Efficient Data Management in the context of the Internet of Things

Dimitrios Amaxilatis

Supervisor: Professor Christos Zaroliagis

Department of Computer Engineering and Informatics
University of Patras

University of Patras October 2018

University of Patras,
Computer Technology and Engineering Department

Dimitrios Amaxilatis October 2018

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Professor Dr. Christos Zaroliagis for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. My sincere thanks also goes my co-advisor, Professor Dr. Ioannis Chatzigiannakis, as without his precious support it would not be possible to conduct this research. I also want to thank Dr. Georgios Mylonas for his guidance in all the time of research and writing of this thesis. I could not have imagined having better advisors and mentors for my Ph.D study.

I also thank my fellow labmates, Lidia Pocero Fraile, Orestis Akrivopoulos and Marios Logaras for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last four years.

Last but not least, I would like to thank my family for supporting me spiritually throughout writing this thesis and my life in general.

Abstract

Technological developments in recent years in both hardware and software have led to an explosion of devices and services in what is often called the Internet of Things. Today in every home in the US there are 11 "smart" devices with sensors capable of accurately describing and watching their environment. This trend is expected to continue further in the future, with this figure rising to 500 in 2022, sharply increasing the volume of information that can be exported from human activity. This plethora of data creates enormous potential for developing applications and services for users to improve their life parameters (from personal comfort to health or transport). The secure and effective processing of these data is a major problem, for which no specific solution has been widely adopted so far to address the significant problems of personal data protection and the efficient management of such data. Technologies such as IPv6 promote cloud management by directly exposing all devices over the Internet and thus allowing easy interaction and remote control, as opposed to Bluetooth LE, where local device management is the only available solution and selected information is published on the Internet using gateway devices. Finally, it is particularly important to enrich the data with external data such as meteorological forecasts or data on traffic and public transport routes.

In this thesis, we are studying the real-time monitoring and organization of sensing infrastructures according to the changing requirements of applications and users. The aim is to design, implement and evaluate experimental self-organizing mechanisms using semantic information to improve the quality of data flows provided at the Internet level. As part of this process, we also seek to combine flows to more efficiently share and manage information. The mechanisms we implement are based on the "semantic entities" model, either as a part of the device network or as part of a web service aiming at balancing computing and storage requirements at the various levels of the network hierarchy. The goal is also to study new data processing and tracking techniques to draw appropriate conclusions, predictions and decisions.

Περίληψη

Οι τεχνολογικές εξελίξεις των τελευταίων ετών τόσο σε επίπεδο υλικού όσο και σε επίπεδο λογισμικού έχουν οδηγήσει σε μια έκρηξη συσκευών και υπηρεσιών στον χώρο που αποκαλείται συχνά Διαδίκτυο των Πραγμάτων. Σήμερα σε κάθε σπίτι στις Η.Π.Α. υπολογίζεται ότι υπάρχουν 11 'έξυπνες' συσκευές με αισθητήρες ικανούς να περιγράψουν και να παρακολουθήσουν με σημαντική ακρίβεια το περιβάλλον τους. Η τάση αυτή υπολογίζεται πως θα συνεχιστεί ακόμη περισσότερο στο μέλλον με τον αριθμό αυτό να ανέρχεται σε 500 το 2022 αυξάνοντας κατακόρυφα τον όγκο πληροφοριών που μπορεί να εξαχθεί από κάθε ανθρώπινη δραστηριότητα . Αυτή η πληθώρα δεδομένων παρέχει τεράστιες δυνατότητες για την ανάπτυξη εφαρμογών και υπηρεσιών προς τους χρήστες για την βελτίωση παραμέτρων της ζωής τους (από τον τομέα της προσωπικής άνεσης μέχρι την υγεία ή τις μεταφορές). Η ασφαλής και αποτελεσματική επεξεργασία των δεδομένων αυτών είναι ένα σημαντικό πρόβλημα για το οποίο δεν έχει μέχρι σήμερα προκριθεί μια συγκεκριμένη λύση που να απαντά στα σημαντικά προβλήματα της προστασίας των προσωπικών δεδομένων και της αποτελεσματικής διαχείρισης των δεδομένων αυτών. Τεχνολογίες όπως το IPv6 προωθούν την διαχείριση στο επίπεδο του ςλουδ με την άμεση επικοινωνία όλων των συσκευών με το διαδίκτυο και των απομακρυσμένο τους έλεγχο σε αντίθεση το Bluetooth LE όπου επιλέγεται η διαχείριση των συσκευών σε τοπικό επίπεδο και η δημοσίευση μόνο επιλεγμένων πληροφοριών στο διαδίκτυο. Τέλος, ιδιαίτερα σημαντικός είναι ο εμπλουτισμός των δεδομένων με εξωτερικά δεδομένα όπως μετεωρολογικές προβλέψεις ή δεδομένα για των κίνηση και τα δρομολόγια των μέσων μαζικής μεταφοράς.

Στην παρούσα διατριβή μελετάμε την σε πραγματικό χρόνο παρακολούθηση και οργάνωση δικτύων αισθητήρων ανάλογα με τις μεταβαλλόμενες απαιτήσεις εφαρμογών και χρηστών. Στόχος είναι ο σχεδιασμός, υλοποίησης και αξιολόγηση με πειραματικούς μηχανισμούς αυτο-οργάνωσης με χρήση της σημασιολογικής πληροφορίας για την βελτίωση της ποιότητας των παρεχόμενων ροών δεδομένων στο επίπεδο του Διαδικτύου. Στα πλαίσια αυτής της διαδικασίας επιδιώκουμε και τον συνδυασμό ροών για τον αποδοτικότερο διαμοιρασμό και διαχείριση των πληροφοριών. Οι μηχανισμοί που υλοποιούμε βασίζονται στο μοντέλο των 'σημασιολογικών οντοτήτων', είτε ως λει-

τουργικό κομμάτι του δικτύου των συσκευών είτε ως μέρος διαδικτυακών υπηρεσιών στοχεύοντας στην εξισορρόπηση των υπολογιστικών και αποθηκευτικών απαιτήσεων στα διάφορα επίπεδα ιεραρχίας του δικτύου. Επίσης, στόχος είναι να μελετήσουμε νέες τεχνικές επεξεργασίας και παρακολούθησης των τελικών δεδομένων για την εξαγωγή κατάλληλων συμπερασμάτων, προβλέψεων και αποφάσεων.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Aspirations

Starting from Wireless Sensor Networks (WSN) and Future Internet (FI) solutions, a set of device and service-based solutions have emerged in the past years to form what is known as the Internet of Things (IoT). A Wireless sensor network (Figure 1.1) refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location. Typically WSNs monitored environmental conditions like temperature, sound, pollution levels, humidity, wind, and so on in hard to reach environments where wired installations were not feasible. Future Internet refers to the research topics that intend to resolve the shortcomings of current Internet related technologies in terms of performance, reliability, scalability, security and other aspects. The term "Internet of things" was coined by Kevin Ashton of Procter

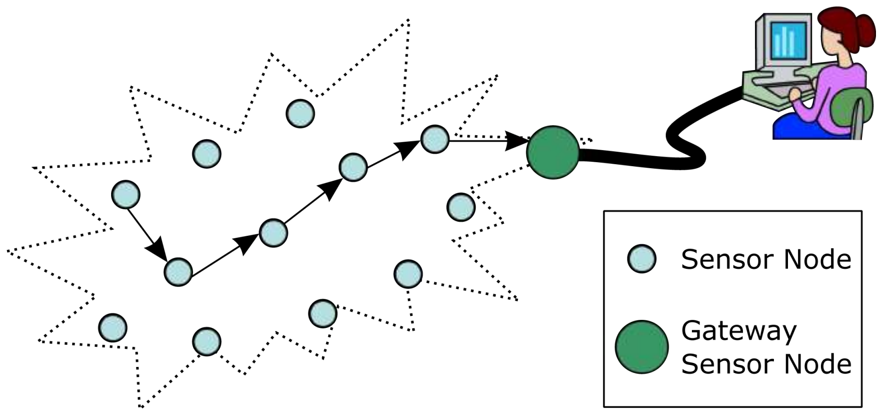


Figure 1.1 An example of a Wireless Sensor Network (source: wikipedia)

& Gamble, later MIT's Auto-ID Center, in 1999 [16]. The Internet of Things can be defined as the network of physical devices embedded with electronics, software, sensors, actuators and connectivity which enables them to exchange data with each other and the Internet using their unique identifiers [41]. Such an infrastructure allows the development of applications and services that facilitate the lives of citizens and workers around the world and bring us closer to the convergence of the physical and digital worlds (Figure 1.2). These applications have provided us with solutions in fields like smart-grids, smart-homes, intelligent transportation or smart-cities. Some successful examples include, amongst others, Nest and Ecobee for smart-heating, Phillips Hue for smart-lighting, Smarthings as a provider for multiple smart-home solutions, Fitbit and Withings for personal fitness trackers, as well as the personal assistants like Alexa from Amazon or Siri from Apple.

Figure 1.2 A smart and connected home

Despite the huge growth of such IoT applications, the usefulness of the data that originate from these systems is still to be validated and proved. The volume of data that can be generated from a single sensor device installed in a house or a wearable device worn by a person can be overwhelming for the device itself and in most cases needs to be offloaded to a data-processing application from which users can access it and connect it to a more meaningful use [19]. Taking into account the rate with which such devices are being integrated into our everyday lives, the result of offloading so much information in the cloud can create huge volumes of data. Such data need to be communicated, processed and stored in real time, while accessing it in its raw format tends to be useless when compared with the knowledge it can generate (e.g., information about a person's lifestyle and diet compared with their daily step count and trips as latitude-longitude coordinates). As a result, the need for a holistic solution on the subject of managing the data generated is still an open subject for research in both academic and enterprise scopes. Similarly, multiple interfaces for communicating and representing such data formats have been used, with none getting a clear step ahead of the others in the path for a global standard, if and when this will be possible.

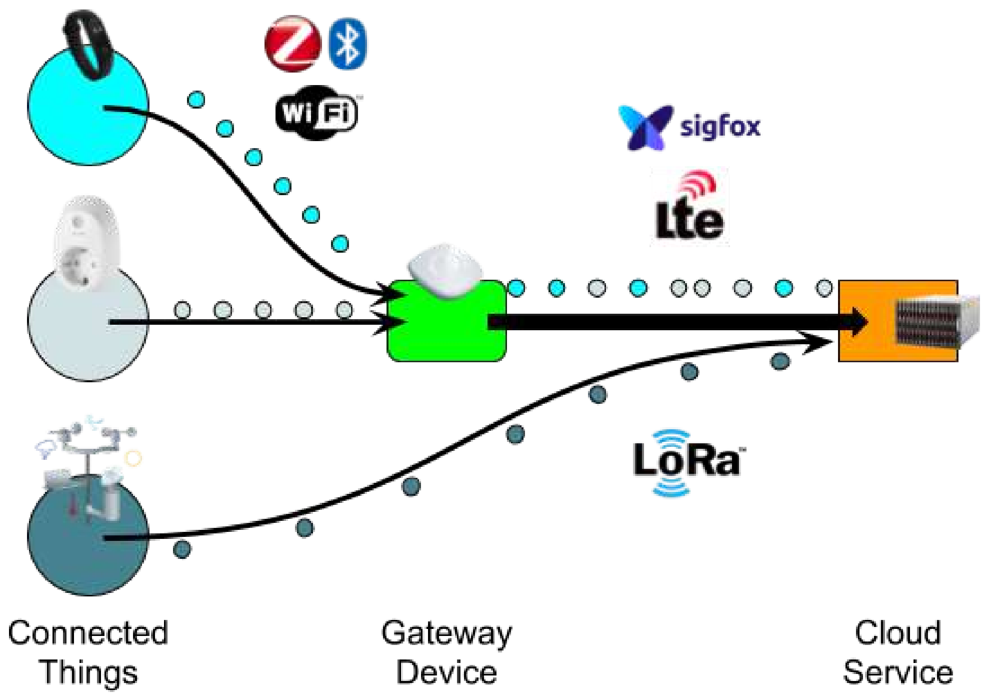


Figure 1.3 Data Streams of the Internet of Things

Another important characteristic of the IoT domain is the huge diversity of the communication mediums, protocols and data representation and exchange formats used. Figure 1.3 showcases how IoT devices communicate the collected data with cloud services using multiple technologies and protocols for local-only or local-to-cloud communications. Each protocol offers different capabilities in terms of throughput, communication range or power consumption, based on the needs for the application use case. For example, communication protocols with limited range like Zigbee[6], Bluetooth LE[34] or WiFi [4] are used to communicate data between wearables or smart-home devices and local gateways. Zigbee and Bluetooth LE are better suited for devices that operate on batteries and exchange limited amounts of data (e.g., wearable fitness trackers or beacons). On the other hand, WiFi is more suited for devices that need more real time communication and can exchange larger chunks of information like smart-plugs or cameras. Such devices are mostly connected to wall outlets as WiFi needs significantly more power and using batteries is not feasible on a long-term basis. On the other end, to communicate information with the Internet, gateways or smart-devices use technologies like xDSL [76] lines, LTE [1] or LoRa [5] and Sigfox [67]. xDSL and LTE offer a higher throughput when compared to LoRa and Sigfox but have a higher installation and operational cost as they require on premises infrastructures, maintenance and recurring costs either as monthly or pay-as-you-go data plans. LoRa and Sigfox are simpler solutions, with no installations on the premises of the user, other than the device itself, and minimal (i.e., a yearly subscription for Sigfox) or no costs (LoRa) but offer significantly reduced throughput.

In all the above cases, the characteristics of the communication module used defines the sampling rate that can be achieved by the sensing modules of the device. A general rule followed in most applications or devices currently available is that devices try to sample and transmit data as fast as possible while maintaining a decent amount of battery life. This is done to ensure that the phenomena monitored are well observed and no desired events are missed due to devices sleeping to conserve power. This behaviour leads to an ever increasing load of data that needs to be processed in order to extract useful information and conclusions.

This fragmentation in the IoT domain presents the main challenge of integration and interoperability to achieve the desired end result of a unified virtual environment. For instance, a data analytics solution needs to ingest data streams from multiple of the sources presented above. Mixing different hardware and technologies makes this much more complex in order to finally make every device or system “talk to” and “understand” each other. Finally, there are all the cyber-security and cyber-pivacy issues for not only wireless communication, but also for the control and operation of the connected assets.

From the user’s or customer’s perspective, it is a costly and time-consuming experience to become familiar and maintain multiple solutions from different vendors, as well as integrating them all together in a single system. Industry standards are being developed (i.e. as wireless protocols), and continue to evolve through open-source frameworks that encourage collaborations among solution developers but global acceptance is hard to achieve due to contradicting interests from the involved partners.

1.2 Goals

In order to provide such a holistic solution it is important to be able to integrate the data that are generated by all of these smart devices in a common base. We need to be able to process them, view them and operate on them using a common methodology and also be able to gain common knowledge from them. This is the most important step towards the unification of the information originating from a fragmented ecosystem that can lead to the development of next-level applications and services with significantly reduced development effort. In this direction and in the context of this dissertation, we study the possible solutions to the aforementioned problems in an environment filled with IoT devices. We focus our work in the following fields:

- The representation of the IoT environment together with its metadata and semantics in a suitable representation format capable of handling the complex relationships that are generated in such a densely populated environment.
- The efficient collection and processing of streaming IoT data, in real time with the minimum processing time and latency, using a modular system that can be extended to support additional data types and devices with limited interventions.
- The extraction of knowledge from the collected and processed IoT data. Raw data are of limited value to end users in most cases. The real value lies on the conclusions that can be extracted from them.

Once we have achieved all 3 goals, we will be able to analyze IoT data in real time offering the outcomes of our work to developers that can then build upon our work and generate a truly intelligent and connected environment.

Generating a common domain where data, metadata and semantics can be described is a key step towards this result, mainly due to the variety of representation formats currently used. Some of these formats have been used and tested for quite some time (e.g., XML), others are too strict and hard to be understood by end users (e.g., RDF) while some are more flexible (e.g., JSON) and decentralized, but this flexibility leads to diverse implementations that are not always compatible. Building on top of this common domain, we need to implement a common processing flow that can be used over all the data generated. For this part, time-based analysis is the most prevalent solution and one that better syncs with the human lifestyles; this is the direction that we will follow as well. Finally, using these two prerequisites, we can then build up a system that can offer additional and more meaningful information on the data received. We can build a system that uses elements like Machine Learning to generate new and combined observations on the physical world, bridging inputs from multiple IoT devices and describing our surroundings in a much more accurate way.

1.3 Contributions

Based on the goals set in Section 1.2 in this dissertation we contribute to the field of data analytics for the Internet of Things in smart environments and the post processing analysis of the IoT data to provide a better understanding of the physical surroundings. We propose new and extensible methods to store the information of IoT installations and our findings allow for fast look-ups across the whole set of IoT devices in the network. Processing the data generated in real-time becomes a more streamlined work flow, using a standardized methodology that allows for changing the data analysis from a single point. Data generated can be easily stored and retrieved from a specifically designed, reliable and efficient storage engine. Using the original and calculated data in a feedback loop also allows us to build additional processing and analysis layers that generate more data and information resulting in more accurate outcomes.

The results of our work can be categorized in the following points:

1. The first goal of our work was to setup a base set of infrastructures for bringing the physical world with its digital representations. We wanted to be able to understand what is where and how it interacts with its environment as well as what it observes in real time. Our work towards this direction was twofold:

   (a) We have developed a graph-based schema to represent IoT installations together with their semantics and meta information. The schema is not based on the traditional relational databases but builds on the ideas of graph theory and utilizes a new database model, the a graph database. Each entity that participates in real world interactions or can be observed by an IoT device is represented as a nodes of a graph, while the interactions themselves are the vertices of the graph. We therefore build a web of entities and relations that are easy to visualize and traverse to find answers to various queries that may arise inside a smart environment. Such queries can look for the the causes of events observed (e.g., what caused the rise of luminosity in room), the available information for an area (e.g., what information is sensed for the building) or even the social interactions and connections of individuals (e.g., which people use the same appliances).

(b) For the analysis of the data we provide a template implementation for setting up a system that can receive, process and analyze an unbounded number of input data streams with no impact on its operation. Such data streams can originate from sources that range from single smartphones to city wide sensor installations. The system is also capable of handling data streams that provide unbalanced volumes of information with no performance drops. The calculation methodology itself is developed as to be easily customized and extended, based on the data types and the calculations required in every environment it is deployed to. This processing engine is capable of identifying the data from each of the sensing devices routing them to the appropriate processor without any prior per-device configuration.

Both the database and processing engine presented in (a) and (b) are used to support a publicly available system that supports over 300 users and consumes data from more than 20 IoT installations with almost 5000 sensing points. Based on our evaluation the database model can easily respond to all queries in real time, even in use cases where the database ranges to tens or even hundreds of thousands of entities. We also performed an extensive evaluation of the processing engine in real world conditions for more than 2 years achieving sub-millisecond processing times per measurement. We stressed our system with data volumes that exceed by far the data collection rates of our real world installation to prove that it is future-proof and scalable. This work is presented in [7, 9, 10, 51, 52].

2. To better understand the collection mechanisms of data from more distributed and uncontrolled IoT installations we also developed a solution that can collect data using smartphones in smart cites carried by volunteers in the context of crowd-sourcing campaigns. This method is used to augment smart-city installations with mobile and inexpensive infrastructures using the power of volunteers local activist groups. Such solutions can be deployed with minimal cost by the officials of a city as a new service that can provide them with important insights for the local conditions, environmental, social or other. This

smartphone application was used in more than 10 experimentation campaigns in cities around Europe (including London in the UK, Santander in Spain, Patras in Greece and Aarhus in Denmark) by more than 50 users over the past 2 years. Results for this work are presented in [8, 11, 12, 36, 37].

3. Finally, we also present a framework for characterizing the data from all the above inputs using Machine Learning in real time by taking advantage of the streaming data mechanism developed in the data processing engine described above. This framework was used to analyze real world data from smart-city IoT infrastructures as well as the data that originate from smaller scale IoT installations. This work is presented in [26].

The outcomes of this dissertation mentioned in this chapter are used in the heart of two EU-funded Horizon 2020 research projects: GAIA[1] and OrganiCity[2] that focus on energy efficiency and smart cities respectively.

- GAIA aims to promote positive behavioural changes within communities regarding energy consumption/awareness using the gamification of real-time, IoT-enhanced energy consumption metrics in trial schools located in Italy, Greece and Sweden.

- OrganiCity is a service for experimentation that explores how citizens, businesses and city authorities can work together to create digital solutions to urban challenges using a set of tools experimenters can use to test and develop their own ideas into viable smart-city application.

1.4 Thesis Organization

The rest of the dissertation is organized as follows:

In Chapter 2, we start by presenting state-of-the-art work in the field of IoT applications, data analysis and data processing. We also present a selected list of research applications that leverage IoT technologies in the areas of Smart Buildings and Smart Cities focusing on data collection, as well as the challenges in analyzing and validating such data.

In Chapter 3, we showcase our solution in the context of representing the IoT ecosystem in the digital world. We present a graph-based storage engine that

[1]http://gaia-project.eu
[2]http://organicity.eu/

can store all the information related to an IoT installation. The data are easily queried through the database's interface and an appropriate programming interface. Chapter 4 builds upon this data storage and contains our design of a streaming data analytics engine designed for the IoT ecosystem. We present the data that can be introduced in our system in Section 4.1, the architecture of the system in Section 4.2 and the customizable modules that process the data based on their metadata presented in the previous chapter in Section 4.2.2. Our work is applied and tested against the data originating from a fleet of buildings. Finally, in Section 4.3, we showcase the evaluation of the system during its real world operation over a period of more that 2 years and a selected set of observations to point out and prove its efficiency.

Chapter 5 focuses on how the aforementioned technologies can be used when the IoT installation scales from a small number sensors inside one building to one or even multiple cities. We present this case of a large scale installation, where the sensor devices are no longer controlled by a single entity or organization but include data from multiple, even untrusted sources or entities raising concerns about their quality, validity or trustworthiness. In this context, we present in Section 5.2 a framework that can be used to crowdsource data using volunteers in a city and how we can validate and annotate them in an automated manner in Section 5.3. Section 5.2.1 presents the evaluation of the crowdsourcing platform in a real world environment as part of an EU research project. Similarly Section 5.3.3 presents a number of evaluation applications that showcase how the implemented solution can provide annotations over the collected data originating from a two different IoT installations. Both applications are tested in the context of OrganiCity, an EU funded project that explores how citizens, businesses and city authorities can work together to create digital solutions to city challenges.

Finally, in Chapter 6 we present conclusions from the work performed in all the above scenaria, lessons learned and possible extensions that need to be investigated in the future.

Chapter 2

Background and Related Work

In this chapter we aim to provide In this chapter present basic concepts and definitions that we will use in the rest of this dissertation. We define what is described as Internet of Things, its characteristics and base application domains together with main usage examples that provide us with incentives for building such applications.

2.1 The Internet of Things

The **Internet of Things** (IoT) is a system of interrelated computing devices, analog and digital machines, objects, animals or people that are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. A **thing**, in the Internet of Things, can be a person with a heart monitor implant, a farm animal with a biochip transponder, an automobile that has built-in sensors to alert the driver when tire pressure is low – or any other natural or man-made object that can be assigned an IP address and provided with the ability to transfer data over a network. Experts estimate that the IoT will consist of about 17.6 billion objects by 2020, a number far less than the original predictions of 1 trillion or 50 billion devices, but still quite high[1] (when compared to the number of devices currently connected to the Internet). It is also estimated that the global market value of IoT will reach $7.1 trillion by 2020 [40].

The IoT allows objects to be sensed or controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and

[1]https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated

economic benefit in addition to reduced human intervention. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, virtual power plants, smart homes, intelligent transportation and smart cities.

2.1.1 Streaming IoT Data

The data generated in the context of the IoT do not follow the typical characteristics for data processing technologies and methodologies used in past computer science use cases as their volume usually exceeds the size a single machine can process in real time. The first attempts to harness such data were made using big data technologies like Hadoop [75]. Such systems use a set of processes that can be run in rounds one after the other and reduce the actual volume of data available to a small and usable information, and were capable of overcoming the original problems encountered in IoT systems. However, they lack a viable solution in the most recent use cases of IoT, where the actual data are not a priori available to the system that will process them. Such data are called **streaming** and are becoming a more and more common example of how the information from IoT installations will arrive to the processing system. Streaming Data is data that are generated continuously by thousands of data sources, which typically send in the data records simultaneously, and in small sizes (order of Kilobytes). Streaming data includes a wide variety of data such as log files generated by customers using your mobile or web applications, ecommerce purchases, in-game player activity, information from social networks, financial trading floors, or geospatial services, and telemetry from connected devices or instrumentation in data centers.

This data needs to be processed sequentially and incrementally on a record-by-record basis or over sliding time windows, and used for a wide variety of analytics including correlations, aggregations, filtering, and sampling. Information derived from such analysis gives companies insights into many aspects of their business and customer activity such as service usage (for metering/billing), server activity, website clicks, and geo-location of devices, people, and physical goods –and enables them to respond promptly to time-critical situations. For example, businesses can track changes in public sentiment on their brands and products by continuously analyzing social media streams, and respond in a timely fashion as the necessity arises.

Streaming data processing is beneficial in most scenarios where new, dynamic data is generated on a continual basis. It applies to most of the industry segments and big data use cases. Companies generally begin with simple applications such as collecting system logs and rudimentary processing like rolling min-max computations. Then, these applications evolve to more sophisticated near-real-time processing. Initially, applications may process data streams to produce simple reports, and perform simple actions in response, such as emitting alarms when key measures exceed certain thresholds. Eventually, those applications perform more sophisticated forms of data analysis, like applying machine learning algorithms, and extract deeper insights from the data. Over time, complex, stream and event processing algorithms, like decaying time windows to find the most recent popular movies, are applied, further enriching the insights.

When comparing such streaming data with batch processing, we need to note that the latter can be used to compute arbitrary queries over different sets of data. It usually computes results that are derived from all the data it encompasses, and enables deep analysis of big data sets. MapReduce-based systems, like Hadoop, are ideal for such batch jobs. In contrast, stream processing requires ingesting a sequence of data, and incrementally updating metrics, reports, and **summary statistics** in response to each arriving data record. It is better suited to real-time monitoring and response functions.

For managing large data streams produced by the Internet of Things, several research prototype implementations and some more commercially-oriented solutions have been presented. There is a large set of requirements, like publishing/subscribing on data streams, interfacing with various technologies and performing real-time analysis that makes it crucial to build systems extending a lot of classical RDBMS systems. One of the fist options made available in this area was Xively[77] (previously known as Cosm or Pachube). It is a secure, scalable platform that connects devices and products with applications to provide real-time control and data storage. Other enterprises, that focused more on offering computing services, like Amazon and Microsoft also offer solutions for IoT data processing with AWS IoT [14] and Azure IoT Suite [50]. Both systems focus on helping device manufacturers collect and send data to the cloud, while making it easy to load and analyze them.

At the core of all similar solutions lies the concept of message exchanges using normally a central service that plays the role of a **Message Broker**. A Message Broker, in the general case, maintains a list of available topics on which client applications can publish updates while other application can subscribe for updates. This design concept allows the development of asynchronous system designs that can be developed independently without any restriction on the hardware or software used.

All data exchanges are done over a set of well-defined interfaces that can be available simultaneously on the same system. Typical examples of such message brokers are Apache Kafka, RabbitMQ, Mosquitto or VerneMQ that implement one or multiple messaging protocols like ActiveMQ or MQTT, which offer similar functionality with trade-offs in features based on the requirements from the application in question.

2.2 IoT applications

2.2.1 Smart Home

Smart Home is one of the most discussed and trending terms in relation to the IoT domain in web searches according to Google Trends as presented in Figure 2.1, reaching “Big Data” in popularity. It refers to the introduction of multiple IoT devices in a residential or office building that extends the boundaries of the traditional Building Management solutions and offers more personalized information on the operation and the parameters of the building itself, as well as its occupants. Recent publications like [46] identity the essential technologies for successful IoT solutions as radio frequency identification, wireless sensor networks, middleware, cloud computing and software for application development. They also identify the enterprise applications sectors in monitoring and control, big data and business analytics, and information sharing and collaboration. [47] presents the solution to developing such systems in the use of a middleware that can envelop older and new components to support common naming, addressing, storage and look-up services. Some of the work available [66], differentiates on the physical location where the analysis of the data takes place, either in house or on centralized services.

In the general case of a **Smart Home**, household devices become equipped with interfaces for wireless communication, forming up a home network. Each home has one or more such networks, and the sensed data from each device are forwarded to central stations, which can be referred as gateways or smart hubs. Nodes of this network are device that have moderate or limited computational power and communication capabilities. The gateway is the one device that has some additional data storage capacity and can perform local processing or simply communicate with devices in the outside world. In the case of larger smart buildings, the gateways may have additional capabilities based on the amount of devices that depend on them. Gateways also play an important role in achieving interoperability across devices of multiple technologies and communication protocols.

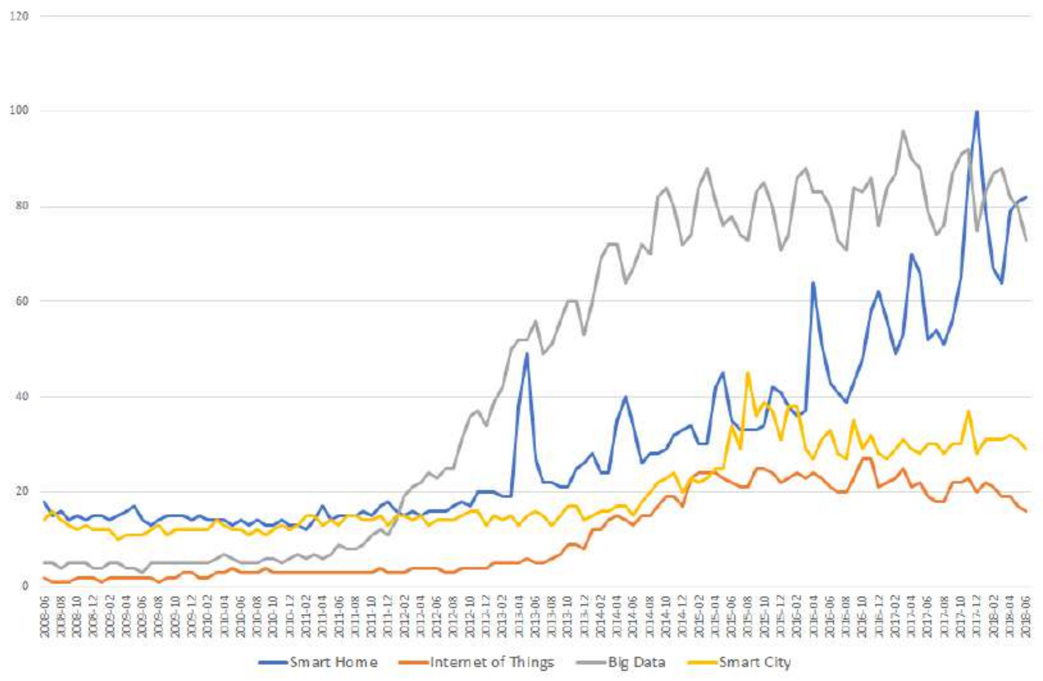


Figure 2.1 Google trends data on Smart Homes, Smart Cities, Big Data and Internet of Things.

2.2.2 Smart Cities

Recent studies [17] have predicted that, by 2050, 70% of the world population will live in urban areas, while more than half of the world's population already lives in cities. In this context, different stakeholders (city planners, politicians, researchers, etc.) implement policies that aim to improve the quality of life in urban environments, also developing initiatives contributing to more efficient and sustainable cities. The fact that cities represent a strategic meeting point between citizens and technology provides an additional dimension that can be exploited for a collaborative and continuous crowd-sourced creativity. This is what we categorize as societal innovation: human beings are immersed in a context which, based on IoT technologies, stimulates the conception of new ideas and solutions addressing the problems that are related to cities. To facilitate the adoption of such solutions, IoT experimentation under real world conditions is crucial to their validation.

