



UNIVERSITÀ
DEGLI STUDI
FIRENZE

UNIVERSITÀ DEGLI STUDI DI FIRENZE
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE (DINFO)
CORSO DI DOTTORATO IN INGEGNERIA DELL'INFORMAZIONE
CURRICULUM: INFORMATICA

SEMANTIC MODELS FOR THE
MODELING AND MANAGEMENT OF
BIG DATA IN A SMART CITY
ENVIRONMENT

Candidate

Mirco Soderi

Supervisors

Prof. Paolo Nesi

Prof. Pierfrancesco Bellini

PhD Coordinator

Prof. Fabio Schoen

CICLO XXXIII, 2017-2020

Università degli Studi di Firenze, Dipartimento di Ingegneria
dell'Informazione (DINFO).

Thesis submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Information Engineering. Copyright © 2021 by
Mirco Soderi.

A Vittoria, Romano, Giovanni

Acknowledgments

I would like to acknowledge the efforts and input of my supervisors, Prof. Paolo Nesi and Prof. Pierfrancesco Bellini, and all my colleagues of the Distributed Systems and Internet Technologies Lab (DISIT LAB) who were of great help during my research.

Abstract

The overall purpose of this research has been the building or the improvement of semantic models for the representation of data related to smart cities and smart industries, in such a way that it could also be possible to build context-rich, user-oriented, efficient and effective applications based on such data. In some more detail, one of the key purposes has been the modelling of structural and the functioning aspects of the urban mobility and the production of instances exploiting the Open Street Map, that once integrated with traffic sensors data, it has lead to the building and displaying of real-time traffic reconstructions at a city level. One second key purpose has been the modelling of the Internet of Things, that allows today to seamlessly and efficiently identify sensing devices that are deployed in a given area or along a given path and that are of a given type, and also inspect real-time data that they produce, through a user-oriented Web application, namely the Service Map. A pragmatic approach to the modelling has been followed, always taking into consideration the best practices of semantic modelling on one side for that a clean, comprehensive and understandable model could result, and the reality of the data at our hands and of the applicative requirements on the other side. As said, the identification of architectures and methods that could grant efficiency and scalability in data access has also been a primary purpose of this research that has led to the definition and implementation of a federation of Service Maps, namely the Super Service Map. The architecture is fully distributed: each Super Service Map has a local list of the actual Service Maps with relevant metadata, it exposes the same interface as actual Service Maps, it forwards requests and builds merged responses, also implementing security and caching mechanisms. As said, the identification of technologies, tools, methods, for presenting the data in a user-friendly manner is also has been a relevant part of this research, and it has led among the other to the definition and implementation of a client-server architecture and a Web interface in the Snap4City platform for the building, management, and displaying of synoptic templates and instances thanks to which users can securely display and interact with different types of data. In end, some effort has been made for the automatic classification of RDF datasets as for their structures and purposes, based on the computation of metrics through SPARQL queries and on the application of dimensionality reduction and clustering techniques. A Web portal is available where directories, datasets, metrics, and computations can be inspected even at real-time.

Contents

Contents	vii
1 Introduction	1
2 Literature review	5
2.1 Knowledge Modeling and Management for Mobility and Transport Applications	5
2.2 Monitoring and Prediction of Vehicular Traffic	7
2.3 Smart City Platforms	7
2.3.1 Federation of API Services	10
2.4 Classification of RDF datasets through Multilayer Metrics . .	12
3 Knowledge Modeling and Management for Mobility and Transport Applications	15
3.1 Introduction	16
3.2 Requirements for Mobility and Transport Applications	18
3.3 Km4City Extended Model	20
3.3.1 Roads	22
3.3.2 Roads partitioning	22
3.3.3 Lanes	22
3.3.4 Restrictions	23
3.4 Knowledge Base Construction	25
3.4.1 The roads infrastructure ingestion: motivations	26
3.5 The Km4City Traffic Reconstruction	26
3.5.1 Real-time density reconstruction: mathematical model	28
3.5.2 Real-time density reconstruction: representation model	29
3.6 Knowledge Base Experimenting	31
3.7 Conclusions	32

4	Snap4City: A Scalable IOT/IOE Platform for Developing Smart City Applications	35
4.1	Introduction	36
4.2	Snap4City Architecture	38
4.3	An Example	42
4.4	Performance Analysis	45
4.4.1	Assessing Communications Bandwidth	48
4.4.2	Assessing Maximum Number of Messages	49
4.5	Conclusions and Future Work	51
5	Snap4city Platform: Semantic to Improve Location Based Services	55
5.1	Introduction	56
5.2	Snap4City Architecture	57
5.3	Km4City Multi-Ontology	58
5.4	Snap4city Location-Based Services	59
5.5	Conclusion. Snap4city Validation Scenarios in Smart Cities .	60
6	Federation of Smart City Services via APIs	63
6.1	Introduction	64
6.2	Requirements and analysis	65
6.3	General architecture	66
6.4	SuperServiceMap Operational Semantic	68
6.5	Advanced Smart City API, ASCAPI	70
6.6	The SuperServiceMap Main Processing Aspects	71
6.6.1	Selection of ServiceMaps to be queried and exceptions	72
6.6.2	Merging ServiceMap results of any Format	72
6.7	Validation and Experimental Results	73
6.8	Conclusion	79
7	Industry 4.0 Synoptics Controlled by IoT Applications in Node-RED	81
7.1	Introduction	82
7.2	Requirements for Industry 4.0 Applications	85
7.3	Snap4City vs Snap4Industry Architecture	89
7.3.1	Snap4Industry Architecture	92
7.4	Synoptics Development Environment	95
7.4.1	GDPR Functionalites and Aspects	98

7.5	Controlling Synoptics From IoT Apps	98
7.6	Conclusions	102
8	Real-Time Traffic Estimation of Unmonitored roads	103
8.1	Introduction	104
8.2	Modeling Road Information For Traffic Flow	106
8.3	City Traffic Flow Model	109
8.3.1	The numerical approximation scheme	111
8.4	City Graph Structure	112
8.4.1	The city graph representation	113
8.4.2	Traffic distribution in the junction	114
8.4.3	Data sensors' integration in the city graph	115
8.4.4	On the graph reduction and road-segment definition	115
8.5	Graphical Representation	116
8.5.1	Road-segments depiction	116
8.5.2	Graphics improvement	118
8.6	Computational Approach	118
8.6.1	Sensors' values	119
8.6.2	Weights' assignment	119
8.6.3	Density computation	120
8.6.4	Graphical representation	121
8.7	Experimental Results	121
8.8	Conclusion	122
9	Traffic Flow Reconstruction from Scattered Data	125
9.1	Introduction	126
9.2	City Traffic Flow Model	128
9.3	Computational Approach	129
9.4	Experimental Results	130
9.5	Conclusions	132
10	How COVID-19 Lockdown Impacted on Mobility and Environmental data	135
10.1	Introduction	136
10.2	Snap4City overview	137
10.3	Impact of lockdown on traffic data	138
10.4	Impact of lockdown on parking facilities and deductions	145
10.5	Impact of lockdown on environmental data and deductions	147

10.6	Conclusions	149
11	Classification of RDF datasets through Multilayer Metrics	151
11.1	Introduction	152
11.2	Metrics	154
11.2.1	Absolute Metrics	154
11.2.2	Graph Count Metric (GCM)	155
11.2.3	Instance Count Metric (ICM)	156
11.2.4	Same As Metric (SAM)	156
11.2.5	Subclass Count Metric (SCM)	156
11.2.6	Property Count Metric (PCM)	156
11.2.7	Object Classes Metric (OCM)	157
11.2.8	Language Metric (LAM)	157
11.2.9	Ontology Metric (OME)	157
11.2.10	Instantiated Object Properties By Class (IOP)	158
11.2.11	Instantiated Data Properties By Class (IDP)	158
11.2.12	Vocabulary Metric (VOM)	158
11.2.13	Weighted Number of Relatives (WNR)	159
11.2.14	Coupling Between Resources (CBR)	159
11.3	Middle-Level Derived Metrics	160
11.4	Metrics Categorization	162
11.5	Metrics Computation Tool	162
11.5.1	The Datameter Web Portal of DISIT Lab	166
11.6	Experimental Results	167
11.7	Principal Component Analysis	170
11.8	Classification of RDF datasets	180
11.9	Conclusions	185
12	Conclusion	187
A	Appendix	193
A.1	Acronyms of RDF Dataset Metrics	193
B	Publications	197
	Bibliography	199

Chapter 1

Introduction

In the context of smart city, real-time routing and traffic prediction applications have specific requirements as for their needed input data, shaping, and efficient retrieval. The recent applications and services need to have integrated data store for efficient data access and deduction. The models at the state of the art are not satisfactory. Therefore, Sii-Mobility research and development project of Italian Ministry has developed a new model ontology, named the Km4City. It makes possible, among several services also the routing and traffic reconstruction. In the context of this research, the model is identified and processes for knowledge base construction are described. The process takes into account Open Street Map, open data, and real-time data from sensors, mobility and transport operator in multiple formats and protocols, to load them into Km4City KB. The research on data modelling is described in detail in Chapter 3.

Smart City solutions, initially started with open data, are evolving towards data aggregation and semantics. Recently, some of them are also offering IOT support. The combination of IOT and smart city is not an easy task: the data volumes are much higher than those addressed for industrial IOT. The complexity of IOT smart city solutions have been identified by a number of actors. The European commission started to set up the EIP project for stimulating and concerting actions. The Select4Cities project of the European Commission and associated community <http://www.select4cities.eu/> created a challenge to find research solutions satisfying a formalized set of functional and nonfunctional requirements. Snap4City, presented in Chapters 4 and 5, is one of the solutions developed in response to that challenge.

The solution proposed offers a platform where sophisticated IOT applications for controlling city dashboards as well as IOT mobile applications can be developed in few steps. Moreover, a number of development and monitoring tools have been developed. Among them, a special attention is given to the tools and solutions for monitoring communication performance and to perform the assessment of scalability.

In the context of Smart City, it is quite frequent the usage of Smart City API for providing services at Web and mobile applications. Most of the solutions using Smart City APIs are focused on a single city. This means that passing from one city/area to another, the users must change application. This happens for the lack of interoperability among Smart City APIs and/or services. In Chapter 6, the problem of federation of smart city services is addressed by proposing a solution for federating smart city APIs. To this end, a formal model has been proposed to federate API services, with efficiency, security, scalability, and capacity of managing overlapped areas of competence, distributed searches, etc.. These features are typically not all satisfied by classic GIS solutions which federate the services at level of databases.

The push towards Industry 4.0 is constraining the industries to work in integrated supply chains. This implies to be opened at the integration of their production plants with other plants, and to provide access to their data and processes. In most cases, this also means to give access to data and flows to their customers and to perform some synchronization and supervision, and maybe to create integrated control rooms, synoptics and dashboards. This activity is also facilitated by the introduction of IoT solutions with IoT Devices, IoT Brokers, etc., which have a completely different approach with respect to DCS and SCADA solutions usually adopted in the industry for controlling their local productions. In Chapter 7, Snap4Industry with its IoT development environment and framework for implementing the Control and Supervision of Multiple Supply Chains in the view of Industry 4.0 as a service is presented. In particular, the chapter describes the motivations/requirements and the actions performed to extend IoT Snap4City 100% open source platform to comply with Industry 4.0 requirements. The main additions for creating Snap4Industry solution have been on: (i) industry protocols, (ii) custom widgets and synoptics Dashboards, (iii) new MicroServices for Node-RED for enabling the usage of synoptics as event driven devices, (iv) automatized process for producing synoptic templates

according to GDPR.

Due to growing cities, the real-time knowledge about the state of traffic is a critical problem for urban mobility and the road-segment traffic densities estimation influences the efficiency of fundamental smart city services as smart routing, smart planning for evacuations, planning of civil works on the city, etc. Nevertheless, the traffic-related data from navigator Apps (e.g., TomTom, Google, Bing) are too expensive to be acquired. Also, the traditional sensors for the traffic flow detection are very expensive, and they are usually not dense enough for a correct traffic monitoring. In order to overcome such problems, there is a space for low cost and fast solutions for dense traffic flow reconstruction. In Chapters 8 and 9 we propose a real-time visual self-adaptive solution to reconstruct the traffic density at every location of a wide urban area leveraging the detections from a few fixed traffic sensors deployed within the area of interest. The method is based on fluid dynamic models to simulate macroscopic phenomena as shocks formation and propagation of waves backwards along roads.

According to the COVID-19 lockdown and successive reopening a number of facts can be analysed. The main effects have been detected on: mobility and environment, and specifically on traffic, environmental data and parking. The mobility reduction has been found to be quite coherent with respect to what has been described by Google Global mobility report. On the other hand, in Chapter 10 a number of additional aspects have been put in evidence providing detailed aspects on mobility and parking that allowed us to better analyze the impact of the reopening on an eventual revamping of the infection, also taking into account of the R_t index. To this end, the collected data from the field have been compared from those of Google and some considerations with respect to the Imperial college Report 20 have been derived. For the pollutant aspects, a relevant reduction on most of them has been measured and rationales are reported. The solution has exploited the Snap4City IOT smart city infrastructure and data collector and Dashboard in place in Tuscany.

More and more Linked Data repositories (RDF datasets) are being published in the last years. Directories exist where they are catalogued, but it is still not easy to understand at the first glance the ontological model that RDF datasets adopt, and thus in substance their purpose and aim. For example, for each RDF dataset one should be capable to understand if it is a dictionary or the base of an expert system, if it is an encyclopaedia or

a smart city model, and so on. Indeed, the idea behind triples is to have the possibility of reusing them as Web of things and definitions. Proposals can be found in the literature of metrics used for assessing the quality and purpose of RDF datasets. In Chapter 11, we revise, analyse, and classify those metrics according to their aim and meaning, and we define a set of additional metrics. Then, a Principal Component Analysis is performed for identifying those aspects that are of major interest for the automatic classification of RDF datasets, and for identifying those metrics that are the most suitable for quantifying each of such aspects. RDF dataset clusters are generated based on Principal Components. Five clusters result from the process (thesauruses, lists, gazetteers, semantic networks, and classification schemas), that correspond to structural categories of the DCMI/NKOS KOS Types Vocabulary. This way, clustering by purpose of RDF datasets is made systematic, that is the key contribution that this work provides.

Chapter 2

Literature review

This chapter gives a brief survey of related work on the semantic modeling and management of big data in smart cities and industries.

2.1 Knowledge Modeling and Management for Mobility and Transport Applications

Below here, the main smart city data models that can be suitable for mobility and transport are reviewed.

The iCity model in [94] is a comprehensive data model focused on the transportation systems where public and private activities and points of interest can be found. The data model in [80] has been developed for the purposes of the VITAL project, and it appears to be mainly focused on the domain of the mobility and transport, since it enables the prediction of the traffic congestions and the automatic detection of the incidents within the city of Istanbul. The iCity and the VITAL data models both rely on the OTN [19] ontology for the representation of the structural aspects. The major drawback of such a choice is that house numbers are not geolocated. Instead, the range of the house numbers that can be found in a segment of road is modelled, that is very likely to lead to an imprecise routing. CityPulse in [6] is focused on the domain of the mobility and transport in cities and it enables real-time IoT stream processing and large-scale data analytics for the provisioning of innovative ICT services. StarCity in [103] is aimed at

supporting semantic traffic analytic and reasoning for cities. The StarCity data model does not include any concept related to the modelling of the road infrastructure. Komninos et al. in [97] propose a data model based on a review about the smart city applications related to the domains of energy and transport. The model does not include a deep modelling of the roads. Indeed, while dedicated concepts can be found that model roads with specific relevance, destination, localization, and traffic direction, other key elements are completely missing, e.g. lanes, restrictions, street numbers, road segmentation, and other. Also, while sensors are modelled, their observations are not. Abid et al. in [24] proposed a semantic data model for managing and resolving the problems that exist in cities such as water leak, street faults, broken street lights, and potholes. The Abid et al. data model does not include any concept related to the road infrastructure, sensing devices, and observations, since it is focused on the modelling of forms filled by citizens or intended professionals and related to generic disruptive events. The intent of the PRISMA project in [64] is to develop a data model for smart cities able to describe and integrate data from multiple domains, such as geographic information, public transportation, road maintenance, waste collection, and urban faults management. Finally, the Open Street Map (OSM) is a collaborative planet map licensed as open data and empowered by useful software tools. The PRISMA and OSM data models do not include any concept for modelling the sensing devices, and particularly the traffic sensors, and the observations of such sensing devices. The OSM data modelling does not suit the needs of real-time routing and traffic reconstruction solutions.

In [51] an Open Urban Platform for a Sentient Smart City has been proposed. It was based on Km4City Ontology as smart city model including multiple domains such as: mobility, culture, tourism, POI, etc. Km4City has been based on a number of vocabularies such as: (i) the OTN vocabulary, that has been exploited to model traffic [19] that is more or less a direct encoding of GDF (Geographic Data Files) in OWL; (ii) dcterms: set of properties and classes maintained by the Dublin Core Metadata Initiative; (iii) foaf: dedicated to the description of the relations between people or groups; (iv) vCard: for a description of people and organizations; (v) wgs84_pos: vocabulary representing latitude and longitude, with the WGS84 Datum, of geo-objects.

2.2 Monitoring and Prediction of Vehicular Traffic

Simple network area as freeways or rings, where incoming and outgoing roads have a limited number, is monitored by the studies that consider only sensors in fixed locations, see for instance [137] [130]. Also often, the traffic state analysis is related to the monitored area in terms of short-term traffic flow prediction on fixed points and no information is given where sensors are not located [85] [107]. Other studies both use stationary sensors and GPS data from mobile applications to analyze the state of traffic in an urban network [5] [110]. In [136] it is proposed a real-time road traffic prediction with spatiotemporal correlations based on data coming from both the loop detectors and the taxis to predict the speed and volume of traffic in a business district of an urban area. In [84] a smartphone-based crowd sensing system for traffic regulator detection and identification is proposed. As for the prediction/reconstruction, in [109] a deep Restricted Boltzmann Machine and Recurrent Neural Network architecture is utilized to model and predict traffic congestion evolution based on (GPS) data from taxis. Fluid dynamic models such as the macroscopic model proposed in [58] can also be found in the literature that allow observing the network in the time evolution through waves formation. The fluid dynamics concepts are applied to traffic situation for the first time in [104], and independently in [123] with limited scope. The application of the fluid dynamics concepts to traffic flows leads to the formulation of the equation of the conservation of cars. The solution of the above equation can be obtained by finite differences in [76], bringing to a solution of partial differential equations [93], [115]. The first step to develop a computerized solution for the proposed reconstruction problem is to discretize time and space by means of a time-space grid. Following the well-known Cell Transmission Model [66] [67], a road can be partitioned into a series of units, and similarly, the time is divided into a series of durations.

2.3 Smart City Platforms

The world of ICT solutions for smart city is very wide encompassing services at different levels of complexity and coverage, from classical Open Data portals (CKAN [7], Open- DataSoft [20], ArcGIS and OpenData [1]) to sophisticated solutions that provide data aggregation and Smart City APIs fa-

cilitating the development of Web and mobile applications (Km4City [120], City SDK [7]). Only few of them offer development tools for data analytics and application development [33]. The IoT contributed to the development of smart cities with new solutions deploying sensors/actuators with a set of protocols directly enforced in the devices, such as: OneM2M [88], ETSI M2M [87], MQTT [16], COAP, SigFox, AMQP, LoraWAN, Green Button Connect [11]. A great impulse to the field has been given by the European Union. A set of challenges has been proposed by EIP European commission actions for Urban Platforms [23], mainly on clean power for vehicles, multi-modality routing, smart logistic, sustainable mobility, etc. As a reply to the challenge, our proposed solution started from the exploitation of Km4City platform and its smart city API (<http://www.km4city.org>) [33], adding IOT/IOE capabilities and scalable management. Many other proposals can be found in the literature. According to [13] more than 450 platforms for IOT have been presented that provide implementation for the functional blocks described in [125]. A comparison of the most representative IOT solutions at the state of the art is reported in Fig. 2.1. Among them, according to the information collected, we can see solutions that clearly declare to be suitable for the smart city scenarios and IOT such as Kaa, Bosch, FIWARE, CISCO, IBM and Carriots and few more. To be noted here about the Kaa IoT Platform, the first listed, that despite an open source version of the platform is available for download, it is no longer supported both as for its related documentation and possible issues. A new version of the platform has been produced that constitutes an improvement of the open source version under many aspects, but it is not open source.

Among the graphical environments for the specification of applications working with streams of IOT data, NodeRed [18] is probably the most effective for rapid prototyping and has a large portability and limited footprint. Other solutions, as Eclipse, are based on Java and present larger footprint and more complex languages to be used. Moreover, NodeRed can be used in conjunction with Kafka, Spark, R studio, external services, etc., adding simple blocks, and thus the scalability on massive access to sensors can be delegated to data driven data collectors in blocks also exploiting IOT Directory/Discovery services.

Many efforts are nowadays devoted to the creation of virtual bridges among these platforms in order to guarantee the development of cross-platforms (also named horizontal) applications, that is applications able to

	Open Source end-to-end	Scalability IOT	Execution scalability	Visual Programming end-to-end applications	Advanced Smart City API	Multi Domain Semantic Platform	Process Allocation IOT API, scalability computing	Standard based Modules and IOT	City Dashboard H24/7	Multi-protocol on IOT
<i>KAA</i>	Y	Y	Y	N	Y	N	Y	?	N	Y
<i>IOT IGNITE</i>	Y (client)	Y	N	Y	N	N	N	N	N	MQTT only
<i>PTC ThingWorx</i>	N	Y	Y	Y	N	N	N	Y	(via SQL, UE, AL)	Y (via extensions)
<i>BEZIRK</i>	Y	N	N	N	N	Y	N	Y	N	Y
<i>Bosch IoT Suite</i>	N	Y	Y	Y	Y	N	N	Y	Y	Y
<i>FIWARE</i>	Y	(Y)	N	N	Y	N	N	Y	Y	Y
<i>CISCO Jasper</i>	N	Y	?	N	N	N	N	N	N	N
<i>IBM Watson IoT</i>	N	Y	Y	Y (Node-RED)	Y	Y	N	Y	(Y)	Y
<i>Siemens MindSphere</i>	N	Y	?	Y (via ATOS)	N	N	N	Y	Y (via IBM)	Y
<i>Carriots</i>	N	Y	?	N	N	N	N	?	Y	(MQTT only)
<i>Thingsboard</i>	Y	Y	Y	N	N	N	N	N	N	(MQTT, CoAP, http)
<i>IOT eclipse.ora</i>	Y	Y	?	N	N	N	N	Y	N (via BIRT)	Y

Figure 2.1: Comparison among representative IOT platforms. Legend: Y/N present/not present feature; (Y) partially present; ? no evidence from docs

connect sensors belonging to different platforms. European projects (like OpenIoT, BIGIoT, Biotope, INTER-IoT, SymbIoTe) are moving in this direction and their idea is to offer facilities across all layers of the network stack in order to improve the interoperability among the different components involved in the management of sensors, actuators and network infrastructures. XGSN [89] (extension of the GSN middleware [92]) is one of the first middleware (at the base of the OpenIoT project) for the IoT that supports a Domain Ontology for mitigating the semantic interoperability issues arising when integrating heterogeneous physical and virtual sensors. It exploits the SSN Ontology [108] for semantically annotating sensor data and observations in order to provide a standardized queryable representation that makes it easier to share, discover, integrate and interpret the data. Snap4city is moving in the same direction of this system with the following peculiarities. First, we adopt the Km4City Ontology that allows to better represent all the kinds of data that can be generated in a city (and not only the sensor/actuator data). Moreover, Snap4city adopts intelligent approaches for the semantic discovery of new sensors/actuators and their classification in the domain Ontology. Finally, Snap4city combines modern solutions in order to develop a scalable and efficient architecture that easily adapts to the millions of events that need to be treated in the context of smart cities.

A large number of smart city projects are focused on creating big data infrastructure and solutions such as REPLICATE H2020, RESOLUTE H2020, Triangulum H2020, EIP [23]. In [90] the case of smart city IOT integration has been discussed for the city of Santander without proposing details and performance analysis of the solution. In [98], a smart city IOT architecture has been proposed without addressing the aspects of scalability. As outlined in Fig. 2.1, none of the analyzed solutions is capable to address all the requirements. Most of them lack access to smart city data via API, many others are not scalable, a large number of them are limited in supporting multiple IOT protocols and data formats. We remark that the concept of scalability in the context of smart city is one of the most challenging with respect to those in the context of IOT for Industry 4.0 and agriculture applications due to the huge number of data flows related to the inhabitants.

2.3.1 Federation of API Services

In the context of Smart Cities, not all cities/areas are becoming smart in the same manner [81]. In most cases, the cities are focused to a set of smart

services, for example: smart parking, smart education, smart gov, smart lighting, etc., according to their needs and strategies. In most cases, vertical applications have been implemented for years as separate pillars, and only recently there is push on integrating data and/or services to exploit higher level machine learning business intelligence tools and control room dashboards [46]. For Smart City applications, it is quite frequent the usage of Smart City API to provide services and data via Web and mobile applications. Examples of Smart City APIs are: Km4City API [33], E015 [139], and [99]. Others candidate clients could be control room Dashboards used by city operators, City Major and city Councilman [46]. Most of the solutions (Web or mobile) using the Smart City API are focused on a single city. This means that, passing from one city/area to another, the users have to change mobile application to get the same service. This happens for the lack of interoperability among Smart City API, SCAPI. They are not standardized, see the review of Smart City API on [117]. An alternative solution is offered by global services (such as Google) which sadly not covers local services using private data of the city. A large part of the services proposed via Smart City API are geolocalized and provide different results according to the user's ID assigned profile: organization, role level, past activity, preferences, etc. The User ID implies the management of Personal Data, and thus the GDPR has to be applied [37]. In the context of GIS (Geographic Information Systems) data exchange the federation of services is largely diffused [63]. Most of the GIS solution for federation are based on federating datastores. The classic GIS interoperability is limited to 1:1 exchange of geographical data for example exploiting protocols as WFS. In alternative, noSQL storages can be designed to support the smart city services as well, also supporting geolocated information. Among the noSQL storage solutions, the most suitable to manage geolocated information together with city entities relationships are those based on RDF model (Resource Description Framework) [78]. RDF stores can be federated at level of semantic model. On the other hand, federated RDF storages are far to be a good support for federated smart city services [112], since the concepts of federation has to be at service level, for example, for routing by taking into account different geographical areas addressed by different RDF stores.

2.4 Classification of RDF datasets through Multilayer Metrics

Linked Data guidelines have been adopted by an increasing number of data providers over the years, leading to the creation of a global data space containing billions of assertions, the Web of Data [55], [2], [54], distributed across a huge number of RDF datasets.

Many proposals can be found in the literature about descriptive metrics for RDF datasets. L. Ding, et al., in [71], in the attempt of building Swoogle, a Semantic Web Search and Metadata Engine, have characterized RDF datasets by: (i) character encoding; (ii) language; (iii) representation format; (iv) ontology ratio (i.e., the share of triples devoted to the ontology representation); (v) links among datasets (Semantic Web Documents); (vi) reuse and evolution of ontologies over the time. A. Langegger, et al., in [101], have proposed to count instances per class and to build histograms (per class, property, value type) to be used by a Semantic Web Integrator and Query Engine (SemWIQ) [100]. S. Campinas, et al., in [61], have built a RDF dataset, the Sindice-2011 Dataset, based on documents containing semi-structured data gathered from the Web since 2009, and have computed a set of metrics focusing on: (i) representation format; (ii) reuse of ontologies; (iii) anatomy of triples; (iv) volume of triples; (v) length of literals. S. Auer et al., in [28], have proposed a formal definition for a statistic applied to an RDF dataset, and a comprehensive set of metrics for quality, coverage, and privacy analysis. Also, they have proposed a generic RDF representation of statistics. Fabio Benedetti et al., in [52], have proposed a set of metrics suitable for drawing a RDF dataset overview, and for measuring interlinking of data. F. Fioravanti and P. Nesi, in [75], have proposed a set of Object-Oriented Metrics aimed at providing a prediction and/or an a-posteriori estimation of the complexity and subsequent cost/effort that is necessary for the developing and maintaining an Object-Oriented Software [56]. Some of them can be applied (with adjustments) to the estimation of the complexity, and to the estimation of the development and maintenance costs of a semantic data model. More specifically, these metrics have inspired our Predictive Model Complexity metric, described in Chapter 11. Other inspiring metrics that address the cohesion and other key aspects of Object-Oriented software solutions, e.g. those defined by Chidamber, et al., in [62], and by Bucci, et al., in [59], have led to the definition of two of our Endpoint Metrics: the

Weighted Number of Relatives (WNR), and the Coupling Between Resources (CBR). Vandenbussche, P. Y., et al., in [132], have proposed a system for a continuous real-time monitoring of availability, performance, interoperability and discoverability of SPARQL endpoints. While points of contacts can be found with our work, our proposal differentiates for the quantity and quality of both endpoint-level and global metrics (think of global rankings of class usages, or literal languages, for example), and even more clearly for the objectives of the work, since the main achievement of our work is the clustering of RDF datasets by structure/purpose, that is not among the goals of the research activity presented in [132]. Fernández, J. D., et al., in [74], have proposed and computed a set of new structural metrics for RDF datasets, with the objective of highlighting aspects of how information is represented that could be of help for designers and developers of data management systems. Therefore, while points of contacts can be found with our work, and their defined metrics could even be taken into consideration as endpoint-level metrics to be added to our set in future iterations/refinement of the clustering process, objectives are remarkably different, and one of the most evident implications of that, is that the performing of global considerations and measurements from local detections is not formally defined, updated at real-time, and made systematic in their work, as it is in the ours. Debattista, J., et al., in [68], have proposed a methodology, an operative metric definition language and a software system including a Web interface for the assessment of RDF datasets quality, that allows to build rankings of RDF datasets based on user-defined metrics, and on weights that the same users assign to each of their defined metrics. Mihindukulasooriya, N., et al., in [114], have defined a set of endpoint-level metrics that appears to be a subset of the endpoint-level metrics that are defined in this work, and make available a Web tool to allow users to compute and display values for such metrics for a given endpoint that is of interest for the user. Lorey, J., in [105], has proposed a set of metrics for the assessment of performances of SPARQL endpoints (latency, throughput, execution time of join and random-access queries).

Chapter 3

Knowledge Modeling and Management for Mobility and Transport Applications

In the context of smart city, real-time routing and traffic prediction applications have specific requirements as for their needed input data, shaping, and efficient retrieval. The recent applications and services need to have integrated data store for efficient data access and deduction. The models at the state of the art are not satisfactory. Therefore, we have developed a new ontology, namely the Km4City, in the context of the Sii-Mobility research and development project of Italian Ministry. It makes it possible, among several services, also the routing and traffic reconstruction. In this chapter, the model and the processes for knowledge base construction are described. The process takes into account Open Street Map, open data, and real-time data from sensors, mobility and transport operator in multiple formats and protocols, to load them into Km4City KB. A Traffic Reconstruction solution based on differential equations has been also presented. The data model and the solution developed with related algorithms are part of Km4City (www.km4city.org), and accessible for test and usage from <http://servicemap.km4city.org> and from <http://firenzetraffic.km4city.org> that covers the whole

*Tuscany, one of the largest regions in Italy.*¹

3.1 Introduction

Mobility and Transport is a very complex field for research data production and management. Many interrelated challenges must be faced for building an intelligent mobility and transportation system. Rich routing services for tourists, citizens, professional delivering, emergency services, that leverage real-time data and take into consideration both private and public methods of transport for computing the best possible route with respect to heterogeneous criteria submitting to dynamic constraints are a key component of an intelligent mobility and transportation systems. They include, for instance, point-to-point multimodal urban routing, green routing considering fuel consumption, gas stations and recharge points network topology, routing solutions for emergencies (fire brigade, first aid), and waste collection. GPS-equipped devices, and the traditional (static) traffic sensors deployed on street in many urban agglomerates can be used to assess, reconstruct, predict the traffic flows in urban environments. Key tasks in this context are: traffic flow reconstruction from the fixed traffic sensors without engaging third parties or somewhat relying on their sensing devices; prediction of the final destination of a (flow of) vehicle based on origin, vehicle characteristics, date and time, and other variable criteria; prediction of the number of vehicles that will pass in a specific time interval (traffic flow prediction); reconstruction of the path of a specific vehicle; air quality prediction; detection of traffic anomalies using taxi GPS data; traffic light configuration, etc. In addition to the above issues also those enabled by the possibility of tracking the vehicles by using on board devices (OBD) boxes that record the position, the acceleration, and in some cases the video of what is going on around the vehicle. These solutions allow to reconstruct the vehicle behavior on, and have an enormous value for insurance, police department, and law, opening the path for integrated autonomous vehicle management. Connecting vehicles, each other and to the surrounding infrastructures, is also a key challenge that could be useful to minimize queues, suggest trajectories, avoid collisions, etc.

¹This chapter has been published as “Knowledge Modeling and Management for Mobility and Transport Applications” in *IEEE TeC4C’18, 1st International Workshop on Technology Convergence for Smart Cities, Philadelphia, PA, USA* [42].

Understanding the user behavior is also a very relevant value when estimated on single travelers, to provide suggestion and assistant. The problem of identifying the locations, services, events that could be of interest for a citizen or tourist based on his previous visits to similar points of interest (POI), trips, social activities, current position, expected movements, and a wide set of further parameters, should also be addressed by an intelligent mobility and transportation system. As a result, for that further steps forward could be done in the provisioning of real-time context-aware mobility-related applications for citizens, tourists, shoppers, students, workers that populate our modern cities, data models are needed that are general enough to model the urban environment in its whole, deep enough to model all the information that the state-of-the-art solutions need, and real-time.

It is easily understood that connected drive, rich routing, predictions and reconstructions of the traffic flows, identification and recommendation of the POIs, all are faces of the same medal. For almost all of the above mentioned applications, the availability of a detailed city data model and representation is mandatory. The routing algorithms need to know the detailed information of each street segment and cross; the routing algorithm should take into account about the present and predicted traffic flow in each street segment; the multimodal routing should know the user behavior which in turn should know where and when each public means will pass; suggestions and assistant hints to the users should take into account the city structure not only in terms of street and public transportation services, but taking into account about all possible POI and services in the city.

As described in the next section, the solution at the state of the art for modeling city knowledge are not suitable to fully support the above mentioned algorithms and applications. This chapter is focused on presenting a data model that supports these aspects and provides the evidence of its functioning with real cases. The presented work has been realized in the context of Sii-Mobility (the national project of the Italian Ministry smart city mobility and transport), <http://www.sii-mobility.org>. The research and development project was aimed at developing a big data infrastructure for mobility and transport, sustainable mobility, connected drive, smart parking, smart traffic, routing, etc. To this end, this chapter is structured as follows. Section 3.2 discusses the identified main requirements for smart city mobility and transport. In Section 3.3, the Km4City architecture is outlined. In Section 3.4 the Km4City data model is described

(see also <http://www.km4city.org>). In Section 3.5, the Km4City traffic reconstruction solution is described that is running at today in Florence (<http://firenzetraffic.km4city.org>). In Section 3.6, a glimpse on the volume of data that are managed in Km4City is provided. Conclusions are drawn in Section 3.7. A review of the main existing data models that encompass concepts related to mobility and transport can be found instead in Section 2.1 of Chapter 2 .

3.2 Requirements for Mobility and Transport Applications

In this Section, the main requirements that a knowledge base should have to cope with the above mentioned mobility and transport applications are presented. It has to provide support for modeling:

- geolocated points that outline the roads' paths, indicating their identifiers and coordinates;
- road segments outlined by geolocated points, have to provide: a unique identifier; information associated as traffic direction(s), length, relevance of the road, and segment type (such as if it locates within a parking, link, roundabout, crossroad, toll booth, square, pedestrian area, railway level crossing, rest area, and so on); time needed for traversing them (for each time slot of the day and the week) computed through a traffic flow reconstruction;
- roads where the above road segments are located, and for each road a unique identifier and a list of contained elements (e.g., segments, street civic numbers, services, cycle paths, etc.);
- lanes indication for the road or segment where they are drawn, the traffic direction, the count with separated indication of the reserved lanes;
- traffic restrictions for expressing the mandatory maneuvers at the crossroads, the prohibition for some vehicles to traverse a road or lane, the speed limits, and similar;

- cross roads with their rules and red light semaphores time cycles, supporting or not the connected drive information about the change of status (for example received in ETSI format);
- traffic events: which can be planned, e.g. ordinances, or unexpected, e.g. accidents. In both cases, the roads/segments may change its functionality level, thus routing has to change as well;
- traffic signals including traditional signage, variable message panel, dynamic signals;
- gas stations, indicating their position, the available fuel, the status and the prices for products;
- public transport including path changes, stops, time schedule for each ride, price, for example collected from GTFS data;
- car park positions and status in real time, for both street parking and silos parking (for example, received in DATEX format);
- parkings may also include areas in which delivering services can change their distribution vehicle, for example passing from truck to bike;
- restricted traffic zones, with their shape, gates for entering, their rules, time schedule, number of passages, etc.;
- bike and car sharing, with rack position and status (for some kind of bike sharing), or location of vehicles for floating solutions, which in practical make car and bike sharing more similar each other;
- POIs, Point of Interests, representing any kind of service (gov-services, commercial, cultural, health, etc.) in the city, with their properties for a comprehensive naming and labelling, including alternative (possibly multilanguage) names and labels, so that effective full text searches could be performed including their geolocation.

In an integrated knowledge model for mobility and traffic, it is very important to reach the information browsing among the city entities and to be capable to have in each point and from each city entity links to the connected and near elements, since they enrich the context, and therefore open to spatial and temporal drill down functionalities. For example, for parking prediction one may need to know the traffic of the zone, for routing it is

important to know the time to traverse each street segment for estimating the fastest path, for identifying the user behavior it is important to know its velocity and acceleration (that can be obtained from the mobile), but if you also know the path of the busses you have an increased probability of understanding if a person is moving by bus or car.

The above mentioned information is not typically available from a unique source. For example the city structural information may be collected from city archive, or regional archive, or from OSM. The OSM data models do not include concepts for modelling traffic sensors, and do not suit the needs of real-time routing and traffic reconstruction solutions. Moreover, OSM does not provide the information for its direct usage for routing and traffic reconstruction, or suggestion. The OSM data has to be processed to arrange data to be easily accessible efficiently as explained in the following. All the other information can be collected from multiple sources [33]: open data of the city from the municipality such as the real time data as traffic flow sensors, parking status from specialized mobility operators, public transport plan and status from the mobility operator, etc.

3.3 Km4City Extended Model

Before presenting the process for knowledge base construction, in this section, the enriched Km4City model is presented, with a deepening on the road transport infrastructure that is the foundation of the above mentioned mobility and transport applications.

In the Km4City the Regions, Provinces, Municipalities, and Hamlets (that also model the city districts) are represented. Within each Municipality, a set of Roads can be found, each partitioned into a set of small linear segments that outline its path, the RoadElements. The StreetNumbers are also modelled and linked to the Roads and RoadElements where they locate, and to the Entries that model the geolocated entrance doors where the street numbers are affixed. The number of lanes that are drawn on the asphalt, their traffic directions and possible restrictions are also modelled through dedicated concepts such as Lanes, LanesCount, Lane. A wide set of Restrictions, applied to entire roads or parts of them, is also represented, and it integrates the temporary modifications to the viability. In Figure 3.1, the main concepts and relations that can be found in the Km4City street graph data model are depicted.

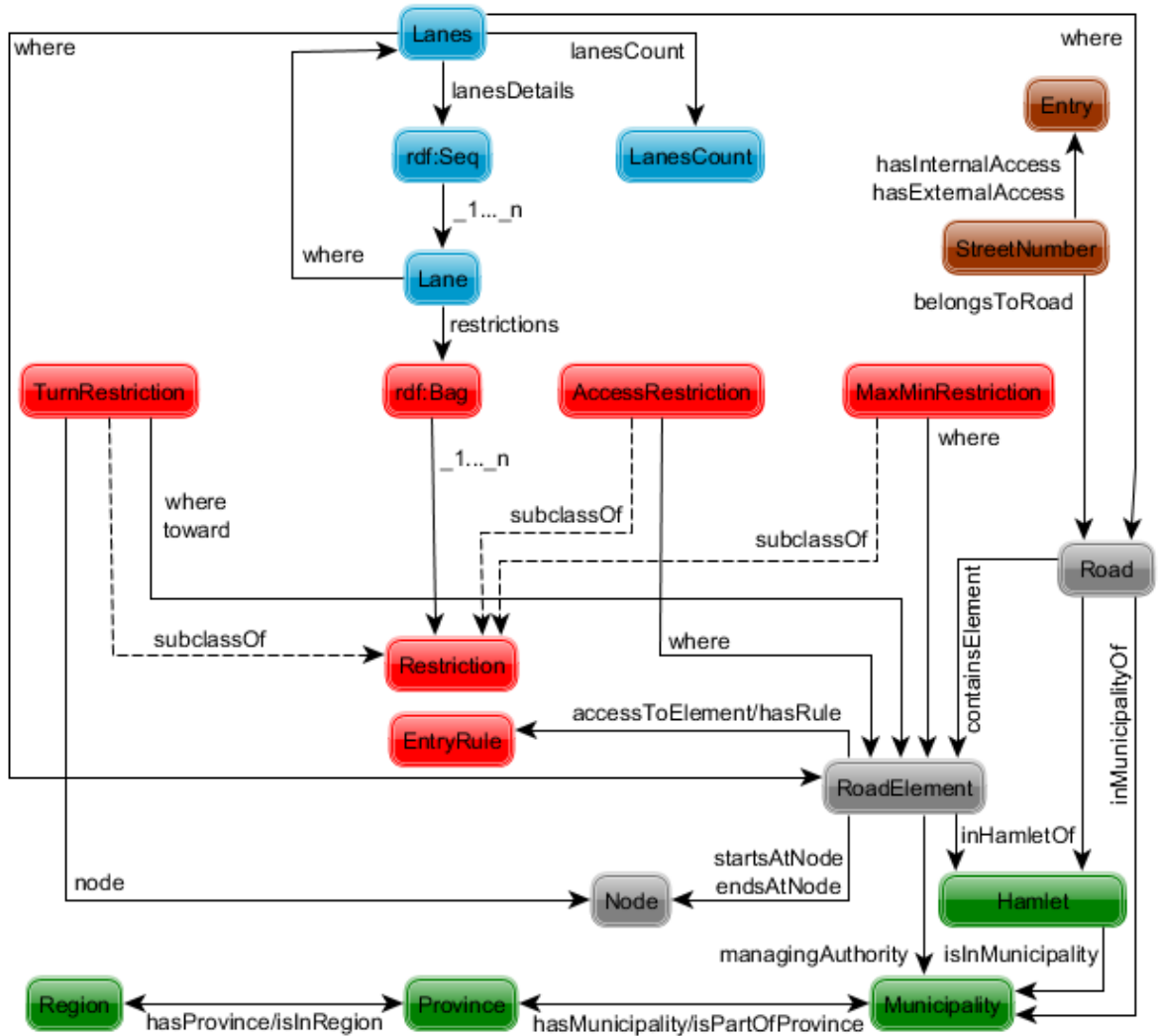


Figure 3.1: The Km4City street graph data model: an overview

3.3.1 Roads

Roads are represented within the Km4City data model through the Road concept. The name of the road is modelled through four different properties: `roadType`, the generic name of the road; `roadName`, the distinguishing name of the road; `extendName`, the full name of the road, resulting from the concatenation of generic and distinguishing name; `alternative`, an alternative name for the road. This hierarchical structure has been adopted for getting high efficiency and effectiveness of the full text searches. Also, the separate indication of the generic name of the road, enables a road categorization. This is why it is called `roadType`. The road path is outlined by (typically and relatively) short linear segments that join two consecutive nodes in the road path, the `RoadElements`, that are connected to the Road through the `containsElement` property. The `RoadElements` junctions are modelled as geolocated `Nodes` and are connected to the `RoadElements` through the `startsAtNode` and `endsAtNode` properties.

3.3.2 Roads partitioning

Road elements are contextualized, and the context is provided through the `elementType` property. It has a key role in routing and traffic reconstruction, e.g., roundabout segments always are one-way and must be traversed counterclockwise. Road elements have a class, e.g., motorway, primary, secondary, etc. modelled through the property `elementClass`, which allows inferring restrictions and architectural features. The length, width, operating status (under construction, in use, or abandoned), and traffic direction (positive, negative, both, or none) are also modelled through dedicated properties that are crucial for routing applications and traffic predictions.

3.3.3 Lanes

The lanes have a key role for routing applications, that exploit them for suggesting the lane to traverse based on the type of vehicle, the lane restrictions, and the planned route. Traffic reconstruction applications also use them to assess the road capability. The heterogeneity of the situations imposes an as flexible as possible modelling. The Lanes concept models a set of lanes. These properties are defined for it: `where`, the Road or `RoadElement` where the lanes are; `direction`, traffic direction; `lanesCount`, the number of the lanes (see below); `lanesDetails`, possibly filled with a RDF Seq of Lane

instances sorted left to right in the driving direction, through which possible restrictions are modelled (see below). The `lanesCount` property is filled by a `LanesCount` instance where the `undesigned` property is typically set and filled with the count of the lanes that can be traversed by all types of vehicle. Other properties can be set that take their names from types of vehicle and that are filled with the number of lanes reserved to the specific type of vehicles. These main properties are set for a `Lane`: `turn`, indicates the mandatory maneuver at the end of the lane; `restrictions`, possibly filled with a RDF Bag of `Restriction` instances.

3.3.4 Restrictions

For that routing and traffic reconstruction services could be offered, traffic restrictions need to be modelled. We have identified three macro categories: mandatory maneuvers at the crossroads; access regulations that apply to roads and segments of road; maximum/minimum speeds, weights, sizes, and similar. In real-world, restrictions can be found that apply to just a traffic direction, or to a subset of the vehicle categories, weekdays, and daytimes. An example of the model directly taken from the online linked data model can be accessed from <http://log.disit.org/service/?graph=9cf1ec16fbb429ae55a5597ee028afe> by using the <http://log.disit.org> tool of DISIT lab [50], while the counterpart on Map can be accessed via <https://servicemap.disit.org/WebAppGrafo/api/v1/?queryId=121fe068e5b521d3b7e303format=html>, see Figure 3.2. As depicted, the passage from map to model is performed by following the Linked Open graph link.

As a result, the above concepts have been shaped for modelling the traffic restrictions in the Km4City Ontology:

- **TurnRestriction**, through which the mandatory maneuvers at crossroads are modelled, with these properties: `where`, the source segment of road; `toward`, the destination segment of road; `restriction`, i.e. if the maneuver is mandatory, or forbidden; `day_on`, `day_off`, `hour_on`, `hour_off`, for modelling applicability limitations to specific weekdays and daytimes; `except`, a possible list of exempted vehicle categories;
- **AccessRestriction**, through which the access regulations for roads and segments of road are modelled, with these properties: `where`, the `Road` or `RoadElement`; `who`, the possible vehicle category; `direction`, the possible traffic direction; `access`, i.e., if the restriction represents a prohi-

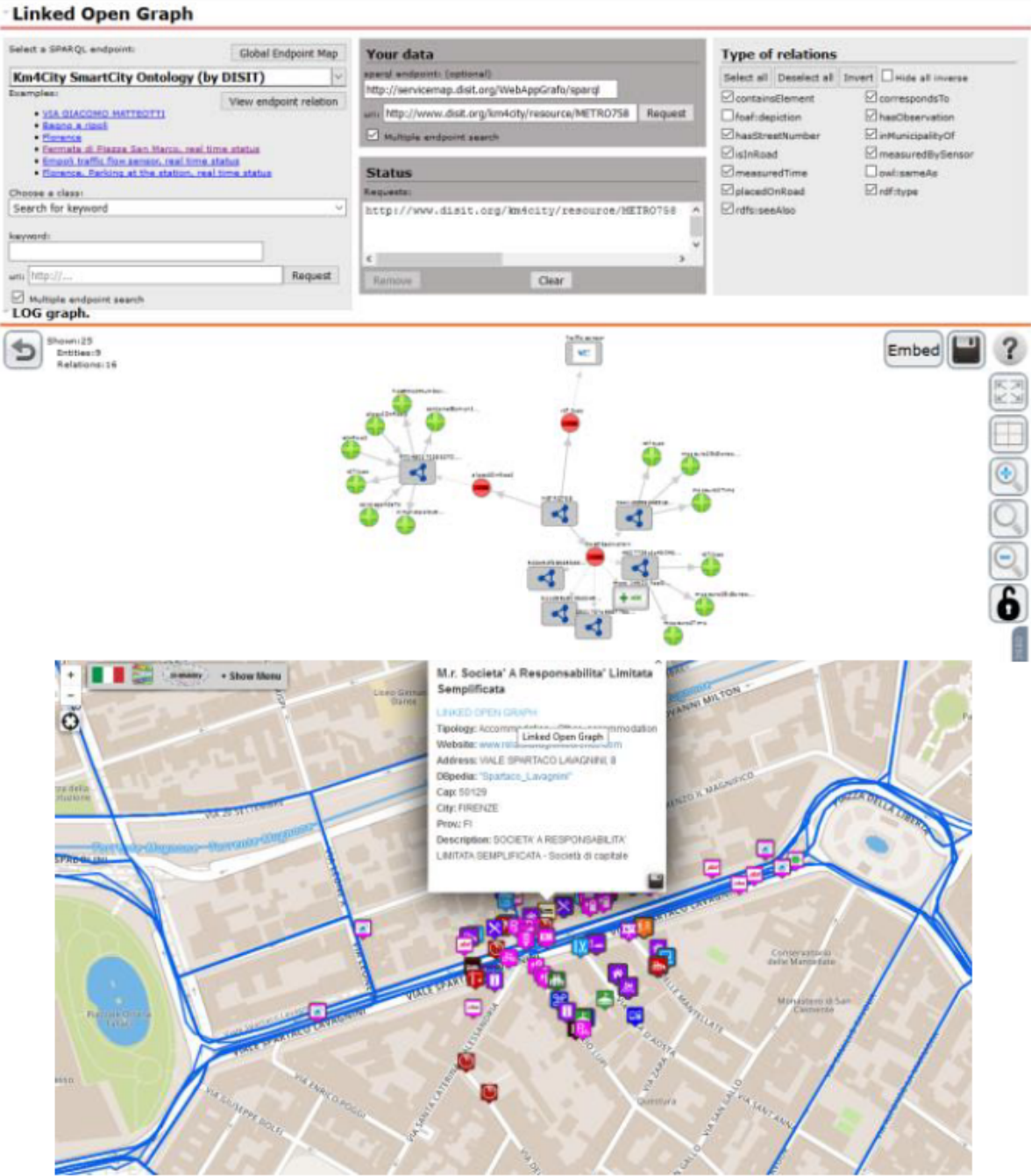


Figure 3.2: Above) Knowledge base access as linked data via LOG, Below) map view access via ServiceMap.km4city.org

bition or an exclusive permission; condition, the possible applicability conditions;

- **MaxMinRestriction**, that models the limitations that are expressed in the form of maximum or minimum speed, weight, width, length, axle load, and similar, with these properties: where, the Road or RoadElement; what, i.e. whether it is a max speed, min speed, max weight, or what else; limit, the limit value; direction, possible traffic direction; condition, possible applicability conditions.

