1a) to compute predictions. Collected data are saved in the data storage (1b, 1c), represented by both the Data Shadow and the KB, supported by the ServiceMap tool. The data analytic processes access data via SCAPI (1d) and dedicated microservices (1e), and save them back (1f, 1g). The results are accessible for dashboards and mobile apps (1h) via SCAPI, microservices, and/or web sockets. Scenario 2–Solution #2 collects critical events from on field operators (2a). They are saved in the storage (2b, 2c) and shown in dashboards and mobile apps (2f) via (2d). Actions and changes made on these dashboards and apps are saved back on the storage via (2e). Specific messages (Scenario 2–Solution #3) can then be produced by dedicated data analytic processes for early warning (3a), exposed in push through microservices (3b), and generated by IoT apps (3c) for informative mobile apps, connected drive devices, variable message panels (in case of displaying alerts or traffic detours), telegram, SMS, call, etc. (3d). On Scenario 2–Solution #4, users with authorized privileges (e.g., city officials and operators) can act on the what-if analysis dashboard/tool to add and modify road constraints (4a) and save them as private data (4b). Such data are exploited by a dedicated data analytic process (4c) for dynamic routing in order to suggest alternative traffic paths based on defined constraints. The information involved in Solutions #1, #2, #3, and #4 are taken into account by decision-makers. In this case, the City Operator may create a what-if Scenario 2–Solution #5 via (5a) that takes into account planned events, data on traffic, pollution, as well as critical events occurring in that moment to create a scenario that may change the viability of the city (the road graph and their organization in terms of transit direction, speed limits, etc.). The resulting scenario, obtained by exploiting flows (1i, 2h, 3e, 4c), is saved into data storage as private data (5b); therefore, it can also be shared and used to perform some simulation in the what-if dashboard (5c). The chosen Scenario 2–Solution #6 (6a) can be communicated and consolidated, and it can be also saved in data storage as private data (6b) and may be shared with other operators. Once approved definitively, it can be actuated (6c, 6d) by performing actions, such as closing a specific road, changing the viability, providing connected drive information, sending messages to the mobile app, and to variable message panels (6e).

3.3. Requirements

According to scenarios analyzed in the context of several smart applications and tools into the multitenant large infrastructure of Snap4City, a set of high-level requirements has been identified for the back-office solutions to enable smart city operators to manage the

complexity. The requirements emerged from several requirement analyses performed for each Snap4City deployed solution according to specific workshops with focus groups of operators and stakeholders, followed by document analysis, prototyping, and review in multiple steps. As a result, the most relevant high-level requirements for smart platform management have been identified and state that the platform for developers and operators has to provide support to:

- cope with a large number of data types, exploiting a unified data model and tool, such as: IoT devices (sensors and actuators), POI (Point of Interest), typical time trends, predictions, heatmaps, constrained scenarios, traffic flows, private data, user profiles, maps, orthomaps, shapes, GIS data, trajectories, origin destination maps, BIM, etc.; they need to be indexed according to their semantic relationships (spatial, geographical, and temporal) to facilitate the search and extraction of new knowledge for smart applications;
- compose highly dynamic smart city solutions, from simple (data rendering of info and statistic data) to complex (early warning, predictions, what-if analysis, and simulations) without limiting the flexibility by relegating logic in coding;
- control and manage the status of the processes and data flows using the same tool and interface (the high number of developers on several tenants and the presence of free trials increase the need for platform control and robustness). This means that it has to be easy to pass: (i) from the data description and values to the processes that have been involved in their production, ingestion, and/or exploitation, and (ii) from a process, dashboard, application, or data analytics to the specific data exploited and/or produced, while the API identification is too generic access.

The above-listed requirements imply features to be provided by the collaborative development environment for smart cities and other domains. In addition, in particular, during the solution life cycle (development, maintenance, exploitation/extension, and change). They are independent of the kind of architecture adopted. This approach would allow the operators to optimize the services and (cons)train the developers to exploit data and services in place rather than redeveloping from zero each solution. These requirements motivated the research and the modeling of the solution described in this paper. Thus, reducing the development time reducies the effort needed to (i) identify and exploit the patterns developed by other developers and to (ii) identify the relationships to solve the problems. Thus, making it easy for the operators to manage large-scale platforms in which multiple solutions and developers.

4. Snap4City Architecture

The above-discussed requirements have been taken into account in extending the open source Snap4City platform (also strictly based on open-source tools and libraries) with the adoption of the UKM and tools described in this paper. This section provides an overview of the Snap4City platform as reported in Figure 4, for the minimum useful to understand the formal modeling and solutions presented in this paper to cope with the problems identified above.

Snap4City supports on the same platform different geoareas, cities, topics, and operators in tenants (which are called organizations). Each of them may have multiple users, from developers to decision makers, and end-users, for example in smart parking [10]. Snap4City manages heterogeneous data sources coming from: external services, in pull/push modalities, open data, data providers, IoT networks, protocols, formats, etc. All the data can be processed in real-time and indexed in OpenSearch. In addition, data are semantically indexed on a Knowledge Base (implemented as a triple store, RDF, Resource Description Language) according to Km4City ontology for data management geographically and temporally [13] and augmented by UKM vocabulary for the platform management as described in this paper. Collected and indexed data can be queried using SCAPI and microservices, which are exploited by data analytics processes, simulations, forecasts, deductions, etc., to

produce new knowledge and hints that can be provided directly on the user interface as well as stored for further analysis.

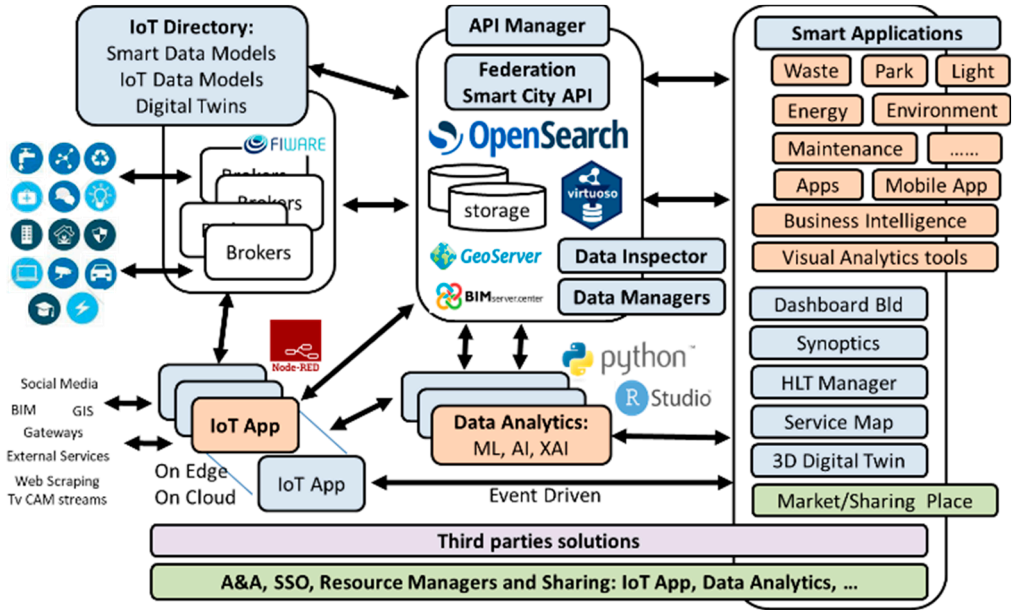

Figure 4. Snap4City platform architecture. The orange blocks are those typically developed to add new functionalities, applications and/or services to the platform.

IoT apps implement processes for business logic, data transformation/integration, data analysis management, data-driven flows, and workflows mentioned above with the Node-RED visual programming language, thanks to more than 180 smart city framework microservices of Snap4City [4]. IoT apps can be executed on premises, on the edge and on the cloud. Historical and real-time data, as well as the new knowledge extracted from data, are made available for the front-end, data analytics, and IoT apps via SCAPI [56] and microservices. The SCAPIs are exploited by dashboards, what-if analysis, real-time monitoring, and mobile apps. In the smart city context, the data ingestion area needs very articulated solutions due to the complexity of managing several kinds of heterogeneous data sources and providers. The Snap4City data gathering process is realized to pose the bases to produce new knowledge and speed up the decision processes [70]. Data connections are typically bidirectional to manage static, quasi-static, and real-time information/data, and acting them in pull/push, data-driven. Static data may include maps, POI, etc. Real-time data may arrive/be sent also from/to web and mobile apps, dashboards containing actuator widgets such as switches, dimers, buttons, etc. The Snap4City interoperability is described in [4] and in https://www.snap4city.org/65, accessed on 20 August 2024.