In recent years, experimentation with Future Internet (FI) technologies has been led by commercial companies and research centers, with less involvement from external stakeholders like citizens or decision makers. Due to the slow uptake, it is evident that a more holistic approach is needed, where all the relevant actors are involved in order to produce more useful and engaging applications. In this sense, IoT technologies are contributing to the creation of innovation ecosystems, where the FI provides an opportunity to the research community to modernize the existing solutions and adopt new ones, validated through the involvement of multiple stakeholders. Deployment, testing and evaluating solutions in cities under real conditions enables the possibility to conceive new or improve existing urban city services, such as waste or transportation management. Furthermore, the access to the vast amounts of urban data, generated from numerous sources, enables the design, implementation and ability to assess new techniques and algorithms that have the potential to outperform existing ones.

Recently, experimentation with IoT-related technologies in urban environments has attracted quite a lot of attention, especially in Europe. In this context, many research projects have been conceived to experiment with large scale infrastructures, developing different pilots that evaluate the proposed use-cases in the urban landscape. In such initiatives, the reader can find how different smart-city applications, outdoor deployments, and indoor installations in buildings targeting energy efficiency have been carried out. SmartSantander [62] is probably a characteristic example of how a massive deployment of IoT devices is used within an IoT facility to allow experimenters to conduct research using innovative solutions in the city context. The experimentation of smart city architectures, services and applications in real-world urban environments has essentially been pioneered, deployed over a very large-scale IoT infrastructure in the city-center of Santander.

2.2.3 Crowdsourcing and Experimentation-as-a-Service

Projects like IoT-Lab investigate crowdsourcing and IoT services for supporting multi-disciplinary research tasks [42]. However, relatively little attention has been given to combining both an urban IoT infrastructure, comprising stationary and mobile IoT nodes, with a crowd-sensing component utilizing “transient” IoT resources contributed by citizens. The area of participatory sensing using crowdsourcing and harnessing ubiquitous technologies is discussed extensively in [35], where authors provide both the theoretical background and a review of a number of approaches currently utilized. However, although a number of technical advancements were made in these projects with respect to making such IoT infrastructures available to the research community and the industrial sector, there is still a lot of issues that need to be solved in a more coherent and holistic manner. The most relevant one can be considered as how to empower citizens or other stakeholders to participate in the societal change of the smart city innovation ecosystems. In this sense, the Experimentation-as-a-Service

(EaaS) paradigm has been conceived to achieve this goal, although depending on the project, it has been implemented using different approaches. In [63], EaaS is a federated platform that provides reconfigurable on-demand access to a set of resources, allowing researchers to rapidly deploy experiments based on services belonging to different smart city domains. Although the user stands as the key actor of the experimentation, an integral framework and tools have not been defined allowing them to be part of the co-creation process. On the other hand, in [45] the EaaS concept is based on the creation of an Application Program Interface (API) that enables executing experiments over multiple existing IoT testbeds. In this case, the EaaS uses semantic-based technologies to provide an agnostic layer that enables federating the several IoT experimentation facilities. However, they have not designed the mechanisms to actively engage citizens in both defining application scenarios and participating in their conception, usage and therefore validation. The implementation of EaaS solutions has recently grown in relevance, catching the attention of large companies and organizations. As an example, IBM (Armonk, NY, USA) provides an EaaS cloud-based platform that enables users to demonstrate and verify new products and technologies [61]. Also, in the city of Bristol (UK), a joint venture company provides a digital infrastructure that can be used by companies and developers to build and test a wide range of applications and smart city services [55].

2.2.4 Extracting Knowledge from IoT Data

As mentioned before, a central question concerning such streaming big data originating from IoT installations is whether or not the generated data can be trusted. And when the data can be trusted, how do we make sense out of them by extracting knowledge, i.e., something actually useful, going beyond a technology demonstrator? Moreover, how do we provide usefulness to the owners of the data and how do we involve them in the whole process? Essentially, such questions get to the point of understanding how do we actually make the analysis of the data smarter to achieve what we need in the level of a smart city itself. Part of the answer to this question lies in creating more "useful" information out of raw sensors or other kind of data representing observations of the urban environment. For example, certain events generate data reported by the city sensing infrastructure, but are, more often than not, missing an appropriate description. Consider the case of a traffic jam inside the city center; it generates sensed values in terms of vehicles' speed, noise and gas concentration. Moreover, in most cases, multiple devices or services, while missing useful correlations in the data streams, report such values. Adding data annotations to such smart city data through machine learning, or crowdsourcing mechanisms, can help reveal a huge hidden potential in smart cities.

Although there have been pioneering studies and applications on combining human and machine intelligence, research in this field is still at its infancy stage. [35] presents a vision on the potential of combination patterns of human and machine intelligence, identifying three possible patterns sequential, parallel and interactive. Moreover, in [18] authors present a crowd-programming platform that integrates machine and human based computations. Their system integrates mechanisms for challenging tasks like human task scheduling, quality control, latency due to human behavior etc.

SONYC [69] is an example of a project with a very specific use-case, employing machine-learning algorithms to classify acoustic readings into various types of noise encountered inside an urban environment. Similarly, learning from the crowds, by using the crowdsourced labels in supervised learning tasks in a reliable and meaningful way is investigated in [60, 73]. In the above cases, each system builds what can be described as a taxonomy that is to be used for the process of knowledge extraction on top of the available datasets. Taxonomies are ubiquitous in organizing information, by grouping digital objects/content to categories and/or mapping them to abstract concepts expressing meanings, entities, events etc. Most of the modern social networking applications (like Flickr) or online collaborative tools (like Stack-Exchange) are relying heavily on an underlying taxonomy. Building and curating a taxonomy is a challenging task that requires deep knowledge of the data characteristics. Taxonomies are usually created by small groups of experts and target a very specific application domain. Folksonomies, on the other hand, are quite popular in online applications and they are categories of tags collectively organized by the users of the applications. Such taxonomies usually have weaknesses like double entries, misclassified tags, entries with typos or ambiguities in the categories, but with simple processing in the background, it is possible to normalize them and map their content to better established knowledge bases (like Wikipedia or WordNet). In [24], the authors propose a workflow that creates a taxonomy from collective efforts of crowd workers. It uses a feedback loop that suggests multiple categories to each part that needs to be categorized and then uses a different set of workers (or volunteers) to reduced the proposed categories to a single best-suggested choice. This loop can later on be used to eliminate duplicate and empty categories, and to nest related categories in an iterative process. Such an approach can produce results that are similar in cost and quality to the ones generated by experts.

2.3 Making use of IoT Data

2.3.1 Building Energy Efficiency using IoT

In general, about 75% of buildings in Europe are residential [30] and cause a significant amount of greenhouse gas emissions (in 2009 residential buildings were responsible for 68% of the total energy usage in buildings [31, 48]).Keeping this in mind, we can acknowledge that systems that provide real-time monitoring and actuation on multiple buildings over IoT infrastructures are going to be used in a highly increasing rate. Such systems can focus either on energy consumption or on environmental parameters monitoring, but can also be tailored in order to support other applications, like smart storage or manufacturing.

In recent years, the European Union has been aggressive in promoting energy efficiency in buildings, through a multitude of small and large-scale projects and initiatives. Build Up [32] is a EU-specific portal for gathering all sorts of resources for energy efficiency in buildings, e.g., energy-saving research project results, national regulations and legislation. ICT4SaveEnergy [64] was a large-scale multinational research project that involved energy efficiency in multiple types of public buildings - theaters, enterprise offices, stadiums, schools, universities. Its continuation, [21], focuses on 4 public university sites located at Helsinki, Lulea, Lisboa and Milan. SMARTSPACES [68] is another similar large-scale project, taking place at 11 pilot sites, also researching energy efficiency in public buildings of various types. More focused on school buildings are the VERYSCHOOL [71] and ZEMEDS [78] projects, with the latter further focused on Mediterranean countries and targeting retrofitting of energy-saving components.

2.3.2 Behavioural Change in Education using Energy Efficiency IoT Data

Affecting the behavioral characteristics of the citizens' interaction within the buildings where they live, learn and work will have a great impact on the overall reduction of energy consumption [48]. In the last few years there has been a wealth of activity on facilitating exactly this kind of goals. A small part of it focuses on the educational community, i.e., faculty, staff, students and parents. It should come as no surprise that raising awareness among young people and changing their behavior and habits concerning energy usage is key to achieving sustained energy reductions. Specifically in the EU, people aged under 30 represent about a third of the total population [27]. In addition, young people are very sensitive to the protection of the environment so raising awareness among children is much easier than other groups of citizens.

As we are now rapidly approaching 2020, it is as important as ever to address the skills that will enable all citizens to make informed and well-thought choices. Also, another ubiquitous fact should also come to mind: *we cannot manage what we cannot measure*. It is necessary to monitor the impact of our current behavior or the effect of potential behavioral changes in order to have a clearer picture with respect to e.g., our everyday energy consumption. Furthermore, environmental education, as part of the broader issue of science, is an important component of the EU cultural heritage [57]. In fact, EU considers environmental education one of the most prominent instrument to influence human behaviour towards more environmentally sustainable patterns [54]. Hence, associating environmental education and game-based learning will lead to students taking over a leading role in the educational process, setting questions, investigating the possible answers and looking for alternative explanations to come up with a fitting model.

Several software products for monitoring sensor data exist, able to capture sensor data from proprietary data formats or protocols and visualize them. Regarding visualisation, [33] discusses the most common approaches with respect to feedback design in eco-conscious work for the past decade, from the perspective of both ICT and psychology, providing insights to their strengths and weaknesses. [72] is an example of a typical engineering-focused approach, that utilises a number of skeuomorphic metaphors to enable smart home feedback creation on smartphones. While such systems provide end-users and developers with powerful tools and front-ends, we believe they should also be paired with multiple approaches, offering multiple possibilities to interact with the system. [29] are examples of large-scale smart metering deployments that used Web portals (for electricity and water) and other visualization tools to support the system and engage end-users to participate. Their findings support the notion that the use of multiple approaches, with respect to visualization and feedback, serves such purposes well. We have followed a similar line of thought while implementing our own user interfaces and will continue to evolve our approach in future revisions of the system.

Regarding similar work carried out in Europe, a related action was [44], in the context of the GEN6 research project. We build upon such results and have already integrated aspects of this specific project into our own. Other projects like [21] and [68] target the educational sector as well, but focus mostly on a university level while we target students of younger age. They also mostly leave out the environmental and health part of building monitoring, which is a major component in our work. Projects like [22] have instead focused more on providing educational material and game-focused activities to promote learning aspects of sustainability and energy efficiency. We embrace such methods as well, but we chose to build an IoT infrastructure to utilize the actual environment of school buildings; we believe that by using this infrastructure and the data it produces, it will be easier and more effective to build tools that better reflect the everyday reality in school life and provide more meaningful feedback, with respect to the impact of any potential changes in the behavior of students and school staff. [71] uses a similar approach to our own, however, we utilize a larger number of installations and target a broader audience.

Recently, several works introduced such concepts in the curricula of schools participating in research projects. [65] produced several guidelines and results regarding good energy saving practices in an educational setting, [71] also produced certain related results.

2.4 Graphs

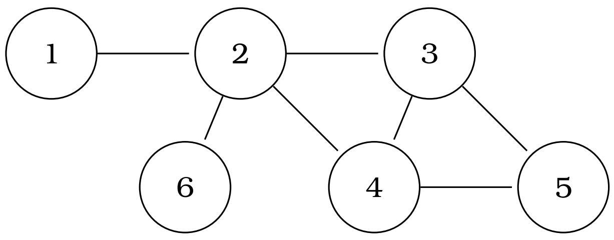


Figure 2.2 Example of a Graph.

Graphs are the basic subject studied by graph theory. The word "graph" was first used in this sense by James Joseph Sylvester in 1878. In mathematics, and more specifically in graph theory, a graph is a structure amounting to a set of objects in which some pairs of the objects are in some sense "related". The objects correspond to mathematical abstractions called vertices (also called nodes or points) and each of the related pairs of vertices is called an edge (also called an arc or line) [70].

Typically, a graph is depicted in diagrammatic form as a set of dots for the vertices, joined by lines or curves for the edges. Graphs are one of the objects of study in discrete mathematics. An example of a simple graph is available in Figure 2.2, with 6 vertices:

$$(1,2,3,4,5,6)$$

and 7 edges:

$$((1,2),(2,3),(3,4),(3,5),(4,5),(2,4),(2,6)) \text{ is available in}$$

Edges in a graph may be directed or undirected. A directed edge represents a relationship that is expressed from the starting to the ending node of the edge. For example, in a family tree a directed edge can express that a person "is the parent of" another person. In a more strict mathematical definition it is written as an ordered pair $G = (V, E)$ with V a set whose elements are called vertices, nodes, or points; E a set of ordered pairs of vertices, called directed edges. A directed graph similar to the on presented in the previous figure is available in Figure 2.3. This time the graph has again 6 vertices:

$$\text{V=}(1,2,3,4,5,6)$$

but 7 directed edges:

$$\text{E=}((1,2),(2,3),(3,4),(3,5),(4,5),(2,4),(2,6),(1,6),(6,1))$$

As we can see from the figure, edges can exist in both directions between two edges.

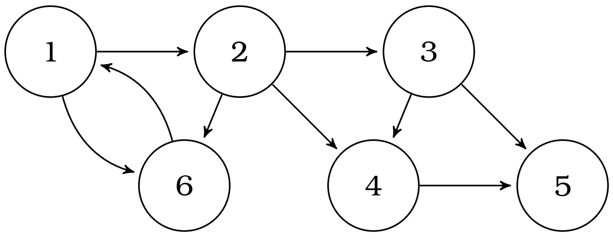


Figure 2.3 Example of a Directed Graph.

A Property graph, builds upon the concept of a directed graph and allows for the vertices and edges of the graph to have **names** or **labels**. These edges are always directed and both edges and vertices can be associated with additional properties as key/value. Figure 2.4 depicts an example of a property graph. This graph, contains 6 vertices (named `bob`, `john`, `alice`, `bike`, `car`, `hoverboard`) and 5 edges (e.g., `bob-drives->bike`). Also vertices can contain additional properties in the format of key/value pairs like for example the vertice `car` has a `brand` property with value of `BMW`.

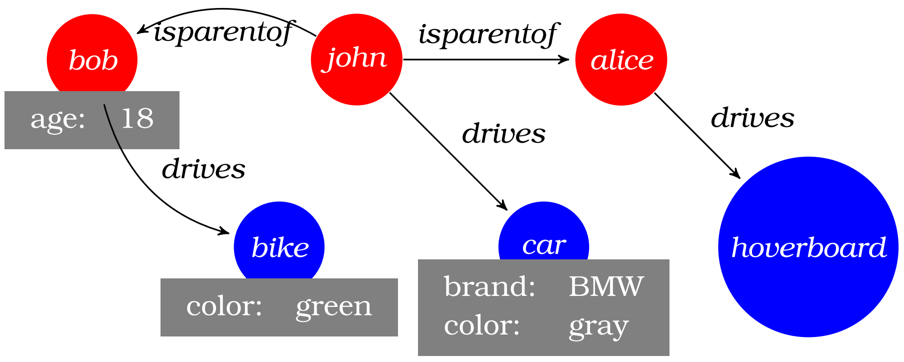


Figure 2.4 Example of a Property Graph.

Chapter 3

Representing the IoT

In order to interact with an IoT installation, an appropriate representation for it in the digital world is required. Each installation comprises multiple elements that produce, consume or transfer data. Every element in this ecosystem potentially has multiple relations with other ones creating a complex schema that can be hard to understand or visualize as the scale of the installation increases. To overcome this problem and simplify our interaction with this digital image representation, formats more appropriate than the traditional relational databases need to be applied. At the same time, these formats need to be easy to understand and be used by non tech-savvy users like artists or activists in their work with this, now accessible, living part of our environment.

In this chapter, we focus on a simple case of a building populated with IoT devices for monitoring and limited actuation use cases to extract information about all the involved entities and their possible interactions. This information will help us define a representation schema that best fits our requirements, while it is suitable for use in a high performance computation system.

3.1 A Typical Smart Building IoT Installation

As mentioned before, we focus on a building-wide installation. Such an installation is used as the basis for a number of EU funded research projects that target energy efficiency in public school buildings. It is based on the installation of custom (Figure 3.1) and off-the-self IoT devices in a number of Greek public school buildings to monitor energy consumption and environmental parameters (Figure 3.2).

The generated information, supplemented by a set of software tools aims to help educate students on energy and environmental matters while achieving better

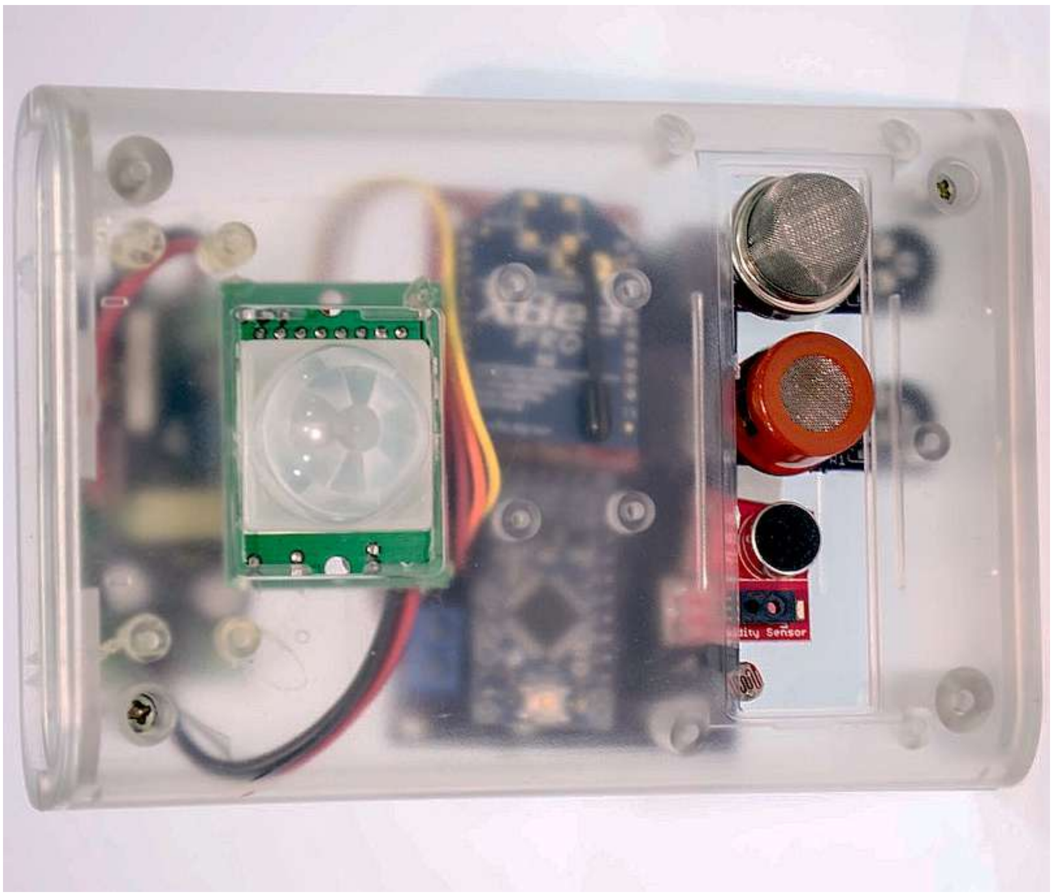


Figure 3.1 Custom IoT device containing temperature, humidity, noise, luminosity and gas concentration sensors based on the Adruino™ micro-controller.

energy efficiency in the buildings in question. By focusing on increased energy awareness and behavioral transformation within students and teaching staff, the projects “Greenmindset” and “GAIA” envisage multiple benefits apart from the savings to be achieved in terms of energy consumption. Historically, energy expenses in schools have been treated as relatively fixed and inevitable. However, evidence shows that a focus on energy use in schools yields an array of important rewards in concert with educational excellence and a healthy learning environment [25]. Educational buildings constitute the 17% of the non-residential building stock (in m^2) in the EU [30]. Since energy costs are the second largest expenditure within school district budgets, exceeded only by personnel costs [3], significant savings can be carved out for reallocation to needed services, if energy consumption can be reduced.

The educational sector presents a very interesting and important case for the problem monitoring and managing of a very large number of IoT installations situated in a very fragmented and decentralized manner. In national educational systems we have literally thousands of buildings spread throughout a country, usually, with very different characteristics in terms of construction, age, size, etc. It is reasonable to expect a diverse set of device providers working under the same interoperability framework. Due to this, the tools that interact with such an environment need to be hardware independent, supporting elements from multiple manufacturers that inter-operate under the same framework and are commonly represented and interacted with.

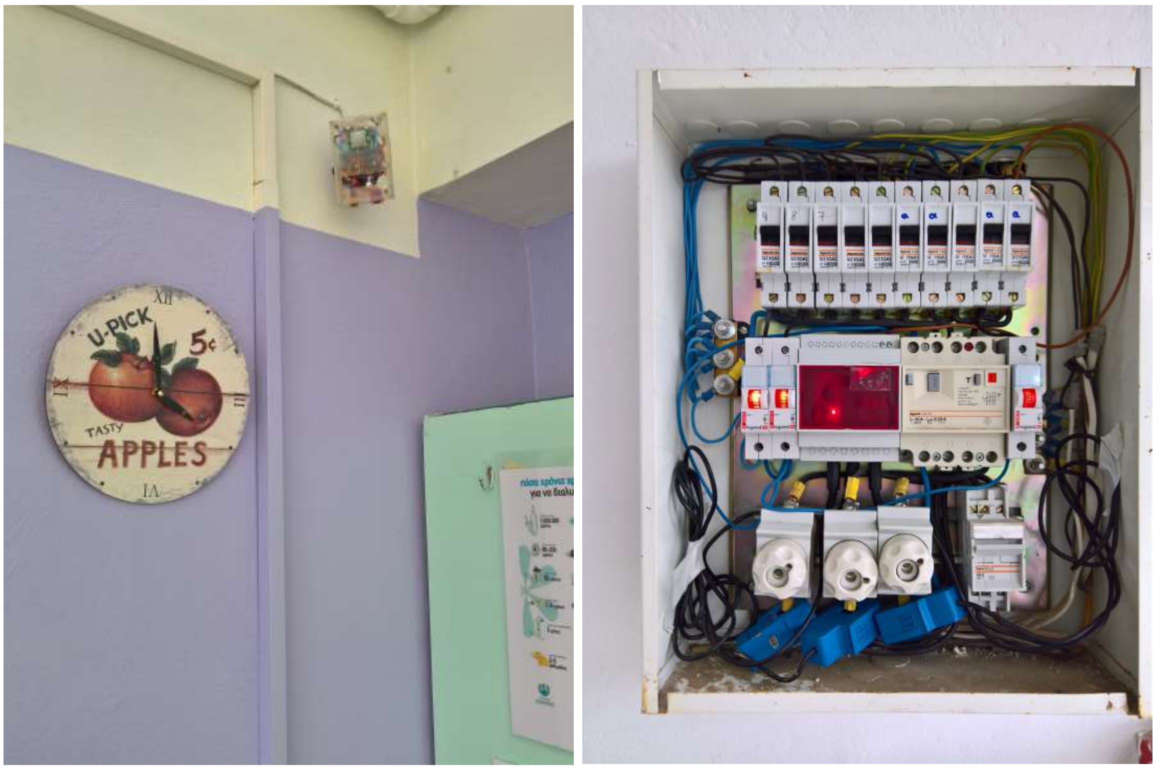


Figure 3.2 Example of two devices installed in a school building (an environmental meter on the left and a power meter on the right)

The goal of the GAIA platform is direct it to allow users directly compare the collected data of their school to other similar buildings participating in the project, by carefully taking into account environmental parameters like the time of year, location or weather. The final platform needs also to support multiple end-user groups that inherently exist in the educational sector: students, educators, building administrators and other administrative staff. In such a building, a typical installation based on the learnings of the GAIA project is comprised of:

- a power meter device installed in the main junction box of the building

- 5-10 environmental meter devices, installed in the a subset of the building's classrooms and common areas.
- a weather station device installed on the roof of the building (if easilly accessible).

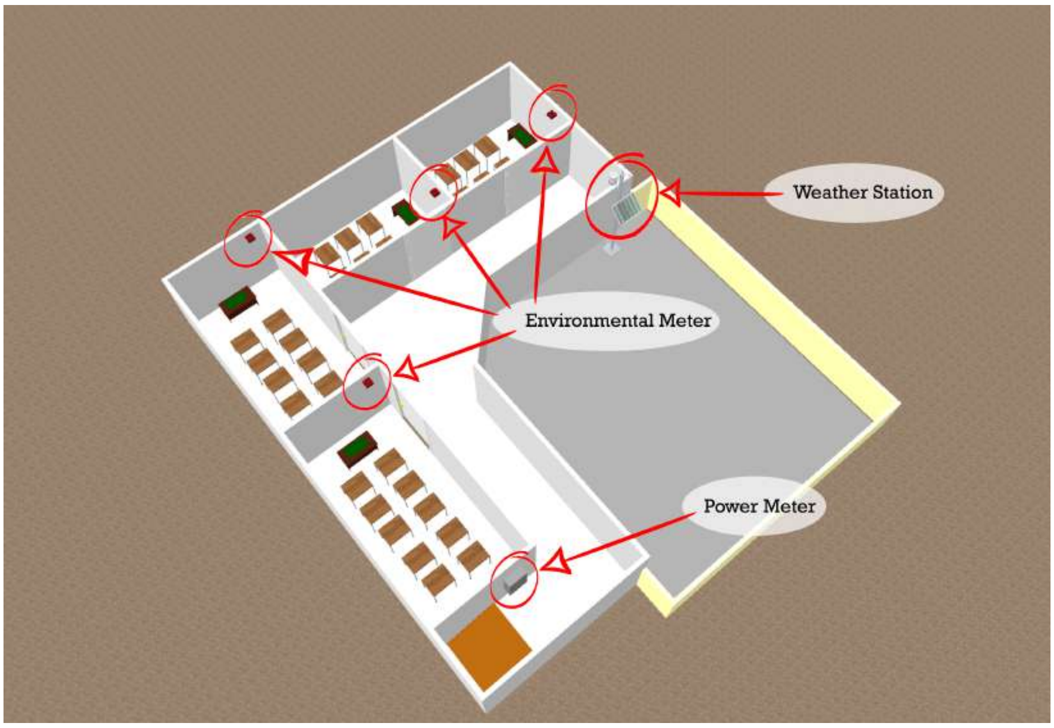


Figure 3.3 An example of a typical installation inside a school building with a weather station, 4 environmental meter and one power meter devices.

3.2 Key Elements

Based on the previous discussion, we can summarize the involved elements in the following categories:

- **Users**: Users are people that will interact with or live inside the installation.
- **Sensing devices**: Sensing devices are IoT devices that are installed and have the role of data producers. Their information can be transferred, consumed or collected for future use.

- **Actuator devices**: Actuator devices are IoT devices that can be controlled to invoke change in the physical world. They include elements like light switches or the set value of the thermostat in an HVAC system.
- **Gateway devices**: Gateway devices are IoT devices that simply have the role of transferring information between the installation and the Internet. Their role is crucial in installations where the other IoT devices installed are not Internet-enabled and cannot communicate directly with the rest of the system.
- **Observed Phenomena** Observed Phenomena refers to the physical phenomena that can be observed from a digital sensing or actuator devices.
- **Units of Measurements** Units of Measurements refer to the convention used to quantify an observed phenomenon.
- **Locations** Locations refer to the physical and logical groupings that can be applied to any of the elements presented above. They can describe a building, a school unit or a classroom but also a more abstract grouping of a set of users inside multiple school communities.

In a similar manner there are relationships between the elements presented above that are defined as follows:

- **Ownership** Ownership defines the relation in which users have the full control and permissions on a set of sensing, actuator or gateway devices and physical locations. This relationship is important for administrative and security purposes and can be used for the management of the whole installation.
- **Access**: Access defines the right of a user to view the information produced by a set of sensing devices or physical location or control an actuator device. Such rights can be limited based on the type of the user as for example teachers and students may need access to detailed information only about their own classroom while they are allowed to view only limited data on the rest of their school building. Similarly, students may not be allowed to control actuator devices even in their own classroom as elements like the HVAC system is configured only by their building managers or an authorized teacher (e.g., the principal).
- **Membership**: Membership refers to the fact that each one of the sensing actuator and gateway devices as well as users are part of a physical or logical

location. For example sensing devices are part of a physical classroom that is part of a building, while they are also part of the local educational branch (i.e., the prefecture the school belongs to). Similarly users also belong to user groups that are tied either to their actual physical location or study groups that may be defined withing their schools.

- **Sensing Attributes**: Sensing attributes refer to the sensing and actuator devices and are used to define what type of information each one of the devices generates, or what type of device it controls. The can be information about the physical phenomenon or the measurement unit that is measured or even whether the data generated are raw values or need a post processing to be valid.

3.3 Digital Representation

To provide a digital representation of what we described before we choose to use a graph database. Such a database is more appropriate to our use case as it can easily bind elements and their relations to vertices and edges of the graph. It also provides as with a common representation and storage mechanism for each element or relationship using key-value pairs for their attributes and simplifies searches inside the graph in the form of graph traversals.

3.3.1 Graph Databases

We base our design in the Neo4j Graph Platform[1]. The Neo4j native graph database is an enterprise grade solution for creating transactional applications. Neo4j comes bundled with the Cypher graph query language allows for expressive and efficient operations on the database both in terms of time and computing resources. Using a graph database we are able to leverage complex and dynamic relationships in use-cases with highly connected data to generate insight and a better understanding of the relationships between the entities of our application. Graph databases are the best way to represent and query data of any size or value. Based on the usage of the data from the graph we can separate graph databases in two separate spaces:

- Transactional, persisted online graphs, typically accessed in real time inside the context of an application.

[1]https://neo4j.com/product/

- Offline graph analytics applications, that are composed of a series of batch steps executed over a set of collected data.

The fist one mostly refers to applications that include social or user interactions while the second one refers mostly to statistical and data mining applications.

Formally, a graph is just a collection of vertices and edges-or, in less intimidating language, a set of nodes and relationships that connect them. Neo4j, follows the Property Graph Model, a simple yet powerful enough model to describe the vast majority of graph use-cases. A property graph has the following characteristics:

- It contains nodes and relationships
- Nodes contain properties (as key-value pairs)
- Relationships are named and directed, and always have a start and end node
- Relationships can also contain properties

Querying graph data in Neoj4 is possible using Cypher. Cypher is an expressive (yet compact) graph database query language. Although currently specific to Neo4j, its close affinity with our habit of representing graphs as diagrams makes it ideal for programmatically describing graphs. Cypher is arguably the easiest graph query language to learn, and is a great basis for learning about graphs. Other graph databases have other means of querying data. Many support the RDF query language SPARQL [59] and the imperative, path-based query language Gremlin [39] but Cypher is a lot easier to learn and use allowing even non-experts to express simple queries in a matter of minutes.

The Neo4j database runs on top of the JVM in two modes: as an embedded database, that is stated as part of an application but remains persisted on the disk of the application's host, or as a standalone database server. Based on the application's use-case the developers are able to select the best mode that fits their case. Using Neo4j as a standalone server allows for clustered, sharted and replicated setups like the ones presented in Figure 3.4.