3.4 Knowledge Base Construction

As above mentioned, the data sources exploited for populating the Km4City Knowledge Base, KB, are: OSM, from where the structural information related to the street graph are gathered; the Open Data that are made available by the Public Administrations of the territories that are at today covered by the project, where both the infrastructural and realtime information about the public transport are contained, in the form of GTFS feeds; data coming from sensing devices deployed within the territories of interest in a IoT perspective, such as traffic sensors, ice detectors, smart benches detectors, smart waste detectors, smart beacons (street lights). OSM data is ingested through a process that is made up of the following stages: (i) extraction of OSM file into a relational database; (ii) data transformation through SQL scripts; (iii) loading of the relational data to the KB as RDF triples, through a triplification tool named Sparqlify. ETL jobs and transformations are instead executed within an ETL engine, for extracting the GTFS, traffic, and other real-time data, transforming it, and loading it to the KB or to a separated database for an improved efficiency. Sensing devices are represented in the KB, and brokers are also represented that describe how to access the real-time raw data produced by a specific sensing device reading directly from the sensing device or from the observations stored in the relational database. Real-time traffic flow sensors are needed to identify, and predict, traffic congestions for multimodal routing and traffic flow prediction/reconstruction. Also mandatory is the reconciliation of the traffic sensors with the nodes of the street graph, so that a contextualization could be provided to the sensors' observations and their integration in the city graph. Moreover, the real-time data about the position of buses and similar should be modelled to provide rich applications.

3.4.1 The roads infrastructure ingestion: motivations

Data about roads, infrastructures, restrictions, and so on, have been proved to be not efficient enough for the purposes of a real-time application. Indeed, OSM provides (possibly compressed) huge XML files. Therefore, storing them in memory is not a choice. On the other hand, one single sequential scan is not enough for retrieving all the information that is necessary for routing and traffic reconstruction and performing a set of sequential scans for retrieving each single piece of data is not a choice since it is not efficient enough. Also, geographical queries cannot be efficiently executed on XMLs, and the data model of the Open Street Map is not as suitable as the Km4City Ontology for the purposes of routing and traffic reconstruction. An offline ingestion of the Open Street Map is therefore necessary. The ingestion is performed populating a relational database with dedicated extensions for geographical queries and generating then RDF triples starting from the data stored in the relational database. In the process, OSM data is also filtered on regions of interest, and an RDF graph is generated for each region. This way, efficiency is furtherly improved. Also, not all properties that are available on the Open Street Map are of interest for our purposes (think of the datetimes of creation or modification, the author of the modification, and other features that relate to roads or nodes that are not exploited in our algorithms at today). Therefore, not only a vertical (geographical) cut is performed in the data transformation, but also a horizontal cut. Some integrations have also been necessary on the Km4City ontology to model those data that were not of interest (and were not stored therefore) before the introduction of routing and traffic reconstruction, and are instead of interest now, such as (i) the lanes (with their counting, features, and restrictions), (ii) the mandatory manoeuvres at the crossroads, (iii) the access rules for roads and segments of road, the speed limits, and other prescriptions, (iv) two-way roads represented as two separated one-way roads, one for each traffic direction, and (v) artful traffic restrictions for preventing U-turns at crossroads.

3.5 The Km4City Traffic Reconstruction

For the purposes of the traffic reconstruction, a directed graph is built from the road infrastructure that is stored in the Km4City KB: each two-way road is represented through two edges. Moreover, U-turns are considered to be prohibited at crossroads, and the TurnRestriction instances are also

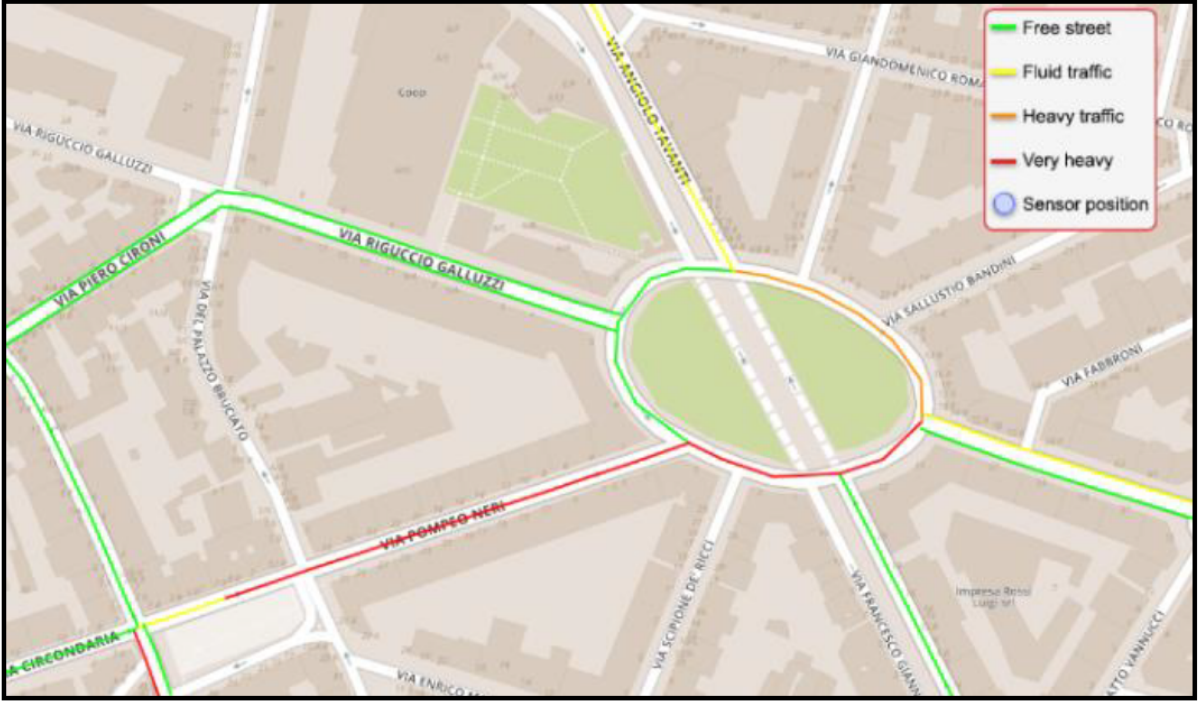


Figure 3.3: Real-time traffic flow reconstruction in Florence

leveraged. Of course, such a representation plays a crucial role to understand the vehicle dynamics in a network and an appropriated depiction tool is also required in order to effectively display the separated traffic directions on a map (Figure 3.3 shows ways opened in both directions in the area of Florence).

In this section, the Km4City graphical low-cost (i.e. that only leverages the existing fixed traffic sensors) application is presented that reconstructs the traffic flow at every location of the wide urban area of Florence based on a self-adaptive mathematical model inspired to the fluid dynamic (deeply discussed in the following). The model is anyway general enough to be applied to any territory.

3.5.1 Real-time density reconstruction: mathematical model

The proposed reconstruction algorithm predicts the traffic flow in real-time at every 20 meters long segment of road from scattered data and it displays the predicted values through coloured lines (which symbolize the state of traffic) traced on a map, along the road paths. The mentioned algorithm is based on a mathematical model for fluid dynamic flows on networks. The city graph is composed by a finite number of RoadElements that meet at some Nodes (junctions). Starting from traffic flow data sensors integrated in Km4city data model, we have developed a reconstruction algorithm to detect the state of traffic in the unmonitored roads. Each sensor gives the state of the traffic in a fixed position counting the number of vehicles which pass through the supervised area, and all the data are simultaneously uploaded with a time slot of 10 minutes. Associating a sensor to a node in the graph, the sensor data can be interpreted as a “source of traffic” which propagates itself on its neighbouring roads. Modelling the dynamics of such a propagation, we reconstruct the state of the traffic where sensors are not present. Of course, the computational complexity of the present model primarily depends on the dimension of the considered city graph and the traffic data update (every 10 minutes) does not exceed the running time of the tool.

Here we deal with the macroscopic model proposed in [58] and it allows observing the network in the time evolution through waves formation. The fluid dynamics concepts are applied to traffic situation for the first time in [104], and independently in [123] with limited scope. In a single road, this nonlinear model is based on the conservation of cars described by the following scalar hyperbolic conservation law

$$\frac{\partial \rho(t, x)}{\partial t} + \frac{\partial f(\rho(t, x))}{\partial x} = 0 \quad (3.1)$$

with: boundary conditions $\rho(t, a) = \rho_a(t)$, $\rho(t, b) = \rho_b(t)$, and initial values $\rho(0, x) = \rho_0(x)$. In particular $\rho(t, x)$ denotes the vehicular density and the function $f(\rho(t, x))$ is the vehicular flux which is defined as the product $\rho(t, x)v(t, x)$ where $v(t, x)$ is the local speed of the vehicles. The solution of the above equation can be obtained as finite differences in [58], bringing to a numerical solution of partial differential equations [93], [115]. The first step to perform the computation is to discretize time and space by means of a time-space grid (see [66], [67]). A RoadElement can be partitioned into

a series of segments having step size $\Delta x = 20m$, and similarly, the time is divided into a series of durations with step size Δt . The time-space region bounded within duration h and segment m is referred to as a cell and it is denoted as (h, m) and the number of vehicles contained in segment m at the end of duration h is denoted as $n(t_h, x_m) = n(h\Delta t, m\Delta x)$. By using the notation u_m^h for $u(t_h, x_m)$ when u is a continuous function on the (t, x) plane and considering u as an exact solution of (1), we assume the following discretization by means of the finite difference scheme for the evolution of the vehicular density in each RoadElement:

$$u_m^{h+1} = u_m^h - \frac{\Delta t}{\Delta x} (F(u_m^h, u_{m+1}^h) - F(u_{m-1}^h, u_m^h)) \quad (3.2)$$

where F denotes the numerical flux. To setup the above numerical approximation in the whole graph we have to opportunely impose both boundary conditions and conditions at the junctions in (2). Moreover, given t in the daily time, for the traffic distribution on each junction we estimate a matrix $A(t) = a_{ij, j=n+1, \dots, n+m, i=1, \dots, n}$ such that $0 < a_{ji} < 1$ and $\sum_{j=n+1}^{n+m} a_{ji} = 1$, for $i = 1, \dots, n$ and $j = n+1, \dots, n+m$, where a_{ji} is the percentage of drivers arriving from the i -th incoming road that take the j -th outgoing road. Given a junction, the main idea is to define a numeric value related to elementClass and lanesCount properties (see Section 3.3.2) associated to the crossing ways such that they assume a weight in the calculus of $a_{ji}(t)$.

3.5.2 Real-time density reconstruction: representation model

A city network having N roads which are constituted by $M_1, \dots, M_N$ RoadElements, respectively, can be represented by means of a data table D having $(M_1 + \dots + M_N)$ rows. Each row represents a RoadElement in the graph and it contains the associated characteristics. Consequently, a node J where n incoming and m outgoing RoadElements unite, is implicitly represented by means of the corresponding sub-table D' of D with $n+m$ rows. In particular, the incoming RoadElements are represented by the n rows of D' having J as their endsAtNode, while the outgoing RoadElements are the m rows of D' having J as their startsAtNode. Since the traffic network is represented by means of a directed graph, each road opened in both directions, has been represented by means of two distinct RoadElements having opposite directions. More precisely, each RoadElement e opened in both directions, such

Initial node	Road-segment URI	End node
node A	s	node J
node J	s/INV	node A
node J	s'	node B
node B	s'/INV	node J

Table 3.1: A node's representation in the case of roads opened in both directions

that it is placed between the nodes A and A' , is represented in the graph by means of a RoadElement e (denoting it by its URI) running from A to A' and a RoadElement e/INV (denoting it by its URI followed by the suffix “/INV”) running from A' to A where startsAtNode and endsAtNode are inverted respect to e . Moreover, if a junction J on the network meets two different RoadElements e and e' opened in both directions placed between the nodes A and J , and the nodes J and B , respectively, then from the previous description a such junction is represented by means of a table as shown in Table 3.1.

In this scenario, the percentage of drivers arriving from the RoadElement e (or e') cannot turn in the RoadElement e/INV (or e'/INV) in order to avoid a U-turn and then the corresponding value in the associated traffic distribution $a_{ij}(t)$ is equals to 0. The value 0 is also applied when a turn in an outcoming road is not permitted in a junction according to the turnRestriction relation.

Often, in a road opened in both directions a pair of corresponding traffic sensors is deployed, one per direction. In this case, we map the two sensors to two distinct nodes, possibly adding an artful node. For the sake of clearness, we illustrate our approach through an example. Considering Table 3.1, we suppose that two sensors, labelled with *Sens1* and *Sens2*, control the vehicles in the direction of the RoadElement e and e/INV , respectively. The idea consists into increasing the number of the nodes in the graph by adding a new node in the network. A such node can be denoted by the suffix “BIS” and the representation in Table 3.1 is improved as shown in Table 3.2 where each traffic sensor is univocally mapped to a node. Roughly speaking, the data sensor integration problem is solved by associating the measured density data to the RoadElement having such a node as its startsAtNode.

Finally, the graphical representation of a given RoadElement on a map is

startsAtNode	RoadElement URI	endsAtNode
node A	e	node J
node JBIS	e/INV	node A
node J	e'	node B
node B	e'/INV	node JBIS
node J	Sens1	node J
node JBIS	Sens2	node JBIS

Table 3.2: A representation of a couple of sensors located in the same point in the network

essentially constituted by the straight line connecting its startsAtNode and endsAtNode coordinates, denoted as (x_S, y_S) , and (x_E, y_E) respectively.

Then, a graphical representation of a RoadElement opened in both directions is made up of two parallel lines, one for each direction. Given a such RoadElement, we are going to consider the equation q of the straight-line passing from the startsAtNode (or endsAtNode) coordinate such that it results perpendicular to the line passing from (x_S, y_S) to (x_E, y_E) . Then, in q we find two points having the same distance from the startsAtNode (or endsAtNode) coordinate. Such points constitute the coordinates of the parallel lines. According to the slope of the line it is possible to deduce the right vehicle trip and we opportunely associate each line to the desired direction.

The validation approach of the model is based on the computation of the tool excluding the data from a selected traffic sensor and we estimate the error between the predicted value in the sensor location respect to the actual data. The accuracy of the whole model is estimated by considering the same procedure for each data sensor leading to a global accuracy close to 75% during a real-time performance.

3.6 Knowledge Base Experimenting

The described algorithm is tested in a rectangular area of the metropolitan network of Florence delimited by the coordinates (11.190046; 43.756291) and (11.288442; 43.816788), representing its left-down and right-up corners, respectively. The considered city graph is constituted by 736 nodes (or junctions), 1273 roads and 60 data sensors. During each time slot (10 minutes) 10729 data containing the predicted vehicular density in road segment hav-

ing length 20 meters are stored. Applying the same reconstruction algorithm to the whole Tuscany region consisting of about 10000 streets' kilometers we expect to consider 500000 data stored every 10 minutes, obtaining big data in terms of predicted vehicular densities in Tuscany region, every day.

The Km4City KB for Italy, to which all of the following statistics refer to, contains at today, considering both structural and real-time data, with all their properties, over 300 million triples. The extract of the OSM for Italy, one of the data sources that we leverage at today for the road infrastructure (the others are the extracts for Belgium and Finland), contains about 160 million nodes, 17 million ways, 260 thousand relations, and 50 million tags. In the Km4City KB for Italy are stored at today 834 municipalities (with their 295 districts) spread across 20 provinces including 4 regional capitals, i.e. Venice, Bologna, Florence, Cagliari, spread across five regions, with about 190 thousand kilometers of roads distributed on 560 thousand roads portioned in 6.3 million segments.

As for the railway transport, 50 railway lines can be found in the Km4City KB at today, partitioned in 86 sections and over 3.5 thousand elements, which connect 641 stations and 18 good yards. As for the urban public transport, 86 lines can be found in the Km4City KB, over which 411 routes are defined that are partitioned into 9744 sections that connect over 1.5 thousand bus stops. Also, over five million GTFS stop times and over 173.000 AVM records are stored at today in the Km4City KB. As for the bike mobility and multimodal transport, 39 monitored bike sharing racks with about 250.000 status reports. As for the real-time data related to the private mobility, 83 car parks are monitored, and over half a million status reports are stored at today in the Km4City KB. As for the real-time data related to the air quality, pollen, traffic flows, weather, in the Km4City KB can be found at today 47 air quality stations with over five hundred observations, 5 pollen monitoring stations with over 5.000 pollen observations, 751 traffic sensors with over 1.6 million observations, over 75.000 weather predictions, and about 4.700 weather reports. Also, about two thousand events and over 180.000 services are stored at today in the Km4City KB.

3.7 Conclusions

In this chapter, a unified model addressing information for mobility and transport applications for smart city has been proposed. In this chapter we

presented the identified requirements and the state of the art of the possible solution. In order to satisfy the requirements a new model has been developed by starting from Km4City model. In this chapter, the model identified and process for the construction of the knowledge base have been described. The process of Km4City KB construction takes into account: Open Street Map, open data, and real-time data from sensors, mobility and transport operator in multiple formats and protocols, to load them into a graph database. The solution proposed has been shown to work by presenting the Km4City Traffic Reconstruction solution based on differential equations. The solution developed and related algorithms are part of Km4City (www.km4city.org), and made accessible for test and usage from <http://servicemap.km4city.org> and from <http://firenzetraffic.km4city.org> that covers the whole Tuscany, one of the largest regions in Italy. In end, relevant statistics and considerations related to Tuscany and other covered areas have also been reported.

Chapter 4

Snap4City: A Scalable IOT/IOE Platform for Developing Smart City Applications

Smart City solutions, initially started with open data, are evolving towards data aggregation and semantics. Recently, some of them are also offering IOT support. The combination of IOT and smart city is not an easy task, the data volumes are much higher than those addressed for industrial IOT. The complexity of IOT smart city solutions have been identified by a number of actors. The European commission started to set up the EIP project for stimulating and concerting actions. The Select4Cities project of the European Commission and associated community <http://www.select4cities.eu/> created a challenge to find research solutions satisfying a formalized set of functional and nonfunctional requirements. Snap4City presented in this chapter is one of the solutions developed in response to that challenge. The proposed solution offers a platform where sophisticated IOT applications for controlling city dashboards as well as IOT mobile applications can be developed in few steps. Moreover, a number of development and monitoring tools have been developed. Among them, in this chapter, a special attention is given to the tools

and solutions for monitoring communication performance and to perform the assessment of scalability.^{1 2 3}

4.1 Introduction

The world of ICT solutions for smart city is very wide encompassing services at different levels of complexity and coverage, from classical Open Data portals (CKAN [7], Open- DataSoft [20], ArcGIS and OpenData [1]) to sophisticated solutions that provide data aggregation and Smart City APIs facilitating the development of Web and mobile applications (Km4City [120], City SDK [7]). The former solutions are mainly suitable for collecting and sharing open data files and their indexing on the basis of corresponding descriptive metadata. Open data, in those cases, can be uploaded by providing files in different formats: CSV, XLS, XML, SHP, etc. On the other hand, the latter solutions based on data aggregation and Smart City APIs try to create uniform data models describing the city entities and are focused on few domains (e.g. mobility and transport, culture and information, collection of feedback from the city users, e-health). In more detail, most of the Smart City solutions provide to some extent: open data management, control room dashboards, data analytics, big data stores, data mining and data warehouse at the data ingestion, smart city API. Only few of them offer development tools for data analytics and application development [33].

The effectiveness of smart city services is enabled by the availability of solutions capable to combine public (open) and private data owned and managed by City Operators addressing specific domains (e.g. transport, mobility, energy, health, tourist operators, culture and events). These city stakeholders provide data and services of different volumes and at different time and space granularities. For example, in the city, we can have few energy operators with capillary house distribution and data collection, many public transport operators with thousands of vehicles/buses (millions of travels per

¹This chapter has been published as “Snap4City: A Scalable IOT/IOE Platform for Developing Smart City Applications” in *Int. Conf. IEEE Smart City Innovation, Cina 2018, IEEE Press* [31].

²*Acknowledgments:* Our thanks goes to the Select4Cities Consortium for supporting our work with useful feedbacks about the real needs of smart cities like Antwerp, Copenhagen, and Helsinki. We also thanks to Km4City for the usage of data for testing and validation in the early phases.

³Select4Cities Consortium and the European Commission with Grant Agreement No:688196

year), some telecom operators with a very large number of mobile devices and millions of calls per year, and IOT networks of tens to hundred thousand of sensors with several millions of messages per year. Different granularity implies different methods for collecting and for providing access to data such as publication of open data files and/or statistics, real-time publication of data updates in several different protocols for different purposes.

The IOT solutions arrived in the context of Smart City with new solutions deploying sensors/actuators with a set of protocols directly enforced in the devices, such as: OneM2M [88], ETSI M2M [87], MQTT [16], COAP, SigFox, AMQP, LoraWAN, Green Button Connect [11]. The IOT revolution has also opened the path for the integration of personal IOT devices and their exploitation in the context of personal applications; for example, for adding specific house temperature sensors and actuators, for managing personal health care sensors (e.g., glycosometer), training devices. Furthermore, the exploitation of contextual data coming from the city for taking personal decision is also very relevant. In most cases this issue is neglected and fully in charge of the single users. As a result, IOT solutions for smart city, providing support for the final users in exploiting city and personal data in the context of personal IOT applications, are really a few. See for example classical IBM, Google or AWS solutions that do not provide specific contextual and local data but facilitate the usage of your personal data only.

Considering the general scenario of IOT for Smart City, Select4Cities EC project and associated community (<http://www.select4cities.eu/>) launched a challenge as a competitive PCP (Pre-Commercial Procurement) based on research activity according to the definition of EC. The challenge aimed to look for platforms that could cover a wide range of requirements at the same time in the context of IOT/IOE (internet of thing/everything) and smart city applications, with the aim of experimenting them on different cities in Europe such as Helsinki, Copenhagen and Antwerp. Among the identified requirements, there are the capability of the IOT/IOE solutions: to serve as a City Dashboard (this might mean to have centralized unique control room or to have several control rooms for each domain); to serve as an open city platform at service of different operators; to cope with real-time communications of different kinds and formats; to manage data referential; and, to be capable to support the creation of a Living Lab enabling the city users and city stakeholders to develop their own applications by using different and suitable tools, and to put them at the disposal of the city. A

set of challenges has been proposed by EIP European commission actions for Urban Platforms [23], mainly on clean power for vehicles, multimodality routing, smart logistic, sustainable mobility, etc.

In response to the above-mentioned research and development challenge, we have constituted the Snap4City team (<http://www.snap4city.org>) combining two research teams from the University of Firenze and University of Milano. The team has worked since the beginning of the 2017 for setting up the Snap4City platform. It is an open source IOT/IOE platform satisfying the above-mentioned requirements and a number of non-functional requirements regarding: open sources, scalability, standard compliance, robustness, distributed, managing heterogeneous communications, interoperability, security and privacy respect. At the present development stage, the first version of the Snap4City platform has been deployed and validated against several challenging cases as described in the rest of the chapter, and it has been tested twice by the Select4Cities team. The solution proposed started from the exploitation of Km4City platform and its smart city API (<http://www.km4city.org>) [33], adding IOT/IOE capabilities and scalable management.

This chapter is structured as follows. Section 4.2 presents the main requirements for IOT smart city solution and the proposed architecture. Section 4.3 provides an example of what can be realized by using the Snap4City solution taking into account a mobility and transport scenario for infomobility. In Section 4.4 two scenarios of performance analysis for the proposed solution are presented. One is for the assessment of communications and the other one is for general scalability and workload. Conclusions are drawn in Section 4.5. A comprehensive discussion of related works can be found instead in Section 2.3 of Chapter 2. .

4.2 Snap4City Architecture

The main architecture of Snap4City consists of a set components developed for:

- Collecting data from open data, real-time data, and IOT that are produced by different sources, city operators, and also by the users devices, social media.
- Storing and managing data in a knowledge base (KB) and tabular

noSQL storages, and indexing them for data retrieval with inference, spatial-temporal reasoning, facet search, and drill down on time and space.

- Creating IOT applications, data driven and/or periodic, based on MicroServices [118]. The IOT Applications can be data flows extending NodeRED of IBM, which may also exploit Spark and Kafka capability, and personalized data analytics.
- Creating services/processes by means of a number of easily accessible tools for developing data analytics in R, Java, Python, etc., as well as for creating data transformation in ETL, Karma or NIFI.
- Executing and controlling in a reliable and scalable manner smart city processes (IOT, ETL and data analytic, exploiting data) on cloud infrastructure, that can be used to create smart city MicroServices for computing values periodically and/or in real-time.
- Showing and navigating on data results via (i) city dashboards (for control room or for operators) for data drill down on time and space, and in turn facilitate the production of specific city dashboards for decision makers and city operators, at different levels; (ii) smart city API; and (iii) bulk data results.
- Providing access to data and services via (i) MicroServices for IOT and ETL applications, and via (ii) Advanced Smart City API for Web and Mobile App and dashboards.

Fig. 4.1 presents an overview of the Snap4City architecture, in which the data ingestion section has been implemented by using: (i) Km4City solution for ETL processes supporting protocols such as: OneM2M, ETSI, DATEX, Rest Call, FTP, Web Services, etc., mainly for ingesting data as Open Data, referral data, and real-time data provided by external services [33], (ii) a number of IOT Brokers to cover the connection with IOT devices, namely FiWare Orion Broker (NGSI protocol), Mosquitto (MQTT protocol) and RabbitMQ (AMQP protocol).

With the aim of abstracting the complexity of managing multiple IOT brokers and protocols, a new tool called IOT Directory has been developed in Snap4City. The IOT Directory allows registering IOT Brokers and collecting their corresponding registered devices, and propagating this information into

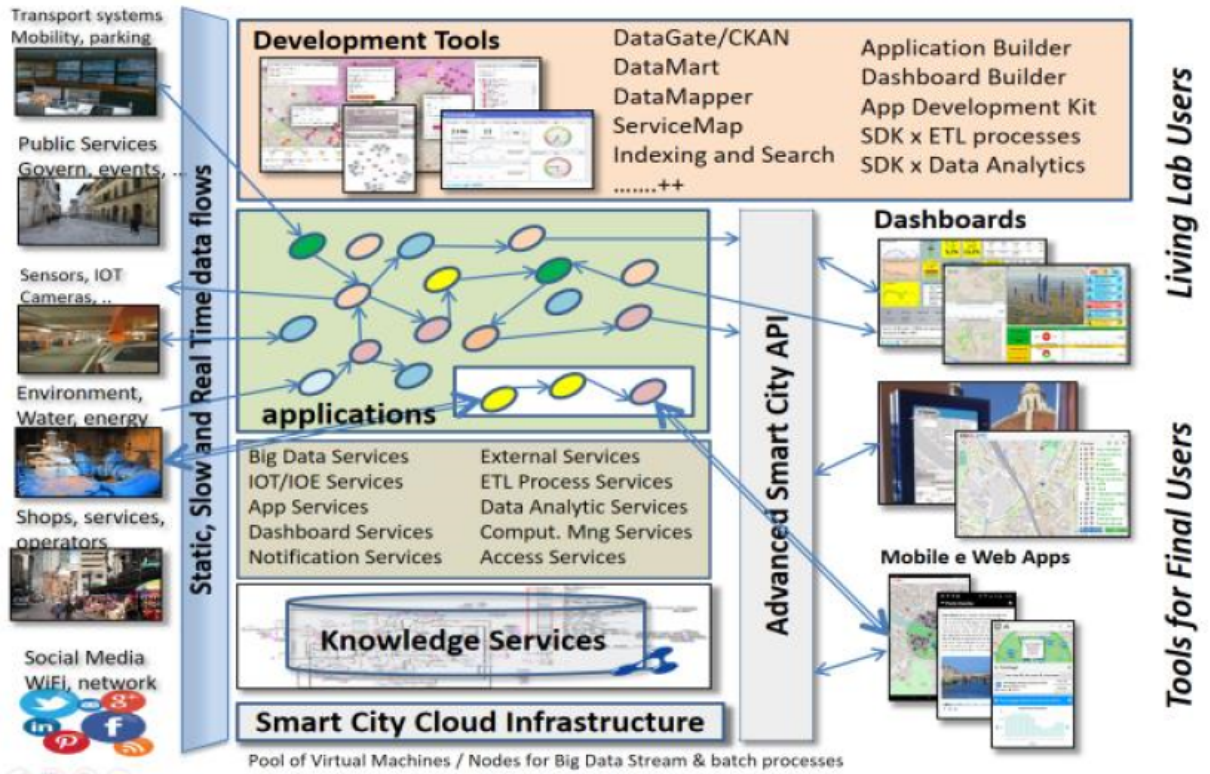


Figure 4.1: Snap4City Architecture Overview

the Km4City KB. The IOT Directory also provides support for IOT applications and processes that would like to subscribe to receive data driven from IOT devices, registered on the brokers. So that, when an IOT application is designed, the IOT application refers to the IOT directory/discovery to look for a device instead of searching for them manually on all single brokers. For example, in the NodeRED approach, one should connect each single IOT Broker for each single sensor/device. Instead, in Snap4City case, a new block and service abstracting the whole set of devices is available and it has been called IOT Directory and made accessible also for NodeRED users. An IOT Application developer exploits the IOT Directory to search and discover the most suitable devices to be used, by searching through metadata and/or on map. Discovering capabilities based on the use of maps are offered that rely on the KB services.

According to Km4City solution [33], the city entities' models and relationships are stored into a KB grounded on Km4City ontology, while the historical and real-time data are collected in Apache Phoenix storage. In order to cope with the IOT devices and allow them to be discovered in the map, each device has to be registered on the KB with its attributes. This way, its current and historical values are accessible for data analytics and further reasoning. To this end, the IOT values have to be logged into the Hbase storage. This automated feeding process is performed by a set of processes (one for each IOT Broker) realized in Apache NIFI. The same set of processes also performs the indexing for general view, facet and timeline drill down on Developer Dashboard. In this manner, both historical/referral data and real-time data are accessible via Advanced Smart City API that exploit both the KB Services for spatial-temporal reasoning in SPARQL and Hbase/Phoenix in SQL, integrated. Moreover, the Advanced Smart City API are accessible as MicroServices for IOT applications in NodeRED. To this end, to implement the MicroServices, a number of NodeRed Blocks have been realized and deployed into NodeRED tools. The Advanced Smart City API and Knowledge Services also provide data and information to Dashboards that can be produced by the newly developed Dashboard Builder tool.

The Snap4City architecture is completed by a number of tools such as: (i) Dashboard Builder for creating city dashboard and IOT actuator for them; (ii) ServiceMap for working with the KB and generating smart city API; (iii) a set of Developers Dashboards for monitoring consumed resources on

the platform (data stored, cloud resources at level of VM, hosts and containers, and network traffic among MicroServices); (iv) IOT Applications; (v) Mobile and Web Applications; (vi) IOT/IOE devices; (vii) ETL processes. With regard to the management of cloud resources for the IOT applications, ETL and Data Analytics processes are put in execution on a pool of VM as containers. Containers are realized by Docker while the Marathon-Mesos solution is used for their management. The whole set of VM constituting the solution is also allocated on cloud. Therefore, the system may keep under control the consumption of resources also controlling when they have to be turned on/off. Before passing to describe the performance analysis, a functional example is reported in the next section that can be regarded as a proof of a large number of the above-mentioned functional requirements.

4.3 An Example

In order to put in evidence the capability of the Snap4City solution with respect the IOT challenge presented by Select4Cities, and the state of the art tools, consider the following scenario of mobility and transport in which the following entities are involved:

- A Mobility Operator of the city or some mobility stakeholder in front of a Web page which may be interested in understanding the city dashboard status in terms of real time and referral data regarding mobility and transport (traffic sensors, parking status and predictions, triage in the hospital, environmental data, etc.), and may act on the duration of the red-light semaphore in a specific case, and on speed limit in major paths of the city.
- One or more drivers sitting on their public vehicles and touring the city (such as a bus, an ambulance, a police car, a garbage collector track) that may need to monitor the city traffic conditions and environmental data, and may also need to act on the red-light semaphore (the one controlled by the Mobility Operator).

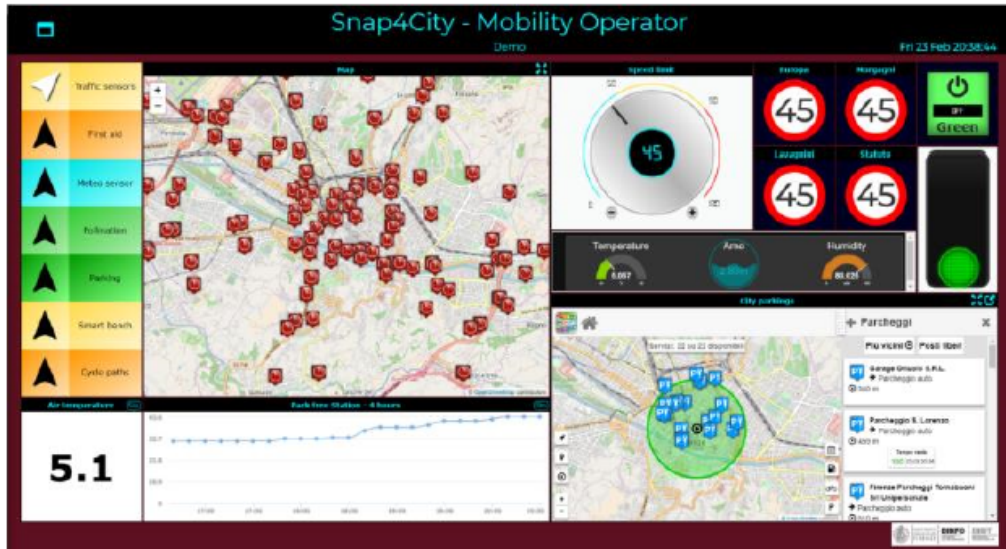
Each of them has his own Dashboard to control, one on the Web and the other on mobile as depicted on Fig. 4.2. The Mobility Operator may change the speed limit acting on the dimer with double control (fine and by step), which is an IOT device encapsulated into the City Dashboard. Any action

on IOT dimer provokes the sending of a message to an IOT Broker (IOT Orion broker).

Thus, an IOT application has been designed to update (in data driven) the changes of the dimer (telemetry protocol) and thus to propagate the new speed limit to a number of Speed Limit Plates along a major path in the city. Also the involved Speed Limit Plates are IOT devices/actuators, which in turn are read by the former Dashboard to allow the Mobility Operator to verify if the information has been correctly propagated. In the same manner, the Speed Limit on the mobile of Driver is updated since he is registered on the limit corresponding to his location in that moment. The presented demonstrator can be accessed to play with by using the links in Fig. 4.2.

Both the Driver and the Mobility Operator may act on their corresponding Dashboard for obtaining the trend and realtime value of the measured traffic and parking conditions. The Mobility operator may obtain that result by acting on the selector on the left side of the city dashboard, while the Driver acting on the map of the mobile IOT App (by using the finger keep pressed on the map for a while, equivalent to the right button of the mouse) and selecting the point, thus simulating the movements on the car.

In this integrated scenario, the Mobile application has been developed by an IOT/IOE application in NodeRED flow with the addition of Snap4City MicroServices as depicted in Fig. 4.3. In order to allow monitoring and assessing the network workload, in the flow, each data received/requested (RX) by the flow, and transmitted (TX) is also logged into the EventLog block. It collects the data, and it indexes them to monitor the flow by using the AMMA tool (Application and MicroService Monitor and Analyzer) for understanding the traffic on the infrastructure, implemented as a Banana Dashboard on a multi facet and sharded index on SOLR. A more elegant solution would be to hide the log into the Smart City MicroServices. In addition, the Mobile Operator may act on the red-light semaphore pressing the green button on the up-right corner of the Dashboard of Fig. 4.2. If the semaphore light is green, the duration of green is increased of 10s for each push. On the contrary, when the button is pressed, and it is red since 15s, it is changed to green, otherwise the change to green is delayed of 15s or red will be passed (this guarantees a minimum flow in the other direction). The same actions can be obtained by the Driver, who also sees the countdown until the semaphore changes its state. Other logic can be easily enforced



(a) <https://www.disit.org/dashboardSmartCity/view/index.php?iddashboard=MjU3>



(b) <https://iot-app.snap4city.org/nodered/nr15/ui>

Figure 4.2: (a) Mobility Operator, (b) the view on mobile device of the Driver

(e.g. to make the reservation without acting on the red duration).

The above described logic and transformation has been implemented with the IOT application reported in Fig. 4.4. Also in this case, the monitoring of the traffic flow has been implemented by adding specific EventLog blocks. Please note that in the above example, (i) the Mobility Operator dashboard reported in Fig. 4.2 has been implemented by using the Dashboard Builder for Snap4City also including IOT devices embedded into the dashboard and exploiting real-time data, referral data, and data driven events (received and produced), (ii) the IOT/IOE application in the hands of the Driver has been developed in NodeRED with the flow reported in Fig. 4.4, in which most of the functionalities have been realized through MicroServices developed by Snap4City enforcing access and exploitation of the Smart City API of Km4City, (iii) the IOT access has been realized using the services of IOT Directory and IOT Discovery tools for Snap4City, (iv) the EventLog has been used for monitoring the network usage by the IOT and ETL applications in the infrastructure. This latter issue will be more evident when in the following a view of the AMMA tool (Application and MicroService Monitor and Analyzer) will be discussed. The solution for IOT smart city has also to cope with security and privacy of the city users. To this end, the IOT-smart city solutions have to provide end-to-end secure connection, for example using TLS/HTTPS messaging for the IOT and HTTPS for accessing the tools.

4.4 Performance Analysis

The adoption of IOT applications in the context of smart city may imply to give at each single city user the possibility of having and programming one or more IOT Applications. These whole set of IOT applications and data processes running in the smart city back-office have to be maintained under control, since some of them could be (i) intentionally developed for creating problems, (ii) developed by inexperienced users that could produce IOT applications that in some unplanned manner may create large network traffic, (iii) adopted in terms of instances by many people thus creating a large number of instances performing the same actions, which will become computationally inefficient and will uselessly increase the costs. In the latter case, repeated functions/computations could be leveraged to a shared city service (IOT app as well or ETL), thus simplifying the costs and the flows.

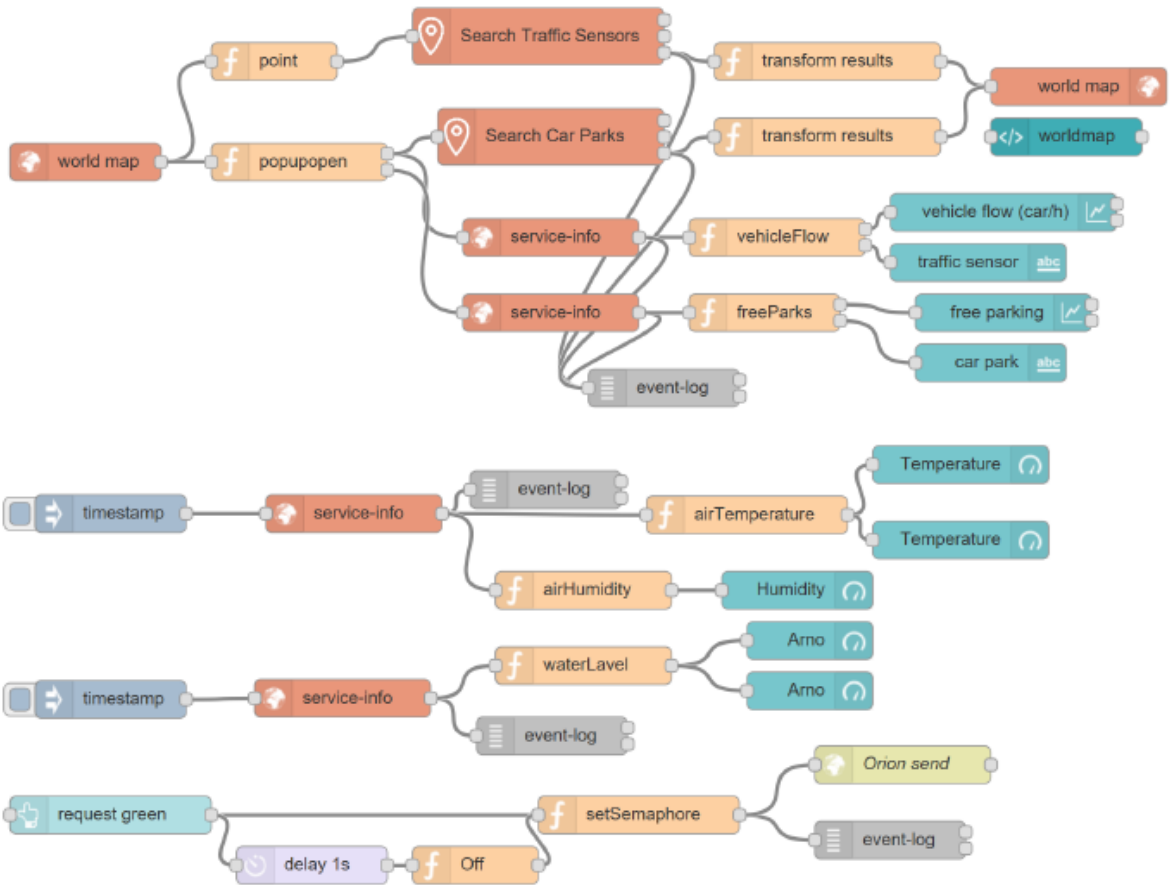


Figure 4.3: IOT application in which data are collected from users actions, referral data from MicroServices of Smart City API Km4City and from IOT brokers

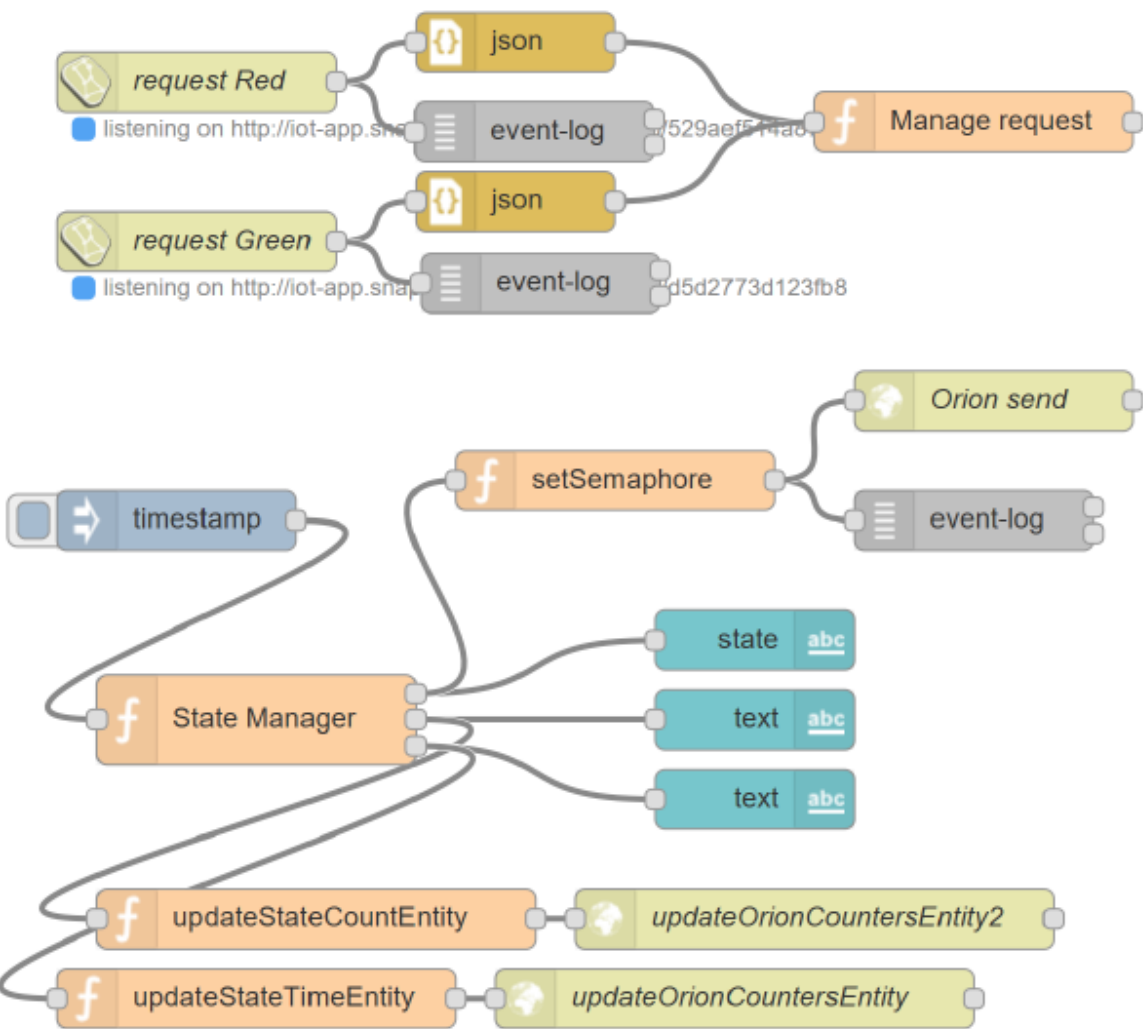


Figure 4.4: IOT app implementing the logic of red-light control of the semaphore

Therefore, the continuous analysis of the back-office network traffic and cloud workload helps the smart city operator to keep clean and sustainable the solution. Most of the IOT platforms have business models based on the number of messages or Kbyte exchanged. For these reasons, the performance analysis of the proposed solution has been conducted addressing two different aspects and scenarios:

- Daily and real-time: the possibility of putting in the hands of developers and smart city managers an AMMA tool for real-time monitoring and control of the consumption of communications bandwidth among the IOT applications, external services, storage, Micro-Services and smart city API, file system, etc.;
- Sporadically for validation: the assessment of the architecture in terms of maximum number of messages that can be managed by brokers and applications when a certain amount of cloud resources are available for running the processes that put in execution the IOT applications as containers on cloud.

4.4.1 Assessing Communications Bandwidth

In order to assess the communication workload among the several service processes (MicroServices) in the smart city IOT back-office, a large number of entities has to be monitored. Among them, those entities that mainly provoke the traffic are: ETL, DataAnalytics and IOT [33]. The communications may be among applications, with respect to external services, due to periodic processes, accessing to the smart city API, etc. Therefore, the whole set of data flow volumes and messages have to be logged and the log data has to be easily partitionable and inspectable by the developers to understand over the time line and on map: (i) how much a given application, application area and services are communicating, in terms of volume and messages, (ii) how many messages are exchanged among applications, services, etc.⁴ (iii) which are the top applications in terms of communication workload, etc. The communication activities are related to messages, for each of which one should have a clear view of its purpose and cost:

- transmitting/receiving data to/from;

⁴Note that in most of the IOT applications on the market the prices exposed, and thus the business models strongly depend on the number of messages and/or the volume of data exchanged in terms of KByte/MByte.

- motivation: IOT (via IOT brokers or directly to devices), data store (save and load), dashboards (user interface interaction), file system access, external services via rest call API, WS, FTP, etc.; and Smart City API;
- protocol: MQTT, COAP, AMQP, HTTP, OneM2M, ..;
- process agent kind: ETL, Data Analytics or IOT apps;
- mastered by an IP-local to/from an external service at a given IP;
- and, in the case of external service, the URL should be a selection aspect as well as the service scope which can be internal or external to the smart city cloud.

For these reasons, an extension to the above-mentioned architecture has been realized to collect the stream of EventLog data and indexing them into a large SOLR index, thus allowing the access with a facet viewer to the whole set of data, and so permitting to the developer to perform facet search and drill down over timeline as depicted in Fig. 4.5, where the first set of pie charts and histograms is representing the distribution of the volumes of message, while the second is representing the distribution of the flows in terms of message size. In the observed period, most of the messages have been due to the database storage of ETL in TX. While in terms of volume (message size), also the file system has been strongly involved and there is balance in TX/RX. From the latter histograms, it is also possible to identify which are the most exploited services involved: parking, smart benches, waste collector, bike sharing racks, electric vehicles.

By selecting with the mouse the single bars, it is possible to insert a filtering and request to all the other representation to recompute the graphs. In this manner, the analyst can study the contributions produced and received by each single IOT application towards all the other services, over time and kind.

