Software Platforms for Smart Cities:
Concepts, Requirements, Challenges, and a Unified Reference Architecture

Eduardo Felipe Zambom Santana, University of S˜ao Paulo
Ana Paula Chaves, Federal Technological University of Paraná
Marco Aurelio Gerosa, University of S˜ao Paulo
Fabio Kon, University of S˜ao Paulo
Dejan S. Milojicic, HP Labs

With the growth of the urban population, the infrastructural problems and limited resources of thousands of cities around the world affect negatively the lives of billions of people. Making cities smarter can help improving city services and increasing the quality of life of their citizens. Information and communication technologies (ICT) are a fundamental means to move towards smarter city environments. Using a software platform on top of which Smart City applications can be deployed facilitates the development and integration of such applications. However, there are, currently, significant technological and scientific challenges that must be faced by the ICT community before these platforms can be widely used. This paper surveys the state-of-the-art in software platforms for Smart Cities. We analyzed 23 platforms with respect to the most used enabling technologies as well as functional and non-functional requirements. We classified the enabling technologies for the development of Smart Cities platforms in four categories – Cyber-Physical Systems, Internet of Things, Big Data, and Cloud Computing – and related them to the main functional and non-functional requirements. Based on these results, we derived a reference architecture to guide the development of next-generation software platforms for Smart Cities. Finally, we enumerate the most frequently cited open research challenges and discuss opportunities in the future.


Additional Key Words and Phrases: Wireless sensor networks, Software platforms

ACM Reference Format:
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Since 2009 most of the world population lives in cities [United Nations 2009] and current resources and infrastructure are hardly enough to cope with the increasing demand generated by population growth and geographic concentration [Caragliu et al. 2011]. Thus, making cities smarter can help optimize resource and infrastructure uti-
lization in a more sustainable way. One approach to achieve this is using the large amount of data generated by multiple city sources, such as sensor networks, traffic systems, user devices, and social networks. This data can be combined in creative ways to create integrated services and applications, improving city services and making better use of city resources. However, using all these data sources in an efficient and effective way is a difficult challenge.

In the research described in this paper, we evaluated initiatives for developing Smart City systems in a wide range of scenarios, such as transportation [Djahel et al. 2014], traffic control [Barba et al. 2012], air pollution [Vakali et al. 2014], waste management [Perera et al. 2014], health care [Hussain et al. 2015], public safety [Galache et al. 2014], water [Pérez-González and Díaz-Díaz 2015], energy [Yamamoto et al. 2014], and emergency management [Asimakopoulou and Bessis 2011].

Most of these previous solutions focus on a specific domain. These systems are normally targeted at a specific problem and they are developed from scratch with little software reuse. They do not interoperate; an approach that clearly leads to duplication of work, incompatible solutions, and a non-optimized use of resources.

Nevertheless, all these different domains require basic services from the underlying software infrastructure that could be provided by a novel, generic software platform which could include facilities for application development, integration, deployment, and management, easing the construction of sophisticated Smart Cities applications.

Using software platforms for Smart Cities environment is a promising solution to address those issues. We define a software platform for Smart Cities as an integrated middleware environment that supports software developers in designing, implementing, deploying, and managing applications for Smart Cities.

Many challenging issues still need to be addressed before a highly effective software platform for Smart Cities can be created. These include enabling interoperability between the multiple systems in a city, guaranteeing the privacy of the citizens, managing large amounts of data, supporting the required scalability, and dealing with a large variety of sensors.

The objective of this paper is to perform a comprehensive analysis of the most relevant functional and non-functional requirements for platforms for Smart Cities, according to the existing literature. Based on that, we derive a reference architecture addressing these requirements. With this survey, we intend to clarify important aspects of the design, development, and management of platforms for Smart Cities. To achieve that, our survey examines 23 software platforms for Smart Cities, to answer the following general research question:

What are the elements required for the development of a highly-effective software platform for enabling the construction of highly-scalable integrated Smart City applications?

We conceived three more specific research questions:

RQ1: “What are the enabling technologies used in state-of-the-art software platforms for Smart Cities?”
RQ2: “What are the requirements that a software platform for Smart Cities should meet?”
RQ3: “What are the main challenges and open research problems in the development of next generation robust software platforms for Smart Cities?”

To answer research question RQ1, we identified the most relevant enabling technologies employed in platforms for Smart Cities. As described in Section 2.2, we grouped them into four main categories: Internet of Things (IoT) [Atzori et al. 2010], applied...
to control sensors and actuators responsible for retrieving information from the city; **Big Data** [Mayer-Schönberger and Cukier 2013], to support storage and processing of the data collected from the city; **Cloud Computing** [Armbrust et al. 2010], to provide elasticity to the services and data storage; and **Cyber-Physical Systems** [White et al. 2010], to enable the interaction of systems with the city environment. To answer RQ2, we identified the most relevant functional and non-functional requirements for the development of a platform for Smart Cities, as described in Section 3.3. Finally, to answer RQ3, we explored the main challenges in developing software platforms for Smart Cities pointed out by researchers, as discussed in Section 4.

Combining the results of the three research questions, we derived a reference architecture. This architecture presents components to implement a software platform for Smart Cities, based on the most common enabling technologies, the requirements, and challenges surveyed during this research. Furthermore, we also discuss the critical implications of platforms for Smart Cities.

The remainder of this paper is organized as follows. Section 2 presents the definition of Smart Cities and introduces the four enabling technologies for platforms for Smart Cities. Section 3 presents the platforms, architectures, and implemented systems for Smart Cities, grouped according to the enabling technologies that each platform uses. Section 4 points out challenges and open research problems in the development of a platform for Smart Cities. In Section 5, we present a unified reference architecture for Smart Cities. In Section 6, we discuss the relationship between the requirements and the enabling technologies as well as their implications for the development of software platforms for Smart Cities. Section 7 presents the related work and, finally, Section 8 presents our conclusions.

### 2. MAIN CONCEPTS

In this section, we introduce concepts used in this survey. First, we present definitions of Smart Cities. Then, we discuss the most adopted enabling technologies for the development of software platforms for Smart Cities.

#### 2.1. Smart Cities Definitions

The term Smart City has many different definitions. Some of those definitions are not in the software context, focusing only on social or business aspects. Regarding software systems, many authors define a Smart City as the integration of social, physical, and IT infrastructure to improve the quality of city services [Caragliu et al. 2011; Hollands 2008]. Other authors focus on a set of Information and Communication Technology (ICT) tools to create an integrated Smart City environment [Hollands 2008; Washburn et al. 2009; Hall et al. 2000].

Giffinger et al. [Giffinger et al. 2007] assert that a Smart City has six main dimensions: smart economy, smart people, smart governance, smart mobility, smart environment, and smart living. Many authors accept this definition [Hernández-Muñoz et al. 2011; Papa et al. 2013] and there are even benchmarks to produce a ranking of the smartest city using those dimensions.

In their definition of Smart Cities, Washburn et al. [Washburn et al. 2009] and Hall et al. [Hall et al. 2000] emphasize the idea of the integration of software services and applications to improve regular city services, improving the life of their citizens. Following this idea, Kanter and Litow [Kanter and Litow 2009] declare that creating independent software for each city domain is not sufficient to create an environment for Smart Cities. They defend that all city subsystems (such as transport, education, energy, water) must be linked in a network as an organic whole to provide integration

---

1 Smarts Cities in Europe - http://www.smart-cities.eu
among all city subsystems. Also, in Caragliu et al. [Caragliu et al. 2011], the definition of Smart Cities highlights the sustainability and management of natural resources as a significant benefit of Smart Cities.

We agree with the vision that a city must have an integrated environment to facilitate the interoperability between the city sub-systems. Therefore, we propose the following definition.

A Smart City is a city in which their social, business, and technological aspects are supported by Information and Communication Technologies to improve the experience of the citizen when interacting with the city. To achieve that, the city provides public and private services that operate in an integrated, affordable, and sustainable way.

Based on this definition, we can observe that it is desirable to integrate services and applications in a unified technological infrastructure to make a city smarter. A sensible way to make the above definition a reality is by means of a well-designed software platform providing the infrastructure to deal with large volumes of data, a wide variety of devices and applications, the interoperability between the systems, and other problems related to Smart City environments.

There are many smart cities initiatives in many cities around the world with different maturity levels and with applications in different domains. Most of the initiatives are in Europe [Caragliu et al. 2011; Manville et al. 2014], USA\(^2\), Japan, and South Korea [Liu and Peng 2013]. Also, there are some isolated initiatives in other countries such as Brazil [Fortes et al. 2014] and the United Arab Emirates [Janajreh et al. 2013]. Figure 1 presents a map with cities that have at least one Smart City project described in the literature. The map shows that most of the projects are in developed countries, a few are in developing countries, and none in poor countries.