Neo4j stores graph data in a number of different store files on disk. Each store file contains the data for a specific part of the graph (e.g., there are separate stores for nodes, relation-ships, labels, and properties). The division of storage responsibilities -particularly the separation of graph structure from property data-facilitates performant graph traversals, even though it means the user's view of their graph and the actual records on disk are structurally dissimilar. Figure 3.5

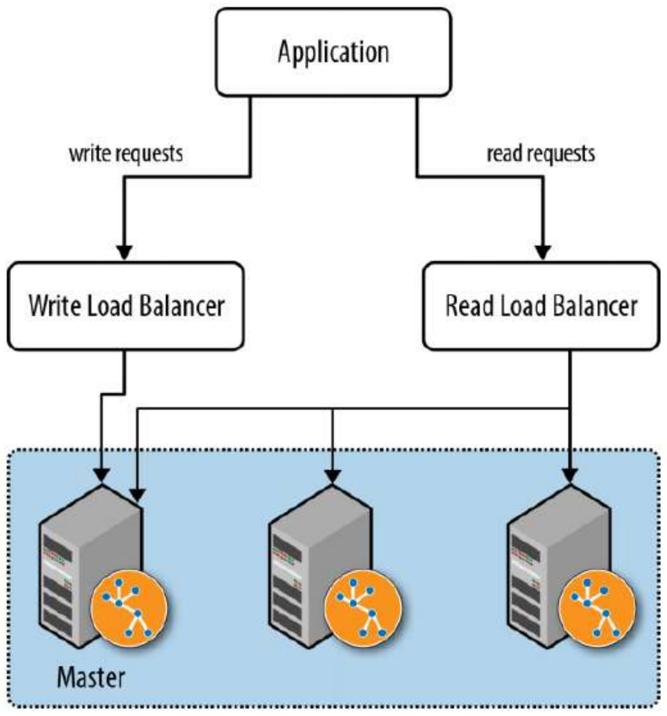


Figure 3.4 Neo4j clustered setup using read/write load balancers to direct requests to a different database instances.

describes the physical storage of the Neo4j data by depicting the structure of nodes and relationships on disk. The node store file stores node records. Every node created in the user-level graph ends up in the node store, the physical file for which is `neostore.nodestore.db`. Like most of the Neo4j store files, the node store is a fixed-size record store, where each record is nine bytes in length. Fixed-size records enable fast lookups for nodes in the store file. If we have a node with id 100, then we know its record begins 900 bytes into the file. Based on this format, the database can directly compute a record's location, at cost $O(1)$, rather than performing a search, which would be cost $O(logn)$. Correspondingly, relationships are stored in the relationship store file, `neostore.relationshipstore.db`. Like the node store, the relationship store also consists of fixed-sized records. Each relationship record contains the IDs of the nodes at the start and end of the relationship, a pointer to the relationship type (which is stored in the relationship type store), pointers for the next and previous relationship records for each of the start and end nodes, and a flag indicating whether the current record is the first in what's often called the relationship chain. Figure 3.6, shows how the various store files interact on disk. Each of the two node records contains a pointer to that node's first property

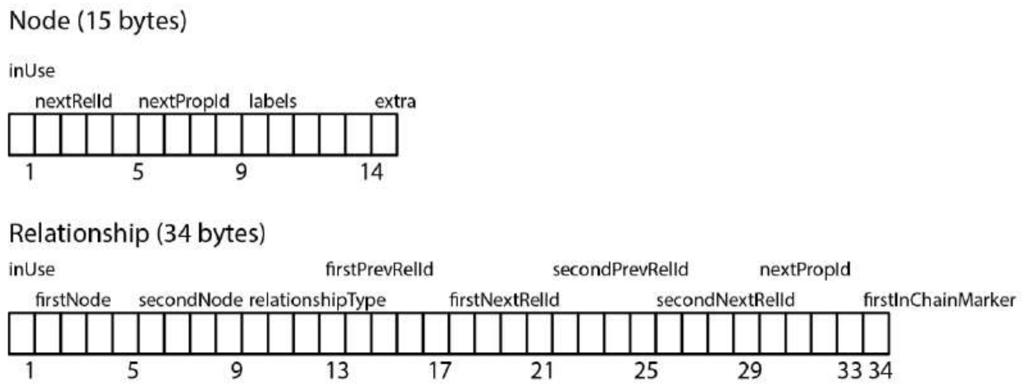


Figure 3.5 Neo4j node and relationship records as they are stored on disk.

and first relationship in a relationship chain. To read a node's properties, we follow the singly linked list structure beginning with the pointer to the first property. To find a relationship for a node, we follow that node's relationship pointer to its first relationship (the `LIKES` relationship in this example). From here, we then follow the doubly linked list of relationships for that particular node (that is, either the start node doubly linked list, or the end node doubly linked list) until we find the relationship we're interested in. Having found the record for the relationship we want, we can read that relationship's properties (if there are any) using the same singly linked list structure as is used for node properties, or we can examine the node records for the two nodes the relationship connects using its start node and end node IDs. These IDs, multiplied by the node record size, give the immediate offset of each node in the node store file.

3.3.2 The GAIA Graph Database Schema

In our case we developed a schema that contains the aforementioned elements and their relationships and applied it to the information of the installation of GAIA project. We used and embedded instance of Neo4j to persist the data of the system on disk due to some performance and stability issues we faced with the initial versions of the Neo4j server distribution. This did not limit the development of the different applications and services of our system as all functionality and operations on the actual database are handled through a single point, secured web-based application programming interface (API).

Listings 3.1 and 3.2 contain the generated schema for two of the described element and a single relationship between them in Java. Figure 3.7 shows an

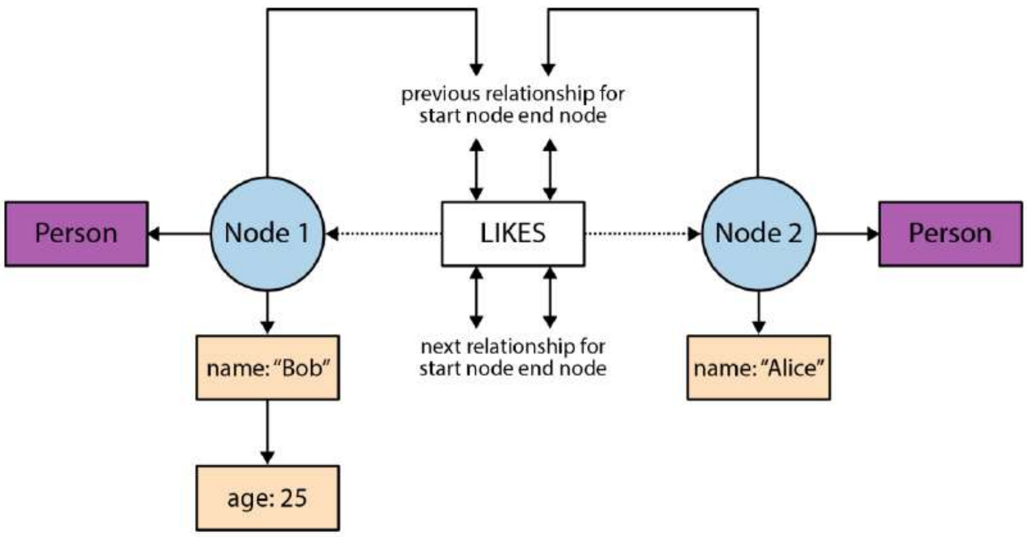


Figure 3.6 Neo4j nodes and relationships as they are physically stored on disk and how they point to each other.

example of how tow of the core entities of the GAIA schema are stored on disk and connected with each other. The final list of entities in the production version of the GAIA schema contains the following:

- `User`: for storing user related information
- `Site`: for storing schools and classrooms,
- `Resource`: for storing sensing points,
- `Gateway`: for storing gateway nodes that communicate with sensing devices,
- `Property`: for storing sensing capabilities and metadata

Also the GAIA schema contains the following relationships:

- `ResourceProperty`: to link a Resource entity with the Properties showing its sensing parameters,
- `GatewayProperty`: to link a Gateway entity with the Properties showing its connectivity capabilities
- `IsPartOf`: to link a Resource entity with the location it physically resides in,
- `ShareWith`: to provide Site access to Users of the system,
- `SubSite`: to easilly describe the structure inside school buildings.

```
NodeEntity
public class Resource implements Serializable {
    GraphId
    private Long id;

    private String uri;

    private String name;

    private long creationDate;

    RelatedTo(type = "providedby")
    private Gateway gateway;

    RelatedTo(type = "ispartof")
    private Set<Location> locations;

    private Set<String> tags;
}
```

Listing 3.1 Example of a node entity in the graph database

```
RelationshipEntity(type = "ispartof")
public class IsPartOf implements Serializable {

    GraphId
    Long id;

    StartNode
    private Resource resource;

    EndNode
    private Site site;

}

RelationshipEntity(type = "property")
public class ResourceProperty implements Serializable {

    GraphId
    Long id;

    StartNode
    private Resource resource;

    EndNode
    private Property property;

    private String predicate;

}
```

Listing 3.2 Example of a relationship entities in the graph database

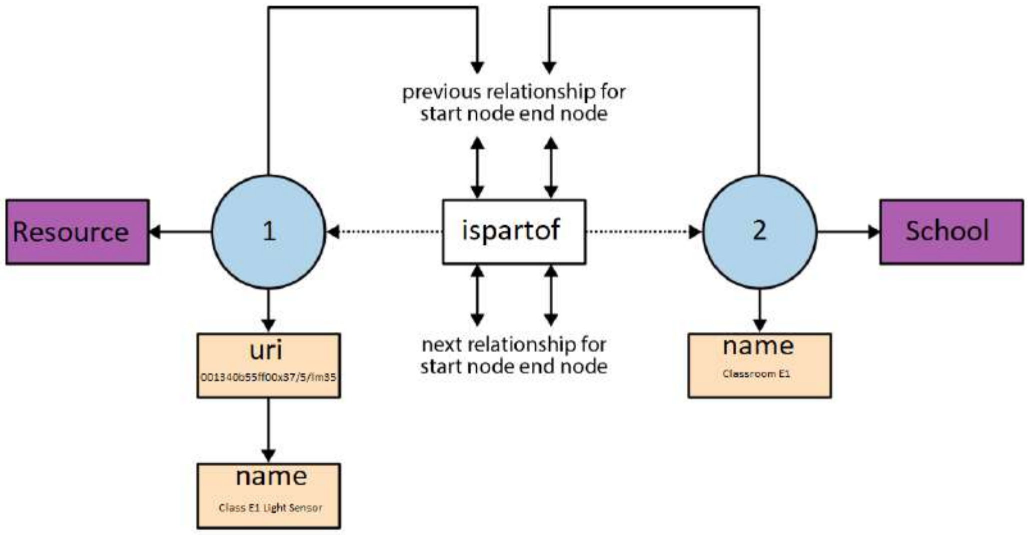


Figure 3.7 Neo4j nodes and relationships for the GAIA project as they are physically stored on disk and how they point to each other.

3.3.3 Comparing the Graph and Relational Database Schemas

In this section we are going to present how some of the entities and relationships presented above could be implemented in a traditional relational database. Based on this implementation we will show how the respective implementation in a graph database compares with the relational one.

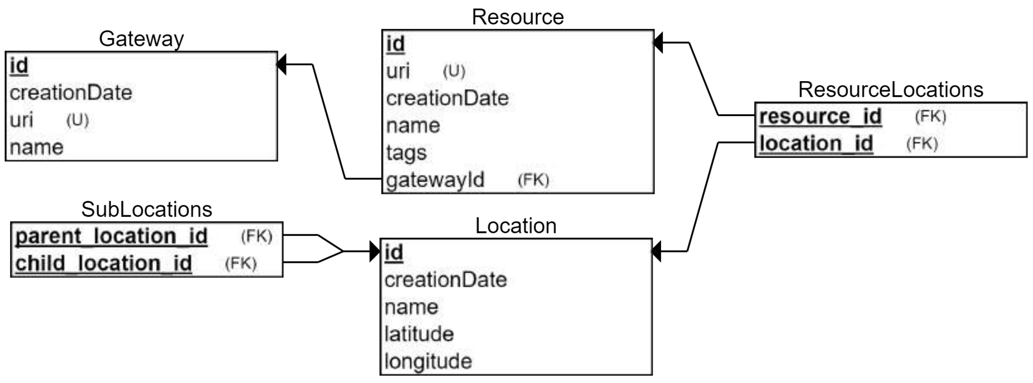


Figure 3.8 Entity Relational Diagram for the GAIA schema.

Figure 3.8 shows part of an Entity Relationship diagram for the GAIA schema. This implementation follows the principles of relational databases like `MySQL`, `MariaDB` and `PostgreSQL`. It contains 3 entities (`Resource`, `Gateway` and `Location`) together with one helper table needed to represent the many-to-many relationship between Resources and Locations and one for the recursive relationship of the Locations to express hierarchical structures. The diagram also contains the many-to-one relationship between Resources and Gateways expressed with a single foreign key in the Resource table. Out of hand we can observe that in this case we need to store two new tables to express some of the relationships of our schema.

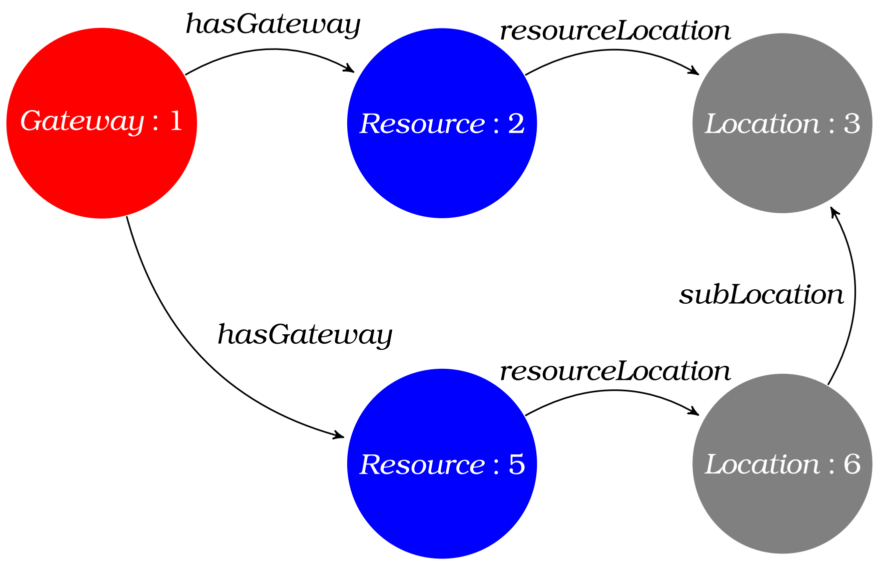


Figure 3.9 Entity Diagram for the GAIA schema in as a graph.

On the contrary, to express the same schema in a graph database we need a schema similar Figure 3.9. This schema contains the 3 entities and the relationships between them. As we can see, we do not need additional storage for any relationship, as the graph database stores each edge separately and therefore we can describe each relationship of our graph as an additional entry in a single “relationship” schema, as we described in Section 3.3.1. Properties of the vertices and edges are stored in a similar manner to the relational databases.

3.3.4 Real World Application

Thus far 18 school buildings (Table 3.1) have been involved in the project, spreading in 3 countries (Greece, Italy, Sweden), covering a range of local climatic conditions and educational levels (primary, secondary, high school and university). Electricity consumption meters are installed in all of these buildings, along with sensors monitoring indoor and outdoor conditions as described above. The vast majority of the rooms monitored are used for teaching purposes and the rest for other activities like teacher/staff rooms, etc. The year of construction of these buildings ranges from

1950 to 2000. To represent all the information of the GAIA project the graph database contains a total of 7332 vertices and 47864 edges in total for the deployment and uses a total of 113 MB in disk space. From those vertices, 4749 are sensing points, are 1012 locations, 681 are sensing properties and 300 are users of the system.

Parameter	**#**	**Description**
Educational Buildings	18	13 Greece 4 Italy 1 Sweden
Sensing Points	1000	$\geq$ five sensors per device
Students	5500	students in all levels
Teachers	900	teachers in all levels
Sensing Rate	30 s	classroom sensors

Table 3.1 Data of the GAIA deployment.

Multiple queries on the data have been defined. Some of them are presented in Listing 3.3 and the average times for executing them is presented in Table 3.2.

Query	**Result Size**	**Time (ms)**
List all School locations	18	6.7
List all Classroom locations	180	12.6
List all devices	1000	5.4
Find Classrooms of School	10	1.8
Find devices of Classroom	5	1.4

Table 3.2 Execution times of typical queries for the graph database.

```
//List all School locations
MATCH (n:"School") WITH n RETURN n

//List all Classroom locations
MATCH (n:"Classroom") WITH n RETURN n

//List all Sensing and Actuator devices
MATCH (n:"Resource") WITH n RETURN n

//Find all Classroom locations inside a School location
MATCH (n:"School")-[r_c1:"IN"]-(c1:"Classroom")
WHERE n."id" = { "XXX" } WITH n RETURN c1

//Find all Sensing and Actuator devices in a Classroom location
MATCH (n:"Classroom")-[r_r1:"PLACED_IM"]-(r1:"Resource")
WHERE n."id" = { "XXX" } WITH n RETURN r1
```

Listing 3.3 Graph query examples in Cypher

A snapshot of the graph If we could isolate a subset of the graph that describes a single school building we would end up with a graph similar to Figure 3.10. In this graph we see the owner of the installation, the `User` 8gym. The school building is called $8^{th}HighSchool$ and we can identify in the figure the ownership relationship between the `User` and `School` entities. The school building is in our case composed of 3 `Classrooms` (1,2,3). The `Classrooms` are connected to the `School` with a relationship called `in`. This relationship is directed, and points from the smaller location to the one that encloses it, physically or logically. On the other side of the graph, we can see the `Gateway` node (with the name 1) that is similarly owned by the `User`. Connected to this `Gateway` we can find 3 `Resources` (012,022,032). These `Resources` are also connected with the `Gateway` in the graph with a `of` relationship. These `Resources` are also part of a physical location and collect data for them. As a result they are also connected to these `Classrooms` with the `placed in` relationship.

In each of these entities and relationships we also have stored also a set of properties that contain the parameters of the entity like the password for the `User` or the location of the `School`. We do not represent this information here to provide a more clear picture about how the graph is built. A more complex snapshot of a school generated from the Neo4j database is available in Figure 3.11.

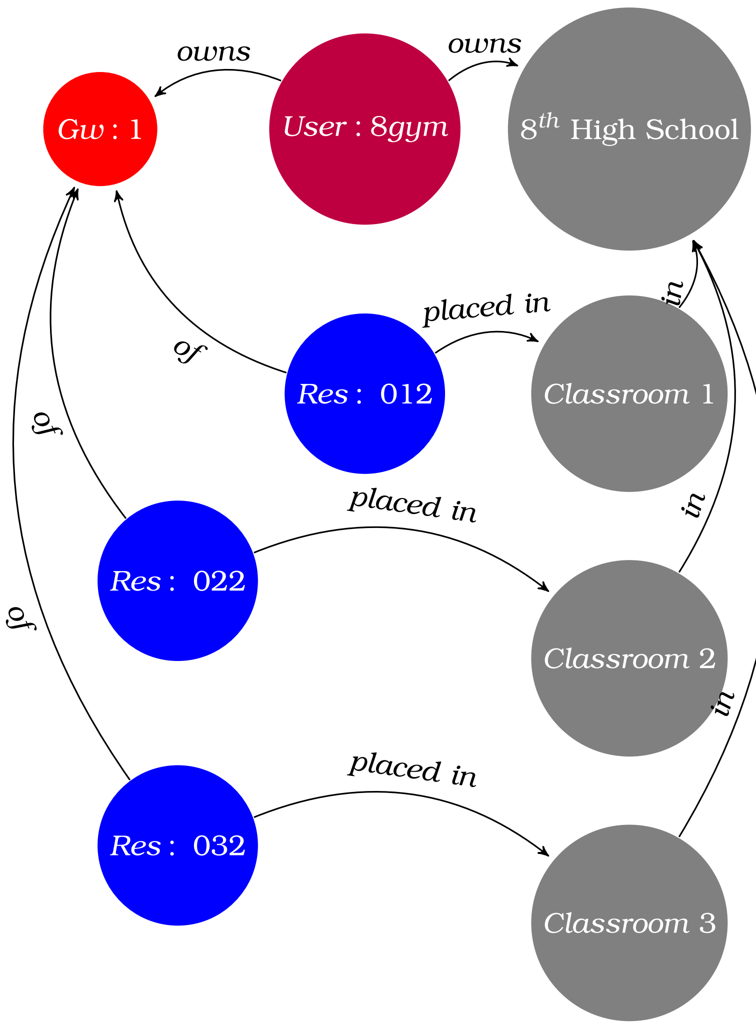


Figure 3.10 Snapshot of a part of the GAIA graph.

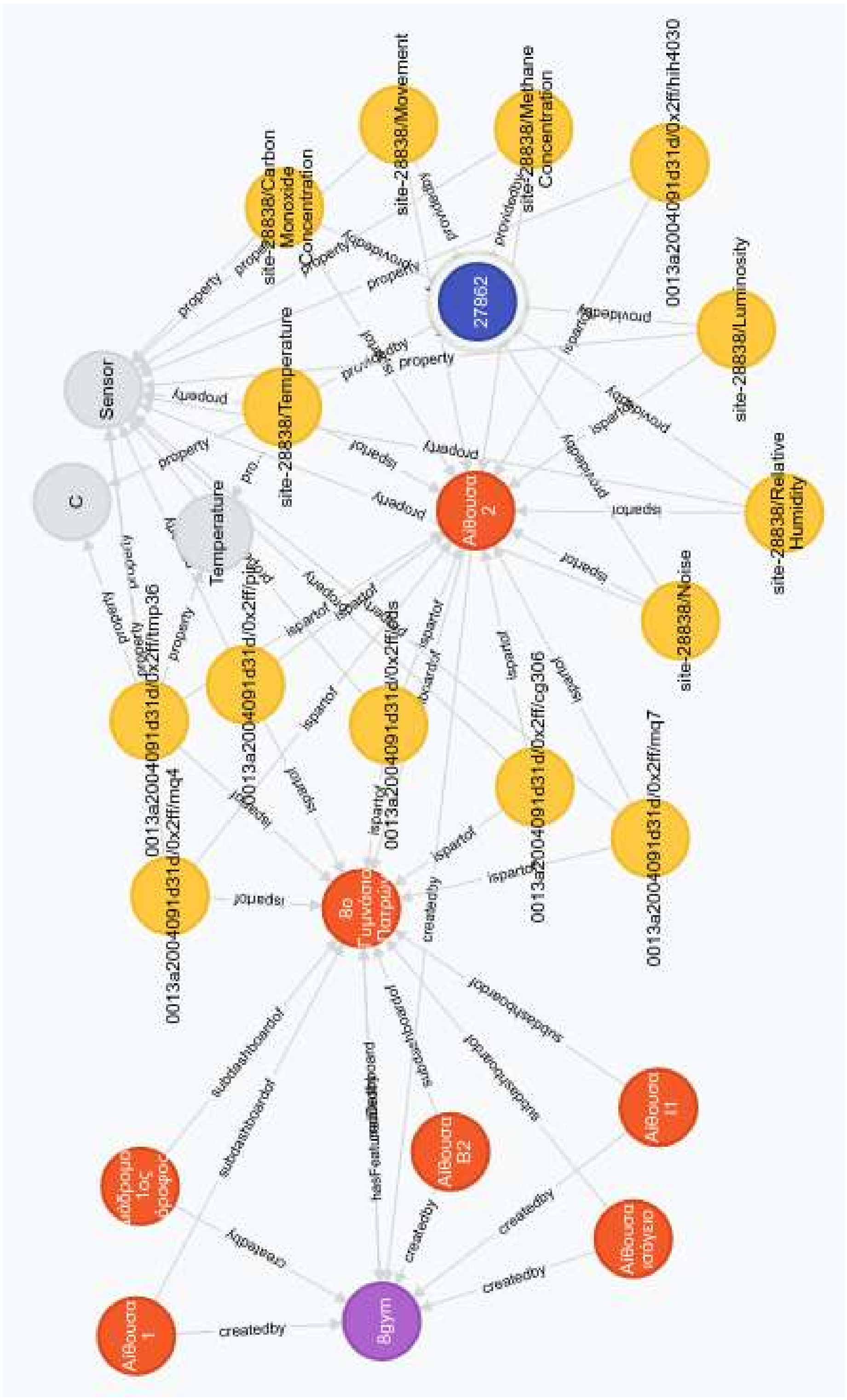

site-28838/Movement
site-28838/Methane Concentration
0013a2004091d31d/0x2ff/hih4030
site-28838/Carbon Monoxide Concentration
27862
site-28838/Luminosity
Sensor
site-28838/Temperature
C
Temperature
Αίθουσα 2
site-28838/Relative Humidity
site-28838/Noise
0013a2004091d31d/0x2ff/tmp36
0013a2004091d31d/0x2ff/pir
0013a2004091d31d/0x2ff/cg306
0013a2004091d31d/0x2ff/mq7
0013a2004091d31d/0x2ff/mq4
8o Γυμνάσιο Πατρών
Αίθουσα B2
Αίθουσα 11
Αίθουσα 1
8gym
property
providedby
ispartof
subdashboardof
hasFeatureOfInterest
createdby

Figure 3.11 Snapshot of a part of the GAIA graph taken from the Neo4j database.

Chapter 4

Data Collection

The next milestone, after managing to represent the complex structures of an IoT deployment is to build a system that is capable of receiving all the data that such a deployment will produce. This system needs to cover a set of basic requirements to be successful future-proof (easy to maintain and extend):

- **openness**, to support a number of different IoT ecosystems,
- **versatility**, to support different application domains, e.g., energy efficiency and educational scenarios,
- **scalability**, to support a very large number of buildings and IoT sensing end-points,
- **up-to-date support** of modern practices in the design of the system, i.e., cloud-based solutions, easy deployment, etc.

To provide such a solution, we base our work on the use of open-source, well established application frameworks that are used by many software developers around the world. These technologies are also easy to extend and support new technologies and function over multiple infrastructures reducing the chances of vendor lock situations. Also, we base our work on services that can easily be scaled vertically or horizontally to support the increasing needs of the deployment that will grow over time, minimizing the need to redesign any part of the system.

In the rest of this chapter, we present how we designed a system that is destined to analyze the data from a large scale IoT deployment in real-time and offer them to different users in the context of the EU project GAIA. Our design was based on the needs of the specific deployment but was also focused on the scalability of the platform in the future to support new hardware and new requirements as they arose.

4.1 Devices

In each of the buildings participating in the GAIA pilot installations, we deployed sensor devices that measure (a) the overall power consumption of the building, (b) the environmental comfort within each individual class (see below for more details), and (c) the weather conditions and air pollution levels in each building. These devices can be split into three different categories based on their origin and operation type. In general, we use (i) custom made IoT devices that communicate using an IEEE 802.15.4 local network[58], (ii) proprietary off-the-self IoT devices that communicate using IEEE 802.11 and 3G in areas that we cannot easily connect to, and (iii) sensors from legacy Building Management Systems (BMS) that are already installed in a number of school buildings. Most indoor IoT nodes form IEEE 802.15.4 networks (Zigbee or plain) and communicate with their respective edge devices by establishing ad hoc multihop bidirectional trees, set up at the time of the deployment and maintained throughout the network lifetime. The outdoor nodes are connected via Power Over Ethernet cables to transfer both electricity and maintain communication over a single cable, while in some other cases we also used IEEE 802.11 and supplied the weather stations with batteries and solar panels to harvest energy from the sun. On the transport and session layers, the devices communicate using either a custom protocol or Zigbee for the discovery of resources and transmission of measurements. In the rest of this section, we provide some more details on the categories of devices we have integrated.

4.1.1 Custom IoT Devices

Environmental Comfort The *environmental comfort meters* measure various aspects affecting the well-being of the building's inhabitants, such as thermal (satisfaction with surrounding thermal conditions), visual (perception of available light) comfort and overall noise exposure. We also monitor room occupancy using passive infrared sensors (PIR). These devices are also equipped with XBee wireless transceivers, in order to access an IEEE 802.15.4 network and transmit the measurements to the cloud services via our custom-made gateways. For more details regarding the design and technical specification of the devices, see [58]. Images of the device and its installation are available in Figure 4.1.

Power Consumption The *power consumption meters* installed measure the apparent power and the electrical current drawn from the network by each school building.

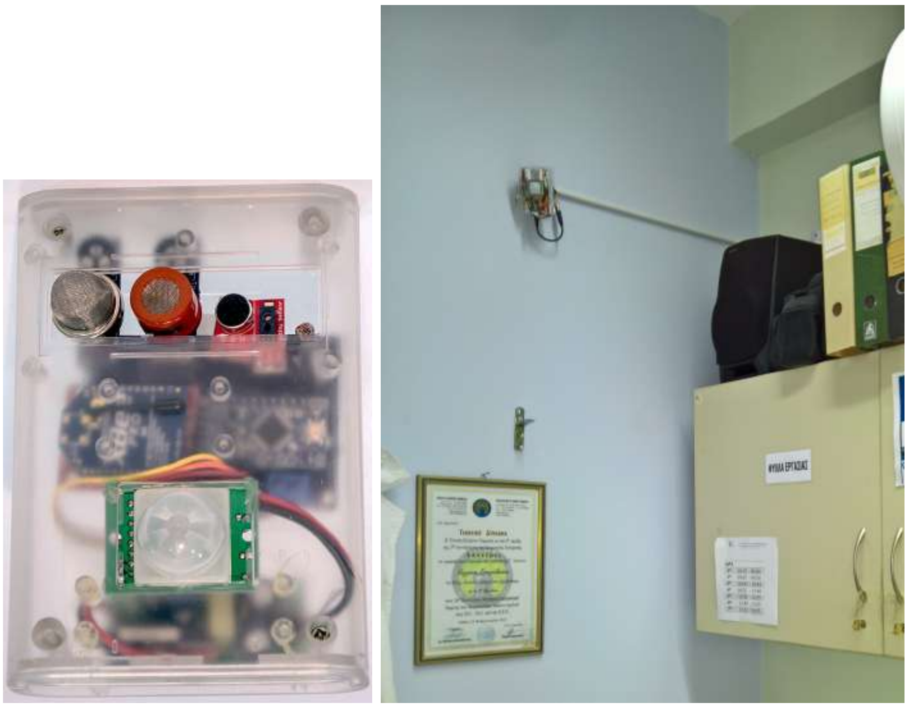


Figure 4.1 A custom Arduino-based environmental comfort meter device installed in school buildings.

Regarding the electrical setup, 3-phase electric power installations are a common practice for most public and private non-housing buildings, such as schools, in Greece. Three separate single-phase supplies, with a fourth neutral connection, provide a constant voltage to power most common single-phase appliances. In order to measure the total power consumption of such an installation, it is necessary to independently measure the power consumption of each phase and add up the total consumption, as if the installation consisted of three separate lines. Meters are situated on the main distribution board of each such building to measure each one of the three-phase power supply of the building. These devices are equipped with XBee wireless transceivers, in order to access an IEEE 802.15.4 network and transmit the measurements to cloud services via the custom made gateway nodes. For more details regarding the design and technical specification of the devices, see [58]. Images of the device and its installation are available in Figure 4.2.