In Snap4City, real-time data entities have to be registered into the IoT Directory and Broker. The registration automatically inserts triples into the KB for semantic indexing, including establishing relationships with related city entities. The reference element from storage and KB is the entity ID called ServiceURI (SURI).

5. Unified Knowledge Model

This section presents the UKM for representing applications, processes, operators, and data flows in the framework. The method to design the UKM is enforced in the paper. In the sense that we analyzed the state of the art looking for a solution to our identified requirements. The requirements are reported in Section 3. Then, a number of scenarios have been formalized, as those reported in Section 3 (just as the most representative). As a successive step, we analyzed the Snap4City platform and architecture to understand technical limitations and relationships among the actually involved entities, also observing the list of problems detected by the developers and managers in the past years. This al-

lowed us to classify them as related to applications, processes, operators, and data flows. These concepts have been modeled into an ontology, which has been validated by using a set of queries with inference to be sure to have modeled a solution that could be actually used for producing/inferring the answers to our questions.

In the following subsections, we formalize the entities and relationships described in the architecture by means of an ontological model and tools satisfying the requirements identified in Section 3.3. The ontological model has been used for reasoning about the relationships among the entities involved in the data, process, and application management, as described in the following. The UKM is the main tool to perform reasoning in detecting problems in the complex structure of several applications, processes, data, dashboards, etc., which could be in place at the same time in large smart city infrastructure. The ownerships and the delegations provided from the entities' owners are not reported into the ontology to remain GDPR compliant (General Data Protection Regulation) [71]. On the other hand, this information is accessible for the administrator and for the users according to the granted accesses they have, and directly from each entity management. The LOG (linked open graph) tool provides direct access to those user interfaces [72].

The ontology at the basis of the UKM has been developed using an iterative methodology that starts with a set of competency questions that the ontology should help to answer. The most relevant competency questions defined were:

- "if an IoT sensor fails, which are the dashboards and applications impacted"
- "which are the most critical IoTApps whose stop would impact more than N dashboards"
- "which are the most complex solutions providing at least M dashboards connected each other"
- "which are the most requested data sources used from widgets on the basis of dashboards visualization request count".

The UKM ontology is new, and it exploits only basic vocabularies such as FOAF, DCterms, and Km4City (Snap4City knowledge model for the data). ontology has been tested, checking for inconsistencies of classes and instances using OWL reasoners, and it has also been verified that the inferred facts made sense. Moreover, the UKM ontology has been validated by using OOPS! (OntOlogy Pitfall Scanner!, http://oops.linkeddata.es/, accessed on 10 July 2024) to search for common problems in the ontology definition [73]. The UKM model and RDF store have been kept separate from the KM4City ontology, as UKM is used to describe the tools and mechanisms to produce and access data described using the Km4City ontology; however, for some classes, there have been defined equivalence relations (e.g., s4c:Sensor ≡ km4c:IotSensor). Moreover, during the development of UKM, no other openly available ontology has been found to be reused.

5.1. UKM: Classes and Object Properties

A graphical interactive representation of the UKM classes and object properties is shown in Figure 5 to give an idea of the actual complexity managed by a smart city multiorganization platform such as Snap4City. The graph rendering is based on the Linked Open Graph (LOG) tool for browsing on triple stores and knowledge bases [72] representation. The LOG is an integrated tool of Snap4City.

In the UKM, the most general class is named *s4cThing*, which represents any Snap4City thing. The UKM base URI for classes and properties is http://model.snap4city.org/s4c/ (accessed on 20 August 2024) while for instances it is http://model.snap4city.org/resource/ (accessed on 20 August 2024). The ontology and the instances are published as linked data, thus the url, when opened in a web browser, shows a human-readable representation of the entity. In the following (as well as in Figure 5), we will refer to entity names, omitting the base URI. The *s4cThing* class is the parent class of: the *ProcessingThing* class that represents any processing element; the *Data* class that represents the data managed by the platform; and the *AdministrativeThing* class representing any element used to administer *Data* and *ProcessingThings* such as users, groups of data elements, groups of users, or organizations.

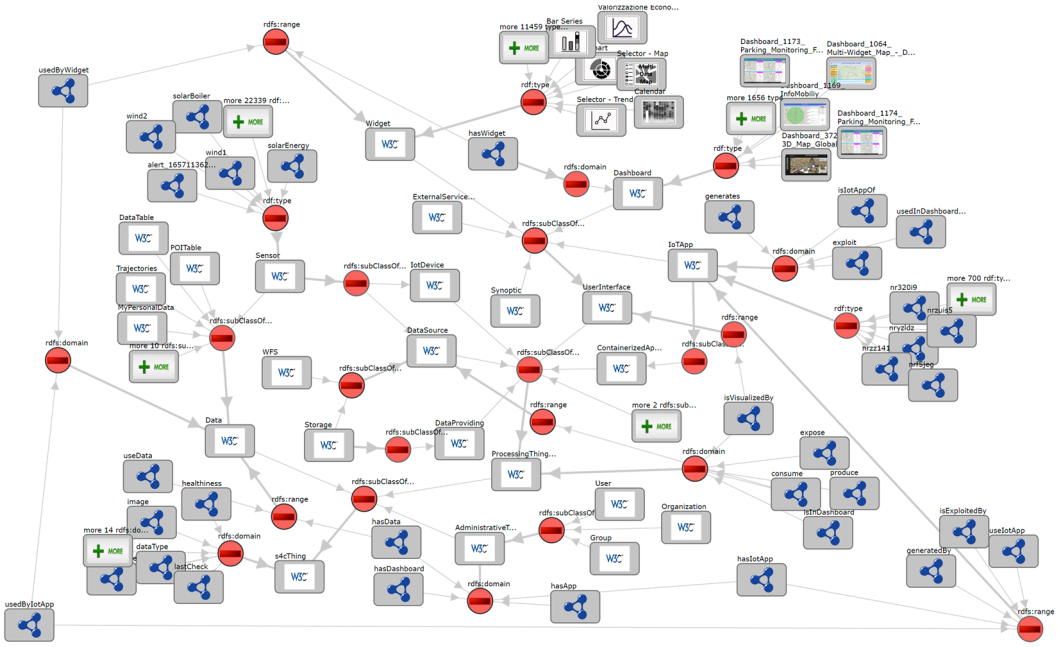

Figure 5. Knowledge graph representation of the UKM, exploiting the Linked Open Graph (LOG) tool. The graph is showing UKM classes, relations, data properties, and also an overview of instances for dashboards, widgets, and Data used. https://www.snap4city.org/ldgraph/?graph=7298d9910a65a76cfea328b5be7c9918, accessed on 20 August 2024. The circles represent the relationships in the triples, while the rounded box are Objects or Subjects in the triples.

The *ProcessingThing* class is specialized in a number of subclasses: *Containerizedapp* class that represents any application deployed dynamically on the platform, it is further specialized in *DataAnalytic* apps (based on Python or R scripts), *Iotapp* (node-red based apps), and *PortiaCrawler* app (for data crawling from web pages). Other data processing elements are *DataSource* that provides data into the platform and *DataProviding* entities that provide data outside of the platform. The *storage* class is defined as a child of both *DataSource* and *DataProviding*. In the *ProcessingThing* are also included the IotDevices and the *IotBrokers* that are used to manage them, as well as any element with a *UserInterface* as *dashboards*, *widgets*, *IotApps*, *Synoptics*, and *ExternalServices*.

The *Data* class is used to represent the data elements managed by the platform that can be owned by platform users and whose values can be visualized in dashboard widgets. This data element can be: sensors, heatmaps, KPIs, point-of-interests (POIs), traffic flow maps, and many others.

The *AdministrativeThing* class is specialized in the *User* class that represents any snap4city platform user, the *Organization* class that represents a group of users belonging to the same organization, the *UserGroup* class that represents a group of users of the organization, and the *Group* class is used to group data elements to facilitate administration.