![Smart Cities initiatives around the world.](image)

2.2. Enabling Technologies

To answer the question “What are the main enabling technologies used in state-of-the-art software platforms for Smart Cities?”, we present the most adopted enabling

technologies that we found in our literature review. We observed four main technologies used by software platforms for Smart Cities: Cyber-Physical Systems, Internet of Things, Big Data, and Cloud Computing. In this section, we give an overview and relate them to Smart City research. Those technologies will be used later in this paper to group the analyzed platforms and help to understand better the requirements that the platforms must address.

Figure 2 presents an overview of the four enabling technologies that we found in our survey and examples of how they can contribute to a platform for Smart Cities.

**Internet of Things**
- Hardware (Sensors, Actuators)
- Middleware
- Data Collection

**Big Data**
- Data Processing
- Data Storing
- Data Analyses
- Data Visualization

**Cyber-Physical Systems**
- Computation in Physical Systems
- City Actuation

**Cloud Computing**
- Hosting Services
- Hosting Store and Compute
- Elasticity and Scalability

**Fig. 2. Platforms for Smart Cities Enabling Technologies**

### 2.2.1. Cyber-Physical Systems
Cyber-Physical Systems (CPS) can be characterized as the use of computation and communication technologies to improve the features of physical systems. Wan et al. [Wan et al. 2010] define CPS as integrations of computation with physical processes. The authors suggest the use of local and remote computational models in networked embedded computers to monitor and control physical processes.

Many real-world applications already use CPS [White et al. 2010], such as flight avionics, electronic medical devices, power grid control systems, and Smart Cities. However, some authors [Wan et al. 2010] claim that existing ICT solutions do not support applications with dynamically changing physical contexts. Thus, applying CPS should introduce this requirement to Smart City applications. According to Gurgen et al. [Gurgen et al. 2013], CPS enables applications to become aware of the changes in the physical context adapting their execution according to it.

An example of a Cyber-Physical System related to Smart Cities is the WreckWatch [White et al. 2010], an application for detecting traffic accidents. This application was developed for Android devices and polls the device accelerometer and GPS for current speed and acceleration information. If the car has a strong deceleration, the information is processed with a mathematical accident prediction model. If the model indicates a traffic accident, the application reports the accident to an accident response server.
2.2.2. Internet of Things. Coetzee and Eksteen [Coetzee and Eksteen 2011] define IoT as a vision where objects become part of the Internet. According to the authors, the objects have to be uniquely identified, with recognized position and status, and accessible to the network. Gubbi et al. (2013) [Gubbi et al. 2013] define three components in an IoT environment: the hardware that includes sensors, actuators, and embedded communication hardware; a middleware that processes and stores data received from the hardware; and a presentation layer in which users can access, manipulate, and visualize data extracted from the hardware. In this sense, this is very similar to what we expect from a platform for Smart Cities.

The very large number of devices used to collect data from a city forces platforms for Smart Cities to use IoT technologies. The data collected from these devices must be transmitted via interconnected networks so that they can be grouped and processed to provide advanced Smart City services. Zanella et al. [Zanella et al. 2014] present many possible uses of Internet of Things in Smart Cities. Some examples are: monitoring historical buildings health, detecting the load level of waste containers, monitoring the noise in central areas of the city, monitoring the conditions of traffic lights, and monitoring the usage of energy in Smart Homes.

2.2.3. Big Data. Many authors define Big Data as a set of techniques and tools to store and manipulate large data sets, where traditional technologies, such as relational databases and sequential processing tools, cannot deal with the vast volume of data. Many authors [Chen et al. 2014; Demchenko et al. 2014] describe four main characteristic to Big Data:

— Volume: the scale of data generated and collected is increasing rapidly and the Big Data tools have to deal with this challenge. In Smart Cities, it is important because the volume of data will be huge, coming from many data sources distributed in the city.

— Variety: the data can be collected from different sources, that can have structured, semi-structured or unstructured formats such as video records, relational databases, and raw texts. It is also important in Smart Cities because the data from the city will be collected from many data sources such as surveillance cameras, sensors, and citizen devices.

— Velocity: the data processing must be fast, and in some cases in real-time, or it can be useless, such as the data gathered from car sensors, the analysis of social networks, and information about the city traffic. It is real in Smart Cities which have to response to problems, such as traffic jams and accidents, in a few time.

— Veracity: because of a large amount of data collected and the use of multiple data sources, it is important to ensure the quality of data, because errors in the data or the usage of unreliable sources can compromise data analysis. In cities, it can occur also, such as wrong GPS readings, malfunction sensors, and suspicious users.

Figure 3 relates the four Vs of Big Data with the Smart Cities needs.

In Smart Cities context, Big Data tools are employed to support the amount of data generated from city devices. Sensor networks regularly transmit data about city conditions such as temperature, air quality, and pluviometry. Citizens generate data using smartphones and social networks. Vehicles send their positions continually.

Many Big Data tools are already being used by platforms for Smart Cities. Some of these tools are NoSQL databases [Khan et al. 2013; Bain 2014] such as MongoDB and HBase, parallel data processing tools [Parkavi and Vetrivelan 2013; Takahashi et al. 2012] such as Apache Hadoop and Apache Spark, real-time data streams processing tools [Girtelschmid et al. 2013] such as Apache Storm, and visualization tools [Khan et al. 2013] such as RapidMiner.
Al Nuaimi et al. [Al Nuaimi et al. 2015] discuss many possible applications of Big Data tools and applications in Smart Cities. For example, recognizing traffic patterns using historic data to realize the causes and avoid traffic jams, facilitate the decisions of city governments using the analyzes of great data sets, and forecast the use of resources, such as electricity, water, and gas, in different situations using historic and real-time data.

2.2.4. **Cloud Computing.** Cloud Computing offers a very large, elastic, and highly available infrastructure for both data storage and computation, which is essential for complex Smart City systems. In addition, a Smart City environment can be highly dynamic, requiring reconfigurations of the underlying infrastructure, which is also supported by Cloud Computing.

Many authors, such as [Distefano et al. 2012; Aazam et al. 2014] described the idea of combining IoT and Cloud Computing, coining the term “Cloud of Things”. Their idea is to store and process all the data from an IoT network in a cloud computing environment, which is currently used in some Smart City projects [Mitton et al. 2012; Tei and Gurgen 2014].

Another concept related to a cloud computing environment in Smart Cities is Software as a Service (SaaS). The work of Perera et al. [Perera et al. 2014] extended this concept, using the term “Sensing as a Service”. The idea is to provide the sensor data with a cloud computing infrastructure. The ClouT platform, presented in [Tei and Gurgen 2014], also uses the concept of software services and defines the terms City Application Software as a Service (CSaaS) and City Platform as a Service (CPaaS).

Some authors relate the use of Cloud Computing, Big Data, and IoT [Chen et al. 2014; Aazam et al. 2014] because a cloud environment is an ideal infrastructure to store data and execute services. Hence, the data generated from an IoT middleware can be stored and processed in a cloud environment using Big Data tools. This synergistic combination helps to support important non-functional requirements such as scalability, elasticity, and security.
3. PLATFORMS FOR SMART CITIES

In this section, we describe various platforms for Smart Cities found in the literature. All platforms use at least one of the enabling technologies presented in Section 2.2.

To find these studies, we used the following query string: (“Smart City” or “Smart Cities”) and (Platform or Middleware or Architecture). We did not include in our search other terms that are more rarely used to describe the application of ICT in cities, such as “Knowledge City”, “Intelligent City”, and “Connected City”. Figure 4 illustrate the use of these expressions in recent years using Google Trends.

![Trends in Smart City related terms](image)

However, the expression “Digital City” is still used. Thus, we analyzed the definition of this expression and the differences with “Smart Cities”. We found that, normally, the description of a digital city relates to the use of digital technologies in a city, but not with the goal of making smart services and improving the overall infrastructure of the city. In a digital city, the integration of the multiple systems is not at stake. The differences between these two concepts are discussed by Cocchia [Cocchia 2014] and by Yin et al. [Yin et al. 2015].

The next subsection describes existing platforms, developed as research projects with different approaches. Subsection 3.2 shows systems developed using these platforms. Finally, in Subsection 3.3 we present a set of functional and non-functional requirements extracted from our analysis of the platforms and systems described in the previous Subsections.

3.1. Platform Categories

To facilitate the presentation, we divided the platforms into five categories, according to the enabling technologies that each platform uses. Figure 5 presents an overview of the platforms for Smart Cities that we analyzed. In this Figure, we can observe that all platforms use Cloud Computing. Almost all of them use at least one more enabling technology, more commonly IoT and Big Data.

3.1.1. Internet of Things and Cloud Computing. In this section, we present the platforms that use both IoT and Cloud Computing as enabling technologies.