4.4.2 Assessing Maximum Number of Messages

In order to assess the maximum number of messages per minute which can be managed by a set of IOT applications with a certain number of resources, a test bed has to be set up. To this end, we have allocated 6 virtual machines (VM), with 300 Docker Containers each, and with IOT application processes,

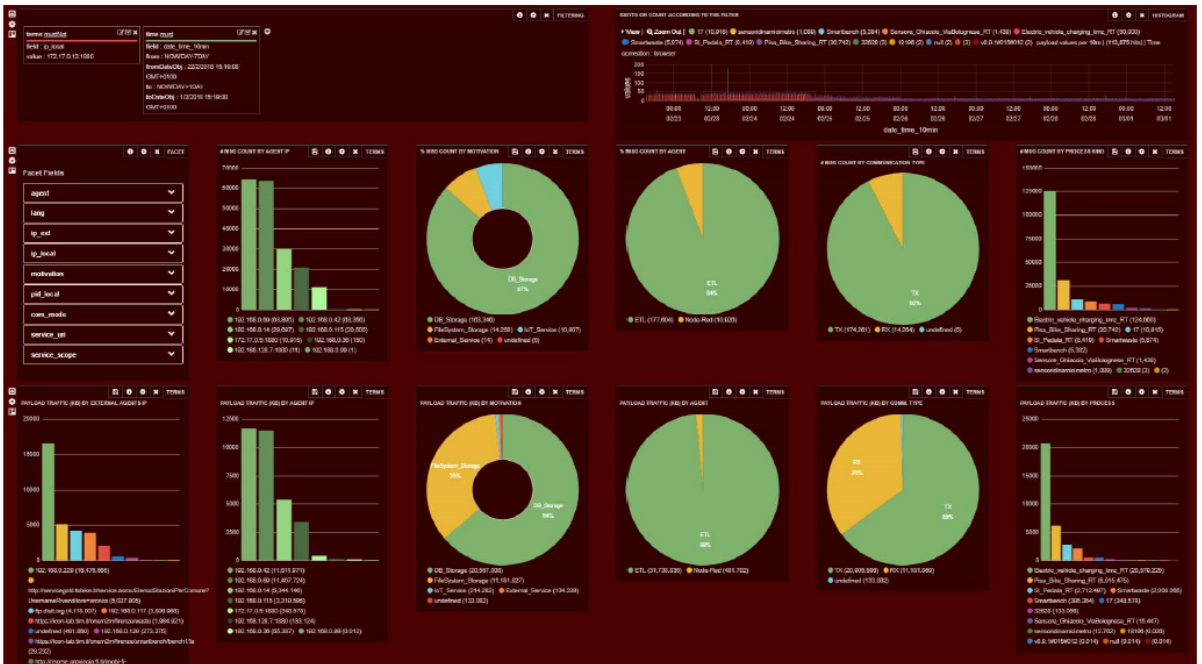


Figure 4.5: AMMA tool: monitoring in real-time data streams with facet search and interactive dashboard for segmenting the traffic: origin destination.

equipped with 196KByte of memory, and 79 GByte Ram. Each host had 24 cores at 2.299 GHz, on 128 GByte RAM: up to the 75VM process in a safe/live range. Each single IOT Application was based on a NodeRED, which can sustain about (in that condition) 120 msg/s MQTT in input thus performing 120 REST Calls per second. Thus obtaining 240 events per second: 96.000 events per second per Host/Node.

In Fig. 4.6, a monitoring panel (also implemented as IOT application) for the workload is presented. On the left bottom corner the trend of the number of healthy IOT applications/containers is presented. So that, 300 healthy IOT applications have been registered. The test started by using a variable number of MQTT messages generated on a Mosquitto broker which was sending new values of data to the 300 allocated IOT applications. During the same test time, the amount of messages sent per second has been changed arriving at 1 Million of messages per minute. Each IOT application of the test, in effect, collects an MQTT message and sends a message to a MicroService of data store. Therefore, for the cluster, the total number of messages per minute was about 2 million. It has been reached by using a number of IOT MQTT brokers with the aim of arriving at the limit when the maximum acceptable exploitation of memory and CPU resources was reached. The maximum workload has been fixed at the 75% of Cloud VM resource exploitation. When the limit is reached, the IOT Applications in Container are not capable to react, and thus the manager identifies them as notrunning. Fig. 4.6 depicts the case which represents the limit for the resources available in that configuration.

The scalability of the solution may be obtained quite easily since the overhead of the solution is close to 13ESX, Marathon per Node, plus an offset due to the needs of having 3 VM for the masters that may have managed up to 16 Hosts/nodes, thus doubling resource is needed every other 16 Hosts/nodes. In our test case, 6 VM have been managed. At the same time the monitoring of cloud resources has been obtained by using the cloud infrastructure tools as reported in Fig. 4.7. The picture represents a real condition in which VM with IOT contained managed by Mesos work together with other VM with other processes and services.

4.5 Conclusions and Future Work

Smart City solutions initially started with the production of open data and

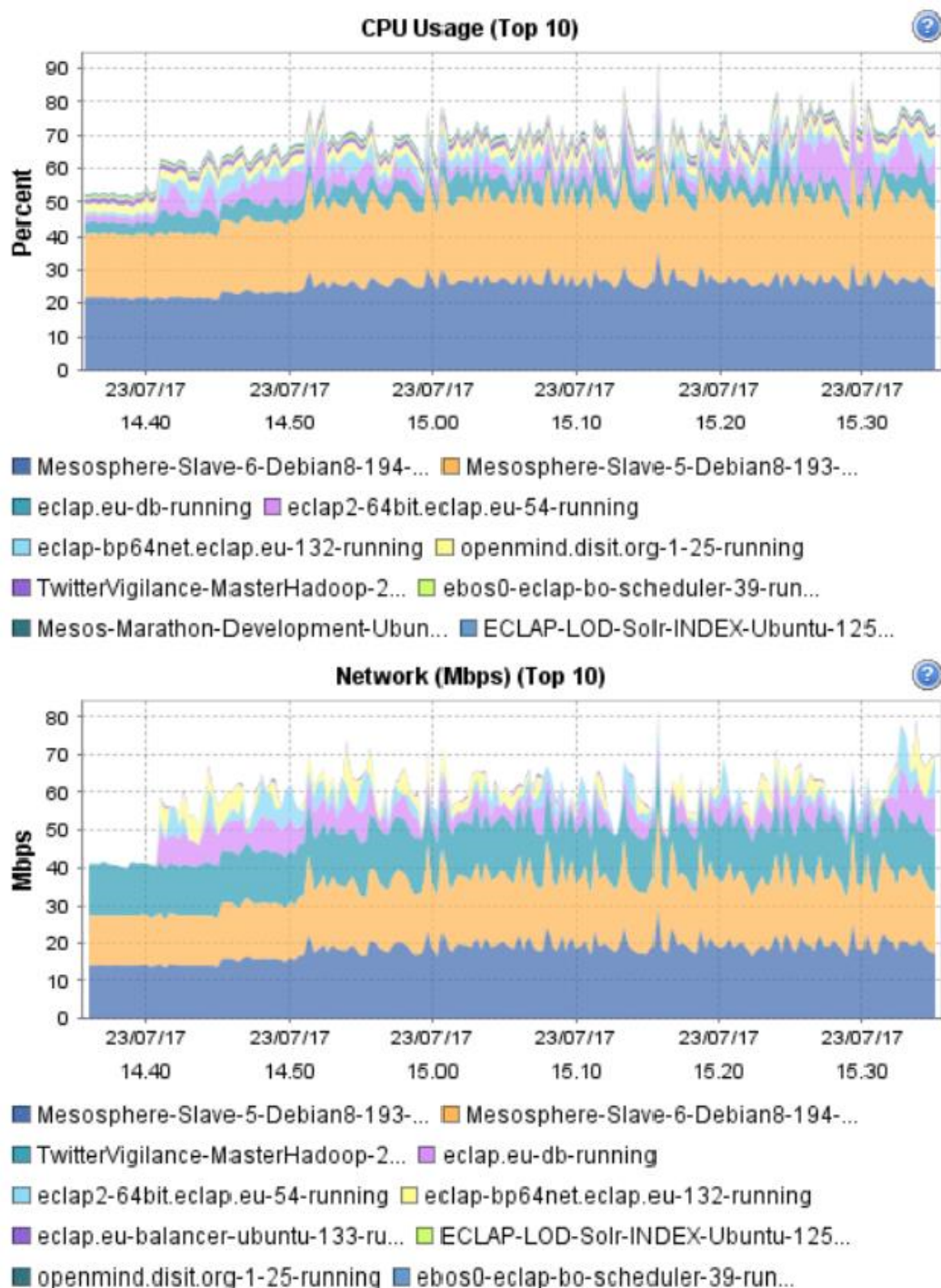


Figure 4.6: Monitoring sent/received messages per minute on VM cluster including IOT apps container managed by Marathon-Mesos-Docker



Figure 4.7: Monitoring cloud resources during the workload test for the IOT smart city applications

their collection on managing open data tools. Then smart cities evolved towards data aggregation and semantics modeling of city entities, benchmarks, and datasets. Recently, some of them are also offering IOT support. The combination of IOT and smart city is not an easy task, the data volumes are much higher than those addressed for industrial IOT. The complexity of IOT smart city solutions have been identified by a number of actors. The European commission started set up the EIP project for stimulating and concerting actions. On the other hand, the Select4Cities project of the European Commission and associated community <http://www.select4cities.eu/> created a challenge to find research-based solutions satisfying a formalized set of functional and nonfunctional requirements. Snap4City solution that has been presented in this chapter, is one of the solutions approved and developed in response to that challenge. Snap4City solution is based on MicroServices directly provided by a number of applications and Smart City API. Snap4City users can develop in few steps sophisticated IOT applications exploiting MicroServices that can control city dashboards as well as IOT mobile applications. Moreover, a number of development and monitor-

ing tools have been designed and developed. Among them, in this chapter, a special attention was given to the tools and solutions for monitoring communication performance among IOT, storage, smart city API, external services, etc. Another aspect addressed in this chapter consisted on the test to perform the assessment of scalability feature of the IOT smart city infrastructure in terms of maximum number of messages exchanged per minute fixed the number IOT processes and corresponding resources. Future work will be devoted to put together these aspects to develop algorithms for automated scaling and IOT cloud management.

Chapter 5

Snap4city Platform: Semantic to Improve Location Based Services

Location-based Services are becoming crucial in the most recent smart city scenarios, when, according to scholars and urban planners, one of the main objectives is to improve the living conditions of citizens who act and move within an urban or regional area. This chapter aims to present the Snap4city platform, an infrastructure capable to collect large geographical and statistical data from different sources, integrate them into a semantic compliant with the Km4City multiontology and finally provide LBSs for multiple users. The Snap4City platform is currently running in many geographical contexts (such as Antwerp, Helsinki, Santiago de Compostela, etc. Italian cities such as Florence, Cagliari, Pisa, Livorno, Modena, Bologna, etc. and entire geophysical zones: Finland, Belgium, Tuscany, Sardinia, Garda Lake, etc.). This allows to present preliminary results for system validation.

1 2

¹This chapter has been published as “Snap4city Platform: Semantic to Improve Location Based Services” in *15th Conf. on Location Based Services, LBS2019, Advances in Cartography and GIScience of the ICA* [48].

²*Acknowledgments:* The authors would like to thank the TRAFair CEF project of the EC with grant AGREEMENT No INEA/CEF/ICT/A2017/1566782 also all the companies and partners involved. Snap4City and Km4City are open technologies and research of

5.1 Introduction

The amount of information available today is such that issues relating to the quality and significance of it must be addressed already in the designing of an architecture supporting the smart city. Reference scenarios must be well defined to analyse the specific needs of citizens and their behaviour and the actions that contribute to achieving them. Within these scenarios the geo-referenced information is crucial in order to get context-sensitive description and analysis of emergent local practices. In addition, each city has its specific problems, due to its geographical position, its geomorphology or history and culture that make of it a unicum. For these reasons, even though digital and technology-driven approaches are often considered in literature as a universal solution, when it is intended to replicate a model in different cities, individual specificities must be taken into account and therefore develop strategies that may get inspiration from other contexts but are as unique and specific as the city itself. In this logic, the experimentation conducted in different urban and regional areas through Snap4City architecture highlights the paradigm shift, since it does not adopt a simply technology-driven but more specifically data-driven approach. Big Data, open data, sensors, IoT/IoE for monitoring, controlling and managing urban developments, resources, urban infrastructures, energy consumption, traffic congestion, waste, pollution, risks and people, are the tools for a governance inspired by a data-driven urbanism, according to which the expected changes are a consequence of a decision-making process. [25]. The nowadays availability of geospatial knowledge and the worldwide generation of geospatial data allow to design services that can support decision making and help to find solutions to daily life problems. However, “the lack of explicit semantics inhibits the dynamic selection of those data, services, and geoprocessing workflows needed for processing geospatial information and discovering knowledge in a data-rich distributed environment” [138]. An ontology-based approach allows the representation and semantic interoperability of geospatial data and related processes. The integration of semantic information makes LBSs intelligent and able to enhance a smart city. In order to cover all these aspects and manage the large amount of data coming from different sources, from the Smart Cities Open Data Portals to the IoT/IoE connected devices (sensors, actuators and various agents rapidly increasing in number in a smart city context), a Big Data

approach is necessary. One of the most innovative tools, is to flank to a Big Data Platform, the possibility of aggregating data at a semantic level allowing data to communicate among themselves. Thus, the aggregated data, can be used to extract new knowledge and develop services for citizens increasingly performing and focused on their needs. In such a context a Big Data Platform must be compliant with: i) Data Precision in terms of Geospatial representation and timing; ii) Semantic Data Aggregation; ii) Scalability in terms of data stored/managed and services provided; iv) Guarantee of different Privacy Levels.

5.2 Snap4City Architecture

The Snap4City infrastructure respects all the requirements above described. Snap4City is a Big Data Platform compliant with the KM4City multiontology, managing and aggregating data from EU cities and regions.

In Fig. 5.1, the Snap4City Architecture is depicted. The starting point is the Big Data Processing (harvesting, formalization, geo-localization enrichment, storage, etc.), all the ingested data are aggregated and collected in the Snap4City Knowledge Base, in compliance with the Km4City multi-ontology [51], and indexed in order to speed up and facilitate data retrieval. A such semantic aggregation of data poses the basis to provide LBSs to the final users, always at the forefront both in terms of functionalities offered and of performance achieved. These activities are included in many European (Snap4City, TRAFair, Replicate, etc.) and national (Sii-Mobility, Ghost, MOSAIC, etc.) research projects. Thus, the aggregated data can be analysed and used to produce Smart LBSs by generating predictions and suggestions for final users. The Snap4City solution allows to ingest and manage Big Data coming from different providers, External Services, OD Portals, ArcGIS Enterprise Servers, IoT/IoE devices, applications and services, and finally provides LBSs, available to the final users thanks to the set of visual tools developed. A Dashboard System allows all registered users to easily create a set of different thematic dashboards (weather, mobility, energy, etc.), respecting user authorizations and privacy level on the data (Snap4City is GDPR compliant). These interactive dashboards support decision-making processes for: Public Administrations, tourists, citizens, developers, etc. Moreover, Snap4City allows the creation of data-driven applications: based on Microservices, exploiting mobile and web apps, flows of

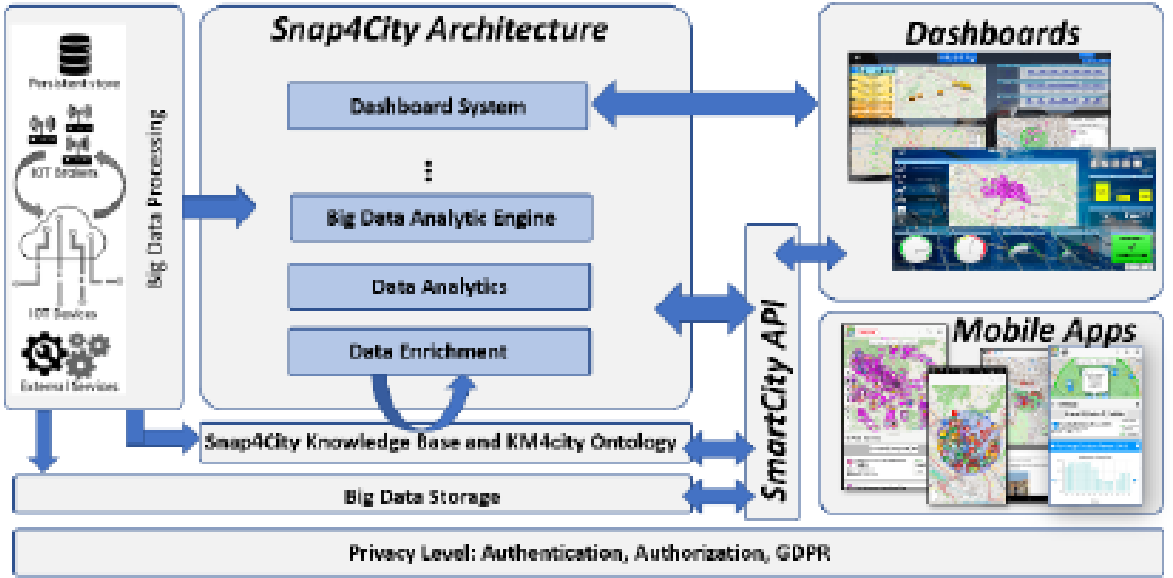


Figure 5.1: Snap4City Architecture

processing data and IoT data [49].

5.3 Km4City Multi-Ontology

The need to standardize and aggregate data becomes increasingly necessary in a Big Data context, for this reason the Snap4City solution is based on the Km4City multi-ontology. The development of Km4City started in 2013 to interconnect the data provided by the Tuscany Region, the Open Data of the City of Florence (Static and Real Time). Some notable works included are: (i) the Ontology of Transportation Networks [14]; (ii) the Semantic Sensors Network Ontology [108]; (iii) the IoT Lite Ontology [53]; (iv) the Schema.org Ontology [82], that now also includes the GoodRelations project. Km4City enables interconnection, storage and the retrieval of heterogeneous data from many different sources. Its main sections are: (i) administration; (ii) road infrastructure; (iii) POIs; (iv) public transport; (v) Internet of Things (IOT). Predictions and early warnings, mobility plan-

ning, multimodal routing, resiliency, are just a few of the challenges that can be faced to improve city governance and people behaviours, and therefore citizens' quality of life, tourists' satisfaction, environmental sustainability. On the other hand, IoTApps are required to: (i) collect huge amounts of heterogeneous data at real-time; (ii) be resilient and ensure safety, security and privacy of distributed data and communications; (iii) enable machine learning and data analysis; (iv) enable effective data displaying [36]. The Km4City Ontology also includes a set of concepts for modelling real-time status and predictions, and a flexible modelling of IoT/IoE entities has been achieved, introducing IoTSensor and IoTActuator, and IoTBroker with its specializations NGSIBroker, MQTTBroker, etc. Types of measurements and managed input signals are modelled as DeviceAttribute instances, each having a `value_type` property filled by an SSN Property [108] that allows to distinguish devices that measure temperatures from those that measure traffic flows and so on.

5.4 Snap4city Location-Based Services

LBS are largely dependent on the data coming from IoT/IoE and from the mobile user's location: the understanding of where the user is and what are his/her needs, is one of the primary objectives of a service provider system.

Some of the most common LBS applications include: POI, local events, multimodal routes, emergency, asset tracking, weather forecast, statistics, previsions, etc. All the services can be requested by the users (pull) or sent to them in form of recommendations (push). The Snap4City most common LBS requested by the users locate: i) around a user (GPS point); ii) in a specific geographical area described (e.g. a polygon in Well-known text format); iii) along a line; iv) related to the Public transport lines, routes, timetables, etc. (Fig. 5.2). The Snap4City Recommendation System is capable to identify user needs and send suggestions based on user's position in the city (via Mobile Apps, totem, etc.).

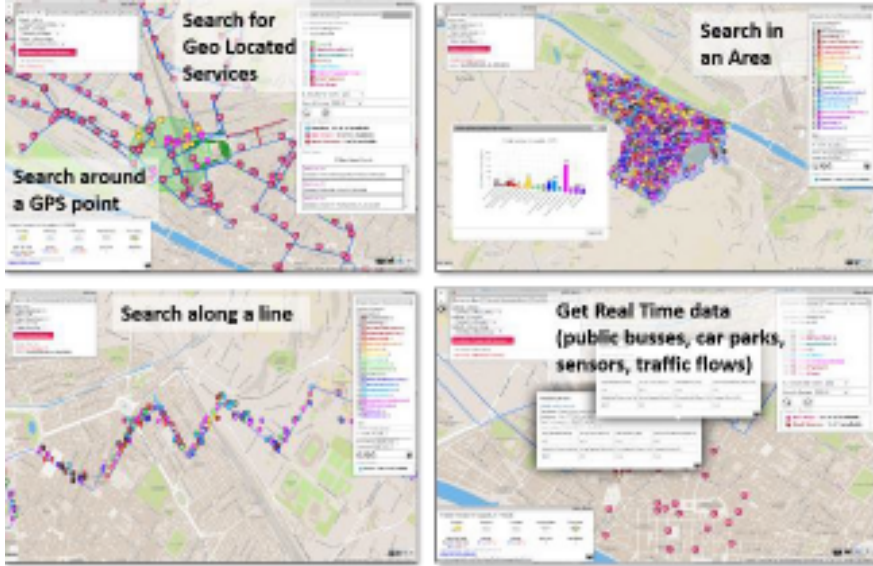


Figure 5.2: Snap4City Geographical searches

5.5 Conclusion. Snap4city Validation Scenarios in Smart Cities

The Snap4city platform is currently used in many cities and regions. The Snap4city APIs allow accessing all these information from different applications. Some mobile applications have been developed (e.g. Helsinki in a Snap, Antwerp in a Snap and Tuscany in a Snap) to leverage all these geographically annotated information and provide useful services to the citizens or tourists on the move from their mobile phones (find services nearby, time tables, heatmaps of pollution or weather conditions, etc.).

The platform allows building custom dashboards using the data coming from the city. The dashboards can use maps allowing users to locate services and access to data associated to them, such as that developed for the city of Helsinki, Fig. 5.3. The experimentation of the platform in the context of the Select4Cities project (Helsinki, Antwerp) has been successfully completed.

Table 5.1 reports the Snap4city geographical search APIs use, in different cities in the month of July 2019. Most of the APIs used are related to geographical searches (93%) and performed in a circular shape; while searches



Figure 5.3: City of Helsinki dashboard

Geo search Type	N. requests	
Coord & radius	445,608	93.32%
Rectangle	31,248	6.54%
Polygonal area/path	655	0.13%
Total	477,511	

Table 5.1: Snap4city geographical search API.

in a rectangular area reach the 6.5% of total. The advanced searches inside a complex polygonal area or along a path cover the 0.13% of the requests.

Chapter 6

Federation of Smart City Services via APIs

In the context of Smart City, it is quite frequent the usage of Smart City API for providing services to Web and mobile applications. Most of the solutions using Smart City APIs are focused on a single city. This means that passing from one city/area to another, the users must change application. This happens for the lack of interoperability among Smart City APIs and/or services. In this chapter, the problem of federation of smart city services is addressed by proposing a solution for federating smart city APIs. To this end, a formal model has been proposed to federate API services, with efficiency, security, scalability, and capacity of managing overlapped areas of competence, distributed searches, etc.. These features are typically not all satisfied by classic GIS solutions which federate the services at level of databases. The solution has been developed in the context of Snap4City European platform enhancing former Km4City API of Sii-Mobility national project with Snap4City (<https://www.snap4city.org>).¹

¹This chapter has been published as “Federation of Smart City Services via APIs” in *Proc. of 6th IEEE International Workshop on Sensors and Smart Cities, with IEEE SmartComp, 14-17 Sept. 2020, Bologna, Italy.* [47].

6.1 Introduction

In the context of Smart Cities, not all cities/areas are becoming smart in the same manner [81]. In most cases, the cities are focused to a set of smart services, for example: smart parking, smart education, smart gov, smart lighting, etc., according to their needs and strategies. Also often, vertical applications are implemented as separate pillars (see also Subsection 2.3.1 in Chapter 2).

In this chapter, a solution for federating Smart City API on geographic area is presented. The study has been motivated by the need of scaling up the Km4City Smart City APIs from single cities to regions and from regions to a number of European areas, in the context of the Snap4City PCP for Helsinki, and Antwerp [37]. The proposed solution aims to satisfy a set of requirements that are not presently satisfied by traditional GIS solutions. Indeed, our proposed solution does not need to: (i) create a strong cohesion among the different services and thus does not leave to the single city the decision to join the federation or not and neither to the single client Mobile application, (ii) connect GIS services via WFS/WMS services, (iii) migrate data among services. For the design, implementation and validation of the solution we have used and enhanced the former Km4City API and ontology [117], which has been implemented as the so called ServiceMap. For this reason, the federation of Smart City API has been provided in the so-called SuperServiceMap. The validation has been performed by considering 4 large areas and smart city services in place covering: Tuscany region of 3.5 M inhabitants in the center of Italy, north of Italy and Sardegna island, Antwerp and north of Belgium, and Helsinki and south of Finland.

The chapter is organized as follows. In section 6.2, the requirements identified for federating Smart City API are presented. Section 6.3 presents the general architecture and solution for federating large smart city services, which may cover single cities, as well as regions. In Section 6.4, a formal operational model and semantic for the federation of smart city API and services are presented. Section 6.5 describes the main families of the Advanced Smart City API which can be exploited in the federated network. Section 6.6 provides the evidence of how the requirements identified in Section 6.2 have been satisfied. Validation and performance have been reported in Section 6.7. Conclusions are drawn in Section 6.8.

6.2 Requirements and analysis

A distributed solution for a set of cities that is based on Smart City API (SCAPI), say a federated network of Smart City APIs (called in the following federated SCAPI), should satisfy a set of requirements, outlined in this section. The federated SCAPI network connects a set of SCAPI services (called Nodes) which are independently offered and maintained by single cities/areas. They provide their own data and services. It should not be confused with the federation or collection of APIs in a common basket to expose them uniformly in terms of definitions but managed by different organizations providing this service to a single city/area mainly as in E015 [139]. A more general view is that of ALMANAC [128], [57], which can be regarded as E015 with a limited scope in terms of services, among different cities. Thus, according to our definition the federated SCAPI network of nodes should:

1. guarantee that nodes do not need to permanently share data as in P2P solutions or as in federated databases. This requirement must be satisfied to ensure that the data are located only in the nodes authorized to manage them;
2. support distributed search on the federated SCAPI network;
3. support nodes of any size in terms of number and volume of the datasets that are used in the nodes. In addition, the geospatial size and shape of each node may be: (i) not regular (nor a circle but a shape), and multiple connected (so called multi-polygon), (ii) partially overlapped with other nodes, (iii) totally included into those of other nodes, (iv) disjointed and even far each other (this means that the union of all the areas can be disjointed with respect to the global map of the Earth);
4. support nodes with a different number of services available. This implies that not all kinds of services and data may be necessarily available in all nodes;
5. support nodes with georeferenced services or not. Non-georeferenced services extend over the whole area covered by the node, and they are not binded to a specific location or GPS coordinate;
6. respond to API calls in terms of services in transparent manner passing from one node to another or when the service needs to provide results coming from more nodes;

7. support access control to prevent access to data and services by not authorised users, since the passage of a user from one SCAPI node to another of the federated SCAPI network may imply the sending of requests which may try to access private data/services;
8. support the addition/removal of nodes in the network without the need of fully restructuring the network and modifications have an immediate effect without any service reloading or disruption;
9. provide results in real-time also when a large number of nodes/areas are involved. The implementation should also provide support for creating redundant solutions with high resilience;
10. provide the response in a format that is coherent with the format of the expected responses of the single services. Thus, the results of the federation may need to be merged to produce the response in any format: JSON, XML or HTML.

6.3 General architecture

In order to satisfy the requirements, the architecture reported in Fig. 6.1 has been realized. The architecture includes a Master of the federated SCAPI network implemented as Smart City Servers with SCAPI (in Snap4City they are the ServiceMap tools [51]). In order to avoid having a single point of failure, the Master can be replicated into each node and the list of Super services on top of ServiceMap SCAPI can be put accessible in one or more Web servers for update. Each ServiceMap has a representation of the multi-polygon addressed by the nodes (with their data/services) and thus of their partitioning over the nodes of the federated SCAPI network. In more detail, each node/SCAPI may register in the Master and Super network the descriptor of the multi-polygon area of its competence. This approach permits to the Supers to redirect the queries to the nodes that could provide the service and data. Thus the Supers do not need to hold the data of the nodes and perform the distribution of queries only to the involved nodes, to finally collect the results and performs data fusion. The Supers as well as the ServiceMaps may also implement some query caching solution like all the other SCAPI.

The most relevant aspects of the procedure are mainly related to the: (i) identification of nodes to be involved according to the requests received with

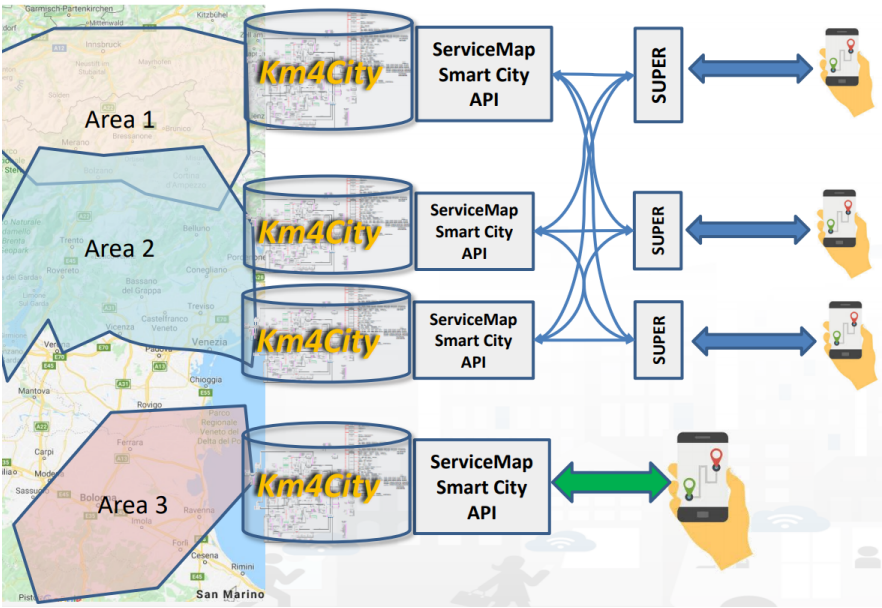


Figure 6.1: General Architecture of Super ServiceMap and Federated SCAPI network

a complexity that may depend on the size of the node descriptors (number of polygons); (ii) distribution of query in each Super with a complexity $O(1)$, the execution time would be top to the slowest node, exploiting multithreading approach; (iii) collection of results and their fusion to avoid duplicates and compounding the results, with a complexity depending on the size S of the elements $O(S)$. The main complexity is typically due to point (iii). When the nodes are created to split the complexity of computing services and thus to improve the performance, the workload on nodes has to be balanced. Moreover, in certain cases of (i), it is not possible to identify a priori one and only one node to be queried for a specific data/service. Therefore, in some cases all nodes are queried.

6.4 SuperServiceMap Operational Semantic

In this section, the ServiceMap model is presented and it is shown how different models are combined to create a SuperServiceMap exploiting former SCAPI based on Km4City plus some extension. In the context of the SuperServiceMap solution, a ServiceMap is defined by the following tuple:

$$ServiceMap = \langle E, G, A, D, loc, gs, ss, inf, path \rangle$$

where:

- E is the set of city/IOT entities managed by the ServiceMap. The city entities can be POI, services, IOT devices, road segments, etc.;
- $G \subset W = [-180, 180] \times [-90, 90]$ is a subset of the WGS84 geographic coordinates representing the geographic area covered by the ServiceMap;
- A is the n -dim space that models the arguments that can be provided for a search request;
- D is the m -dim space that models the data associated with an entity (e.g., name, address, category);
- $loc : E \rightarrow \mathcal{P}(G)$ is a function mapping for each entity a geographic point or area in G ;²

² $\mathcal{P}(X)$ is the powerset of X , set of all possible subsets of X

- $gs : \mathcal{P}(W) \times A \rightarrow \mathcal{P}(E)$ is a function that models geographic search that given a subset of W and some other arguments in A returns a subset of entities E that belong to the geographic area so that $\forall X \subset W \wedge e \in gs(X, a) \rightarrow loc(e) \cap X \neq \emptyset$;
- $ss : A \rightarrow \mathcal{P}(E)$ is a function modelling searches that do not involve geographic info, so that given the search parameters returns a subset of the entities satisfying the search;
- $inf : E \rightarrow D$ is a function which returns the list of details of a city entity;
- $path : G \times G \times A \rightarrow Seq(E)$ is a function that given two geographic points and other parameters returns a sequence of entities in E representing the “best” path/route (modal or multimodal, with different travel means).

This model allows to have entities with no geographic position i.e., $loc(E) = \emptyset$, in this case the gs function will never return these entities while they can be retrieved using the ss function. An important property of function gs is that if $X \cap G = \emptyset$ that can be read: searching outside the geographic coverage area of a ServiceMap will not give results.

Given a set of ServiceMaps $SM_1, SM_2, \dots, SM_N$ we can build the Super-ServiceMap to manage all entities in $ServiceMap_i = \langle E_i, G_i, A, D, loc_i, gs_i, ss_i, inf_i, path_i \rangle$ defining:

- $E = \cup_{i=1}^N E_i$
- $G = \cup_{i=1}^N G_i$
- $loc = \cup_{i=1}^N loc_i$
- $gs(X, a) = \cup_{i=1}^n gs_i(X, a)$
- $ss(a) = \cup_{i=1}^n ss_i(a)$
- $inf = \cup_{i=1}^N inf_i$
- $path = \cup_{i=1}^N path_i$

The sets A of possible arguments and D of possible details are not tied to any particular ServiceMap. No coherence problems arise if the set of city entities of different ServiceMaps are disjointed ($\forall i \neq j E_i \cap E_j = \emptyset$). While,

if some common city entities exists – i.e., $\exists e \in E_i \wedge e \in E_j$ it has to hold: $loc_i(e) = loc_j(e)$ and $inf_i(e) = inf_j(e)$.

Moreover, it should be noted that function *path* of SuperServiceMap, defined as the union of $path_i$ functions over the different G_i , does not associate a value when applied to two points belonging to disjoint G sets. The error management allows to split the problems in distinct routing queries at the corresponding ServiceMap services.

6.5 Advanced Smart City API, ASCAPI

This section is devoted to the description of the Advanced SCAPI, ASCAPI. The full list and semantic of input arguments for API (space A) can be retrieved from the online documentation of the ASCAPIs on <https://www.km4city.org/swagger/external/index.html>. The ASCAPIs have been strongly extended with respect to the former Km4City Smart City API of [51]. The extensions have been designed and performed to extend the number of data types, domains and city entities addressed, and to allow the federation of SCAPI based knowledge bases.

The *Service discovery and information* family of APIs allows to perform a search for services (i) around a given position, (ii) near to a given service, (iii) within the boundaries of a bounding box, (iv) within the boundaries of a given geographic area (which can be defined as a WKT shape), (v) within the boundaries of a given municipality, (vi) through one of the above filtered on the basis of a full-text to be searched. The same API allows to retrieve full details about a given service of interest, identified through its URI (services are stored as resources in semantic knowledge bases, ServiceMaps – also said the triplestores – each identified through a URI, that also is a URL, in a Linked Data). This is modelled with the *inf* function that is used to retrieve information about an entity. If it is not possible to recognize from the URI the ServiceMap that is managing the referred entity the only choice is asking to all ServiceMaps and caching the (service, ServiceMap id) pair for future requests. This is performed in the middle layer as described in the following.

The *Event Search* family of APIs allows to retrieve events in the region (past, current and future). The geographical region of interest is specified by the user through the selection input parameter.

The *Address and geometry search by GPS* family of APIs allows to retrieve a full address (municipality, road name, and civic number) that cor-

responds to the nearest to a given position, also possibly including public transport lines or other points of interest that locate nearby the selected position, possibly furtherly filtered through a full-text search.

The *Search paths in a geographic area* family of APIs allows to retrieve paths of the public transports that locate nearby a geographical position or within the boundaries of a geographical area. In this case, it searches for all the bus-stops in the supplied area and it returns all the bus lines that pass through those stops. The key input argument for the identification of the POI is the *selection*, modelled as *gs* function.

The *Shortest Route/Path Finder API* allows users to get the best route from a given starting point to a destination through a modal or multimodal routing. In this case, it is identified the ServiceMap managing them (via cache or via request) and if exists a ServiceMap that matches for both the source and destination it is queried otherwise an error is returned. The management of the error allows to split the problems in distinct routing queries to the corresponding ServiceMap services.

6.6 The SuperServiceMap Main Processing Aspects

In this section, the implementation of SuperServiceMaps is discussed in some more detail. The ServiceMap exposing SCAPI are implemented in Java and RDF triple stores. The SCAPI provides results to the SuperServiceMap or directly to the clients by using JSON. Each ServiceMap may have or not its own SuperServiceMap interface and in that case may be connected to all the other ServiceMaps as depicted in Fig. 6.1. The SuperServiceMap implements the ASCAPI and the logic reported in Section 6.4, indeed: (i) it identifies and queries the ServiceMaps on the basis of the received requests and their georeferences, (ii) it submits the queries to one or more ServiceMaps and gets the results, (iii) it combines the results and provides them to clients. Thus, when multiple ServiceMaps are queried, the request is delivered in parallel to all of them by the SuperServiceMap. Then the responses are collected and merged, duplicates are stripped out, resources are sorted with the same criterion that the ServiceMaps use, and the final response is provided back to the end user.

6.6.1 Selection of ServiceMaps to be queried and exceptions

The selection of the ServiceMap(s) to be queried is generally based on the geographic areas involved in the request. Thus, only the ServiceMaps addressing/intersecting that area are queried. The pair (service URI, Service Map) is then cached to reduce the number of requests from one service to another. In the case of multiple ServiceMaps insisting on the same area, in order to maximize performance, or making an a priori balance on services, it has to be possible to define a priority among the different ServiceMaps on the basis of the query or on services. This mechanism is practically used to satisfy Requirement 5. The passage from one SuperServiceMap to another can be also very easy since the Mobile Application can have a direct link with the SuperServiceMap and area for which it has been natively developed. If the Mobile Application finds out that all results come from a Service Map that is not its reference Service Map, it may perform a switch.

6.6.2 Merging ServiceMap results of any Format

ServiceMap ASCAPI (as smart city API) may be invoked to request results in JSON or HTML [51]. Results in JSON can be easily merged, while results in HTML may include interaction plus relative URLs to JavaScript, style sheets, images, and other can be found. This approach is very suitable for mobile clients which need to immediately provide the results without reformatting. Thus, for HTML, the SuperServiceMap needs to combine multiple results coming from different ServiceMaps which provide different base URLs. Thus, the HTML results received from the ServiceMaps are parsed, relative URLs are identified, and are replaced with full URLs.

The above subsections have discussed the solution to satisfy some of the above-mentioned requirements. Req. 1 is satisfied by the solution since the different ServiceMap/SuperServiceMap servers do not copy the data of the other services in local but only know the presence of the other services and have their high-level descriptor in terms of area of competence. Reqs. 2, 3, 4, 8 and 9 are satisfied and performance are shown in Section 6.7. Req. 7 is satisfied according to security aspects addressed at level of single ASCAPI as described in [37]. Req. 10 is described in Section 6.6.2. Reqs. 5 and 6 are discussed in Section 6.6.1.

6.7 Validation and Experimental Results

The solution has been validated, and we have also studied the performance in 4 different real operative conditions. To test the efficiency and effectiveness of the proposed approach, the following conditions have been set up: (i) three areas of interest with their ServiceMap have been taken: (1) Tuscany Region (about 519 Million triples), (2) City of Antwerp and City of Helsinki (about 203 Million triples) including two disjointed areas, and (3) Italy regions of Sardinia, Emilia, Veneto, Lombardia and Santiago de Compostela (about 321 Million triples, with a number of disjointed areas), the data and areas are accessible from <https://www.snap4city.org>, see also Fig. 6.2 that presents the overlapping between the two different KBs API for Tuscany and Emilia of Italy; (ii) a heterogeneous set of ASCAPI requests has been identified including geo and non geo based requests; (iii) five different solutions have been considered (see Table 6.1):

- FSSM: the SuperServiceMap is forced to forward requests to all ServiceMaps (three ServiceMap VMs each of which with 16 GByte of RAM and 12 cores) any query that it receives from clients, and it merges the results;
- PSSM: the SuperServiceMap queries a selection of the most suitable ServiceMaps (among the three ServiceMap VM each of which with 16 GByte of RAM and 12 cores) to be involved on the basis of the geoinfo included in the query, and merges the received results;
- ASM: the client sends the query directly to the ServiceMap of its referred area (a ServiceMap with 16 GByte of RAM and 12 cores);
- GSM: the client queries a ServiceMap in which all the data (triples) of the three areas have been stored (a VM with 16 GByte of RAM and 12 cores);
- PGSM: as GSM but with a ServiceMap VM with 48 GByte of RAM and 36 cores.

Measures for FSSM, PSSM and ASM have been performed by querying the real operative services in the corresponding cities and thus with several thousands of users that were performing access at the same time on the services. GSM and PGSM were instead autonomous servers isolated from

Query/Kind		Times of response (ms)			
		FSSM	PSSM	ASM	GSM PGSM
Get all services in a radius of 500 m from the center of Florence	A	5076	4670	<u>4033</u>	139329 132306
Get all services in a radius of 500 m from the center of Antwerp	A	1670	<u>1474</u>	1485	81161 76720
Get all services in a radius of 500 m from the center of Helsinki	A	2377	2148	<u>1622</u>	171713 160890
Get all events in a radius of 20 km from Florence this month	B	1,762	906	621	142 124
Get all events in a radius of 20 km from the center of Antwerp this month	B	1124	531	<u>478</u>	696 697
Get all events in a radius of 20 km from the center of Helsinki this month	B	656	525	404	179 131
Locate a given bar in the Municipality of Florence	C	2051	1572	1456	3259 761
Locate a given theatre in the Municipality of Antwerp	C	1581	1520	1404	783 703
Locate a given hotel in the Municipality of Helsinki	C	1272	1261	1086	908 599
Get full details about a given bar in Florence	D	535	314	196	1257 120
Get full details about a given theatre in Antwerp	D	327	233	132	206 94
Get full details about a given hotel in Helsinki	D	325	229	<u>106</u>	198 146

Table 6.1: VALIDATION PERFORMED IN TERMS OF TESTED QUERIES ON THE FEDERATED NETWORK (UNDERLINED THE BEST TIMES AND IN BOLD THE BEST TIMES BETWEEN PSSM AND PGSM). CONTINUES.

Query/Kind		Times of response (ms)			
		FSSM	PSSM	ASM	GSM PGSM
Locate all bars in a radius of 1 km from a given bar in Florence	E	4755	4671	<u>3976</u>	6725 5745
Locate all theatres in a radius of 1 km from a given theatre in Antwerp	E	1689	977	<u>279</u>	983 696
Locate all hotels in a radius of 1 km from a given hotel in Helsinki	E	1627	1234	<u>285</u>	982 841
Locate all cinemas in Florence or in its immediate nearby	F	307	242	<u>185</u>	275 202
Locate all healthcare centers in Antwerp or in its immediate nearby	F	449	337	264	207 <u>163</u>
Locate all pre-primary schools in Helsinki or in its immediate nearby	F	837	748	588	171 <u>56</u>
Get all events in Florence or in its immediate nearby this month	G	155	154	162	235 <u>62</u>
Get all events in Antwerp or in its immediate nearby this month	G	939	187	124	271 <u>56</u>
Get all events in Helsinki or in its immediate nearby this month	G	920	178	111	159 <u>72</u>
Locate restaurants in the district of Katajanokka, in Helsinki	H	539	351	<u>266</u>	919 365
Locate public transport stops in the small district of Borgerhout, in Antwerp	H	1118	1000	<u>961</u>	363071 342253
Locate a footwear shop in Rifredi (Florence)	H	506	247	230	198 <u>178</u>

Table 6.2: VALIDATION PERFORMED IN TERMS OF TESTED QUERIES ON THE FEDERATED NETWORK (UNDERLINED THE BEST TIMES AND IN BOLD THE BEST TIMES BETWEEN PSSM AND PGSM). CONTINUED.

Query/Kind		Times of response (ms)			
		FSSM	PSSM	ASM	GSM PGSM
Get all events in Rifredi (Florence) this month	I	392	126	82	57 50
Get all events in Katajanokka, in Helsinki, this month	I	623	190	<u>114</u>	397 366
Get all events in the small district of Borgerhout, in Antwerp, this month	I	156	128	118	76 76
Get public transport routes that traverse the district of Rifredi (Florence)	J	1948	907	<u>878</u>	* *
Get public transport routes that traverse the district of Katajanokka Helsinki	J	4100	655	<u>533</u>	* *
Get public transport routes that traverse the district of Borgerhout, in Antwerp	J	1591	1377	<u>1222</u>	* *
Get the full address of the center of Florence	K	583	376	<u>288</u>	1002 355
Get the full address of the center of Antwerp	K	775	769	668	<u>641</u> 656
Get the full address of the center of Helsinki	K	799	757	<u>588</u>	778 768

Table 6.3: VALIDATION PERFORMED IN TERMS OF TESTED QUERIES ON THE FEDERATED NETWORK (UNDERLINED THE BEST TIMES AND IN BOLD THE BEST TIMES BETWEEN PSSM AND PGSM). CONTINUED.

Query/Kind		Times of response (ms)			
		FSSM	PSSM	ASM	GSM PGSM
Get public transport agencies that operate within 5 km from the center of Florence	L	141	121	<u>70</u>	319 206
Get public transport agencies that operate within 5 km from the center of Antwerp	L	183	149	149	54 35
Get public transport agencies that operate within 5 km from the center of Helsinki	L	121	104	73	61 35
Find the shortest path from the center to the airport in Florence	M	2040	1983	<u>1916</u>	2746 1977
Find the shortest path from the center to the airport in Antwerp	M	605	449	352	223 157
Find the shortest path from the center to the airport in Helsinki	M	375	366	338	243 223
Get all kind of services on a Linestring crossing the border on Tuscany and Emilia Romagna	N	2117	1919	–	NA 2004

Table 6.4: VALIDATION PERFORMED IN TERMS OF TESTED QUERIES ON THE FEDERATED NETWORK (UNDERLINED THE BEST TIMES AND IN BOLD THE BEST TIMES BETWEEN PSSM AND PGSM). CONTINUED.



Figure 6.2: ServiceMaps borders overlapping in Italy (left), ServiceMap Areas (right), in Europe.

the actual services. Then, they must be regarded as the best cases, and not as a reference for the operative conditions. Despite of that, in some cases the global solutions have gone in error, and they have not provided any result.

A number of clients have been set up to stress the configurations sending 100 times the same query and measuring the typical response time as reported in Table 1. The ASM column reports the performance in the case of direct query to the correct ServiceMap. FSSM could be considered a worst case, since no strategies for distribution are applied.

The PSSM solution has obtained a mean performance improvement of the 28% with respect to the FSSM, by taking into account all the queries. The ASM solution based on direct query still performs in average the 22,8% better than the PSSM solution. GSM solution has been demonstrated to be under dimensioned in terms of resources to provide interesting results, it provides a mean error of 45% worst with respect to the ASM with a variance in the order of 10^7 . The solution PGSM seems to be the better ranked when the queries are simple as in queries of kind: B, C, D. On the other hand, in most of the cases, the performance obtained by the PGSM solution is not drastically better than the PSSM which provides performance of the same order in most of the queries. Moreover, the PSSM solution provides results also in queries J in which the GSM and PGSM are not capable to provide results to the users providing an error code after 10 minutes (marked as * in Table 1). In those cases, the advantage of using the distributed solution PSSM is evident with respect to the single VM solutions PGSM. Moreover, the PGSM solution, while being the best for simple queries, presents a very large variance with

respect to the ASM reference solution (in the order of 10^8), due to queries that do not produce acceptable results for typical geospatial queries such as those of type A, where services in the nearby of a given point are requested. PGSM has been found to be the most efficient solution for simple queries (queries to which the PSSM also provides a response in a fraction of second), since it does not pay the network overhead of asking to the ServiceMaps, and waiting for responses. This means that the SuperServiceMap may implement more sophisticated strategies to improve the service performance in specific cases. In fact, the solution has also put in place a rule-based module to change the strategies according to the services, area, and query shape.

A remarkable test bed for the proposed approach is also the case in which the geographical region of interest falls across the boundary between two ServiceMaps. In those cases, the SuperServiceMap using PSSM identifies the specific ServiceMaps to be queried and carefully merges responses, composing a response that does not include duplicates. As an example, you can query the SuperServiceMap based on PSSM for all services which are located along the path that joints cities of Firenze and Bologna (approximately 80Km) tracing the path of the Highway A1, E35. Browsing to the following address: [http://www.disit.org/superservicemap/api/v1/?selection=wkt:LINESTRING\(11.248284683089423%2043.77705897010958,11.156274185042548%2043.82958894455573,11.259271011214423%2044.48868359713514,11.338648304801154%2044.49415965702086\)](http://www.disit.org/superservicemap/api/v1/?selection=wkt:LINESTRING(11.248284683089423%2043.77705897010958,11.156274185042548%2043.82958894455573,11.259271011214423%2044.48868359713514,11.338648304801154%2044.49415965702086)) you get 100 of the 697 POIs.