Weather and Atmosphere Stations These provide information on the outdoor atmospheric conditions including precipitation levels, wind speed and direction. The *atmospheric meters* monitor atmospheric pressure and the concentration of selected pollutants, to provide insights on the pollution levels near school buildings. These

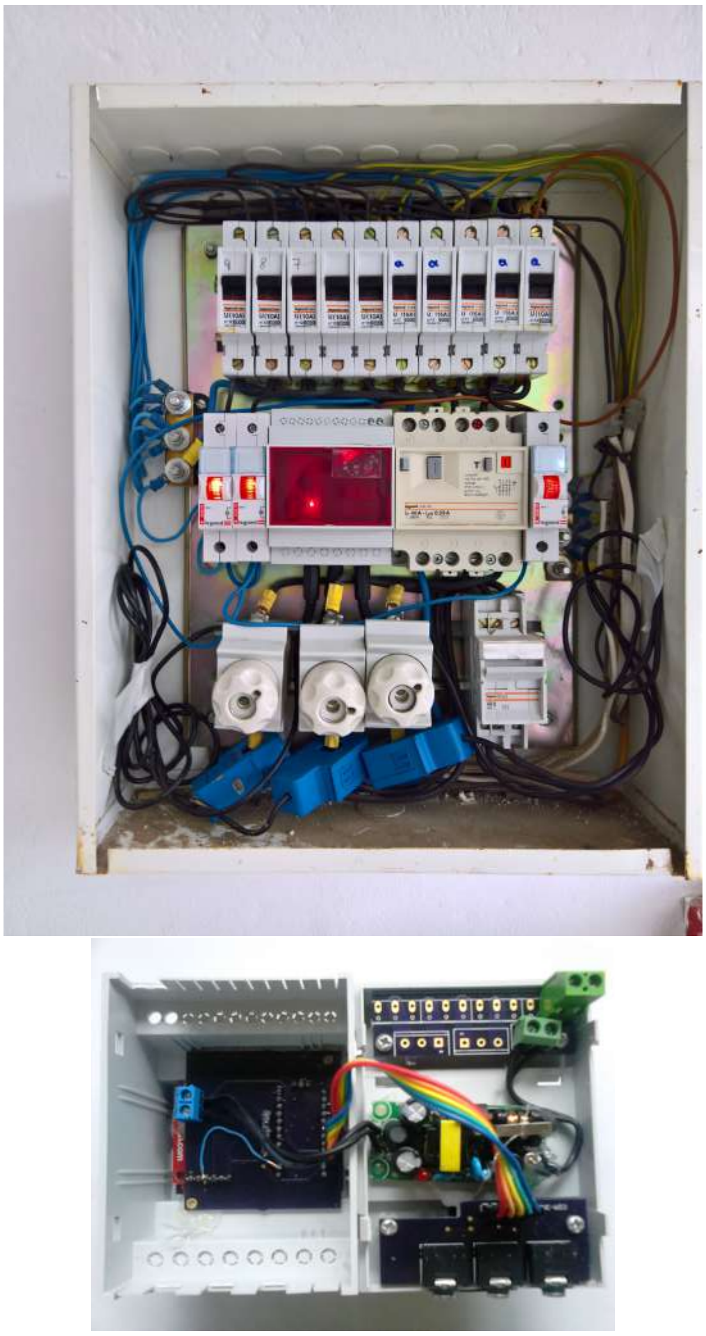


Figure 4.2 A custom Arduino-based power meter device installed in school buildings.

devices communicate with our system directly via Ethernet or WiFi and are powered using Power-Over-Ethernet or are plugged into the sockets of the building when available. For more details regarding the design and technical specification of the devices, see [58]. Images of the device and its installation are available in Figure 4.3.

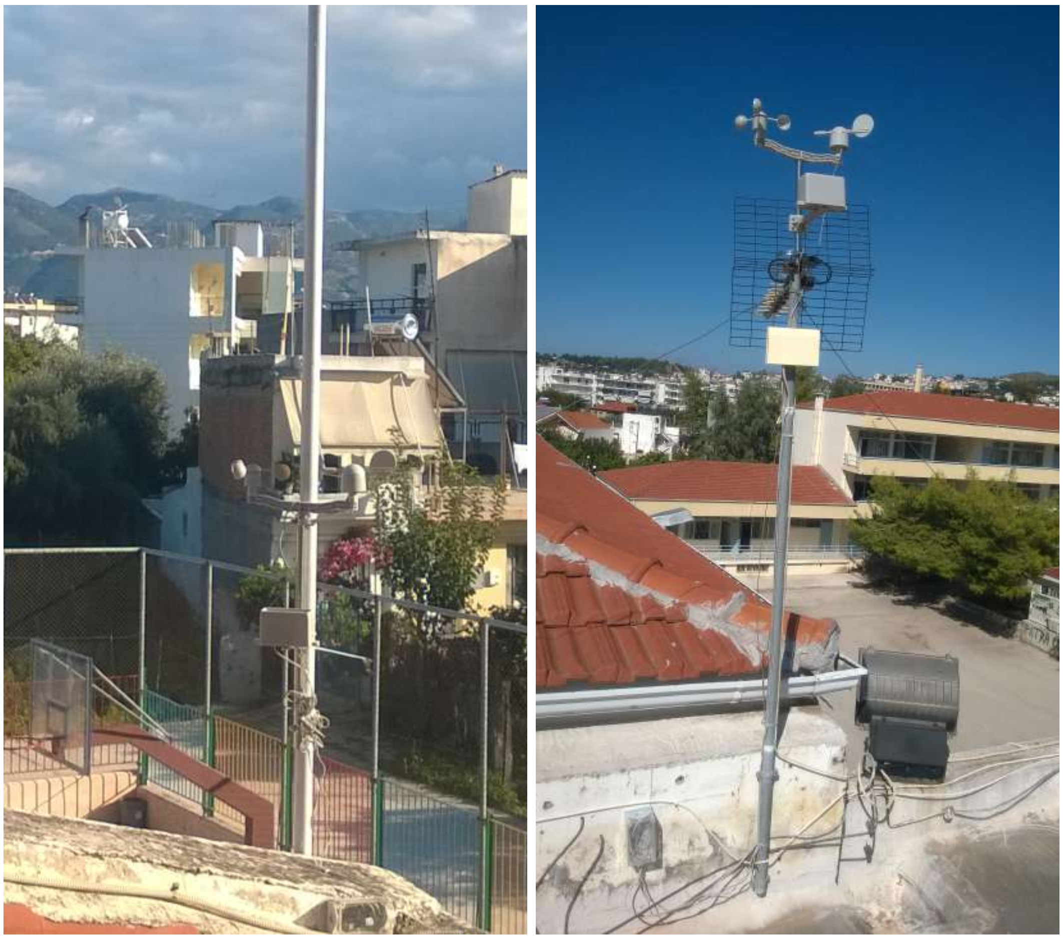


Figure 4.3 A custom Arduino-based weather station meter device installed in school buildings.

4.1.2 Proprietary Devices

Meazon In certain locations where the installation of our custom devices was not feasible due to connectivity or other restrictions, we have installed a number of Meazon[1] power meters and sensors. These sensors communicate with Meazon's proprietary data infrastructure and their data are then pushed to our platform. On the hardware side, these devices communicate using Zigbee to a central gateway device that is either connected to the Internet via Ethernet or use 3G in order to communicate directly with Meazon's proprietary cloud services. Images of the device and its installation are available in Figure 4.4.

[1]https://meazon.com/

Figure 4.4 A Meazon power meter device.

synField Similarly, in some of the buildings instead of our custom weather stations, we used some off-the-self synField[2] weather stations that offered us WiFi connectivity to avoid installing additional cables on the roofs of the buildings, as well as energy harvesting via solar panels. In this case, the weather stations communicate via WiFi directly to proprietary cloud services. Images of the device and its installation are available in Figure 4.5.

4.1.3 Legacy installations

BMS In one of the schools involved (a large technical high school/college) a BMS was already in place, utilized by the building manager and other technical staff to monitor and control several aspects of the day-to-day business. However, this system provided little to none standard interfaces to external systems. To integrate its infrastructure to our system, a custom application was developed to poll periodically the collected data directly from the application's database and transmit the data to our platform.

4.2 Architecture

Our goals it to build a system that is capable of handling an unlimited amount of data in real-time without any delays or outages. A graphical representation of the components used in the actual implementation that handles all the data originating

[2]http://synfield.synelixis.com/

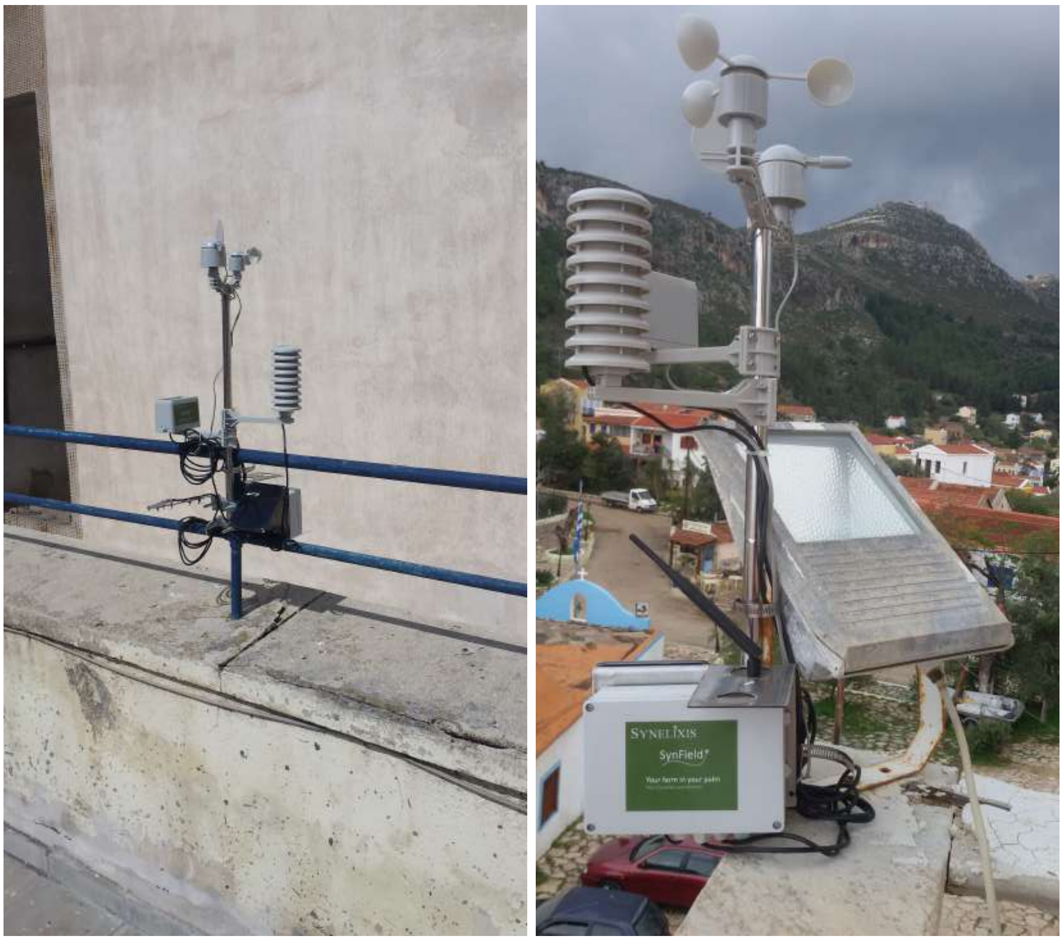


Figure 4.5 synField weather stations installed in school buildings.

from the educational buildings of GAIA is presented in Figure 4.6. The system is designed based on the microservices [3] approach. It is therefore comprised of a set of loosely connected services that communicate using web interfaces and a central message exchange service. This central point of our architecture is a message broker service that allows all other services to publish updates or subscribe to data streams and receive notifications as they appear. Its role is to merge the various data inputs from the IoT devices and feed information to the a central processing service that analysis the data and reintroduce them to the message broker for further processing or storage. The Continuous Computation Engine that will be presented in the rest of this section subscribes to updates on the broker and publishes the final calculated results back to the broker for storage and distribution to other attached services. As we will showcase in the next chapters this is allows us to easily add new

[3]https://en.wikipedia.org/wiki/Microservices

services to the system and gives us the freedom we need to implement each service in the best suited environment using the best tools available each time. Appropriate Application Programming Interfaces are provided to retrieve information from the system (historical data or schema information) to build user-facing applications that appear on the right hand side of Figure 4.6.

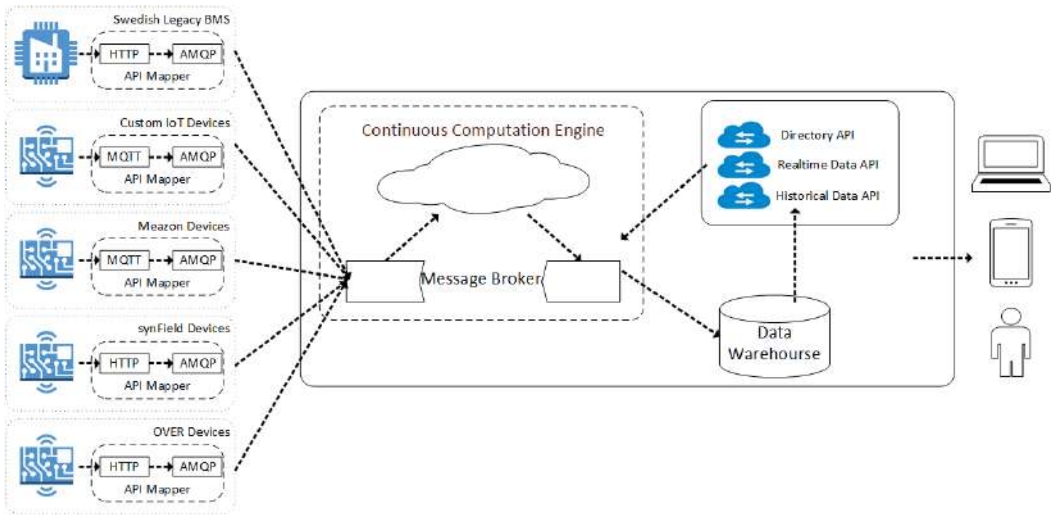


Figure 4.6 Educational building-specific IoT architecture

We also include in the architecture diagram the different data input services on the left hand side of figure and the Data Warehouse service that collects all the analyzed data and stores them in an internal database. Our platform provides a unified API for retrieving data from multiple sites and multiple hardware platforms with transparency. Each hardware device integrated in the platform is mapped to a *resource*. Resources are self-described *Entities* and are also software/hardware agnostic. The Data API acts as a wrapper function and hides much of the lower-level plumbing of hardware specific API calls for querying and retrieving data, providing a common API for retrieving historical or real-time data from resources.

To facilitate integration between the existing hardware and software technologies, the exchange of the information occurs through *API Mappers*. The API Mapper acts as a translation proxy for data acquisition and it is responsible for polling the devices infrastructure through proprietary APIs and translating the received measurements in a ready to process form for the platform. In general, the API Mapper transforms data to and from the API. The data input type can be, based on each device's capabilities, either poll based and/or push-based. In more detail, the API Mapper is capable of receiving data from the IoT devices but also of sending messages/commands to the devices. Furthermore, according to the system design, the API Mappers introduce scalability and modularity in the platform. Our solution offers two separate types of API Mappers for integration with external services and to retrieve IoT sensor data: (a) Polling API Mapper and (b) Message Queue API Mapper. Both solutions were used to integrate with data originating from the IoT installations that originate from different devices provided by different manufacturers.

4.2.1 Gathering data from the installations

Polling API Mapper

The first solution (Polling API Mapper) is based on polling of remote APIs that contain the collected data. The API mapper does not poll the actual IoT devices put an interface made available by their manufacturers. The IoT devices themselves forward their data to the manufacturers' backend using proprietary protocols that were not publicly available.

A usage example is the following: weather stations are installed in a subset of the school buildings of GAIA. Data produced by such stations are accessible through a field manager application that provides historical information through both a dedicated web interface (Figure 4.7) and a secured RESTful API. To integrate them into our system, the weather API Mapper was implemented based on the RESTful API provide by Synfiedl. The IoT devices update the backend every 5 minutes and our applications queries it every 5 minutes for updated data. When new data is found, it is formatted to the internal format of our system and forwarded for processing and analysis. The data is then processed and can be accessed by the users of the GAIA platform. A similar implementation, based on the Polling API Mapper, is used to integrate legacy data provided by a web-based building management system (Figure 4.8) installed in one of the schools of the project. This system does not offer any kind of programming interface to retrieve data. In that case, we used the HtmlUnit [4] GUI-Less browser for Java programs to simulate accessing the data of the web-interface inside the API Mapper. Once we access the pages of the building manager system we extract the latest measurements and format them to the internal format of our system as before. Then the data are ready to be forwarded for processing and analysis and finally available to the users of GAIA.

[4]http://htmlunit.sourceforge.net/

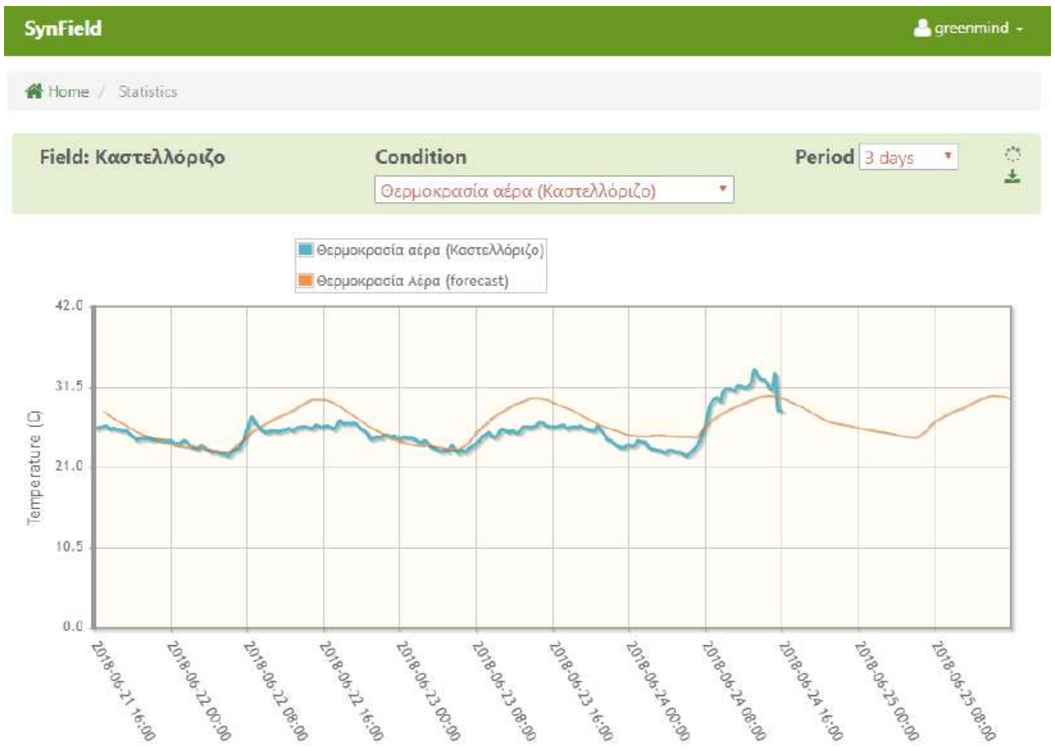
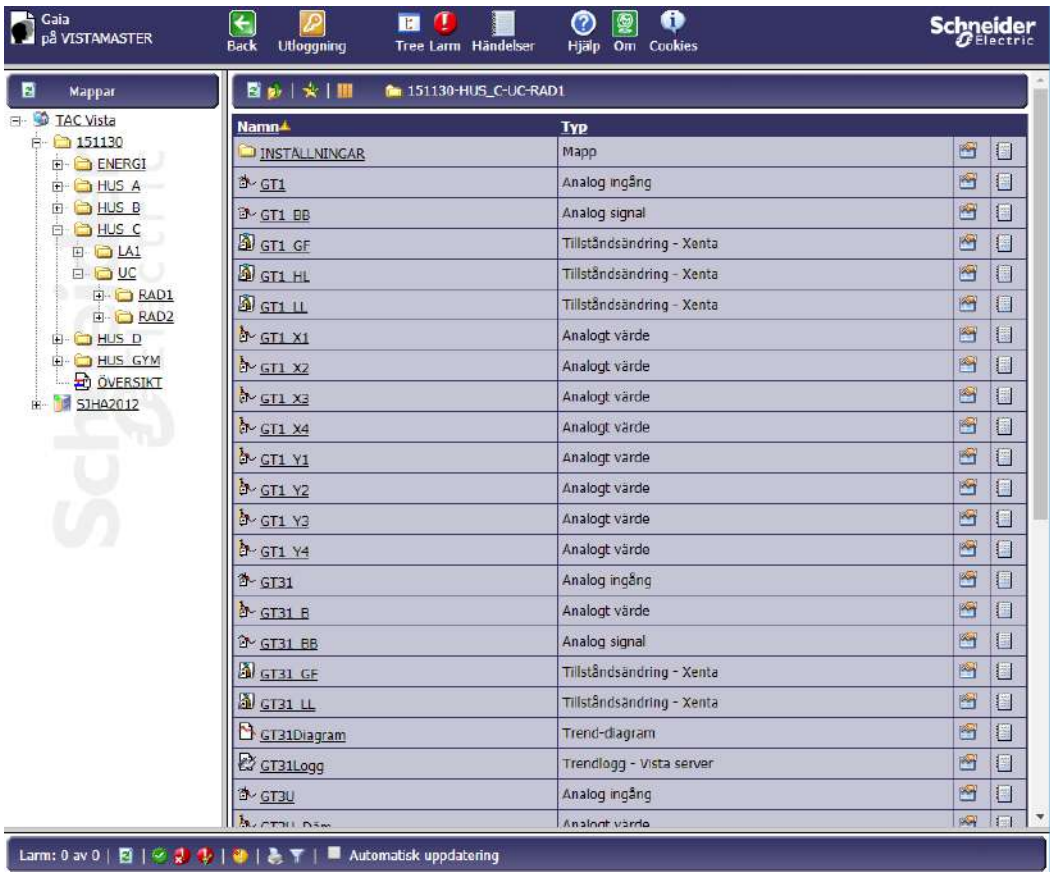


Figure 4.7 The synField Web Interface.

Push-based API Mapper

The second solution is used when the IoT devices themselves or the backend of the device manufacturer is capable of offering publish/subscribe capabilities. In that case, either the devices or an external service is capable of pushing the IoT data (generated or gathered) to a provided endpoint. The API Mapper is then able to receive the new measurements asynchronously and format them to the internal format of our platform. The data is then forwarded for processing and analysis and made available to the GAIA users. In our case, we use this option on a set of custom IoT devices that are installed inside the schools and provide the main volume of data for our system. Each device installed forwards its data every a few seconds (typically 30) to a local gateway device placed in each building. The gateway then forwards the data to a configured MQTT broker in a text based format: the topic of the message refers to the device and sensor that generated the message while the actual payload represents the value generated. For example, if a sensor with a hardware (MAC) address `124B00061ED466` publishes a temperature value of 20 degrees Centigrade, the topic is `124B00061ED466/temperature` and the message is `20`. All sensors forward their measurements periodically (every 30 seconds) or based on events (i.e.,



Namn	Typ
INSTÄLLNINGAR	Mapp
GT1	Analog ingång
GT1_BB	Analog signal
GT1_GF	Tillståndsändring - Xenta
GT1_HL	Tillståndsändring - Xenta
GT1_LL	Tillståndsändring - Xenta
GT1_X1	Analogt värde
GT1_X2	Analogt värde
GT1_X3	Analogt värde
GT1_X4	Analogt värde
GT1_Y1	Analogt värde
GT1_Y2	Analogt värde
GT1_Y3	Analogt värde
GT1_Y4	Analogt värde
GT31	Analog ingång
GT31_B	Analogt värde
GT31_BB	Analog signal
GT31_GF	Tillståndsändring - Xenta
GT31_LL	Tillståndsändring - Xenta
GT31Diagram	Trend-diagram
GT31Logg	Trendlogg - Vista server
GT3U	Analog ingång

Figure 4.8 The web-based building management system installed in one of the GAIA buildings.

when motion is detected) and the API mapper receives, transforms and forwards them to the processing engine so that they are available for the GAIA users.

4.2.2 Data Processing

Our target for the analysis of the data retrieved from the devices is to implement an engine that is capable of handling the data streams originating from the devices, as presented above, in real time with minimum latency. To achieve this, we need a framework that is scalable and can operate using a messaging mechanism to exchange data between its sub-components. Apache Storm[5] is a free and open source distributed real-time computation system that we use to guarantee this behaviour. It allows us to split all the steps of the processing of the data into tasks that are executed asynchronously but with the correct order, forming a pipeline of

[5]http://storm.apache.org/

transformations that are applied in the original data entering the processing engine. An abstract view of that schema is presented in Figure 4.9. This flow is called a **processing topology** in the context of Apache Storm. In it we can spot the tasks that feed data into the system (called **spouts**) presented as faucets, the tasks that transform the data (called **bolts**) presented as water drops and the arrows that show the flow of data from each task to the next in line. Data are always transferred on all exchanges using **tuples**. Tuples are indexed key value pairs that can be serialized on the sender's end and deserialized upon receipt.

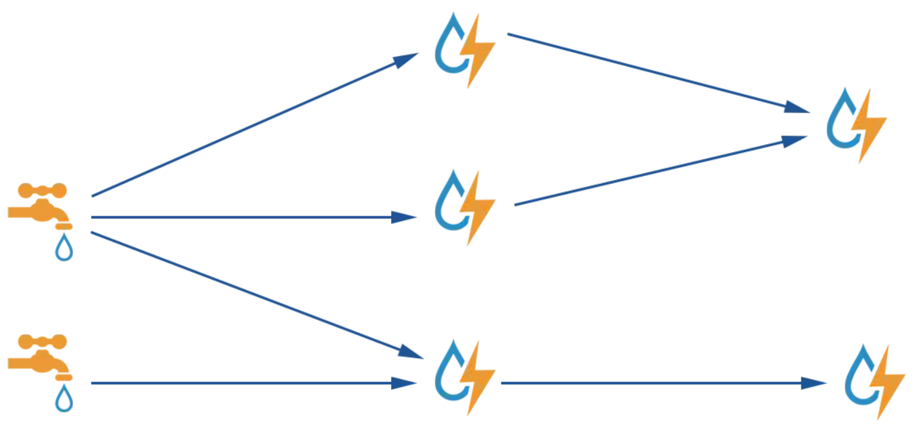


Figure 4.9 Abstract view of an Apache Storm processing topology.

Spouts Spouts are tasks that are polled with high frequency and have the role of picking up or receiving new data from external sources. In our case they maintain an open connection to the available message broker, receive updates asynchronously an add all new messages to an internal queue. This queue is then polled by Storm and all the objects found in it are forwarded for processing. A simple spout is available in Listing 4.1. In this listing we can identify two methods. `receive` is the one called when a new message arrives from the external message broker and the payload of the message is appended to the end of the queue. `nextTuple` is polled continuously by Storm to discover new payloads and forward them for processing as tuples.

```
public class StringSpout extends BaseRichSpout {

    Queue<byte[]> queue = new Queue<>();
    SpoutOutputCollector collector;

    Override
    public void receive(byte[] body) {
        this.queue.offer(body);
    }
```

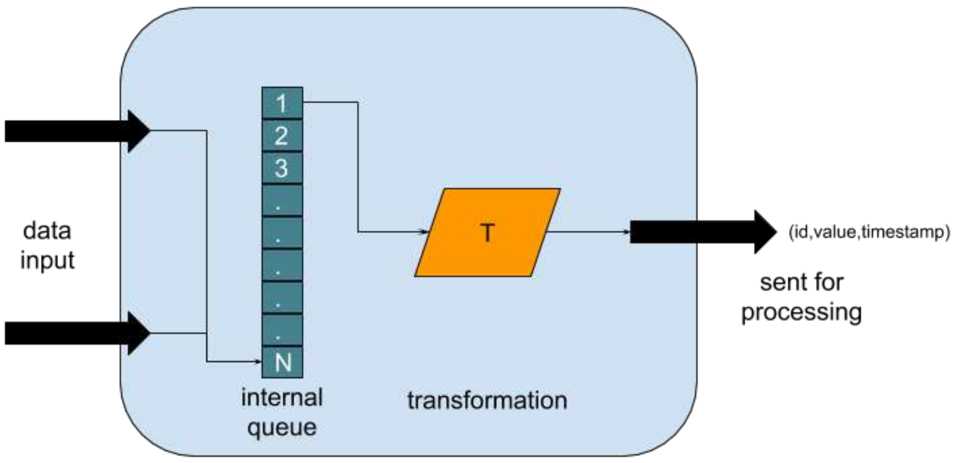


Figure 4.10 Graphical representation of a Storm Spout.

```
    Override
    public void nextTuple() {
        // Pop latest thing off the queue
        byte[] nextObj = this.queue.poll();
        if (nextObj == null) {
            // Silent return & sleep to avoid spamming the CPU
            Utils.sleep(5L);
        } else {
            String message = new String(nextObj);
            Values outTuple = new Values();
            outTuple.add(0, message);
            collector.emit(outTuple);
        }
    }
}
```

Listing 4.1 A simple Storm Spout

Bolts Bolts are tasks that are executed based on a received tuple. The goal of Storm is to break up every calculation needed into smaller tasks that can be specifically tuned and independently scaled for better performance. In Listing 4.2 we present an example of a simple bolt that calculates the maximum, minimum and average value of all the values received so far. Once the values are calculated the bolt creates an array of them and emits them for possible further processing or storage. In a slightly different implementation, this bolt could be replaced by 3 individual bolts that calculate the maximum, minimum and average value respectively. In that case and as the averaging is the most complex operation we could assign more instances running in parallel to this bolt in order to increase its throughput. Finally, a special type of bolt is what we call a **"data sink"** and is placed in the end of the processing topology. Its role is to receive the final tuple as a result and forward it to an external service for storage or further use. Such sinks are implemented or provided by Apache Storm for common databases such as MySQL[6], MongoDB[7] and others.

```
public class StatisticsBolt  extends BaseBasicBolt {
    List<Double> values = new ArrayList<>();
    Double min=Double.MAX_VALUE, max=Double.MIN_VALUE, avg=0;

    Override
    public void execute(final Tuple tuple,
        final BasicOutputCollector collector) {
        if (max < value) { max = value; }
        if (min > value) { min = value; }
        values.add(value);
        avg = avg(values);

        final Object[] output
            = new Object[3] { max, min, avg};
        collector.emit(new Values(output));
    }
}
```

Listing 4.2 A simple Storm Bolt

Time-based data analysis

That calculation engine needs to provide us statistics for different rolling timespans. In our case, we decided that we need statistics for each observed parameter for 5 minute, 1 hour and 1 day intervals in order to provide data in different granularities. We choose those values as they are the most appropriate for generating useful end-user visualizations and additional statistics. In each stage of the data analysis we maintain a buffer that contains the received values and uses them to re-calculate the aggregated values once a new value is received. This buffer, is actually a double ended queue where we add on the one side the latest value received and remove from the other side the oldest values that are now our of scope for the timespan we are interested in. Such a strategy provides us with 12 5 minute intervals for every hour (minutes 0-5, 5-10 etc.) and 24 hour intervals in each day. The final outcome of our calculation engine for each observed parameter is therefor a set of multiple queues with 48 values on each granularity. The 48 values were selected to provide a more extended view on the level observed that is meaningful in a visualization and a yet not memory consuming. An example of the resulting outcome from our caluclation engine is presented in Listing 4.3.

[6]https://www.mysql.com/
[7]https://www.mongodb.com

```
{
    keyName: "0013a20040b55ff0/0xe44/cur/2",
    latestTime: 1525335805765, //timestamp in unix time
    latest: 1836.4736842105262,
    minutes5: [
        1836.4736842105262,
        13241.596159571429,
        12218.473685185187,
        12009.260433888889,
        11984.854856666669,
        18481.245200307698,
        16743.183172,
        ... //48 values
    ],
    minutes60: [
        51290.65881952271,
        192838.42803524583,
        219732.656716715,
        71440.66035250713,
        57744.96627651688,
        72314.9046214876,
        74556.74366975091,
        ... //48 values
    ],
    day: [
        809954.4193813745,
        1778599.4514102784,
        1505638.3105379613,
        3287908.3959045573,
        1639454.896280735,
        1551001.504456223,
        1414542.523241626,
        ... //48 values
    ]
}
```

Listing 4.3 An example outcome of our caluclation engine

Forming a processing topology

Using the building blocks described in the previous section we are able to easily build a processing topology that is capable of handling all the data originating from the GAIA installation. We have implemented multiple topologies that focus on calculating aggregated values for different kind of data. Specifically we use topologies for power measurements, environmental data, weather data and events. For each of those data types a different kind of aggregation is required:

- Power Consumption needs to be totaled in each of the timespans we are interested in. The power meters provide us with an absolute value of power consumed every 30 seconds in Watt-hours. To calculate the total power consumed we need to calculate the sum of all values received in each timespan (e.g., in 5 minues or 1 day).