Many object properties have been defined, allowing to state relations between the *s4cThing* elements. For example, the values shown in a widget of a dashboard can be generated using an *IoTapp;* the *generatedBy* property is used to state that widget values are generated by an Iotapp; moreover, the *hasWidget* property states that a dashboard has a specific widget. The *useIotapp* property is used to state that a dashboard uses the services of an Iotapp, and it is defined as a super property of the property chain "*hasWidget* o *generatedBy*" which allows to infer that a dashboard depends on an IotApp. The general transitive property *dependsOn* is the parent property of *useIotapp* and other properties such as *consume*, *exploit*, *isRegisteredTo*, and *useData*. Using the *dependsOn* property, we are able to infer general direct and indirect dependency relations. For example, in the case of a dashboard with a widget whose value is produced by an Iotapp using a DataAnalytic

application leveraging some specific data elements, the dependency of the dashboard from these data elements can be inferred.

Another case is the *hasIotapp* property used to state that an *AdministrativeThing* owns an application, while the property *hasUser* is used to state that an *Organization* has a specific user. Defining the *hasIotapp* as a super property of the property chain "*hasUser* o *hasIotApp*" can be used to infer that any user belonging to an organization lends all user applications to the organization. Please consider that, in this specific case, the triples asserting the relations with users are not publicly available for GDPR constraints, and the inferred relation among the organization and the IoT apps has not this constraint.

Other properties such as *consume* describes the action of consuming/reading data by a process (the domain is represented by the *ProcessingThing* class, while the range by the *DataSource* class); *produce* describes the action of producing/writing data by a process on/to a device, application, or storage (the range is *DataProviding* class); *useIotapp* property models the usage of IoT apps by dashboards; *exploit* property models in the IoT app controlling data analytical processes and services (in terms of time scheduling, reusing analytic results for other tasks, conditional logic, etc.); *generates* represents the action of creating widgets by IoT apps directly on dashboards; *register* property regards the registration of IoT devices on IoT brokers; *expose* property models of the microservices, which are used by many different tasks (visualization, internal use as custom logic for IoT applications, API management, process scheduling, etc.); the *hasWidget* property is used to describe the different widgets contained in a certain dashboard; *useData* refers to the specific data source used by a certain widget.

5.2. UKM: Data Model Properties

A data property schema has been defined in the UKM ontology, as *dataModel*, for describing flows among different processes and services of a Digital Twin smart application. The *dataModel* is organized through the following properties: Bulleted lists look like this:

- *semantic* is parent of *nature* and *subnature* subproperties to classify data according to the taxonomy of Km4City ontology, which includes 20 categories for nature (e.g., environment, mobility, healthcare, cultural activity, tourism etc.) and more than 500 categories for subnature (e.g., bus stop, car park, traffic sensor etc., as subsets of the mobility nature).
- *technical* includes subproperties such as: *valueName* (the name of data); *valueType* (which describes the type of measured/provided value, such as temperature, humidity, speed, vehicle flow, etc.); *valueUnit* (reporting the unit of measurement, depending on the selected value type, e.g.,: "°C" for temperature, "m/s" or "Km/h" for speed, etc.); *dataType* (the technical data format, e.g., integer, float, boolean, datetime, json), etc. Please note that valueType, valueUnits, and dataTypes are taken from a dictionary of terms. Moreover, it is also possible to state that a device and its values are *certified*, using, for example, a blockchain.
- *realTime* tags the last valid data values (*lastValue* subproperty) and the date and time at which it has been obtained (*lastDate* subproperty).
- *dataStatus* describes data healthiness via the *healthiness* subproperty, according to a set of defined rules and conditions and a set of subproperties.
- *administration* includes the *organization* and *ownership* subproperties to describe the specific organization to which the data belongs and users' ownership.

Whenever a problem with data is reported, the administrator has to be informed. To this end, he/she needs to verify the problem, recover where the data has been produced, and inspect all possible back-end processes and workflows that acquire, produce, transform, or visualize such data or entity and which may have caused the unexpected behavior. To this end, several relationships have to be established among data, processes, visualization, users, and applications. In particular: Bulleted lists look like this:

- *process* data property, giving information about all the process metadata (e.g., the process/application name, the name of the job or script), as well as details about the

data source (IP, data provider responsible, provider server's name, storage modality). Giving evidence of data sources: IoT Device, IoT Broker, KB, SCAPI. A link attribute shows if data has one or more connections and links with other resources.

- *licensing* data property, providing licensing information and details about the data owner.

In the UKM, the above-mentioned attributes are common for all High-Level Types, HLT, to manage data in a uniform manner by the UKM and by the dashboard rendering tools. The HLT model describes how different data types are modeled and processed: ingested, produced, etc. In the following, the most relevant HLTs are listed and described in detail according to their relevance in the platform.

The Sensor and Actuators HLT corresponds to device data (from some broker in pull or push). Their reliability and update may depend on the ingestion approach, on tools, and by third-party tools, etc. Virtual devices may be generated from IoT apps, user actions on dashboards (for example via widgets: switch, keypad), Web Scraping processes, data analytics, databases, files, etc. KPI (Key Performance Indicator) can be geolocalized, KPI and time series, for example, for personal tracking via GPS coordinates. POI are used to model geolocalized data not associated with a time series of values, while they have a static GPS location. The approach of modeling POI, KPI, events, devices, sensors, and actuators with the same data model concept is adopted in FIWARE with the Smart Data Models [74] and in Snap4City [75].

In snap4City, the HLT concept is extended to many data types, such as GIS data, BIM (Building Information Modeling) models, traffic flows, External Services, etc., which are mandatory smart city entities that can be produced, processed, and visualized according to the UKM. Most of these complex data are produced by data analytics and are unsuitable to be managed by brokers. GIS data may be WMS (Web Map Service), and WFS (Web Feature Service) data may refer to direct links to end points of GIS WFS services providing a complex JSON including GIS data. This kind of data may include Orthomaps (background maps), heatmap matrices, origin destination matrices, etc. BIM models represent 3D models of buildings. Traffic flows are data representing traffic flow for each road element addressed, changing over time.

6. Exploiting UKM in Scenario 2 Analysis

In this section, we are presenting how the UKM can support users in managing processes, troubleshooting applications based on data flows, and creating new processes in the platform. These aspects are relevant for the development of simple and complex solutions and for immediate identification of problems occurring in the complex platform at run time. The implementation of UKM provides support for direct identification and inspection of related elements by managing all the relationships among the processes, tools, and heterogeneous data of the smart city to support the infrastructure management. All the development activities create instances into the global UKM model in the platform. Thus, any development view (data view, process, and dashboard) can be easily accessed by using corresponding online development tools. The developers can access the underlined UKM model of their own artifacts by starting from the dashboards and from the data instances via Data Inspector. Other accesses can be established as well; on the other hand, they are the most natural entry points since when a problem arises, it is typically manifested on rendering tools (i.e., dashboards) or on some alarm provided on data stream healthiness (i.e., Data Inspector).

6.1. Accessing to the UKM Model via Dashboards

In the following, the scenario proposed and illustrated is an instance of the UKM graph of Figure 5 and corresponds to the interactive control room for simulation and what-if analysis in mobility and environment scenarios described in Section 2.

The UKM of the proposed Scenario 2 is illustrated in Figure 6. The main entry is an informative dashboard for citizens and city operators called "Mobility and Environment what-if Analysis" (https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MjE5MA== (accessed on 20 August 2024)). It contains several widgets showing historical data, real-time data, and predictions from traffic, air quality, and weather sensors on IoT devices. Moreover, these devices are visualized on a map supporting interactive what-if analysis, provided by a dedicated Data Analytic process in the back end. The what-if analysis allows to calculate and show on map alternative routings (by different means, such as car, public transportation, bike, and pedestrian), given in input single or multiply connected blocked areas, defined graphically by the user, which may be due to traffic restrictions for exceeding pollutant thresholds, road works, etc.