Indeed, data about the Tuscany region are stored in a dedicated ServiceMap, while data about all other covered areas of Italy are stored in another. For such a query, the PSSM provides back the list of services in 1919 ms while querying the PGSM it takes 2004 ms.

6.8 Conclusion

In this chapter we presented a solution for federating Smart City API. The solution aims to solve the requirements that presently cannot be solved by traditional GIS solutions. Thus we would avoid migrating data, providing federation at level of APIs, involving nodes of any size, combining them autonomously so that each of them may decide to join or not, leaving the possibility of having different kind of services, enabling the movements from among federate areas, prevent the access and respect GDPR and data security, combining services, etc. For the design, implementation and validation

of the solution we have used and enhanced the former Km4City API and ontology [117], which has been implemented as the so called ServiceMap. For this reason, the federation of Smart City API has been provided in the so-called SuperServiceMap. The validation has been performed by considering 4 large areas and smart city services in place covering: Tuscany region (3.5 M inhabitants) in the center of Italy, north of Italy and Sardegna island, Antwerp and north of Belgium, and Helsinki and south of Finland. The validation has shown that the solution performs better than single centralized services in many cases, except for the cases in which simple direct queries are performed. This means that the SuperServiceMap may implement a number of more sophisticated strategies to improve the service performance in specific cases. In fact, the solution may also put in place a rule-based module to change the strategies according to the services, area, and query shape.

Chapter 7

Industry 4.0 Synoptics Controlled by IoT Applications in Node-RED

The push towards Industry 4.0 is constraining the industries to work in integrated supply chains. This implies to be open at integrating their production plants with other plants, and to provide access to their data and processes. In most cases, this also means to give access to data and flows to their customers and to perform some synchronization and supervision, and maybe to create integrated control rooms, synoptics and dashboards. This activity is also facilitated by the introduction of IoT solutions with IoT Devices, IoT Brokers, etc., which have a completely different approach with respect to DCS and SCADA solutions often adopted by industries for controlling productions. In this chapter, Snap4Industry with its IoT development environment and framework for implementing the Control and Supervision of Multiple Supply Chains in the view of Industry 4.0 as a service are presented. In particular, this chapter describes the motivations/requirements and the actions performed to extend IoT Snap4City 100% open source platform to comply with Industry 4.0 requirements. The main additions for creating Snap4Industry solution have been on: (i) industry protocols, (ii) custom widgets and synoptics Dashboards, (iii) new MicroServices for Node-RED for

*enabling the usage of synoptics as event driven devices, (iv) automatized process for producing synoptics templates according to GDPR. The research has been founded into SODA RT project.*¹

²

7.1 Introduction

Most of the Industry 4.0 solutions present control systems as DCS (Distributed Control Systems), SCADA (Supervisory control and data acquisition), PLC (programmable Logic controller), etc. PLC are typically directly connected to the production machines and work with a high rate regular periodic loop by reading the status of all variables/inputs, computing control outputs, and finally producing outputs to the actuators (the new setting) [102]. The loop is controlled by precise period since in most cases the computation of the control function depends on the value of the period ($T_2 - T_1$), such as in the case of PID control (proportional, integrative, and derivative control models), etc. The period corresponds to the sampling period and to the acting period. Above the PLC level, DCS/SCADA solutions are typically applied that may put together more than one PLC solution with supervision activities. In most cases, the DCS/SCADA solutions provide some HMI (human machine interface) for the control room operation. In some cases, the PLC are replaced by RTU (Remote Terminal Units), more general-purpose computer-based solutions for low-level control resulting more programmable, while PLC are typically programmable by using the so-called Ladder Diagrams. PLC programming is based on simple logic equations (substantially a set of clauses in the form: If <Condition> then <Action>) visually formalized with the above semantic: reading all data for computing Conditions, computing Conditions and Actions, and finally applying Actions by writing outputs. There are streams of discussions in which DCS is considered more process-oriented while SCADA more event-

¹This chapter has been published as “Industry 4.0 Synoptics Controlled by IoT Applications in Node-RED” in *Proc. of the IEEE iThings, November 02-06, 2020* [35].

²*Acknowledgments:* The authors would like to thank SODA project of ALTAIR, and 5G partners as Wind-3, ESTRA, GIDA, Prato, etc., for which we have developed a part of the solutions described in this chapter. Snap4City is 100% open source and has been founded by European Commission Select4Cities PCP project under the grant agreement No. 688196. Snap4Industry, Snap4City and Km4City are 100% open source technologies and the platform of DISIT Lab can be accessed at <https://www.snap4city.org>.

driven, but in reality, the provider of those solutions are improving them thus making top level DCS and SCADA products very similar. In fact, in most cases, they are used as interchanging terms when a network of control systems is coordinated to work together, and data collected are stored into some database for historical data observation mainly. In addition, DCS/SCADA also provide suitable control graphic interface which represent the plant as synoptics. The synoptics may represent the status of the plant in real-time with icons, animated icons, numbers, graphs, etc., and in some cases, they are also interactive. This means that the same synoptic elements can be used to send commands to the control network for changing parameters, turning on/off some switch, controlling valves, etc.

In large industry plants, multiple DCS/SCADA controllers are present and hierarchical solutions are needed for: (i) collecting data with different rates, both inputs and values of the control parameters also acted by the users in the control room, (ii) creating the history of the data values and taken decisions, (iii) monitoring the status of the system and eventually generating some higher level alarms and actions, (iv) representing the higher level status of the system with some graphical representation and synoptics. At low level, the DCS/SCADA typically communicates with PLC and devices through MODBUS, Profibus, CanBUS, FieldBus, HART, AS-I, and other low-level protocols, while at a higher level, they may communicate through protocols such as OPC/OPC-UA, DDS or OMG, MTConnect, etc.

In the context of Industry 4.0 applications, IoT solutions are going to be integrated in the factory, and among different plants and factories, playing a sort of role of glue among the different processes, by offering data/status and actions (see Figure 7.1). Among them, the devices such as IoT Edge and IoT Gateways allow to collect sensor data from the plant and close the loop with some actuator. In addition, similarly to RTU they can send data towards the DCS/SCADA or on cloud for telemonitoring and control [126], [106]. This approach sees the usage of Node-RED / node.js solutions applied in many cases [129], [119], to implement more flexible solutions and to take into account in the plant also of other external parameters in the connection of the different plant machines and equipment. In fact, Node-RED can support protocols such as ModBus, OPC/OPC-UA, MQTT, AMQP, NGSI, I2C, TCP IP, RS232, etc., and can be used on hardware based on ARM, i86, etc., and on many operating systems (Windows, Linux, Raspberry Pi, etc.). Thus, a mixt of IoT and Industry 4.0 and also many home-automation pro-

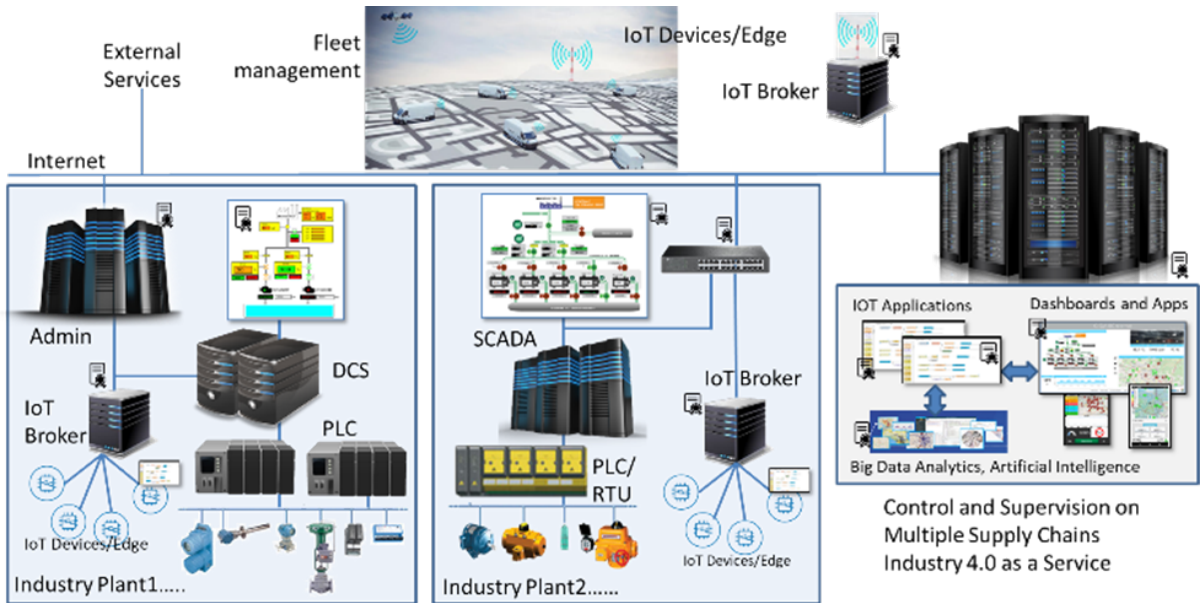


Figure 7.1: The integration of Industry 4.0 with the support of IoT, Big Data and cloud technologies

protocols (ALEXA, Philips HUE, TP Link, KNX, ENOCEAN, ZigBee, ...) are supported on Node-RED. Thus, Node-RED being based on node.js engine in JavaScript, is probably one of the best solutions for system integration. On one hand, Node-RED may be not very effective in terms of performance since the round trip of reading, computing and acting presents some limitations in terms of performance with respect to RTU developed in native code as C++/C, Assembly, and Python, and even wrt PLC. On the other hand, Node-RED approach can be a suitable solution for implementing IoT Gateways mixing up different protocols and data sources and thus for rapid prototyping and when high performances are not requested. In fact, there are a number of industrial solutions that use Node-RED as RTU gateway with control logic such as SmartTech [<https://netsmarttech.com/page/st-one>], for running H24 processes in production lines of relevant customers (see also <https://iotobox.com/>).

In this chapter, Snap4Industry with its IoT development environment and framework for implementing the Control and Supervision of Multiple

Supply Chains in the view of Industry 4.0 as a service are presented. In particular, this chapter describes the motivations/requirements and the actions performed to extend IoT Snap4City platform to comply with Industry 4.0 requirements. Snap4City is 100% open source and it is available at [<https://www.snap4city.org>], [31], whoever interested can download and install it locally or on a cloud without any fee. The main additions for creating Snap4Industry solution have been: (i) the addition of a number of industry protocols, (ii) the possibility of producing custom widgets and synoptics and connecting them respecting security into a Dashboard system, (iii) the addition of a number of new MicroServices for Node-RED for enabling the usage of synoptics as event driven I/O devices, (iv) the usage of WebSocket secure for the communication with custom Synoptics and Widget for dashboards, (v) the automated process for producing synoptic templates and instances according to GDPR. Please note that the additions have been incorporated in the Snap4City platform to enlarge the coverage of the solution.

This chapter is structured as follows. In Section 7.2, the requirements of a solution for control and supervision of multiple supply chains from cloud, in the view of Industry 4.0 as a service is presented. Section 7.3 describes the former Snap4City architecture and in Section 7.3.1 the changes and additions performed to cover the industry-derived requirements are presented. In Section 7.4, the development environment for creating synoptics based on SVG graphics is presented. The aim was to enable the creation of custom widgets and synoptics into dashboards without the need of manual intervention on the server side, neither on programming. Everything had to be done through the dedicated Web interface. Section 7.5 presents the integration of custom synoptics with the visual programming in Node-RED and thus the MicroServices developed based on WebSockets and their performance. Conclusions are drawn in Section 7.6.

7.2 Requirements for Industry 4.0 Applications

As described above, the aim of the research reported in this chapter is to design and implement a solution for covering the business logic and the human machine interaction of Industry 4.0 high level applications in the context of high level control even among different factories working on the

same or in anyway connected supply chain. Therefore, the main needs of the solution that can cover Figure 7.1 can be formalized as follows. Please note that the context is the usage of the solution for enabling the control and supervision of multiple supply chains based on Industry 4.0, by using IoT, cloud, machine learning and big data technologies, thus implementing in some sense the Industry 4.0 as a Service, I4aaS.

The solution has to support:

1. functionalities which are present at level of RTU, SCADA and Cloud for telecontrol, from Control Room and remote devices. The support may consist in the integration of the functionalities of these elements and/or the interoperability with already present solutions.
2. programming the business and control logics with a uniform visual program, which could be used at low level as well as in the integration on cloud. DCS and SCADA present their own programming languages for implementing business processes, while the remote control or the supply chain integration has to be performed through Web-oriented programming. Most IoT development environments are mainly based on developing applications for IoT Devices, or for IoT Edges with native programming languages as it happens in the RTU. The adoption of a uniform programming model (at least on Edge and Cloud) provides more flexibility in moving the processes and reusing the code.
3. a large number of different protocols covering classical DCS/SCADA but also IoT, Industry 4.0, home automation and smart city. This means that the language has to be capable to perform requests using many different available protocols [72], such as pull protocols to obtain data (e.g., Rest Call, Web services, FTP, HTTP/HTTPS, etc.), and push protocols to receive data via data-driven subscriptions (WS, MQTT, NGSI, COAP, AMQP, etc.).
4. direct communication among business logic processes which can be executed at level of IoT Edge, as well as on Cloud/servers. The possibility of having processes on cloud would guarantee the possibility of exploiting high performance computational capabilities for example for predictions, advanced planning, and integration along the supply chains.

5. registration of IoT Devices which are provided via some IoT Broker, regardless of their protocols. This feature is strongly relevant for the registration of IoT devices integrating the control in the industrial plants. In fact, the data coming from the former generation of production machine are typically insufficient for controlling the production process and distributed supply chain among different plants. Thus, additional IoT devices are added and have to be integrated in the higher-level control and remote monitoring. This will allow the other partners of the supply chain to monitor the previous phases of production and thus they can be aware in real-time when their activity will be needed. For example, a DCS would present to the higher-level control a subset of data, etc. The data collected from those devices have to be kept strictly private and in accordance with the GDPR guidelines [33], [131].
6. exploitation of big data storage to save and retrieve historical data related to the whole supply chain but managing the access to them according to the GDPR [37]. This implies that each factory has to control the access right to its data and provide the access to the other industrial partners of the same supply chain in a controlled manner. The whole data can be used for creating a knowledge base expert system for computing KPIs (key Performance Indicator) and deductions.
7. applications activated by event driven protocols as well as supporting pull protocols. Applications have to integrate paradigm of data driven with that of functional and have to be capable to exploit computational resources and data on local/premise (e.g., IoT Edge) as well as on cloud. This means that from IoT Edge it must be possible to execute computations on Cloud or locally.
8. execution of data analytics like business intelligence and machine learning solutions on powerful processors thus exploiting historical data, big data store on cloud and on premise if any, for example for computing predictions, long-term settings for production plan, detection of anomalies, identification of maintenance actions, etc.
9. exploitation of External Services, for example for collecting: costs of energy/gas, environmental data, contextual marketing data, fluctuation of contextual data, etc.
10. creation of local dashboards on IoT Edge and Gateways, which have

to be capable to report and interact with the IoT Applications on local basis.

11. production and exploitation of synoptics for graphic rendering of data and plant status and for collecting actions according to the event driven paradigm. They have to be integrated and controllable directly from the IoT Applications.
12. creation of powerful dashboards (e.g., user interface) for remote control (telecontrol, telemonitoring, control supply chain, etc.) with respect to those that can be produced on Edge computer. Dashboards for remote control have to be capable to represent data and to collect actions from the users, such as new events for settings/controls the plant. Communication with Dashboard has to be event driven, secure and according to GDPR. Data visualization should be expressive enough, including values in real time data, time trends, histograms, pie charts, Kiviat, barlines, multitrends, etc., but also actuations as buttons, switch, dimmers, selections, etc.

In addition, the solution has to satisfy a number of non-functional requirements, such as: robustness (in terms of availability and fault tolerance), scalability (to be capable of serving from small to very large businesses with corresponding volumes of processes per second), security (authentication, authorization, secure connection, etc.), full privacy respect (compliance with data privacy, according to the GDPR), and openness in terms of being an open source and having the possibility of adding new modules and functionalities, etc.

In this chapter, we are focusing the attention on requirements 11 and 12 related to the accessible functionality of providing integrated synoptics also connected with IoT Apps. Therefore, in this chapter, the evidence of the changes performed on addressing SVG with WebSocket into the infrastructure has been reported. The possibility of addressing Synoptics has led to the creation of a number of new MicroServices that were not present in the Node-RED library, and thus a huge improvement for the community. Most of the other requirements mentioned above are verified by the former Snap4City solution. This fact is also remarked in the next section and in [49], [37], [38].

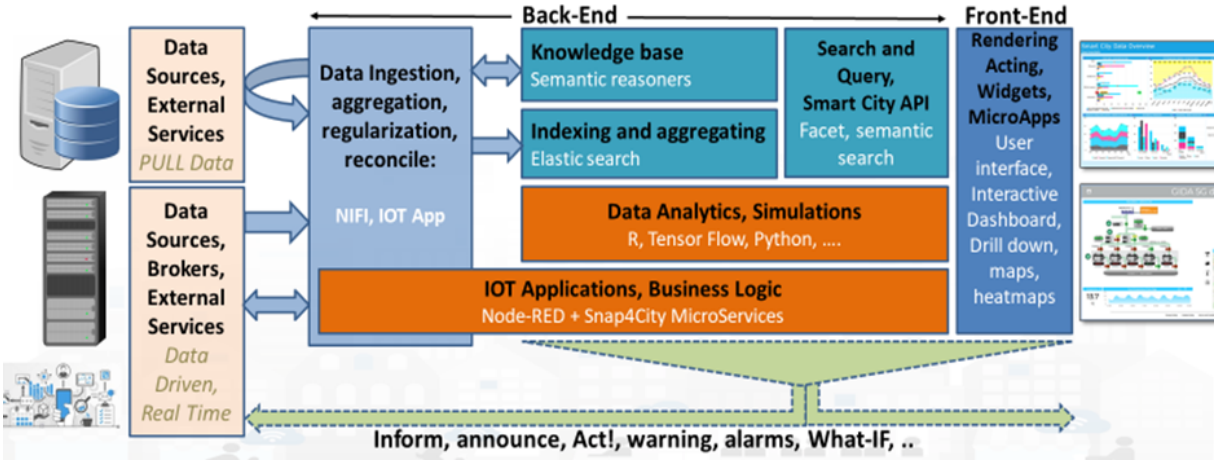


Figure 7.2: Snap4City general architecture

7.3 Snap4City vs Snap4Industry Architecture

In this section, the general architecture of Snap4City is presented in Figure 7.2, and the specific evolution and solution for addressing the above-mentioned industry 4.0 aspects (and thus Snap4Industry) is presented in Figure 7.3.

Snap4City has been developed in response to the research challenge of satisfying all the Select4Cities PCP requirements [<https://www.select4cities.eu/>]. It allows the creation and management of user communities to collaboratively (i) create IoT Solutions, (ii) exploit open and private data with IoT/IoE, respecting GDPR (General Data Protection Regulation European guidelines), and (iii) create/use processes and IoT Applications that could run on Edge, Mobile and cloud, with the capability of interacting each other and with users via messages, Dashboards and Applications. In Snap4City, a special attention has been devoted to the creation of an integrated development environment for IoT Apps (cloud and edge), based on Node-RED, with dashboards, and supporting data analytics and resource sharing, thus minimizing any required technical expertise from programmers. To this end, a large number of visual elements developed as MicroServices for the IoT are present. The Snap4City developed tools provide assistance during the development life cycle, from data gathering to dashboard production, as well

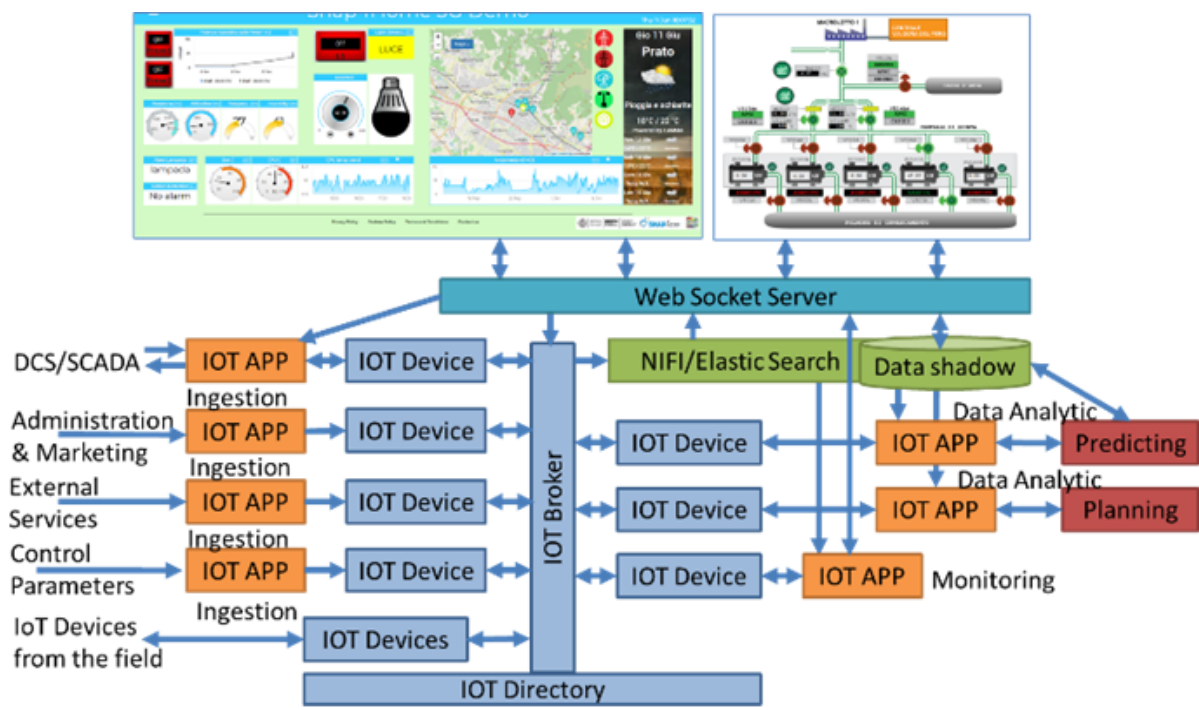


Figure 7.3: Snap4Industry: example of Functional Architecture. On the user interface: Dashboard with custom widgets and Synoptic panels

as in IoT App development, in the sharing of results with other developers and in the community, as self-assessment, together with security support, as described in [37]. In Snap4City, data provided from External Services or operators are collected in pull via REST calls using scheduled processes as IoT App. IoT Apps are Node-RED Applications which can be executed on cloud as well as on IoT Edge. The collected data are regularized/reconciled, according to the Km4City Knowledge Base [31], and then pushed into the Big Data Cluster storage. The reconciliation process allows for the reconnection of data to entities already in place, for example, by connecting new instances with the already in place elements of the knowledge base.

The data arriving in push (data-driven) are managed by several IoT Brokers, to which the Apache NIFI distributed cluster is subscribed. The NIFI process is in charge of performing a regularization/reconciliation task on the data, according to the Km4City Knowledge Base, and then it pushes the resulting data into the Big Data Cluster storage, while indexing and thus creating the Data Shadow. The Big Data cluster storage includes an RDF store for semantic reasoning, and an ElasticSearch Index. The whole solution presents textual indexes on all fields, semantic indexes, and elastic aggregations to perform advanced “faceted” filtering queries [31].

The IoT Apps can be executed on-cloud or on-Edge. When IoT Applications are executed on IoT Edge, which is located on the field, it may directly communicate with the IoT Apps or Dashboards on cloud or by means of IoT Brokers (to which all of the others can be subscribed). On such grounds, without the risk of losing generality, in Snap4Industry, the IoT Applications can be obtained as follows:

IoT App = Node-RED + Snap4City MicroServices

The IoT App exploits the basic nodes of Node-RED Node.JS plus Snap4City MicroServices, which have been extended from the smart city to Industry 4.0. The Snap4City solution has implemented a large set of MicroServices which provide an easy and formalized access to all the Smart City services [38]. The Snap4City MicroServices are distributed into two official libraries of Node-RED nodes by the JS Foundation portal. The two libraries are dedicated to final users (basic) and to developers (advanced). The version for Developers (to be installed on top of the basic version for final users) presents several nodes/blocks that can accept complex JSON message inputs to create strongly dynamic IoT Applications. Both Libraries of Snap4City Nodes can be installed in any Node-RED tool on any operating system: Linux,

Raspberry pi, Windows, etc.

In the following subsections and sections, the main improvements from former Snap4City and the present solution Snap4Industry are described. The main changes have been: (i) the possibility of producing custom widgets and synoptics and connecting them respecting security, (ii) a number of new MicroServices for enabling the usage of synoptics as event driven devices in/out, (iii) the usage of WebSocket secure for the communication with Synoptics, (iv) the automatized process for producing synoptics templates and instances also supporting GDPR. Please note that the additions have been incorporated in the Snap4City platform to enlarge the coverage of the solution.

7.3.1 Snap4Industry Architecture

In Figure 7.3, an instance of the functional architecture for Snap4Industry is presented. The example reported is mainly focused on creating an Industry 4.0 supervisor for control room and remote control from cloud (as in Figure 7.1) with the capability of computing production plans and predictions. Both production plans and predictions may be performed several times per day. The data analytic computations need 15 minutes to be performed and produce resulting data for the next 48 hours in advance. The predictions and plans are computed on the basis of the data coming from the:

- DCS/SCADA which includes the status of the industrial plant (different lines of productions) including the status of the physical storage of the products and production stock silos
- Administrative data which are the orders and the business targets of the successive days.
- External Services which are energy costs, prime matter cost, marketing context, real time data from the commercial chain, data from the delivering fleets, etc.
- Control Parameters are a set of parameters which are useful for the planning and prediction algorithms. For example, the quality of production, some target key performance indicators, the eventual flag to put offline a segment of the supply chain, etc.

In Snap4Industry, IoT Apps are realized as Node-RED processes or as Dockers containers on cloud or as IoT Edges (in RTU as Raspberry Pi or

Linux). Data are received in push and/or pull, and almost every data message with several attributes is considered an IoT Device instance. To this end, the IoT Devices must be registered, and their data structure formalized. In Snap4City, the IoT Device formal registration is performed by means of the IoT Directory [131], [37].

DCS/SCADA data messages are received in push and converted into an IoT Device instance via an IoT App which is posting them on IoT Orion Broker Fi-Ware and from that to the Data Shadow Storage implemented by using NIFI/Elastic Search. Administrative and Marketing data coming from the administration of the factory are collected by accessing to the databases, e.g., Oracle, DB2. This information may include the new target production for the factory, that are the orders, and it is converted into an IoT Device instance by using an IoT App. Every time an IoT Device instance is created it is marked with a timestamp. This allows to recover the acquisition time instant and to create into the Data Shadow the history of the acquisitions and execution of Data Analytics. External Services are acquired by means of an IoT App and converted in IoT Devices instance and saved in the storage. A similar operation is performed for the Control Parameters. Once the above described data inputs are collected (with different rate of update), the production plan and predictive Data Analytic can be computed. The execution of Data Analytics algorithm is performed by calling them from IoT App and executing Data Analytics in Docker containers allocated on cloud. The resulting prediction and production plans can be saved into the storage or saved as IoT Devices instance as well, and accessible from Dashboards.

All the processes (IoT App, IoT Brokers, Web Sockets, etc.) are activated/fired by the arrival of new messages/events and thus are data driven processes. The data messages in input are temporized differently. For example:

- DCS/SCADA can produce a new dataset every second or more and the registration on the new data on broker leads to send the message to NIFI that in turn activates the IoT App of monitoring which produces the KPI to be shown on synoptic via Web socket. With the aim to perform high level control, to save all messages may not be needed; thus, a message every 30-15 second could be enough. This may also depend on the velocity of the production process phase.
- Administrative data sources produce asynchronous data regarding the

new production orders. So that, typically, they may provoke some recomputing for planning and prediction a few numbers of times per day max.

- External Services are typically sampled according to their rate of change, for example many times per day.
- Control Parameters need to be produced every time a new computation of production plan and/or prediction is performed.
- IoT Devices from the field may be sensors and actuators providing data and accepting to perform actions completely asynchronously with respect to the DCS/SCADA processes. They are typically event driven and may adopt typical IoT protocols towards IoT Brokers registered both on IoT Directory.
- Dashboard can be used to: (i) just monitor the production or the status and in this case they can be given in access to the all partners of the supply chain, (ii) produce events/settings for the general integrated system (e.g., the pressing of a button, the setting of a value), and in this case the access has to be strongly regulated. Thus, the pressing of the button could be performed several times per minute, while the effects on production plan and prediction is relative. The new value is produced as a Web Socket message [127] for a corresponding device that reaches the broker, which in turn is sending the message to all subscribed processes and IoT App. On the other hand, the actions on the dashboards can produce some action on DCS/SCADA as well.

In the context of the above example, it is particularly relevant to save the data adopted for the execution of the predictions and plans since they may need to be executed again and again with small differences on the parameters, and for improving predictive algorithms based on machine learning (also activated on Docker Containers by IoT App). In addition, a number of dashboards are produced for comparing KPI as computed by: (i) planned results and setting of control, (ii) actual production, and (iii) predicted values and controls.

7.4 Synoptics Development Environment

In this section, the introduction of synoptics and their integration into the Snap4City solution is presented. This new functionality has constrained to change the architecture as described in the following. A synoptic panel is a graphic representation of the plant, in which graphic elements may present data, animations and can be interactive. For example, to turn on/off a switch, change a valve status, move slider, insert a value, etc. In Figure 7.4, an example is reported. Typically, these graphic layouts are designed in SVG (Scalable Vector Graphics) or other graphics approaches. The advantage of SVG is its integration on Web browsers. The resulting graphics factory layout (synoptic) is typically addressed by proprietary solutions which allow to connect them to DCS and SCADA with proprietary tools. In Snap4Industry all the solution is based on Open Source components and the new developments have been also released on Affero GPL on GITHUB/-DISIT, Snap4City.

In the context of IoT solutions, the possibility of connecting the Synoptics with IoT Application has been identified very interesting to cover some of the above-mentioned requirements. In the DCS/SCADA, the usage of the SVG is mainly confined to create synoptics. On the other hand, the power of SVG could be also exploited for creating single Widgets in Dashboards: a single switch, a lamp, an alarm blinking alarm, etc. (as reported in Figure 7.3 on the left). Thus, our approach started with the possibility of using the design of SVG graphics as a generic tool for creating Custom Widget Templates that can be instantiated to create both single widget and complex Synoptics for Dashboards. At the instantiation time the Template is used to create an actual Widget defining the connections with specific WebSocket streams according to Figure 7.3, to play the role of event driven graphic interface.

In Figure 7.5, the procedure needed to create an SVG based Custom Widget into a Dashboard for IoT Control is reported. The production starts with the creation of the SVG graphic design by using the Inkscape open source SVG Editor (<https://inkscape.org/news/2020/05/04/introducing-inkscape-10/>). The production of SVG with the editor has to follow a few of guidelines to be sure that the modality in which variables are defined into the SVG are in effect compatible with the successive steps and Snap4Industry tools. The guidelines are accessible on: <https://www.snap4city.org/595> The SVG may: contain animations, visualize effects according to the value of some messages received, collect data from the users, be freely zoomed, provide

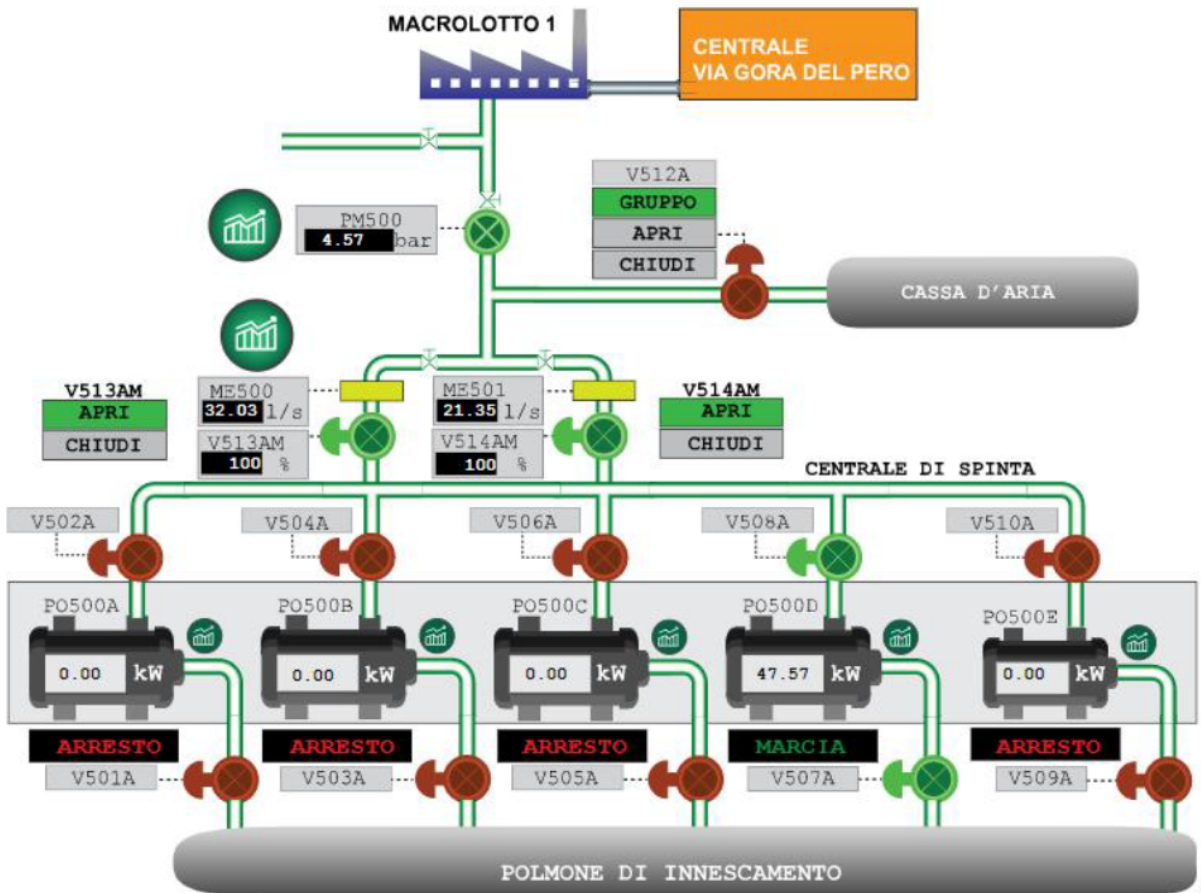


Figure 7.4: An example of synoptic, GIDA Plant in Prato. Demo also accessible on <https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MjE4Mg==>

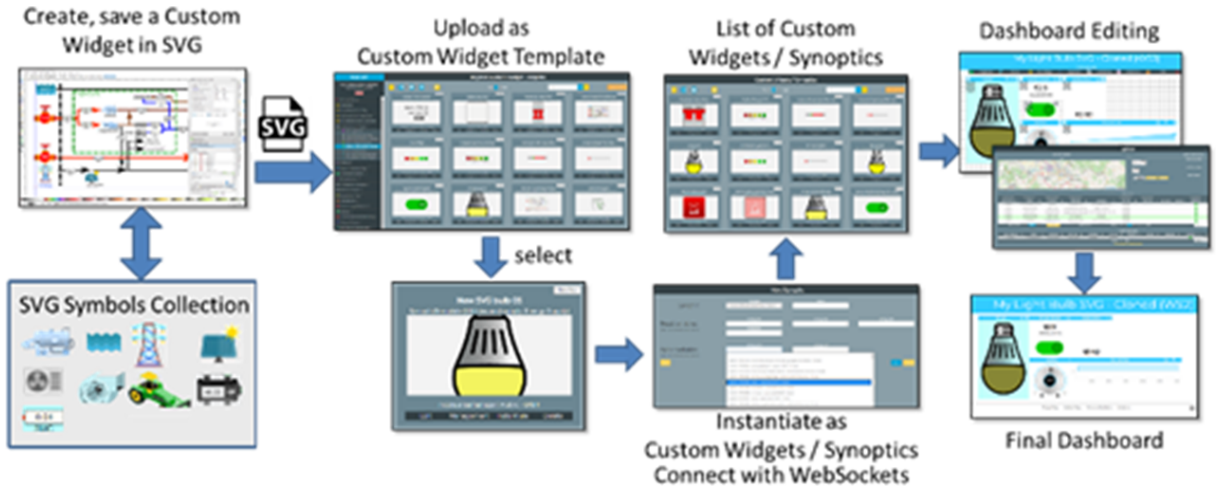


Figure 7.5: The process for producing Custom Widgets into Dashboards for graphic nitrated IoT Control

transparent background, etc. The user may access to the SVG of the Custom Widget Templates to change them, and re-upload as modified / improved versions as well. In addition, they can be created by exploiting a large part of the market and open libraries of SVG symbols. Once the SVG of the Template is created, it can be uploaded into the Snap4City service which parses the file to identify the SVG variables in Writing and Reading. They can be integer, float, status, and strings. Thus, a Custom Widget Template is created as private of the user, according to GDPR, and can be shared with other users or published for all only by the owner.

A Custom Widget Template can be instantiated for creating specific Custom Widgets/Synoptics. To this end, the identified variables in Reading and Writing have to be associated with some real variable in the system or directly connected with an IoT App. The processes and the different kinds of connections are discussed in the next section.

Please note that, Custom Widget/Synoptics are considered in the Snap4City environment a HighLevelType. So that, it is possible to select them as internal elements in the composition of a Widget for the Dashboards. Then the dashboard widget can be resized, coloured, etc.

7.4.1 GDPR Functionalites and Aspects

Please note that each SVG Template, and each of its instances as Custom Widget/Synoptic, as well as the dashboard which is containing it and the related variables connected to the Synoptic, once created are private of the user which has created them. On the other hand, according to GDPR, the user can make them public or can give them in access to other users. When a Dashboard containing a Custom Widget using some MyKPI is made publicly accessible, all its elements have to be public as well otherwise the authentication will be requested to access. In Snap4City/Industry platform, the grant authorization can be provided to Organizations (all users belonging to), Group of Users, and to single users. The accesses can be audited any time and the grant authorization can be revoked at any time, making the distribute control solution developed on the platform secure and respectfully of the privacy of each industry in sharing their data only to their specific partners. In addition, all the connections are performed via a WebSocket Secure, into HTTPS, thus implementing end-2-end secure communications even on Synoptics.

7.5 Controlling Synoptics From IoT Apps

In the connection and interaction from Synoptic (read as SVG Custom Widgets) and the rest of the infrastructure, including IoT Apps and storage, there are 3 main Cases as depicted in Figure 7.6.

In Case 1, the data can be collected from the Synoptic via WS and saved (connected in writing) or read (connected in reading) directly into/from the storage (namely MyKPI or Sensors data for Snap4City terminology). This means that for changing the light intensity the IoT App writes into the storage and it is the storage that provokes an event in the WS for changing the lamp status. It implies that the velocity for closing the loop (moving the slider and observing the change on the lamp) depends on the velocity of writing/reading on/from the storage. Please note that when the slider is moved a new value is saved into the database of MyKPI and an event is created into the IoT App and to the WS of lamp connected. This approach increases a bit the velocity of the round trip. The IoT App uses in this case specific storage Nodes of Snap4City to interface with WS and Storage. For this specific case, the presence of the IoT App is optional since the WS of the Slider can be connected to same MyKPI variable of the Lamp, thus closing

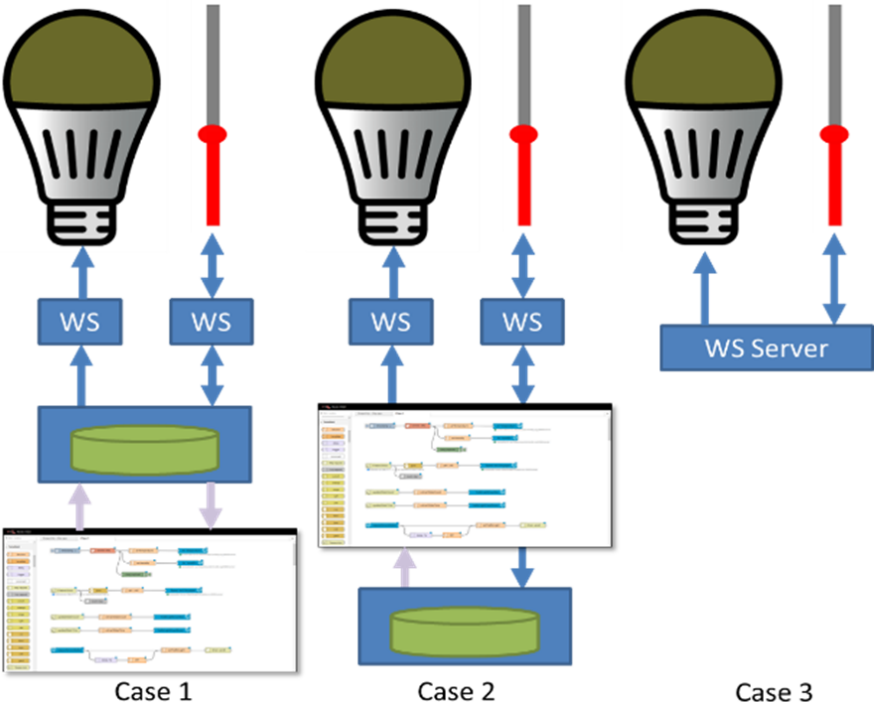


Figure 7.6: Connections with SVG Custom widget cases

the loop on storage cloud. With the aim of assessing performance, a series of messages have been saved into the MyKPI and thus they are automatically posted on WS for both slider and lamp (in fact the slider is producing a data but also reading the value if changes and move the slider accordingly). Thus, it has been possible to sustain without loss of messages up to 5 changes per second. Which means 10 WS messages per second (change the lamp and change the slider). Please note that the round loop is closed passing from the Web page to the WS Server via Internet and on IoT App on cloud, which generates the message. This means that the measure has been performed in real conditions and at the same time is influenced by the capabilities of the Internet connection.

In Case 2, the Synoptic is connected with the WS Server which can be read/written directly from specific Nodes of Snap4City for communicating with Synoptics. This modality has been added to speed up the process and the limit is imposed by the capabilities of WS server and Node-RED to manage event driven processes. In this case, it is possible to save the value in the storage but is not needed to read it again to close the loop that is closed into the Node-RED IoT App on cloud. With the aim of assessing the performance, the IoT App has been subscribed to the WS of the Slider and once the value it sends the value to Lamp and again on the slider with some random changes, so that to create an infinite loop. From the measures performed the round trip is performed at maximum of 20 changes per second without missing data, which means 40 WS messages per second (change the lamp and change the slider). Please note that, also in this case, the round loop is closed passing from the Web page to the WS Server via Internet and on IoT App on cloud, which generates a random value on message at each round.

In Case 3, the Synoptics are connected with the WS Server that directly shortcuts the reading with the writing and thus the loop is closed in nearly real time. Please note that the round loop is closed passing from the Web page to the WS Server via Internet without the IoT App. The maximum number of messages per second without loss is about 2500 messages per second (3 sender and 1 receiver). On the other hand, this modality is only used for user interaction and may be used for closing the loop from two WS belonging to different parts of the supply chain, for example, an action in the plant and a visualization toward a different partner remotely connected.

Therefore, for connecting synoptics with the IoT App, the MicroServices

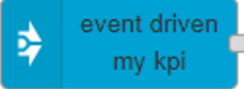
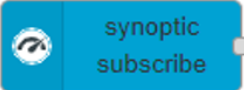
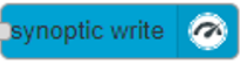
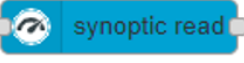
	Send a message into the IoT App once a MyKPI is written.
	Send a message into the IoT App once a Synoptic variable is written.
	Send a message from the IoT App to a Synoptic variable.
	Read the last message of a Synoptic variable.

Figure 7.7: MicroServices of IoT App for Synoptics

(of the Snap4City library on official Node-RED Palette) reported in Figure 7.7 have been implemented. In addition, the Synoptic variables can be changed by connecting their corresponding WS to MyKPI or Sensor values of the Snap4City Snap4Industry platform directly in configuration/instantiation phase, see Figure 7.5.

In the development of solutions, the synoptics may use a mixt of the above-mentioned cases according to the needs in terms of velocity and of saving data changes on storage. In extreme IoT Edge configurations, the IoT App could receive IoT messages from IoT Brokers in push and the data can be directly sent on Synoptic without saving them on storage, or just saving a part of them by subsampling values, thus saving storage and at the same time making possible the data collection for machine learning and predictions.

On the other hand, in most of the cases for supply chain integration, it may need to have together: fast events, machines and productions (for example those for manufacturing), and IoT devices just aside to glued. Slow processes for example may be those for: moving material among plants, heating processes, chemical production, electrolysis, milling, etc. In classical DCS/SCADA slower signals are saved in the storage more often than what is really needed, thus limiting the collection of historical data which is mandatory for Big Data algorithms and predictions. On the other hand, the presented event driven approaches can allow to perform the integration of

the different rates with the aim of high-level supervision and control feasible. When the needed rate is not very high as in the DCS solutions closing the loop aside the numerical control.

7.6 Conclusions

The strong push towards Industry 4.0 has constrained many industries to work in integrated and interoperable supply chains. In most cases, it implied to open at the integration of their production processes with those of other industries in the supply chain, and thus with their customers and suppliers. This fact led to be capable to give access to data and flows and to create integrated dashboards for their customers and to perform some synchronization and supervision, and possibly to create integrated control rooms, synoptics and dashboards, and allow to collect data for big data analytic, business intelligence, machine learning on production processes. This activity has been in most cases facilitated by the introduction of IoT solutions with IoT Devices, IoT Brokers, etc., which provide a completely different approach with respect to DCS and SCADA solutions usually adopted in the industry for controlling their local productions. In this chapter, Snap4Industry with its IoT development environment and framework for implementing the Control and Supervision of Multiple Supply Chains in the view of Industry 4.0 as a service has been presented. In particular, in this chapter we have described the motivations, requirements and the actions performed to extend IoT Snap4City 100% open source platform to comply with Industry 4.0 requirements. The main additions for creating Snap4Industry solution have been on: (i) a number of industry protocols, (ii) tools for producing custom widgets and synoptics and connecting them respecting security into a Dashboard system, (iii) a number of new MicroServices for Node-RED to enable the usage of synoptics as event driven devices in/out, (iv) the usage of WebSocket secure for the communication with custom Synoptics and Widget for dashboards, (v) the automation of the process for producing synoptics templates and instances according to GDPR. The research has been founded into SODA R&D project and tested on chemical plant of ALTAIR, one of the main chemical industry in Italy, and into 5G national project in GIDA plant for water depuration. The specific case of ALTAIR has not been reported for security reason and since it is covered by industry secret.

Chapter 8

Real-Time Traffic Estimation of Unmonitored roads

Due to growing cities, the real-time knowledge about the state of traffic is a critical problem for urban mobility, and the road-segments traffic densities estimation influences the efficiency of fundamental smart city services as smart routing, smart planning for evacuations, planning of civil works on the city, etc. Nevertheless, the traffic-related data from navigator Apps (e.g., Tom-Tom, Google, Bing) are too expensive to be acquired. Also, the traditional sensors for the traffic flow detection are very expensive, and they are usually not dense enough for a correct traffic monitoring. In order to overcome such problems, there is a space for low cost and fast solutions for dense traffic flow reconstruction. We propose a real-time visual self-adaptive solution to reconstruct the traffic density at every location of a wide urban area leveraging the detections from a few fixed traffic sensors deployed within the area of interest. A such method is based on fluid dynamic models to simulate macroscopic phenomena as shocks formation and propagation of waves backwards along roads. Such physical constraints are applied to a detailed street graph which is enriched by specific parameters representing a weight in terms of traffic road capacity. The weight's assignment has been estimated by a stochastic learning approach at each time slot of the day. The accuracy of the proposed model comes from the error

between the reconstructed traffic density and the measured values at the sensor position by excluding each sensor iteratively and reconstructing the flow without it. The proposed reconstruction model has been created by exploiting open and real-time data in the context of Sii-Mobility research project by using Km4City infrastructure in the area of Florence, Italy, for its corresponding Smart City solution.^{1 2}

8.1 Introduction

The number of vehicles is rapidly increasing if compared to the development of the surrounding urban space, and the knowledge about the state of traffic in real-time is a critical problem for urban mobility. Short-term traffic flow prediction plays a crucial role in intelligent transportation in terms of route planning, traffic management and sustainable mobility. Thanks to the today's technologies, it is possible to collect real-time data to detect the state of the traffic in some specific areas. The traditional methods for traffic flow measured via sensors are very expensive and scattered in the city, not enough dense for the above-mentioned applications. Using only stationary sensors we have a reduction of cost for a possible application since they are only positioned in a few number of ways. Nevertheless, in most cases a simple network area as freeways or rings, where incoming and outgoing roads have a limited number, is monitored by the studies that consider only sensors in fixed locations, see for instance [137] [130]. Our approach is different since it considers a complex network area as a city area by means of the only usage of fixed sensors. Nevertheless, other studies both use stationary sensors and wi-fi data from mobile applications to analyze the state of traffic in an urban network. In [5] a solution to the road traffic estimation gathering the data from both mobile data sources and fixed sensors is outlined via various inferences and probabilistic determinations of traffic-related characteristics from the input data samples. In [136] it is proposed a real-time road traffic prediction with spatiotemporal correlations based on data coming from both

¹This chapter has been published as “Real-Time Traffic Estimation of Unmonitored roads” in *IEEE-DataCom'2018, Athen, 2018* [43].