SmartSantander proposes a city-scale experimental research facility to support typical applications and services for a smart city [Sanchez et al. 2014]. The project is centered in Santander, Spain, with smaller facilities in other European cities. The platform processes a large variety of information, including data about traffic conditions, temperature, \( CO_2 \) emissions, humidity, and luminosity. Currently, the project has implanted more than 20,000 sensors in the city.
Padova Smart City [Zanella et al. 2014] uses IoT to create a sensor network with more than three hundred sensors in the city of Padova, Italy. The platform collects environmental data, such as CO₂ emissions and air temperature and monitors street lights. A feature highlighted in this platform is the use of common protocols and data formats to enable interoperability between the different city systems.

The European Platform for Intelligent Cities (EPIC) project [Ballon et al. 2011] proposes a complete IoT Middleware to facilitate the use and management of the Wireless Sensor Network (WSN). This middleware aims to deal with the heterogeneity, interoperability, scalability, extensibility, and configurability problems in a WSN.

ClouT [Tei and Gurgen 2014] proposes a two-layer architecture to collect data from the WSN and manage the sensors and actuators in the city network [Galache et al. 2014]. The first layer is the Sensors and Actuators Layer that handles data from the WSN. The second, the IoT Kernel Layer, manages and monitors the sensors and actuators network.

OpenMTC [Elmangoush et al. 2013] (Open Machine Type Communications) is Machine-To-Machine (M2M) based communication platform for Smart Cities. Its goal is to enable efficient communication among a high number of devices associating them with a set of services. To achieve this, the platform supports standard interfaces to multiple types of devices, efficient data/event processing methods to achieve real-time performance, and easy application development, providing a software development kit.

The analysis of the platforms mentioned above led to the identification of four major functional requirements: management of a WSN, management of the data collected from the city, management of services and applications, and an infrastructure to make the data from the platform available to city applications. This analysis also led to the identification of five non-functional requirements: adaptation, interoperability, scalability, extensibility, and configurability.

As weak points of these platforms, we indicate the lack of pre-processing components to verify the integrity of the data collected from the city and make small analysis in the
data such as aggregations and that most of the platforms do not include a discussion about security concerns on the platform.

3.1.2. Internet of Things, Cloud Computing, and Big Data. In this subsection, we present the platforms that use IoT, Cloud Computing, and Big Data as enabling technologies.

OpenIoT[^3] is an open source middleware for the development of IoT-based applications. It has an API to manage the WSN and a directory service to discover dynamically the sensors deployed in the city; it also has a layer for service definition and access. Big Data tools are used to store and analyze the data from the platform. A Smart City project called Vital [Petrolo et al. 2014] builds on this platform and uses the term “Cloud of Things” to refer to the use of Cloud Computing and IoT.

The Concinnity project [Wu et al. 2014] provides a generic platform as a service (PaaS) for sensor data management and applications. It is applied to build Big Sensor Data Applications. However, this platform focuses on multiple data sources such as the WSN, social networks, and data from platform users. It also includes a service directory where developers can find and publish services facilitating its reuse.

Both platforms, OpenIoT and Concinnity, offer developers tools to implement applications directly on the platform. OpenIoT allows the mash-up of the services defined in the platform and creates automatically a visual interface for end-users. Concinnity provides a set of developer tools, such as a Workflow Editor and Engine, a Service Publisher, and an Application Editor.

Sentilo [Bain 2014] is an open source sensor and actuator platform designed for Smart Cities that looks for openness and interoperability. Sentilo uses IoT concepts to control the WSN, and Cloud Computing to share data with the applications. Big Data tools are used mainly to collect and store data from the sensors ensuring platform scalability. The Sentilo project was originally designed to be deployed in the city of Barcelona; after its deployment, the City released the code under the LGPL and EUPL open source licenses.

The main functional requirements identified for this group of platforms were: management of a WSN, management of data life cycle (collect, store, process), making the data from the platform publicly available, a service directory for application developers, and tools for application development. As non-functional requirements, we identified: interoperability and scalability.

A weak point of these platforms is the lack of streams processing analyses to collect real-time data from the city, an important requirement for many Smart City applications. Another problem is that most of the platforms do not recover citizen data. Despite the privacy problems, it is important to allow context-aware services to the citizens.

3.1.3. Cloud Computing and Big Data. In this subsection, we present the platforms that use Cloud Computing and Big Data as enabling technologies.

Vilajosana et al. [Vilajosana et al. 2013] present a platform for Smart Cities based on Cloud Computing and Big Data, whose main components are data management and service hosting. It includes an Open Data API allowing third-party applications to access the data stored on the platform. Big Data tools are used to collect data streams and analyze data such as prediction and inference.

Scallop4SC (SCALable LOGging Platform for Smart City) [Takahashi et al. 2012; Yamamoto et al. 2014] uses Big Data to process a large volume of data gathered from smart buildings. The platform uses information about the building such as water and energy consumption, temperature, air humidity, and the amount of garbage generated.

[^3]: OpenIoT - https://github.com/OpenIotOrg/openiot
Periodically, the buildings send data to the platform for processing. The objective is to analyze smart building data and, for that, it uses the MapReduce algorithm.

CiDAP [Cheng et al. 2015] is a big data analytics platform deployed into the SmartSantander testbed. The platform uses data collected from SmartSantander and analyzes it to understand the behavior of the city. The main components of this platform are the agents, who collect data from the SmartSantander platform; the Big Data repository for storing the data; the Big Data processing for intensive data processing and analytics, and a CityModel server for communicating with external applications. This platform uses Apache Spark [Zaharia et al. 2010] to process the data.

[Khan et al. 2015] propose a Smart City architecture based on Big Data to achieve the necessary availability and scalability required for a Smart Cities platform. The architecture has three layers: a layer to collect, analyze, and filter data; a layer to map and aggregate data making it semantically relevant; and a layer where users can browse and recover the data processed from the other two layers. The implementation of the architecture use only open source projects and the authors have presented tools for all layers of the architecture [Khan et al. 2013].

WindyGrid [Thornton 2013], an initiative of the City of Chicago, is a platform for Smart Cities whose objective is to present real-time and historical data with a unified view of city operations. Big Data technologies, such as the MongoDB NoSQL database and parallel data processors were used to develop the platform.

SMARTY [Anastasi et al. 2013] is a project aiming at providing tools and services for innovative mobility and flexible city transport systems. Its software platform collects data from multiple sources, such as traffic flow, location of users, weather reports, pollution levels, delays of transport services, and parking availability. A network of low-cost sensors collects data from the city and user posts and messages in social networks are continuously monitored to extract useful knowledge. The platform processes the huge amount of data generated by the city with data mining techniques, such as classification, regression, clustering and frequent pattern analysis.

The platform proposed by Girtelschmid et al. [Girtelschmid et al. 2013] use semantic technologies to create a platform for Smart Cities, adding flexibility in system configuration and adaptation. However, to overcome the performance bottlenecks normally associated with ontology repositories and reasoning tools, the authors combine their semantic techniques with Big Data processing methods.

The main functional requirements identified for this group of platforms were: data management such as collecting, analyzing, and visualizing data, large scale data processing such as batch and real-time processing, and the use of semantic techniques combined with Big Data. As non-functional requirements, we identified: scalability and adaptation.

Most of the platforms of this section do not have an IoT layer and do not indicate how the data is collected from the city, the exception is the CiDAP that uses the SmartSantander testbed as an IoT middleware. Another problem is that most of the platforms do not include a discussion about security concerns on the platform.

3.1.4. Cloud Computing. In this subsection, we present platforms that use only Cloud Computing as enabling technology.

Piro et al. [Piro et al. 2014] present a two layered service platform for the creation of Smart City applications. The first is a low-level layer that controls the communication among the city WSN devices. The second layer collects the data from the devices and provides services for the development of applications that use the data from the city.

U-City [Lee and Rho 2010] is a platform for the creation of smart ubiquitous cities. The platform offers several service management features, such as autonomic service discovery, service deployment, and context-aware service execution. It also offers pre-
defined services such as an inference engine, a context-aware data service, and a portal for the management of the platform.

Gammas is a middleware for the development of Smart City applications [Apolinarski et al. 2014] that supports data acquisition, data distribution, and data integration. The platform also provides an application runtime to facilitate the development and deployment of services using city data and a service registry. The middleware supports context-awareness so that Smart City services can adapt to the citizen situation, behavior, and intent. All the communication in the platform is encrypted to ensure citizen’s privacy and security.

Civitas [Villanueva et al. 2013] is a middleware to support the development of Smart Cities services. It is used to facilitate the development and deployment of Smart City applications and to avoid the emergence of “information islands” [Qiu et al. 2010], i.e., disconnected applications that do not share relevant information. Citizens connect to the middleware via a special device called the Civitas Plug; it is used to ensure the privacy and the security of middleware users. The middleware has two main design principles to facilitate the application integration: Everything is a Software Object, promoting the consistency of the software design and reusability of the middleware, and Independence of the city layout, i.e., city services should not be exposed to the city layout.

