Connecting physical things to a SmartCity-OS

Riccardo Petrolo, Aikaterini Roukounaki, Valeria Loscrì, Nathalie Mitton, John Soldatos

Connecting physical *things* to a SmartCity-OS

Riccardo Petrolo*, Aikaterini Roukounaki†, Valeria Loscrí*, Nathalie Mitton*, and John Soldatos†
*Inria Lille - Nord Europe, France. e-mail: {name.lastname@inria.fr}
†Athens Information Technology, Greece. e-mail: {arou@ait.edu.gr, jsol@ait.gr}

***Abstract*—A Smart City can be seen as a system in which different Internet of Things (IoT) solutions coexist and cooperate. According with this vision, the number of IoT deployments is, nowadays, in continuous expansion and it involves disparate scenarios, from street lighting, waste management, etc. However those initiatives are standalone, based on different protocols and standards, while the Smart City concept requires, on the other hand, integration and interoperability among all its stakeholders. To face this problem, in this paper we introduce the VITAL-OS architecture, that can monitor, visualize, and control all the operations of a city. Then, we present a practical use case of connecting a Sensor Network to this OS and we describe eCACHACA, a ranking mechanism that facilitates the discovery of services provided by each sensor. Performance has been evaluated via experimentation on the FIT IoT-LAB, and results demonstrate the effectiveness in the discovery of resources.**

***Index Terms*—Smart Cities, Internet of Things, Sensor Networks.**

I. Introduction

The Internet of Things (IoT) is in continuous expansion as a result of the huge interest raised in both academia and industry. The number of devices deployed nowadays is already massive - thanks also to the cost reduction of smart technology - and it will reach 50 billion according to CISCO white paper [1]. Thanks to those devices, the IoT will change all the aspects in our lives, e.g., work, health, transport, etc. [2]. Looking at the bigger picture, the Smart City concept represents a clear example of coexistence and cooperation between different IoT ecosystems; it can be seen, indeed, as a system that integrates all the IoT solutions [3], especially the ones that are crucial for a city scenario. To confirm this momentum, in the last years, the Smart City concept gained significant interest behind which there is a real need to make cities ready to face new challenges (e.g., waste management, traffic congestion, etc.). In this context, the IoT has a primary role since it represents the main "supplier" in terms of data streams and information. According with this vision, the number of IoT solutions is, nowadays, rising exponentially involving different scenarios, from street lighting, to traffic intersections management, etc. Nevertheless, those initiatives are standalone, based on different protocols and standards, while the need of *integration* and *interoperability* among all the Smart City stakeholders is clear [4].

To face this problem, the European FP7 VITAL [1] project, introduces an abstract virtualized layer that operates across multiple IoT architectures and platforms. This layer allows the development, deployment and operation of IoT applications for Smart Cities, thereby turning VITAL into an operating system that can monitor, visualize, and control all the operations of a city [5].

In this paper we describe the VITAL-OS architecture and how it is designed to deal within different Smart City scenarios. We present a practical use case of monitoring in a Smart Building context; we first illustrate a discovery and ranking mechanism for Sensor Networks (SN), then, we show how to connect those sensors to VITAL-OS and how to use them. In order to evaluate the performance of the proposed algorithm, we performed experimentation on the FIT IoT-LAB testbed [2].

The paper is structured as follows: in Section II we introduce the VITAL-OS architecture. In Section III we describe, in detail, how to connect physical "things" to VITAL-OS by using a Raspberry Pi 2. The performance is discussed in Section IV, while Section V shows how to use the VITAL-OS. Section VI browses the most influential initiatives towards IoT integration and compares them with the proposed VITAL; Section VII concludes the paper.