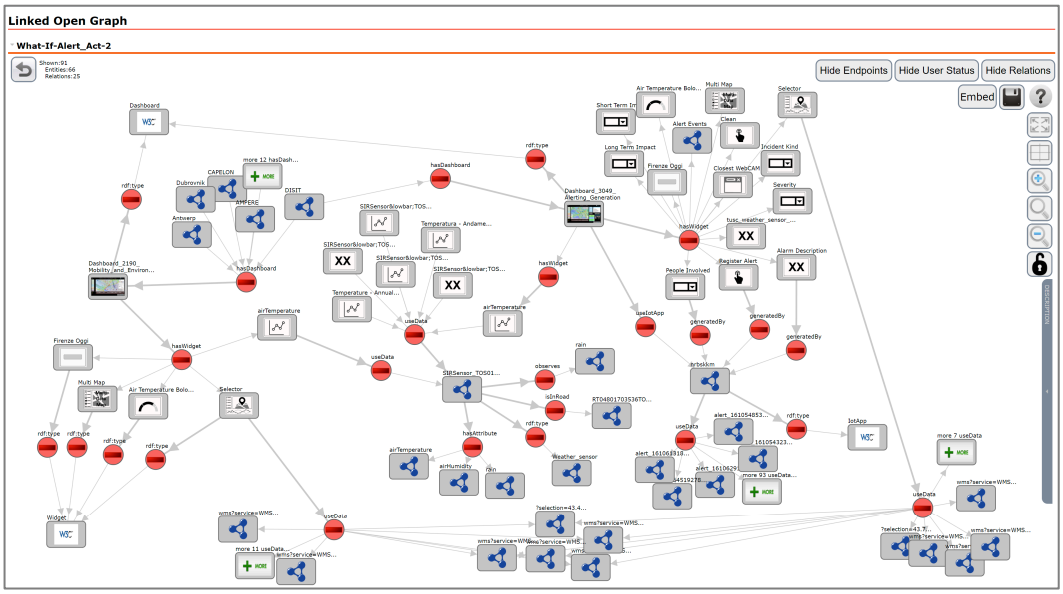

Figure 6. UKM of Figure 3 Scenario 2—interactive control room for simulation and what-if analysis in mobility and environment. https://www.snap4city.org/ldgraph/?graph=624a69652d90a451ef4347788dc50487 (accessed on 20 August 2024). The circles represent the relationships in the triples, while the rounded box are Objects or Subjects in the triples.

Another application is based on the dashboard, called "Alerting Generation" (https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MzA0OQ==, accessed on 20 August 2024) which contains a map sharing many of the data used in the previous dashboard (as it can be noticed on Figure 6, considering the data shared as objects of the useData property of the "Selector" widgets contained in the two dashboards). The latter dashboard allows city operators to define new alerts (on the basis of what-if analysis and data related to traffic, air quality, and weather) and register alerts by means of an actuator (a button widget), which triggers a dedicated IoT app (the node with ID "nrbskkm" and *rdf:type* "IotApp", as shown in Figure 6) to store and dynamically show alerts on the map, in an event-driven way. The graph representation of the UKM can be viewed by means of the Structure tab of the dashboard management panel, which is available for each user's dashboard.

6.2. Accessing the UKM Model via Data Inspector

The Data Inspector, DI, allows to navigate among the relationships of the UKM (data, processes, users). With the DI, the operators and developers may know the status of their data, analyze, understand, and resolve eventual problems; contact the service/data provider in case of its interruption; and exploit the data that is ingested correctly and with continuity (in the case of dynamic data) on dashboards (complying with data ownership

and delegation mechanisms in agreement with GDPR [71]). The DI gives users the ability to monitor data and connected processes: IoT app, data analytics, ETL, storage, and flows up to their final exploitation and/or visualization on dashboards. The DI offers an easy data navigation experience, providing a graphical interface with faceted filters (including full text search and geographical filtering on maps), and suggestions for the most suitable graphical widgets to be used for visualization on the dashboard. The DI tool models many different data features, such as HLT, nature, subnature, value type, value name, etc.

The other relations and connections among data properties and related processes, applications, devices, and services are made explicit and can be navigated through some advanced views of the DI tool. For example, in Figure 7, a view of the DI in the Snap4City front-end is shown. The user can search and inspect its own KPI by clicking on the corresponding row. A panel is opened showing details on the data source (Figure 7A), with attributes and values and information regarding the healthiness, licensing, and ownership. Moreover, it is possible to navigate and visualize the relations with other entities that are related to the specific data by clicking on the buttons of the panel. In this case, it is possible to navigate to the KPI Editor to view/edit (see Figure 7B), as well as navigating and visualizing the IoT apps in which the specific KPI is produced/consumed (Figure 7C). Navigation in the DI can be performed in many different ways and perspectives according to UKM and HLT. Different tools, processes, data, devices, or any other smart city entity become accessible.

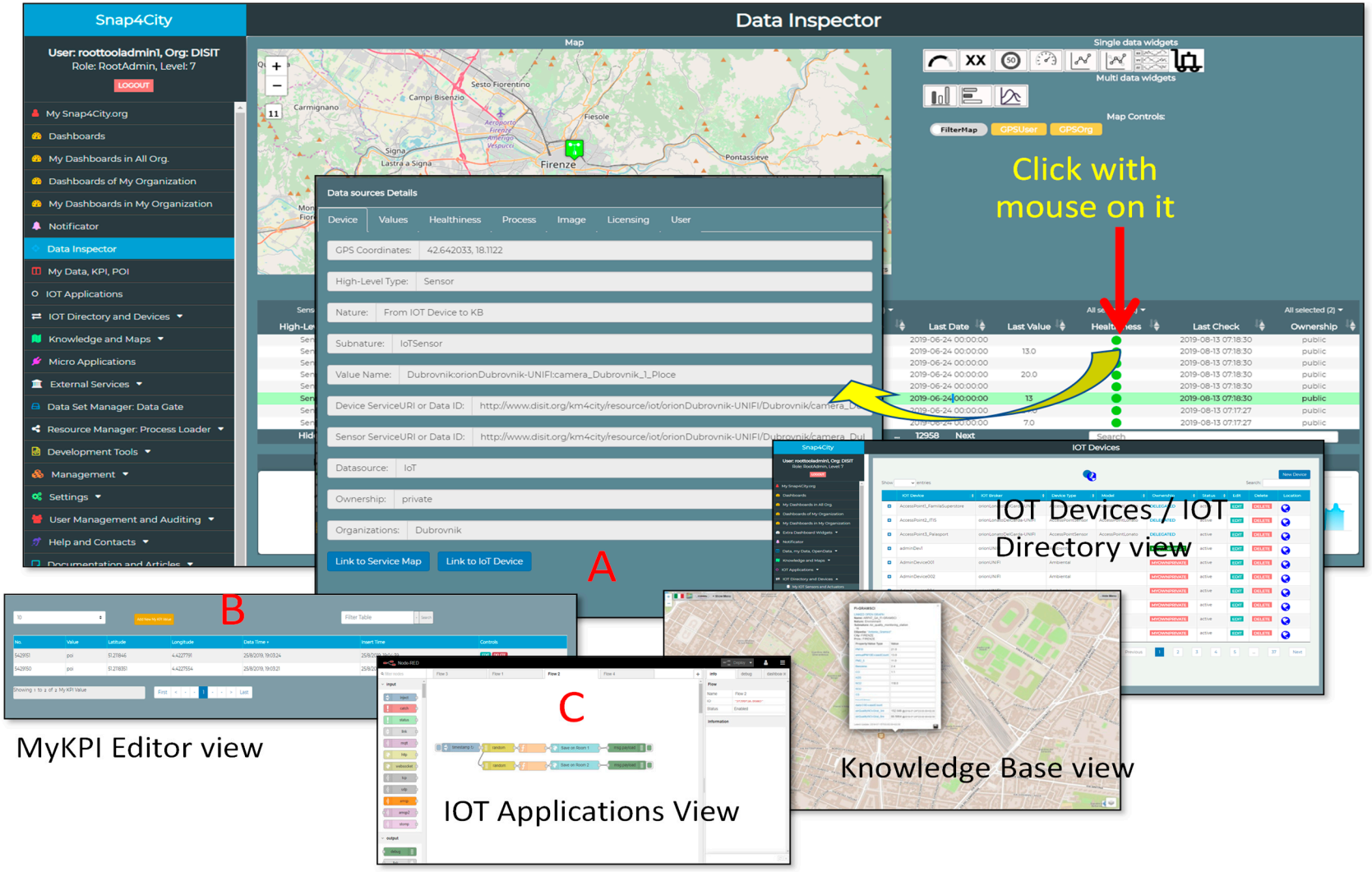

Figure 7. Snap4City Data Inspector with advanced views to navigate data properties, entities, and relations defined in the UKM. The explanation of the letters is reported in the text.