The main functional requirements identified for this group of platforms were: service management and data management. As non-functional requirements, we identified: security, privacy, and context awareness.

A problem of the platforms presented in this section is that none of them use known frameworks to implement its components such as the inference engine and processing tools. It is a problem because it can difficult the maintenance of the platform. Another problem is that the platforms do not describe a mechanism to allow external access to the platform data.

3.1.5. Cloud Computing and Cyber-Physical Systems. In this section, we present platforms that use Cloud Computing and Cyber-Physical Systems (CPS) as enabling technologies.

Gurgen et al. [Gurgen et al. 2013] present a middleware for Smart Cities autonomic services with many “self-” properties such as self-organization, self-optimization, self-configuration, self-protection, self-healing, self-discovery, and self-description. They justify the usage of cloud computing to provide scalability, reliability, and elasticity to the platform. This platform provides to the application developers the contexts of individual users and that of the city.

Privat et al. [Privat et al. 2014] propose another CPS-based platform, whose main characteristic is the self-configuration and self-adaptation capabilities in smart environments, including Smart Cities. The purpose of this platform is to provide a shared distributed software infrastructure that collects data and reacts to changes in the environment.

Wan et al. [Wan et al. 2012] propose an event-based CPS platform with the objective of management and cooperation among M2M components by means of an event manager. This platform provides data and services to third-party applications through a publish/subscribe module. The platform also enables the design of event processing flows to manage the mission-critical wireless messages.

The main functional requirements identified for this group of platforms were: autonomic reaction to changes in the city environment, communication among city devices, and a publish/subscribe mechanism for applications to communicate with the platform. As non-functional requirements, we identified: configurability, adaptation, and context awareness.
The platforms of this section focus on the deployment, configuration, and execution of CPS devices in the city, but it lacks important requirements to this such as the monitoring and publication of the data from the devices. They also do not describe any mechanism to verify the data collected from the city discarding inconsistent data.

3.2. Systems
In this subsection, we illustrate applications of Smart City platforms by describing a few platforms built on top of such platforms. Table I presents an overview of the domains of the analyzed systems.

<table>
<thead>
<tr>
<th>Table I. Domains of Smart City Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>City Sensing</td>
</tr>
<tr>
<td>GAMBAS</td>
</tr>
<tr>
<td>SmartSantander</td>
</tr>
<tr>
<td>Padova Smart City</td>
</tr>
<tr>
<td>OpenIoT</td>
</tr>
<tr>
<td>WindyGrid</td>
</tr>
<tr>
<td>ClouT</td>
</tr>
<tr>
<td>Scallop4SC</td>
</tr>
</tbody>
</table>

The GAMBAS middleware was used for the development of two applications for the public transport in Madrid, Spain. Foell et al. [Foell et al. 2014] present a context-aware urban bus navigator to help users of the public transport system find the best buses to make their travel. Handte et al. [Handte et al. 2014] describe a system that estimates the number of passengers of city buses using smartphone sensing.

SEN2SOC [Vakali et al. 2014] is a system deployed on the SmartSantander platform that uses data streams from the city (e.g., sensor data) and social networks (e.g., Twitter) to create Smart City applications. Two examples of applications are: capturing the emotional state of city inhabitants and visualizing the air pollution in the city. Also in SmartSantander, Vlahogianni et al. [Vlahogianni et al. 2014] present an application to predict the utilization of parking lots in the city.

Two applications were developed using the Padova Smart City platform. Bui and Zorzi [Bui and Zorzi 2011] present a health care system whose main features are monitoring conditions of patients, sending their data directly to doctors, and calling emergency services if the patient has an urgent problem. Bressan et al. [Bressan et al. 2010] present a monitoring application to manage and collect data from all the light posts connected to the platform.

Anagnostopoulos et al. [Anagnostopoulos et al. 2015] present a waste management system implemented using the OpenIoT platform. It presents four models to prioritize critical trash bins such as bins close to schools, hospitals, and gas stations. The system was used to compare the four models according to the amount of CO$_2$ emitted and the traveled distance by the trash trucks.

The WindyGrid platform [Rutkin 2014] provides three main systems to the city of Chicago: Situational Awareness and Incident Monitoring to monitor and act on problems that are occurring in the city; Historical Data Analyses to predict the behavior of city systems such as traffic and health care; and Advanced Real-Time Analytics to
analyse the current situation of city systems. Some examples of the data used in these systems are log of emergency (911) calls, traffic conditions, public buildings information, and surveillance cameras.

Galache et al. [Galache et al. 2014] present four systems developed using the ClouT platform: an alert service to warn citizens about earthquakes in Fujisawa, Japan; a civil protection system which warns the population about environmental risks such as storms and earthquakes in Genova, Italy; a system to help elder people find healthy activities in the city of Mitaka, Japan; and, a sensing application to notify people about events in the city such as cultural acts and traffic accidents in Santander, Spain.

Yamamoto et al. [Yamamoto et al. 2014] present two systems developed for the Scallop4SC platform, both in the energy management domain. The first system offers a tool for the visualization of household energy consumption to analyze data at different levels such as state, city, and neighborhood. The other system is a wasteful energy detection service, available for smart homes.

The analyzed applications show that the most explored domains are traffic with applications to monitor the streets or help citizens to use the public transport and city sensing capturing data from the city using sensors such as air pollution and temperature. Most of the applications are developed externally to the platform, just using one or more services available on the platform.

3.3. Requirements for Smart City Software Platforms

In this section, to answer the second research question “What are the requirements that a software platform for Smart Cities should meet?”, we analyze the functional and non-functional requirements extracted from the analyzed platforms.

We assume that a platform implements a requirement if the literature is describing it explicitly states that or if the platform has a component or module that clearly fulfills that requirement.

3.3.1. Functional Requirements. The main goal of a platform for Smart Cities is to facilitate the development of Smart City applications. Towards this, most of the analyzed platforms implement requirements for collecting data from the city, managing and sharing data, and providing tools to facilitate the development of Smart City applications. Table II presents an overview of the functional requirements for Smart City platforms, which we describe in the following.

— **Data Management:** This is a requirement implemented by most of the platforms for Smart Cities, which includes collection, storage, analysis, and visualization of city data. The analyzed platforms use different techniques for this requirement such as relational databases [Hernández-Muñoz et al. 2011; Lee and Rho 2010], big data tools [Thornton 2013; Cheng et al. 2015], and customized tools implemented by the platform development team [Wu et al. 2014].

— **Applications Run-time:** Some platforms focus on managing the execution of its applications. The goal is to facilitate the deployment and integration of such applications. Some platforms provide a complete environment for developers to deploy their applications [Apolinarski et al. 2014]; others offer an execution run-time service for applications developed with tools provided by the platform [Petrolo et al. 2014; Wu et al. 2014].

— **WSN Management:** A lot of the analyzed platforms have a Wireless Sensor Network (WSN) management layer to control and monitor the devices deployed in the city. Most of these platforms use IoT concepts to organize and manage the WSN [Hernández-Muñoz et al. 2011; Tei and Gurgen 2014]. Other platforms [Bain 2014] do not mention this explicitly but still have a software layer to manage the city network devices. Some platforms include features to manage all the device activities.
such as adding, removing, and monitoring the sensors and actuators. Two platforms describe a WSN deployed in a city: Padova Smart City [Zanella et al. 2014] with 3000 sensors and SmartSantander [Hernandez-Munoz et al. 2011], with more than 15000 sensors.

— **Data Processing:** Some platforms use specific processing components such as inference engines [Lee and Rho 2010], workflow processing [Wu et al. 2014], and big data processing tools [Takahashi et al. 2012]. These components process large data sets, and their main purpose is to analyze, verify, aggregate, and filter the data from the city. In addition, some platforms [Girtelschmid et al. 2013; Cheng et al. 2015] make real-time analyses of data streams.

— **External Data Access:** Almost all platforms describe an interface for external applications to access the platform data. The most common approach is an API to allow the access to the data generated in the city. Some platforms use REST [Hernandez-Munoz et al. 2011; Elmangoush et al. 2013], others use cloud computing concepts [Ballon et al. 2011], and one proposes an open data platform [Zanella et al. 2014]. Also, a platform [Gurgen et al. 2013] uses the publish/subscribe paradigm to make the data and services available to applications.

— **Service Management:** Most of the analyzed platforms adopt a Service-Oriented Architecture in which the platform functionalities are offered by services [Issarny et al. 2011]. Some of them use services to provide features to applications such as access to raw sensors data [Petrolo et al. 2014] and analyzed data [Zanella et al. 2014], and
Software Engineering Tools: Some platforms provide a set of tools for the development and maintenance of services and applications. Some platforms create visual interfaces for describing and implementing applications [Petrollo et al. 2014]. Other platforms provide workflow design tools [Wu et al. 2014] to define data or service flows and create Smart City applications. Moreover, some platforms [Khan et al. 2013] use analytics and reporting tools to facilitate the development of data visualization and reports, and two platforms describe the use of a Smart City application SDK [Elmangoush et al. 2013; Apolinarski et al. 2014].