- Environmental data needs to be averaged in each of the timespans we are interested in. When we want to provide feedback for the temperature or relative humidity inside a building we are interested in the average value observed during time window, reducing the effect of momentary extreme values. Calculating the average value per 5 minutes throughout the day gives us the option to later on calculate also other metrics like the maximum or minimum values inside a day via simple calculations over much smaller data sizes.
- Weather data like the height of rain or the amount of radiation received from the sun requires also to calculate the total amount in each of the time period inspected.
- Event data are typically boolean values that indicate a certain condition inside the building. For example, motion detection sensor provide us with information about whether a room is occupied or not. Similarly, piezoelectric sensor can provide similar information for persons sitting on a chair. Such data can be handled in two different ways. The first option is to calculate the number of times such an event was recorded in each timespan as an absolute number. The second option is to calculate in what percentage of the timespan this event was observed. In our case, we choose to follow the second approach in order to have a common comparison level as all aggregated values will range from 0 to 1, with 0 indicating no events observed, 0.5 indicating the event appearing 50% of the time and 1 describing a constant event.

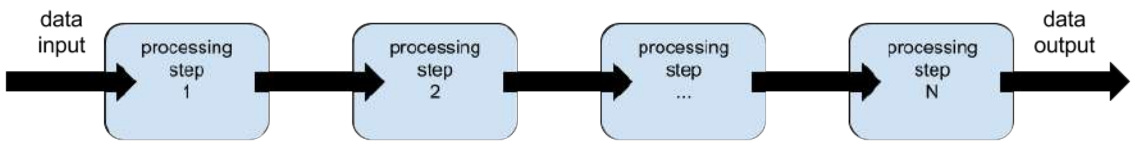


Figure 4.11 Graphical representation of a calculation spout.

To easily achieve these processing methodologies we use a set of interchangeable aggregation components in each Storm bolt that differentiate the final aggregate computed. These modules bind with the bolts on the setup of the topology as presented in Figure 4.12 resulting in a final topology that is presented in Figure 4.13.

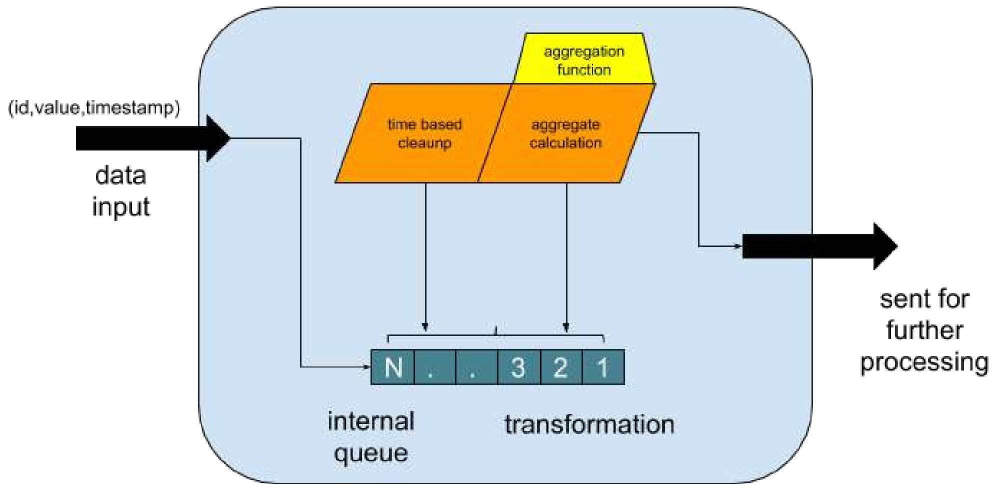


Figure 4.12 Processing bolt with the aggregation module.

An example of the resulting topology is available in Listing 4.4. As this is a production level topology need some additional components to be able to cleanup data (delete sensor spout) or adjust the in-memory data inside for timezone changes (time-shift spout and time adjuster bolt).

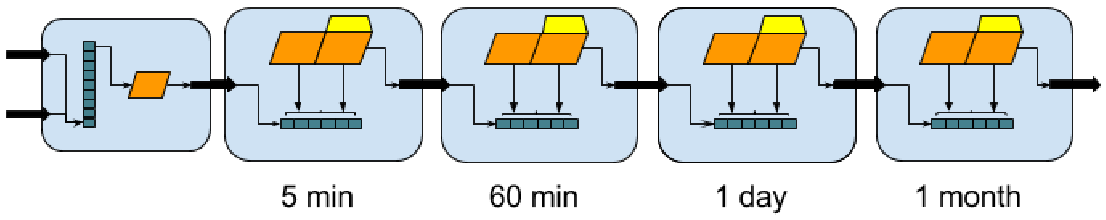


Figure 4.13 Final view of the calculation topology.

```
[spout] sensor measurements input
[spout] delete sensor input
[spout] time-shift input
[bolt] time adjuster
[bolt] aggregation bolt 5 minutes + aggregation type
[bolt] aggregation bolt 60 minutes + aggregation type
[bolt] aggregation bolt 1 day + aggregation type
[bolt] summary bolt
[bolt] data sink bolt
```

Listing 4.4 The final structure of a calculation topology

4.2.3 Data Storage

Once the calculated data are available they need to be stored in an appropriate storage schema that will allow for fast data retrieval and efficient storage. The options to implement such a solution we need to investigate the capabilities of the state-of-the-art solutions for data storage. What is available ranges from traditional SQL databases that provide strict and well defined schemas but tend to suffer when the data volume increases beyond a certain limit to NoSQL databases like Cassandra remove the issues of big data storage but may severely degrade the access times of time based queries. Another important solution that was introduced during the past years is the use of timeseries databases, like InfluxDB[8] and KairosDB[9] that are designed for IoT data and use in their backends NoSQL solutions for storing the raw data.

In our case, we implemented a two-stage storage engine that is designed to target both requirements. We use a MongoDB as the raw data storage and store an instance of the provided summary and the historical data for time-based requests. The first stage stores the summary that is extracted from the analytics engine as a single entry that can be directly retrieved upon request. This method gives as fast responses in the simplest queries that are requested by the majority of the users of the system. The summary presented in Listing 4.3 is what is used for this kind of storage.

The second stage decodes each summary received and stores its data as key-value pairs for each of the time interval they concern. This method allows use to easily query the historical data of our system by providing a set of keys that can be accessed in constant time from the database. In terms of scalability, MongoDB offers use the option to run on sharded clusters where data can be split and replicated for security and accessibility reasons. Listing 4.5 presents how the data are stored in the database. Each entry has a key value the is comprised of the unique identified for the sensor and a suffix that refers to the timespan in question in the format of year-month-day-hour-5minute interval indexes (YYYY/MM/DD/HH/M). Based on the depth of the second index we can understand in what timespan each entry refers to.

```
//entry for the average electrical current consumption
//on the 3rd  May 2018 08:20:00-08:24:59
{
  key: "0013a20040b55ff0/0xe44/cur/2/2018/5/3/8/5",
  value: 1836.4736842105262
```

[8]https://www.influxdata.com/
[9]https://kairosdb.github.io/

```
}
//entry for the average electrical current consumption
//on the 3rd  May 2018 08:00-08:59
{
  key: "0013a20040b55ff0/0xe44/cur/2/2018/5/3/8",
  value: 51290.65881952271
}
//entry for the average electrical current
//on the 3rd  May 2018 00:00-23:59
{
  key: "0013a20040b55ff0/0xe44/cur/2/2018/5/3",
  value: 809954.4193813745
}
```

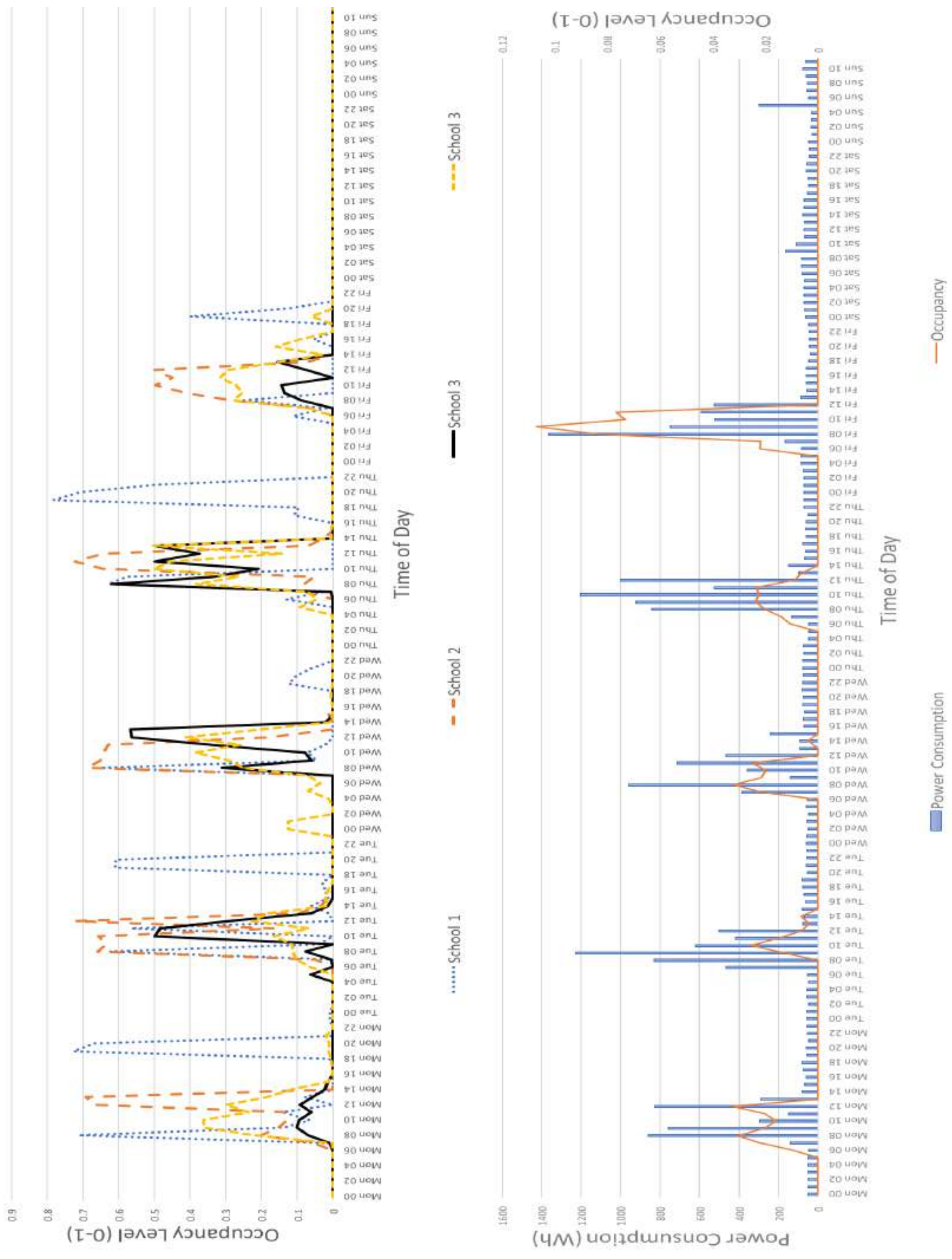
Listing 4.5 Examples of historical data entries in the MongoDB storage

4.3 Evaluation

The evaluation of the system's operation focuses on the following areas: (1) data processing, (2) data access, and, (3) data analysis and statistics. These areas can adversely affect the performance and perception of a system since users need rich data, easily and quickly accessible and real-time information to better understand how their actions affect the building usage. This is more crucial when the data is used in the educational context, i.e., during courses.

4.3.1 Data Analysis

To better present the operation and capabilities of the system, an analysis of the average weekly occupancy of 4 distinct school buildings during May 2017 is presented in this section. Remark that this analysis is similar to the work presented in [15]. Figure 4.14 (a) depicts the occupancy levels of the whole building, as an aggregated occupancy of all the rooms in which a smart motion sensors is installed. All four buildings are elementary schools that follow the same academic schedule with activities starting from 08:00 until 13:30. Remark that the building of "School 1" is also used by a technical school that is used in the afternoon. The data presented are ranged from 0 (no motion detected during this time interval) to 1 (constant motion was detected during the whole time period). This graph shows the actual active hours of the schools that are commonly ranged between 7:00 and 15:00 during weekdays. It also depicts the clear differences between schools of different levels, e.g., the case of `School 1` where a Technical school operates also in the afternoon for classes. Similarly, Figure 4.14 (b) presents the average power consumption of a school as it is measured by our system, versus the occupancy of the building. From the graph, it is clear that the school building consumes power mainly when it is occupied during the weekdays.



(a) (b)

Figure 4.14 (a) Four-week (May 2017) average occupancy levels in four different school buildings. (b) Four-week (May 2017) average power consumption and occupancy levels in a specific school building.

4.3.2 Data Access

Accessing historical data is crucial for building monitoring applications, e.g., when comparing historical data from different time spans and building areas. In such use cases, it is important that an IoT service is capable of providing these data without delays, independently of the targeted time interval. As discussed in [23], application response times larger than 10 s tend to make users lose their attention in the given task, while a 1 s response time is considered the limit for users that are freely navigating an application without waiting for the application's response. In that context, when presenting power consumption statistics, e.g., over the past year, it is important to be able to retrieve and present the stored values within one second, independently of the requested interval (latest values versus older values). Figure 4.15 and 4.16 present the average retrieval times for accessing historical data of a one month duration for the past 12 months, observing minimal differences in the access times independent of the period requested.

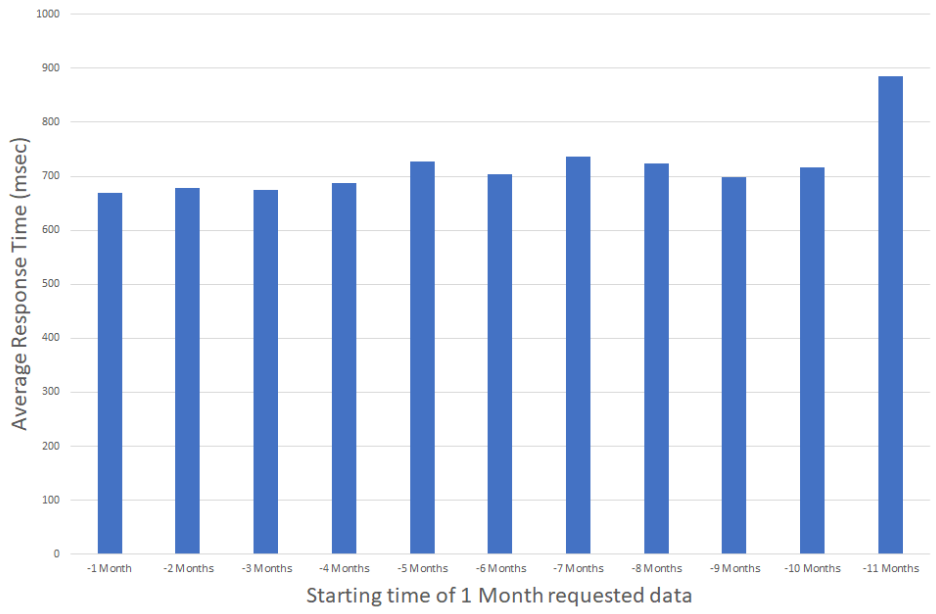


Figure 4.15 Average Response Time for accessing one month data for the past year (daily aggregated values).

Note that, based on the data available from the graphs, the system's response time is independent of the actual time interval while it is actually dependent on the amount of data requested. This is more clear in Figure 4.17, where response times tend to increase as the response times increase when time periods of more than 9 months of data are requested.

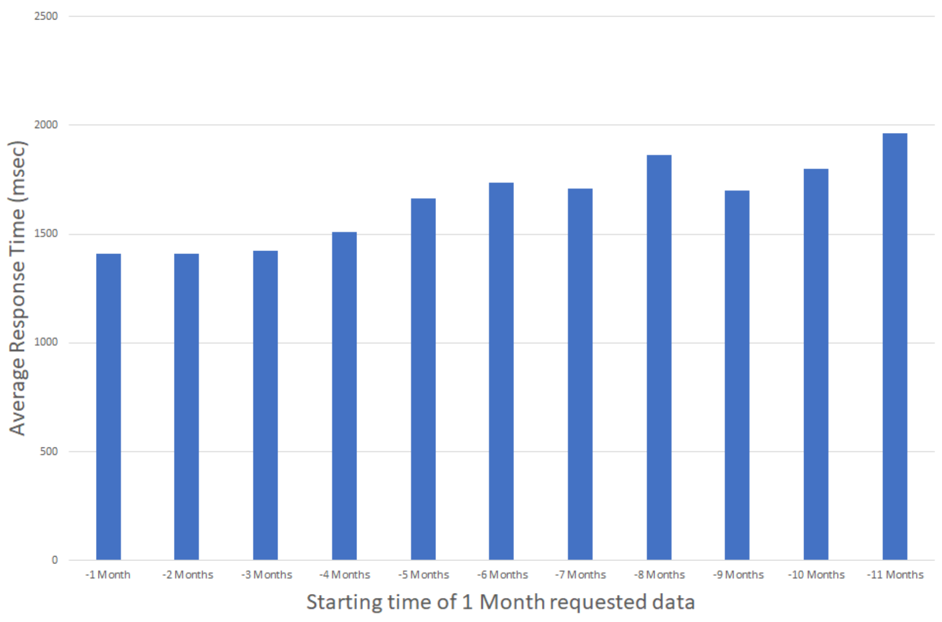


Figure 4.16 Average Response Time for accessing one month data for the past year (hourly aggregated values).

4.3.3 Data Processing

Another important characteristic for evaluating the system's performance is the load of data that the system is able to process at any given time. With the current setup, the fleet of buildings in the system produces an average of 25 measurements per second. The data processing topology currently runs on a single core virtual machine (on an Intel®Core™i5-3340 CPU running at 3.10GHz) with 4GB of RAM. With this configuration and setup, the system is capable of processing up to 500 measurements per second. To increase the number of measurements, the system can support two different options:

- Increase the computing power of the virtual machine, by assigning it to a more powerful host or giving it access to more resources from the host machine.
- Deploy a second instance of the processing topology that is capable of consuming the same number of measurements to reach the required data processing rates.

Based on the nature of sensors deployed, the input data require three different types of aggregation: (1) *averaging* for sensors such as temperature or relative humidity, (2) *total* for sensors such as rain height levels, and (3) *power consumption estimation* based on the electrical current values received from the installation. Each

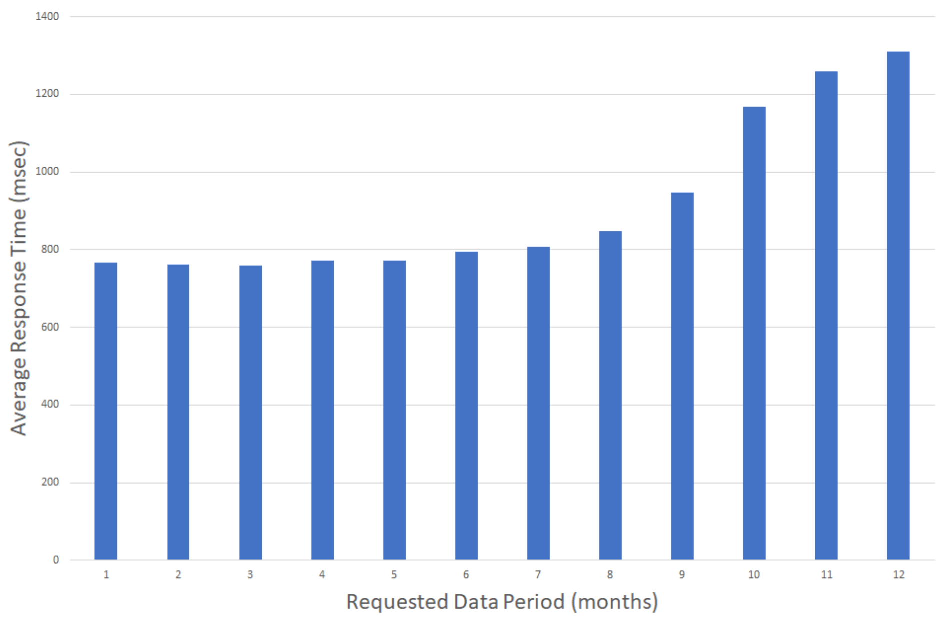


Figure 4.17 Average Response Time for variable time periods ranging from one to 12 months.

Aggregation Type	Execute Latency (ms)	Measurements (%)
Average	0.608	86.4
Total	0.799	0.9
Power Consumption	0.329	12.7

Table 4.1 Execution Latency statistics for the three different aggregation types used in our system.

type of processing requires a different type of aggregation processing and as a result has a different average execution latency, presented in Table 4.1.

4.3.4 Data Retrieval

The rationale behind the splitting of the storage service implemented in two distinct storage mechanisms lays in the fact that user interfaces are built using a very specific and predictable set of data that can be pre-calculated and pre-formatted, reducing the processing time needed to provide such data upon request. In our case, usage data shows that 90% of the requests arriving to the storage service concern the short summary that we are providing while only 10% of them require access to the full historical data storage. Table 4.2 shows how data requests for a single day of usage of the system are distributed on historical data and summary information.

Request Type	Requests (#)	Requests (%)
Summary	398705	90%
Historical	41374	10%

Table 4.2 Data Retrieval Statistics.

4.3.5 Data Representation

The final outcome of the GAIA project is based on providing a web-based building management system provided to all the schools participating in the project as well as teachers and students. This application was developed by a partner of the project based on the APIs provided from our system. A screenshot of the application is presented in Figure 4.18. The application offers a interface for users to browse the locations of their school building and show the historical data collected from the sensors. The data can be presented in 4 different granularities (5 minutes, 1 hour, 1 day, 1 month). The interface allow allows for the comparison of data from multiple sensors by retrieving the data from the provided data storage backend.

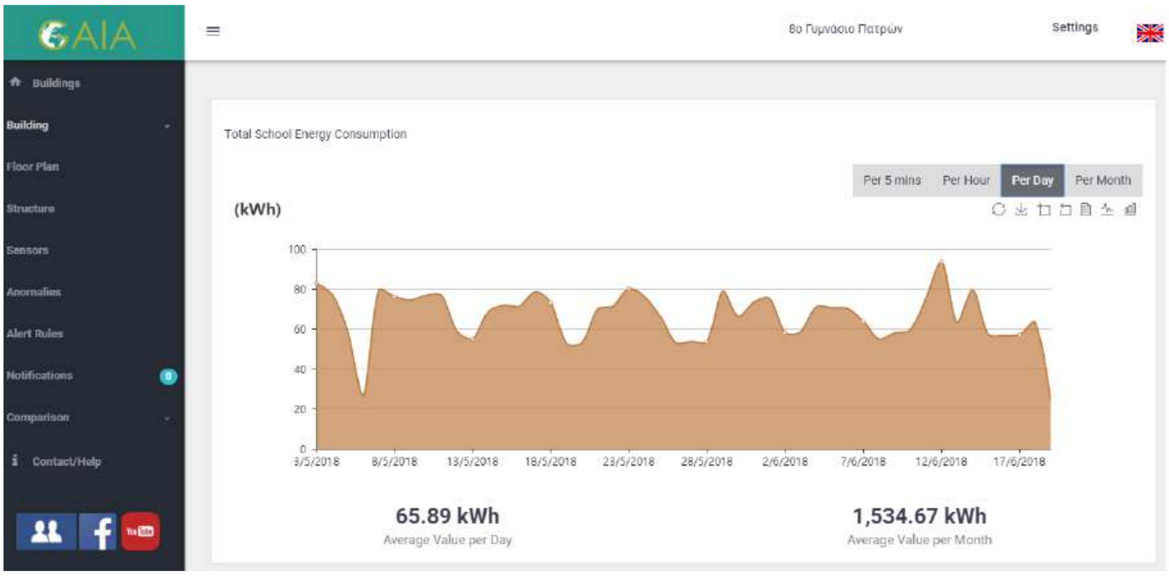


Figure 4.18 The GAIA building manager application Web Interface.

Chapter 5

Collecting Crowd-sourced Data

Keeping in mind what was presented in the previous chapters, we now need to step back and observe how such an installation could behave when the target is not to monitor a limited amount of buildings, but a whole city or even clusters of cities. Such an installation comprises potentially tens of thousands of sensors and can potentially monitor much more variable data types than the ones described so far. Additionally, in a direct comparison to the previous use case, we need to note that the sensor installations are now not owned by a single entity but by the different public services of the city or private entities, are probably heterogeneous by design and deployed in a much larger area than a single building or building complex. Based on this fact, different networking technologies need to be employed for the communication between the sensing devices and the central platform. Communication needs to be long range and in many cases over low power channels that operate on limited battery life ruling out solutions like `WiFi` or `IEEE 802.15.4`. Also, due to the large area that needs to be covered, mobile devices need to be included into the picture. Such devices could be installed on public buses, municipality vehicles or taxis, as well as be used by **volunteers** that move around the city.

In this chapter, we will study one large scale IoT installation that federates IoT infrastructures of multiple cities in Europe, OrganiCity and compare its architecture with the IoT installation of GAIA. OrganiCity is a Horizon2020 project started in 2015 that builds on existing Future Internet installations to build sustainable future city infrastructures and services in collaboration with local communities and enterprises. In this context, we investigate the behaviour of our system and showcase a platform for collecting data using volunteered Android Smartphones.

5.1 OrganiCity

Traditionally, cities have been the meeting point between societal challenges and technological innovation. The main ambition of OrganiCity is to "make the creation and design of technologies and services for cities more inclusive for any user of the smart city". It tackles the aspects of how smart cities can grow organically with the involvement of different stakeholders (citizens, communities, policy makers, activists, scientists, researchers, developers, city service and technology providers), and not be driven solely by engineering visions.

OrganiCity has developed an **Experimentation as a Service** facility, as a research and innovation environment for the co-creation of Future Internet enabled urban knowledge and services. The facility leverages emerging tools and technologies to provide mechanisms that allow users to extract knowledge from different cities, based on the data streams that are generated in the diverse urban ecosystems. To make this more inclusive, the facility delivers a set of tools and enablers to empower any user to be part of the co-creation process. Moreover, the facility provides various means for the engagement of the citizens participating in their validation.

Taking into consideration the resource constraints of cities in times of austerity, it is difficult to create new city infrastructures and/or develop new experiments. Setting up the OrganiCity facility on top of existing city web services permits city makers and service providers to address not only the exploration of new pathways for their services, but also to understand any societal implications in a rapid and more iterative manner.

5.1.1 The Experimentation as a Service model

As mentioned before, OrganiCity offers a flexible Experimentation as a Service (EaaS) framework, allowing researchers and developers of urban infrastructures to implement services by exploiting emerging Future Internet technologies. Research and experimentation on top of the facility should lead to the creation of sustainable, effective, and replicable smart-city solutions and urban technology developments that are successfully adopted by citizens, local communities and society as a whole.

To answer such goals, the design phase of the facility considered questions like how to provide data generated within the cities in a more public domain, whether existing infrastructures are reliable enough for data scale collection with citizens' devices, how creation of actionable knowledge from urban sensors can be scaled, how we can give incentives to and technically facilitate effective citizen-cityacademia–industry co-creation. Different stakeholders such as developers, data analysts, IoT solution manufacturers, urban service providers, activists, sociologists, economists, and citizens were involved, in order to extract requirements for the EaaS facility. As a result, OrganiCity has implemented a set of tools [20, 13] that empower citizens to be part of the co-creation process at different stages of the urban service life-cycle and provide different means for their participatory engagement.

Stakeholders

In this context, the OrganiCity facility supports different types of users, permitting them to manage the whole experiment life cycle [36] and offering a set of tools and enablers that make the co-creation and validation of new solutions more inclusive. It can be used remotely by any stakeholder within smart cities.

Depending on the activity that they intend to do, users can request different permissions, supporting the following roles: experimenters, participants, providers, site managers and facility administrators. *Experimenters* can implement their own solutions using a set of co-creation tools and conduct the corresponding experiment. To validate the solutions, they can invite other participants within the scope of the experiment (e.g., using an application developed by the experimenter). *Providers* are users who do not belong to any experiment, but want to contribute with crowd-sourced data to the facility (e.g., feed data from their own weather stations). *Facility administrators* deal with the management of the entire platform, being able to federate new urban ecosystems within the cities, assigning permissions and roles to other users, configuring the facility parameters and monitoring its activity. Finally, site managers are users belonging to the federated cities that can configure the information related to the urban services, the tags that can be used for annotating the data assets, and so on.

The use of the facility and its tools within the urban ecosystems will guarantee that communities of users can grow around emerging experimental technologies. Experimenters and service providers can leverage active contributors and early technology adopters. These communities will not only inform and contribute to the design of technology, but also provide a basis for exploring new business models derived from the emerging solutions.

5.1.2 Architecture

The OrganiCity platform uses the Microservices Architecture pattern [49]. Each service deployed in the context of OrganiCity is a standalone application that uses a central authentication service [56] (based on OAuth2.0) and provides either a user interface for end-users or a programmatic API (usually RESTful). This allows for different applications to be developed using tools, programming languages and frameworks that best suit each case (i.e., `node.js` for frontend applications, `Java` for back-end services). Additionally, the microservices pattern allows for better fine-tuning of the different parts of the infrastructure based on usage and thus provides better user experience and responsiveness to any number of clients, a major requirement for big cross-city IoT applications.

The overall architecture of the OrganiCity platform is presented in Figure 5.1. This figure presents the three layers of the OrganiCity architecture: (i) the Federation API that aggregates the data from the various smart-city installations, (ii) the core OrganiCity platform that is built around a central publish-subscribe service called Orion Context Broker and (iii) the services layer that communicates with the core platform with the EaaS API and contains both tools provided by OrganiCity as well as services developed by external users.

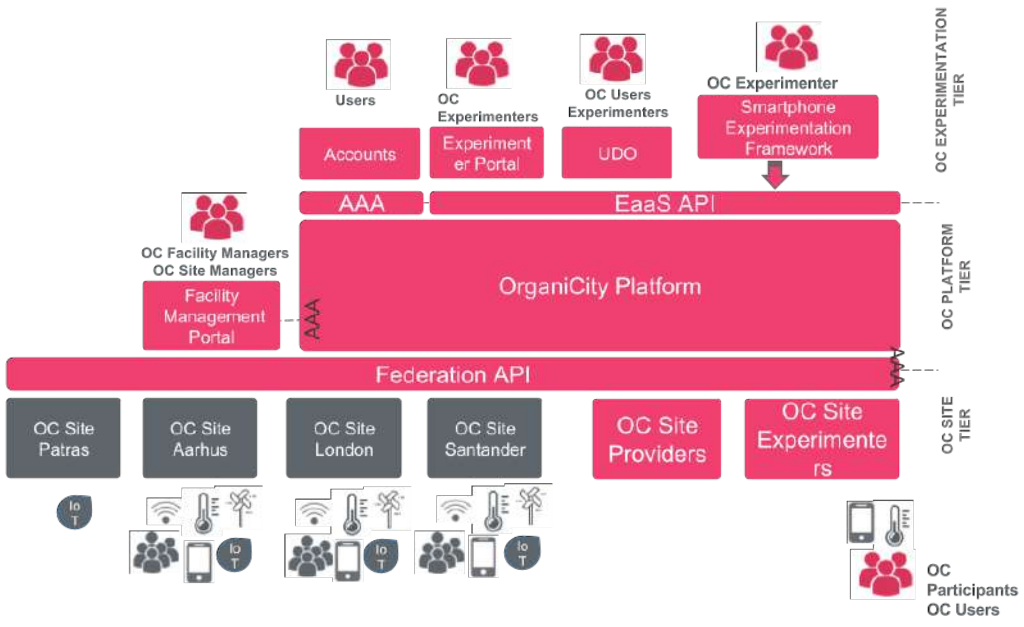


Figure 5.1 OrganiCity Architecture

If we compare directly with the solution presented and developed for the processing of the data originating from the GAIA project in Section 4.1, we can identify a similar pattern:

- The API mappers that are responsible for collecting the data from the sensing infrastructures of GAIA are called **Sites** in OrganiCity. These **Sites** are responsible for aggregating all information provided by the various sensing infrastructures and formatting to the internal data structures used by OrganiCity. To add a new data-source to the system a new Site has to be deployed feeding the data to the system as expected.