II. VITAL-OS

The VITAL-OS architecture, as shown in Figure 1, is organized in three main layers, each of which is made of different modules. In the following, we present the features of the fundamental modules:

- **IoT Platforms and Data Sources**. At this layer, different data sources stand. In order to be virtualized and integrated into VITAL, those systems have to expose an implementation of the well-defined PPI (Platform Provider Interface).
- **IoT Data Adapter**. Its objective is to access low-level capabilities of the IoT Systems (through PPI), and to feed the acquired data and meta-data into the VITAL-OS.
- **Data Management Services (DMS)**. It represents the core of the VITAL-OS; it provides cloud-based functionality for managing data and meta-data.
- **ICO and Service Discovery (SD)**. It is used in order to dynamically discover ICOs (Internet-Connected Objects) and attached services [6] in the scope of horizontal integrated IoT applications spanning multiple platforms and business contexts.
- **Filtering**. It is used in order to reduce the information associated with individual data streams persisted in the DMS, thereby optimizing processing performance and economizing on network bandwidth.

[1] http://vital-iot.eu

[2] http://www.iot-lab.info

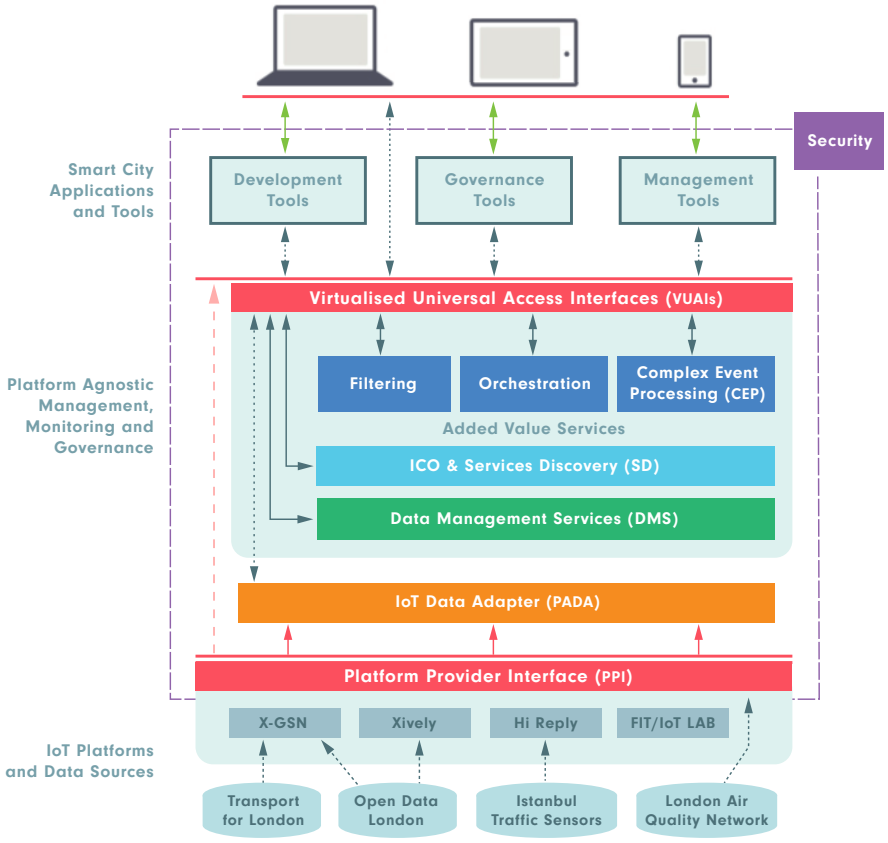


Fig. 1: The VITAL-OS architecture.

- **Complex Event Processing (CEP)**. It enables the processing of data streams in order to identify patterns and/or infer events.
- **Orchestration**. It has been designed to combine and manage multiple services from the above-listed modules, in order to deliver new added-value services. The combinations of the various services are defined by the users as workflows.
- **Development Tools**. This module is available at the *Smart City Applications and Tools* layer. Here, we support the development, integration, deployment and operation of Smart City applications [7].

III. Connect "things" to VITAL-OS: Design and Analysis

We next describe a practical example of connecting a Sensor Network (SN) to the VITAL-OS with the objective to monitor a Smart Building environment. In order to make this connection possible, it is fundamental to use a gateway that communicates, from one side, with the IoT Data Adapter module of the VITAL-OS, and on the other, acts as a sink for the SN. Figure 2 shows an example of interaction between the aforementioned players, while detailed aspects are listed in the following subsections.