Definition of a City Model: Some platforms provide a model of the city to facilitate the manipulation and understanding of the platform data and to facilitate the integration of the collected data. For example, Cheng et al. [Cheng et al. 2015], the city model is used to allow queries in the data from the city sensor network. Privat et al. [Privat et al. 2014] use a finite-state model to represent the possible city data flows.

Based on the functional requirements described above, we can observe that the main platforms activities are to control the city data life cycle: (1) Collecting the data with a WSN, (2) Managing the data in the platform, (3) Processing the data using city models, (4) Sharing the raw and processed data allowing external access. These activities are very related to the enabling technologies such as IoT with the WSN management, Data Management and Processing with Big Data, and Service Management with Cloud Computing.

3.3.2. Non-Functional Requirements. Most of the non-functional requirements of Smart City platforms are related to large, heterogeneous distributed systems such as scalability, adaptation, and interoperability. Other non-functional requirements are related to the manipulation of critical and personal data from citizens such as security and privacy. Table III presents an overview of the non-functional requirements for Smart City platforms, which we describe in the following.

Interoperability: Different devices, systems, applications, and platforms compose a Smart City environment and all these components must operate in an integrated fashion; for example, sensors from multiple vendors, systems implemented in different languages, platforms that share data and users, and legacy systems that have to communicate with the new platforms. Previous work in the field adopted several techniques to handle this requirement: interoperable objects [Villanueva et al. 2013], adopting generic and standard interfaces [Gurgen et al. 2013], applying Semantic Web to integrate all platform components [Girtelschmid et al. 2013], and using a naming mechanism [Cheng et al. 2015] to recognize different devices or data sources.

Scalability: The amount of users, data, and services of a Smart City platform will be huge and can increase over time. For example, in the SmartSantander testbed there were more than 20,000 sensors, in a city of 178,000 inhabitants collecting a large amount of city data [Sanchez et al. 2014] and CiDAP collected more than 50 GBs of data in three months [Cheng et al. 2015]. This non-functional requirement is relevant to many functional requirements such as WSN management [Ballon et al. 2011], data management [Takahashi et al. 2012], and service management [Bain 2014].

Security: Malicious users can make fraudulent use of services and data provided by the platform. Many platforms have a component or describe mechanisms to handle
Table III. Non-Functional requirements for Smart City platforms

<table>
<thead>
<tr>
<th>Software Platforms</th>
<th>Interoperability</th>
<th>Scalability</th>
<th>Security</th>
<th>Privacy</th>
<th>Context Awareness</th>
<th>Adaptation</th>
<th>Extensibility</th>
<th>Configurability</th>
</tr>
</thead>
<tbody>
<tr>
<td>SmartSantander</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenIoT</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concinnity</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Civitas</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gambas</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Khan et al. 2013]</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Girtelschmid et al. 2013]</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Piro et al. 2014]</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Gurgen et al. 2013]</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Vilajosana et al. 2013]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CloudT</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Padova Smart City</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U-City</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sentilo</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WindyGrid</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPIC</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Privat et al. 2014]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SMARTY</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CiDAP</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


— **Privacy:** A Smart City platform collects and manipulates several citizen sensitive data such as medical records, user localization, and consuming habits. The challenge is to use these data while hiding or does not save identifiable information. Some of the strategies used to achieve this requirement are cryptography [Apolinarski et al. 2014], tokens to control the access to the data that users can manipulate [Villanueva et al. 2013], and anonymization [Mylonas et al. 2015].

— **Context Awareness:** As the city and user situation can change over time, many applications and services can provide better results using contextual information. Some platforms use information from users [Apolinarski et al. 2014; Privat et al. 2014] such as location, activity, and language. Other platforms use information from the city [Khan et al. 2013; Cheng et al. 2015] such as traffic conditions, climate, and air quality. Examples of context use are: showing a different language in an application to a tourist and changing the route of a user avoiding polluted areas.

— **Adaptation:** Related to context awareness, many platforms adapt their behavior based on context to achieve fault-tolerance, choose a closer server to improve efficiency, decide for batch or real-time processing, and adapt data from multiple data sources. This requirement is most used in platforms that use CPS as enabling technology [Privat et al. 2014; Wan et al. 2012], but other concepts are used to meet this requirement such as semantic technologies [Girtelschmid et al. 2013].

— **Extensibility:** The capability to add services, components, and applications to the platform is important to assure that the platform is able to meet evolving system requirements and user needs. Muñoz et al. [Hernández-Muñoz et al. 2011] state that
extensibility is valuable because we do not know what are the services a city will need, so it is important to allow easy extension of the platform. Scallop4SC [Takahashi et al. 2012] uses materialized views that developers can extend to implement their applications. Some platforms [Khan et al. 2013; Bain 2014] employ only open source tools facilitating the extensibility of the platform. CiDAP [Cheng et al. 2015] offers extensibility to enable the use of the platform in cities of different scales.

— **Configurability:** A Smart City platform has many configuration options and parameters that define its behavior at execution time such as defining pollution and congestion thresholds and the priority of services. Thus, it is important to allow (re)configuration of the many variables of the platform. Two platforms [Wan et al. 2012; Privat et al. 2014] highlighted the importance of self-configurability capacities because of the huge amount of configurations needed in a Smart City platform. Other platforms [Lee and Rho 2010; Kim and Lee 2014] provide a portal to centralize the configurations.

Based on the non-functional requirements described above, we can observe that some of them are very important to many functional requirements such as Scalability that is valuable to the WSN and Data Management, Security and Privacy that are important to all data requirements, Extensibility that is required to the Service Management, and Configurability that is important to all the functional requirements. The non-functional requirements are very similar to the challenges and open research problems that we present in the section.

4. CHALLENGES AND OPEN RESEARCH PROBLEMS

To answer RQ3 (“What are the main challenges and open research problems in the development of next generation, robust software platforms for Smart Cities?”), we analyzed the challenges pointed out by Smart City research papers. Table IV presents an overview of the main challenges, which we describe in the following.

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Description</th>
<th>Technologies/Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy</td>
<td>Protect data collected from citizens, city, and enterprises.</td>
<td>Cryptography, Anonymization, and Access Tokens</td>
</tr>
<tr>
<td>Data Management</td>
<td>Manage all the data collected in the platform</td>
<td>NoSQL and Relational Databases and processing tools.</td>
</tr>
<tr>
<td>Heterogeneity</td>
<td>Ensure the interoperability of devices and applications</td>
<td>Standards, Ontology, and a City Unified Model.</td>
</tr>
<tr>
<td>Energy Management</td>
<td>Guarantee the electricity used by devices deployed in the city.</td>
<td>M2M techniques.</td>
</tr>
<tr>
<td>Communication</td>
<td>Enable communication among heterogeneous devices.</td>
<td>M2M techniques.</td>
</tr>
<tr>
<td>Scalability</td>
<td>Allow the growth of devices and users connected to the platform.</td>
<td>Distributed tools and algorithms and P2P applications.</td>
</tr>
<tr>
<td>Security</td>
<td>Protect the city data, services, and infrastructure.</td>
<td>Access Tokens and Devices.</td>
</tr>
<tr>
<td>Lack of Testbeds</td>
<td>There are not sufficient testbeds to experiment Smart City solutions.</td>
<td>Simulators.</td>
</tr>
<tr>
<td>City Models</td>
<td>Define a model describing the city.</td>
<td>Web Semantic and Ontology.</td>
</tr>
<tr>
<td>Platform Maintaining</td>
<td>Maintenance of the city systems and infrastructure.</td>
<td>Monitoring and Alert tools.</td>
</tr>
</tbody>
</table>

— **Privacy:** It is the most cited challenge to implement a Smart City platform, the main reason pointed by Hassan et al. [Hassan et al. 2014] and Balakrishna [Balakrishna 2012] is because the data collected from the city includes personal, enterprise,
and governmental data that should not be accessed by other users, mainly malicious users. Moreover, Wan et al. [Wan et al. 2012] discuss legal problems in using data belonging to platform users.

— **Data Management:** Many authors also cite data management as a challenge because the platform has to store and process a large amount of data and use efficient and scalable data storage and processing algorithms [Su et al. 2011; Djahel et al. 2014; Perera et al. 2014]. Data Analysis is also a challenge because it is hard to extract useful knowledge from the data [Hassan et al. 2014]. Another challenge is data trustworthiness; for example, Wu et al. [Wu et al. 2014] claim that because of a large number of data sources it is difficult to ensure that all the data are correct.