6.3. Effect of Faults on IoT Applications

In the presence of complex solutions based on dashboards, IoT apps and data streams, as presented in previous sections, a fault at the level of the IoT app would impact multiple solutions. Thus, the exploitation of UKM and its visual representation are used for inspection and diagnostic. For instance, to check what happens when an IoT app faults, data sources or other instances are represented in the scenario (in order, for example, to simulate a process of ordinary or extraordinary maintenance, testing, etc.). Thus, it can be checked whose processes and applications would be affected by dysfunctions if the IoT app named "nrbskkm" would fault. The operator can achieve this goal by navigating among connected instances or obtaining a list of them. The outcome of this operation is given by the following SPARQL query on the publicly accessible SPARQL endpoint (https://www.snap4city.org/s4c-query/sparql, accessed on 20 August 2024):

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema/#>
PREFIX s4c: <http://model.snap4city.org/s4c/>
PREFIX s4cNR: <https://iot-app.snap4city.org/nodered/>
SELECT * WHERE {
  { ?s s4c:useIotapp s4cNR:nrbskkm. ?s a ?c. } UNION
  { s4cNR:nrbskkm s4c:useData ?o.   ?o a ?c. } UNION
  { s4cNR:nrbskkm s4c:generates ?o. ?o a ?c. }
} ORDER BY ASC(?o)
```

7. Exploiting UKM for Global Analysis

To highlight the complexity and heterogeneity of use cases that can be created and managed by exploiting the Snap4City platform with the support of the UKM, in this section we provide some quantitative assessments of the number of real applications, devices, and visual tools generated and managed by the platform.

At the time of writing the present paper, the Snap4City.org platform is in its largest deployment on production of the Snap4City framework, and it is based on 48 Virtual Machines on the cloud. The infrastructure manages 20 organizations, with about 7500 users 2500 developers. It hosts hundreds of different complex scenarios with a network of 1638 dashboards (among which 525 are connected event-driven to IoT applications), for a total of 10,940 active widgets, including synoptics. Many dashboards exploit the same shared data and ingestion processes implemented as IoT apps, and the same data analytics. In most cases, dashboards are connected to each other via menus and hyperlinks in addition to the other relationships. The platform provides support for:

- an average of 1.8 million of complex new entering data messages per day, from 260,761 distinct data sources, of which: 23,134 from IoT devices, for a total of 125,086 among sensors and actuators (registered on 19 IoT Brokers); 361 heatmaps with time series; 23 traffic flows with time series; and origin destination matrices with time series; 180 registered External Services;
- Processes exploiting hundreds of microservices, and in particular: 415 IoT apps on containers; 82 data analytics on containers and dedicated servers; 8 Web Scraping processes on containers.

The above numbers provide evidence of the usage of the solution in large-scale conditions and thus on the capacity of scalability of the UKM model proposed.

Some of the processes and dashboards may be more critical with respect to others since they address the monitoring and what-if analysis of the system critical infrastructures, such as in the order: energy distribution, mobility and transport, health and hospitals, food distribution, etc. Therefore, for those processes, a deeper and continuous analysis of consistency and completeness should be performed to guarantee the high level of operative conditions [76]. To this end, the UKM provides a model entity and reasoner to facilitate the retrieval of information for monitoring, troubleshooting, and diagnosing their correct behavior and/or dysfunctions. For instance, dashboards' usage in terms of minutes of

usage and response time, the data usage (number of distinct users' access), the timestamp about the last usage of data from the IoT app, logs and number of faults on the IoT app, etc. The UKM enables to perform SPARQL queries for the systematic global analysis of the systems and for the analysis of single applications as networks of dashboards, IoT apps, and data. In the following, some examples of possible analyses are reported to assess and control different platform proper ties at the global and single application level.

Most of the answers to the issues described in Section 1.1 related to the discovery of causes of faults can be directly obtained by browsing the LOG, as in Figure 6. On the other hand, more complex analyses on the system status exploit the inference described in Section 5.1 and related examples. In the following, examples are provided from the most complex to the simpler; the latter could be implemented on traditional models.

Cohesion and exploitation among solutions and/or organizations. There are some solutions that use the same data as others. To understand/learn the:

- cohesion between/among different solutions;
- exploitation of data needed to create a certain solution;
- impact of changing a data set on the solution in place
- impact of disappearing data for licensing time out.

Complexity assessment of the solution/dashboard is defined as an index combining the number of widgets, the number of data sources used, and the number of data produced. Thus, identifying the:

- most complex dashboards, assessed in terms of elements and related connections to each other;
- most complex solutions based on a set of connected dashboards;

Global usage to identify:

- dashboards with faults in some or specific connection;
- empty dashboards;
- dashboards not used, public and/or private, since XX days;
- most used data and dashboards;
- most crucial IoT application, exploited by many solutions;
- average number of data sources per dashboard.

Count and identification of dashboards, which are:

- using a certain data or data set;
- using results produced by a given IoT app;
- exploiting C1 connection, so called passive dashboards;
- creating data via C2 or A2, i.e., acting dashboards;
- acting on business logic, E1, E2 via some IoT app.

For example, it is easy to identify and monitor all the dashboards, widgets, and IoT apps using data from a specific device or sensor (for instance, a metropolitan traffic sensor in the city of Florence having ID "METRO11") with the following SPARQL query:

```
PREFIX disit: <http://www.disit.org/km4city/resource/>
PREFIX s4c: <http://model.snap4city.org/s4c/>
SELECT * WHERE {
  {
    ?d a s4c:Dashboard.
    ?d s4c:hasWidget ?w.
    ?w s4c:useData <http://www.disit.org/km4city/resource/METRO11>.
  } UNION {
    ?app a s4c:IotApp.
   ?app s4c:useData <http://www.disit.org/km4city/resource/iot/orionUNIFI/METRO11>.
  }
}
```

The SPARQL query and results are shown, as they appear in the RDF endpoint in Figure 8.

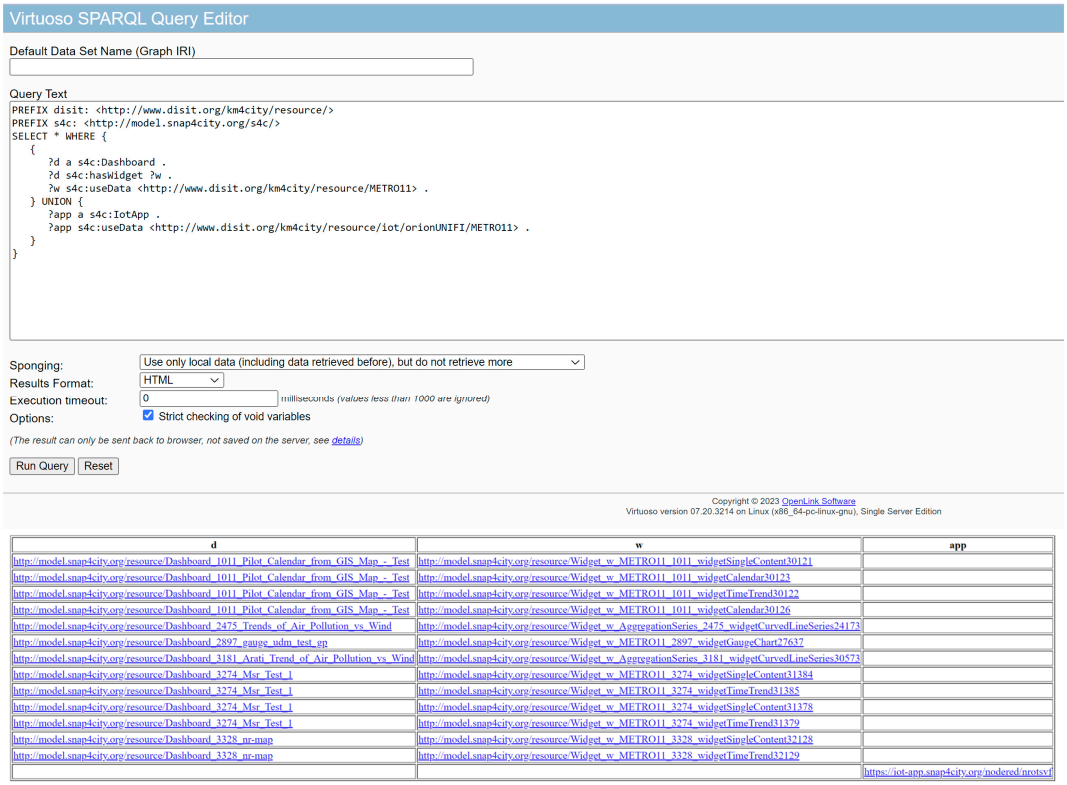

Figure 8. SPARQL query and results showing all the dashboards, widgets, and IoT apps using data from the traffic sensor with ID "METRO11".

Some examples of useful statistics, which can be derived by querying the UKM, are shown in Table 1. In the left column of this table, the query type is reported, while on the right column some numerical insights are provided.

Table 1. Statistics derived by exploiting UKM.

Query Type	Statistics
Empty dashboards	Of the 1638 (among which 525 active) dashboards, 100 are empty (6.1%).
Most Used dashboards	80% of the total amount of dashboard access time is taken by the 21 most used dashboards.
Less Used dashboards	Of the 1638 (among which 525 dashboards are event-driven from the IoT app), only 50 have an access time of 0 min (3%).
Most Used Data	Of the 260,761 distinct data sources, 41 are used at least in 20 different contexts/use cases.
Organization Cohesion for IoT App	137 IoT apps produce data for 2 different organizations.

8. Conclusions

The work reported in this paper addressed the problem of modeling and managing complex relationships in a framework in which data, processes of different kinds (data processing, data analytics, business logic, dashboards, etc.) and developers are involved and in which they produce cross-connected solutions for data processes and views. Recognizing that smart city infrastructures often involve multiple organizations, the UKM aimed to

create a framework that could support collaboration across different entities, ensuring that data and services are accessible and usable by various stakeholders. The addressed problems are those encountered by operators in multitenant smart city infrastructures to (i) identify the causes of problems and dysfunctions at their inception, (ii) identify references to data, processes, and APIs to add/develop new scenarios in the infrastructure, minimizing costs and effort, and (iii) assess the usage of resources.

The solution proposed is grounded on a unified knowledge model, UKM, and a set of specific tools for visual browsing and semantic queries. The UKM semantic model allows to perform complex queries exploiting inference and reasoning on large-scale platforms. Queries can be performed via a visual tool for navigation; the application structures represent visually all the entities and relationships; some of them are offered ready to be used for monitoring the status; others can be performed by using a SPARQL interface. The infrastructure for which it has been developed refers to the smart city and IoT domains in which the need for such a large online platform is growing. The results achieved demonstrated that the UKM and tools allow to perform a direct visual inspection of the model to understand and detect local aspects as dysfunctions, making this feature accessible to all developers on large platforms. In addition, the operators may have a direct tool to monitor the whole framework, exploiting a panel with a set of semantic queries that are capable of extracting on demand and periodically the healthiness of the solutions in place and of the whole framework. The proposed solution has been designed, implemented, and validated in the context of the open source Snap4City.org platform and framework and has been applied in different geographical areas with 20 organizations, 40 cities, and thousands of operators and developers, including free trials and tests, which increase complexity in management to keep processes under control. The solution is presently in place on https://www.snap4city.org (accessed on 20 August 2024) and accessible for its developers, and it has been produced to cope with multiple interconnected scenarios in the context of Herit-Data, which addressed the exploitation of big data and applications for tourism management in six different cities and many applications. Future research activities are focused on organizing the artificial intelligence processes in execution and training by using the MLOps approach. An extension of the UKM would be needed, and it will be grounded on a new version of the Km4City ontology which is under development to cope with a larger set of complex data and more advanced Digital Twins.

In conclusion, the unified knowledge model (UKM) developed and implemented within the Snap4City platform has successfully addressed the challenges associated with managing complex, multi-tenant smart city infrastructures. By providing a unified framework for problem detection, scalability, and multi-organizational collaboration, the UKM has significantly enhanced the efficiency, reliability, and flexibility of smart city platforms. As smart city technologies continue to evolve, the UKM is well-positioned to serve as a foundational model for future developments, enabling cities around the world to become smarter, more efficient, and more responsive to the needs of their citizens.

Author Contributions: Conceptualization, P.N., P.B. and G.P.; methodology, P.N., P.B. and G.P.; software, P.B., G.P. and D.B.; formal analysis, P.N., P.B. and G.P.; investigation, P.N., P.B. and G.P.; resources, P.N.; writing—original draft preparation, P.N., P.B. and G.P.; writing—review and editing, P.N., P.B., G.P. and D.B.; visualization P.N., P.B., G.P. and D.B.; supervision, P.N.; project administration, P.N.; funding acquisition, P.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been co-financed by the European Regional Development Fund in the context of the Herit-Data project https://herit-data.interreg-med.eu/ (accessed on 20 August 2024) and of the ELLIE project of the European Commission.

Data Availability Statement: Data and resources are available in the Snap4City platform: https://www.snap4city.org (accessed on 20 August 2024). Public data is available, while restricted access may be requested for private data.

Acknowledgments: The authors would also like to thank the several partners that are using the infrastructure for actual scenarios and projects. The specific solutions described in this paper have been added in the context of the Herit-Data project https://herit-data.interreg-med.eu/ (accessed on 20 August 2024) cofinanced by the European Regional Development Fund. The authors would like to thank Asti Manuka and Alessandro Lemmo for developing some of the LOG aspects and Michela Paolucci for some of the reviews during the Data Inspector development. Snap4City and Km4City are open technologies and research of DISIT Lab https://www.snap4city.org (accessed on 20 August 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Filipponi, L.; Vitaletti, A.; Landi, G.; Memeo, V.; Laura, G.; Pucci, P. Smart City: An Event Driven Architecture for Monitoring Public Spaces with Heterogeneous Sensors. In Proceedings of the 2010 Fourth International Conference on Sensor Technologies and Applications, Venice, Italy, 18–25 July 2010; pp. 281–286.
2. Badii, C.; Belay, E.G.; Bellini, P.; Cenni, D.; Marazzini, M.; Mesiti, M.; Nesi, P.; Pantaleo, G.; Paolucci, M.; Valtolina, S.; et al. Snap4City: A Scalable IoT/IoE Platform for Developing Smart City Applications. In Proceedings of the International Conference IEEE Smart City Innovation, Bandung, Indonesia, 25–26 October 2018.