²*Acknowledgments:* The authors would like to thank the MIUR, the University of Florence and companies involved for co-founding Sii- Mobility national project on smart city mobility and transport. Km4City is an open technology and research of DISIT Lab. Sii-Mobility is grounded and has contributed to Km4City open solution.

the loop detectors and the taxis to predict the speed and volume of traffic in a business district of an urban area.

Often, the traffic state analysis is related to the monitored area in terms of short-term traffic flow prediction on fixed points and no information is given where sensors are not located [85] [107]. Of course, the increase of the number of sensors can help on this purpose but the cost for a possible application would necessarily increase. On the other hand, the usage of data coming from navigator Apps (e.g., TomTom, Google map, Waze) could be very expensive. Moreover, the traffic flow data provided by navigators are generated on the basis of mobile applications and they provide in many cases large discontinuities, such as closer road-segments have no traffic and a congested situation in 20 meters of distance. This kind of data has to be regularized to be used in routing applications and planning. For example, in [84] a smartphone-based crowd sensing system for traffic regulator detection and identification is proposed, where the data are gathered from the handheld devices located within the running vehicles, also leveraging their Global Positioning System (GPS) sensors. In [109] a deep Restricted Boltzmann Machine and Recurrent Neural Network architecture is utilized to model and predict traffic congestion evolution based on (GPS) data from taxis.

For this reason there is the space for: (i) low cost and fast solutions for dense traffic flow reconstruction from scattered data coming from a limited number of street sensors spread on the city; (ii) autonomously deployable solutions that do not require the engagement of third parties such as taxi drivers, public transport companies or travelers; (iii) open solutions that can be implemented without any restriction deriving from patents or similar; (iv) solutions that predict events that locate very far both in the time and in the space dimension; (v) solutions that do not rely on the simplicity of the road network; (vi) solutions that display their results in a user-friendly manner; (vii) solutions with clear declared precision rate. To this end, our goal is to develop an algorithm for the traffic flow reconstruction monitoring a complex network as an urban area taking into account fixed sensors only, and from them to perform the traffic flow reconstruction in the unmonitored areas to evaluate the traffic density in the urban network. Many cities have public (and open) traffic sensors (as cameras and spire road sensors), so the present chapter can be a useful tool for their sustainable mobility. The proposed method is based both on differential equations and physical constraints ap-

plied to the basis of a detailed street graph which is enriched by several features. The physical structure of the present model is based on the open geodata taken from the Open Street Map (OSM) database which allows us to obtain a restricted description of the network graph modelling the selected area of observation. To achieve our goal such data have to be opportunely adapted to the network computation structure. A stochastic learning approach has been adopted to estimate the weights representing the street segment capacity at each time slot of the day. This allowed modeling the changing conditions of the streets. The proposed solution allows to compute the real-time traffic density reconstruction in unmeasured road segments. The solution has been validated estimating the error in the places where the sensors are positioned, excluding each of them iteratively and reconstructing the flow without it. Then, it has been possible to estimate the error between the reconstructed traffic density and the measured value. This approach allowed setting up a converging algorithm for estimating the city graph from detailed parameters. The proposed reconstruction (and prediction) model has been created by exploiting open data and real-time data by using the Sii-Mobility infrastructure (national smart city project of Italian Ministry of Research for terrestrial mobility and transport, <http://www.sii-mobility.org>) solution based on Km4City model (<http://www.km4city.org>) in the Florence area, Italy, for its corresponding Smart City solution. This chapter is structured as follows. In section 8.2, the knowledge model representing the streets and their relationships is presented and discussed. Section 8.3 presents the city traffic flow model that has been used to describe the flow phenomena and the traffic propagation in the city. In Section 8.4 the city graph structure is introduced, and its graphical representation is described in Section 8.5. The computational model is discussed in Section 8.6 and the experimental results are presented in Section 8.7. Conclusions are discussed in Section 8.8.

8.2 Modeling Road Information For Traffic Flow

The reconstruction algorithm predicts the traffic flow in real-time observations for each 20 meters long segment of road and it displays the predicted values through coloured lines traced on a map, along the street paths. Then, detailed information about the street paths and their characteristics are needed. On the other hand, we cannot access directly to OSM or Web Open Data since the extraction of such data involves a heavy-weight process which

cannot be embedded in real-time applications. Moreover, a fundamental information is not included in OSM: sensors' data. Then, an ingestion process is necessary to import OSM data as well as static and dynamic sensors data. The following data are ingested and stored in the Km4city Knowledge Base (KB): (1) the geolocated points that outline the paths of the roads that fall within the boundary of interest, and for each point: (i) identifier; (ii) coordinates. The retrieval process follows: (1.1) the OSM ways tagged as highway are retrieved; (1.2) the proposed ways are discarded; (1.3) the ways that fall outside the boundary of interest are discarded; (1.4) the OSM nodes are retrieved that outline the paths of the retrieved ways; (1.5) the nodes that fall outside the boundary of interest are discarded; (1.6) instances of Km4City Node are generated in the Km4City KB whose URI (Uniform Resource Identifier) includes the OSM node ID; (1.7) lat and long are set for each Km4City Node accessing the OSM nodes geom; (2) the segments of road that are outlined by the points in (1), and for each segment we build/retrieve and store: (i) unique identifier; (ii) start and end node; (iii) traffic direction(s); (iv) length; (v) relevance of the road; (vi) segment type (such as if it locates within a parking, link, roundabout, crossroad, toll booth, square, pedestrian area, railway level crossing, rest area, and so on). The retrieval process follows: (2.1) the OSM relations with type set to route are retrieved; (2.2) those with route explicitly set to something different from road are discarded; (2.3) those tagged as network are discarded (international road networks duplicate their corresponding national representations); (2.4) those that only include ways tagged as proposed are discarded; (2.5) the OSM ways are retrieved that are part of the retrieved relations; (2.6) the ways that are not tagged as highway or that are tagged as proposed highway are discarded; (2.7) the ways that fall outside the boundary of interest are discarded; (2.8) the ways that do not make part of relations but anyway represent roads that fall within the boundary of interest are added; (2.9) the possible oneway tag is retrieved from all the ways of interest that denotes those ways that can be traversed in a direction only; (2.10) those ways tagged with highway set to construction are identified and they are considered not to be traversable in any direction; (2.11) those ways tagged with junction set to roundabout or with highway set to one of mini_roundabout, turning_circle, turning_loop models roundabouts are considered as one way; (2.12) the value of the highway tag is retrieved because it denotes the relevance of the road that the way represents; (2.13) the node-to-node linear segments are retrieved that are part of all the retrieved

ways; (2.14) a Km4City RoadElement is added to the Km4City KB for each segment, and is assigned a URI within which the Road and the position of the RoadElement can be seen; (2.15) the start and end OSM node are identified for each segment, their corresponding Km4City Nodes are also identified, and they are employed to fill the startAtNode and endsAtNode properties of the RoadElement; (2.16) the distance between the start and end node is computed and employed to fill the length property of the RoadElement; (2.17) the trafficDir is filled based on (2.9), (2.10); (2.18) the elementType is filled based on (2.11); (2.19) the highwayType (relevance) is filled based on (2.12); (3) the roads are retrieved where the segments in (2) are located, and for each of them the following are built/stored: (i) unique identifier; (ii) list of contained segments. A Km4City Road can be found in the Km4City KB for each relation retrieved at steps (2.1) to (2.4) and for each way retrieved at step (2.6). An URI is assigned to each Road that includes the OSM ID of the way or relation and ends by SR or LR respectively. The Road containsElement property is instantiated as many times as the segments retrieved in (2.13) are, and each instance is filled with the RoadElement instance that represents the segment; (4) the lanes that are drawn on the roads of interest are represented, and for each set of lanes in particular: (i) the road or segment of road where they are drawn; (ii) the traffic direction; (iii) the count with separated indication of the reserved lanes. The retrieval process follows: (4.1) for each relation and way retrieved at steps (2.1) to (2.6), the possible lanes tag is evaluated: (4.1.1) if the key also includes a typology of vehicle, the value is the count of the lanes reserved to that typology of vehicle, otherwise the tag value is the total count of the lanes; (4.1.2) if the key also includes forward or backward then the value is the count of the lanes in the specified direction; (4.2) for each lanes tag evaluated in (4.1), a Km4City Lanes instance is added to the Km4City KB with these properties: (4.2.1) where, filled with the Road or RoadElement where they are drawn; (4.2.2) direction, based on (4.1.2); (4.2.3) lanesCount, filled with a Km4City lanesCount instance whose undesignated property is set to the count of the undesignated lanes, and other properties named as typologies of vehicles are possibly set to the count of the lanes reserved to the respective typologies of vehicle; (5) the turn restrictions, and for each restriction: (i) the origin segment of road; (ii) the destination segment of road; (iii) the description of the restriction (whether it is a mandatory or forbidden maneuver). The retrieval process follows: (5.1) the OSM relations with type set to restriction

are retrieved; (5.2) the ways are retrieved that appear as from and to members within the relations in (5.1); (5.3) the OSM nodes where the ways in (5.2) cross are retrieved that appear as via members of the relations in (5.1); (5.4) the relations whose via member falls outside of the boundary of interest are discarded; (5.5) the restriction tag value (the description) is retrieved from the relations in (5.1); (5.6) from both the origin (from) and destination (to) ways, the node-to-node linear segment is extracted that is directly connected to the via node; (5.7) the possible limitations that could apply to the restriction related to days, hours, and typology of vehicles are retrieved from the `day_on`, `day_off`, `hour_on`, `hour_off` and `except` tags possibly applied to the relations in (5.1); (5.8) a `Km4City turnRestriction` is added to the `Km4City KB` for each relation in (5.1) that has not been filtered in (5.4); (5.9) the where and toward properties of the `turnRestriction` are filled with the `RoadElements` that correspond to the segments identified in (5.6); (5.10) the restriction property of `turnRestriction` is set to the value retrieved in (5.5); (5.11) for each tag identified in (5.7), an homonym property is set on the `RoadElement`; (6) the traffic sensors, and for each sensor: (i) position; (ii) list of detected traffic flows. The position and detections of the traffic sensors are ingested through ETL (Extract Transform Load) jobs that read from Web Open Data repositories, and for each sensor a `Km4City SensorSite` can be found in the `Km4City KB`, whose `vehicleFlow` property is set with the detected flow.

Moreover, a set of tools has also been made available based on the `Km4City KB`, the most notable of them for the purposes of the traffic reconstruction is the so-called `Latlon2Node`, that provides the nearest `Node` given a pair of coordinates, and that is leveraged by the reconstruction algorithm for mapping each geo-located traffic sensor to its nearest `Node`. This way, the traffic sensors come to be part of the street graph.

8.3 City Traffic Flow Model

The main concept of the present section is to apply a mathematical model for fluid dynamic (based on conservation laws in first approximation) on networks. Thus, road networks have been studied as directed graphs composed by arcs that meet at some nodes, corresponding to road junctions. Let us remark that fluid dynamic models for traffic flow seem to be the most appropriate to detect macroscopic phenomena as shocks formation and prop-

agation of waves backwards along roads. Here we deal with the macroscopic model proposed in [58] and it allows observing the network in the time evolution through waves formation. The fluid dynamics concepts are applied to traffic situation for the first time in [104], and independently in [123] with limited scope. In a single road, this non-linear model is based on the conservation of cars described by the following scalar hyperbolic conservation law

$$\frac{\partial \rho(t, x)}{\partial t} + \frac{\partial f(\rho(t, x))}{\partial x} = 0 \quad (8.1)$$

with: boundary conditions $\rho(t, a) = \rho_a(t)$, $\rho(t, b) = \rho_b(t)$, and initial values $\rho(0, x) = \rho_0(x)$. In particular $\rho(t, x)$ denotes the vehicular density and the function $f(\rho(t, x))$ is the vehicular flux which is defined as the product $\rho(t, x)v(t, x)$ where $v(t, x)$ is the local speed of the vehicles. In the case of first order approximation, if we assume that $v(t, x)$ is a decreasing function, only depending on the density, then the corresponding flux is a concave function. In this work, we consider the speed of the cars as $v(\rho) = v_{max}(1 - \frac{\rho}{\rho_{max}})$ and then $f(\rho) = v_{max}(1 - \frac{\rho}{\rho_{max}})\rho$, where v_{max} is the limit speed on a given road. These assumptions are known in the literature as the Greenshield's Model. Please note that, if the vehicular density is 0 then also the vehicular flux is 0. Otherwise, when the vehicular density grows, also the vehicular flux grows up to the maximum f_{max} for which the vehicular density assumes its critical value $\rho_c = \rho_{max}/2$. In the case where $\rho(t, x) > \rho_c$, other increases of the vehicular density conduct to a congested scenario of the traffic flow up to its maximum value ρ_{max} where the speed of the cars is 0 and then the vehicular flux is 0. In the presence of discontinuities, the density assumes a discontinuity that propagates backwards along the queue (for example, if a driver brakes).

The solution of the above equation can be obtained by means the finite differences in [76], bringing to a numerical solution of partial differential equations [93] [115]. Many recent studies in traffic flow for Intelligent Transportation System adopted this approach for simple model solutions [130] [58] [93]. In details, this scheme conducts both to an exact solution of the so-called Riemann problem and to approximations of piecewise constant functions. A Riemann problem, which naturally appears in computational fluid dynamics, consists of an initial value problem composed of a conservation equation together with piecewise constant data having a single discontinuity.

8.3.1 The numerical approximation scheme

The first step to develop a computerized solution for the proposed reconstruction problem is to discretize time and space by means of a time-space grid. Following the well-known Cell Transmission Model [66] [67], a road can be partitioned into a series of units having step size Δx , and similarly, the time is divided into a series of durations with step size Δt . The time-space region bounded within duration h and unit m is referred to as a cell and it is denoted as (h, m) and the number of vehicles contained in unit m at the end of duration h is denoted as $n(t_h, x_m) = n(h\Delta t, m\Delta x)$. We also use the notation u_m^h for $u(t_h, x_m)$ when u is a continuous function on the (t, x) plane. Moreover, the relationship $\frac{\Delta x}{\Delta t} > v_{max}$ is required in the discretization scheme, that is, a vehicle should not cross more than one unit Δx within a time step Δt .

As in [58], we assume the following rules in order to determine a unique solution to Riemann problem at junctions: (a) the traffic from the incoming roads is distributed on the outgoing roads according to some fixed coefficients; (b) respect to the previous point, the drivers choose to maximize the fluxes.

To distribute the traffic at each junction we define a matrix where each cell describes the percentage of drivers that arrive from a given incoming road and that take a given outgoing road. We consider the well-known Traffic Distribution Matrix defined as $A = a_{ij, j=n+1, \dots, n+m, i=1, \dots, n}$ such that $0 < a_{ji} < 1$ and $\sum_{j=n+1}^{n+m} a_{ji} = 1$, for $i = 1, \dots, n$ and $j = n+1, \dots, n+m$, where a_{ji} is the percentage of drivers arriving from the i -th incoming road that take the j -th outgoing road (assuming that, on each junction, the incoming flux coincides with the outgoing flux). Considering u as an exact solution of (1), we assume the following discretization by means of the finite difference scheme for the vehicular density evolution in each road:

$$u_m^{h+1} = u_m^h - \frac{\Delta t}{\Delta x} (F(u_m^h, u_{m+1}^h) - F(u_{m-1}^h, u_m^h)) \quad (8.2)$$

where, F denotes the numerical flux and it has the following expression according to the previous rules:

$$F(w, z) = \begin{cases} \min(f(w), f(z)), & w < z \\ f(w), & z < w < \rho_c \\ f_{max}, & z < \rho_c < w \\ f(Z), & \rho_c < z < w \end{cases} \quad (8.3)$$

In order to setup the finite difference scheme in the whole network, we have to impose both boundary conditions and conditions at the junctions.

The first ones occur in the placements at the bound of the observed area where the related roads have one of their endpoints not connected to any junction. For this purpose, considering the original problem in (1), we suppose to assign a condition at the incoming boundary for $x = 0$ as $\rho(t, 0) = \rho_b^{inc}(t)$ and we study the equations only for $x > 0$. In practice, we proceed by inserting an incoming ghost cell and the discretization becomes

$$u_0^{h+1} = u_0^h - \frac{\Delta t}{\Delta x} (F(u_0^h, u_1^h) - F(v_{(inc)}^h, u_0^h)) \quad (8.4)$$

where $v_{(inc)}^h = \frac{1}{\Delta t} \int_{t_h}^{t_{h+1}} \rho_b^{inc}(t) dt$ replaces the ghost value u_{-1}^h . Of course, let $x < N$, for the outgoing boundary we can consider a similar procedure leading to

$$u_N^{h+1} = u_N^h - \frac{\Delta t}{\Delta x} (F(u_N^h, v_{(out)}^h) - F(u_{N-1}^h, u_N^h)) \quad (8.5)$$

Finally, as for the conditions at the junctions, we assume the following discretization for road connected to a junction at the right and left endpoint, respectively $u_N^{h+1} = u_N^h - \frac{\Delta t}{\Delta x} (\gamma_i - F(v_{N-1}^h, u_N^h))$ and $u_0^{h+1} = u_0^h - \frac{\Delta t}{\Delta x} (F(u_0^h, u_1^h) - \gamma_j)$ where γ_i, γ_j are the maximized fluxes.

8.4 City Graph Structure

A city graph structure has to be opportunely modeled in order to apply the above-mentioned numerical solution. The proposed structure is composed by road-segments which can be seen as a convenient concatenation of RoadElements in a given way (see Section 8.2), so that at least three segments insist on each junction (for details about the definition of road-segments see the last part of the present Section). A road-segment inherits its features from its RoadElements and such characteristics are used to estimate the maximal density (and flux) value of a given road-segment in terms of vehicular capacity.

In order to reconstruct the traffic density in the city graph we consider a list (using the terminology of the software R used for the present work) which can be interpreted as a generic vector containing other objects. Each element of the list describes a way in the city and it contains its road-segments and their properties. Moreover, for each road-segment we introduce a data

table, called density table, containing the calculus of its vehicular density according to the numerical approximation scheme described in Section 8.3. In particular, the road-segment length is partitioned into unitary components, called unit, according to the assigned value Δx . Of course, the discretization of the road-segment length admits a rounded number. Then, the (h, m) -th cell of the density table contains the vehicular density in the m -th unit of a road-segment at the duration (iteration) h .

8.4.1 The city graph representation

A city network having N roads which are constituted by $M_1, \dots, M_N$ road-segments can be represented by means of a data table D having $(M_1 + \dots + M_N)$ rows. Each row represents a road-segment of the network and it contains the associated characteristics. Consequently, a junction in a network locating at a specific point on the Earth's surface (denoted as node or junction J) having n incoming roads and m outgoing road-segments is implicitly represented by means of the corresponding sub-table D' of D with $n + m$ rows. In particular, the incoming road-segments are represented by the n rows of D' having as their end node J , while the outgoing road-segments are the m rows of D' having as their initial node J . For each junction J , the associated Traffic Distribution Matrix is calculated according to the percentage of drivers arriving from an incoming road that take the outgoing roads and the traffic flow of an incoming road is divided among the outgoing roads.

Since the traffic network is represented by means of a directed graph, each road-segment opened in both directions has to be represented by means of two distinct road-segments having opposite directions. More precisely, each road-segment s opened in both directions, such that it is placed between the nodes A and A' , is represented in the graph by means of a road-segment s (denoting it by its URI) running from A to A' and a road-segment s/INV (denoting it by its URI followed by the suffix $/INV$) running from A' to A . For the sake of clarity, the density table of s/INV is obtained by inverting the density table of s . Moreover, if a junction J on the network meets two different road-segments opened in both directions, initially labelled with s and s' and placed between the nodes A and J , and the nodes J and B , respectively, then from the previous description a such junction is represented by means of Table 8.1. In this scenario, the percentage of drivers arriving from the road-segment s (or s') cannot turn in the road-segment s/INV (or s'/INV) in order to avoid a U turn and then, the corresponding value in the

Initial node	Road-segment URI	End node
node A	s	node J
node J	s/INV	node A
node J	s'	node B
node B	s'/INV	node J

Table 8.1: A node's representation in the case of roads opened in both directions

associated Traffic Distribution Matrix assumes value 0. The value 0 is also applied when a turn in an outgoing road is not permitted in a junction (see turnRestriction in Section 8.2).

8.4.2 Traffic distribution in the junction

In general, given a junction J, a crucial problem is the calculus of the associated Traffic Distribution Matrix and a formal method has to be derived. The idea is to define a numeric value related to the type of the road-segment in such a way that it assumes a weight in the calculus of the percentage of drivers that take the outgoing road-segments in J. Since a sorting is defined among the road-segment type then we have different weights in accordance with such property. Furthermore, the assignment of weights should be calibrated taking into account some specific properties related to a given junction. For example, the calculus of the percentage of turns depends by the attractions (monuments, events, etc.) in a specific area. Then, the weights have a parametric design and they can also evolve during the daily time (see Section 8.7 for a stochastic learning model improving the accuracy model). Given t in the daily time, the calculus of the coefficients $A_i(t) = a_{ji}(t)_{j=n+1, \dots, n+m}$ describing the percentage of the drivers of the i -th incoming road-segment that take the j -th outgoing road-segment, for each $j = n + 1, \dots, n + m$, is evaluated as follows. Denoting by $w_j(t)$ and l_j the weight at time t and the number of lanes of the j -th outgoing road-segment, respectively, we have:

$$a_{ji}(t) = \frac{R_j w_j(t) \frac{l_j}{G_j}}{\sum_{k=n+1}^{n+m} w_k(t)} \quad (8.6)$$

where R_j denotes the number of outgoing road-segments having the same type of the j -th outgoing road-segment and G_j denotes the sum of their

lanes. In the case of road-segments opened in both directions, we opportunistically modify (5) according to U turn avoidance.

In the city center, specific accesses for the Restricted Traffic Zone, RTZ, are present. In order to improve the network structure in the nearby of the RTZ access points we assign a different value to the percentage of the drivers that take the road-segment containing a given access point.

8.4.3 Data sensors' integration in the city graph

Cameras and spire road sensors are located both in proximity of the highway roads connecting the drivers to the city center and in proximity of the RTZ access points. Each sensor gives the state of the traffic counting the number of vehicles which pass through the supervised area and the data are simultaneously uploaded with a time slot of 10 minutes. By means of the Latlon2Node Km4city tool we associate a sensor to a node in the graph. Then, it admits both incoming and outgoing road-segments. On the principal roads opened in both directions we have a couple of sensors (one for each direction) which are usually associated with the same node. Then, we have to consider distinct nodes according to the directions of the vehicles by adding a new node in the network.

8.4.4 On the graph reduction and road-segment definition

The dimension of an urban network primarily depends on the number of the nodes and roads constituting the related directed graph. To simplify the structure of the graph a simple method consists in filtering the roads by their types. If we only consider a selection of roads' types, the structure of the graph reduces. Once the road types are chosen, it is possible to furtherly reduce the dimension of the graph by deleting some "redundant" nodes. The redundant nodes are the nodes in the same road having unique incoming and outgoing RoadElement. Then, we introduce an algorithm removing such redundant nodes by collecting together their incoming and outgoing RoadElements. The obtained object is called road-segment having its initial node as the initial node of the first RoadElement while its end node is the end node of the last one.

According to the definition of density table the road-segment length is partitioned into units which do not constitute other nodes in the graph. The

units are only used for the calculus of the density reconstruction and they have fixed value Δx . In the present work we have set $\Delta x = 20$ metres, so obtaining an accurate description of the density reconstruction. Units are fundamental information since they implicitly define the incoming and outgoing segments of a given road-segment, while redundant nodes do not give any other contribution to the distribution of the elements in the city graph. Nevertheless, the coordinates of redundant nodes are used to improve the graphical representation of the city graph on a map. The details are presented in the next Section.

8.5 Graphical Representation

In the present work we also consider a graphical representation to evaluate the state of the traffic in the considered area by an intuitive visualization.

8.5.1 Road-segments depiction

The graphical representation of a given road-segment in a map is constituted by the straight line connecting its initial and end node, respectively. Then, the values of the density table associated to that road-segment can be also represented. Since a road-segment is partitioned in units of length Δx , the idea is to allocate new coordinates for each of its unit and the related density value can be associated to the line placed between the new coordinates in the map.

Defining appropriate density ranges based on the road-segments characteristics and assigning a specific color to each range, it is possible to depict the road-segment in graph and visualize the related state of the traffic, simultaneously (see Fig. 8.1). The calculus of the range boundaries can be obtained from the observation of the data sensors in a large period of time taking into account some parameters. From this observation it is possible to collect together the roads having the same characteristics in terms of lanes and types in order to estimate different limits for a given kind of road. We consider four ordered limits for each kind of road to represent four different levels of traffic state, respectively, from a free to a congestion situation. The commonly used colors are: green, yellow, orange and red, considering the mentioned order.

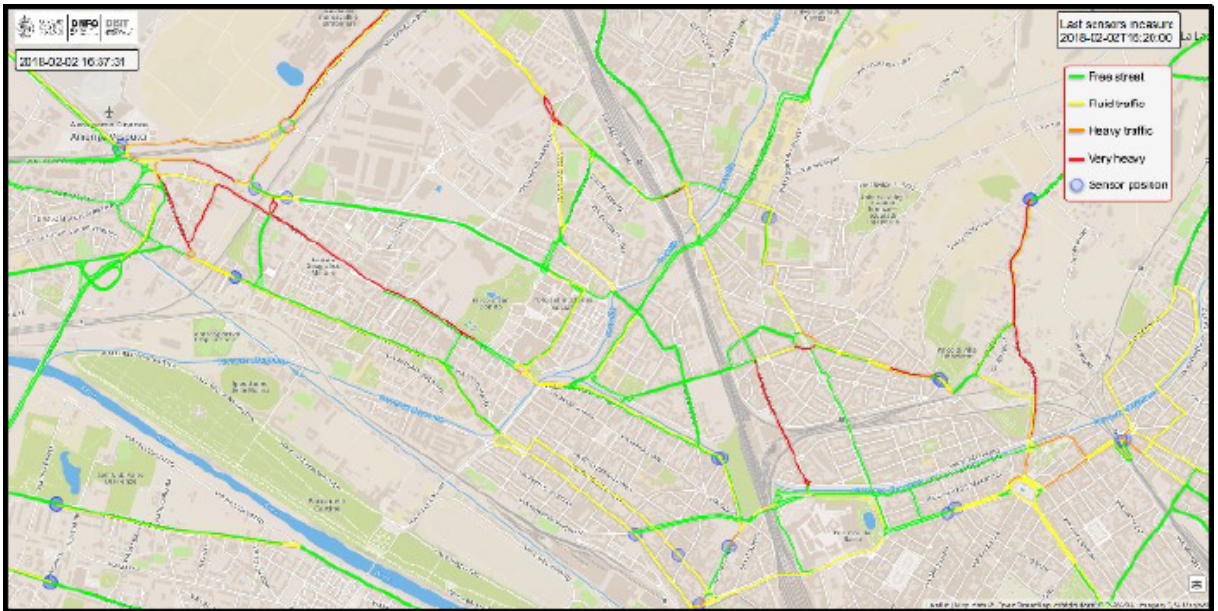


Figure 8.1: The graphical interface of the dynamics system considering the network area of Florence <http://firenzetraffic.km4city.org>

8.5.2 Graphics improvement

In general, the reduction in the number of nodes in the graph improves its complexity but not its graphical representation on a map, especially when a way has many curves. To solve this discordance, we adopted the following approach. Given a road-segment having n units such that it is obtained from the concatenation of k RoadElements. The idea is to allocate coordinates in the maps for each unit. In particular, the coordinates $(g_1, t_1)(g_2, t_2) \dots (g_{n+1}, t_{n+1})$ are calculated from the ones associated to the redundant nodes $(g'_1, t'_1)(g'_2, t'_2) \dots (g'_{k+1}, t'_{k+1})$, where $(g_1, t_1) = (g'_1, t'_1)$ and $(g_{n+1}, t_{n+1}) = (g'_{k+1}, t'_{k+1})$. In order to do that, we have to introduce a procedure to find the coordinate (g_h, t_h) which is the closest coordinate of a redundant node and such that they have distance Δx from (g_{h-1}, t_{h-1}) , with $h > 1$.

Another question concerning the improvement of the graphical representation is related to the depiction of road-segments opened in both directions consisting of two parallel lines symbolizing them. According to the slope of the line it is possible to deduce the right vehicle trip and we opportunely associate each line to the desired direction.

8.6 Computational Approach

Once the city graph is defined, the basics of the computational approach to the density reconstruction in real-time observation can be sketched by means of the following steps:

1. Load city graph
2. For each time slot t :
 - (a) Load sensors' values, and for each sensor, iteratively converge to the actual value
 - (b) Load weights' assignment $W(t)$
 - (c) For each node, iteratively compute the traffic distribution
 - (d) For each road, iteratively compute the density value in its segments
 - (e) Compute a graphics representation

In the following we explain in details the crucial computational step of the mentioned algorithm.

8.6.1 Sensors' values

For each time slot t , the measured data sensor is interpreted as the source of traffic leading into the outgoing roads of the considered junction. In particular, the value of the measured vehicular density $v_M(t)$ is the value of the vehicular density in the outgoing roads of the junction. If the number of outgoing roads is greater than 1, then we apply a similar procedure as in (5) for the data distribution. If the initial node of a road-segment corresponds to the location of a sensor, then the problem related to the integration of data sensor in the graph is solved by inserting the measured data $v_M(t)$ (opportunistically distributed) in the density table of the corresponding road-segment according to (3) where $\rho(t, 0) = v_M(t)$. In particular, a weak solution is given and the density value in the associated value passes from $v_M(t)$ to $v_M(t')$ by means of a smoothing modality, being t' the previous time slot.

8.6.2 Weights' assignment

As mentioned in Section 8.4, the estimation of traffic distribution on each node depends on the weights' assignment of the road-segments in the junction. A weights' assignment evolves during the daily time thus a given road-segment assumes a different weight according to the considered time slot. Here, we introduce a general approach which can be conducted in order to define the weights' assignment $W(t)$ for a given time slot t . The weights' assignment contains all the weights of the road-segments in the city graph. Fixed a period of observation involving the daily time and starting from an arbitrary weights' assignment, we consider a stochastic learning method taking into account the system error. The system error is estimated considering the calculated value with respect to the actual one during the computation of the model excluding the data from a selected sensor. Then, the system error is obtained by means of the mentioned procedure applied to each sensor in the graph. During the daily time, the stochastic method is introduced to learn the different weights of a road-segment minimizing the system error. Briefly, the procedure detects a fixed percentage of roads in the network and it randomly assigns values to their weights from a given range depending on the roads' types. If the actual system error is less than the previous one,

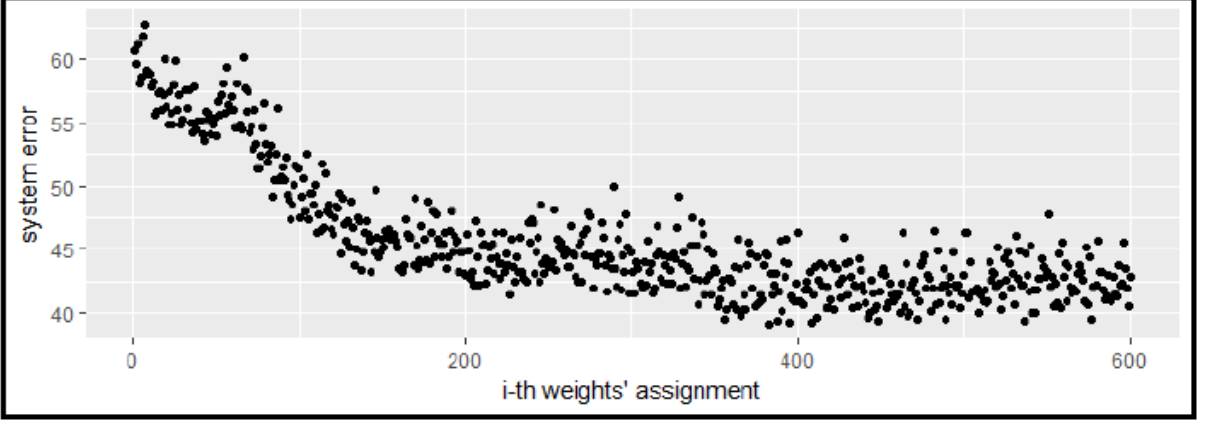


Figure 8.2: Graphical result of the stochastic approach minimizing the system error.

then the actual weights' assignment is considered as the starting point for the next calculus, otherwise the procedure is re-calculated leaving the current assignment unchanged. If the system error is unchanged after a fixed number of experiments, then the procedure is recalculated starting from the assignment giving the closest system error with respect to the current one and the previous assignment cannot be considered as a new starting point (according to the tabu search approach).

A such procedure is then computed in fixed periods T during the daily time. For example, fixed T from 6:00 to 7:00 pm, the results of the described procedure are depicted in Fig. 8.2. In particular, each point of the figure describes the system error value related to the i -th weights assignment $W_i(T)$.

We define the best weights assignment $W_b(T)$ as the weights assignment giving the lowest system error in the period T . Then, in real-time prediction we assign $W(t) = W_b(T)$ for each t in T .

8.6.3 Density computation

Once the data sensors are updated, the density value is propagated from the road-segments locating the sensors. Starting from a new measurement a fixed number of iterations H (durations) is considered to the propagation. According to definition of density table in Section 8.4, the calculus

of the vehicular density in the m -th unit of a road-segment at the duration h is deduced from (2). If a road-segment is located across the boundary of the city, then we distinguish two cases depending of the direction of the road-segment with respect to the graph. If the road-segment represents an incoming arc of the graph, then its vehicular density is computed according to (3) where a bounding value or “ghostvalue” is introduced (a such value can be interpreted as a “ghost-sensors” in the graph). Otherwise, if the road-segment represents an outgoing arc of the graph, then its vehicular density is computed according to (4).

8.6.4 Graphical representation

For each time slot t , after H iterations all the road-segments in the city graph have new predicted density values admitting their corresponding colors. Then, the graphical representation of the city graph in the map is re-depicted.

8.7 Experimental Results

The computational complexity of the present model primarily depends on the dimension of the considered city graph. In particular, it is an $O(H(N + S))$ where: N is the number of nodes, S is the number of units and H is the number of iterations. S is much larger, and it depends linearly on the total amount L of the streets length since $S = L/20mt$. Then, we have an $O(HS)$. As for the validation of the proposed model, we consider a rectangular area of the metropolitan network of Florence delimited by the coordinates (11.190046; 43.756291) and (11.288442; 43.816788), representing its bottom-left and top-right corners, respectively. The considered network is constituted by 736 nodes (or intersections), 1273 road-segments and 60 data sensors. Our validation approach is based on the computation of the model excluding the data from a selected sensor. Then, for each time slot t we estimate the error between the calculated value $v_c(t)$ in the road where the selected sensor locates, with respect to the measured traffic density value $v_d(t)$ from the sensor. The percentage error $e_r(t)$ at time t is calculated as $e_r(t) = 100 \frac{v_d(t) - v_c(t)}{w_d(t)}$. The accuracy of the whole model is estimated by considering the same procedure for each data sensor at time t . Then, the percentage system error is evaluated as $E(T) = \frac{1}{S} \sum_{i=1}^S e_r^i(t)$, where S is

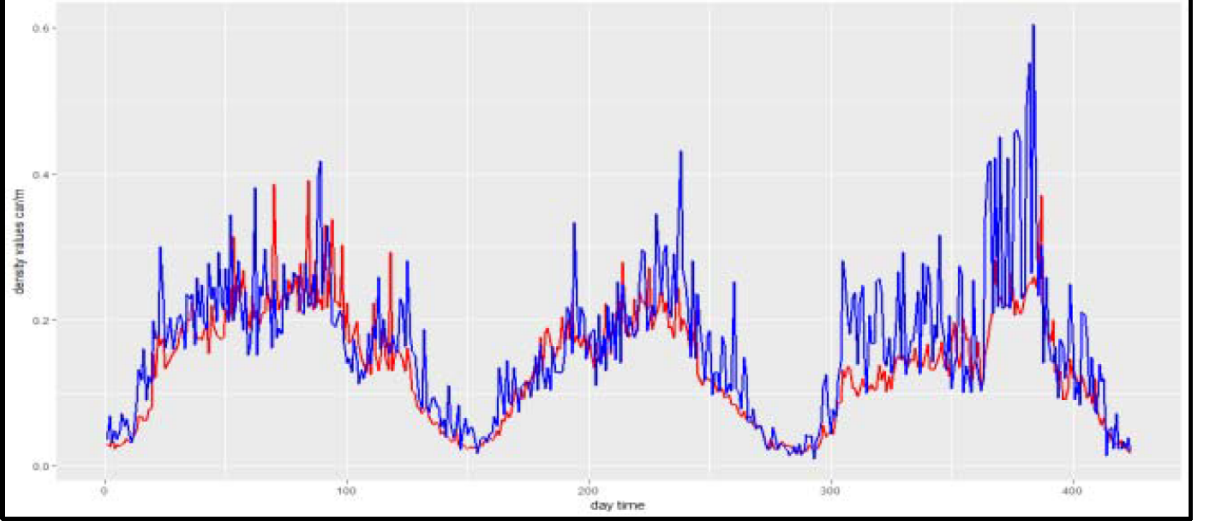


Figure 8.3: 72 hours real-time validation: the predicted and actual values are in red and blue, respectively, giving a mean percentage error equals to 27%

the number of data sensors and $e_r^i(t)$ is the percentage error of the i -th data sensor. An immediate analysis of the detected error suggests that its prediction's precision depends of the number of data sensors placed around the evaluated point. Of course, fixed a period of observation T , the system error in T is $E = \frac{1}{T} \sum_t E(t)$. For example, Fig. 8.3 shows the graphical behavior of the mentioned error related to a selected sensor taking into account 72 hours real-time validation (about 425 time-slots) leading to MPE=27%.

Each data sensor is updated every 10 minutes and $H = 300$ iterations are computed by the tool during each interval. A such number of iterations have to be calibrated in accordance to the time slot, that is, the running time of the tool does not exceed the time slot. Moreover, a such number is connected to the accuracy of the model and the mentioned value leads to a global accuracy close to 75% during a real-time performance.

8.8 Conclusion

We have proposed a real-time visual self-adaptive solution for reconstructing

the traffic density at every location within a wide urban area leveraging the detections from a few fixed traffic sensors deployed within the area of interest. Our solution presents several advantages with respect to the solutions available in the literature providing at the same time: (i) wide temporal and/or spatial horizon (not limited to a few road elements); (ii) supporting complex road structure; (iii) wide applicability; (iv) no needs of third-party engagement on providing data; (v) robust with respect to discontinuous data; (vi) declared clear precision rate; (vii) real-time visual rendering of results (<http://firenzetraffic.km4city.org>). We have addressed such issues through a traffic flow modelling based on the solid mathematics of the fluid dynamics, also relying on the constantly up-to-date road network, road features, traffic restrictions, and traffic sensor detections stored in the Km4City KB, and displaying the reconstruction at real-time through colored lines traced on a street map. We have tested the proposed solution for reconstructing the traffic in the city of Florence, and we have found it to reach a 75% precision also on boundaries and much higher precision inside the observed area.

Chapter 9

Traffic Flow Reconstruction from Scattered Data

Real time traffic flow data in terms of road-segment traffic densities are relevant information at the basis of several smart services, such as smart routing, smart planning for evacuations, planning of civil works on the city, etc. The traditional methods for traffic flow measured via sensors using data from navigator Apps (e.g., TomTom, Google map, Waze) could be very expensive to follow. For this reason, there is the space for low cost and fast solutions for dense traffic flow reconstruction from scattered data coming from a limited number of street sensors spread on the city. The proposed method is based on differential equations and physical constraints applied to a detailed street graph which is enriched of several features. A stochastic learning approach has been adopted to estimate the weights representing in certain sense the road-segments capacity at each time slot of the day. The proposed solution allows computing in real-time the traffic density reconstruction in unmeasured road-segments. The solution has been validated estimating the error in the places where the sensors are positioned, excluding each of them iteratively and reconstructing the flow without it. Then, it has been possible to estimate the error between the reconstructed traffic density and the measured values. This approach allowed setting up a converging algorithm for estimating the traffic density in the whole city graph from

detailed parameters. The proposed reconstruction model has been created by exploiting open and real-time data in the context of Sii-Mobility research project by using Km4City infrastructure in the area of Florence, Italy, for its corresponding Smart City solution.

¹ ²

9.1 Introduction

Short-term traffic flow prediction plays a crucial role in intelligent transportation in terms of route planning, traffic management and sustainable mobility. Thanks to the today's technologies, it is possible to collect real-time data to detect the state of the traffic in some specific areas. The traditional methods for traffic flow measured via sensors are very expensive and scattered in the city, not enough dense for the above-mentioned applications. Only using stationary sensors reduces the cost for a possible application since they are only positioned in a limited number of ways in the urban area. Nevertheless, in most cases a simple network area as freeways or rings, where incoming and outgoing roads have a limited number, is monitored by the studies that consider only sensors in fixed locations, see for instance [137] [130]. Our approach is different since it considers a complex network area as a city area by means of the exclusive usage of fixed sensors. Nevertheless, other studies both use stationary sensors and GPS data from mobile applications to analyze the state of traffic in an urban network [5] [110]. Of course, the increase of sensors can help on the purpose but the cost for a possible application would necessarily increase.

For this reason there is the space for: (i) low cost and fast solutions for dense traffic flow reconstruction from scattered data coming from a limited number of street sensors spread on the city; (ii) autonomously deployable solutions that do not require the engagement of third parties such as taxi drivers, public transport companies or travelers; (iii) open solutions that can be implemented without any restriction deriving from patents or similar; (iv)

¹This chapter has been published as "Traffic Flow Reconstruction from Scattered Data" in *IEEE SMARTCOMP, IEEE international conference on smart computing, 18-20 June, Taormina, Sicily, Italy. 2018* [45].

²*Acknowledgments:* The authors would like to thank the MIUR, the University of Florence and companies involved for co-founding Sii-Mobility national project on smart city mobility and transport. Km4City is an open technology and research of DISIT Lab. Sii-Mobility is grounded and has contributed to Km4City open solution.

solutions that do not rely on the simplicity of the road network; (v) solutions that display their results in a user-friendly manner. To this end, our goal is to develop an algorithm for traffic flow reconstruction to monitor a complex network as an urban area taking into account fixed sensors only and performing from them the traffic flow reconstruction in the unmonitored areas to evaluate the traffic density in the urban network. Many cities have open traffic sensors like cameras and spire road sensors and the present chapter can be a useful tool for their sustainable mobility. The proposed method is based on differential equations and physical constraints applied on the basis of a detailed street graph which is enriched by several features. The physical structure of the present model is based on the open geodata from the Open Street Map (OSM) database which allows us to obtain a restricted description of the network graph modelling the selected area of observation. To achieve our goal such data have to be opportunely adapted to the network computation structure. A stochastic learning approach has been adopted to estimate the weights representing the street segment capacity at each time slot of the day. This allowed modeling the changing conditions of the streets. The proposed solution allows computing in real-time the traffic density reconstruction in unmeasured road-segments. The solution has been validated estimating the error in the places where the sensors are positioned, excluding each of them iteratively and reconstructing the flow without it. Then, it has been possible to estimate the error between the reconstructed traffic density and the measured value. This approach allowed setting up a converging algorithm for estimating the city graph form detailed parameters. The proposed reconstruction (and prediction) model has been created by exploiting open data and real-time data by using the Sii-Mobility infrastructure (national smart city project of Italian Ministry of Research for terrestrial mobility and transport, <http://www.sii-mobility.org>) solution based on Km4City model (<http://www.km4city.org>) in the Florence area, Italy, for its corresponding Smart City solution. This chapter is structured as follows. Section 9.2 presents the city traffic flow model that has been used for describing the flow phenomena and the traffic propagation in the city. The computational model is discussed in Section 9.3 and the experimental results are presented in Section 9.4. Conclusions are discussed in Section 9.5.

9.2 City Traffic Flow Model

The main concept of the present work is to apply a mathematical model for fluid dynamic on networks. To this aim, road networks have been studied as directed graphs composed by arcs that meet at some nodes, corresponding to road junctions. The desired reconstruction algorithm predicts the traffic flow in real-time observation in street segments of length 20 meters and it displays the predicted values through coloured lines traced on a map. Then, a detailed information about the street paths and their characteristics are needed. On the other hand, we cannot access directly to OSM or Web Open Data since the extraction of such data involve a heavy weight process which cannot be embedded in real-time applications. Moreover, a fundamental information is not included in OSM: sensors' data. Then, an ingestion process is needed to import such static and dynamic data in the Km4city Knowledge Base (KB).

Let us remark that fluid dynamic models for traffic flow seem to be the most appropriate to detect macroscopic phenomena as traffic jams and propagation of waves backwards along roads. Here we deal with the macroscopic model proposed in [58], that allows observing the network in the time evolution through waves formation. The fluid dynamics concepts are applied to traffic situation for the first time in [104], and independently in [123] with limited scope. In a single road, this non-linear model is based on the conservation of cars described by the following scalar hyperbolic conservation law

$$\frac{\partial \rho(t, x)}{\partial t} + \frac{\partial f(\rho(t, x))}{\partial x} = 0 \quad (9.1)$$

with: boundary conditions $\rho(t, a) = \rho_a(t)$, $\rho(t, b) = \rho_b(t)$, and initial values $\rho(0, x) = \rho_0(x)$. In particular $\rho(t, x)$ denotes the vehicular density and the function $f(\rho(t, x))$ is the vehicular flux which is defined as the product $\rho(t, x)v(t, x)$ where $v(t, x)$ is the local speed of the vehicles.

The solution of the above equation can be obtained by finite differences in [76], bringing to a solution of partial differential equations [93], [115]. The first step to develop a computerized solution for the proposed reconstruction problem is to discretize time and space by means of a time-space grid. Following the well-known Cell Transmission Model [67], a road can be partitioned into a series of units having step size Δx , and similarly, the time is divided into a series of durations with step size Δt . The time-space region bounded within duration h and unit m is referred to as a cell and it is denoted as (h, m) and the number of vehicles contained in segment m

at the end of duration h is denoted as $n(t_h, x_m) = n(h\Delta t, m\Delta x)$. By using the notation u_m^h , for $u(t_h, x_m)$ when u is a continuous function on the (t, x) plane and considering u as an exact solution of (1), we assume the following discretization by means of the finite difference scheme for the evolution of the vehicular density in each road:

$$u_m^{h+1} = u_m^h - \frac{\Delta t}{\Delta x} (F(u_m^h, u_{m+1}^h) - F(u_{m-1}^h, u_m^h)) \quad (9.2)$$

where F denotes the numerical flux. To setup the above numerical approximation in the whole graph we have to opportunely impose both boundary conditions and conditions at the junctions in (2).

A city graph structure is modeled in order to apply the above mentioned numerical solution. The proposed structure is composed by road-segments and for each junction at least three road-segments insist on it. A city network having N roads which are constituted by $M_1, \dots, M_N$ road-segments, respectively, can be represented by means of a data table D having $(M_1 + \dots + M_N)$ rows. Then, a junction in a network locating at a specific point on the Earth's surface having n incoming roads and m outgoing road-segments is implicitly represented by means of the corresponding sub-table of D with $n + m$ rows. For each junction, the associated Traffic Distribution Matrix (TDM) is calculated according to the percentage of drivers arriving from an incoming road that take the outgoing roads and the traffic flow of an incoming road is divided among the outgoing roads (assuming that, on each junction, the incoming flux coincides with the outgoing flux).

Cameras and spire road sensors give the state of the traffic counting the number of vehicles which pass through the supervised area and the data are simultaneously uploaded with a time slot of 10 minutes. For each time slot t , the measured data sensor is interpreted as the source of traffic leading into the outgoing roads of the nodes where sensors are located.

9.3 Computational Approach

Once the city graph is defined, the basics of the computational approach to the density reconstruction in real-time observation can be sketched by means of the following steps:

1. Load city graph
2. For each time slot t :

- (a) Load sensors' values, and for each sensor, iteratively converge to the actual value
- (b) Load weights assignment $W(t)$
- (c) For each node, iteratively compute the traffic distribution
- (d) For each road, iteratively compute the density value in its segments
- (e) Compute a graphics representation

A crucial problem is the calculus of the TDM on each junction. The idea is to define a numeric value related to the characteristics of a road-segment in such a way that it assumes a weight in the calculus of the percentage of drivers that traverse the outgoing road-segments. Then, the estimation of traffic distribution on each node depends on the weights' assignment of the road-segments in the junction. A weights' assignment evolves during the daily time, thus a given road-segment assumes a different weight according to the considered time slot. Here, we introduce a general approach which can be conducted in order to define the weights assignment $W(t)$ for a given time slot t . The weights assignment contains all the weights of the road-segments in the city graph. Fixed a period of observation involving the daily time and starting from an arbitrary weights assignment, we consider a stochastic learning method taking into account the system error. The system error is estimated considering the calculated value with respect to the actual one during the computation of the model excluding the data from a selected sensor. Then, the system error is obtained by means of the mentioned procedure applied to each sensor in the graph. During the daily time, the stochastic method is introduced to estimate the different weights of a road-segment minimizing the system error.

For each time slot t , after H iterations all the road-segments in the city graph have new predicted density values admitting their corresponding colors, considering four ordered limits to represent four different levels of traffic state, respectively, from a free to a congested situation. The graphical representation of the city graph in the map is so re-depicted (see Fig. 9.1).