— **Heterogeneity:** This is a challenge because of the differences between the devices in a Smart City and the difficulty of relating data from different sources [Wu et al. 2014; Su et al. 2011; Wan et al. 2012]. Naphade et al. [Naphade et al. 2011] raise the problem of managing data across all city systems because of variations in data from different sources. Other authors [Wenge et al. 2014] state that a Smart City platform has to define standards because of heterogeneous devices, systems, and domains.

— **Energy Management:** Some authors cite Energy Consumption as a challenge to be faced by all the components of the platform such as sensors, actuators, and servers [Perera et al. 2014]. Moreover, Hassan et al. [Hassan et al. 2014] point out that the energy management in a Smart City health care application is also important because applications or services in this domain cannot fail.

— **Communication:** As Smart Cities of the future will have a huge amount of devices, enabling the communication among these devices can be a challenge. Some authors [Wan et al. 2012; Hassan et al. 2014] discuss that some domains in a Smart City depend on mission-critical communication to ensure reliability such as health care and public safety. Moreover, Djahel et al. [Djahel et al. 2014] explain that good communication mechanisms are required to share data from the platform to the applications.

— **Scalability:** Within the next decades, the number of connected devices in a Smart City will increase continually [Balakrishna 2012] requiring a good level of scalability in the associated software platform. Moreover, the number of users, of services access, and data stored can increase with the population growth or for some special event in the city. Su et al. [Su et al. 2011] point out that a platform for a Smart City must support large-scale, efficient services.

— **Security:** Security problems are also important in platforms for Smart Cities, Hancke et al. [Hancke et al. 2012] discuss if the city networks will be safe from cyber-terrorism and cyber-vandalism. Gurgen et al. [Gurgen et al. 2013] highlighted the importance of security in CPS platforms because these systems have control of city infrastructure and a malicious user can corrupt the infrastructure, e.g., by tampering with traffic lights and light posts.

— **Lack of Testbed:** The lack of testbeds is cited by Elmangoush et al. [Elmangoush et al. 2013] and Muñoz et al. [Hernández-Muñoz et al. 2011] as a challenge to the development of platforms for Smart Cities because, without the testbeds, it is hard to make tests and experimentation and discover the real challenges that deploying a Smart City platform will present.

— **City Models:** Some authors also argue that it is hard to understand a city and describe an effective and efficient model for it. For example, Wu et al. [Wu et al. 2014] claim that it is necessary to create a useful model of the city to make intelligent decisions. Naphade et al. [Naphade et al. 2011] state that it is required to observe and understand the city activity to not generate unnecessary and empty models. Muñoz et al. [Hernández-Muñoz et al. 2011] state that a unified model of the city is required, so the huge amount of heterogeneous data generated can be shared among applications and services.
— **Platform Maintenance**: Three works state that deploying and maintaining the platform is a challenge. Perera et al. [Perera et al. 2014] discuss the difficulty in maintaining a middleware to manage millions or billions of devices connected to the platform. Similarly, Wenge et al. [Wenge et al. 2014] discuss that the administration of the platform can be a challenge due to its size and the very large number of devices spread across the city. Hancke et al. [Hancke et al. 2012] point out that addressing and coordination issues in the sensor nodes can be a problem, again because of the size of the city sensor network.

5. REFERENCE ARCHITECTURE FOR SMART CITY PLATFORMS

In this section, we present a novel, comprehensive reference architecture to guide the development of next-generation software platforms for Smart Cities. The platform was derived from architectures proposed in previous works with enhancements based on the research described in this paper. First, we describe and analyze the architecture of two platforms presented in the literature: CiDAP and OpenIoT. Then, based on these early work and on the answers to the research questions presented before, we derived a novel reference architecture. Finally, we compare our proposal with the other two architectures.

5.1. CiDAP

The City Data and Analytics Platform (CiDAP) is a Big Data based platform that aims to use the data collected from the city to enable context-awareness and intelligence into applications and services. This platform processes large data-sets collected from an IoT Middleware. Figure 6 presents the architecture of the platform which has the following five main components.

— **IoT-Agents** connect to the IoT middleware and serve as a gateway to the devices available to the platform. Each data source of the IoT middleware is mapped to an IoT-Agent.
— **IoT-Brokers** act as a unified interface to the IoT agents, facilitating the access to the middleware data. This component communicates with the Big Data Repository to send data to be stored and with the CityModel Server to send data to be used directly by applications.

— The **Big Data Repository** stores raw data collected from the city and processed data from the Big Data processing component. The platform uses the CouchDB[^4] NoSQL database, which stores data as JSON documents. This component also has an internal processing tool that makes simple processing such as transforming data into new formats or creating new structured views/tables to index data.

— **Big Data Processing** is responsible for complex or intensive processing using the data stored in the Big Data Repository such as data aggregation or data mining. Also, it processes historical data using batch processes or real-time data using data streams. This component uses Apache Spark for this processing.

— **City Model Server** is the interface of the platform to external applications. The CityModel API allows applications to perform simple queries, complex queries, and subscribe to specific pieces of data from the platform. Simple queries request the latest data from devices, complex queries request aggregated historical data, and subscription is a mechanism for applications to receive data from the devices periodically.

The red, green, and blue boxes in the Figure 6 highlight the concepts used to implement each layer of the platform. The IoT Connector box has components to facilitate the access of IoT devices in the platform. The Big Data box has components to store and analyze the data gathered from multiple sources. Finally, the Cloud Computing box indicates the interface of the platform with external applications which is implemented using cloud services.

CiDAP is mainly concerned with storing and processing a large amount of data in the platform. It is important because of the huge amount of data collected in a city. The strong points of its architecture are the data storage and processing, the real-time and batch processing modules, and the fact that the associated platform was already tested in the SmartSantander testbed.

An important limitation of CiDAP is that the platform does not foresee specific services and tools for application developers and does not allow the deployment of new services in the platform making its extensibility difficult.

### 5.2. OpenIoT

OpenIoT is an Internet of Things platform used by the Vital project [Petrolo et al. 2014] to create a Smart City platform. Figure 7 presents an overview of the platform architecture, which has three layers: the Physical Plane, the Virtualized Plane, and the Utility-App Plane.

The Physical plane is a middleware responsible for collecting, filtering, combining, and cleaning data from sensors, actuators, and devices. This plane acts as an interface between the physical world and the OpenIoT platform. The current version of OpenIoT uses the X-GSN middleware [Calbimonte et al. 2014], an open-source middleware for managing, monitoring, and controlling IoT devices.

The Virtualized plane has the objective of storing data, executing services, and scheduling the execution of these services. The main components of the Virtualized plane are the following:

[^4]: [http://couchdb.apache.org](http://couchdb.apache.org)
— The **Scheduler** receives requests for services and ensures the access to resources that the service needs such as data and data streams. This component is responsible for discovering the sensors required for a service execution.

— The **Cloud Data Storage** keeps all the data from the platform, e.g., data streams collected from the sensors and the data created within the platform such as user profiles, service definitions, and registered applications. For storing data collected from the IoT middleware, OpenIoT uses the Linked Sensor Middleware (LSM) [Le-Phuoc et al. 2012].

— The **Service Delivery and Utility Manager** has three primary functions: handling the combination of the data collected from the IoT middleware, allowing service definitions, and delivering the results of requested services to the platform or to third-party applications. Also, this component keeps track of the usage of the services defined in the platform for accounting and billing.

The Utility-App Plane is the user interface of the platform and it has three main components.

— **Request Definition** enables users to define new applications using the services deployed on the platform, including the definition of service mash-ups.

— **Request Presentation** executes the applications created in the Request Definition component. When a user executes an application, it communicates with the Service Delivery and Utility Manager to retrieve the results from the service executions.

— **Configuration and Monitoring** allows configuration of platform parameters such as periodicity of sensor data reads and monitoring the health of all platform devices and components.
OpenIoT is a complete platform handling almost all the main requirements that we described in the survey. The strong points of this platform are the use of an IoT middleware to configure and collect data from devices, the middleware to store the data collected from sensors, the development tools, and the fact that the platform is open source. However, its architecture does not consider other data sources such as social networks and does not provide support for pre-processing services relevant when dealing with Big Data.

5.3. The Unified Reference Architecture

Based on the answers to the research questions of this survey, the 23 platforms analyzed, and on the two architectures presented above, we derived a novel reference architecture for Software Platforms for Smart Cities. With this reference architecture, we answer the general research question stated in Section 1 (“What are the elements required for the development of a highly-effective software platform for enabling the easy construction of highly-scalable, integrated Smart City applications?”). Figure 8 presents an overview of the architecture.

The lowest level component of the reference architecture is the Cloud and Networking, responsible for the management and communication of the city network nodes. This component has to identify all the devices connected to the platform, including servers, sensors, actuators, and user devices. Using cloud computing concepts
is important to ensure some fundamental non-functional requirements including scalability and extensibility.