A. The Sensor Network

In order to sense the physical environment, we chose Maxfor nodes (Figure 3). As shown in Table I, a node embeds a MSP430 as CPU and CC2420 as radio frequency module, and it is integrated with temperature and light sensors.

The sensor nodes use Contiki-OS [3] and in particular we flashed their firmware with eCACHACA, an extended version of the ranking mechanism CACHACA (Confident-based Adaptable Connected objects discovery to HArmonize smart City Applications) that we proposed in [8]. By running this algorithm, sensor nodes can evaluate and classify their neighborhood and the available services (e.g., temperature, light, humidity, etc); each node indeed, advertises itself and the services that it can observe. CACHACA is based on a rule-based fuzzy inference system and the use of parameters such as the Received Signal Strength Indication (RSSI) and the Timestamp of the last frame received from a neighbor in order to introduce two functions - *Physical* (φ) and *Service* (ω) *Confidence* - used to rank the nodes and services. In this work, we just consider the Physical confidence, since we assume that nodes are offering only one service, i.e., temperature.

[3] http://www.contiki-os.org

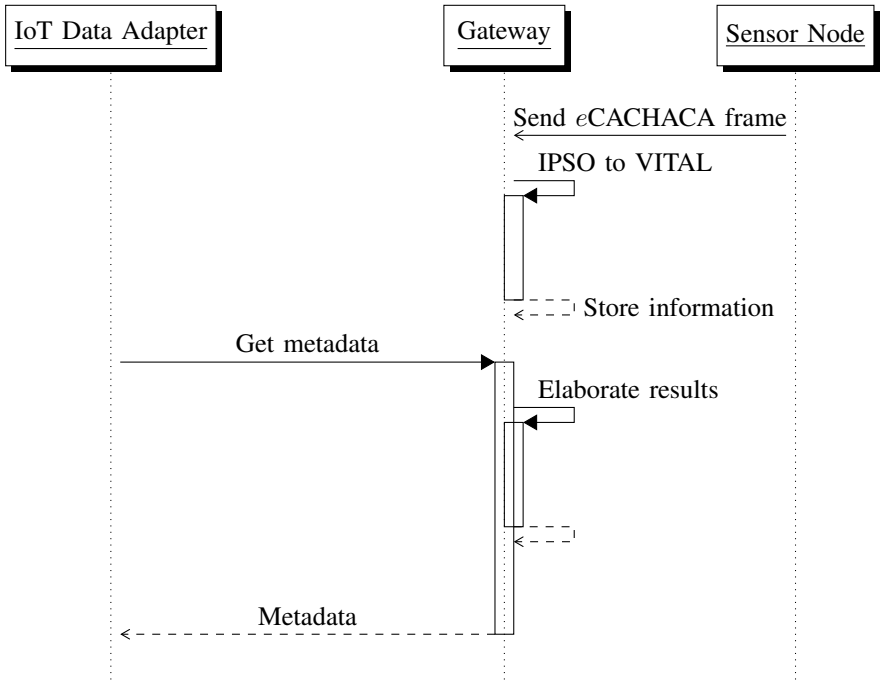


Fig. 2: Gateway - Example of interaction.

Fig. 3: Maxfor sensor node. (source http://www.maxfor.co.kr)

TABLE I: Sensor node specifications.

Parameter	*Specification*
CPU	MSP430
RF Chip	Texas Instruments ®CC2420
Frequency Band	2.4 GHz − 2.485 GHz
Transfer Rate	250 Kbps
RF Power	-25 dBm − 0 dBm
Range	120m [outdoor]; 20-30m [indoor]
Sensor Temperature & Humidity	Sensirion ®SHT11
Sensor Light	600 nm peak

eCACHACA borrows its main features from its "ancestor" - nodes describe their services according to the standardized IPSO [9] format, etc. -. The novelty introduced regards the frame format; as shown in Figure 4, when using eCACHACA, a node advertises also the measured value. This mechanism allows the gateway to have all the information required to publish sensor data to VITAL-OS.