- A publish-subscribe broker is used to exchange the information of the platform between the various services of the platform. This broker does not only exchange the data collected in this case but also contains the metadata and context information we described in Section 3.2.

- Historical data and context information are stored in different infrastructures in both cases. This is done (in both cases) to reduce the strain of historical data queries to the context database, and thus increase the throughput of the system in total.

- External services and developed tools use well defined APIs to communicate with the core platform together with the a central Authentication and Authorization provider.

While the architecture of OrganiCity shows a great deal of similarity with our own design, some differences do exist that are mainly justified by the different requirements of the two projects:

- The entities stored in the Context Broker are organized in a flat single level domain using only their unique identifiers to differentiate them. This option was used inside OrganiCity as the different groups of resources are not of the same size and applying any kind of relational grouping could create non-balanced data groups that could affect the quality of services offered to its users.

- The latest value received for each attribute monitored is stored inside the Orion Context Broker. This was done to avoid the development of an additional Data Storage service in the context of the project as the historical data were stored in the original infrastructures of the federated Sites and the OrganiCity facility had no ownership on the data to replicate them.

From all the above, we can summarize that the design we followed in the development of our own service is valid and could support the operation of a platform even in the scale of OrganiCity.

5.2 Crowdsensing

While a number of research projects, like SmartSantander, focused on building large IoT infrastructures inside city centers and offering researchers and companies with a **testbed** to develop and test their systems and applications current mainstream smartphones can easily provide equivalent alternatives. Smart-phones currently are equipped with a number of integrated sensors and necessary networking interfaces to communicate with each other and with additional IoT devices like smartwatches, fitness trackers (i.e., Bluetooth LE, NFC, etc.) or custom made hardware (i.e., Arduino-based[1]). Also, the use of smartphones can be volunteer based, with participating citizens volunteering their devices to run simple tasks in the background while they commute inside the city during their everyday activities. Such a choice can significantly reduce the costs for a pre-deployed infrastructure or for a data-collection mechanism based on technologies like 4G or drive-by scenaria.

Using volunteers in such experiments and tasks can be tied with activist and citizen groups in order to provide a platform that investigates the human aspects of life inside modern cities and show a more human-centric experimentation inside metropolitan environments. The benefit for the volunteers can be incentive/prize based or based on a self-rewarding mechanism where citizens achieve personal goals, like walking (e.g., reaching a step goal each day), learning their city (e.g., visiting new areas in their city) or competing with their friends in game-like hunts for data.

In the rest of this section, we present a system that leverages volunteered smartphones to perform data collection experiments inside Smart-Cities in the context of OrganiCity, called Sensing-on-the-Go.

5.2.1 Sensing-on-the-Go

Following the concept of IoT experimentation testbeds, where **sensing devices** execute **applications** and collect **data results**, in our case, **smartphones** participate in **campaigns**, which produce **measurements** that are then made available through OrganiCity 's services. As described before, citizens participate in these experiments

[1]https://www.arduino.cc/

transparently during their commute or leisure activities. The overall goal of the platform is that utilizing their smartphones does not cause them any disturbance in their everyday life; like reducing the battery life of the smartphones or engages them in any continuous activities that may distract or annoy them. To build such a system we utilize two discrete components:

- A server/portal component dedicated to submitting, monitoring and managing the execution of data collection **campaigns** that is used by users that want to experiment inside a smart city.
- A **smartphone** component dedicated to the execution of the campaigns on smartphones and the collection of the **results** to be sent back to the server component.

The portal component is designed for users that intend to organize and run a campaign inside the city. This portal offers them the basic interfaces to create and organize their campaigns. The can use it to define the type of data they wish to collect, information that is going to be used to promote their campaign, time and spatial restrictions on the execution of the campaigns as well as any incentivization schemes that may be used. Once submitted, the campaign appears on the smartphone of each volunteer and is available for participation. Collected data are automatically gathered and stored in the back-end of the service without any further action needed by the volunteer or the experimenter. Volunteers also have the ability, at all times, to stop or pause the execution of a campaign, or even opt out of the entire process. The organizers of the campaign can view the final results from the same portal in near real-time, in a fully anonymized format without any critical personal information being revealed for the participating users.

The Smartphone component is based on a application used in the SmartSantander EU project project [53]. Its functionality was redesigned and extended to support newer Android versions and functionality. It takes advantage of the the ability to dynamically launch and register new Services inside an already installed Android application. These new Services do not necessarily belong to the application itself, but are available as third-party applications installed separately on the same phone via official application stores. In other words, the functionality of the main smartphone application can be dynamically updated/augmented by downloading and installing new Service Applications directly from the Google Play Store™(Figure 5.2) similarly to the way plugins can be installed inside modern web browsers. Each new Service Application is based on native Android code that is signed by and distributed by

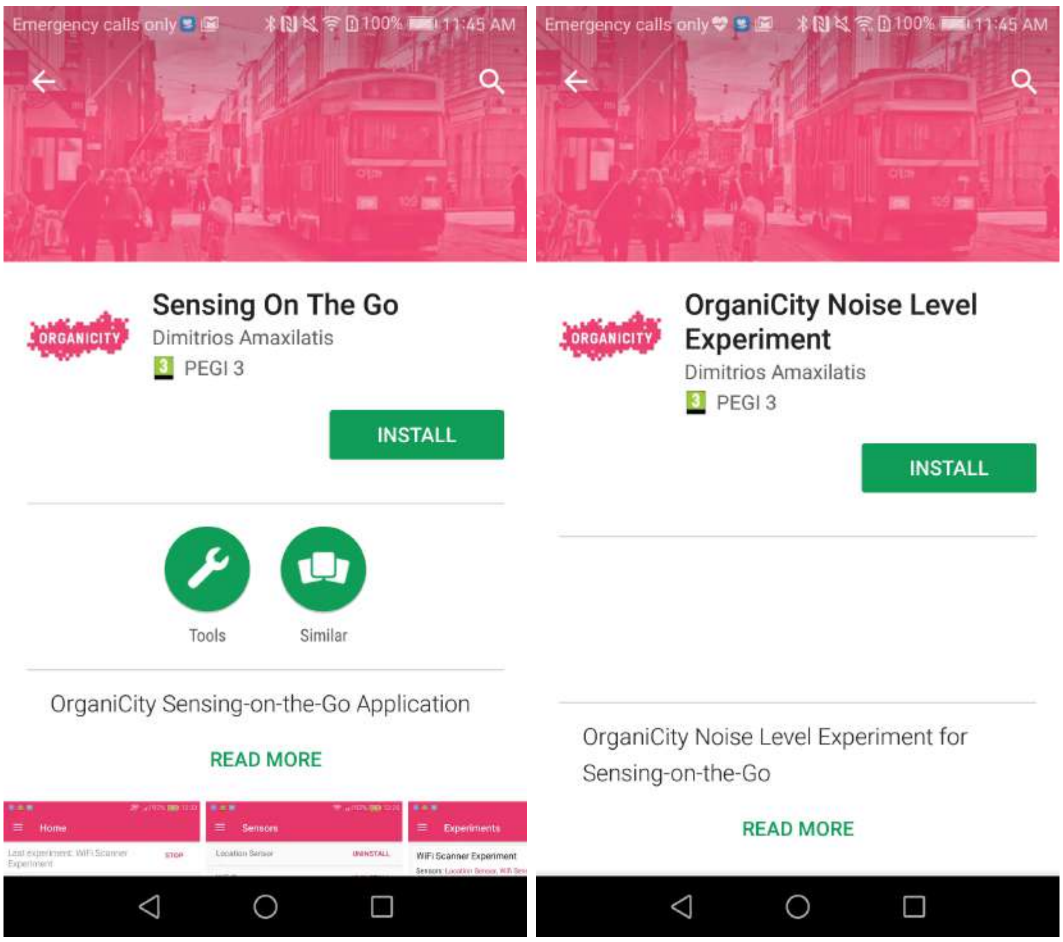


Figure 5.2 Main smartphone and service application in Google Play Store.

its developer. The use of this multi-application schema allows us to run different campaigns on demand using a single central application that orchestrates the experimentation without the need for constant updates every time new functionality needs to be added to the system. Also, functionality that is not provided by the main application can be implemented by the users of the platform and be distributed by its original developers for security, privacy, and intellectual property protection reasons.

To gather data, experimenters can use all the available sensor components available in current Android smartphones, while also maintaining a certain degree of transparency, in order to allow volunteers understand what kind of data they are collecting. Such sensors include (but are not limited to) temperature, humidity, noise, walking steps, location or data generated by accessing interfaces like `WiFi`, `Bluetooth` or `NFC`. A number of basic sensor components are provided by Sensing-on-the-Go as template Service applications for experimenters to use, but additional ones can be implemented based on the specific needs of a campaign. The overall architecture of the system is presented in Figure 5.3. In it we can see how the application's components in the smartphones of the volunteers and the cloud interact with each other and the Google Play Store™ to have access to the sensing modules needed, as well as the services of OrganiCity. The OrganiCity services are used to manage the general information of the data collection campaigns and to store the collected data and present them on the tools of the project.

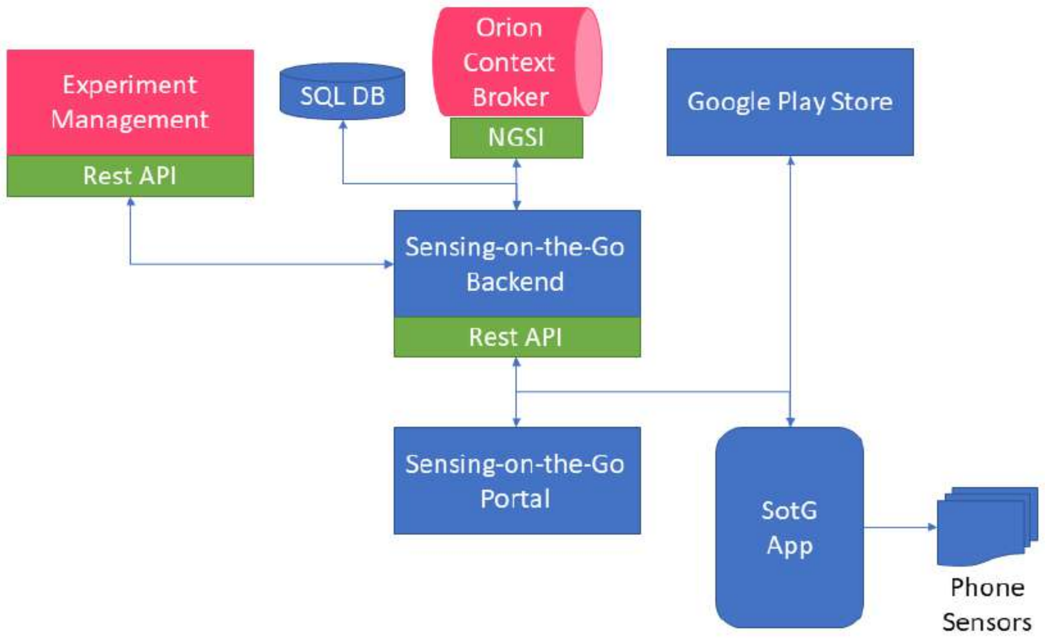


Figure 5.3 Sensing-on-the-Go overall architecture

Defining a campaign

To begin with the setup of a campaign, experimenters need to provide some basic information that describes the main goals of the campaign, the usage intended for the data collected and a link to the outcomes of the campaign. All data provided in this step are made available to volunteers, so that they clearly understand what is executed on their phone and what data are collected and made available to the experimenter.

The next step concerns the designation of the spatio-temporal characteristics of the experimentation. This includes the following restrictions:

- on where the campaigns are going to be executed.

- on when the campaigns are executed.
- on the amount of data needed for the success of the campaign.

The experimenter firstly defines the areas in which the volunteers are asked to move and collect data. The experimenter freely designs polygons of interest without any limit on the number, size or location (Figure 5.4). The polygons can also overlap with each other as the user can later on place restrictions on data collection that create different data characteristics. For example, an experimenter can select the whole city as a big polygon and then draw inside smaller areas of major interest or set different time constraints.

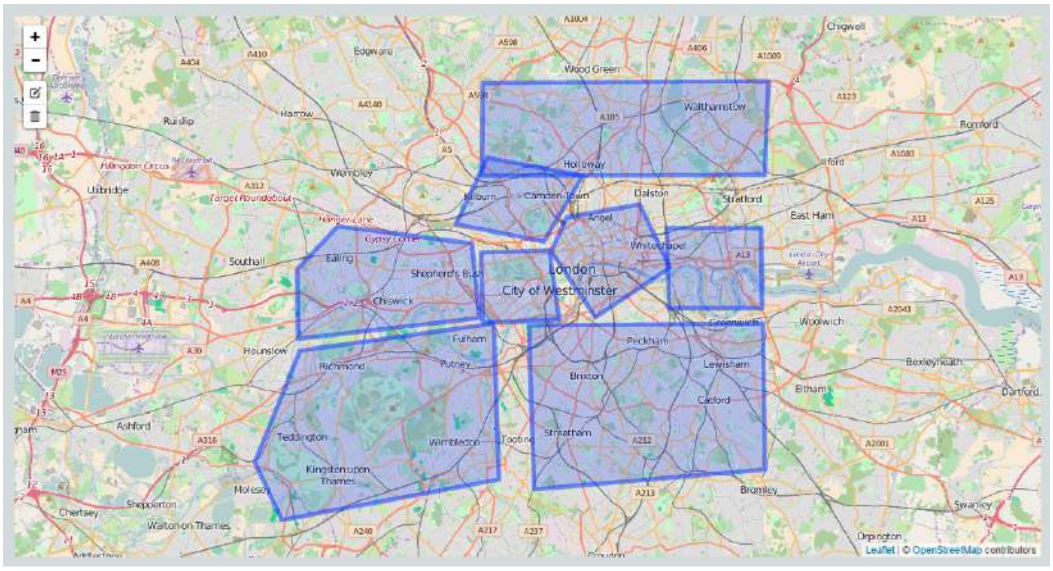


Figure 5.4 View of area polygons defined for a campaign in the city of London.

Based on the areas defined, the experimenter now can define date and time constraints on the data acquisition. Apart from the basic time period of the whole campaign (start and end date), experimenters are able to define discrete time periods of interest during the day (i.e., there is little interest in values between 2-6 a.m.). This is used for example to focus the data collection during office or commute hours, ignoring post midnight periods when traffic in the city in lower in a campaign interested in commute information.

Similarly, data volume restrictions can be defined on each area. Such information can be then used to mark them as complete and encourage volunteers to move in the rest of the areas defined, especially when incentives or rewards are provided.

All these restrictions help in a more homogeneous execution of the campaigns as not only is the geographical area of execution important, but the temporal and data volume plain is also taken into account. Apart from the ability to define these restrictions, the system provides feedback options and interfaces to monitor the progress/current state of execution of the campaigns and the degree to which the constraints are fulfilled during the execution; e.g., the percentage of the requested data already gathered.

Writing the code In order to develop a new sensor plugin, users of the platform need to develop a new Android application based on the provided template [2]. In this application, they need to edit only a single Android Service that collects the data from the smartphone and prepares them for the platform. This service, uses a provided SDK (called `OrganiCitySDK`) to communicate with the base OrganiCity smartphone application. Listing 5.1 showcases a service for collecting temperature sensor measurements, from the integrated temperature sensor, and forwarding them to the Sensing-on-the-Go application. The code required to develop the application is pure Android, and in most cases limited to a less than 100 lines.

```
public class TemperatureSensorService extends Service
    implements SensorEventListener {
  ...
  Override
  public void onCreate() {
    mSensorManager = (SensorManager)
    getSystemService(Context.SENSOR_SERVICE);
    mSensor = mSensorManager.getDefaultSensor(
        Sensor.TYPE_AMBIENT_TEMPERATURE);
  }

  Override
  public int onStartCommand(Intent intent, int flags, int id) {
    if (mSensorManager != null) {
      mSensorManager.registerListener(
          this, mSensor, SensorManager.SENSOR_DELAY_NORMAL);
    }
    return START_STICKY;
  }

  Override
  public void onDestroy() {
    if (mSensorManager != null) {
      mSensorManager.unregisterListener(this);
    }
    super.onDestroy();
  }

  Override
  public void onSensorChanged(SensorEvent sensorEvent) {
    temperature = sensorEvent.values[0];
  }
```

[2]https://github.com/OrganicityEu/sensing-on-the-go/tree/master/sensors/ExampleSensor

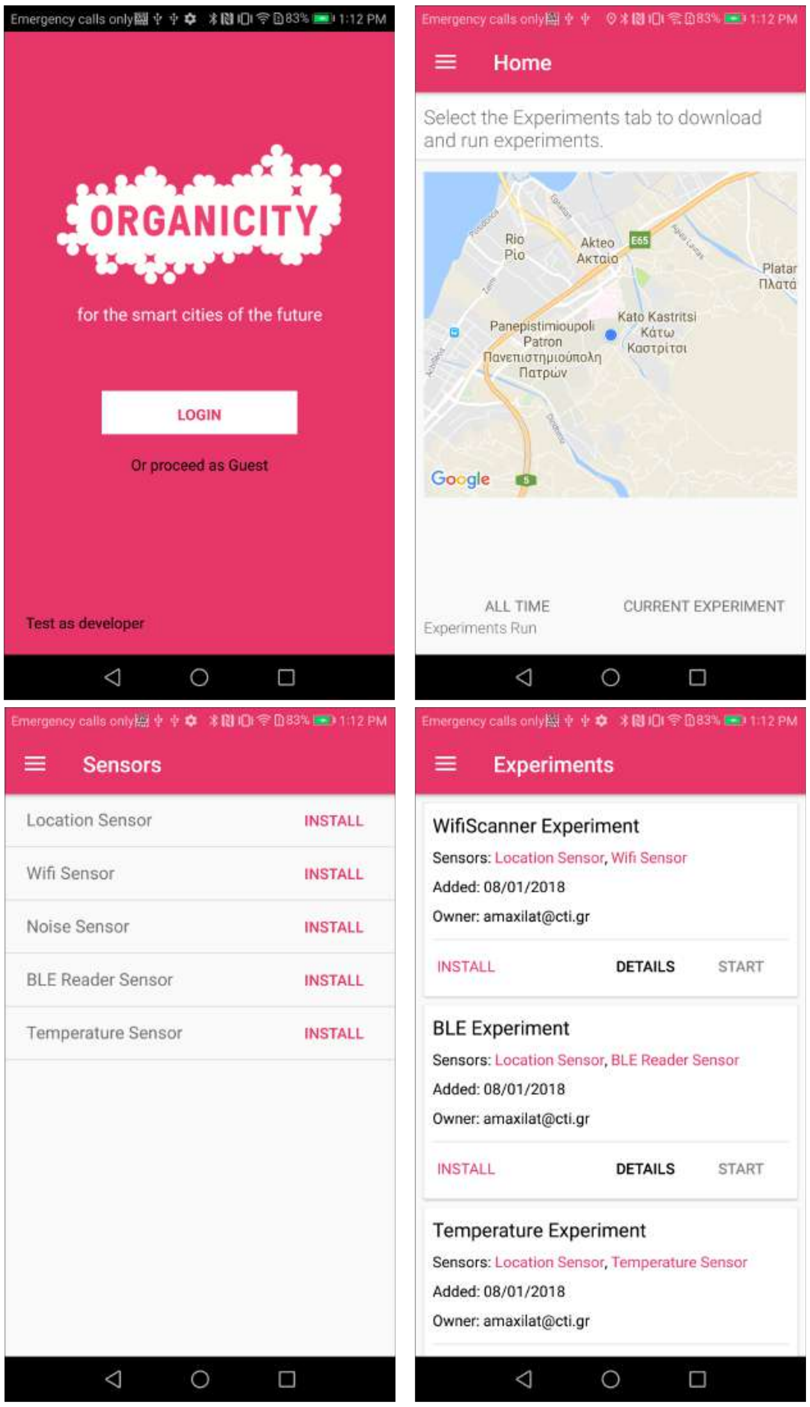
```
  public void publishResults() {
    JsonMessage info = new JsonMessage();
    info.setState("valid");
    List<Reading> r = new ArrayList<>();
    JSONObject jsonObject = new JSONObject();
    jsonObject.put(CONTEXT_TYPE + ".Temperature", temperature);
    r.add(new Reading(
        Reading.Datatype.String, jsonObject.toString(),
        CONTEXT_TYPE + ".TemperatureSensorService"));
    info.setPayload(r);
    mRemoteCallbacks.handlePluginInfo(info);
  }
}
```

Listing 5.1 A Template Sensor for Sensing-on-the-Go collecting temperature measurements

Participating in a campaign

Volunteers are able to install the Sensing-on-the-Go smartphone application through the Google Play Store™ as all other Android smartphone application. Once installed, the application prompts them to register for the Sensing-on-the-Go platform through OrganiCity in order to take advantage of all the benefits of a registered user, and potentially receive rewards from the data collection process if incentives are used. This option is not mandatory, as they can also continue as anonymous users without any personal information provided to the system. Selecting, enabling and participating in crowdsourcing campaigns is done with the touch of a single button. Participants have a list of all the available campaigns along with the descriptions the organizers provided in the setup of the campaign presented before. By clicking the **start** button, the applications notifies them of any additional Service applications that need to be also installed from the Google Play Store, and redirects them there to complete the installation. Once all the pieces are available, the campaign can start and data collection is done in the background. Additional data on statistics for the data collection procedure are available in the home screen of the application in real time (Figure 5.6). These statistics concern both the current campaign executed and the overall statistics for this specific volunteer, if an account has been connected to the phone.

For a higher level of privacy, volunteers can at any time disable the data collection or their participation in all campaigns, using a “power-off” button available on the phone. This button kill all services running in the background and any access to the location of the device or the other sensors of phone.



Select the Experiments tab to download and run experiments.

Location Sensor INSTALL
Wifi Sensor INSTALL
Noise Sensor INSTALL
BLE Reader Sensor INSTALL
Temperature Sensor INSTALL

WifiScanner Experiment
Sensors: Location Sensor, Wifi Sensor
Added: 08/01/2018
Owner: amaxilat@cti.gr

BLE Experiment
Sensors: Location Sensor, BLE Reader Sensor
Added: 08/01/2018
Owner: amaxilat@cti.gr

Temperature Experiment
Sensors: Location Sensor, Temperature Sensor
Added: 08/01/2018
Owner: amaxilat@cti.gr

Figure 5.5 Sensing-on-the-Go Android application views: login screen, home page, sensors view and campaigns view. The home page show the current location of the volunteer, the area of interest and statistics on the data contributed. The sensors view shows a list of all the available sensor plugins to be installed, while the campaigns view shows the active campaigns the volunteer can participate in.

Monitoring a campaign

After launching a new campaign, experimenters can monitor the progress of a campaign in near real-time using the Sensing-on-the-Go portal. The term **near real-time** is used to define that data are not made available upon receipt, due to the fact that they need to be anonymized. This is done in batches, to reduce the chance of any personal data (like the location of the user) being exposed. Therefore, the experimenter can view any time aggregated data for the parameters monitored over two main visualization methods, using graphs and on-map markers.

The graphs mainly refer to the coverage of the targets set by the experimenter on the campaign. The coverage can be spatial or temporal, as the experimenters can view how the data collected are distributed over the days of the experimentation or the hours of the day. This quantitative analysis of the data received is intended not to analyze the actual data of the campaign but to give the experimenter a better idea of how volunteers collect data and participate in the experimentation.

Similarly, the portal offers the option to display the collected information on top of maps, either as points or as a heatmap. The map view offers the option to visualize the amount of collected data based on location. In the first option, points, the experimenter can view for each point of the map the data of the specific measurement, if only one, and the average value, if it is possible, for multiple measurements. For the heatmap view, experimenters can view the hot-spots for data collection as areas with more measurements are painted with a color closer to red while areas without any measurements are painted with blue.

Examples of both visualization methods are provided in figures in the Evaluation section. Options to export the data collected are also available to experimenters in multiple formats, like JSON or CSV so that they can be parsed later on for the appropriate analysis as an outcome of the campaign.

Real World Evaluation

Sensing-on-the-Go was testing in the wild during the two open calls for experimentation of OrganiCity. It was used by 4 independent experimenter teams to perform data collection over these open calls and by volunteers that participated in these campaigns. Also, to verify the operation and performance of the system, we also conducted internally a number of campaigns prior and in parallel with the OrganiCity open calls mainly in the cities of Patras, London and Santander. We here present one of these campaigns that was organized simultaneously in the 3 cities to collect data for the WiFi access points that can be detected in the city streets. Such data can be used to develop a system that provides localization information without the use of GPS inside the city or assist GPS positioning as proposed in recent literature like [43] or [74].

To organize this campaign we used the following Service applications provided by the Sensing-on-the-Go platform:

- **Location Sensor** provides the coordinates of the smartphone at the time of the data collection. Such data are provided in latitude, longitude format based on any source that is already available on the smartphone (GPS, network provided or fused).

- **WiFi Sensor** scan periodically for WiFi Access points and provides a list of them together with information about their type (2.4G or 5G), security if available and SSID.

Both data points are merged in a single measurement and uploaded to the back-end of Sensing-on-the-Go in regular intervals of 60 seconds.

The campaign was available in all 3 cities for a total of 5 days. Participants were mainly members of the OrganiCity development team as this was one of the first campaigns performed to validate the behaviour of our system. A total of 14 smartphones provided data during the campaign in all 3 cities and contributed a total of more than 7500 measurements.

For the experimentation process 13 regions were defined in all 3 cities. Figure 5.4 shows the areas defined for the city of London. As London is the largest of the 3 cities, 8 areas were defined. 3 more were defined in Santander and 2 in Patras. Figure 5.6 show the areas where data were collected in the city of Patras. Based on the view of the map, the data were mainly collected around the central streets of the city and coastal area that citizens use for walking and jogging. The heatmap on the other side available in Figure 5.7 show that there are two cells where the most measurements were collected, with both being city sqares with shops and pedestrian alleys.

Each participant contributed an average of 500 measurements to reach a total of 7634 measurements. Some of the points were gathered outside the areas of interest defined in the campaign. The application is not too strict in enforcing the spatial limits of the data collection and allows for some data to be collected in the borders of the areas. These data are made available to the experimenters in a separate view to assist the organizers in fine-tuning the spatial limits of the campaign.

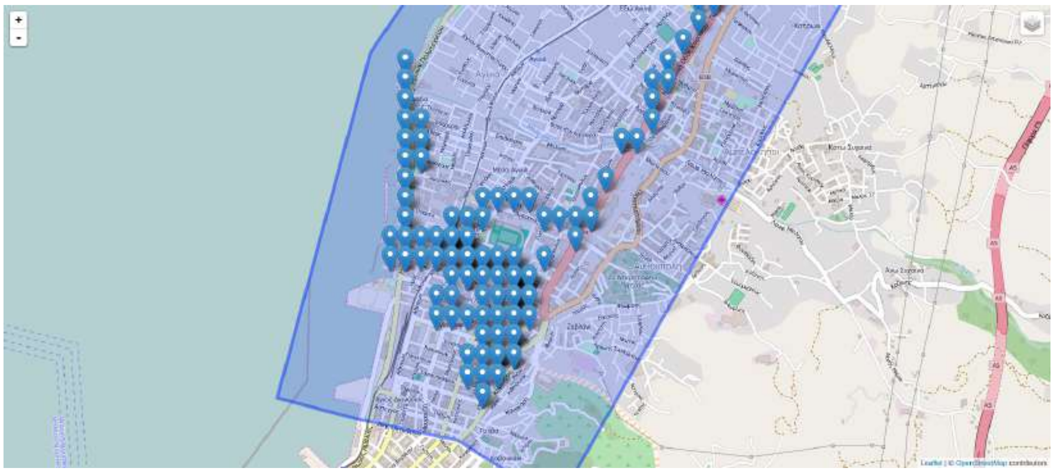


Figure 5.6 Pointmap view of the measurements for the WiFi campaign.

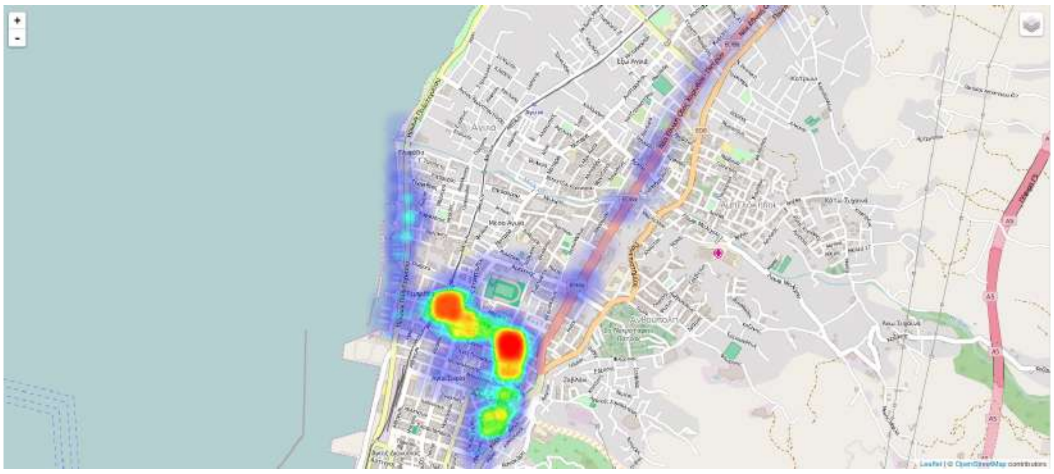


Figure 5.7 Heatmap view of the measurements for the WiFi campaign.

Table 5.1 briefly presents the statistics of the campaign as they are extracted from Sensing-on-the-Go.

5.3 Extracting Knowledge from IoT Data

Although technologies like the ones presented so far in this thesis allow us to collect data from IoT installations and smart-city environments, essential answers are yet to be found revolving around two central questions:

- **what** do we do with all of these data collected?
- and how can we easily **make sense** out of them?

Cities	3
Participants	14
Regions	13
Experimentation Days	5
Measurements	7634
Average Completion Rate	40%

Table 5.1 Statistics for the WiFi Experiment performed for the evaluation of the Sensing-on-the-Go platform.

One answer to both questions is the extraction of **knowledge**. The term **knowledge** refers to something actually useful, going beyond a technology demonstrator. It could be vaguely described as something that provides usefulness to citizens with or without their involvement in the process. Essentially, the questions lead us to as ourselves how do we actually make a city smarter, and what is the definition of a smart city itself.

Part of the answer to this question lies in creating more **useful** information out of raw sensor data or other kind of data representing observations of the urban environment. For example, certain events generate data reported by the city sensing infrastructure, but are, more often than not, missing an appropriate description or **annotation**. Consider the case of a traffic jam inside the city center; it generates sensed values in terms of vehicles' speed, noise and gas concentration. Moreover, in most cases, multiple devices or services, while missing useful correlations in the data streams, report such values. We believe that adding data annotations to smart city data through Machine Learning technologies, or crowdsourcing mechanisms, can help reveal a huge hidden potential in our path towards real smart cities.

5.3.1 Annotating Collected Data

Data annotation in smart cities presents us with a set of key challenges that need to be addressed before such solutions are used at large. In this context, privacy and overall security issues are a central challenge. Consider the case of a volunteer, annotating noise level measurements along his daily commute in a smart city based on proximity to certain events. Even in such simple scenarios, anonymization techniques should be used to ensure that neither personal data, nor the volunteer's interactions are revealed.

Another important issue is the correlation of different types of smart city data that can potentially point to the same event. In other words, how to facilitate knowledge extraction through such data. We currently have data produced by IoT infrastructure installed in cities, however, there is relatively small research focus on discovering relations between these data, e.g., if noise level readings are related to social data referring to a live concert or some other event.