Just on top of the Cloud and Networking infrastructure, the reference architecture includes the **IoT Middleware** and the **Service Middleware**. The former has to manage the city IoT network and enable the effective communication of the platform with the user devices and city sensors and actuators. The Service Middleware has to manage the services that the platform will provide to the applications, performing operations such as publishing, enacting, monitoring, composing, and choreographing these services.

The X-GSN middleware can be used to implement the IoT Middleware, this middleware is already used in the OpenIoT project, another option is to use components of the Sentilo platform, that is also open-source and implement a complete IoT middleware. The CHOREOS framework [Issarny et al. 2011] can be used to implement the Service Middleware, this project aims to enable the choreography of large-scale service-based software systems.

To provide better services to the citizens, it is important for the platform to store some user data and preferences, this is the role of the **User Management** component. But, to ensure user privacy, this data must be properly protected and permission to store it must be acquired from the user. Moreover, as the city platform will have many applications, it can be helpful to offer a single sign-on mechanism.

Social networks will have a major role in Smart Cities. They can be used to retrieve data from city conditions and can be an efficient communication channel between the platform and city government with the citizens. Therefore, it is important to allow the integration of the Smart City platform with existing social networks. This is the role of the **Social Network Gateway**. To implement this gateway many tools can be used, such as Spark Streaming that reads data streams of the Twitter and the Spring Social that is a Java-based framework to facilitate the connection with social networks such as Twitter, Facebook, and LinkedIn.

**Big Data Management** is a module to manage all the data in the platform. It is responsible for storing the data collected from the city and generated by the platform. To this extent, the reference platform has three repositories: (1) an **App Repository** to store applications including its source/binary code, images, and associated documents; (2) a **Model Repository** to store the city models such as traffic model, sensor network model, data model, city maps, and energy distribution model; and (3) a **Data Repository** to store the data collected from sensors, citizens, and applications. Because of the amount of data that a platform for Smart Cities can generate, NoSQL databases can be more suitable than relational databases.

Besides the data storage, the Big Data Management module is also responsible for the processing of the city data. There are two types of data processing that might be more suitable for different situations: **Stream processing**, to perform real-time analytics and data-flow processing and **Batch processing**, to analyze large data-sets. Moreover, this module must be capable of performing useful pre-processing tasks such as data filtering, normalization, and transformation.

The Big Data component also has a **Machine Learning** component to facilitate the understanding of the city by building models of city processes behaviors automatically and making predictions of city phenomena. As a Smart City will produce an enormous amount of data, a **Data Cleaning** component will be responsible for garbage collection, deleting unneeded data and archiving old data on slower, high capacity data stores.

To implement the Big Data Management components, many open-source tools are available. To the repositories, NoSQL Databases, such as CouchDB, MongoDB, and Cassandra, can store the unstructured or semi-structured data such as sensor reads.
and social networks posts. Relational Databases, such as MySQL and PostgreSQL, can store structured data, such as user information and the platform configuration.

To implement the processing engines, many tools are also available. To make batch processing, Apache Hadoop and Apache Spark are widely used by other platforms. Apache Spark also provides a stream data processing tool, likewise Apache Storm. Many tools offer machine learning algorithms to process large data sets such as Weka5, Spark MLlib, and Scikit-Learn6.

By relying on the middleware components described above, application developers and smart city operators will be able to develop and deploy Smart City applications. By using open data and open services provided by a city, common citizens and users may also execute, or even, develop novel applications to run on top of the city smart infrastructure. The applications will use the services and data from the platform, but also will generate and store data on the platform. The platform shall provide an SDK to facilitate the development of applications including tools such as an Integrated Development Environment (IDE), libraries and frameworks for commonly used programming languages, and a Smart City Simulator for debugging and experimenting with applications before real deployment.

All components of the platform must support several non-functional requirements such as scalability, security, privacy, and interoperability. Scalability is fundamental because of the huge amount of devices, data, and services in the platform. Privacy and Security are important because the platform collects, stores, and processes sensible data from the city and citizens. Interoperability will allow the integrated operation of different types of services, devices, and applications. Also, Table V presents the option to implement the reference architecture using tools that the platforms presented in the survey use.

<table>
<thead>
<tr>
<th>Component</th>
<th>Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoT Middleware</td>
<td>Sentilo and X-GSN</td>
</tr>
<tr>
<td>Data Repository</td>
<td>MongoDB, CouchDB, MySQL, IBM DB2, and Redis</td>
</tr>
<tr>
<td>Data Processing</td>
<td>Spark and Hadoop</td>
</tr>
<tr>
<td>Stream Processing</td>
<td>Storm</td>
</tr>
<tr>
<td>Cluster Management</td>
<td>Apache ZooKeeper and Hadoop YARN</td>
</tr>
<tr>
<td>Cloud Environment</td>
<td>OpenNebula and Microsoft Azure</td>
</tr>
<tr>
<td>Data Access</td>
<td>REST APIs and Jersey</td>
</tr>
<tr>
<td>Security</td>
<td>SAML Protocol</td>
</tr>
<tr>
<td>Machine Learning</td>
<td>Weka, Spark MLlib, and Scikit Learn</td>
</tr>
</tbody>
</table>

5Weka - https://weka.wikispaces.com/

5.4. Architectural Discussion

In our architecture, we combined aspects of both platforms described in the beginning of this section. Our Big Data module is similar to the one in CiDAP; both foresee batch and real-time processing and big data storage components. However, we added the idea of an application repository – to store data and meta-data associated with applications so that we can better manage and reflect on the applications executing in the city – and a model repository – to store different types of models associated with various city-related phenomena such as different kinds of maps, data flows, user behaviours, automated processes, and the like.

Similar to OpenIoT, we included a Cloud and Networking layer to manage the devices that collect data from the city and execute service and application components.

ACM Computing Surveys, Vol. 9, No. 4, Article 39, Publication date: March 2015.
We also included a service middleware to support many service-related operations such as deployment, management, composition (via orchestrations and choreographies), and enactment; OpenIoT also provides a Service Delivery component with a more limited support for some of these operations.

We also included some components that are not in these two architectures but were considered relevant in our literature review. the first is the Social Network Gateway, which is important because social networks will connect citizens, the city administration, and service providers and will generate a lot of useful data for city applications. The large majority of the urban population nowadays are immersed in communication and interactions via the most popular social networks; thus, we believe it is of paramount importance to build smart city platforms around the social networks.

Although OpenIoT provides some development tools to create applications using the available services, a Smart City platform will have to provide a complete software development toolkit. This SDK has to be aware of all the components of the platform and enable the construction of sophisticated mash-ups based on them. For example, it has to allow the development of a service using data from the IoT middleware combined with data from social networks generating a data stream that is filtered, processed, distributed to other users that have subscribed to a specific channel and, later, summarized and stored in a long-term persistent storage, maintaining historical records.

6. DISCUSSION
In this section, we discuss the findings of our research. Subsection 6.1 relates the four enabling technologies with the functional and non-functional requirements. Subsection 6.2 discuss the open research challenges. Subsection 6.3 presents the implications of our survey to city stakeholders such as city managers, citizens, and developers. Finally, subsection 6.4 points out the limitations of this survey.

6.1. Enabling Technologies and Requirements
This survey presented multiple approaches for the supply of Smart City platforms. From this study, four highly significant functional requirements emerged: management of sensor and actuator networks; management of the data collected from the city; provisioning, management, and development of services; and an environment for the development and deployment of Smart City applications. These features can be related to the enabling technologies, mapping them to the major functional and non-functional requirements of Smart City platforms.

Technologies around the Internet of Things are used to deal with the management of the sensor and actuator networks and their challenges such as heterogeneity, scalability, and adaptation. Big Data and Cloud Computing are used to deal with the huge amount of data generated from multiple data sources in the city such as the WSN, social networks, and user devices. Big Data tools are required for most data-related activities such as storing, analyzing, and sharing. Cloud Computing provides a scalable and elastic environment to store and process city data.

Figure 9 shows the relation between the implemented functional requirement from platforms and the enabling technologies. For example, it is possible to verify that most of the Big Data platforms handle Data Management and Data Processing. Cloud Computing platforms handle External Data Access and Service Management.

Figure 10 relates the non-functional requirements and the enabling technologies. We can observe that most platforms are concerned with scalability regardless of the enabling technology used. It is possible to verify relationships between other non-functional requirements and the technologies. For example, all the CPS platforms handle configurability. Extensibility is offered mostly by platforms that use Big Data and, interoperability, mostly by platforms that use IoT.
6.2. Challenges and Open Research Problems

Most of the significant challenges and research problems in implementing a platform for Smart Cities is related to data management. The most cited problem in the literature is ensuring the privacy of user data because of the amount of personal and critical data that a platform needs to handle such as user locations and medical records.

The second most cited challenge is heterogeneity because of the large number of different systems, services, applications and devices that a platform has to support.