Listing 1: PPI - Get system metadata.

```
{
  "@context": "http://vital-iot.eu/contexts/system.jsonld",
  "id": "http://example.com/rpi2_ppi",
  "type": "vital:VitalSystem",
  "name": "RPi2 PPI Implementation",
  "description": "VITAL compliant system to retrieve data coming from RPi2",
  "services": [
    "http://example.com/rpi2_ppi/observation"
  ],
  "sensors": [
    "http://example.com/rpi2-ppi/sensor/95",
    "http://example.com/rpi2-ppi/sensor/3",
    "http://example.com/rpi2-ppi/sensor/20",
    "http://example.com/rpi2-ppi/sensor/44"
  ]
}
```

Fig. 4: Frame format in eCACHACA.

B. The PPI to VITAL-OS

In order to be part of the VITAL-OS, the gateway should provide and expose a PPI implementation. Through that implementation, it is possible to retrieve:

- Information about the system (e.g., its status).
- Information about the services exposed and how to access them.

- Information about the sensors managed and what they observe.
- Observations made by the sensors.

According to VITAL specifications [10], the PPI is defined as a set of RESTful web services. Listing 1 shows an example of a response to the GET SYSTEM METADATA primitive, in which the gateway exposes information about the network (e.g., its type and description), as well as a list of the available services and the managed sensors.

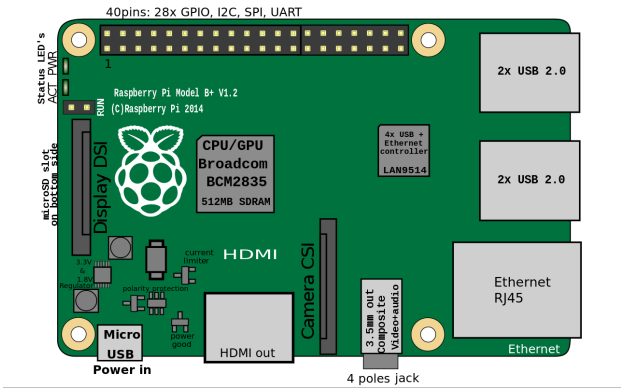


Fig. 5: Raspberry Pi 2 - Model B. (source http://wikimedia.org)

C. The gateway

The gateway represents the adapter between the VITAL-OS and the SN; it allows IoT Data Adapter to access data produced by the underlying SN and, at the same time, it acts as a sink for the sensors. In this work we used a Raspberry Pi 2 Model B (Figure 5), which has a quad-core ARM Cortex-A7 CPU, with 1 GB of RAM, and is equipped with 4 USB ports, 1 Ethernet port, and 40 GPIO pins. As operating system we adopted Raspbian [4], a free version of Debian optimized for the Raspberry Pi hardware. We chose this device because of its hardware capabilities and physical size.

The main tasks of the gateway are:

- Managing the sensor network.
- Formatting the information received from sensor nodes based on the VITAL ontology.
- Storing information into a local database.
- Exposing a PPI implementation via the web.

IV. PERFORMANCE EVALUATION

To evaluate the performance of eCACHACA, we ran experiments on FIT (Future Internet of Things) IoT-LAB - a platform suitable for testing small wireless sensor devices over large scales. This choice allowed us to experiment with a realistic environment -. We used the following metrics (measured at the gateway): *(I)* the number of neighbors discovered, *(II)* the time required, and *(III)* the Physical confidence. In order to study the behavior of the network, we varied the number of nodes from 10 to 90.

Figure 6a shows the percentage of NEIGHBORS DISCOVERED by the gateway as a function of the NUMBER OF NODES. With 10 nodes deployed, the gateway discovers all of them; while when we increase the number of nodes, this performance downgrades but it is still acceptable, indeed 80% of nodes is discovered.

The time required to discover those nodes is represented in Figure 6b. When the network is not dense (10 nodes), the discovery process is fast; all the nodes are discovered in less then 15 seconds. However, even when increasing the number of nodes, the gateway can discover them in less than 70 seconds.