9.4 Experimental Results

The computational complexity of the present model primarily depends on the dimension of the considered city graph. In particular, it is an $O(H(N +$

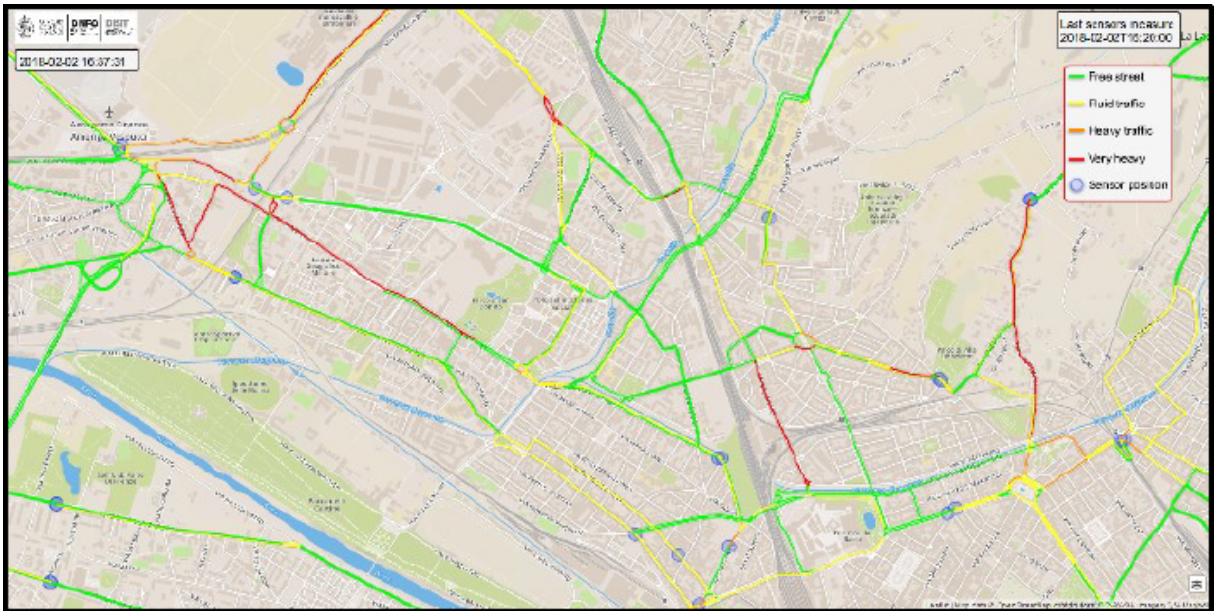


Figure 9.1: The graphical interface of the dynamics system considering the network area of Florence <http://firenzetraffic.km4city.org>

S)) where: N is the number of nodes, S is the number of units and H is the number of iterations. S is much larger and it depends linearly on the total amount L of the streets length since $S = L/20m$. Then, we have an $O(HS)$. As for the validation of the proposed model, we consider a rectangular area of the metropolitan network of Florence which is constituted by 736 nodes (or intersections), 1273 road-segments and 60 data sensors. Our validation approach is based on the computation of the model excluding the data from a selected sensor. Then, for each time slot t we are going to estimate the error between the calculated value $v_c(t)$ in the road where the selected sensor locates, with respect to the measured traffic density value $v_d(t)$ from the sensor. The percentage error $e_r(t)$ at time t is calculated as $e_r(t) = 100 \frac{|v_d(t) - v_c(t)|}{v_d(t)}$. The accuracy of the whole model is estimated by considering the same procedure for each data sensor at time t . Then, the percentage system error is evaluated as $E(t) = \frac{1}{S} \sum_{i=1}^S e_r^i(t)$, where S is the number of data sensors and $e_r^i(t)$ is the percentage error of the i -th data sensor. The mentioned approach leads to a global accuracy close to 75% during a real-time performance.

9.5 Conclusions

We have proposed a real-time visual self-adaptive solution for reconstructing the traffic density at every location within a wide urban area leveraging the detections from a few fixed traffic sensors deployed within the area of interest. Our solution presents several advantages with respect to the solutions available in the literature providing at the same time: (i) wide temporal and/or spatial horizon (not limited to a few road elements); (ii) supporting complex road structure; (iii) wide applicability; (iv) no needs of third-party engagement on providing data; (v) robust with respect to discontinuous data; (vi) declared clear precision rate; (vii) real time visual rendering of results (<http://firenzetraffic.km4city.org>). We have addressed such issues through a traffic flow modelling based on the solid mathematics of the fluid dynamics, also relying on the constantly up-to-date road network, road features, traffic restrictions, and traffic sensor detections stored in the Km4City KB, and displaying the reconstruction at real-time through colored lines traced on a street map. We have tested the proposed solution for reconstructing the traffic in the city of Florence, and we have found it to reach a 75% precision also on boundaries and much higher precision inside

the observed area.

Chapter 10

How COVID-19 Lockdown Impacted on Mobility and Environmental data

*With regard to the COVID-19 lockdown and successive reopening a number of facts can be analysed. The main effects have been detected on: mobility and environment, and specifically on traffic, environmental data and parking. The mobility reduction has been assessed to be quite coherent with respect to what has been described by Google Global mobility report. On the other hand, in this chapter a number of additional aspects have been put in evidence providing detailed aspects on mobility and parking that allowed us to better analyze the impact of the reopening on an eventual revamping of the infection, also taking into account of the R_t index. To this end, the collected data from the field have been compared with those of Google and some considerations with respect to the Imperial college Report 20 have been derived. For the pollutant aspects, a relevant reduction on most of them has been measured and rationales are reported. The solution has exploited the Snap4City IOT smart city infrastructure and data collector and Dashboard in place in Tuscany.*¹

¹This chapter has been published as “How COVID-19 Lockdown Impacted on Mobility and Environmental data” in *Bollettino della Società Geografica Italiana, FuPress, June 2020* [32].

10.1 Introduction

Italy has been one of the first countries in Europe to be invaded by the so-called COVID-19/coronavirus infection. Thus, the local and national governments implemented a number of Non-Pharmaceutical Intervention (NPIs), the lockdown, initially on specific regions and then in the whole country, aiming at reducing social contact and propagation and thus at reducing mobility and their motivations, creating a social distancing, banning social events, closing public events, services and restaurants, etc. In Tuscany, DISIT Lab with Snap4City infrastructure and service collects every day a large amount of data from several sources. The lockdown has been progressively imposed from the 5th to the 10th of March, and it has been discontinued the 4th of May, starting with a progressive reopen of production activities, still leaving closed the social events, entertainments, restaurants, etc. Also the research activity restarted, but we at DISIT Lab have never closed, we have remained for the whole duration of the lockdown in smart working modality as we are now, continuously supporting the Snap4City infrastructure and services, and developing according to the large number of research projects we had in place in the period.

In this chapter, according to the above facts, we present an impact assessment of the effects of lockdown on data collected in the period with respect to previous weeks, months and year for the same variables and data observed and collected. The main effects have been detected on: mobility, environment, social media and people flows, while this report is mainly focused on mobility, transport and environmental aspects. For these aspects different data collected and deductions can be provided. Therefore, for each of these domains and/or for each kind of data a separate discussion is presented in the following sections.

The chapter is organized as follows. In section 10.2, an overview of Snap4City is reported. Section 10.3 describes the impact of lockdown on mobility and possible deductions, also taking into account the data of Civil protection for the area of Tuscany and the computation of the R_t . Section 10.4 describes the impact of lockdown on parking facilities and deductions. In Section 10.5, the impact of lockdown on environmental data and deductions is presented. In Section 10.6, conclusions are drawn.

10.2 Snap4City overview

Snap4city (<https://www.snap4city.org>) has been developed to provide many online tools and guidelines to involve all different kinds of organizations (e.g., Research Centers and Universities, small business, large industries, public administrations, and local governments) and citizens (e.g., city operators, resource operators, companies, tech providers, category Associations, corporations, research groups, advertisers, city users, community builders) [31] [49] [116] [29] [34] [39] [37]. Full training on Snap4City is accessible on <https://www.snap4city.org/577>

Snap4City improves city services, security and safety by offering a sustainable solution for smart city and Living Lab, thus attracting industries and stakeholders. Snap4City is capable to keep under control the real-time city evolution reading sensors, computing and controlling key performance indicators (KPI), detecting unexpected evolutions, performing analytics, taking actions on strategies and alarms. Snap4City supports the city in the process of continuous innovation on services and infrastructures, with control and supervision, tools for business intelligence, predictions, anomaly detection, early warning, risk assessment, what-if analysis, also setting up strategies for increasing city resilience with respect to unexpected unknown events.

Thanks to knowledge base support, Snap4City provides flexible solutions to get immediate insights and deductions of the city status and evolution, exploiting ultimate artificial intelligence, data analytics and big data technologies, activating sentient solutions collecting, and exploiting heterogeneous data of any kind, from any data source (open and private; static, real time, event driven, streams, certified and personal). Snap4City solution provides a flexible method and solution to quickly create a large range of smart city applications exploiting heterogeneous data and enabling services for stakeholders by IOT/IOE, data analytics and big data technologies.

Snap4City applications may exploit multiple paradigms as data driven, stream and batch processing, putting co-creation tools in the hands of: (i) Smart Living Lab users and developers a plethora of solutions to develop applications without vendor lock-in nor technology lock-in, (ii) final users customizable / flexible mobile Apps and tools, (iii) city operators and decision makers specialized / sophisticated city dashboards and IOT/IOE applications for city status monitoring, control and decision support. Snap4City satisfies all the expected requirements of ENOL, EIP-CPP, Select4Cities

challenge PCP and much more, and it is 100% open source, scalable, robust, respects user needs and privacy; provides MicroServices and easily replaceable tools; compliant with GDPR; provides a set of tools for knowledge and living lab management, and it is compliant with more than 70 protocols including end-to-end encrypted communication.

Snap4City is an official platform of FiWare, an official library of JS Foundation Node-RED, registered on E015, present on EOSC marketplace, and BeeSmartCity MarketPlace, etc. Snap4City obtained the 1st place award by Select4Cities partners and PCP (Antwerp, Copenhagen and Helsinki). Snap4City is also the platform of Herit-Data project (Sustainable Heritage Management towards Mass Tourism Impact thanks to a holistic use of Big and Open Data from Interreg MED program and co-financed by FESR).

Snap4City provides services and data of several cities/Organizations as: Firenze, Helsinki, Antwerp, Lonato del Garda, Santiago de Compostela, Pisa, Prato, Pistoia, Lucca, Arezzo, Grosseto, Livorno, Siena, Massa, Modena, Cagliari, Valencia, Pont du Gard, Dubrovnik, Western Greece, Mostar; and from regions as Tuscany (last 5 cities/organizations, including also Florence, are partner of Herit-data project and services have been provided within such project), Garda Lake, Sardegna, Belgium, Finland, Emilia Romagna, Spain, etc. Snap4City is open to your contributions, using Snap4City tools and contributing in improving them, adding more tools and features, etc. Please join the community on this portal and on GitHub/disit at <https://github.com/disit>

10.3 Impact of lockdown on traffic data

In the context of mobility of people the data sources that could provide the evidence of impact of the above described situation are traffic flow sensors (which express the number of vehicles traversing the road, and thus the density of vehicles per time slot, for example: number of vehicles passing every 15 minutes). Those sensors are typically placed in the:

1. main roads of the city monitoring inflow and outflow of the city.
2. main entrance of the restricted traffic zone, RTZ.
3. internal main roads of the city (not directly addressed in this report).
4. high speed roads of the region (not directly addressed in this report).

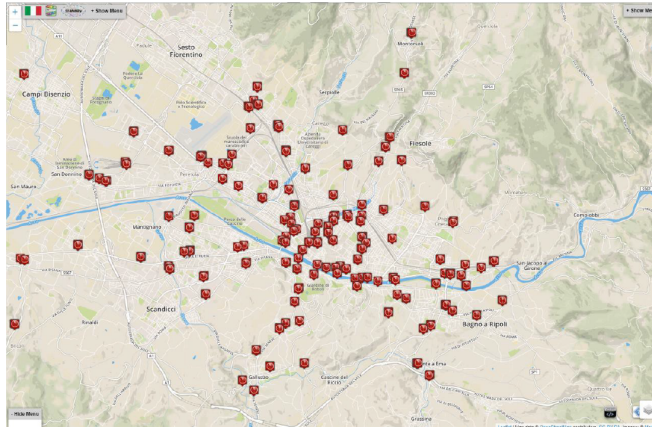


Figure 10.1: Placement of traffic flow sensors in the city, <https://servicemap.snap4city.org/>

For cases (1) and (2), the in/out flows are computed on the basis of the traffic flow sensors placed on the main roads connecting the city with the rest of countryside (see Figure 10.1). So that, those traffic sensors are describing the effective number of vehicles that enter/exit in/from the city every 10/15 minutes and thus allow to compute the total number of moving vehicles in the day. In Florence, in normal conditions we have an average of 290000 vehicles entering and exiting every day, in both directions and the total flow is almost balanced at the end of the day and even hour by hour. For vehicles we intend equivalent vehicles, which means that cars are counted 1, bus are counted 2.5, motorbikes 0.5, etc. Please note that, taking into account traffic flow sensors, it is possible to reconstruct the traffic flow in the other road segments of the city in which the flow sensors are not present as described in [44], [45]. See also <https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MTc5NQ==> for public Dashboard with traffic flow reconstruction in real-time. The traffic flow reconstruction allows to actually understand which is the effective usage of the city roads without the need to install a large number of sensors to create a dense network of expensive sensors.

As regarding the traffic flow, the following Figure 10.2 shows the impact of the lockdown on the inflow and outflow of vehicles in Florence (which is the point 1 of the above list). The first part of Figure 10.2 Dashboard reports the

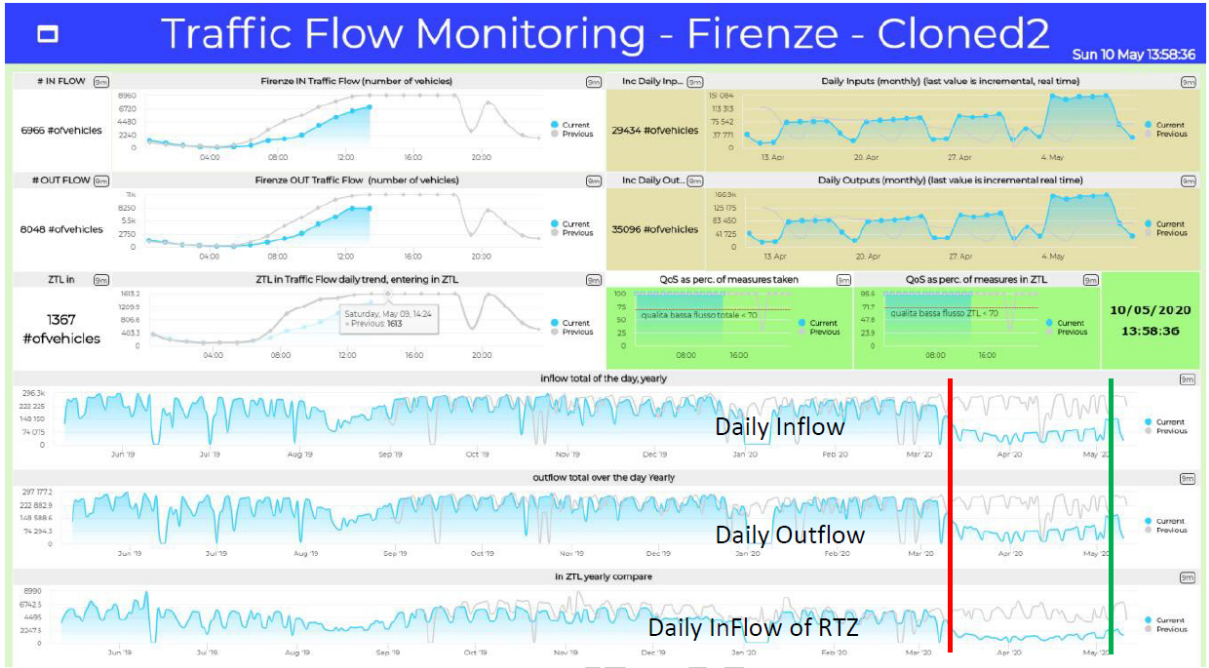


Figure 10.2: Traffic flow dashboard from Snap4City for Florence area, accessible from: <https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MjY1MQ==>

trend over H24 for in/out flows of the city and RTZ inflow, compared with the trend of the previous day. In the second half of the dashboard, the trend of daily counting of vehicles entering (inflow) and exiting (outflow) from Florence city and its RTZ inflow, in the last 12 months are reported, and compared with the previous year data (in grey). Please note that the graphs present a weekly periodic trend (see also canyons due to the weekends), and some holes due to mistakes in the sensors network and communications.

In more detail, the trends are presented of whole daily counting of vehicles entering (inflow) and exiting (outflow) from Florence in the period of January and May 2020 (in blue), with respect the corresponding values measured on the 2019 on the same dates (in grey). Please note that, the lockdown has been started effectively the 9th of March (a Monday), while in the first days of April the total flow traffic was reduced at the 18% of the former (53000

vec. per day with respect to the 284000). On the other hand, the 4th and 5th of May (the first and second days of the reopening) the reduction was still of the 52% (148.000 with respect to 284000). It can be similarly stated for the RTZ, where the reduction in the first days of April has been at the 20%, and at the reopen in May of 46%. This reduction does not take into account the movements of people for working, or biking, and neither the flows internal to the city, a part for the flow into the RTZ. On the other hand, a similar reduction has been also recorded taking into account all the other traffic flow sensors inside the city as in Figure 10.1, and along the high-speed road as FiPiLi in Tuscany.

Similarly, from Google Global Mobility report [10], [26], it can be observed a reduction of mobility (in this case measured from the Google mobile Apps) of the people as reported in Figure 10.3. In this case, as reported by Google the reduction has been at the 30% regarding the baseline (so that -70%), at 75% for grocery & pharmacy, at 30% for parking, at 33% for transit in stations, at 45% for working, while an increment of local residential movements have been recorded with at a 128% with respect to the previous conditions. Comparing the data of Google with respect to those collected from the actual sensors, it appears evident that the data of Google cannot be combined each other since the different trends are only relative reductions or increment with respect to the corresponding baseline that is unknown. So that, the actual reduction of mobility for working in terms of number of people is unknown. On the other hand, the reduction to the 30-25% of mobility is confirmed for baseline, parking and transit stations; which is coherent with more precise values estimated by sensors still in Florence area.

The remaining 20% of traffic flow during lock down has been mainly due to the mobility of mandatory services for example for the activities to guarantee the functionalities of hospital, supermarkets, pharmacies, public transportation for those workers, etc. In [21] (the Report 20 of Imperial College), the effect of the recovery of traffic and mobility of +20% and +40% from the 4th of May with respect to the lockdown conditions have been analysed. In [21], the model proposed puts in connection the increment of traffic with the R factor that should remain below 1 to control the epidemy. In Figure 10.4, the Simulation presented in [21] regarding Tuscany for different cases in which a different rate of increment of mobility is supposed.

The effective impact of the registered increment of mobility in Florence is very hard to be assessed since the model is supposing that the return-

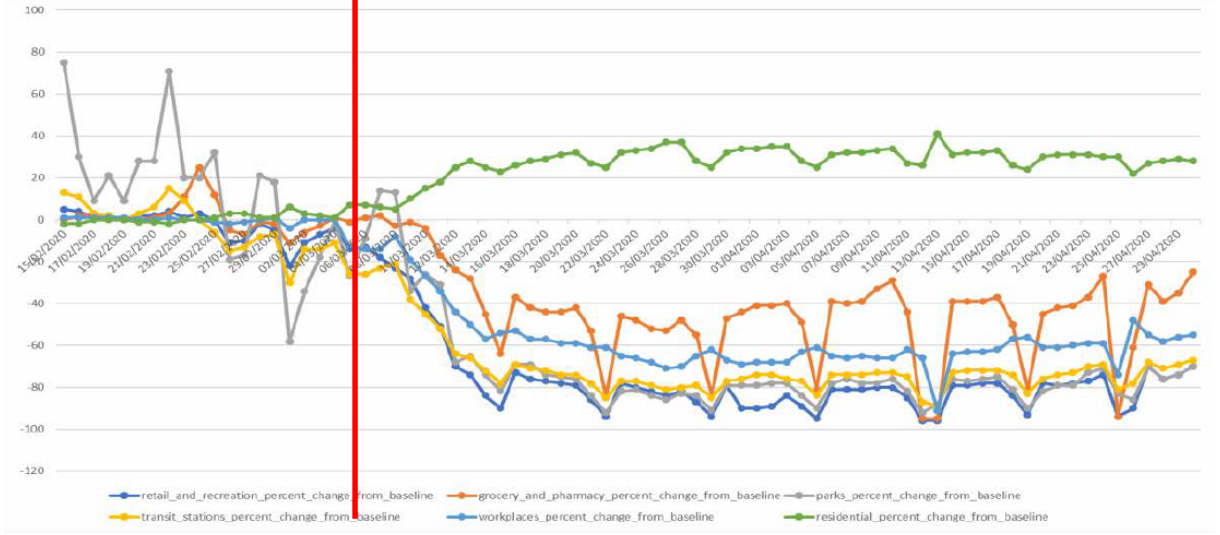


Figure 10.3: Reduction of mobility from Google Global mobility data [10], [26]. Red line marks the start of the lockdown.

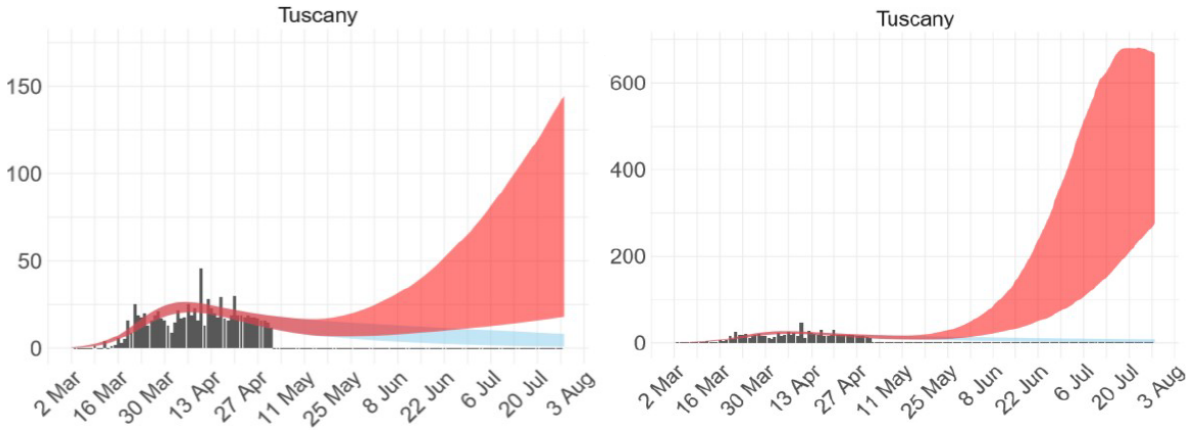


Figure 10.4: Simulation presented in [21] regarding Tuscany: (i) in red the trend of the deaths as confidence intervals over time in the cases of +20 (on the left), +40 (on the right) of mobility, (ii) in cyan the trend of the confidence values over time of the possible deaths in the cases of lock down continuation, and (iii) in black the officially registered number of deaths.

ing of the mobility would be performed with the same human behaviour of before COVID-19. On the other hand, most of the population has understood to use the NPIs. Moreover, the model presented in [21] does not clarify which amount of mobility they have supposed to have during lockdown. If they have supposed the mobility very close to zero, the model in that sense has been optimistic. This means that the actual forecast of the number deaths with respect to the increment of traffic and mobility is very hard to be produced even with large bounds of confidence. On the other hand, at the 10 of May, the traffic in first week of restart after the lockdown has been recovered to the 52%, which is relevant increment of mobility. The official R_t is produced by the Tuscany Region for the whole region has been estimated to be in decreasing trend and estimated at the 9th of May equal to 0.54 (https://www.ars.toscana.it/banche-dati/dati-sintesi-sintcovid-aggiornamenti-e-novita-sul-numero-dei-casi-deceduti-tamprovenienza=home_ricerca&dettaglio=ric_geo_covid&par_top_geografia=090), while the computed value on our side has been confirmed at 0.54 for Florence (R_t has been always above 1 up to the 7th of April, and close to 1 for several days after, thus decreasing below 1 only in May. It was close to 3.5 the 9 March when the lockdown has been started). These estimations are typically affected by relevant bounds of confidence. In Figure 10.5, it is shown the Dashboard reporting the most relevant trends for the number of Cases in the provinces of Tuscany and the trends of R_t for all the provinces of Tuscany are reported. From those data it is evident that some of the provinces still have problems on R_t very close to 1. Please note that the computation of R_t is typically performed 4-5 days in delay with respect to the actual value. The computed R_t values are an approximation with respect to <https://www.datacamp.com/community/tutorials/replicating-in-r-covid19>

The dashboard of Figure 10.5 is in Italian language and includes: the different trend at national level, on the left bottom corner the % of variation of daily cases which is consolidated index of the infection, the % of variation for the deaths for the major regions, etc. At the time of wiring of this chapter, most of the provinces in Tuscany has apparently reached a point close to maxim. It is even much more evident from the LOG view of the same graphs: <https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MjU4MA==>

This Dashboard reports the trends of cases for the 10 provinces of Tuscany Region, and the aligned trends at 100 cases of the curves for the most

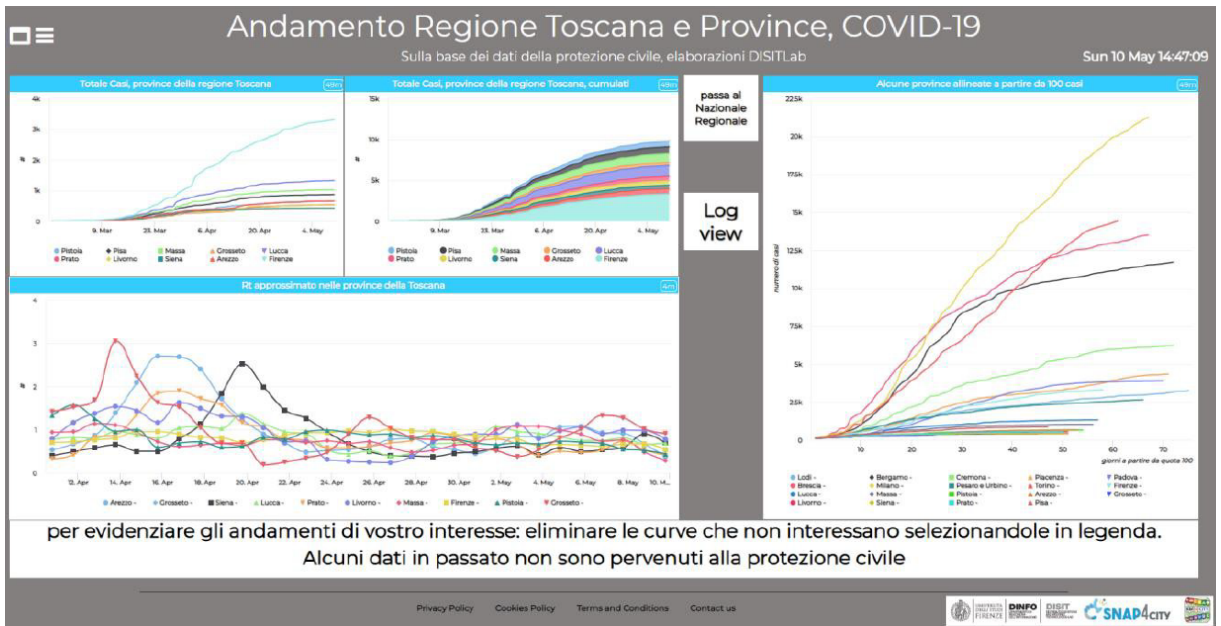


Figure 10.5: Data visual analytics on the basis of the COVID-19 data coming from Italian Civil Protection

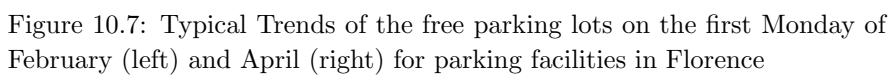
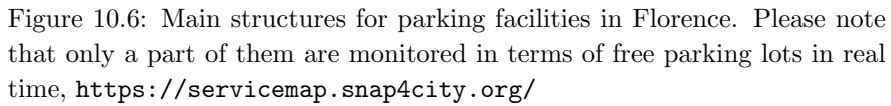
infected provinces in Italy with respect to those of Tuscany (on the right side of the Dashboard). From this last picture it is evident that in Tuscany most of the provinces have a rate for cases which much lower with respect to the most infected provinces in Italy that are: Milano, Torino, Brescia and Bergamo.

10.4 Impact of lockdown on parking facilities and deductions

In the context of mobility and transport relevant data sources that could be affected by the above described situation are Parking status sensors placed close to the most relevant infrastructures (which allows to count the number of free parking slots over time) such as: hospital, railways stations, justice palace, markets and shopping centers, social and entertainment areas. In Florence, and Tuscany there is a number of infrastructure parking lots in which the number free spaces are counted directly at the entrance/exit. On the other hand, we are focusing on this chapter only on the major parking lots in Florence as depicted in Figure 10.6, a large number of them are controlled and provide real time data. For most of them the Snap4City mobile Apps also provide predictions and real time monitoring data.

The trends of free parking lots have been already studied in the past with the aim of providing short and long terms predictions [40]. The typical trends for the days during working days and weekend were strongly different, resulting more critical in the working days in which a number of parking infrastructures run out of free parking lots. For example, in Figure 10.7, the comparison of the typical trends before the lockdown (left) and in the middle of the lockdown (right). Also in this case, there is some more information with respect to the Google global mobility report [10], [26] that registered a global reduction at the 30%. In effect also from our data the global average reduction has been at the 31%, while the detailed data provide much more meaning in understanding how this 30% is actually composed.

As reported in Table 10.1, the percentage of reduction in the exploitation of parking facilities in Florence is reported. The reduction has been estimated by clustering them according to their main purpose and services to: hospital, stations, social hubs, market, and justice. It is evident that the reduction of mobility at the 20% (above reported and discussion) also impacted on the parking lots serving the railway stations. Markets area ob-



main Purpose	% of reduction
Hospitals	55,9
railway stations	84,2
social hubs	85,8
markets	45,1
justice	96,1
AVERAGE REDUCTION OF the	69,8

Table 10.1: Reduction of exploitation in parking facilities in Florence depending on their main utilization and purpose.

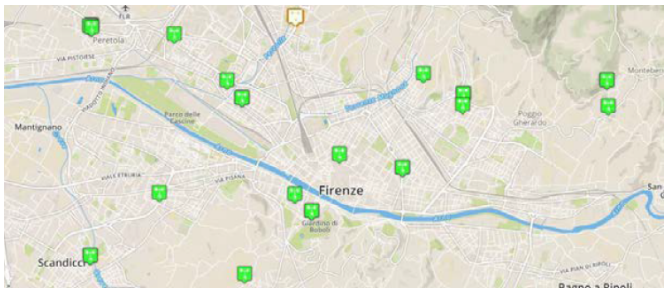


Figure 10.8: Air quality and pollution sensors in the area of Florence, as from <https://www.snap4city.org>

tained a lower reduction due to their primary necessity of services. A strong reduction has been registered to justice and social oriented parking area that have been activities totally stopped by the lockdown.

10.5 Impact of lockdown on environmental data and deductions

Figure 10.8 reports the air quality and pollution station in the Florence area. They are a small part of the air quality and pollution stations of which data are collected by Snap4City. Their data come from sensors of CNR IBE, and ARPAT. Not each of them, but in total they are capable to read: PM10, PM2.5, CO, CO2, NO, NO2, O3, temperature, humidity, and few more.

In the context of environmental monitoring relevant data sources that

could be affected by the above described situation are the values of the pollutants assessed by the above-mentioned sensors which are located in the city. The most relevant parameters that could be affected by the lockdown should be (some of these descriptions have been derived from those of Wikipedia):

- NO₂, NO as NO_x Nitrogen Oxides. NO_x is a generic term for the nitrogen oxides that are most relevant for air pollution, produced from combustion, namely nitric oxide (NO) and nitrogen dioxide (NO₂). NO_x gases are also produced naturally by lightning. Other sources are house heating and industrial plants.
- CO: Carbon monoxide is a colorless, odorless, and tasteless gas that is slightly less dense than air. It is toxic to animals that use hemoglobin as an oxygen carrier (both invertebrate and vertebrate) when encountered in concentrations above about 35 ppm, although it is also produced in normal animal metabolism in low quantities, and is thought to have some normal biological functions. In the atmosphere, it is spatially variable and short lived, having a role in the formation of ground-level ozone.
- CO₂: Carbon dioxide (chemical formula CO₂) is a colorless gas with a density about 60% higher than that of dry air. It occurs naturally in Earth's atmosphere as a trace gas. The current concentration is about 0.04% (412 ppm) by volume, having risen from pre-industrial levels of 280 ppm. Natural sources include volcanoes, hot springs and geysers, and it is freed from carbonate rocks by dissolution in water and acids. Because carbon dioxide is soluble in water, it occurs naturally in groundwater, rivers and lakes, ice caps, glaciers and seawater. It is present in deposits of petroleum and natural gas. CO₂ is produced by all aerobic organisms when they metabolize carbohydrates and lipids to produce energy by respiration. It is returned to water via the gills of fish and to the air via the lungs of air-breathing land animals, including humans.
- PM₁₀, PM_{2.5}: Particulate Matter measures are expressed in microgram for cube meter of particles of 10/2.5 micrometers or less, PM_{2.5} are more critical. The size of the particle is a determinant respiratory tract the particle will come to rest when inhaled. Larger particles are generally filtered in the nose, but particulate smaller than about 10

micrometers can settle in the bronchi and cause health problems. The 10-micrometer does not represent a strict boundary between respirable and non-respirable particles. The sources can be natural (soil erosion, marine spray, volcanoes, forest fires, pollen dispersion, etc.) or anthropogenic (industries, heating, vehicular traffic and combustion processes in general). The major components of atmospheric particulate matter are sulphate, nitrate, ammonia, sodium chloride, carbon, mineral dust and it is estimated that in some urban contexts more than 50% is of secondary origin.

An impact of lockdown can be observed comparing the February/March period with respect to April in the most crowded areas of Gramsci and Ponte alle Mosse in Florence. The different trends allow to compare the traffic flow (that has been strongly affected by the lockdown) with respect to the environmental variables such as: PM10, NO2, CO, etc. Among them the most influenced by the lockdown has been the NO2 due to its dependence to the traffic flow, heating and industrial activities. On the other hand, also the other pollutants have been influenced by the reduction of human activities. In order to assess the impact of the lock down on those environmental and air quality indicators a comparison of the average values of the above mentioned pollutant has been estimated to see the percentage of decrement with respect to the normal period, namely comparing February 2020 with respect to April (the central part of the lockdown). The results are reported in Table 10.2, from which is it is evident that the reduction of traffic at the 20% described above is one of responsible of the reduction of NO2 at the 39%. On the other hand, the reduction has not been proportional since the probably the heating of the houses have been increased in the same period, so that partially compensating the decrement of traffic and industry, which in Florence are not impacting very much in the city. Please note that the effects of the traffic into cities and the estimation of predictions for the NOX on the basis of traffic, wind direction, and structured of the city are studies and produced by the TRAFair project, of which the results for Tuscany area are available on Snap4City Dashboards [121].

10.6 Conclusions

The analysis performed identified a strong reduction of the mobility and transport activities coherently to what has been described by Google Global

	CO2	CO	NO2	PM10
Percentage at which the average value of the pollutant has been reduced (February wrt April, 2020)	41,06	47,54	38,84	61,99
Percentage at which the MAX value of the pollutant has been reduced (February wrt April 2020)	38,89	34,62	63,02	60,21

Table 10.2: Reduction of the pollutant in the period of Lockdown with respect to February 2020.

mobility report and in addition putting in evidence a number of detailed aspects that allowed us to better judge the impact of the lockdown and potential effects of the reopening on an eventual revamping of the infection computing and observing the R_t and its trend. To this end, the collected data from the field have been compared from those of Google and some considerations with respect to the Imperial college Report 20 have been derived. It is probably too early to derive conclusions since this report has been written the 10th of May, only 6 days after the formal reopening of the 4th. The resulting traffic volume has been reprimed to more than the 53%, which is a +33% with respect to the lockdown situations. This indicator and similar ones will be monitored and highlighted in view of the future return of tourists in Florence UNESCO site (as pilot of Herit-Data project). For parking, we still have to see an increment of their usage with respect to the lockdown conditions. As regard the environmental variable and thus of air quality the amount of pollutant has been strongly reduced by the lockdown. The largest reductions have been recorded for NO2 (also due to the reduction of traffic), CO, PM10 and CO2 for the reduction of human activities.

Chapter 11

Classification of RDF datasets through Multilayer Metrics

More and more Linked Data repositories (RDF datasets) are being published in the last years. Directories exist where they are catalogued, but it is still not easy to understand at the first glance the ontological model that RDF datasets adopt, and thus in substance their purpose and aim. For example, for each RDF dataset one should be capable to understand if it is a dictionary or the base of an expert system, if it is an encyclopaedia or a smart city model, and so on. Indeed, the idea behind triples is to have the possibility of reusing them as Web of things and definitions. Proposals can be found in the literature of metrics used for assessing the quality and purpose of RDF datasets. In this chapter, we revise, analyse, and classify those metrics according to their aim and meaning, and we define a set of additional metrics. Then, a Principal Component Analysis is performed for identifying those aspects that are of major interest for the automatic classification of RDF datasets, and for identifying those metrics that are the most suitable for quantifying each of such aspects. RDF dataset clusters are generated based on Principal Components. Five clusters result from the process (thesauruses, lists, gazetteers, semantic networks, and classification schemas), that correspond to structural categories of the DCMI/NKOS KOS Types Vocabulary. This way, clustering by purpose of RDF datasets is made system-

*atic, that is the key contribution that this work provides. Distributed Java applications have been developed to compute metrics. The study has been supported by Sii-Mobility project on smart city services for mobility and transport.*¹

11.1 Introduction

Linked Data guidelines have been adopted by an increasing number of data providers over the years, leading to the creation of a global data space containing billions of assertions, the Web of Data [55], [2], [54], distributed across a huge number of RDF datasets. For that the efficiency and effectiveness of data retrieval and linkage could be assured, a characterization of RDF datasets as for their structure/purpose is necessary. As the set of the data sources increases in size, it becomes more and more important to systematize and automatize the process. With this goal in mind, we have reviewed metrics proposed in the literature for the characterization of data and data model of a RDF dataset, and we have defined some additional metrics. Principal Components have been extracted from full set of existing and newly defined metrics, and a semantic has been identified for each of them. After that, a clustering of RDF datasets has been performed based on Principal Components, and a semantic has been identified for each cluster. Remarkably, it has been possible to give each cluster a semantic that matches one of the RDF dataset categories defined in the DCMI/NKOS KOS Types Vocabulary, that allow us to conclude that a key step has been actually made in the direction of the automatic categorization of RDF datasets by structure/purpose.

Many proposals can be found in the literature about descriptive metrics for RDF datasets. See Section 2.4 in Chapter 2 for a detailed review.

In Table 11.1, metrics at the state of the art are classified, each with the indication its main purpose, type, and source.

This chapter is structured as follows. In Sections 11.2, 11.3, 11.4 we propose a multilayer set of metrics, made up of Endpoint Metrics (direct measurements at the RDF dataset level), Middle-Level Derived Metrics (indirect measurements at the RDF dataset level), and High-Level Metrics (global

¹*Acknowledgments:* The authors would like to thank the MIUR, the University of Florence and companies involved for co-founding Sii-Mobility SCN 112 national project on smart city mobility and transport in which a large usage of ontologies and knowledge based has been done, defining and exploiting the Km4City ontology, which in turn is based on a large number of other vocabularies [51].

Purpose	Type	Metric
Quality	Cohesion	IM [71], EX [71], Subclass Usage [28]
	Connection	TM/IN [71], Rank [71], Links [28]
Cost	Coverage	CPV [71], IPV [71]
	Cohesion	Class/Property Hierarchy Depth [28]
Description & Data Retrieval	Coverage	SWD Encoding [71], Language [71], Language Species [71], Schema Heterogeneity [61]
	Density	Frequency of Predicates [61], Literal length [61] [28]
	Volume	Histograms [101]; Entity Size [61]; Triples, Resources, Literals, Blank Nodes, Typed & Labeled Instances [28]; Number of Triples, Instances and Instantiated Classes [52]
	Stats	Numeric Property Range & Avg. [28]
Characterization	Coverage	Top-100 classes document frequency and probability distribution, and Resource URI coverage by RDF datasets/predicates [61]
	Density	Ontology Ratio [71], Triple Pattern [61], Term Distribution [61]
	Volume	Data Format [61]; Class, Property, and Datatype Instances, Language Usage, indegree & outdegree [28]; Class & Property Instances [52]; Extensional Metrics [52]
	Connection	Subject, Predicate and Object Vocabulary Histogram [28]

Table 11.1: Purpose and type of the metrics at the state of the art

measurements). In Section 11.5 a categorization of metrics used and/or defined in this chapter is provided. In Section 11.6, challenges faced for computing the metrics are outlined. In Section 11.7, the results obtained from the computation of the defined metrics on accessible RDF datasets are provided. In Section 11.8, Principal Component and Cluster Analysis results are presented. In Section 11.9, conclusions are drawn. In Appendix A the list of Acronyms with their definition to help the reader in following the definitions.

11.2 Metrics

In this section, metrics that are proven to have a key role in the clustering by purpose of RDF datasets are described in detail. For each of them, it is indicated what it measures, and why, also including SPARQL queries used for its computation when applicable. Some of these metrics, named as Absolute Metrics in the following, are absolute measurements performed on RDF datasets, and they are therefore not suitable for clustering, since the size of the RDF dataset intended as the quantity of the triples stored in it has an impact on them, and similar datasets under a given point of view could therefore appear to be very different just because they are very different in size. Other metrics, named as Relative Metrics, are based on Absolute Metrics, but they are thought to be invariant to the RDF dataset size, and they are proven to be actually more suitable for clustering, as we will see in the following, when we will present the extraction of Principal Components.

11.2.1 Absolute Metrics

A list follows of metrics that measure absolute quantities at RDF datasets, and that are proven to be useful for clustering datasets by purpose, directly or indirectly (as factors of useful Relative Metrics).

The Triples Count Metric (TCM) [28] [52], measures the number of triples of a given endpoint, including the triples that represent the ontology (or the ontologies). It is a simple and meaningful measure, since it allows to reply to the question “How Big are Data out there?”. Indeed, many definitions of Big Data have been proposed over the time by the scientific and the business community, that do not clarify the relation between the high volume

of data and the Big Data [135]. Measuring the size of the Big Data could help understanding what Big Data actually are in the real-world at today, and how the concept evolves over the time. The TCM metric has been proposed in [28] and [52]. When possible, i.e., in RDF datasets which are not very large, a response to the query is provided in a reasonable time and that are exposed through endpoints that accept SPARQL 1.1 queries (the COUNT function was not available in SPARQL 1.0), the metric is computed as follows: `SELECT COUNT(*) { ?s ?p ?o }`

In those cases, when it is not possible to compute the metric through the simple submission of such a query due to language limitations or to the size of the RDF dataset, triples are retrieved chunk by chunk, and counted programmatically segment by segment with a very time consuming process. The same applies to all count metrics defined in this chapter.

The Class Count Metric (CCM) measures the number of distinct classes in the RDF dataset. It is computed through the observation of the triples in the form “<A> a ”. Therefore, only the instantiated classes contribute to the measure, even if the ontology is loaded in the RDF dataset. The CCM Metric provides an indication about how much thematic the RDF dataset is. Indeed, a huge number of instantiated classes indicate that there is a huge number of different concepts represented in the RDF dataset, that is what can be often observed in the general purpose RDF datasets. On the other hand, a limited number of instantiated classes indicate that the concepts that are represented within the RDF dataset are almost all of the same type, that is what can be observed in the thematic RDF datasets, with limited domain. The metric has been proposed in [28], [52]. When possible, it is computed as follows: `SELECT COUNT(DISTINCT ?o) { ?s a ?o }`

11.2.2 Graph Count Metric (GCM)

The Graph Count Metric measures the number of the graphs included into the RDF dataset. GCM could be of help for estimating the RDF dataset scope and boundaries, and probably also the number of data sets and/or domains in the store. It should be noted, that a RDF dataset can model several points of view, e.g. geographic, temporal as time/date, family of entities for modeling domains, and so on. This aspect seems to be neglected in the literature about store categorization, and assessment. CGM metric can be computed by using: `SELECT COUNT(DISTINCT ?g) { GRAPH ?g { ?s ?p ?o } }`

11.2.3 Instance Count Metric (ICM)

The Instance Count Metric measures the number of resources in the RDF dataset is a volume/size metric. ICM metrics, combined with the total number of triples TCM could help identifying the RDF datasets that are used at support fo knowledge base from those only modelling domains with a few of instances. The metric has been proposed in [28], [52], and can be computed as follows: `SELECT COUNT(DISTINCT ?s) { ?s a ?o }`

11.2.4 Same As Metric (SAM)

The sameAs property is part of the Web Ontology Language [15], and it is used to relate resources that represent the same thing. On one hand, the property is also abuses as investigated by Harry Halpin et al., [79]. On the other hand, we consider the counting of this property instances in RDF dataset, that could provide an indication about how much the resources are connected each other and with the external, and how much the reasoning is supported, that are key aspects of a RDF dataset. The computation of this metric has been proposed in [28], while a the count of instances for properties has been proposed in [52]. For the computation of SAM metric, it can be used: `SELECT count(*) {?s owl:sameAs ?o}`

11.2.5 Subclass Count Metric (SCM)

The Subclass Count Metric indicates how many classes defined in the model included into the RDF dataset are specializations of one or more other classes. It results from the count of the triples where the predicate is `rdfs:subClassOf` [133], and can be performed as follows: `SELECT (COUNT(?s) AS ?count) WHERE { ?s a ?c . ?c rdfs:subClassOf ?o }`

11.2.6 Property Count Metric (PCM)

The Property Count Metric counts the number of distinct predicates in the store. PCM can be combined with other metrics to provide useful indications for a user-oriented characterization of RDF datasets. The production of a simple index to count distinct properties instantiated in the RDF dataset has been proposed in [28]. PCM can be computed as follows: `SELECT COUNT(DISTINCT ?p) { ?s ?p ?o }`

11.2.7 Object Classes Metric (OCM)

The Object Classes Metric provides the number of distinct classes to which the fillers of triples belong. OCM metric, proposed in this chapter, can be useful for generating user-oriented characterizations of RDF datasets, and can be computed as follows: `SELECT COUNT(DISTINCT ?c) {?s ?p ?o . ?o a ?c }`

11.2.8 Language Metric (LAM)

Literals can be stored in the store with associated meta-data to identify the language in which the literal is written. A sample SPARQL query [9], [22], follows that inserts a triple whose filler is a literal with an associated language: `INSERT DATA { <http://foo.com#s> rdfs:label "str"@en }`

Also, lang function of the SPARQL [9], [22], provides the language of the input variable, in those (rare) cases when it is specified. The result of LAM metric, defined in this chapter, is a JSON object including keys languages, which have been associated with at least one literal in the RDF dataset, and whose values are the number of the literals that are found for each language. It should be noted that, the language string is filled by the users that insert the triples. Therefore, it may be un-correct, and this fact has to be taken into account in computing the metric. The investigation of languages in the RDF dataset is of great interest to understand the wideness of the applications, and geographical extent of the RDF dataset.

11.2.9 Ontology Metric (OME)

The Ontology Metric investigates the representation of the ontology (or ontologies) in the store. The result of OME computation is a JSON where all the triples of each ontology is represented. Indeed, ontologies also can be represented in form of tuples [133]. The assessment of OME for a number of stores can be used to produce statistics about how much diffused the habit of exploiting a data model. The simplified version of OME has been proposed in [52] with metric called Intensional Knowledge Triples (IKT). IKT can be used to perform a guess of the main topic of the RDF dataset, i.e., a user-oriented gross categorization by topic.

11.2.10 Instantiated Object Properties By Class (IOP)

This metric provides an indication about which object properties are used for subject's class. Thus, for each subject's class and object property, IOP indicates how many triples can be found in the RDF dataset that set the given property for a subject of the given class. IOP metric can be computed by the algorithm proposed in [52], where a compromise is reached between efficiency of computation. Please note that the computation of IOP on very large RDF datasets may lead to a timeout. IOP metric corresponds to Subject Class (Sc) metric proposed in [52], and similar measure has been also proposed in [28], where the partitioning is operated by subject, instead of by subject class. IOP metric could be of help in making reasonable guesses about the informative content of the RDF dataset, and therefore about its adherence to a given user need. The assessment of IOP for a number of stores could be of help in intercepting differences in the diffusion of a given property among different concepts (classes).

11.2.11 Instantiated Data Properties By Class (IDP)

This metric provides an indication about which data properties are used in store by subject's class. For each pair (class, property) the metric indicates how many tuples can be found in the store such that (i) the subject is of type class; (ii) the predicate is property. The metric can be computed by using the algorithm proposed in [52], where a compromise is reached between precision and efficiency working for very wide RDF datasets where queries may lead to a timeout. The metric corresponds to the Subject Class to Literal (Scl) metric that is proposed in [52], and similar measure has been also proposed in [28] where the partitioning is operated by subject, instead of by subject class. The IDP metric could be of help in making reasonable guesses about the informative content of the RDF dataset, and therefore about its adherence to a given user need. Comparing the metric assessed in different stores may allow intercepting differences in the diffusion of a given property across different concepts (classes).