Moreover, smart city data are unreliable by nature either they are supplied by humans or IoT infrastructures. In some cases, they can even be malicious, as the sensing infrastructure could be easily accessed and influenced or suffer from hardware failures and malfunctions. In most cases, the hardware utilized aims for large-scale deployments (i.e., has to be cost-effective), thus being not so accurate or hard to re-calibrate after the initial installation. Additionally, environmental conditions, e.g., excessive temperature or humidity, may have an effect on the sensitivity of the sensing parts.

End-user engagement in the process of data annotation and knowledge extraction is another major challenge. Users' contribution is twofold: end-users can contribute to a smart city system by crowd-sourcing or by adding annotations. Although most current crowd-sourcing platforms utilize a desktop or web interface, it should not be limited to that. It can also be performed through smartphones and be incorporated to the user's everyday life. The interaction of end-users through such a tool could help relate in a more personal way and help maintain the interest in continuous participation. Moreover, annotation of events or sensed results could be more interactive and focus at users, or even user groups, near the actual space of the event in question. Smart city facilities usually integrate a large number of data sources of various types sharing observations for environment, air quality, traffic, transport, social events and so on. These data sources might be static (they are not streaming data and have a fixed value until they are updated by an offline process) or might be dynamic (streaming data constantly). Building a taxonomy on this multithematic environment is not straight forward as some subcategories of tags might be shared between different types of data sources and other might be orthogonal. Moreover, as the dynamic data sources have a temporal dimension, annotations might characterize the overall behavior and observations of the data sources or observations falling into a specific time interval.

Furthermore, as data sources might be mobile (e.g., an IoT device on a bus or a smartphone) an annotation might characterize a specific location inside the city and for a specific time interval. Embedding in the taxonomy such spatiotemporal characteristics introduces new requirements and extensions to the traditional methods. Standards like W3 Web annotation data model and protocols do not cover sufficiently these requirements. Finally, implementing machine learning algorithms suited to smart city data and real-time processing is another major challenge. Handling citywide data introduces additional complexity, especially when considering relations between different data types and sensing devices. Current mobile devices have enough processing power to handle a broad set of use-cases, especially when dealing with data from integrated sensors (e.g., [19] uses on-device processing to classify urban noise sources). This could also be utilized as a means to enhance privacy, since processing would be performed locally, without requiring sensitive data to be uploaded to the cloud.

In the rest of this section, we present the design and implementation JAMAiCA (JAva MAChine Annotation). JAMAiCA is a service destined to aid smart city data annotation through crowdsourcing and machine learning techniques, like classification and anomaly detection. It is currently part of the OrganiCity project ecosystem, operating in real world conditions, analyzing the data submitted to the platform. On the one hand, it aims to simplify the creation of more automated forms of knowledge from data streams, while on the other hand it serves as a substrate for crowdsourcing data annotations via a large community of contributors that participate in the knowledge creation process.

5.3.2 JAMAiCA

JAMAiCA is composed of two distinct software components. The first is an annotation taxonomy and storage engine with the appropriate programming interfaces to create, update and delete annotations of smart-city assets. The second component is a streaming data processing engine that is capable of analyzing incoming data using machine learning techniques and store the results in the first component.

Annotation Component

The Annotation Component is responsible for maintaining a directory of all possible annotations in the form of **tags**. Tags are simple indicators of the annotated parameter, similar to the way tagging is performed in photos in social networks, or the use of hashtags in social status updates. Tag domains are created as collections of tags (e.g., `high`, `normal` and `low`) with a similar contextual meaning. Tag domains can be generic as those mentioned before or more application specific (e.g., the tag `contains a beach` for images). Users of the system can either select one of the tag domains already available, or create their own specifically for their application. Annotations are stored with additional information like numeric or text values. These entries can be user comments, a number that describes the abnormality of an observation, or a confidence indicator.

Model The underlying data model of the Annotation Component and the relations between its entities are described in the rest of this section. The schema is comprised mainly of the following entities:

- **Tag**: Tags represent the actual annotation labels to be used by end-users for the annotation process. The entity is described in more details in Table 5.2.
- **TagDomain**: TagDomains represent collections of **tags**. Usually a tag domain is associated with a user or a real-world phenomenon that can be identified by the data (e.g., temperature) specifying which tags can be used to characterize it. The entity is described in more details in Table 5.3.
- **Annotation**: Annotations are relationships between the **assets** of a smart-city and **tags**. The entity is described in more details in Table 5.4.
- **Asset**: Assets are sensor points of a smart-city that can be annotated. The assets are not stored in the internal database of the Annotation Service but referenced by the added **annotations**. This is done in the context of OrganiCity but in a general version does not to be the case.

The parameters of each entity are available in more detail in the following tables. Figure 5.8 shows visual examples of **tags** and **tagDomains**.

(a) Fields

Name	Data Type	Required	Description
`id`	numeric	Y	The unique id of the Tag in the internal DB
`name`	string	Y	A user friendly name for the Tag
`urn`	string	Y	The unique id of the Tag available to users
`user`	string	Y	The id of the user that added the Tag

(b) Relationships

Name	Target	Description
`HAS`	`TagDomain`	Every Tag must be a member of a TagDomain
`TAGGING`	`Annotation`	Every Tag can characterize one or more Annotations

Table 5.2 Tag entity fields (a) and relationships (b).

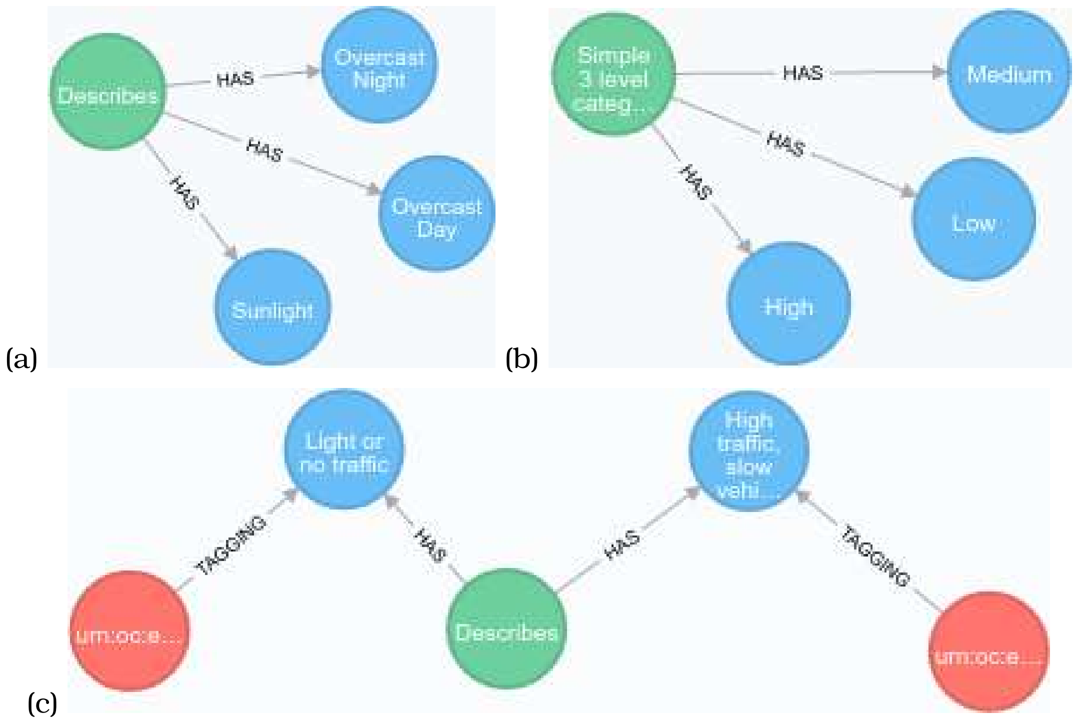


Figure 5.8 Visual representations of `tags` (blue), `tagDomains` (green), `annotations` (orange) and their relations. (a) TagDomain: Luminosity levels - Tags: Overcast Night, Overcast Day, Sunlight (b) TagDomain: Simple 3 level categorization - Tags: High, Medium, Low (c) TagDomain: Traffic Levels - Tags: Light or no traffic, High traffic, Assets: urn:oc:e...:1, urn:oc:e...:2

Implementation The Annotation Service is implemented as a standalone Java web application. It is built using the Spring Boot framework and internally uses an SQL database to store the `tags`, `tagDomains` as well as the `annotations` added. SQL was used as it is capable of bot the storage of the annotation taxonomy as well as the annotations added by end users. Queries are implemented using the Spring Data HATEOAS (Hypermedia as the Engine of Application State) and helped us develop queries on the system data with minimal code requirements resulting in a standardized API. The Annotation Service has itself no actual user interface but allows other services of the OrganiCity platform to develop their own flexible interfaces for data annotation. For example, the most clear interface developed is available in the main OrganiCity web interface. Using this interface, users of OrganiCity can view the data collected the IoT devices and add their own annotations on top of them. This is implemented by a simple question-response schema that presented to each users based on the type of data presented. An example of this interface is available in Figure 5.9. In this interface, we can see that bellow the data of the IoT device a UI

(a) Fields

Name	Data Type	Required	Description
`id`	numeric	Y	The unique id of the TagDomain in the internal DB
`description`	string	Y	A user friendly description for the TagDomain
`urn`	string	Y	The unique id of the TagDomain available to users
`user`	string	Y	The id of the user that added the TagDomain

(b) Relationships

Name	Target	Description
`HAS`	`Tag`	Every TagDomain can contain one or more Tags

Table 5.3 TagDomain entity fields (a) and relationships (b).

(a) Fields

Name	Data Type	Required	Description
`annotationId`	numeric	Y	The unique id of the Annotation in the internal DB
`assetUrn`	string	Y	The unique id of the OC Asset
`datetime`	string	Y	The time of the Annotation
`numericValue`	numeric	N	A number concerning the Annotation
`textValue`	string	N	A text message concerning the Annotation
`user`	string	Y	The id of the user that submitted the Annotation

(b) Relationships

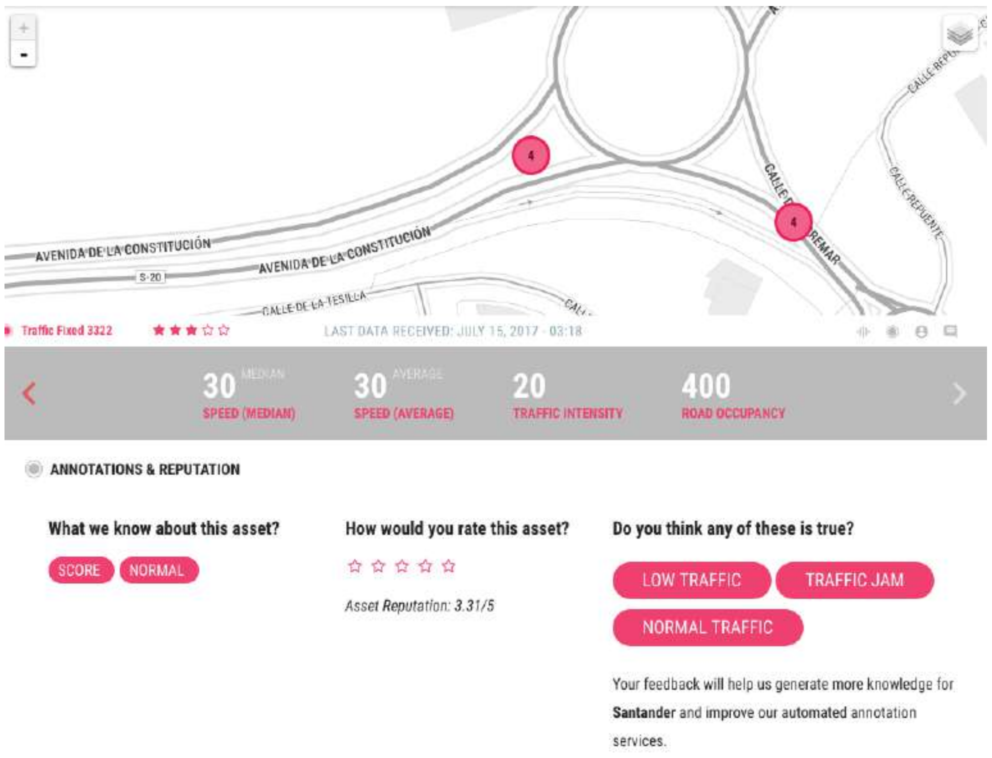
Name	Target	Description
`TAGGING`	`Tag`	Every Annotation characterizes an Asset with a Tag

Table 5.4 Annotation entity fields (a) and relationships (b).

slice is dedicated to the Annotations Service and the reputation that is based on the functionality of the Annotations Service. In it we can view the following:

- On the left hand side of the user interface, we observe knowledge that the platform has for the given asset that the time of viewing. This is either information about the IoT device added by users or information that is generated by the machine learning component to be described bellow.

- The middle part, contains information about the reputation of the IoT device inside OrganiCity. This information is generated as a calculation on top of all the annotations added by users.



Your feedback will help us generate more knowledge for **Santander** and improve our automated annotation services.

Figure 5.9 Annotation user interface in OrganiCity. It contains the geographical location of the IoT device along with the knowledge available in the platform, the reputation of the device in the platform and the interface through which users can submit their own knowledge on the device

- Finally, the right hand side part of the user interface contains a set of questions addressed to the users concerning the data that are presented above. Each question, concerns a specific **tagDomain** of the annotation service and each response is translated as a **tag**. When a user selects one of the answers, the respective annotation is stored increasing the knowledge of the system.

Machine Learning Component

The Machine Learning Component orchestrates the machine learning process, including managing the executed jobs, training the instances with provided or retrieved data, and the exchange of real-time city data. Our system is capable of performing both anomaly detection and classification jobs over the streaming data. In both cases, after annotation jobs are added to the system the initial training data need to be submitted. After the initial training data are submitted, the annotation job starts with each data point examined and the result posted to the Annotation Component. The system is designed to be agnostic of the actual machine learning implementation, as it capable of using multiple libraries to run the classification or the anomaly detection on the data. This gives us flexibility to experiment with different machine learning algorithms and the ability to provide extra functionality in the future.

Model The underlying data model of the Machine Learning Service and the relations between its entities are described in the following tables (Table 5.5 and 5.6). The schema comprises mainly the following entities:

- **Classification Configuration**: An entity that is stored for each classification job to be executed. Each job concerns a subset of the assets of the OrganiCity platform. The selection of the assets is performed using the Orion Context Broker's subscription API. For each subscription we need three parameters: a pattern on the Asset's id, a pattern on the Asset's type and the attribute to be classified. Each subscription on the Orion Context Broker is identified by a subscription id that is also stored to identify expired subscriptions. Additionally, each entry contains the **tagDomain** that the classifications results in and an indicator to disable the process temporarily. The entity is described in more details in Table 5.5.

- **Classification Training Datum**: Data supplied by the user in order to train the classification model. The entity is described in more details in Table 5.6.

Name	Data Type	Required	Description
`typePattern`	string	Y	A pattern based on the OrganiCity Asset Model for asset types to enable the subscription for data updates
`idPattern`	string	Y	A pattern based on the OrganiCity Asset Model for asset Ids to enable the subscription for data updates
`attribute`	string	Y	The attribute that needs to be classified based on the OrganiCity Asset Model used to enable the subscription for data updates
`subscriptionId`	string	Y	The id of generated subscription for data updates from OrganiCity
`tags`	string	Y	The **tagDomain** of the Annotation component used in this classification job
`enable`	boolean	Y	An indicator to disable the classification process on demand

Table 5.5 Classification Configuration entity fields.

Name	Data Type	Required	Description
`id`	numeric	Y	A unique identifier for this training datum
`classificationConfigId`	string	Y	A foreign key to the Classification Configuration this datum refers to
`tag`	string	Y	The **tag** of the **tagDomain** of the Annotation component
`value`	numeric	Y	The training datum to be used

Table 5.6 Classification Training Data entity fields.

Machine Learning frameworks As the underlying implementation of the machine learning component we have experimented in the context of this thesis with two different frameworks: Jubatus [38] and JavaML [2].

Jubatus is designed to run as a standalone service that is setup and executed as a service in a Unix or Windows based system and can be accessed using as an RPC server. The advantage of such an execution is that the same instance can be used by multiple services to execute classification on a specific model. Unfortunately, each instance of Jubatus needs to be configured using a specific dataset and assigned to a specific job. In our case, as we need to instantiate multiple different jobs for each different user of the service, this cases a large waste of resources, due to the need for an instance management layer to be able to dynamically spawn new Jubatus processes. JavaML on the other side, operates as a library inside another application. This helped us easily and dynamically spawn new processes that can be handled in a much resource efficient way.

Both frameworks are used in the same way. Once an instance for both is setup, it needs to be trained using the training data provided by the user. These data are supplied to the framework and the internal model for the classification is generated. Once this is setup, the system is ready to receive the new data from the IoT devices. For each update, the machine learning framework is queried and the resulting classification is returned. The respective **tag** is then found using the internal database of the machine learning component and the annotation is published back to the Annotation component. The instance of the machine learning framework is kept in memory to speedup consecutive updates to the values of an IoT device.

5.3.3 Real World Evaluation

To verify the performance of our system, we setup a classification job for analyzing atmospheric pollution in London, based on data for the particulate matter concentration (PM_{10}). As training data for our test case, we used data of the same area from the past 12 months (1000 nominal, erroneous and error data points). We then let the system operate for 20 days, analyzing more than 40000 sensor measurements (translated to an average of 6 measurements every 15 minutes). The distributions of the values for the training data and the sensor data are presented in Figure 5.10 and Figure 5.11. A big part of the sensor measurements received is -1, pointing out a malfunction in the sensor devices, while another part of the measurements is greater than 50 $\mu g/m^3$, the level the European Union considers dangerous when exceeded for more than 35 days in a year. Our system was able to detect all values that were either negative or greater than the 50 $\mu g/m^3$ limit and record the time and date these values were abnormal.

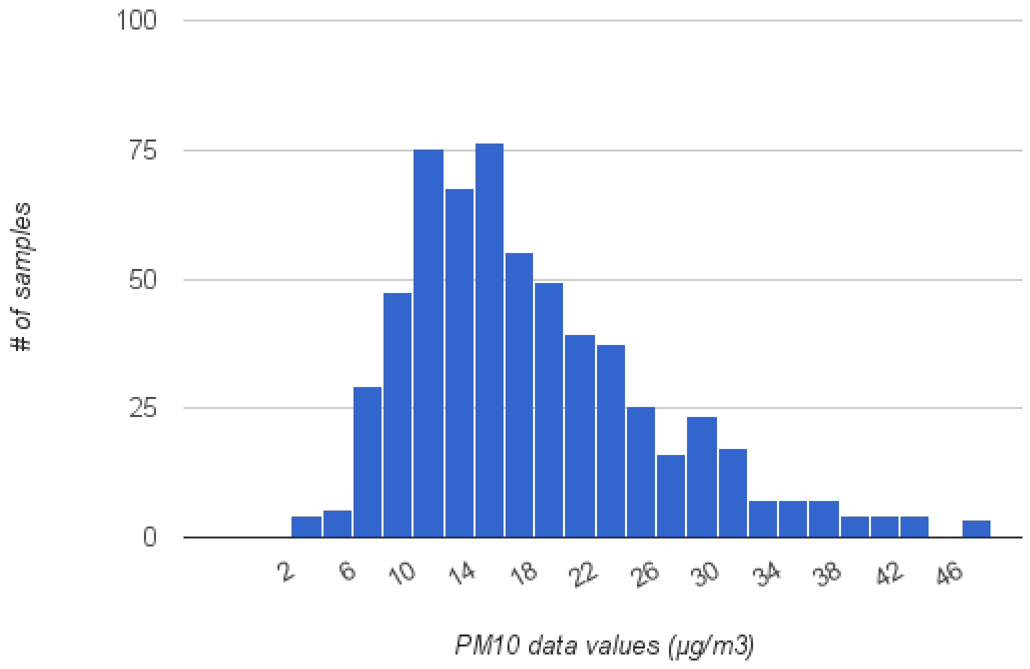


Figure 5.10 Distribution of training data for the PM10 experiment.

These 20 experimentation days helped us show that the data generated by smart city installations are not always trustworthy. A big number of the deployed sensor devices proved to be malfunctioning during our experiment (negative values), while a small number of measurements provided by the rest of the devices differed from the expected levels. The system performed without any problems for the whole duration of the experiment, on a virtual machine with limited resources (8*GB* of hard disk, 2*GB* RAM and 2 CPU cores) proving the system's scalability.

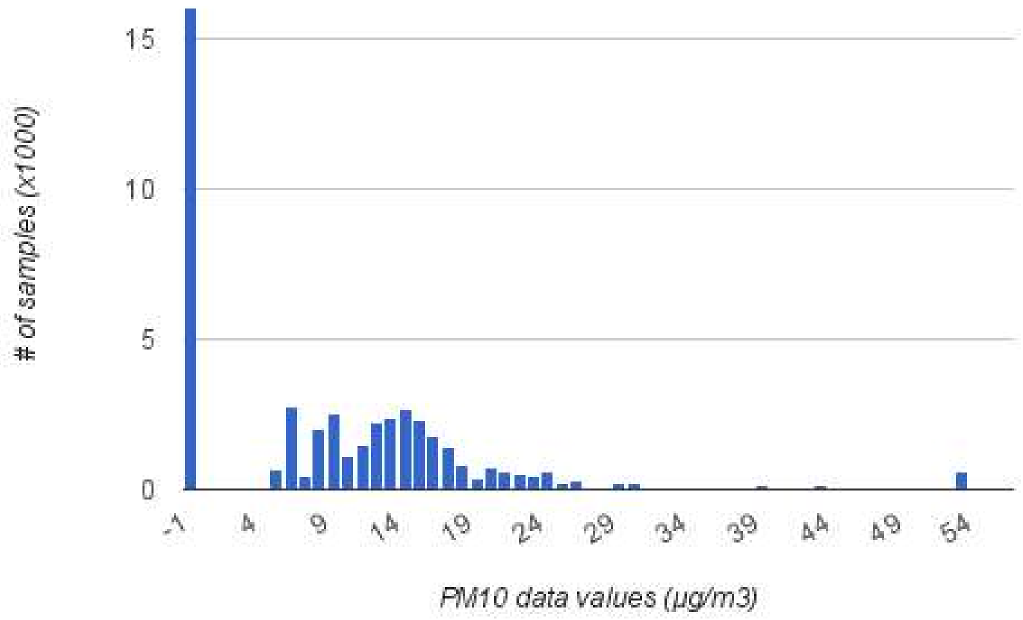


Figure 5.11 Histogram of actual sensor data received during the 20 experiment days. Negative values indicate malfunctioning sensors.

A second experiment was conducted using data from IoT devices that are placed in the streets of Santander, Spain. These devices are equipped with magnetic loop sensors that count the number of cars passing over in fixed periods of time. We use the data from those sensors to estimate the usage of the streets and a possible congestion, thus estimating the traffic levels inside the city. We defined a total of 3 categorizations for the traffic levels: `low`, `normal` and `congestion`. In this case we experimented with two different strategies. In the first strategy we added to the system a single value that represented each classification category as training data, while in the second run we defined multiple values. The distribution of the training data in both cases is presented in Table 5.7. The resulting classifications are in both cases the same. This is due to the fact that the training data in the second case are equally distributed in the three segments (0–15, 15–55, 55–100). As a result, we realized that in cases where the data in each classification category do not follow a non-normal distribution, using a much smaller dataset is equally effective as a larger dataset.

In cases where the classification domains of a physical phenomenon are more complex, like for example the classification of a received temperature in good,

Experiment	Classification Category	Training Data #	Data
1	`low`	1	0
	`normal`	1	30
	`congestion`	1	80
2	`low`	100	0-15
	`normal`	100	15-55
	`congestion`	100	55-100

Table 5.7 Classification Training Data for the street traffic experiment.

acceptable and extreme values our classification data can extend to more than one "buckets". The histogram of the data used as input for this use case is presented in Figure 5.12. For our next test case, we use the relative humidity data collected from sensors placed inside school classrooms of the GAIA project.

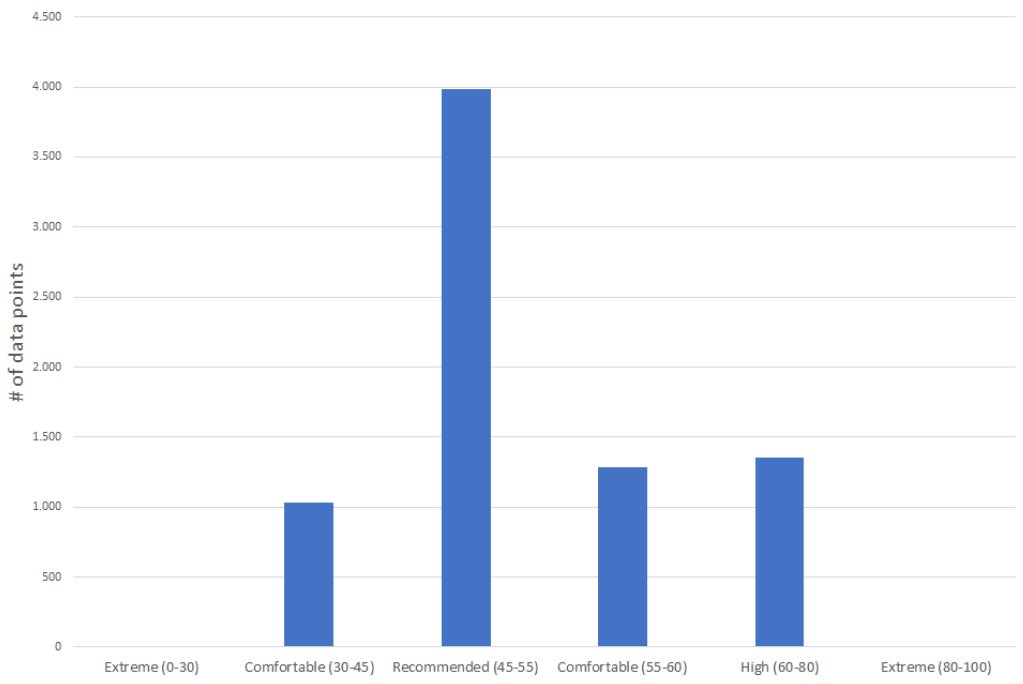


Figure 5.12 Distribution of data in the given categories for the relative humidity inside school classrooms.

Such, data are collected by the infrastructure presented in Chapter 4 and are fed directly to the JAMAiCA machine learning component for analysis. We base our analysis in established regulations by the European Union or other global organizations like ASHRAE[3], where we can read that controlling relative humidity is important for children's comfort and for the prevention of moisture accumulation, which can lead to mold growth. In general, relative humidity shall be between 30% and 50%[28]. From our work, we see that the data from the Greek public schools are well within the acceptable limits, with only a few measurements (17%) moving away from the acceptable parameters. It is also important to note that when we focus on the office hours (8–15) the conditions inside the classrooms improve significantly, with the measurements exceeding the acceptable values reduced to a 6%. In both cases, the recommended conditions are met at the 66% of the time with the rest being conditions that can be characterized as comfortable but not perfect. The resulting annotations are presented in Figure 5.13 for both office hours and the whole day.

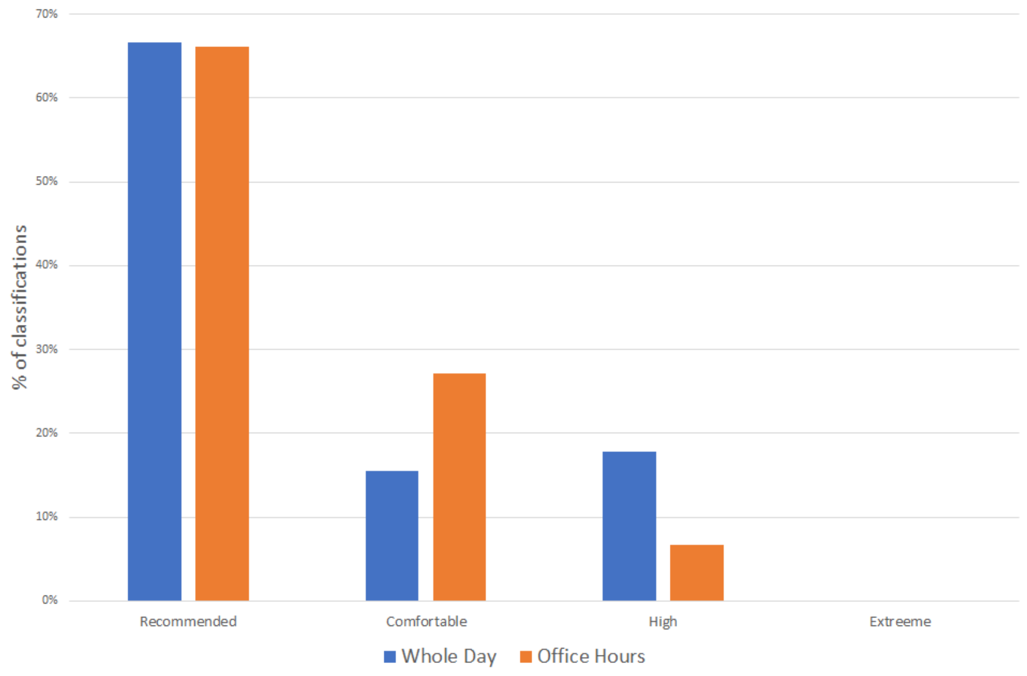


Figure 5.13 Distribution of classifications in the given categories for the relative humidity inside school classrooms.

[3] American Society of Heating, Refrigerating and Air-Conditioning Engineers https://www.ashrae.org

Chapter 6

Conclusions & Future Work

This chapter presents the main conclusions from our work in the context of this dissertation, the lessons we learned and possible future extensions.

6.1 Conclusions

In the context of this dissertation we dealt with the problems that arise from the introduction of more and more smart and connected devices in our everyday environments. This increasing use of such devices is what makes an improved data analysis methodology essential for making use of the data that are generated.

Our work was based on two real-world use cases concerning a fleet of educational school buildings and a federation of European cities as part of two EU funded research projects, providing us with a testing ground to apply and validate our solutions. As a result, we can now showcase applications that are capable of efficiently collecting and analyzing data from those two sources and providing them to end-users and application developers in near-real-time. We are also capable of generating additional knowledge from the raw data generated by IoT installations or from crowdsourcing. We also proved that our system architecture is capable of handling various data types and inputs with minimal changes based on its modular design while remaining salable and efficient under any level of traffic.

6.2 Lessons Learned

During our work and interaction with the IoT domain, we were also presented with some important lessons that should be taken into account in any future work on the same or similar fields.

IoT devices tend to be extremely unreliable especially when installed in the wild outside of our control. Some of the users tend to distrust the devices and cause problems, intentionally or not, to the installation, thus causing problems in maintaining the data streams from the installations. Also, hardware sensor failures or connectivity issues are a common case and will appear in any installation independently of their cost or manufacturer.

Finally, in the case of crowdsensing, it is important to note that the user interfaces developed are extremely important in order to keep interest on the application. In the case where the user interface is even a little more complex than what is required, users tend to lose focus and will to use any kind of application.

6.3 Future Work

An important extension to the work presented in this dissertation would be the pursue of the possibility of splitting the analysis of the collected data in more than one layers. In this case, some of the processing could happen on the premises of the user, thus reducing the need for data transfer over the Internet. This case will also help us reduce any possible data loses from networking and connectivity issues, which are quite common with commercial internet connections. This trend, called edge or fog computing is a highly interesting and promising field of research that can significantly increase the efficiency of the system. More extensions could be focused to adding more data types in the analysis modules of our system like video or audio as their nature could reveal more complex requirements that were not investigated here.

Bibliography

[1] 3GPP. *The Mobile Broadband Standard*. accessed on 2018-05-31. URL: http://www.3gpp.org/.