The last analysis (Figure 6c) provides information about the PHYSICAL CONFIDENCE. When 10 nodes are deployed, the physical aspects measured at the gateway are more close to "Excellent". While, when increasing the number of nodes, the Physical confidence is "Good". This is due to the interference that possibly occurs when the network is more dense.

V. USING VITAL-OS

VITAL-OS provides a development tool that integrates all VITAL-OS functionality and makes them accessible to smart city application developers. The fact that all functionality are exposed via virtualised interfaces (i.e. RESTful web services) facilitates the task of integrating them into a single tool.

The development tool is based on Node-RED [5], an open-source tool for wiring the IoT that we have extended with nodes specific to the purposes and the needs of the VITAL-OS, as shown in Figure 7.

The development tool is used by developers that want to build and deploy smart city applications that involve multiple IoT platforms, architectures and business contexts. For that purpose, the tool enables them to access and compose the following VITAL-OS functionality:

- ICO and Service discovery.
- Complex Event Processing.
- Filtering.
- IoT system, ICO, data stream, security, configuration, SLA, and component management.
- Business Process Management.

The developers can even use the tool to directly access PPI primitive implementations, like the one we have provided for the SN.

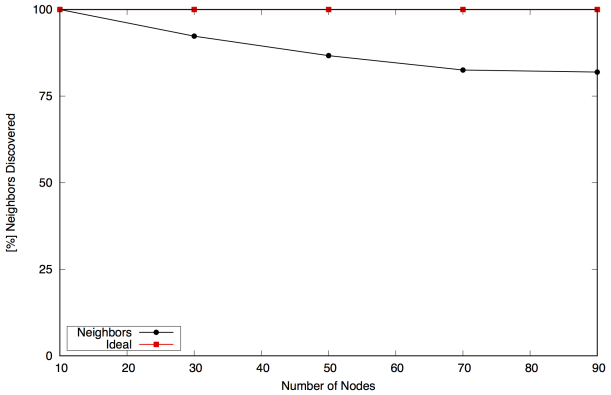
Based on the above and in the context of our example, a developer can use the VITAL-OS development tool to build a website where users will be able to find out how each sensor has classified its neighbors.

VI. RELATED WORK

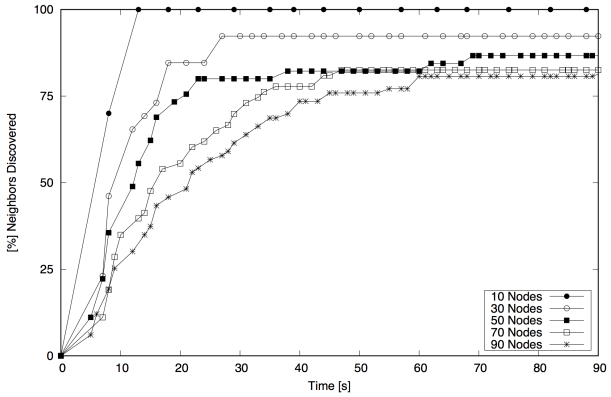
In the last decade, many middleware solutions have been proposed in literature with the objective to gather and integrate information from disparate data sources. In the following, we survey the most representative ones:

[4]https://www.raspbian.org

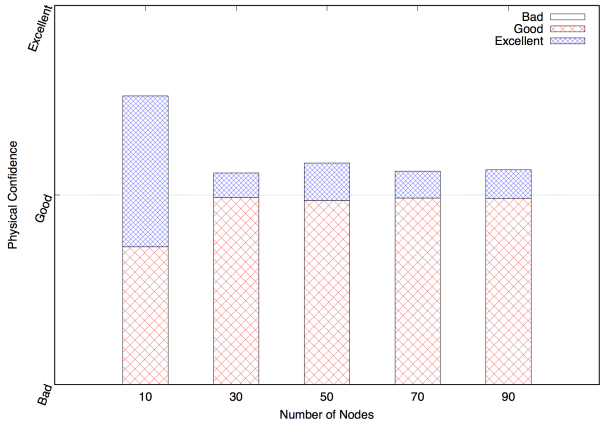
[5]http://nodered.org



(a) Number of Nodes vs. Neighbors Discovered.