11.2.12 Vocabulary Metric (VOM)

The Vocabulary Metric is aimed at assessing the diffusion of Linked Open Vocabularies, LOV [133] via their terms. The VOM metric return a JSON object whose keys are: (i) the URIs of the Linked Open Vocabularies [133]

that are used in the RDF dataset, and (ii) the counts of the occurrences of the terms in the RDF dataset. The VOM metric, and most of all the aggregated metrics that could be derived from it, helps monitoring the popularity of a vocabulary over the time, finding correlations, and making innovative analysis about what makes that success (or the failure) of a vocabulary. In addition, it could help the semantic knowledge architects to identify (and choose) the most widely used terms, that are very likely to lead to an improved support of the reasoning in a global scale. The computation of this new original metric is based on the identification of properties and classes that are in use within the RDF dataset, and on the submission of ad-hoc SPARQL requests to the LOV endpoint to identify those classes and properties that are terms of a LOV, each having the following form: `SELECT * { <class/property URI> rdfs:isDefinedBy ?v . ?v a http://purl.org/vocommons/voaf#Vocabulary }`

11.2.13 Weighted Number of Relatives (WNR)

This metric is aimed at estimating an index to guess how complex are the relations of inheritance among classes that are represented/instantiated in the RDF dataset. It is inspired by (and it shares its motivation with) a set of analogous metrics that can be found in the literature, such as: the Depth of Inheritance (DIT) [62], the Number Of Superclasses (NSUP) [59], the Number Of Children (NOC) [62], and the Number of Subclasses (NSUB) [59]. For each class, the WNR is determined as follows:

$$WNR_c = \sum_{p \in P} dist_{c,p} + \sum_{s \in S} \frac{1}{dist_{c,s}} \quad (11.1)$$

where: (i) P is the set of the parent classes of class c; (ii) S is the set of subclasses of class c; (iii) dist is the distance between two classes in the inheritance tree. The metric value for a RDF dataset results from the sum of the values computed at each of the classes that are part of the data model loaded in the RDF dataset.

11.2.14 Coupling Between Resources (CBR)

This metric has been inspired by the Coupling Between Objects metric [62], from which it borrows its motivation. For each class C, instantiated within a RDF dataset, the CBR results the cardinality of classes C' that satisfy at

least one of the following criteria/feature: (i) at least one instance of classes C' is the filler of a triple where an instance of C is the subject; or (ii) one instance of C' is the subject of a triple where an instance of C is the object; (iii) C' is the filler in a triple whose subject is C ; (iv) C' is the subject in a triple whose filler is C . The value of CBR metric for a RDF dataset is estimated as the sum of the values computed for each class of the RDF dataset.

11.3 Middle-Level Derived Metrics

In this section, a set of metrics to assess RDF dataset status are presented. They exploit the metrics defined in Section 11.3, and therefore they can be regarded as Middle Level Derived Classes.

- Data Property Count Metric (DPCM) counts the number of Instantiated Data Properties by Class (IDP) metric.
- Languages Count Metric (LCM) counts the number of distinct languages of the store, so that: $LCM = \dim(LAM)$.
- Vocabulary Count Metric (VCM) counts the number of distinct vocabularies per endpoint as $VCM = \dim(VOM)$.
- Vocabulary Sum Metric (VSM) counts triples where a term of an external vocabulary appears. It is calculated summing the values in the Vocabulary Metric (VOM).
- Property Count by Class Metric (PCCM) counts the number of distinct pairs (subject class, predicate) instantiated in the store: $PCCM = OPCM + PCCM$.
- Subclass Ratio Metric (SRM) is defined as $SRM = SCM / CCM$ and provides an indication of how deep the modelling of the data is within the RDF dataset, and how much the reasoning is supported within the RDF dataset.
- Languages Sum Metric (LSM) counts the number of literals with language in store. It is computed summing up the values of LAM, excluded the one that relates to literals having no indication about their language.

- Triples Per Resource (TPR) is defined as $TPR = TCM/ICM$. This metric indicates in average how many triples can be found for each entity in the store. It can be regarded as quantification of the data volume/cost metric, and only in specific case related to complexity.
- Ontology Size Metric (OSM) is the total number of the triples that represent the data model, which corresponds to counting the JSON objects in the Ontology Metric (OME).
- Object Property Count Metric (OPCM) counts the number of Instantiated Object Properties by Class (IOP) metric.
- Object Property Sum Metric (OPSM) counts the number of triples whose filler is a resource, the sum of numeric values in the Instantiated Object Properties by Class (IOP) metric.
- Data Property Sum Metric (DPSM) counts the number of the triples whose filler is a literal, the sum of numeric values in the Instantiated Data Properties by Class (IDP) metric.
- Class Ratio Metric (CLRM) is TCM/CCM . It is a measure of cardinality for classes.
- Property Ratio Metric (PRM) is TCM/PCM . It is a measure of density of triples with respect to properties.
- Property Ratio by Class Metric (PRCM) is $TCM/PCCM$. It is a measure of density of triples with respect to classes and properties.
- Literals Sum Metric (LISM) is the total count of literals, and Literal per Language Metric (LPLM) is $LISM/LCM$. Both them are volume metrics that contribute characterizing a RDF dataset as for its intended audience/purpose.
- Vocabulary Ratio Metric (VRM) is VSM/VCM . It is a measure of density of triples with respect to vocabularies.
- Data Property Ratio Metric (DPRM) is $DPSM/DPCM$. It is a measure for the weight of semantic models in RDF datasets.

11.4 Metrics Categorization

In Tables 11.2, 11.3, a characterization of metrics described in previous sections is reported, where the metrics reviewed from the literature are reported with the citation. The classification is based on metric type and boundary.

As for the boundary of investigation, metrics can be: (i) internal, if they are computed by using internal elements of the triple-store; (ii) external, if they have to be computed taking into account what can be found across the boundaries of the investigated RDF dataset; (iii) global, if they are computed considering the totality of the alive assessed RDF datasets.

As for the metric type, metrics can measure a: (i) coverage, they are typically global metrics measuring features of RDF datasets regarding the usage of vocabularies and accepted syntax; (ii) volume, they indicate how many elements of given type can be found in a store; (iii) density, they are metrics regarding how many elements of a given kind are present in a store with respect to the volume of the store estimated on the basis of some volume metrics; (iv) cohesion, they indicate how many classes of the store are interlinked with others within and outside the store, for example due to inheritance, mutual exclusion, and other relations; (v) connection, they indicate how many elements are interlinked through object properties.

A characterization of metrics can be also performed on the basis of the main purpose for which they have been defined (see Table 11.4), such as: (i) quality assessment, e.g., for evaluating how much the guidelines for building semantic data models and linked datasets have been followed in the RDF dataset; (ii) cost estimation (or a-posteriori determination), related to the building and/or maintenance of a semantic data model; (iii) informative, e.g., metrics that measure the available non-structured descriptive information; (iv) characterization, i.e. the automatic labelling of a RDF dataset through a process that involve feature (metric) selection and clusters building.

11.5 Metrics Computation Tool

The computation of metrics is performed through a set of Java applications each of which is responsible of a specific aspect of the computation, namely the:

- setup for the retrieval of the list of the endpoints and RDF datasets from a configurable list of endpoint directories;

Boundary	Type	Metric
Internal	Density	Inst. Obj. Prop. By Class (IOP) [52] Inst. Data Prop. By Class (IDP) [52] Inst. Obj. Prop. By Obj. (IPO) [52] Triples Per Resource (TPR) Class Ratio Metric (CLRM) Property Ratio Metric (PRM) Property Ratio by Class Metric (PRCM) Instance per Class Metric (IPCM) Object Property Ratio Metric (OPRM)
	Semantic Model	Object Classes Metric (OCM) Data Property Ratio Metric (DPRM) Property Count Metric (PCM) [28] Ontology Size Metric (OSM)
	Volume	Inst. Class Metric (ICLM) [28] [52] Inst. Property Metric (IPM) [28] [52] Property Count by Class Metric (PCCM) Object Property Count Metric (OPCM) Object Property Sum Metric (OPSM) Data Property Sum Metric (DPSM) Literals Sum Metric (LISM) Resource Size Metric (RSM) Triples Count Metric (TCM) [28] [52] Class Count Metric (CCM) [28] [52] Graph Count Metric (GCM) Instance Count Metric (ICM) [28] [52] Languages Sum Metric (LSM) Literal per Language Metric (LPLM)
	Cohesion	Broader Count Metric (BCM) [27] [134] Subclass Metric (SCM) Subclass Ratio Metric (SRM) Weighted Number of Relatives (WNR) “Same As” Metric (SAM) [28] Coupling Between Resources (CBR)
	System	Alive Endpoint Metric (AEM) Version Metric (VME) Language Metric (LAM) Languages Count Metric (LCM) Localized Triples Ratio (LTR) Ontology Metric (OME) [52] Ontology Ratio Metric (ORM)

Table 11.2: Metrics characterization by boundary and type (continue)

Boundary	Type	Metric
External	Density	Vocabulary Ratio Metric (VRM) Predictive Model Complexity (PMC)
	Connection	“See Also” Metric (SEM) Vocabulary Metric (VOM) Vocabulary Count Metric (VCM) Vocabulary Sum Metric (VSM) Links Metric (LIM) [28]
Global	Coverage	SPARQL 1.1 Endpoints Ratio (R11) Linked Namespaces Ratio (LNR) Vocabulary Ratio (VOR) Vocabulary Terms Ratio (VTR) Ontology Ratio (ONR)
	Density	Alive Endpoints Ratio (AER) Language Ratio (LAR) Avg. Localized Literals Ratio (AG-LTR) Class Ratio (CLR) Property Ratio Ratio (PRR)
	Volume	Alive Endpoints (AES) SPARQL 1.1 Endpoints (E11) Count Metrics Global Triples Count Metric (G-TCM) Global Broader Count Metric (G-BCM) Global Class Count Metric (G-CCM) ... Global Graph Count Metric (G-GCM) Global Instance Count Metric (G-ICM) Global Property Count Metric (G-PCM) Language Ranking (LAK) Class Ranking (CLK) Property Ranking (PRK) Vocabulary Terms Ranking (VTK) Avg. Resource Size (AG-RSM)
	Cohesion	Avg. Broader Count Metric (AG-BCM) Avg. Weighted Num. of Rel. (AG-WNR) Avg. Coupling Between Res. (AG-CBR) Avg. Subclass Metric (AG-SCM) Avg. Subclass Ratio Metric (AG-SRM)
	Connection	Linked Namespaces Ranking (LNK) Vocabulary Ranking (VOK)

Table 11.3: Metrics characterization by boundary and type (continued)

Purpose	Metrics
Quality	Alive Endpoint Metric (AEM) Broader Count Metric (BCM) [27] [134] Links Metric (LIM) [28] Ontology Metric (OME) [52] Subclass Metric (SCM) Subclass Ratio Metric (SRM) Vocabulary Metric (VOM)
Cost	Predictive Model Complexity (PMC) Weighted Number of Relatives (WNR) Coupling Between Resources (CBR)
Description & Data Retrieval	Version Metric (VME) Triples Count Metric (TCM) [28] [52] Class Count Metric (CCM) [28] [52] Graph Count Metric (GCM) Instance Count Metric (ICM) [28] [52] Property Count Metric (PCM) [28] Localized Triples Ratio (LTR)
Characterization	“Same As” Metric (SAM) [28] “See Also” Metric (SEM) Object Classes Metric (OCM) Language Metric (LAM) Instantiated Class Metric (ICLM) [28] [52] Instantiated Property Metric (IPM) [28] [52] Instantiated Obj. Prop. By Class (IOP) [52] Instantiated Data Prop. By Class (IDP) [52] Instantiated Obj. Prop. By Obj. (IPO) [52] Triples Per Resource (TPR)

Table 11.4: Main purposes of the metrics proposed in this chapter

- scheduler initializes the computation sessions, identifying the full set of the SPARQL endpoints to be queried, and scheduling the Alive Endpoint Metric (AEM) for each of those, and the full set of the active High-Level Metrics, before going to sleep in the background waiting for the completion of all the scheduled computations;
- computation module iteratively takes in charge all the scheduled computations by means of a multithread approach;
- database module where resulting data metrics are stored including the computational process status. Since most of the metrics may depend on the results of other metrics according to their definition the former values are used in the computation of the latter.

Computations are organized in sessions and tasks for each endpoint. The scheduler has to respect the capabilities of the end-points avoiding stressing them too much with dozens of queries at the same time.

11.5.1 **The Datameter Web Portal of DISIT Lab**

The Datameter web portal and service (<https://www.disit.org/datameter>) has been developed and published as a part of this research. DataMeter, and it contains (i) the listing of RDF dataset directories each with its most relevant metadata and related resources, such as the listing of monitored RDF datasets discovered thanks to it; (ii) the listing of monitored RDF datasets/endpoints, each with its most relevant metadata and a listing of related resources such as measurements of metrics that have been performed over the time on the given RDF dataset; (iii) the listing of metrics, with indication of their names, boundaries (endpoint vs global), descriptions, and current statuses (whether they are computed or considered to be obsolete and not currently computed), and with listings of their related resources such as the measurements that have been performed over the time of the given metric; (iv) the listing of sessions of measurements each with its most relevant metadata and the listing of measurements performed in the session; (v) the listing of the measurements, and their status at real-time.

An interactive full text search is also available. Also, all listings and resources are available in RDF format, exploiting the HTTP content negotiation, and borrowing concepts and properties from Linked Open Vocabularies [133] such as the DCMI Metadata Terms [41], the Schema.org

vocabulary [3], the Simple Knowledge Organization System [77], the Friend Of A Friend vocabulary [17], the Vocabulary for Attaching Essential Metadata, the Vocabulary of Interlinked Datasets [4], the Provenance Vocabulary Core Ontology [96], the Lifecycle Schema, and the Dataset Quality Vocabulary [12].

11.6 Experimental Results

The endpoint directories that we have queried to recover a list of the endpoints to be investigated have been:

- <https://old.datahub.io>;
- <http://linkeddatacatalog.dws.informatik.uni-mannheim.de>;
- <http://data.gov.uk>.

In Fig. 11.1, an extract of High-Level Metric values obtained by executing the assessment process is presented.

According to Fig. 11.1, a number of facts can be noticed:

- a relevant number of monitored RDF datasets is not alive (see the AES, and AER metrics);
- a large part of alive endpoints accepts SPARQL 1.1 queries (see the E11, and the R11 metrics), that reflects the dynamicity of the universe of the semantic knowledge. The 33
- the average size of a RDF dataset is of hundreds of million triples (see the AG-TCM metric). It should be said anyway, that the average is raised by a limited number of very large RDF datasets.
- Hundreds of thousands broader relations can be found in average (see the AG-BCM metric), that indicates that a fairly high degree of connection can be found among concepts. It should be considered anyway that remarkable differences can be found among RDF datasets, the largest part of the connections being wrapped in a small number of RDF datasets.
- the heterogeneity of concepts that are subject of a triple (see the G-CCM, and the AG-CCM metrics) is greater than the heterogeneity of

Metric	Value
Endpoints	607
AES (AER)	235 (38.715%)
El1 (R11)	157 (66.809%)
G-TCM	35,715,809,750
G-BCM	46,567,416
G-CCM	453,248
G-OCM	39,721
G-PCM	136,879
G-GCM	1,167,891
G-ICM	925,109,610
G-SAM	399,363,851
G-SEM	3,975,020,472
G-SCM	8,225,423,100
AG-TCM	190,993,635
AG-BCM	249,024
AG-CCM	2,424
AG-OCM	212
AG-PCM	732
AG-GCM	6,245
AG-ICM	4,947,110
AG-SAM	2,135,636
AG-SEM	21,256,794
AG-SCM	43,986,220
ONR	10.213%
G-IOP	owl:Class rdf:type 3.0M sgd_vocab:Resource rdf:type 1.7M owl:Class void:inDataset 1.5M
G-IDP	owl:Class bio2rdf_vocabulary:uri 1.5M owl:Class dct:identifier 1.5M owl:Class bio2rdf_vocab:namespace 1.5M
G-IPO	rdf:type rdf:Statement 1.9G void:inDataset dct:Dataset 1.4G ctd_vocabulary:article pubmed_vocabulary:Resource 1.2G
LAK (LAR)	Unspecified 295,588,821 (77.472%) English (en) 47,438,878 (12.433%) Spanish (es) 7,360,290 (1.929%)
LNK (LNR)	Dbpedia 64 (27.234%) Umbel 52 (22.128%) Semantic Science 36 (15.319%)
CLK	ncbigene_vocabulary:Resource 187M http://geovocab.org/geometry 103M #Geometry clinicaltrials_vocabulary:Resource 101M
CLR	owl:Class 60.000% owl:Ontology 58.723% owl:AnnotationProperty 54.043%
PRK	rdf:type 1,051,467,221 rdfs:label 35,967,607 owl:sameAs 34,183,700
PRR	rdf:type 74.043% rdfs:label 42.128% rdfs:comment 35.319%
ONR	10.213%
VOK (VOR)	rdf 179 (76.170%) rdfs 164 (69.787%) Foaf 131 (55.745%)
VTK (VTR)	rdf:type 173 (73.617%) rdf:Property 118 (50.213%) rdfs:Class 109 (46.383%)

Figure 11.1: High-Level Metric Values (rankings truncated at the 3rd place)

Endpoint	# of instantiated classes (CCM)
Dbpedia-Live	370,819
Dbpedia in Greek	14,439
DbkWik	12,071
LinkedDrugs	4,964
LinkedSpending	2,574
TAXREF-LD	1,904
Influence Tracker Dataset	1,875
Wings Workflow Provenance	1,746
OpenEI	1,429
Dbpedia	1,000
...	...

Table 11.5: Endpoints ranked by the number of the used classes

objects (see the G-OCM, and the AG-OCM metrics), that suggests that objects mostly represent at today widely used data types, instead of types of items of the real-world.

- most of the literals have not an explicit indication of the language in which they are written (see the LAK, and the LAR metrics). Scrolling down the ranking of languages (see the LAK, and the LAR metrics again), we observe that the same language is sometimes indicated in different ways for different literals. This is the case of the English language, that can be found indicated as en, EN, en-us, eng. It can be considered to be an index of a low degree of maturity of the development of localized RDF datasets, and it also indicates that some sort of human supervision is somewhat necessary.
- Looking at the ranking of the probabilities of finding a class in a RDF dataset (see the CLR metric), we can see that the top of the ranking is occupied by the owl:Class but it can be found in the 60

Results outlined in Table 11.5 put is evidence the top end-points in terms of CCM metric, cloud count. In the places, general-purpose RDF datasets are present.

Further considerations can be performed by investigating Endpoint Metrics as for their empirical distribution. More specifically, the skewness-kurtosis plot proposed by Cullen and Frey in [65] has been used to help

the choice of a distribution to fit each of those Endpoint Metrics that bear a numeric value. A nonparametric bootstrap procedure (i.e., computed on bootstrap samples by random sampling with replacement from the original data) [73] has been performed to consider the uncertainty of the estimated values of kurtosis and skewness from data. We have accomplished such an estimation to get an indication about the possible empirical distribution for a selection of the Endpoint Metrics, considering the normal, uniform, logistic, exponential (for which there is only one possible value for the skewness and the kurtosis), gamma, lognormal and beta distributions.

Figure 11.2 depicts the skewness-kurtosis plot for Triple Count Metric, TCM, applied on all the above mentioned triple stores, on the basis of which we can observe that the beta distribution suits better than others [113]. This fact is also true for most of the metrics estimated at level of RDF dataset (i.e., those that concern used vocabularies, languages of literals, and other).

11.7 Principal Component Analysis

In this section, Principal Component Analysis (PCA) [91] is presented. The PCA aimed at understanding the underlying structure of the set of metrics, presented in sections 11.3 and 11.4. The first goal is to identify the number of the principal components (PCs) that have to be taken into account, in order to find aspects of interest for the description, assessment and characterization of RDF datasets. Typically, the PCA allows to reduce the dimensionality of the set of metrics needed to describe their variance thus minimizing information loss. The variations in the correlated variables/metrics data can be described in a new set of uncorrelated variables (PCs) that are ordered so that the first few retain most of the variation present in all the original variables/metrics [30].

PCA has been performed on the most representatives 185 endpoints on the basis of the above mentioned Endpoints and metrics. Table 11.6 reports the relevance of factors and eigenvalues of the PCA in descending order of explained variance. The corresponding percentage of the variance is reported in the third column of the table. The first component carries more than the 17.8% of the total variability of the data. The variability of the second factor is about 11%, while Factor 3 explains about 9% of the total variability. As described in the following, the test of Cattell [83] has been used to identify the optimal number of factors. Thus, according to the screen test of Cattell

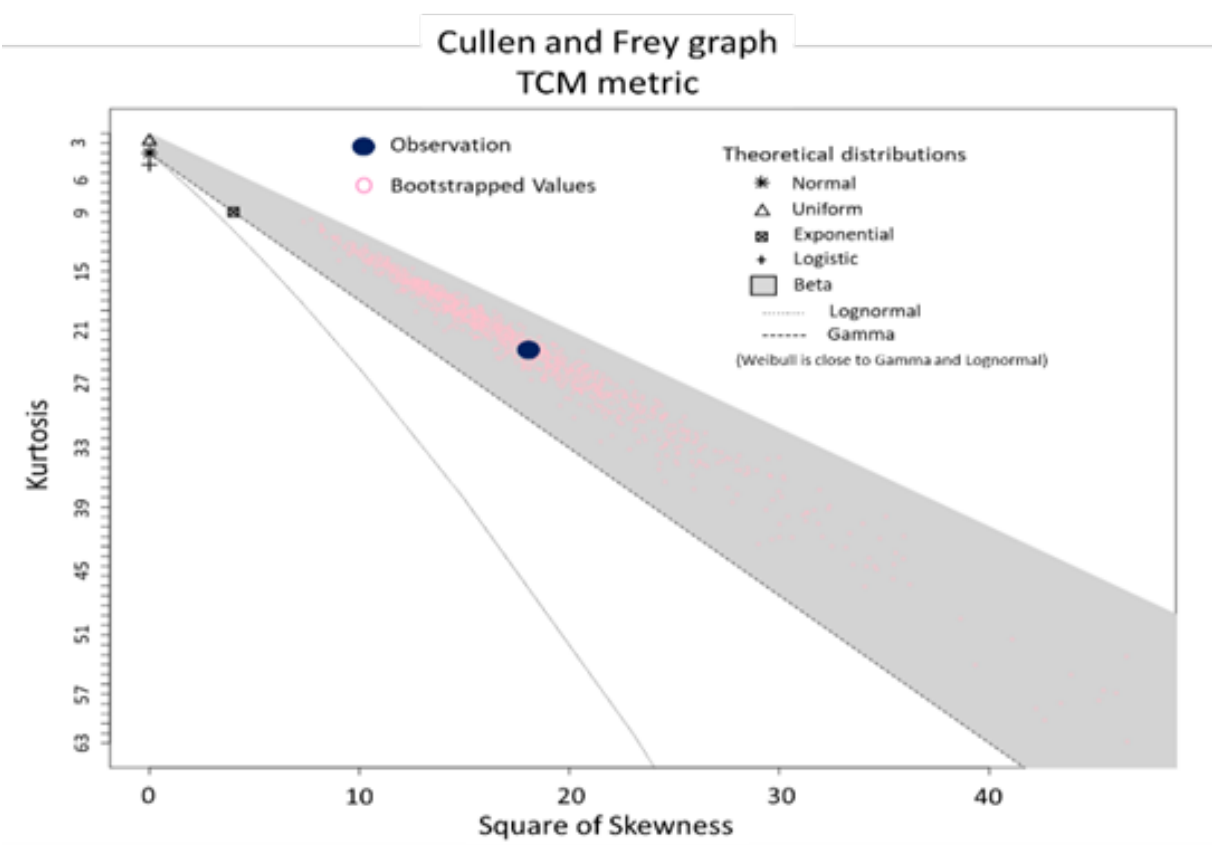


Figure 11.2: Skewness-kurtosis plot for the TCM metric)

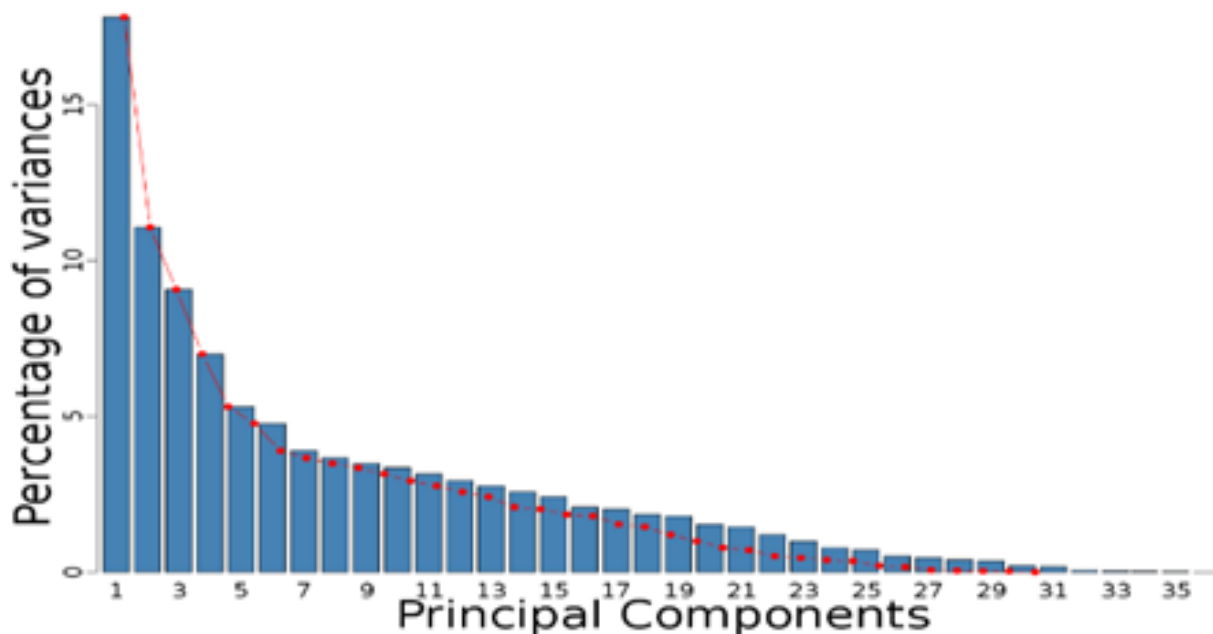


Figure 11.3: Distribution of the percentage of variance from PCA)

(see Figure 11.3), the number of components selected is 6, that corresponds to an elbow in the curve/distribution, i.e., a change of slope, and the 55% of the total explained variance.

In Table 11.7, numbers in bold correspond to the most relevant metrics in some component, while the less relevant metrics are reported in grey. For example, the PC1 is mainly explained by metrics ICM, LSM, LISM, OPCM, OPSM, DPSM, and LPLM. Then the 23 metrics marked in bold (selected from the initial 37) can be used to characterize the RDF datasets, thus reducing the domain assessment.

From the analysis, it is possible to construct a factor map reporting a vector representing the correlation between each feature and the principal components (the axis of the graph). In Figures 11.4, 11.5, 11.6, 11.7, and 11.8, factor map with only metrics that mainly explain the variability of factors (with higher correlation) are reported.

It should be noted that ONLY 4 of the metrics that are the most meaningful for the characterization of RDF datasets (namely the ICM, SAM, TCM,

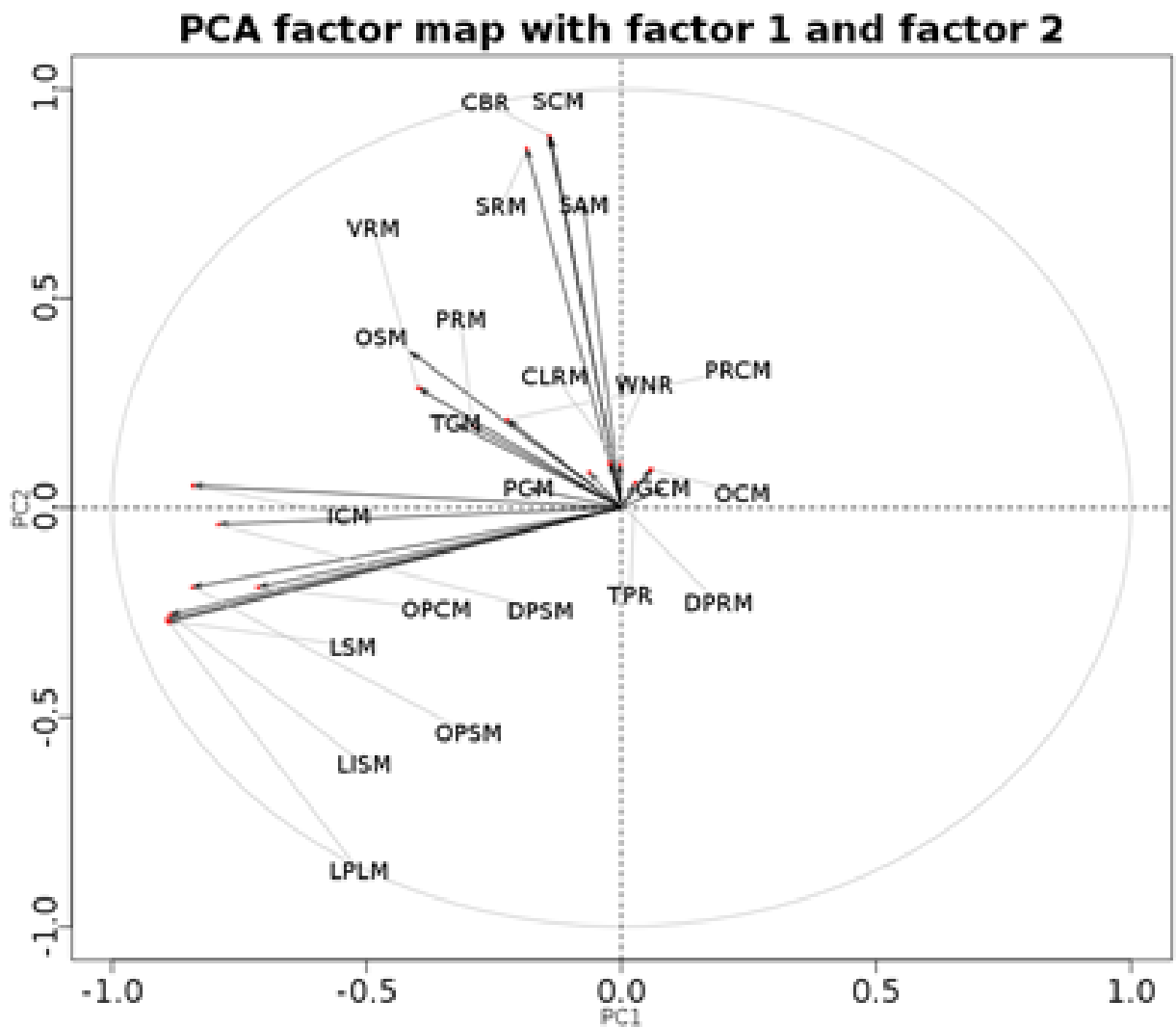


Figure 11.4: PCA factor map with factor 1 and factor 2)

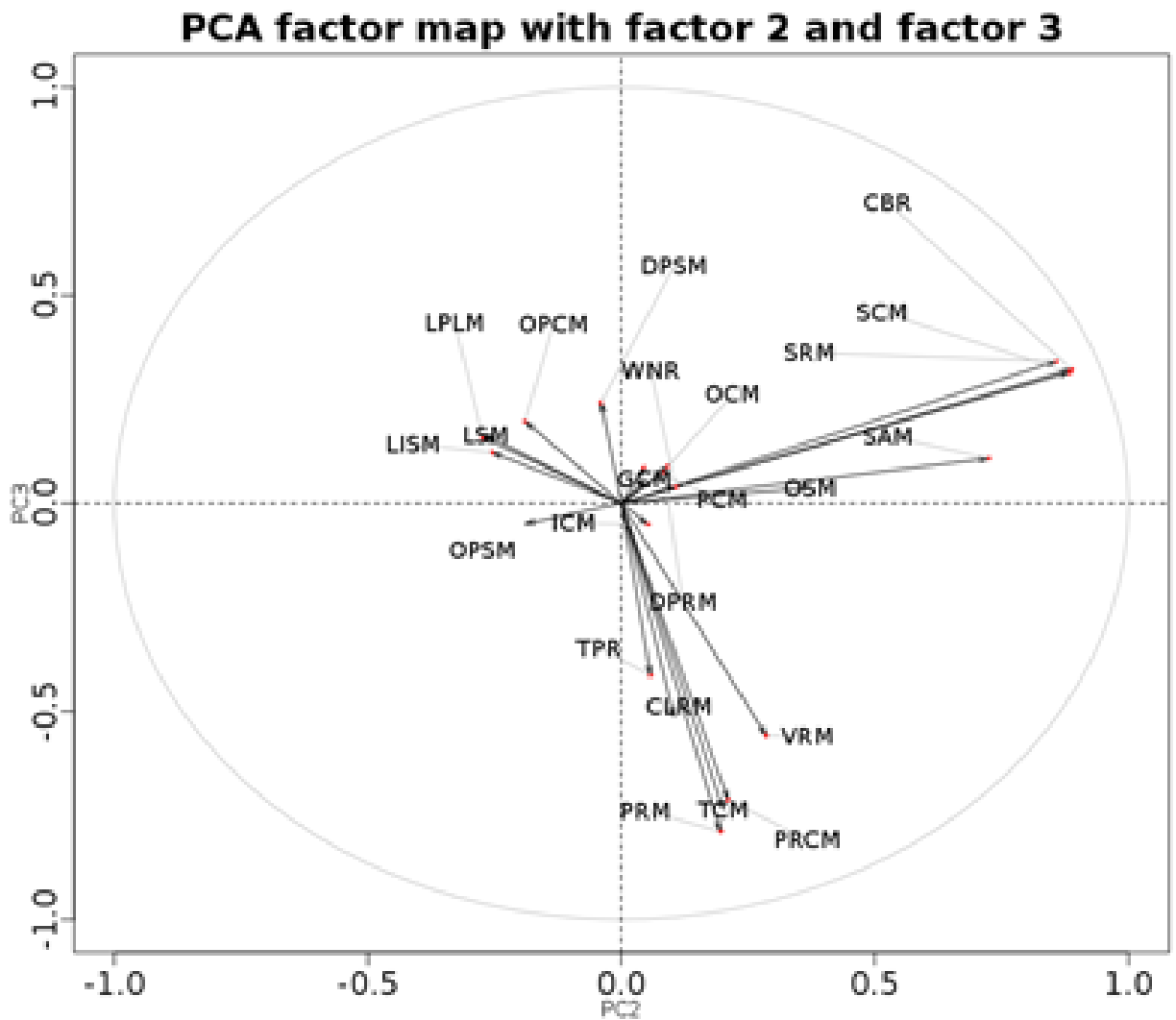


Figure 11.5: PCA factor map with factor 2 and factor 3)

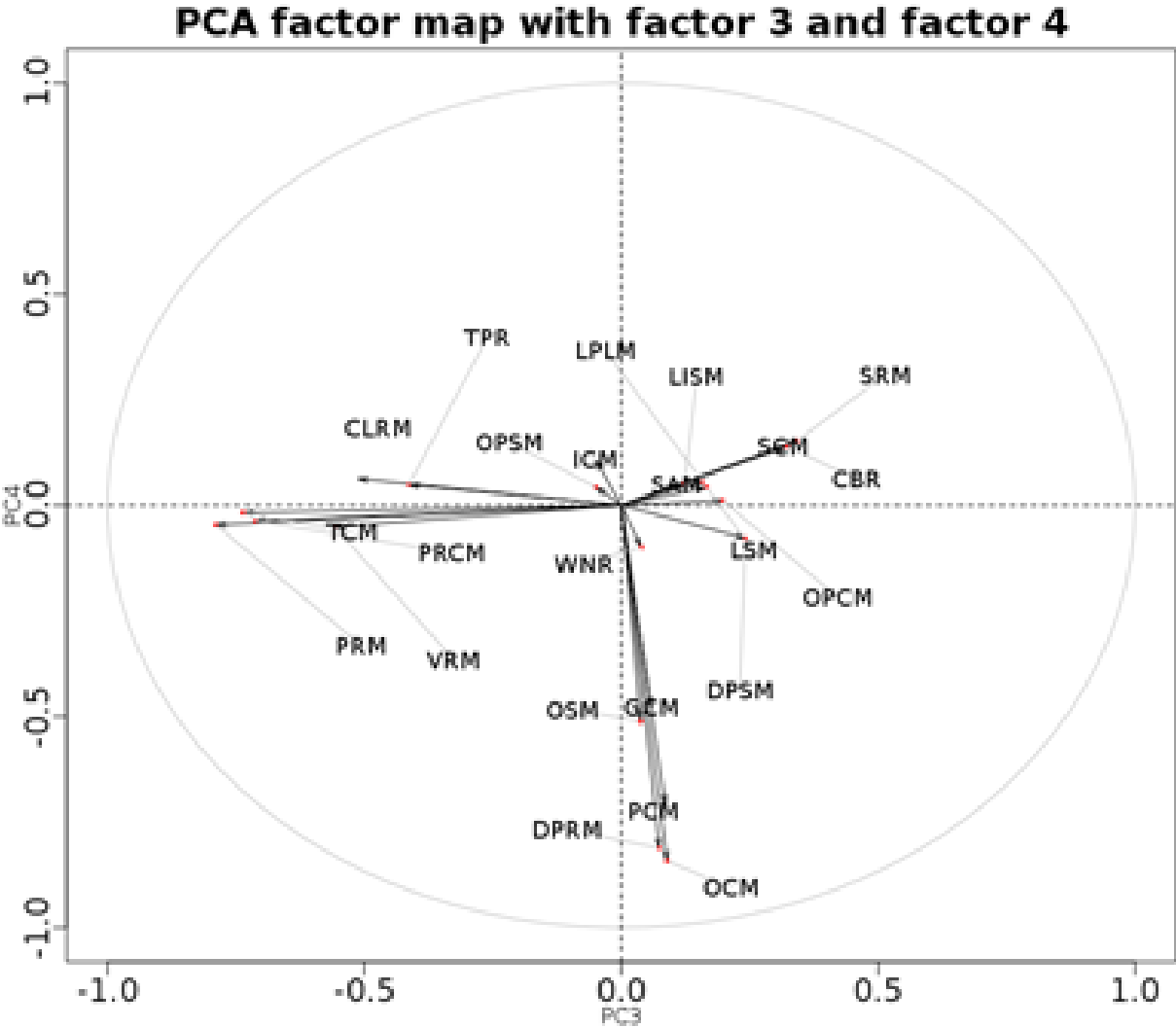


Figure 11.6: PCA factor map with factor 3 and factor 4)

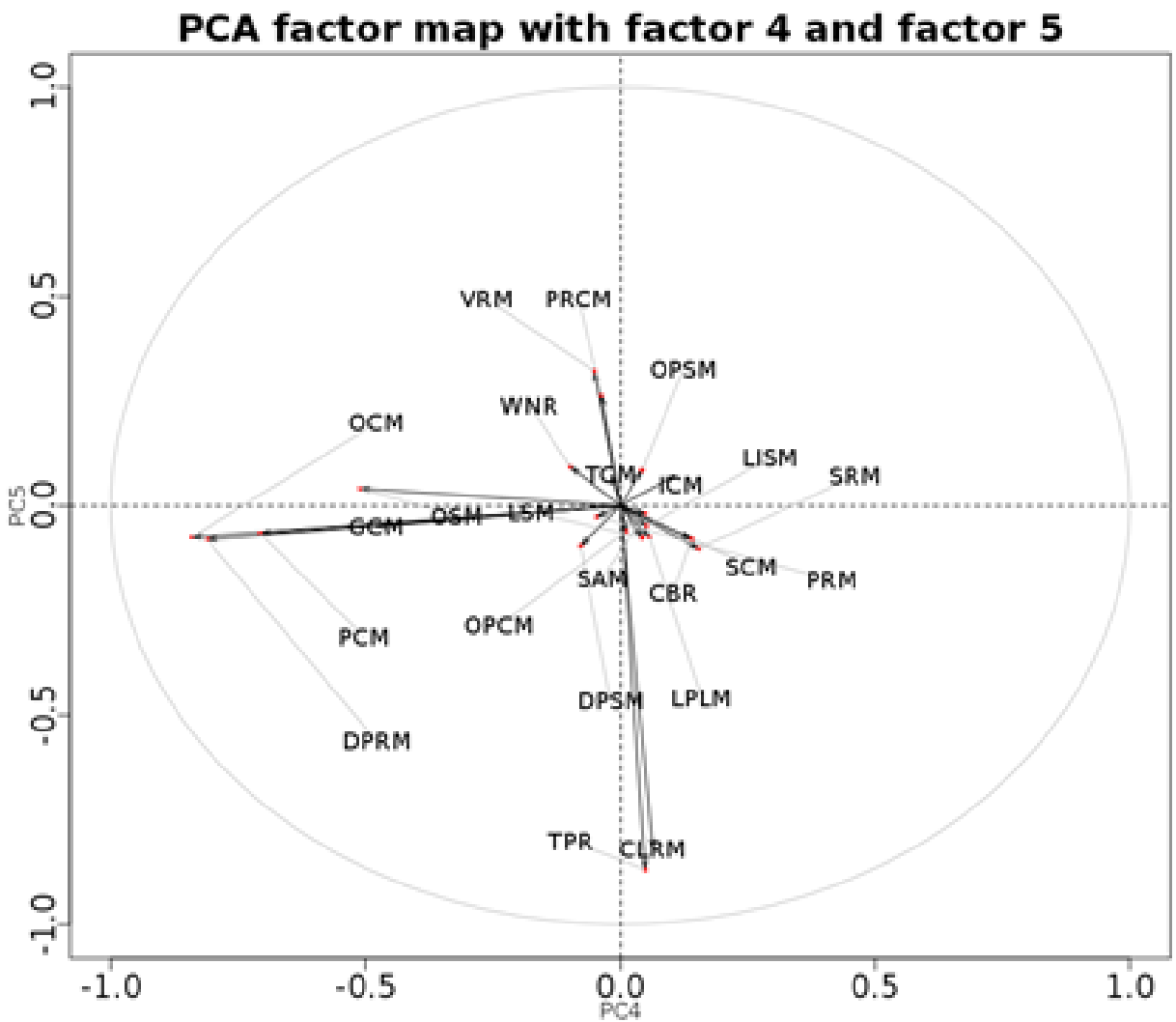


Figure 11.7: PCA factor map with factor 4 and factor 5)

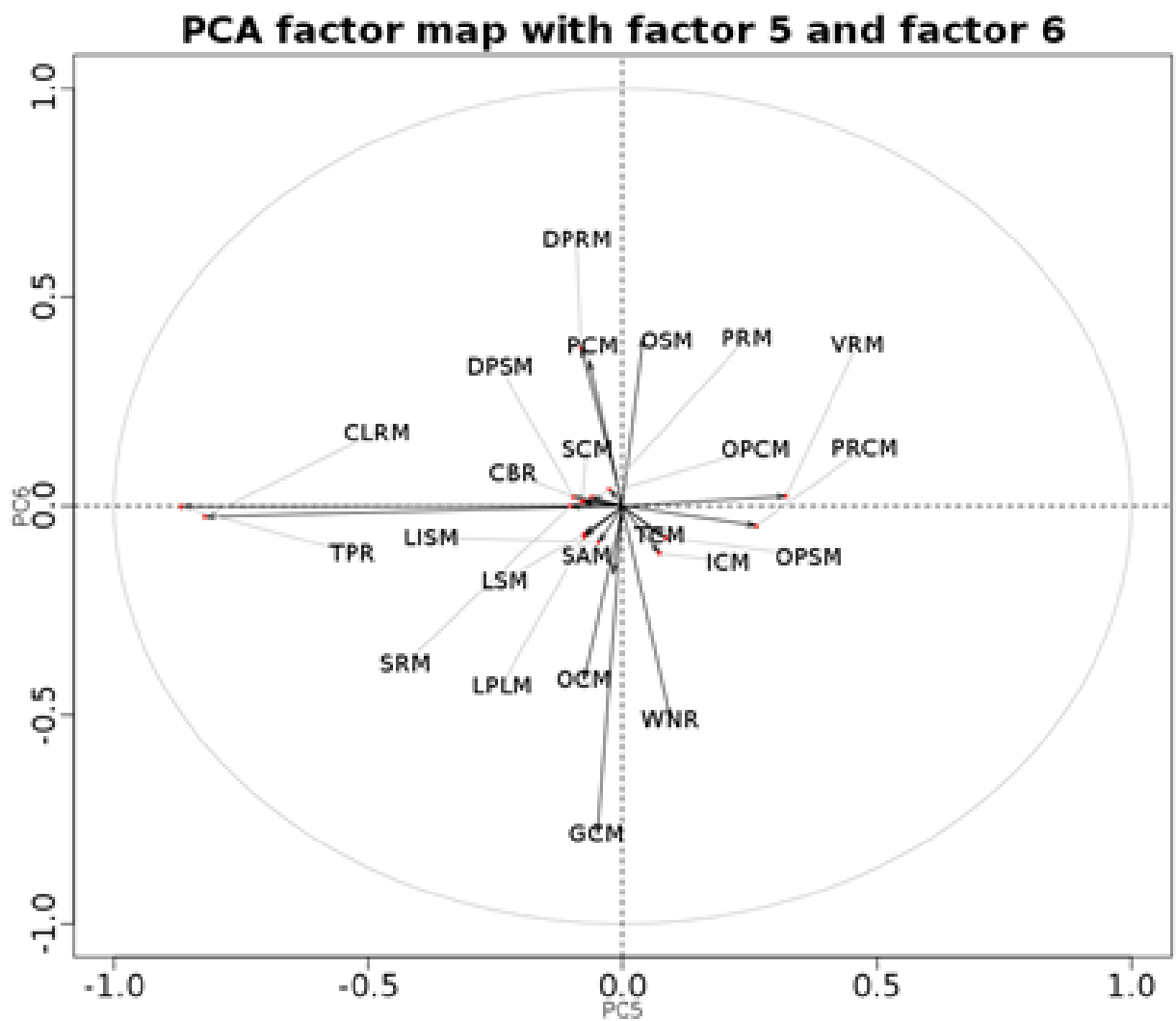


Figure 11.8: PCA factor map with factor 5 and factor 6)

Factors	Eigenvalue	% Variance	% Cumulative Variance
1	6,412	17,812	17,812
2	3,984	11,066	28,878
3	3,266	9,072	37,950
4	2,520	7,000	44,950
5	1,910	5,307	50,257
6	1,716	4,768	55,025
7	1,403	3,898	58,922
8	1,319	3,665	62,587
9	1,255	3,486	66,074
10	1,206	3,350	69,423
11	1,135	3,153	72,577
12	1,056	2,933	75,510
13	0,996	2,767	78,277
14	0,928	2,577	80,854

Table 11.6: Importance of principal components (first 15 factors)

and PCM) are borrowed from existing literature, while the others have been defined in the context of this work. They are representative of the first PC but are not the most representative metrics for the PC. This means that the newly metrics in this chapter are in most cases slightly more representative and capable to describe also PC5 and PC6.

The identified Principal Components can be interpreted as follows.

PC1 represents the volume in terms of detailed properties and literals. Metrics describing this component are counting of instance properties (OPCM, OPSM, DPSM), languages (LSM, LPLM), literals (LISM, LPLM), instances (ICM), etc., therefore they are capable to describe the RDF dataset volume and weight from different points of views for instances. Triple stores characterized by this PC1 may be implemented as real knowledge base, with huge number of instances in any aspects.

PC2 represents the internal cohesion among concepts. Metrics describing this component are assessing the volume of relationships among classes (SOM, SRM, SAM, BCM), the cohesion and coupling (CBR). Triple stores with relevant measures of internal cohesion may be dictionaries, concept engine for reasoning, etc. The quantification of this key feature passes through the analysis of predicates such as the owl:sameAs but also through the anal-

Metrics	PC1	PC2	PC3	PC4	PC5	PC6
LSM	-0,352	-0,134	0,092	0,027	-0,055	-0,051
LPLM	-0,351	-0,137	0,088	0,036	-0,054	-0,056
LISM	-0,350	-0,128	0,068	0,033	-0,035	-0,066
OPSM	-0,333	-0,095	-0,027	0,027	0,062	-0,058
ICM	-0,332	0,026	-0,028	0,068	0,052	-0,087
DPSM	-0,312	-0,020	0,134	-0,049	-0,070	0,017
OPCM	-0,282	-0,095	0,109	0,007	-0,044	0,016
CBR	-0,056	0,445	0,179	0,088	-0,057	0,009
SCM	-0,055	0,443	0,175	0,089	-0,058	0,007
SRM	-0,073	0,430	0,189	0,096	-0,075	0,000
SAM	-0,029	0,363	0,060	0,031	-0,014	-0,124
PRM	-0,116	0,098	-0,437	-0,029	-0,019	0,031
TCM	-0,129	0,101	-0,407	-0,011	0,053	-0,052
PRCM	-0,088	0,106	-0,394	-0,023	0,191	-0,035
VRM	-0,157	0,143	-0,309	-0,032	0,233	0,018
OCM	0,023	0,046	0,049	-0,530	-0,054	-0,317
DPRM	-0,025	0,042	0,041	-0,511	-0,059	0,289
PCM	-0,071	0,023	0,048	-0,445	-0,048	0,266
OSM	-0,163	0,187	0,021	-0,322	0,029	0,308
TPR	0,010	0,029	-0,229	0,031	-0,629	-0,001
CLRM	-0,001	0,052	-0,283	0,040	-0,594	-0,019
GCM	0,032	0,024	0,032	-0,302	-0,036	-0,598
WNR	-0,009	0,054	0,022	-0,062	0,068	-0,389
VME	-0,101	0,015	-0,044	0,047	-0,037	0,164
BCM	-0,024	0,271	0,032	0,071	0,026	-0,044
CCM	-0,001	0,082	-0,081	-0,020	0,098	-0,107
SEM	-0,019	0,076	0,026	0,032	0,042	-0,064
LCM	0,016	-0,012	0,017	-0,039	-0,030	-0,135
VSM	-0,135	0,155	-0,201	-0,008	0,183	0,039
VCM	0,117	-0,004	0,037	0,044	0,091	0,100
PCCM	0,011	-0,011	0,026	-0,015	0,021	0,080
DPCM	-0,193	0,003	0,067	0,010	0,029	0,064
IPCM	-0,136	-0,042	0,089	0,062	-0,114	-0,076
OPRM	-0,049	0,009	-0,115	0,010	0,161	-0,055
ORM	-0,004	0,056	0,083	0,037	-0,035	0,074
LTR	-0,149	-0,066	0,113	0,040	-0,116	-0,023

Table 11.7: Principal component loadings for the first six factors

ysis of predicates that are peculiar of the data model loaded in the RDF dataset and that keep connected concepts defined within it, characterizing resources in terms of other resources. Predicates `rdfs:subClassOf` and `skos:broader` are also useful for quantifying the internal cohesion of a RDF dataset, since they model interconnections among concepts.