3. Krylovskiy, A.; Jahn, M.; Patti, E. Designing a Smart City Internet of Things Platform with Microservice Architecture. In Proceedings of the 2015 3rd International Conference on Future Internet of Things and Cloud, Rome, Italy, 24–26 August 2015; IEEE: New York, NY, USA, 2015.
4. Fanfani, M.; Palesi, L.A.I.; Nesi, P. Microservices' libraries enabling server-side business logic visual programming for digital twins. *SoftwareX* **2024**, *27*, 101805. [CrossRef]
5. Ruohomäki, T.; Airaksinen, E.; Huuska, P.; Kesäniemi, O.; Martikka, M.; Suomisto, J. Smart City Platform Enabling Digital Twin. In Proceedings of the 2018 International Conference on Intelligent Systems (IS), Madeira, Portugal, 25–27 September 2018; IEEE: New York, NY, USA, 2018.
6. Lv, Z.; Li, X.; Lv, H.; Xiu, W. BIM Big Data Storage in WebVRGIS. *IEEE Trans. Ind. Inform.* **2019**, *16*, 2566–2573. [CrossRef]
7. Adreani, L.; Bellini, P.; Colombo, C.; Fanfani, M.; Nesi, P.; Pantaleo, G.; Pisanu, R. Digital Twin Framework for Smart City Solutions. In Proceedings of the DMSVIVA 2022, The 28th International DMS Conference on Visualization and Visual Languages, Pittsburgh, PA, USA, 29–30 June 2022.
8. Lundgren, J.T.; Peterson, A. A Heuristic for the Bilevel Origin–Destination Matrix Estimation Problem. *Transp. Res. Part B Methodol.* **2008**, *42*, 339–354. [CrossRef]
9. Hashem, I.A.T.; Chang, V.; Anuar, N.B.; Adewole, K.; Yaqoob, I.; Gani, A.; Ahmed, E.; Chiroma, H. The Role of Big Data in Smart City. *Int. J. Inf. Manag.* **2016**, *36*, 748–758. [CrossRef]
10. Bilotta, S.; Palesi, L.A.I.; Nesi, P. Predicting free parking slots via deep learning in short-mid terms explaining temporal impact of features. *IEEE Access* **2023**, *11*, 101678–101693. [CrossRef]
11. Pearl, J. The Seven Tools of Causal Inference, with Reflections on Machine Learning. *Commun. ACM* **2019**, *62*, 54–60. [CrossRef]
12. Naphade, M.; Banavar, G.; Harrison, C.; Paraszczak, J.; Morris, R. Smarter Cities and Their Innovation Challenges. *Computer* **2011**, *44*, 32–39. [CrossRef]
13. Bellini, P.; Benigni, M.; Billero, R.; Nesi, P.; Rauch, N. Km4City Ontology Building vs Data Harvesting and Cleaning for Smart-City Service. *Int. J. Vis. Lang. Comput.* **2014**, *25*, 827–839. [CrossRef]
14. Diouf, P.S.; Boly, A.; Ndiaye, S. Variety of Data in the ETL Processes in the Cloud: State of the Art. In Proceedings of the 2018 IEEE International Conference on Innovative Research and Development (ICIRD), Bangkok, Thailand, 11–12 May 2018; pp. 1–5.
15. Ptiček, M.; Vrdoljak, B.; Gulić, M. The Potential of Semantic Paradigm in Warehousing of Big Data. *Automatika* **2019**, *60*, 393–403. [CrossRef]
16. Abdallaoui, H.E.A.E.; Fazziki, A.E.; Ennaji, F.Z.; Sadgal, M. A System for Collecting and Analyzing Road Accidents Big Data. In Proceedings of the 2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Sorrento, Italy, 26–29 November 2019.
17. Chang, C.H.; Jiang, F.C.; Yang, C.-T.; Chou, S.C. On Construction of a Big Data Warehouse Accessing Platform for Campus Power Usages. *J. Parallel Distrib. Comput.* **2019**, *133*, 40–50. [CrossRef]
18. Diouf, P.S.; Boly, A.; Ndiaye, S. Performance of the ETL Processes in Terms of Volume and Velocity in the Cloud: State of the Art. In Proceedings of the 2017 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS), Salmabad, Bahrain, 29 November–1 December 2017; pp. 1–5.
19. Deibe, D.; Amor, M.; Doallo, R. Big Data Storage Technologies: A Case Study for Web-based LiDAR Visualization. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 December 2018; pp. 3831–3840.
20. Ali, A.R. Real-Time Big Data Warehousing and Analysis Framework. In Proceedings of the 2018 IEEE 3rd IEEE International Conference on Big Data Analysis, Seattle, WA, USA, 10–13 December 2018.

21. Ptiček, M.; Vrdoljak, B. Semantic Web Technologies and Big Data Warehousing. In Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 21–25 May 2018; pp. 1214–1219.
22. Ul Hassan, C.A.; Irfan, R.; Shah, M.A. Integrated Architecture of Data Warehouse with Business Intelligence Technologies. In Proceedings of the 2018 24th Int. Conf. on Automation and Computing (ICAC), Newcastle upon Tyne, UK, 6–7 September 2018; pp. 1–6.
23. Priya, I.; Pathak, I.; Tripathi, A. Big Data, Cloud and IoT: An Assimilation. In Proceedings of the 2018 Second International Conference on Advances in Computing, Control and Communication Technology (IAC3T), Allahabad, India, 21–23 September 2018; pp. 34–40.
24. Ge, M.; Bangui, H.; Buhnova, B. Big Data for Internet of Things: A Survey. *Future Gener. Comput. Syst.* **2018**, *87*, 601–614. [CrossRef]
25. Inoubli, W.; Aridhi, S.; Mezni, H.; Maddouri, M.; Nguifo, E.M. An Experimental Survey on Big Data Frameworks. *Future Gener. Comput. Syst.* **2018**, *86*, 546–564. [CrossRef]
26. Li, W.; Tropea, G.; Abid, A.; Detti, A.; Le Gall, F. Review of Standard Ontologies for the Web of Things. In Proceedings of the 2019 Global IoT Summit (gIoTS), Aarhus, Denmark, 17–21 June 2019; IEEE: New York, NY, USA, 2019; pp. 1–6.
27. Nadal, S.; Herrero, V.; Romero, O.; Abelló, A.; Franch, X.; Van-summeren, S.; Valerio, D. A Software Reference Architecture for Semantic-aware Big Data Systems. *Inform. Softw. Technol.* **2017**, *90*, 75–92. [CrossRef]
28. Haller, A.; Janowicz, K.; Cox, S.J.D.; Lefrançois, M.; Taylor, K.; Le Phuoc, D.; Lieberman, J.; García-Castro, R.; Atkinson, R.; Stadler, C. The Modular SSN Ontology: A Joint W3C and OGC Standard Specifying the Semantics of Sensors, Observations, Sampling, and Actuation. *Semant. Web J.* **2018**, *10*, 9–32. [CrossRef]
29. Vila, M.; Sancho, M.R.; Teniente, E.; Vilajosana, X. Semantics for Connectivity Management in IoT Sensing. In Proceedings of the 40th International Conference on Conceptual Modeling (ER 2021), Virtual, 18–21 October 2021; pp. 297–311.
30. Sejdiu, B.; Ismaili, F.; Ahmedi, L. IoTSAS: An Integrated System for Real-Time Semantic Annotation and Interpretation of IoT Sensor Stream Data. *Computers* **2021**, *10*, 127. [CrossRef]
31. Hogan, A.; Blomqvist, E.; Cochez, M.; d'Amato, C.; de Melo, G.D.; Gutierrez, C.; Kirrane, S.; Gayo, J.E.L.; Navigli, R.; Neumaier, S.; et al. Knowledge Graphs. *ACM Comput. Surv.* **2021**, *54*, 1–37. [CrossRef]
32. Dhungana, D.; Haselböck, A.; Meixner, S.; Schall, D.; Schmid, J.; Trabesinger, S.; Wallner, S. Multi-factory Production Planning Using Edge Computing and IIoT Platforms. *J. Syst. Softw.* **2021**, *182*, 111083. [CrossRef]
33. Gheisari, M.; Najafabadi, H.E.; Alzubi, J.A.; Gao, J.; Wang, G.; Abbasi, A.A.; Castiglione, A. OBPP: An ontology-based framework for privacy-preserving in IoT-based smart city. *Future Gener. Comput. Syst.* **2021**, *123*, 1–13. [CrossRef]
34. De Nicola, A.; Villani, M.L. Smart City Ontologies and Their Applications: A Systematic Literature Review. *Sustainability* **2021**, *13*, 5578. [CrossRef]
35. OneM2M Ontology ETSI Technical Specifications. Available online: https://www.etsi.org/deliver/etsi_ts/118100_118199/118112/02.02.02_60/ts_118112v020202p.pdf (accessed on 20 August 2024).
36. W3C Thing Description (TD) Ontology. Available online: https://www.w3.org/2019/wot/td (accessed on 20 August 2024).
37. Bauer, M.; Kovacs, E.; Schülke, A.; Ito, N.; Criminisi, C.; Goix, L.W.; Valla, M. The Context API in the OMA Next Generation Service Interface. In Proceedings of the 2010 14th International Conference on Intelligence in Next Generation Networks, Berlin, Germany, 11–14 October 2010; pp. 1–5.
38. Jara, A.J.; Serrano, M.; Gomez, A.; Fernandez, D.; Molina, G.; Bocchi, Y.; Alcarria, R. Smart Cities Semantics and Data Models. In Proceedings of the International Conference on Information, Technology & Systems (ICITS 2018), Advances in Intelligent Systems and Computing, Libertad, Ecuador, 10–12 January 2018; Springer: Berlin/Heidelberg, Germany, 2018; Volume 721.
39. Caballero, V.; Valbuena, S.; Vernet, D.; Zaballos, A. Ontology-Defined Middleware for Internet of Things Architectures. *Sensors* **2019**, *19*, 1163. [CrossRef]