(b) Neighbors Discovered - temporal trend.



(c) Number of Nodes vs. Physical Confidence.

Fig. 6: *e*CACHACA performance.

- *Xively* [6]. It offers a public cloud that simplifies and accelerates the creation, deployment and management of sensors in a scalable way. Anyway, it is not designed for a Smart City context, since it is possible to retrieve data just from own devices.
- *ThingSpeak* [7]. It is an open data platform that enables to collect, store, analyze, visualize, and act on data from sensors or actuators, such as Arduino, Raspberry Pi, etc. Anyways, it does not offer semantic interoperability among the different data sources.
- *OpenIoT* [8]. It provides a cloud-based middleware infrastructure capable to deliver on-demand access to IoT services. Nevertheless, it lacks of an ontology that describes city concepts.

The aforementioned approaches strengthen the importance of solutions based on cloud and semantic technologies; however, these initiatives are not yet ready to deal within a smart city context. The VITAL-OS, instead, is carefully designed to monitor, visualize, and control all the operations of a city.

[6]https://xively.com
[7]https://thingspeak.com
[8]https://github.com/OpenIotOrg/openiot

VII. Conclusions

There is nowadays, a real demand to make cities smarter in order to face challenges - i.e., waste management, traffic congestion, etc. - caused by the population growth. In this context, one key role is played by the Internet of Things and its data streams that can be converted into relevant information used to address the above issues. According with this vision, the number of IoT solutions is, nowadays, increasing, but on the other hand those initiatives are standalone and based on different protocols and standards. The VITAL-OS presented in this paper deals with this problematic, by introducing an abstract virtualized layer that operates across multiple IoT architectures and platforms. This layer represents the end-point thanks to which it is possible to monitor, visualize, and control all the operations of a city.

In order to validate the VITAL-OS architecture, we presented a practical example of monitoring in a Smart Building environment. We used a gateway that communicates, from one side, with the IoT Data Adapter of the VITAL-OS, and on the other, acts as a sink for the SN. The firmware of the nodes is flashed with *e*CACHACA, an algorithm that enables the discovery and the ranking of the neighbors and the available

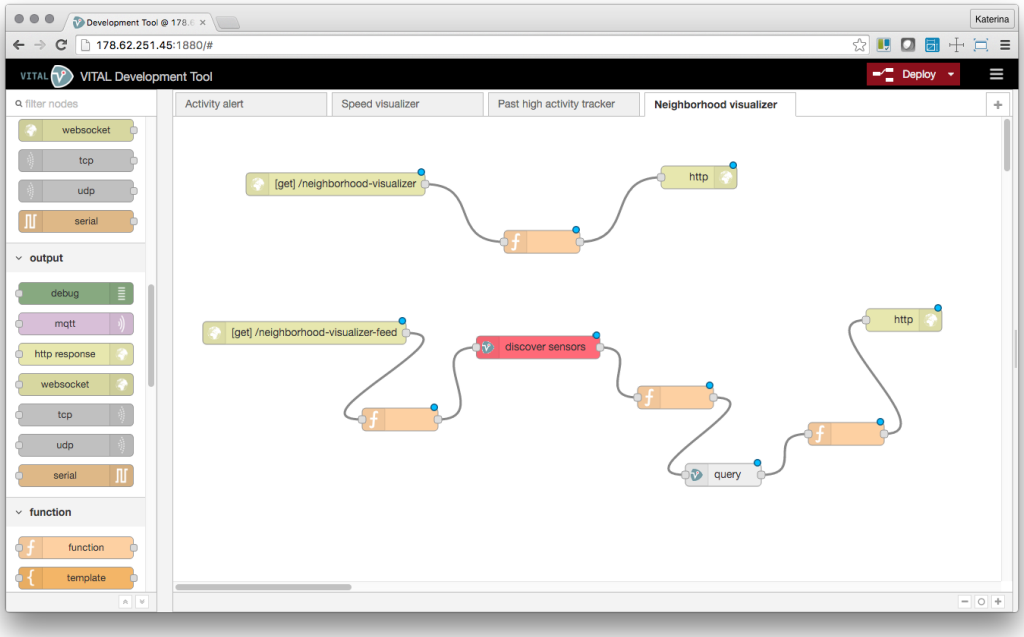


Fig. 7: The VITAL-OS development tool.