ACM Computing Surveys, Vol. 9, No. 4, Article 39, Publication date: March 2015.
It was a surprise that just three authors cited scalability as a problem because, certainly, it will be a great challenge to support the huge number of devices, users, data, and services in a large metropolis.

An important and understudied issue is how to create a generic platform to support the requirements of different cities. Some literature focuses on a particular city such as the WindyGrid, SmartSantander, and Padova Smart City. Other platforms provide solutions without discussing the characteristics of the cities in which that solution should be applied. The studies proposing generic solutions for Smart Cities lack a discussion concerning how the components of these platforms could be adapted to fit cities with different sizes and characteristics.

6.3. Implications

This paper presented the most important features that software platforms for Smart Cities should handle. The results give important references for several city stakeholders such as platform developers, application developers, city managers, system operators, end-users, and Smart City researchers. In this section, we discuss the potential implications of our findings for these stakeholders.

The enabling technologies highlight the infrastructure needed to build Smart Cities. City managers can use this information to improve their investment decisions. Big Data and Cloud Computing deal with an enormous volume of data storage and network infrastructure to access data and services. The city must be equipped with sensors, actuators, and Internet services to take advantage of the Internet of Things and Cyber-Physical Spaces. Besides, the survey can help Smart City application and system developers to decide what technologies to use.

The reference architecture highlights the functional and non-functional requirements that platforms and applications developers should consider when developing software for Smart Cities. For platforms developers, this survey indicates that it is necessary to deal with big heterogeneous and distributed systems and critical and personal data in an effective and efficient way. For applications developers, the reference architecture shows what kind of services and data they can use to provide better experiences to their end-users. By discussing examples of these systems, we show to end-users, or citizens, the range of system domains that can be developed to facilitate their daily routine, such as urban mobility, air pollution, and health care.

Finally, this survey can also help Smart City researchers by discussing the main open research questions and challenges to be overcome to build smarter cities. These challenges can guide future work on this research area.

6.4. Limitations

In this survey, we decided to describe only the most cited enabling technologies used by Smart City platforms. However, we found other technologies less employed such as M2M Communications and the Semantic Web. These non-cited technologies are used by few platforms or are used to solve a small problem and are not a fundamental architectural component of the platform. Thus, there might be key technologies that end up being very relevant in the future that have not yet been identified in this survey.

We use the main paper of the research to describe components, requirements, and features of the platforms. Other papers or the site of the project can define different aspects.

Also, in this research, we classified the papers according to the enabling technologies only when they were explicitly mentioned. However, we have noticed that, in some papers, they were pointed out as a motivating aspect or future work. For example, Khan et al. [Khan et al. 2013] do not explicitly mention IoT in the architecture, but...
the authors discuss the possibility of using smart hardware such as sensor networks or smart household appliances, which can be organized in an IoT system.

7. RELATED SURVEYS
In our literature search, we found four papers that also surveyed platforms and applications for Smart Cities.

Da Silva et al. [da Silva et al. 2013] surveyed architectures of Smart Cities platforms analyzing the requirements handled by the platforms. However, they do not distinguished functional and non-functional requirements and did not address future research and open challenges in the area.

Yin et al. [Yin et al. 2015] conducted a survey on Smart Cities. Although the paper presents some platforms, the main goals of their work was to understand the concept of Smart Cities, identifying the enabling technologies and Smart City research issues. In our work, we studied Smart City software platforms and the related ICT problems in a systematic way to derive the major functional and non-functional requirements, a reference architecture for the field, and the technical and research open challenges.

Al Nuaimi et al. [Al Nuaimi et al. 2015] reviewed the use of Big Data tools and concepts in applications for Smart Cities. The paper presents mainly the relation between the challenges to creating applications for Smart Cities and the use of Big Data tools. It also identifies Smart City requirements that Big Data tools can address. It has some similarities with our work, but we conducted a more general and comprehensive survey.

Finally, Botta et al. [Botta et al. 2015] presented a study of the integration of Cloud Computing and the Internet of Things, defining this novel paradigm as CloudIoT. They describe applications that use this paradigm such as health care, transportation, and smart cities. The paper presents platforms that use the two concepts, some of which are also presented here such as OpenIoT and ClouT.

8. CONCLUSIONS
Smart City is a concept that has gained increased attention in academic, industrial and governmental circles. While the urban population is growing, the infrastructure and resources required to support citizens are often insufficient, leading to a degradation in public services. Information and Communication Technologies can provide important tools to minimize this problem, helping to improve the sustainable use of resources, city services, and the citizens' quality of life.

Using a software platform rather than ad hoc solutions is more robust and sustainable way to support the features needed by a Smart City environment. In this paper, we surveyed the current research on Smart Cities platforms aiming to discover its most relevant requirements and how to facilitate the development, integration, and deployment of Smart City applications. We analyzed 23 studies from different groups, proposing multiple approaches for the development of a software platform to answer our general research question “What are the elements required for the development of a highly-effective software platform for enabling the easy construction of highly-scalable, integrated Smart City applications?”.

Based on the analyzed projects, we derived a unified reference architecture supporting the main requirements needed to build a software platform for Smart Cities. Thus, this paper contributes to the state-of-the-art by providing a guide to help software developers and city managers to decide what are the necessary components to handle the functional and non-functional requirements of a software platform for Smart Cities.

The reference architecture is based on the answers of the three research sub-questions. RQ1 (“What are the enabling technologies used in state-of-the-art software platforms for Smart Cities?”) showed us that the Internet of Things, Cloud Computing,
Big Data, and Cyber-Physical Systems are the most cited enabling technologies. Answering RQ2 (“What are the requirements that a software platform for Smart Cities should meet?”), we could relate these technologies to the requirements that a software platform must handle. For example, most of the Big Data platforms mention Data Management as a requirement, while Configurability is strongly related to CPS platforms. In this way, an important contribution of this survey is to discuss what are the requirements to be implemented when using a specific enabling technology. In contrast, it helps to decide which technology to use when a specific functional or non-functional requirement is desirable.

Finally, to answer RQ3 (“What are the main challenges and open research problems in the development of next generation, robust software platforms for Smart Cities?”), we presented the most cited challenges and open research problems according to the literature. These challenges were considered when deriving the reference architecture. In this sense, an important contribution of this survey, especially for developers and researchers of software platforms, is to identify which platform components should be the focus on future works.

This survey described several Smart City initiatives, but all of them are still in their initial phases and pose multiple challenges and open problems to be addressed. A collaborative effort of research groups, commercial companies, NGOs, and governments is required to tackle the multitude of scientific, technical, political, and social problems related to the establishment of really-smart cities, reaching the ultimate goal of improving the quality of life of all of its citizens, irrespective of its social and financial situation.

REFERENCES


Wolfgang Apolinarzki, Umer Iqbal, and Josiane Xavier Parreira. 2014. The GAMBAS middleware and SDK for smart city applications. In Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on. 117–122. DOI: http://dx.doi.org/10.1109/PerComW.2014.6815176


sive IoT Data. In Service-Oriented Computing and Applications (SOCA), 2014 IEEE 7th International Conference on. 324–327. DOI: http://dx.doi.org/10.1109/SOCA.2014.47


Anbu Parkavi and Nagarajan Vetrivelan. 2013. A smart citizen information system using Hadoop: A case study. In Computational Intelligence and Computing Research (ICIC), 2013 IEEE International Conference on. 1–3. DOI: http://dx.doi.org/10.1109/ICIC.2013.6724276


Kehua Su, Jie Li, and Hongbo Fu. 2011. Smart city and the applications. In Electronics, Communications and Control (ICECC), 2011 International Conference on. 1028–1031. DOI: http://dx.doi.org/10.1109/ICECC.2011.6066743


Santana, E. F. Z. et al.


Flix J. Villanueva, Maria J. Santofimia, David Villa, Jess Barba, and Juan Carlos Lopez. 2013. Civitas: The Smart City Middleware, from Sensors to Big Data. In Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2013 Seventh International Conference on. 445–450. DOI: http://dx.doi.org/10.1109/IMIS.2013.80


Rong Wenge, Xiong Zhang, Dave Cooper, Li Chao, and Sheng Hao. 2014. Smart city architecture: A technology guide for implementation and design challenges. Communications, China 11, 3 (March 2014), 56–69. DOI: http://dx.doi.org/10.1109/CC.2014.6825259


ChuanTao Yin, Zhang Xiong, Hui Chen, JingYuan Wang, Daven Cooper, and Bertrand David. 2015. A literature survey on smart cities. Science China Information Sciences (2015), 1–18. DOI: http://dx.doi.org/10.1007/s11432-015-5397-4


Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. 2014. Internet of Things for Smart Cities. Internet of Things Journal, IEEE 1, 1 (Feb 2014), 22–32. DOI: http://dx.doi.org/10.1109/JIOT.2014.2306328