PC3 represents the volume and density of triples. Metrics describing this aspect are those counting triples (TCM), and the triples per properties, per vocabulary, etc. (e.g., PRM, PRCM, VRM).

PC4 represents the volume/weight of semantic model in the triple store. Metrics describing this aspect are those assessing the size of distinct class relationships (PCM, OCM), the number of triples for the data model (OSM), data properties (DPRM). All these metrics are representing aspects of volume, size about the semantic model.

PC5 represents the density of triples per objects and subjects in the storage, a sort of cardinality. Metrics describing the triple density for object or subject (TPR, CLRM). The assessment of these metrics can describe how much the storage aimed at providing instances with respect to the wideness of the model definition. A RDF dataset is said to have a high cardinality if it often occurs that the same property is instantiated several times for a same subject. Intercepting such an aspect of a RDF dataset is of huge interest for performing some sort of user-oriented characterization, such as for identifying those RDF datasets that act as directories, listings, indexes, and similar.

PC6 represents the structure of the triple store in terms of graphs. Metrics describing this aspects are counting the number graphs (GCM), or assessing the complexity of the semantic model included in the RDF dataset in terms of inheritance relationships mainly. PC6 can be regarded as a measure of how much extensive is a triple store in term of graphs in which the RDF dataset is partitioned, the count of the instantiated classes and properties, the count of the different languages that can be found within it, and similar.

11.8 Classification of RDF datasets

According to the above metrics, we tried to find similarity patterns among RDF datasets, and thus to perform a classification among them. To this end, a clustering approach has been adopted on the collected sets of metrics presented above.

As a first step the statistical tendency on the metrics values has been verified using the Hopkins Statistic [69] that measures the probability that a given dataset is generated by a uniform data distribution. In this case, the Hopkins Statistic has been obtained equal to 0.0435 and thus, the hypothesis that the data is generated by a uniform distribution has been rejected. Therefore, the dataset could be is clusterable.

As a second step, three clustering techniques have been considered and compared in terms of validation measures. The first algorithm was the K-means (KM) clustering that minimizes the within-class sum of squares for a fixed number of clusters [122]. The second technique was the partitioning around medoids (PAM) [111], which is a more robust version of the K-means technique also called k-medoids. The third clustering technique was the Model-based Expectation-Maximization (EM) algorithm [95]. The internal measures (Silhouette Width, Connectivity and Dunn Index) have been used as validation measures to compare the considered clustering techniques. The Silhouette Width [70] is the average of the degrees of confidence of each observation (Silhouette values) in a clustering assignment. The Silhouette Width lies in the interval $[-1,1]$: well-clustered observations have values near 1 while poorly clustered observations have values near -1. The Connectivity [124] indicates the degree of connection of the clusters. The Dunn Index [86] is the ratio between the smallest distance between observations not in the same cluster and the largest intra-cluster distance, and it should be maximized. In terms of clustering techniques validity, the PAM clustering has resulted the best clustering technique according to all three internal validity measures (Fig. 11.9): for the Dunn Index PAM with 7 cluster centres is the best approach (PAM-7), for the Connectivity measure PAM with 5 cluster centres (PAM-5) is the best approach and for the Silhouette Width measure PAM with 8 cluster centres (PAM-8) is the best approach.

In addition to the three solutions of PAM-5, PAM-7 and PAM-8, K-means clustering approaches with 5 and 9 cluster centres (KM-5 and KM-9) have been considered as relevant in according to Connectivity and Silhouette measures.

Therefore, five clusterings have been performed, using PAM-5, PAM-7, PAM-8, KM-5, and KM-9. Then, a consolidation (cross-clustering) has been performed (see Figure 11.10), that has produced a set of cross-clusters such that two RDF datasets locate together in a cross-cluster if and only if the two RDF datasets locate together whatever well-known clustering algorithm

Validation Measures:							
# of Clusters		5	6	7	8	9	10
PAM	Connectivity	13,116	18,166	23,530	25,395	27,890	31,004
	Dunn	0,160	0,218	0,228	0,227	0,127	0,126
	Silhouette	0,789	0,8041	0,813	0,823	0,802	0,793
KM	Connectivity	16,169	18,432	20,432	22,332	26,515	28,979
	Dunn	0,200	0,2265	0,226	0,2265	0,144	0,168
	Silhouette	0,757	0,753	0,749	0,763	0,776	0,712
EM	Connectivity	241,679	248,122	261,420	267,239	111,356	128,563
	Dunn	0,000	0,000	0,000	0,000	0,000	0,000
	Silhouette	-0.177	-0.327	-0.318	-0.323	0.092	0.102

Figure 11.9: Internal Validity Measures Results (in table are reported only the results for 5 to 10 cluster centres).

you use of the above-mentioned.

The cross-clustering has led to the identification of five clusters named as: thesauruses, lists, gazetteers, semantic networks, and classification schemas, and described as follows.

Labels are borrowed from the structural categorization of Knowledge Organization Systems (KOS) elaborated by the Dublin Core Metadata Initiative (DCMI) and the Networked Knowledge Organization Systems, Services, and Structures (NKOS) Task Group, and resulted in the NKOS KOS Types Vocabulary [8].

Thesauruses characterize for: (i) their size, since they are relatively small-sized with respect to both the number of triples (TCM) and the number of resources (ICM), that both are good representatives respectively of PC3 and PC1; (ii) their peculiar cohesion, since a relatively conspicuous number of see also interconnections can be found (SAM), while a relatively small quantity of triples can be found where an external vocabulary is used that contrasts in turn with the wide variety of the Linked Open Vocabularies (VRM), that both are good representatives respectively of PC2 and PC3; (iii) their relatively heavy-weight data models (ORM). Most of RDF datasets in this cluster are: (i) mappings of genes to diseases, such as the ones of the BIO2RDF project; (ii) lexical mappings such as language translations, word abbreviations, and similar, see the Dbmary project for an example; (iii) bindings of people to organizations; (iv) polls, such as the votations at the Italian Camera; (v) is-a databases entirely devoted to mapping concepts to their generalizations, such as that maintained in the context of the WebIsALOD

```

Begin:
  CM = {PAM5, PAM7, PAM9, KM5, KM9, ...}
  CS = {Ccm}, cm ∈ CM; clusters sets
  Ccm = {Ccm,i}, i ∈ [1, Ncm]; Ncm = # of cm clusters
  I ← Ccm
  for each cm' ∈ (CM \ cm)
    I' ← ∅
    for each Ccm',i ∈ Ccm'
      for each Ij ∈ I
        I' ← I' ∪ (Ij ∩ Ccm',i)
      next;
    next;
    I ← I'
  next;
  return I
end;

```

Figure 11.10: The cross-clustering result consolidation algorithm

project.

Lists characterize for: (i) the usage of a relatively small number of predicates to represent a huge amount of literal data (DPRM, PC4); (ii) a relatively huge number of resources for each class, as it can be seen from values of the IPCM metric, a (fairly weak) representative of PC1, computed for RDF datasets in this cluster. Most of RDF datasets in this cluster are: (i) library catalogues, such as the one of the National Central Library of Florence; (ii) semantic-empowered thematic collections of both digital and papery assets spread all over the World, such as the dataset built in the context of the Munnin project; (iii) wide (both geographically and as for the covered topics) indexes of digital datasets such as that exposed at <https://semanticweb.cs.vu.nl/>; (iv) digital encyclopaedias, such as those produced in the context of the Dbpedia and the DbkWik projects; (v) country-wide government open data such as those made available by the Czech Republic (NKOD).

Gazetteers characterize for: (i) their peculiar cohesion, since a relatively huge exploitation of relations such as broader (BCM), subclass (SCM), and same as (SAM) can be observed (metrics all explain the PC2), and also a wide usage (VSM, PC3) of a wide set (VCM, PC1) of Linked Open Vocabularies can be found; (ii) the relatively huge localization of literals, intended as the presence of huge amounts of literals that bear an explicit indication of the language in which they are written (LSM, LTR, the former of which explains better than any other metric the PC1). Most of RDF datasets in this cluster contain descriptions of territories, boundaries, namings, geographic taxonomies, geolocated items, and similar. Examples are the sample dataset housed by the Faculty of Computer Science and Engineering of Skopje at <http://linkeddata.finki.ukim.mk/sparql>, and the dataset exposed at <http://data.lenka.no/sparql>.

Semantic networks characterize for: (i) the huge number of graphs (GCM, PC6); (ii) the small number of resources (ICM, PC1); (iii) the weak cohesion as for the same as (SAM), see also (SEM), and subclass (SCM) relations (metrics all explain the PC2); (iv) the light-weight data model (ORM, PC3); (v) the small number of literals that bear an explicit indication of the language in which they are written (LSM, LTR, that both explain the PC1); (vi) the small set of predicates (PCM, PC4). Most of RDF datasets in this cluster are part of the wide network of semantic repositories that address the life sciences and are built and maintained in the context of the Bio2RDF [60]

open-source project.

Classification schemas characterize for the heterogeneity of referenced resources as for their classes (OCM, PC4), that contrasts with the relatively very small number of triples filled by a resource (OPSM, PC1). Most of RDF datasets in this cluster are managed by educational or government institutions and contain (i) literature, (ii) reports, (iii) models, (iv) historical reviews, (v) taxonomies of subjects of study, (vi) inventories, mainly related to (i) arts, and (ii) humanities. Examples are the humanities and social sciences DB at <https://isidore.science/>, the dataset maintained in the context of the Linked Life Data project, the art-related and culture-related data housed at the Getty Vocabularies, the RDF dataset of the Private Technical University of Loja, and other.

11.9 Conclusions

Moving from metrics already proposed in the literature that address semantic datasets and Object Oriented Programming (as for the assessment of cost and complexity of data models), we have defined a metric framework including a multi-layered set of indexes, some of which relate to single RDF datasets while others relate to aggregates of RDF datasets. The assessment for the stores is based on complex indexes computed efficiently exploiting foundational indexes and foundational indexes computed with an optimistic approach: a (multi-versioned) one-shot query in first, and more elaborated techniques for those situations where the easy solution is found not to be applicable. Newly defined metrics measure have been defined assessing volumes, densities, relations among resources and among concepts within and across the boundaries of RDF datasets, with the aim of quantifying the quality and cost of semantic datasets, contributing to the description and indexing of data within them, and performing structural characterizations, so partially overlapping existing proposals, and extending to areas of investigation left uncovered. The preliminary schematization of what should be measured, how it could be measured, and why it should be measured, that you can find proposed in a number of useful tables, have played a key role in the identification of areas of investigation to be covered, and methodologies. Through the Principal Component Analysis, we have identified the most meaningful aspects of semantic datasets (the weight, the internal and external cohesion, the verbosity, the cardinality, and the extent), and met-

rics that best suit for their classification. Through the application of a motivated selection of well-known clustering algorithms, and through an innovative technique of consolidation of clustering results that originate from the application of different algorithms (the cross-clustering), we have partitioned a large range of worldwide monitored RDF datasets into five groups. Looking into those groups, we have found that they correspond to five of the structural categories defined in the NKOS KOS Types Vocabulary (thesauruses, lists, gazetteers, semantic networks, and classification schemas). A Web portal has been developed where metrics, RDF datasets and measurements can be consulted.

Chapter 12

Conclusion

This chapter summarizes the contribution of the thesis and discusses avenues for future research.

In Chapter 3, a unified model addressing information for mobility and transport applications for smart city has been proposed. The chapter presented the identified requirements and the state of the art of the possible solution. In order to satisfy the requirements a new model has been developed by starting from Km4City model. In the chapter, the model and the process for the construction of the knowledge base have been described. The process of Km4City KB construction has taken into account: Open Street Map, open data, and real-time data from sensors, mobility and transport operator in multiple formats and protocols, to load them into a graph database. The solution proposed has been shown at work by presenting the Km4City Traffic Reconstruction solution based on differential equations. The solution developed and related algorithms are part of Km4City (www.km4city.org), and made accessible for test and usage from <http://servicemap.km4city.org> and from <http://firenzetraffic.km4city.org> applied to the whole Tuscany, one of the largest regions in Italy. Main numbers and considerations related to the area of Tuscany and other trials have been also reported.

Smart City solutions initially started with the production of open data and their collection on managing open data tools. Then smart cities evolved towards data aggregation and semantics modeling of city entities, benchmarks, and datasets. Recently, some of them are also offering IOT support. The combination of IOT and smart city is not an easy task, the data volumes are much higher than those addressed for industrial IOT. The complexity of

IOT smart city solutions have been identified by a number of actors. The European commission started setting up the EIP project for stimulating and concerting actions. On the other hand, the Select4Cities project of the European Commission and associated community <http://www.select4cities.eu/> created a challenge to find research-based solutions satisfying a formalized set of functional and non-functional requirements. Snap4City solution that has been presented in Chapters 4 5 is one of the solutions approved and developed in response to that challenge. Snap4City solution is based on MicroServices directly provided by a number of applications and Smart City API. Snap4City users can develop in few steps sophisticated IOT applications exploiting MicroServices that can control city dashboards as well as IOT mobile applications. Moreover, a number of development and monitoring tools have been designed and developed. Among them, a special attention is given to the tools and solutions for monitoring communication performance among IOT, storage, smart city API, external services, etc. Another addressed aspect consisted on the test to perform the assessment of scalability feature of the IOT smart city infrastructure in terms of maximum number of messages exchanged per minute fixed the number IOT processes and corresponding resources. Future work will be devoted to put together these aspects to develop algorithms for automated scaling and IOT cloud management.

In Chapter 6 a solution has been presented for federating Smart City API. The solution aims to solve the requirements that presently cannot be solved by traditional GIS solutions. Thus we have described as in our proposed approach, we have avoided migrating data, providing federation at level of APIs, involving nodes of any size, combining them autonomously so that each of them may decide to join or not, leaving the possibility of having different kind of services, enabling the movements from among federate areas, preventing the access and respect GDPR and data security, combining services, etc. For the design, implementation and validation of the solution we have used and enhanced the former Km4City API and ontology [117], which has been implemented as the so called ServiceMap. For this reason, the federation of Smart City API has been provided in the so-called Super-ServiceMap. The validation has been performed by considering 4 large areas and smart city services in place covering: Tuscany region of 3.5 M inhabitant in the center of Italy, north of Italy and Sardegna island, Antwerp and north of Belgium, and Helsinki and south of Finland. The validation has shown

that the solution performs better than single centralized services in many cases, except for the cases in which simple direct queries are performed. This means that the SuperServiceMap may implement a number of more sophisticated strategies to improve the service performance in specific cases. In fact, the solution has also put in place a rule-based module to change the strategies according to the services, area, and query shape.

The strong push towards Industry 4.0 has constrained many industries to work in integrated and interoperable supply chains. In most cases, it implied to open at the integration of their production processes with those of other industries in the supply chain, and thus with their customers and suppliers. This fact led to be capable to give access at data and flows and to create integrated dashboards for their customers and to perform some synchronization and supervision, and may to create integrated control rooms, synoptics and dashboards, and allow to collect data for big data analytic, business intelligence, machine learning on production processes. This activity has been in most cases facilitated by the introduction of IoT solutions with IoT Devices, IoT Brokers, etc., which provide a completely different approach with respect to DCS and SCADA solutions usually adopted in the industry for controlling their local productions. In Chapter 7, Snap4Industry with its IoT development environment and framework for implementing the Control and Supervision of Multiple Supply Chains in the view of Industry 4.0 as a service have been presented. In particular, the motivations, requirements and the actions performed to extend IoT Snap4City 100% open source platform to comply with Industry 4.0 requirements have been described. The main additions for creating Snap4Industry solution have been on: (i) a number of industry protocols, (ii) tools for producing custom widgets and synoptics and connecting them respecting security into a Dashboard system, (iii) a number of new MicroServices for Node-RED to enable the usage of synoptics as event driven devices in/out, (iv) the usage of WebSocket secure for the communication with custom Synoptics and Widget for dashboards, (v) the automation of process for producing synoptics templates and instances according to GDPR. The research has been founded into SODA R&D project and tested on chemical plant of ALTAIR, one of the main chemical industry in Italy, and into 5G national project in GIDA plant for water depuration. The specific case of ALTAIR has not been reported for security reason and since it is covered by industry secret.

In Chapters 8 and 9, we have proposed a real-time visual self-adaptive

solution for reconstructing the traffic density at every location within a wide urban area leveraging the detections from a few fixed traffic sensors deployed within the area of interest. Our solution present several advantages with respect to the solutions available in the literature providing at the same time: (i) wide temporal and/or spatial horizon (not limited to a few roads elements); (ii) supporting complex road structure; (iii) wide applicability; (iv) no needs of third-party engagement on providing data; (v) robust with respect to discontinuous data; (vi) declared clear precision rate; (vii) real-time visual rendering of results (<http://firenzetraffic.km4city.org>). We have addressed such issues though a traffic flow modelling based on the solid mathematics of the fluid dynamics, also relying on the constantly up-to-date road network, road features, traffic restrictions, and traffic sensor detections stored in the Km4City KB, and displaying the reconstruction at real-time through colored lines traced on a street map. We have tested the proposed solution for reconstructing the traffic in the city of Florence, and we have found it to reach a 75% precision also on boundaries and much higher precision inside the observed area.

The analysis performed in Chapter 10 identified a strong reduction of the mobility and transport activities coherently to what has been described by Google Global mobility report and in addition putting in evidence a number of detailed aspects that allowed us to better judge the impact of the lockdown and potential effects of the reopening on an eventual revamping of the infection computing and observing the R_t and its trend. To this aim, the collected data from the field have been compared from those of Google and some considerations with respect to the Imperial college Report 20 have been derived. It is probably too early to derive conclusions since the report has been closed the 10th of May, only 6 days after the formal reopening of the 4th. At that time, the resulting traffic volume had been repristinated to more than the 53%, which is a +33% with respect to the lockdown situations. This indicator and similar ones will be monitored and highlighted in view of the future return of tourists in Florence UNESCO site (as pilot of Herit-Data project). For parking, we still had to see an increment of their usage with respect to the lockdown conditions. As regard the environmental variable and thus or air quality the amount of pollutant has been strongly reduced by the lockdown. The largest reductions have been recorded for NO₂ (also due to the reduction of traffic), CO, PM₁₀ and CO₂ for the reduction of human activities.

In Chapter 11, moving from metrics already proposed in the literature that address semantic datasets and Object Oriented Programming (as for the assessment of cost and complexity of data models), we have defined a metric framework including a multi-layered set of indexes, some of which relate to single RDF datasets while others relate to aggregates of RDF datasets. The assessment for the stores is based on complex indexes computed efficiently exploiting foundational indexes computed with an optimistic approach: a (multi-versioned) one-shot query in first, and more elaborated techniques for those situations where the easy solution is found not to be applicable. Newly defined metrics measures have been defined assessing volumes, densities, relations among resources and among concepts within and across the boundaries of RDF datasets, with the aim of quantifying the quality and cost of semantic datasets, contributing to the description and indexing of data within them, and performing structural characterizations, so partially overlapping existing proposals, and extending to areas of investigation left uncovered. The preliminary schematization of what should be measured, how it could be measured, and why it should be measured, that you can find proposed in a number of useful tables, have played a key role in the identification of areas of investigation to be covered, and methodologies. Through the Principal Component Analysis, we have identified the most meaningful aspects of semantic datasets (the weight, the internal and external cohesion, the verbosity, the cardinality, and the extent), and metrics that best suit for their classification. Through the application of a motivated selection of well-known clustering algorithms, and through an innovative technique of consolidation of clustering results that originate from the application of different algorithms (the cross-clustering), we have partitioned a large range of worldwide monitored RDF datasets into five groups. Looking into those groups, we have found that they correspond to five of the structural categories defined in the NKOS KOS Types Vocabulary (thesauruses, lists, gazetteers, semantic networks, and classification schemas). A Web portal has been developed where metrics, RDF datasets and measurements can be consulted.

Appendix A

Appendix

In this appendix the list of the acronyms used in Chapter 11 can be found.

A.1 Acronyms of RDF Dataset Metrics

In Tables A.1 A.2, the acronyms used in Chapter 11 are reported with their corresponding extended name. They also refer to metrics or other concept exploited.

Acronym	Extended Name
AEM	Alive Endpoint Metric
AER	Alive Endpoints Ratio
AES	Alive Endpoints
BCM	Broader Count Metric
CBO	Coupling Between Objects
CBR	Coupling Between Resources
CCM	Class Count Metric
CLK	Class Ranking
CLR	Class Ratio
CLRM	Class Ratio Metric
DCMI	Dublin Core Metadata Initiative
DIT	Depth of Inheritance
DPCM	Data Property Count Metric
DPRM	Data Property Ratio Metric
DPSM	Data Property Sum Metric
E11	SPARQL 1.1 Endpoints
EM	Expectation Maximization
GCM	Graph Count Metric
HLM	High Level Metric
ICLM	Instantiated Class Metric
ICM	Instance Count Metric
IDP	Instantiated Data Properties By Class
IOP	Instantiated Object Properties By Class
IPCM	Instance per Class Metric
IPM	Instantiated Property Metric
IPO	Instantiated Object Properties By Object
KOS	Knowledge Organization Systems
LAK	Language Ranking
LAM	Language Metric
LAR	Language Ratio
LCM	Languages Count Metric
LIM	Links Metric
LISM	Literals Sum Metric
LNK	Linked Namespaces Ranking
LNR	Linked Namespaces Ratio
LOV	Linked Open Vocabulary
LPLM	Literal per Language Metric
LSM	Languages Sum Metric
LTR	Localized Triples Ratio

Table A.1: Acronyms (continue)

Acronym	Extended Name
NOC	Number of Children
NSUB	Number of Subclasses
NSUP	Number of Superclasses
OCM	Object Classes Metric
OME	Ontology Metric
ONR	Ontology Ratio
OPCM	Object Property Count Metric
OPRM	Object Property Ratio Metric
OPSM	Object Property Sum Metric
ORM	Ontology Ratio Metric
OSM	Ontology Size Metric
PAM	Partition around Medoids
PCCM	Property Count by Class Metric
PCM	Property Count Metric
PMC	Predictive Model Complexity
PRCM	Property Ratio by Class Metric
PRK	Property Ranking
PRM	Property Ratio Metric
PRR	Property Ratio Ratio
R11	SPARQL 1.1 Endpoints Ratio
RSM	Resource Size Metric
SAM	“Same As” Metric
SCM	Subclass Count Metric
SEM	“See Also” Metric
SRM	Subclass Ratio Metric
TCM	Triples Count Metric
TPR	Triples Per Resource
VCM	Vocabulary Count Metric
VME	Version Metric
VOK	Vocabulary Ranking
VOM	Vocabulary Metric
VOR	Vocabulary Ratio
VRM	Vocabulary Ratio Metric
VSM	Vocabulary Sum Metric
VTK	Vocabulary Terms Ranking
VTR	Vocabulary Terms Ratio
WNR	Weighted Number of Relatives

Table A.2: Acronyms (continued)

Appendix B

Publications

This research activity has led to several publications in international journals and conferences. These are summarized below.¹

Journals

1. C. Badii, P. Bellini, S. Bilotta, D. Bologna, D. Cenni, A. Difino, A. Ipsaro Palesi, N. Mitolo, P. Nesi, G. Pantaleo, I. Paoli, M. Paolucci, **M. Soderi**. “How COVID-19 Lockdown Impacted on Mobility and Environmental data”, *Bollettino della Società Geografica Italiana*, FuPress, 2020.

Submitted

1. Pierfrancesco Bellini, Paolo Nesi, Irene Paoli, **Mirco Soderi**. “Classification of RDF datasets through Multilayer Metrics”, *Journal of Web Semantics*, 2019.

International Conferences and Workshops

1. P. Bellini, S. Bilotta, P. Nesi, M. Paolucci, **M. Soderi**. “Traffic Flow Reconstruction from Scattered Data”, in *IEEE SMARTCOMP, IEEE international conference on smart computing*, Taormina (Italy), 2018.
2. P. Bellini, S. Bilotta, P. Nesi, M. Paolucci, **M. Soderi**. “Real-Time Traffic Estimation of Unmonitored Roads”, in *IEEE-DataCom’2018*, Athen (Greece), 2018.

¹The author’s bibliometric indices are the following: H -index = 3, total number of citations = 21 (source: Google Scholar on Month October, 2020).

3. C. Badii, E. G. Belay, P. Bellini, D. Cenni, M. Marazzini, M. Mesiti, P. Nesi, G. Pantaleo, M. Paolucci, S. Valtolina, **M. Soderi**, I. Zaza. “Snap4City: A Scalable IOT/IOE Platform for Developing Smart City Applications”, in *Int. Conf. IEEE Smart City Innovation*, Guangzhou (China), 2018.
4. P. Bellini, S. Bilotta, D. Cenni, P. Nesi, M. Paolucci, **M. Soderi**. “Knowledge Modeling and Management for Mobility and Transport Applications”, in *IEEE TeC4C’18, 1st International Workshop on Technology Convergence for Smart Cities*, Philadelphia (Pennsylvania, USA), 2018.
5. P. Bellini, P. Nesi, M. Paolucci, **M. Soderi**, P. Zamperlin. “Snap4city Platform: Semantic to Improve Location Based Service”, in *15th Conf. on Location Based Services, LBS2019, Advances in Cartography and GIScience of the ICA*, Vienna (Austria), 2019.
6. P. Bellini, D. Nesi, P. Nesi, **M. Soderi**. “Federation of Smart City Services via APIs”, in *Proc of 6th IEEE International Workshop on Sensors and Smart Cities, with IEEE SmartComp*, Bologna (Italy), 2020.
7. C. Badii, P. Bellini, D. Cenni, N. Mitolo, P. Nesi, G. Pantaleo, **M. Soderi**. “Industry 4.0 Synoptics Controlled by IoT Applications in Node-RED”, in *Proc. of the IEEE iThings*, Virtual conference due to COVID-19 pandemic, 2020.

Bibliography

- [1] “Arcgis opendata,” <http://opendata.arcgis.com/>.
- [2] “Berners-lee, tim. linked data - design issues. retrieved: July 2006.” <http://www.w3.org/DesignIssues/LinkedData.html>.
- [3] “Board, dcmi usage. dcmi metadata terms. (2008).”
- [4] “Brickley, d., and l. miller. friend of a friend vocabulary specification. (2001).”
- [5] “C. h. chapman and o. b. downs, assessing road traffic flow conditions using data obtained from mobile data sources. u.s. patent no. 7,831,380. 9 nov. 2010.”
- [6] “Citypulse consortium. final report. deliverable, citypulse consortium, 2016.”
- [7] “Ckan,” <http://ckan.org>.
- [8] “Dublin core metadata initiative. nkos task group. kos types vocabulary. oct. 2015.” <http://nkos.slis.kent.edu/nkostype/nkostype.rdf>.
- [9] “E. prud’hommeaux and a. seaborne. sparql query language for rdf. w3c recommendation 15 january 2008,” <http://www.w3.org/TR/rdf-sparql-query/>.
- [10] “Google llc. google covid-19 community mobility reports.” <https://www.google.com/covid19/mobility/>.
- [11] “Green button connect,” <http://www.greenbuttonconnect.com/>.
- [12] “Hartig, olaf, and jun zhao. provenance vocabulary core ontology specification.” <http://trdf.sourceforge.net/provenance/ns.html>.
- [13] “Iot analytics. list of 640+ enterprise iot projects,” <https://iot-analytics.com/product/list-of-640-iot-projects/>.
- [14] “Lorenz, b., ohlbach, h. j., and yang, l. (2005). ontology of transportation networks.”
- [15] “Mcguinness, deborah l., and frank van harmelen. owl web ontology language overview. w3c recommendation 10.10 (2004): 2004.”

- [16] “Message queue telemetry transport (mqtt), oasis std., 2014.”
- [17] “Miles, alistair, and sean bechhofer. skos simple knowledge organization system reference. w3c recommendation 18 (2009): W3c.”
- [18] “Nodered,” <https://nodered.org/>.
- [19] “Ontology of transportation networks, deliverable a1-d4, project rewerse, 2005,” <http://rewerse.net/deliverables/m18/a1-d4.pdf>.
- [20] “Opendatasoft,” <https://www.opendatasoft.com/>.
- [21] “Report 20: Using mobility to estimate the transmission intensity of covid-19 in italy: A subnational analysis with future scenarios.”
- [22] “S. harris and a. seaborne. sparql 1.1 query language. w3c working draft. july 2012.” <http://www.w3.org/TR/sparql11-query/>.
- [23] “Specification for urban platforms, eip project, version 2.2, 2016, european innovation partnership for smart cities and communities.”
- [24] T. Abid, H. Zarzour, M. R. Laouar, and M. T. Khadir, “Smart cities based on web semantic technologies.” in *Ubicomp/ISWC 2016 Adjunct. Heidelberg*, 2016.
- [25] M. Acuto, K. Steenmans, E. Iwaszuk, and L. Ortega Garza, “Informing urban governance? boundary spanning organisations and the ecosystem of urban data.” *Area*, vol. 51, 2019.
- [26] A. Aktay *et al.*, “Google covid-19 community mobility reports: Anonymization process description (version 1.0),” *ArXiv abs/2004.0*, 2020.
- [27] R. Albertoni, M. D. Martino, and P. Podesta, “A linkset quality metric measuring multilingual gain in skos thesauri.” in *LDQ@ ESWC 2015*, 2015.
- [28] S. Auer *et al.*, “Lodstats-an extensible framework for high-performance dataset analytics.” in *International Conference on Knowledge Engineering and Knowledge Management*, 2012.
- [29] M. Azzari, C. Garau, P. Nesi, M. Paolucci, and P. Zamperlin, “Smart city governance strategies to better move towards a smart urbanism,” in *The 18th International Conference on Computational Science and Its Applications (ICCSA 2018)*, 2018.
- [30] E. B and H. T, *An introduction to applied multivariate analysis with R*. Springer Science and Business Media, 2011.
- [31] C. Badii, E. Belay, P. Bellini, D. Cenni, M. Marazzini, M. Mesiti, P. Nesi, G. Pantaleo, M. Paolucci, S. Valtolina *et al.*, “Snap4city: A scalable iot/ioe platform for developing smart city applications,” in *Proceedings of the 2018 IEEE Smart World*, 2018.

- [32] C. Badii, P. Bellini, S. Bilotta, D. Bologna, D. Cenni, A. Difino, A. I. Palesi, N. Mitolo, P. Nesi, G. Pantaleo, I. Paoli, M. Paolucci, and M. Soderi, "How covid-19 lockdown impacted on mobility and environmental data," *Bollettino della Società Geografica Italiana, FuPress*, 2020.
- [33] C. Badii, P. Bellini, D. Cenni, A. DiFino, P. Nesi, and M. Paolucci, "Analysis and assessment of a knowledge based smart city architecture providing service apis," *Future Generation Computer Systems*, 2017.
- [34] C. Badii, P. Bellini, D. Cenni, A. Difino, P. Nesi, and M. Paolucci, "User engagement engine for smart city strategies," in *IEEE International Conference on Smart Computing*, 2017.
- [35] C. Badii, P. Bellini, D. Cenni, N. Mitolo, P. Nesi, G. Pantaleo, and M. Soderi, "Industry 4.0 synoptics controlled by iot applications in node-red," in *Proc. of the IEEE iThings, November 02-06, 2020*, 2020.
- [36] C. Badii, P. Bellini, A. Difino, and P. Nesi, "Sii-mobility: An iot/ioe architecture to enhance smart city mobility and transportation services," *Sensors*, 2019.
- [37] —, "Smart city iot platform respecting gdpr privacy and security aspects," *IEEE Access*, 2020.
- [38] C. Badii, P. Bellini, A. Difino, P. Nesi, G. Pantaleo, and M. Paolucci, "Microservices suite for smart city applications," *Sensors, MDPI*, 2019.
- [39] C. Badii, P. Bellini, P. Nesi, and M. Paolucci, "A smart city development kit for designing web and mobile apps," in *IEEE Smart World, Ubiquitous Intelligence and Computing, Advanced and Trusted Computed, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBD-Com/IOP/SCI)*, 2017.
- [40] C. Badii, P. Nesi, and I. Paoli, "Predicting available parking slots on critical and regular services exploiting a range of open data," *IEEE Access*, 2018.
- [41] F. Bauer and M. Kaltenbock, *Linked open data: The essentials*. mono-/monochrom, 2011.
- [42] P. Bellini, S. Bilotta, D. Cenni, P. Nesi, M. Paolucci, and M. Soderi, "Knowledge modeling and management for mobility and transport applications," in *IEEE TeC4C'18, 1st International Workshop on Technology Convergence for Smart Cities, Philadelphia, PA, USA*, 2018.
- [43] P. Bellini, S. Bilotta, P. Nesi, M. Paolucci, and M. Soderi, "Real-time traffic estimation of unmonitored roads," in *IEEE-DataCom'2018, Athen, 2018*, 2018.

- [44] —, “Real-time traffic estimation of unmonitored roads,” in *IEEE-DataCom’2018*, 2018.
- [45] —, “Wip: Traffic flow reconstruction from scattered data,” in *IEEE International Conference on Smart Computing (SMARTCOMP)*, Taormina, Italy, 2018.
- [46] P. Bellini, D. Cenni, M. Marazzini, N. Mitolo, P. Nesi, and M. Paolucci, “Smart city control room dashboards: Big data infrastructure, from data to decision support,” *Journal of Visual Languages and Computing*, 2018.
- [47] P. Bellini, D. Nesi, P. Nesi, and M. Soderi, “Federation of smart city services via apis,” in *Proc of 6th IEEE International Workshop on Sensors and Smart Cities, with IEEE SmartComp, 14-17 Sept. 2020, Bologna, Italy.*, 2020.
- [48] P. Bellini, P. Nesi, M. Paolucci, M. Soderi, and P. Zamperlin, “Snap4city platform: Semantic to improve location based service,” in *15th Conf. on Location Based Services, LBS2019, Advances in Cartography and GIScience of the ICA*, 2019.
- [49] P. Bellini, P. Nesi, M. Paolucci, and I. Zaza, “Smart city architecture for data ingestion and analytics: Processes and solutions,” in *Proceedings IEEE 4th International Conference on Big Data Computing Service and Applications, Big Data Service*, 2018.
- [50] P. Bellini, P. Nesi, and A. Venturi, “Linked open graph: Browsing multiple sparql entry points to build your own lod views,” *Journal of Visual Languages and Computing*, 2014.
- [51] P. Bellini *et al.*, “Km4city ontology building vs data harvesting and cleaning for smart-city services.” *Journal of Visual Languages and Computing*, vol. 25.6, 2014.
- [52] F. Benedetti, S. Bergamaschi, and L. Po, “Online index extraction from linked open data sources.” in *LD4IE@ISWC*, 2014.
- [53] M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor, “Iot-lite: a lightweight semantic model for the internet of things.” in *2016 Intl IEEE Conferences on Ubiquitous Intelligence and Computing, ..., UIC/ATC/ScalCom/CBDCCom/IoP/SmartWorld*, 2016.
- [54] T. Berners-Lee, J. Hendler, and O. Lassila, “The semantic web,” *Scientific American*, vol. 284.5, 2001.
- [55] C. Bizer, T. Heath, and T. Berners-Lee, “Linked data-the story so far.” *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 5.3, 2009.
- [56] A. P. Black, “Object-oriented programming: Some history, and challenges for the next fifty years.” *Information and Computation*, vol. 231, 2013.

- [57] D. Bonino *et al.*, “Almanac: Internet of things for smart cities.” in *2015 3rd International Conference on Future Internet of Things and Cloud*, 2015.
- [58] G. Bretti, R. Natalini, and B. Piccoli, “A fluid-dynamic traffic model on road networks.” *Archives of Computational Methods in Engineering*, vol. 14, 2007.
- [59] G. Bucci *et al.*, “Metrics and tool for system assessment.” in *Fourth IEEE International Conference on Engineering of Complex Computer Systems*, 1998.
- [60] A. Callahan, J. Cruz-Toledo, P. Ansell, and M. Dumontier, “Bio2rdf release 2: Improved coverage, interoperability and provenance of life science linked data,” in *ESWC 2013*, 2013.
- [61] S. Campinas *et al.*, “The sindice-2011 dataset for entity-oriented search in the web of data.” in *1st international workshop on entity-oriented search (EOS)*, 2011.
- [62] S. R. Chidamber and C. F. Kemerer, “A metrics suite for object-oriented design.” *IEEE Transactions on software engineering*, vol. 20.6, 1994.
- [63] L. Cinquini *et al.*, “The earth system grid federation: An open infrastructure for access to distributed geospatial data.” *Future Generation Computer Systems*, vol. 36, 2014.
- [64] S. Consoli *et al.*, “A smart city data model based on semantics best practice and principles.” in *International World Wide Web Conference 2015 Companion*, 2015.
- [65] A. Cullen and H. Frey, *Probabilistic Techniques in Exposure Assessment, 1st edition*. Plenum Publishing Co., 1999.
- [66] C. Daganzo, “The cell transmission model: a dynamic representation of highway traffic consistent with the hydrodynamic theory,” *Transportation Research B*, vol. 28, 1994.
- [67] —, “The cell transmission model, part ii: Network traffic,” *Transportation Research B*, vol. 29, 1995.
- [68] J. Debattista, S. Auer, and C. Lange, “Luzzu-a methodology and framework for linked data quality assessment,” *Journal of Data and Information Quality (JDIQ)*, 2016.
- [69] J. Debattista, C. Lange, and S. Auer., “daq, an ontology for dataset quality information.” in *LDOW 2014*, 2014.
- [70] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the royal statistical society. Series B (methodological)*, 1977.

- [71] L. Ding *et al.*, “Swoogle: A semantic web search and metadata engine.” *Proc. 13th ACM Conf. on Information and Knowledge Management*, 2004.
- [72] J. Dizdarevic, F. Carpio, A. Jukan, and X. Masip-Bruin, “A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration,” *ACM Comput. Surv.*, 2019.
- [73] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap, 1st edition*. Chapman and Hall, 1994.
- [74] J. D. Fernández, M. A. Martínez-Prieto, P. de la Fuente Redondo, and C. Gutiérrez, “Characterising rdf data sets,” *Journal of Information Science*, 2018.
- [75] F. Fioravanti and P. Nesi, “Estimation and prediction metrics for adaptive maintenance effort of object-oriented systems.” *IEEE Transactions on software engineering*, vol. 27.12, 2001.
- [76] S. K. Godunov, “A finite difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics,” *Mathematics Sbornik*, vol. 47, 1959.
- [77] R. V. Guha, D. Brickley, and S. Macbeth, “Schema. org: evolution of structured data on the web.” *Communications of the ACM*, vol. 59.2, 2016.
- [78] A. Gyrard and M. Serrano, “Connected smart cities: Interoperability with seg 3.0 for the internet of things.” in *2016 30th Int. Conf. on Adv. Inf. Networking and Applications (WAINA)*, 2016.
- [79] H. Halpin *et al.*, “When owl: sameas isn’t the same: An analysis of identity in linked data.” in *International Semantic Web Conference*, 2010.
- [80] A. Helmreich, S. Abdulaziz, Z. Jan, A. Kazmi, and M. Serrano, “D3.1.2 virtual models, data and metadata for icos v2,” *Deliverable, VITAL Consortium*, 2015.
- [81] J. M. Hernández-Munoz *et al.*, “Smart cities at the forefront of the future internet.” *The future internet assembly*, 2011.
- [82] A. Hernich, C. Lutz, A. Ozaki, and F. Wolter, “Schema.org as a description logic,” in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.
- [83] K. HF, “The application of electronic computers to factor analysis,” *Educ Psychol Meas*, 1960.
- [84] S. Hu *et al.*, “Smartroad: smartphone-based crowd sensing for traffic regulator detection and identification,” *ACM Transactions on Sensor Networks (TOSN)*, vol. 11.4, 2015.

- [85] W. Huang *et al.*, “Deep architecture for traffic flow prediction: deep belief networks with multitask learning,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 15.5, 2014.
- [86] H. J. K. J. and K. DB, “Computational cluster validation in post-genomic data analysis,” *Bioinformatics*, 2005.
- [87] L. F. J., R. Y., and C. E., “A feasibility study on developing iot/m2m applications over etsi m2m architecture.” in *Proc. of Int’l Conf. on Parallel and Distributed Systems*, 2013.
- [88] S. J. *et al.*, “Toward a standardized common m2m service layer platform: Introduction to onem2m.” *IEEE Wireless Communications*, vol. 21.3, 2014.
- [89] C. J.-P., S. S., E. J., and A. K., “Xgsn: An opensource semantic sensing middleware for the web of things,” in *7th Int’l Workshop on Semantic Sensor Networks*, 2014.
- [90] J. Jin, J. Gubbi, S. Marusic, and M. Palaniswami, “An information framework for creating a smart city through internet of things,” *IEEE Internet of Things Journal*, 2014.
- [91] I. Jolliffe, “Principal component analysis,” *International encyclopedia of statistical science*, 2011.
- [92] A. K., H. M., and S. A., “A middleware for fast and flexible sensor network deployment,” in *Proc. 32nd Int’l Conf. on Very Large Data Bases*, 2006.
- [93] P. Kachroo and S. Sastry, “Travel time dynamics for intelligent transportation systems: Theory and application,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, 2016.
- [94] M. Katsumi and M. Fox, “icity ontology version 1.0 report,” University of Toronto Transportation Research Institute, Tech. Rep., 2017.
- [95] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: An introduction to cluster analysis*. Wiley, 1990.
- [96] A. Keith and M. Hausenblas, “Describing linked datasets-on the design and usage of void, the vocabulary of interlinked datasets.” in *Linked Data on the Web Workshop (LDOW 09)*, 2009.
- [97] N. Komninos, C. Bratsas, C. Kakderi, and P. Tsarchopoulos, “Smart city ontologies: Improving the effectiveness of smart city applications.” in *Journal of Smart Cities*, 2015.
- [98] A. Krylovskiy, M. Jahn, and E. Patti, “Designing a smart city internet of things platform with microservice architecture,” in *3rd Int’l Conf. on Future Internet of Things and Cloud*, 2015.

- [99] —, “Designing a smart city internet of things platform with microservice architecture.” in *3rd Int.Conf. on Future Internet of Things and Cloud*, 2015.
- [100] A. Langegger, W. WoB, and M. Blochl, “A semantic web middleware for virtual data integration on the web,” in *Proceedings of the European Semantic Web Conference 2008*, 2008.
- [101] A. Langegger and W. Woss, “Rdfstats-an extensible rdf statistics generator and library.” in *20th International Workshop on Database and Expert Systems Application*, 2009.
- [102] R. Langmann and L. F. Rojas-Pena, “A plc as an industry 4.0 component.” in *2016 13th International Conference on Remote Engineering and Virtual Instrumentation (REV)*, 2016.
- [103] F. Lécué *et al.*, “Semantic traffic diagnosis with star-city: Architecture and lessons learned from deployment in dublin, bologna, miami and rio.” in *The Semantic Web ISWC 2014*, 2014.
- [104] M. J. Lightlill and G. B. Whitham, “On kinematic waves. ii. a theory of traffic flow on long crowded roads,” in *Proceedings of the Royal Society A* 229, 1955.
- [105] J. Lorey, “Identifying and determining sparql endpoint characteristics,” *International Journal of Web Information Systems*, 2014.
- [106] Y. Lu, “Cyber physical system (cps)-based industry 4.0: A survey.” *Journal of Industrial Integration and Management*, vol. 2.03, 2017.
- [107] Y. Lv *et al.*, “Traffic flow prediction with big data: a deep learning approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16.2, 2015.
- [108] C. M., B. P., B. L. *et al.*, “The ssn ontology of the w3c semantic sensor network incubator group,” *J. of Web Semantics*, vol. 17, 2012.
- [109] X. Ma *et al.*, “Large-scale transportation network congestion evolution prediction using deep learning theory,” *PloS one*, vol. 10.3, 2015.
- [110] —, “Large-scale transportation network congestion evolution prediction using deep learning theory.” *PloS one*, vol. 10.3, 2015.
- [111] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.
- [112] K. Makris *et al.*, “Ontology mapping and sparql rewriting for querying federated rdf data sources.” in *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, 2010.
- [113] J. B. McDonald and Y. J. Xu, “A generalization of the beta distribution with applications,” *Journal of Econometrics*, 1995.

- [114] N. Mihindukulasooriya, M. Poveda-Villalon, R. García-Castro, and A. Gomez-Pérez, “Loupe-an online tool for inspecting datasets in the linked data cloud,” in *International Semantic Web Conference (Posters and Demos)*, 2015.
- [115] P. Nesi, “Variational approach to optical flow estimation managing discontinuities,” *Image and Vision Computing*, vol. 11, 1993.
- [116] P. Nesi and M. Paolucci, “Supporting living lab with life cycle and tools for smart city environments,” in *The 24th International DMS Conference on Visualization and Visual Languages, DMSVIVA2018*, 2018.
- [117] P. Nesi *et al.*, “Km4city smart city api: an integrated support for mobility services,” in *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2016.
- [118] S. Newman, *Building Microservices*. OReilly Media, Inc., 2015.
- [119] A. Nicolae and A. Korodi, “Node-red and opc ua based lightweight and low-cost historian with application in the water industry,” in *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, 2018.
- [120] B. P., B. I., N. P., and R. N., “Graph databases methodology and tool supporting index/store versioning,” *J. of Visual Languages and Computing*, 2015.
- [121] L. Po, F. Rollo, J. R. R. Viqueiray, R. T. Ladoz, A. Bigi, J. C. Lopezx, M. Paolucci, and P. Nesi, “Trafair: Understanding traffic flow to improve air quality,” in *The 1st IEEE African Workshop on Smart Sustainable Cities and Communities (IEEE ASC2 2019)-In conjunction with the 5th IEEE International Smart Cities Conference*, 2019.
- [122] C. RB, “The screen test for the number of factors,” *Multivar Behav Res*, 1966.
- [123] P. I. Richards, “Shock waves on the highway,” *Operation Research*, vol. 4, 1956.
- [124] P. J. Rousseeuw, “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” *Journal of computational and applied mathematics*, vol. 20, 1987.
- [125] B. S., S. M., M. S., and D. S., “Role of middleware for internet of things: A study,” *Int’l J. of Computer Science and Engineering Survey*, 2011.
- [126] K. S. and A. W. Colombo, “Architecting the next generation of service-based scada/dcs system of systems,” in *IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society*, 2011.

- [127] D. R. Silva *et al.*, “Latency evaluation for mqtt and websocket protocols: an industry 4.0 perspective.” in *2018 IEEE Symposium on Computers and Communications (ISCC)*, 2018.
- [128] J. n. C. Soto *et al.*, “Towards a federation of smart city services.” in *International Conference on Recent Advances in Computer Systems*, 2015.
- [129] M. Tabaa *et al.*, “Industrial communication based on modbus and node-red.” *Procedia computer science*, vol. 130, 2018.
- [130] T. Q. Tang, W. S. Shi, X. B. Yang, Y. P. Wang, and G. Q. Lu, “A macro traffic flow model accounting for road capacity and reliability analysis,” *Physica A: Statistical Mechanics and its Applications*, vol. 392, 2013.
- [131] S. Valtolina, F. Hachem, B. Barricelli, E. Belay, S. Bonfitto, and M. Mesiti, “Facilitating the development of iot applications in smart city platforms,” in *International Symposium on End User Development*, 2019.
- [132] P. Y. Vandenbussche, J. Umbrich, L. Matteis, A. Hogan, and C. Buil-Aranda, “Sparqls: Monitoring public sparql endpoints.” *Semantic web*, 2017.
- [133] P.-Y. Vandenbussche *et al.*, “Linked open vocabularies (lov): a gateway to reusable semantic vocabularies on the web,” *Semantic Web*, vol. 8.3, 2017.
- [134] S. Wang, S. Schlobach, and M. Klein, “What is concept drift and how to measure it?” in *International Conference on Knowledge Engineering and Knowledge Management*, 2010.
- [135] J. S. Ward and A. Barker, “Undefined by data: a survey of big data definitions.” *arXiv preprint*, 2013.
- [136] B. Williams, P. Durvasula, and D. Brown, “Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models,” *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1644, 1998.
- [137] S. Yang, S. Shi, X. Hu, and M. Wang, “Spatiotemporal context awareness for urban traffic modelling and prediction: Sparse representation based variable selection,” *PloS one*, vol. 10.10, 2015.
- [138] P. Yue, J. Gong, L. Di *et al.*, “Integrating semantic web technologies and geospatial catalog services for geospatial information discovery and processing in cyberinfrastructure,” *Geoinformatica*, vol. 15, 2011.
- [139] M. Zuccalà and E. S. Verga, “Enabling energy smart cities through urban sharing ecosystems.” *Energy Procedia*, vol. 111, 2017.