services. Performance has been evaluated via experimentation on the FIT IoT-LAB, and results demonstrate the effectiveness in the discovery of resources. Indeed, even when the network is highly dense, the gateway discovers 80% of the available nodes in less then 70 seconds.

In the future, we plan to improve the performance of our algorithm by introducing more sophisticated criteria for the evaluation of the neighborhood. At the same time, we also aim to analyze *e*CACHACA in more complex use cases in which nodes are mobile.

Acknowledgments

This work is partially supported by CPER Nord-Pas-de-Calais/FEDER DATA and by the European Community in the framework of the VITAL (Virtualized programmable InTerfAces for innovative cost-effective IoT depLoyments in smart cities) FP7 project under contract number FP7-SMARTCITIES-608662. The authors acknowledge help and contributions from all partners of the project.

References

[1] D. Evans, "The Internet of Things - How the Next Evolution of the Internet is Changing Everything," *CISCO white paper*, pp. 1–11, 2011.

[2] O. Vermesan, P. Friess, P. Guillemin, S. Gusmeroli, S. Harald, A. Bassi, I. S. Jubert, M. Mazura, M. Harrison, M. Eisenhauer, and P. Doody, "Internet of Things Strategic Research Roadmap," in *IoT European Research Cluster*, 2011, pp. 9–52.

[3] Libelium, "50 Sensor Applications for a Smarter World," http://www.libelium.com/top_50_iot_sensor_applications_ranking.

[4] R. Petrolo, V. Loscri, and N. Mitton, "Towards a smart city based on cloud of things, a survey on the smart city vision and paradigms," *Transactions on Emerging Telecommunications Technologies*, pp. 1–11, 2015.

[5] M. Serrano, A. Kazmi, E. Dilek, Y. Yaslan, S. Oktug, J. Soldatos, and A. Lennis, "The future of smart cities: A practical case of connecting cities with vital-os," in *Proceedings of IEEE/IFIP PACS - International workshop on Platforms and Applications for Smart Cities in conjuction with Network Operations and Management Symposium (NOMS)*, Istanbul, Turkey, Apr. 2016.

[6] R. Petrolo, S. Guzzo Bonifacio, V. Loscri, and N. Mitton, "The discovery of "relevant" data-sources in a Smart City environment," in *Proceedings of SSC - 2nd International IEEE SMARTCOMP Workshop on Sensors and Smart Cities*, St. Louis, Missouri, United States, May 2016.

[7] A. Roukounaki, J. Soldatos, R. Petrolo, V. Loscri, N. Mitton, and M. Serrano, "Visual Development Environment for Semantically Interoperable Smart Cities Applications," in *Proceedings of EAI International Conference on Interoperability in IoT*, Rome, Italy, Oct. 2015.

[8] R. Petrolo, V. Loscri, and N. Mitton, "Confident-based Adaptable Connected objects discovery to HArmonize smart City Applications," in *Proceedings of IFIP WD - Wireless Days*, Toulouse, France, Mar. 2016.

[9] Z. Shelby and C. Chauvenet, "The IPSO Application Framework draft-ipso-app-framework-04," Tech. Rep., 2012. [Online]. Available: http://www.ipso-alliance.org/wp-content/media/draft-ipso-app-framework-04.pdf

[10] J. Soldatos and K. Roukounaki, "Specification and Implementation of Virtualized Unified Access Interfaces V2," 2015, VITAL Consortium, Deliverable D3.2.2. Available from http://vital-iot.eu [accessed 14 March 2016].