40. Raghavan, S.; Simon, B.Y.L.; Lee, Y.L.; Tan, W.L.; Kee, K.K. Data Integration for Smart Cities: Opportunities and Challenges. In *Computational Science and Technology. Lecture Notes in Electrical Engineering*; Alfred, R., Lim, Y., Haviluddin, H., On, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 603, pp. 393–403.
41. Huang, C.Y.; Chiang, Y.H.; Tsai, F. An Ontology Integrating the Open Standards of City Models and Internet of Things for Smart-City Applications. *IEEE Internet Things J.* **2022**, *9*, 20444–20457. [CrossRef]
42. *Open Geospatial Consortium Standard 15-078r6*; OGC SensorThings API Part 1: Sensing. Open Geospatial Consortium: Arlington, VA, USA, 2016.
43. *Open Geospatial Consortium Standard 12-019*; OGC City Geography Markup Language (CityGML) Encoding Standard. Open Geospatial Consortium: Arlington, VA, USA, 2012.
44. *Open Geospatial Consortium Standard 14-005r3*; OGC Indoorgml. Open Geospatial Consortium: Arlington, VA, USA, 2014.
45. Komninos, N.; Bratsas, C.; Kakderi, C.; Tsarchopoulos, P. Smart City Ontologies: Improving the Effectiveness of Smart City Applications. *J. Smart Cities* **2019**, *1*, 31–46. [CrossRef]
46. Bellini, P.; Nesi, P. Performance Assessment of RDF Graph Databases for Smart City Services. *J. Vis. Lang. Comput.* **2018**, *45*, 24–38. [CrossRef]

47. Arman, A.; Bellini, P.; Nesi, P. Searching for Heterogeneous GeoLocated Services via API Federation. In Proceedings of the ICCSA2022, 22nd International Conference on Computational Science and its Applications, Malaga, Spain, 4–7 July 2022; LNCS Springer Verlag: Heidelberg, Germany.
48. Scuotto, V.; Ferraris, A.; Bresciani, S. Internet of Things: Applications and Challenges in Smart Cities: A Case Study of IBM Smart City Projects. *Bus. Process. Manag. J.* **2016**, *22*, 357–367. [CrossRef]
49. Strickland, E. Cisco Bets on South Korean Smart City. *IEEE Spectr.* **2011**, *48*, 11–12. [CrossRef]
50. Adrianto, D.; Lin, F.J. Analysis of Security Protocols and Corresponding Cipher Suites in ETSI M2M Standards. In Proceedings of the 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), Milan, Italy, 14–16 December 2015; pp. 777–782.
51. Chen, H.C.; You, I.; Weng, C.E.; Cheng, C.H.; Huang, Y.F. A Security Gateway Application for End-to-End M2M Communications. *Comput. Stand. Interfaces* **2016**, *44*, 85–93. [CrossRef]
52. IoT-A. *Converged Architectural Reference Model for the IoT, v2.0*; SAP: Zurich, Switzerland, 2012.
53. Hakim, A.E. Internet of Things (IoT) System Architecture and Technologies. *White Pap.* **2018**, *10*, 2.
54. Aguru, A.D.; Babu, E.S.; Nayak, S.R.; Sethy, A.; Verma, A. Integrated Industrial Reference Architecture for Smart Healthcare in Internet of Things: A Systematic Investigation. *Algorithms* **2022**, *15*, 309. [CrossRef]
55. Anthopoulos, L.; Fitsilis, P. Exploring Architectural and Organizational Features in Smart Cities. In Proceedings of the 2014 16th International Conference on Advanced Communication Technology, ICACT, Pyeong Chang, Republic of Korea, 16–19 February 2014; IEEE: New York, NY, USA, 2014.
56. Badii, C.; Bellini, P.; Cenni, D.; Difino, A.; Nesi, P.; Paolucci, M. Analysis and Assessment of a Knowledge Based Smart City Architecture Providing Service APIs. *Future Gener. Comput. Syst.* **2017**, *75*, 14–29. [CrossRef]
57. Datta, S.K.; Bonnet, C. Next-Generation, Data Centric and End-to-End IoT Architecture Based on Microservices. In Proceedings of the 2018 IEEE International Conference on Consumer Electronics–Asia (ICCE-Asia), Jeju, Republic of Korea, 24–26 June 2018.
58. Zimmermann, O. Microservices Tenets: Agile Approach to Service Development and Deployment. *Comput. Sci. Res. Dev.* **2017**, *32*, 301–310. [CrossRef]
59. Dragoni, N.; Giallorenzo, S.; Lafuente, A.L.; Mazzara, M.; Montesi, F.; Mustafin, R.; Safina, L. Microservices: Yesterday, Today, and Tomorrow. In *Present and Ulterior Software Engineering*; Springer: Cham, Switzerland, 2017; pp. 195–216.
60. Sinaeepourfard, A.; Garcia, J.; Masip-Bruin, X.; Marín-Torder, E. Towards a Comprehensive Data Lifecycle Model for Big Data Environments. In Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT), Shanghai, China, 6–9 December 2016; IEEE/ACM: New York, NY, USA, 2016; pp. 100–106.
61. Rahman, L.F.; Ozcelebi, T.; Lukkien, J. Understanding IoT Systems: A Life Cycle Approach. *Procedia Comput. Sci.* **2018**, *130*, 1057–1062. [CrossRef]
62. Diamantini, C.; Mircoli, A.; Potena, D.; Storti, E. Process-aware iIoT Knowledge Graph: A Semantic Model for Industrial IoT Integration and Analytics. *Future Gener. Comput. Syst.* **2023**, *139*, 224–238. [CrossRef]
63. Beheshti, A.; Benatallah, B.; Motahari-Nezhad, H.R.; Ghodratnama, S.; Amouzgar, F. BP-SPARQL: A Query Language for Summarizing and Analyzing Big Process Data. In *Process Querying Methods*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 21–48.
64. Quirino, G.K.; Nardi, J.C.; Barcellos, M.P.; Falbo, R.A.; Guizzardi, G.; Guarino, N.; Longo, A.; Livieri, B. Towards a Service Ontology Pattern Language. In *Conceptual Modeling. ER 2015. Lecture Notes in Computer Science*; Johannesson, P., Lee, M., Liddle, S., Opdahl, A., Pastor López, Ó., Eds.; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9381.
65. Falbo, R.A.; Quirino, G.K.; Nardi, J.C.; Barcellos, M.P.; Guizzardi, G.; Guarino, N.; Longo, A.; Livieri, B. An Ontology Pattern Language for Service Modeling. In Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, 4–8 April 2016; pp. 321–326.
66. Hefnawy, A.; Bouras, A.; Cherifi, C. IoT for Smart City Services: Lifecycle Approach. In Proceedings of the International Conference on Internet of Things and Cloud Computing, Dalian, China, 22–23 October 2016; pp. 1–9.
67. Bellini, P.; Fanfani, M.; Nesi, P.; Pantaleo, G. Snap4City dashboard manager: A tool for creating and distributing complex and interactive dashboards with no or low coding. *SoftwareX* **2024**, *26*, 101729. [CrossRef]
68. López-Riquelme, J.A.; Pavón-Pulido, N.; Navarro-Hellín, H.; Soto-Valles, F.; Torres-Sánchez, R. A Software Architecture Based on FIWARE Cloud for Precision Agriculture. *Agric. Water Manag.* **2017**, *183*, 123–135. [CrossRef]
69. Bellini, P.; Bilotta, S.; Ipsaro Palesi, L.A.; Nesi, P.; Pantaleo, G. Vehicular Traffic Flow Reconstruction Analysis to Mitigate Scenarios with Large City Changes. *IEEE Access* **2022**, *10*, 131061–131075. [CrossRef]
70. Zhoua, Q.; Simmhanb, Y.; Prasanna, V. Knowledge-infused and Consistent Complex Event Processing over Real-time and Persistent Streams. *Future Gener. Comput. Syst.* **2017**, *76*, 391–406. [CrossRef]
71. Badii, C.; Bellini, P.; Difino, A.; Nesi, P. Smart City IoT Platform Respecting GDPR Privacy and Security Aspects. *IEEE Access* **2020**, *8*, 23601–23623. [CrossRef]
72. Bellini, P.; Nesi, P.; Venturi, A. Linked Open Graph: Browsing Multiple SPARQL Entry Points to Build Your Own LOD View. *Int. J. Vis. Lang. Comput.* **2014**, *25*, 703–716. [CrossRef]
73. Poveda-Villalón, M.; Gómez-Pérez, A.; Suárez-Figueroa, M.C. Oops!(ontology Pitfall Scanner!): An Online Tool for Ontology Evaluation. *Int. J. Semant. Web Inf. Syst. IJSWIS* **2014**, *10*, 7–34. [CrossRef]

74. Cirillo, F.; Solmaz, G.; Berz, E.L.; Bauer, M.; Cheng, B.; Kovacs, E. A Standard-based Open Source IoT Platform: FIWARE. *IEEE Internet Things Mag.* **2019**, *2*, 12–18. [CrossRef]
75. Bellini, P.; Palesi, L.A.I.; Giovannoni, A.; Nesi, P. Managing complexity of data models and performance in broker-based Internet/Web of Things architectures. *Internet Things* **2023**, *23*, 100834. [CrossRef]
76. Bellini, E.; Coconea, L.; Nesi, P. A Functional Resonance Analysis Method driven Resilience Quantification for socio-technical Systems. *IEEE Syst. J.* **2019**, *14*, 1234–1244. [CrossRef]