[2] Thomas Abeel, Yves Van de Peer, and Yvan Saeys. "Java-ml: A machine learning library". In: *Journal of Machine Learning Research* 10.Apr (2009), pp. 931–934.

[3] US Environmental Protection Agency. *Energy Star Building Upgrade Manual*. Tech. rep. EPA, 2008.

[4] Wi-Fi Alliance. *Wi-Fi*. accessed on 2018-05-31. URL: https://www.wi-fi.org/.

[5] LoRa Alliance. *LoRa*. accessed on 2018-05-31. URL: https://lora-alliance.org/.

[6] Zigbee Alliance. *Zigbee.org*. accessed on 2018-05-31. URL: http://www.zigbee.org/.

[7] Dimitrios Amaxilatis, Orestis Akrivopoulos, Georgios Mylonas, and Ioannis Chatzigiannakis. "An IoT-based solution for monitoring a fleet of educational buildings focusing on energy efficiency". In: *Sensors* 17.10 (2017), p. 2296.

[8] Dimitrios Amaxilatis, Dennis Boldt, Johnny Choque, Luis Diez, Etienne Gandrille, Sokratis Kartakis, Georgios Mylonas, and Lasse Steenbock Vestergaard. "Advancing Experimentation-as-a-Service Through Urban IoT Experiments". In: *IEEE Internet of Things Journal* (2018).

[9] Dimitrios Amaxilatis, Ioannis Chatzigiannakis, and Georgios Mylonas. "Design and Implementation of a Platform for Smart Connected School Buildings." In: *AmI (Workshops/Posters)*. 2015.

[10] Dimitrios Amaxilatis, Ioannis Chatzigiannakis, Georgios Mylonas, Lidia Pocero, Dimitrios Zarras, and Andreas Koskeris. "Green mindset: using IoT to promote energy efficiency and sustainability in Greek public schools". In: *Proceedings of the 19th Panhellenic Conference on Informatics*. ACM. 2015, pp. 297–302.

[11] Dimitrios Amaxilatis, Evangelos Lagoudianakis, Georgios Mylonas, and Evangelos Theodoridis. "Managing smartphone crowdsensing campaigns through the Organicity smart city platform". In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM. 2016, pp. 1460–1465.

[12] Dimitrios Amaxilatis, Georgios Mylonas, Luis Diez, Evangelos Theodoridis, Verónica Gutiérrez, and Luis Muñoz. "Managing Pervasive Sensing Campaigns via an Experimentation-as-a-Service Platform for Smart Cities". In: *Sensors* 18.7 (2018). ISSN: 1424-8220. DOI: 10.3390/s18072125.

[13] Dimitrios Amaxilatis, Georgios Mylonas, Evangelos Theodoridis, Guillem Camprodon, Veronica Gutiérrez, and Luis Diez. *Tools for crowd-sourcing of data annotations*. Tech. rep. OrganiCity, 2018. URL: https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5af406537&appId=PPGMS.

[14] *Amazon Web Services IoT*. URL: https://aws.amazon.com/iot.

[15] Omid Ardakanian, Arka Bhattacharya, and David Culler. "Non-intrusive techniques for establishing occupancy related energy savings in commercial buildings". In: *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. ACM. 2016, pp. 21–30.

[16] Kevin Ashton. *That "Internet of Things" Thing*. accessed on 2018-03-23. URL: http://www.rfidjournal.com/articles/view?4986.

[17] Equity Assessment. "Urbanization and health". In: *Bull World Health Organ* 88 (2010), pp. 245–246.

[18] Daniel W Barowy, Charlie Curtsinger, Emery D Berger, and Andrew McGregor. "Automan: A platform for integrating human-based and digital computation". In: *Communications of the ACM* 59.6 (2016), pp. 102–109.

[19] *Big Data*. accessed on 2018-03-23. URL: https://en.wikipedia.org/wiki/Big_data.

[20] Dennis Boldt, Dimitrios Amaxilatis, Georgios Mylonas, Etienne Gandrille, Arturo Medela, Veronica Gutiérrez, Luis Díez, Johnny. Choque, Luis Muñoz, Juan Echevarría, and Camprodon Guillem. *OrganiCity EaaS model and APIs - Final*. Tech. rep. OrganiCity, 2018. URL: https://www.ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5aff3c923&appId=PPGMS.

[21] Green Smart Campus. *Building-User Learning Interaction for Energy Efficiency*. URL: http://greensmartcampus.eu.

[22] *Carbon Detectives portal*. URL: http://www.carbondetectiveseurope.org/.

[23] Stuart K. Card, George G. Robertson, and Jock D. Mackinlay. "The Information Visualizer, an Information Workspace". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '91. New Orleans, Louisiana, USA: ACM, 1991, pp. 181–186. ISBN: 0-89791-383-3. DOI: 10.1145/108844.108874. URL: http://doi.acm.org/10.1145/108844.108874.

[24] Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. "Cascade: Crowdsourcing taxonomy creation". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2013, pp. 1999–2008.

[25] Kate Crosby and Anisa Baldwin Metzger. *Powering Down: A toolkit for Behavior-based Energy Conservation in K-12 Schools*. Tech. rep. U.S. Green Building Council (USGBC), 2012.

[26] Aikaterini Deligiannidou, Dimitrios Amaxilatis, Georgios Mylonas, and Evangelos Theodoridis. "Knowledge co-creation in the OrganiCity: Data annotation with JAMAiCA". In: *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*. IEEE. 2016, pp. 717–722.

[27] Audiovisual Education and Culture Executive Agency. *Key Data on Education in Europe 2012*. Tech. rep. ISBN 978-92-9201-242-7. EACEA P9 Eurydice, EUROSTAT, 2012. DOI: 10.2797/77414.

[28] Andrey Egorov, Marco Martuzzi, Michal Krzyzanowski, Otto Hänninen, Ulla Haverinen-Shaughnessy, Martin Täubel, Otmar Geiss, Stylianos Kephalopoulos, Josefa Barrero, Claudia Wendland, et al. *School environment: policies and current status*. Tech. rep. World Health Organization, 2015, p. 39.

[29] Thomas Erickson, Ming Li, Younghun Kim, Ajay Deshpande, Sambit Sahu, Tian Chao, Piyawadee Sukaviriya, and Milind Naphade. “The Dubuque Electricity Portal: Evaluation of a City-scale Residential Electricity Consumption Feedback System”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’13. Paris, France: ACM, 2013, pp. 1203–1212. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2466155. URL: http://doi.acm.org/10.1145/2470654.2466155.

[30] Buildings Performance Institute Europe. *Europe’s buildings under the microscope: A country-by-country review pf the energy performance of buildings*. Tech. rep. ISBN: 9789491143014. BPIE, 2011.

[31] Statistical Office of the European Union. *Energy Usage of Buildings*. Tech. rep. Data extracted from http://epp.eurostat.ec.europa.eu. Eurostat, 2011.

[32] *European Web Portal for energy efficiency in buildings*. accessed on 2018-01-06. URL: http://www.buildup.eu.

[33] Jon Froehlich, Leah Findlater, and James Landay. “The Design of Eco-feedback Technology”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’10. Atlanta, Georgia, USA: ACM, 2010, pp. 1999–2008. ISBN: 978-1-60558-929-9. DOI: 10.1145/1753326.1753629. URL: http://doi.acm.org/10.1145/1753326.1753629.

[34] Bluetooth Special Interest Group. *Bluetooth Low Energy*. accessed on 2018-05-31. URL: https://www.bluetooth.com/.

[35] Bin Guo, Zhu Wang, Zhiwen Yu, Yu Wang, Neil Y Yen, Runhe Huang, and Xingshe Zhou. “Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm”. In: *ACM Computing Surveys (CSUR)* 48.1 (2015), p. 7.

[36] Verónica Gutiérrez, Dimitrios Amaxilatis, Georgios Mylonas, and Luis Muñoz. “Empowering Citizens Toward the Co-Creation of Sustainable Cities”. In: *IEEE Internet of Things Journal* 5.2 (2018), pp. 668–676.

[37] Verónica Gutiérrez, Evangelos Theodoridis, Georgios Mylonas, Fengrui Shi, Usman Adeel, Luis Diez, Dimitrios Amaxilatis, Johnny Choque, Guillem Camprodom, Julie McCann, et al. “Co-creating the cities of the future”. In: *Sensors* 16.11 (2016), p. 1971.

[38] Shohei Hido, Seiya Tokui, and Satoshi Oda. “Jubatus: An open source platform for distributed online machine learning”. In: *NIPS 2013 Workshop on Big Learning, Lake Tahoe*. 2013.

[39] Florian Holzschuher and René Peinl. “Performance of Graph Query Languages: Comparison of Cypher, Gremlin and Native Access in Neo4J”. In: *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. EDBT ’13. Genoa, Italy: ACM, 2013, pp. 195–204. ISBN: 978-1-4503-1599-9. DOI: 10.1145/2457317.2457351. URL: http://doi.acm.org/10.1145/2457317.2457351.

[40] Chin-Lung Hsu and Judy Chuan-Chuan Lin. “An empirical examination of consumer adoption of Internet of Things services: Network externalities and concern for information privacy perspectives”. In: *Computers in Human Behavior* 62 (2016), pp. 516–527. ISSN: 0747-5632. DOI: https://doi.org/10.1016/j.chb.2016.04.023. URL: http://www.sciencedirect.com/science/article/pii/S0747563216302990.

[41] *Internet of things*. accessed on 2018-03-23. URL: https://en.wikipedia.org/wiki/Internet_of_things.

[42] Stevan Jokic, Aleksandra Rankov, Joao Fernandes, Michele Nati, Sébastien Ziegler, Theofanis Raptis, Constantinos M Angelopoulos, Sotiris Nikoletseas, Orestis Evangelatos, Jose Rolim, et al. “IoT Lab—Crowdsourced Experimental Platform Architecture”. In: *Proceedings of the 5th International Conference on Information Society and Technology (ICIST 2015), Kopaonik, Serbia*. 2015, pp. 8–11.

[43] Russel K Jones, Farshid Alizadeh-Shabdiz, Edward J Morgan, and Michael G Shean. *Techniques for computing location of a mobile device using calculated locations of wi-fi access points from a reference database*. US Patent App. 15/411,278. 2017.

[44] Aristotelis Kretsis, Panagiotis. Kokkinos, Emmanouel Varvarigos, Vassilis Nikolopoulos, E. Gkioxi, and I. Hatzakis. “GEN6 EU-project: Greek Pilot”. In: *Proceedings of the 18th Panhellenic Conference on Informatics*. PCI ’14. Athens, Greece: ACM, 2014, 25:1–25:5. ISBN: 978-1-4503-2897-5. DOI: 10.1145/2645791.2645837. URL: http://doi.acm.org/10.1145/2645791.2645837.

[45] Jorge Lanza, Luis Sanchez, David Gomez, Tarek Elsaleh, Ronald Steinke, and Flavio Cirillo. “A Proof-of-Concept for Semantically Interoperable Federation of IoT Experimentation Facilities”. In: *Sensors* 16.7 (2016), p. 1006.

[46] In Lee and Kyoochun Lee. “The Internet of Things (IoT): Applications, investments, and challenges for enterprises”. In: *Business Horizons* 58.4 (2015), pp. 431–440. ISSN: 0007-6813. DOI: https://doi.org/10.1016/j.bushor.2015.03.008. URL: http://www.sciencedirect.com/science/article/pii/S0007681315000373.

[47] Chi Harold Liu, Bo Yang, and Tiancheng Liu. “Efficient naming, addressing and profile services in Internet-of-Things sensory environments”. In: *Ad Hoc Networks* 18 (2014), pp. 85–101. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2013.02.008. URL: http://www.sciencedirect.com/science/article/pii/S1570870513000280.

[48] Alessandra Marini, Chiara Passoni, Paolo Riva, Paolo Negro, Elvira Romano, and Fabio Taucer. *Technology options for earthquake resistant, eco-efficient buildings in Europe: Research needs*. Tech. rep. ISBN: 978-92-79-35424-3. Institute for the Protection and Security of the Citizen, Joint Research Centre, European Commission, 2014. DOI: 10.2788/68902.

[49] *Microservices pattern*. accessed on 2018-03-23. URL: https://en.wikipedia.org/wiki/Microservices.

[50] *Microsoft Azure IoT*. URL: https://www.microsoft.com/en-us/internet-of-things/azure-iot-suite.

[51] Georgios Mylonas, Dimitrios Amaxilatis, Helena Leligou, Theodore Zahariadis, Emmanouil Zacharioudakis, Joerg Hofstaetter, Andreas Friedl, Federica Paganelli, Giovanni Cuffaro, and Jimm Lerch. "Addressing behavioral change towards energy efficiency in European educational buildings". In: *Global Internet of Things Summit (GIoTS), 2017*. IEEE. 2017, pp. 1–6.

[52] Georgios Mylonas, Ioannis Chatzigiannakis, Dimitrios Amaxilatis, Federica Paganelli, and Aris Anagnostopoulos. "Enabling Sustainability and Energy Awareness in Schools Based on IoT and Real-World Data". In: *Pervasive Computing* (2018). to appear.

[53] Georgios Mylonas, Evangelos Theodoridis, and Luis Muñoz. "Integrating Smartphones into the SmartSantander Infrastructure". In: *IEEE Internet Computing* 19.2 (2015), pp. 48–56. DOI: 10.1109/MIC.2015.25. URL: https://doi.org/10.1109/MIC.2015.25.

[54] L. Nicolae. *Council of Europe Plenary Session: Youth Education for Sustainable Development*. 2005.

[55] *Open Programmable City*. accessed on 2018-01-06. URL: http://www.bristolisopen.com/.

[56] *OrganiCity GitHub repository*. accessed on 2018-03-23. URL: http://organicityeu.github.io/.

[57] Jonathan Osborne and Justin Dillon. *Science Education in Europe: Critical Reflections*. 2008.

[58] Lidia Pocero, Dimitrios Amaxilatis, Georgios Mylonas, and Ioannis Chatzigiannakis. "Open source IoT meter devices for smart and energy-efficient school buildings". In: *HardwareX* 1 (2017), pp. 54–67. ISSN: 2468-0672. DOI: http://dx.doi.org/10.1016/j.ohx.2017.02.002. URL: http://www.sciencedirect.com/science/article/pii/S2468067216300293.

[59] Bastian Quilitz and Ulf Leser. "Querying Distributed RDF Data Sources with SPARQL". In: *The Semantic Web: Research and Applications*. Ed. by Sean Bechhofer, Manfred Hauswirth, Jörg Hoffmann, and Manolis Koubarakis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 524–538. ISBN: 978-3-540-68234-9.

[60] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. "Learning from crowds". In: *Journal of Machine Learning Research* 11.Apr (2010), pp. 1297–1322.

[61] IBM Research. *Experimentation as a Service.* accessed on 2018-01-06. URL: http://www.inetservicescloud.com/ibm-launches-experimentation-as-a-service-offering/.

[62] Luis Sanchez, Luis Muñoz, Jose Antonio Galache, Pablo Sotres, Juan R. Santana, Veronica Gutierrez, Rajiv Ramdhany, Alex Gluhak, Srdjan Krco, Evangelos Theodoridis, and Dennis Pfisterer. "SmartSantander: IoT experimentation over a smart city testbed". In: *Computer Networks* 61.Supplement C (2014). Special issue on Future Internet Testbeds – Part I, pp. 217–238. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.bjp.2013.12.020. URL: http://www.sciencedirect.com/science/article/pii/S1389128613004337.

[63] Juan R Santana, José A Galache, Toyokazu Akiyama, Levent Gurgen, Morito Matsuoka, Martino Maggio, and Shuuichirou Murata. "FESTIVAL: towards an intercontinental federation approach". In: *International Conference on Mobile Networks and Management.* Springer. 2015, pp. 269–280.

[64] *SAVE ENERGY.* FP7 project. URL: http://www.ict4saveenergy.eu/.

[65] *School of the Future.* FP7 project. 2016. URL: http://www.school-of-the-future.eu/.

[66] Fadi Shrouf and Giovanni Miragliotta. "Energy management based on Internet of Things: practices and framework for adoption in production management". In: *Journal of Cleaner Production* 100 (2015), pp. 235–246. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2015.03.055. URL: http://www.sciencedirect.com/science/article/pii/S0959652615002760.

[67] Sigfox. *Sigfox.* accessed on 2018-05-31. URL: https://www.sigfox.com/en.

[68] *SMARTSPACES Project.* URL: http://www.smartspaces.eu.

[69] *Sounds of New York City project.* accessed on 2018-01-06. URL: https://wp.nyu.edu/sonyc/.

[70] Richard J Trudeau. *Introduction to graph theory.* Courier Corporation, 2013.

[71] *VERYSchool project- Learning by doing.* URL: http://www.veryschool.eu/.

[72] Markus Weiss, Thorsten Staake, Friedemann Mattern, and Elgar Fleisch. "PowerPedia: Changing Energy Usage with the Help of a Community-based Smartphone Application". In: *Personal and Ubiquitous Computing* 16.6 (2012), pp. 655–664. ISSN: 1617-4909. DOI: 10.1007/s00779-011-0432-y.

[73] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. "The multidimensional wisdom of crowds". In: *Advances in neural information processing systems.* 2010, pp. 2424–2432.

[74] Maarten Weyn and Frederik Schrooyen. "A Wi-Fi assisted GPS positioning concept". In: *ECUMICT, Gent, Belgium* (2008).

[75] Tom White. *Hadoop: The Definitive Guide.* 1st. O'Reilly Media, Inc., 2009. ISBN: 978-1449311520.

[76] *xDsl, Digital Subscriber Line.* accessed on 2018-05-31. URL: https://en.wikipedia.org/wiki/Digital_subscriber_line.

[77] *Xively platform.* URL: http://xively.com.

[78] *ZEMedS project (Zero Energy MEDiterranean Schools).* URL: http://www.zemeds.eu.

Appendix A

Publications

Book Chapters

I. Chatzigiannakis, G. Mylonas, I. Mavrommati and D. Amaxilatis, Using IoT-based BigData Generated Inside School Buildings. Accepted for publication as book chapter to BigData-enabled Internet of Things: Challenges and Opportunities. Accepted, to bepublished by the Institution of Engineering and Technology (IET)

Journals

D. Amaxilatis, , D. Boldt, J. Choque, L. Diez, E. Gandrille, S. Kartakis, G. Mylonas and L. Vestergaard (2018), Advancing Experimentation-as-a-Service Through Urban IoT Experiments in IEEE Internet of Things Journal
https://doi.org/10.1109/JIOT.2018.2871766

G. Mylonas, I. Chatzigiannakis, D. Amaxilatis, F. Paganelli and A. Anagnostopoulos (2018), Enabling Sustainability and Energy Awareness in Schools Based on IoT and Real-World Data in IEEE Pervasive Computing Special Issue on Real-World IoT Deployments, accepted to appear October-November 2018

D. Amaxilatis, G. Mylonas, L. Diez, E. Theodoridis, V. Gutiérrez, L. Muñoz (2018), Managing Pervasive Sensing Campaigns via an Experimentation-as-a-Service Platform for Smart Cities in Sensors, 18(7), 2125
https://doi.org/10.3390/s18072125

V. Gutiérrez, D. Amaxilatis, G. Mylonas and L. Muñoz (2017), Empowering citizens towards the co-creation of sustainable cities in IEEE Internet of Things Journal, 5(2), 668-676 ,
https://doi.org/10.1109/JIOT.2017.2743783

D. Amaxilatis, O. Akrivopoulos, G. Mylonas and I. Chatzigiannakis (2017), An IoT-Based Solution for Monitoring a Fleet of Educational Buildings Focusing on Energy Efficiency, in Sensors 17(10), 2296,
https://doi.org/10.3390/s17102296

V. Gutiérrez, E. Theodoridis, G. Mylonas, F. Shi, U. Adeel, L. Díez, D. Amaxilatis, J. Choque, G. Camprodom, J. A. McCann and L. Muñoz (2016), Co-Creating the Cities of the Future in Sensors 16(11): 1971,
https://doi.org/10.3390/s16111971

In Refereed Conferences

G. Mylonas. D. Amaxilatis, H. Leligou, T. Zahariadis, E. Zacharioudakis, J. Hofstaetter, A. Friedl, F. Paganelli, G. Cuffaro and J. Lerch (2017), Addressing behavioral change towards energy efficiency in European educational buildings in Global Internet of Things Summit (GIoTS), Geneva,
https://doi.org/10.1109/GIOTS.2017.8016258

A. Deligiannidou, D. Amaxilatis, G. Mylonas and E. Theodoridis (2016), Knowledge co-creation in the OrganiCity: Data annotation with JAMAiCA in WF-IoT pages: 717-722,
https://doi.org/10.1109/WF-IoT.2016.7845492

D. Amaxilatis, E. Lagoudianakis, G. Mylonas and E. Theodoridis (2016), Managing smartphone crowdsensing campaigns through the organicity smart city platform in UbiComp Adjunct pages: 1460-1465,
https://doi.org/10.1145/2968219.2968588

D. Amaxilatis, I. Chatzigiannakis, G. Mylonas, L. Pocero, D. Zarras and A. Koskeris (2015), Green mindset: using IoT to promote energy efficiency and sustainability in Greek public schools, in Panhellenic Conference on Informatics pages: 297-302,
https://doi.org/10.1145/2801948.2801974

D. Amaxilatis, I. Chatzigiannakis and G. Mylonas (2015), Design and Implementation of a Platform for Smart Connected School Buildings, in AmI (Workshops/Posters)

Other Publications

Journals

L. Pocero, D. Amaxilatis, G. Mylonas and I. Chatzigiannakis (2017), Open source IoT meter devices for smart and energy-efficient school buildings in HardwareX, Volume 1, Pages 54-67
https://doi.org/10.1016/j.ohx.2017.02.002

In Refereed Conferences

G. Mylonas, C. Triantafyllis and D. Amaxilatis (2018), An Augmented Reality Prototype for supporting IoT-basedEducational Activities for Energy-efficient School Buildings. in 2018 European Conference on Ambient Intelligence workshop on Behavioral Change and Ambient Intelligence for Sustainability (BRAINS 2018).

C. Tziortzioti, D. Amaxilatis, I. Mavrommati and I. Chatzigiannakis (2018), IoT sensors in sea water environment: Experiences from a short summer trial. in 2018 European Conference on Ambient Intelligence workshop on Behavioral Change and Ambient Intelligence for Sustainability (BRAINS 2018).

D. Amaxilatis and K. Giannakopoulou (2018), Evaluating Retailers in a Smart-buying Environment using Smart City Infrastructures. in IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops).
https://doi.org/10.1109/PERCOMW.2018.8480304

K. Chantzis, D. Amaxilatis, I. Chatzigiannakis and J. D. P. Rolim (2014), SCLD-ATP: Symmetric Coherent Link Degree, Adaptive Transmission Power Control for Wireless Sensor Networks in SENSORNETS pages: 5-16
https://doi.org/10.5220/0004696200050016

K. Chantzis, D. Amaxilatis, I. Chatzigiannakis and J. D. P. Rolim (2014), Symmetric Coherent Link Degree, Adaptive Throughput-Transmission Power for Wireless Sensor Networks in DCOSS pages: 26-34
https://doi.org/10.1109/DCOSS.2014.13

I. Chatzigiannakis, D. Amaxilatis and S. Livathinos (2015), A collective awareness platform for energy efficient smart buildings, in Panhellenic Conference on Informatics pages: 295-296
https://doi.org/10.1145/2801948.2801993

Technical Reports

D. Amaxilatis, M. Logaras, O. Michail and P. G. Spirakis (2015), NETCS: A New Simulator of Population Protocols and Network Constructors in CoRRabs/1508.06731

Competitions

D. Amaxilatis and I. Chatzigiannakis (2016), Competition: An Adaptive Protocol Stack for High-Dependability based on the Population Protocols Paradigm in EWSN pages 291-292

Appendix B

Extended Summary in Greek

Αποδοτική διαχείριση δεδομένων στα πλαίσια του Διαδικτύου των Αντικειμένων

Εισαγωγή

Ο όρος Διαδίκτυο των Αντικειμένων (Internet of Things) εχει εμφανιστεί τα τελευταία χρόνια στη βιβλιογραφία και την καθημερινή ζωή εκατομυρίων ανθρώπων. Αποτελεί την φυσική συνέχεια της έρευνας που πραγματοποιήθηκε τα προηγούμενα χρόνια στον τομέα των ασύρματων δικτύων αισθητήρων (Wireless Sensor Networks) και αναφέρεται, σύμφωνα και με τον Kevin Ashton της Procter & Gamble [16] που πρωτο-χρησιμοποίησε τον όρο, στην ένθεση ηλεκτρονικών συσκευών με αισθητήρες, ενεργοποιητές καθώς και δυνατότητες επικοινωνίας σε φυσικά αντικείμενα με αποτέλεσμα την δημιουργία ενός δικτύου ανταλλαγής δεδομένων και πληροφοριών για την αλληλεπίδραση του φυσικιού και του ψηφιακού περιβάλλοντος. Μια τέτοια υποδομή επιτρέπει την ανάπτυξη ενός μεγάλου αριθμού υπηρεσιών σε πολλαπλά πεδία εφαρμογής όπως τα έξυπνα-δίκτυα, τα έξυπνα σπίτια, τις εφυείς μεταφορές και άλλα. Μερικά τέτοια παραδείγματα είναι μεταξύ άλλων οι συνδεδεμένοι θερμοστάτες των Nest και Ecobee, οι έξυπνοι λαμπτήρες φωτισμού της Phillips, οι συσκευές της Smarthings για μια μεγάλη γκάμα εφαρμογών αυτοματισμών σπιτιού αλλά και τα Fitbit και Withings για την παρακολούθηση της καθημερινής άσκησης και ζωής των χρηστών ή οι προσωπικοί βοηθοί Alexa της Amazon και Siri της Apple.

Παρά την τεράστια ανάπτυξη τέτοιων εφαρμογών, η χρησιμότητα των δεδομένων που προέρχονται από αυτά τα συστήματα πρέπει ακόμη να επικυρωθεί και να αποδειχθεί. Ο όγκος των δεδομένων που μπορούν να δημιουργηθούν από μία συσκευή αισθητήρων που είναι εγκατεστημένη σε ένα σπίτι ή μια φορητή συσκευή ενός ατόμου μπορεί να είναι εξωντοτική για την ίδια τη συσκευή και στις περισσότερες περιπτώσεις πρέπει να μεταφερθεί σε μια εφαρμογή επεξεργασίας δεδομένων μέσω της οποίας μπορεί να γίνει πιο αποτελεσματική χρήση των δεδομένων. Καθώς ο αριθός αυτός των συσκευών που χρησιμοποιούμε στην καθημερινή μας ζωή αυξάνεται σημαντικά μέρα με τη μέρα ο όγκος των δεδομένων αυτών αυξάνεται με ακόμη μεγαλύτερους ρυθμούς. Αυτό έχει σαν αποτέλεσμα, την ανάγκη για μια ολοκληρωμένη ανάλυση του θέματος της διαχείρισης των δεδομένων αυτών που παράγονται από έξυπνες συσκευές, ένα ανοιχτό θέμα για την έρευνα τόσο σε ακαδημαϊκά όσο και σε επιχειρηματικά πλαίσια.

Ένα άλλο χαρακτηριστικό αυτών των δικτύων συσκευών, είναι ο μεγάλος κατακερματισμός τους, τόσο στο πεδίο των κατασκευαστών όσο και στις τεχνολογίες συλλογής, μεταφοράς, επεξεργασίας και παρουσίασης των δεδομένων που συλλέγονται. Κύριος λόγος για αυτό το είναι τόσο το νεαρό της ηλικίας των τεχνολογιών όσο και η έλλειψη ισχυρών πρωτοκόλλων και διεθνών προτύπων που να ακολουθούν οι κατασκευαστές. Ο κατακερματισμός αυτός οδηγεί στην αναπόφευκτη ανάγκη για την δημιουργία δομών και μεθοδολογιών διασύνδεσης όλων αυτών των τεχνολογιών σε ένα κεντρικό σύστημα το οποίο μπορεί να πραγματοποιήσει όλες τις απαραίτητες επικοινωνίες και να προσφέρει τις διασυνδέσεις και διεπαφές μεταξύ αποκομένων τομέων του Διαδικτύου των Αντικειμένων.

Στόχος αυτής της Διδακτορικής Διατριβής (ΔΔ), είναι η μελέτη του πλαισίου λειτουργίας του Διαδικτύου των Αντικειμένων και όλων των τεχνολογιών που αυτό περιλαμβάνει για τον σχεδιασμό εφαρμογών που γεφυρώνουν πιθανά προβλήματα διαλειτουργικότητας και λειτουργούν αποτελεσματικά σε αυτό το νέο περιβάλλον. Πιο συγκεκριμένα επικντρονώμαστε στα εξής:

- Την αποδοτική ψηφιακή αναπαράσταση της δομής του Διαδικτύου των Αντικειμένων με βάση όλες τις ιδιαιτερότητες που αυτή διαθέτει.
- Την αποδοτική επεξεργασία των δεδομένων που προκύπτουν από εγκαταστάσεις Διαδικτύου των Αντικειμένων ανεξάρτητα με το μέγεθος της εγκατάστασης και τον τύπο των δεδομένων που αυτή παράγει.
- Την εξαγωγή πληροφοριών χρήσιμων για τους χρήστες των συστημάτων Διαδικτύου των Αντικειμένων με βάση τόσο τα πρωταρχικά δεδομένα που παράγονται από αυτές αλλά και τις διασυνδέσεις μεταξύ τους.

Για την επίτευξη αυτών των στόχων υλοποιήθηκαν στα πλαίσια αυτής της ΔΔ τα εξής:

- Ένα σχήμα αποθήκευσης και αναπαράστασης των πληροφοριών και μετα-δεδομένων εγκαταστάσεων Διαδικτύου των Αντικειμένων με χρήση γραφημάτων. Σε αντίθεση με τα τις παραδοσιακές σχεσιακές βάσεις δεδομένων που χρησιμοποιούνται στην πλειονότητα των διαδικτυακών εφαρμογών, τα γραφήματα μας δίνουν την δυνατότηνα να δημιουργήσουμε ένα ευέλικτο και επεκτάσιμο σχήμα που προσφέρει πιο εκφραστικά και αποτελεσματικά ερωτήματα. Η χρήση αυτών των ερωτημάτων αποφέρει σημαντικά μικρότερους χρόνους απάντησης και απαιτήσεις αποθήκευσης δεδομένω.
- Ένα σύστημα ανάλυσης δεδομένων προσαρμοσμένο στην συνεχή ροή πληροφοριών από εγκαταστάσεις Διαδικτύου Αντικειμένων. Το σύστημα αυτό μπορεί να επεξεργάζεται ροές δεδομένων με πολύ μεγάλους ρυθμούς σε εξαιρετικά χαμηλούς χρόνους απόκρισης.

Επίσης, διαθέτει μεγάλη ευελιξία στην ανάλυση των δεδομένω καθώς είναι παραμετροποιήσιμο για να μπορεί να δεχθεί κάθε πιθανό τύπο δεδομένων αλλά και σε κάθε βαθμό χρονικής ή τύπο στατιστικής ανάλυσης.

- Ενα σύστημα συλλογής δεδομένων από κινητές συσκευές εθελοντών για την καλύτερη κατανόηση των μεθόδων και των προβλημάτων που προκύπτουν από την αποκεντρομένη και άναρχη δομή ενός τέτοιου δικτύου.

- Τέλος, μια δομή ανάλυσης δεδομένων με χρήση τεχνολογιών μηχανικής μάθησης σε πραγματικό χρόνο για την εξαγωγή περεταίρω πληροφοριών από τα ακατέργαστα δεδομένα των εγκαταστάσεων.

Για την καλύτερη δυνατή εκτίμηση των δυνατοτήτων και των αποτελεσμάτων της χρήσης αυτών των τεχνολογιών τις εφαρμόσαμε στην πράξη στα πλαίσια 2 ερευνητικών προγραμμάτων της Ευρωπαϊκής Ένωσης, το GAIA[1] και το Organicity[2]. Το έργο GAIA αποσκοπεί στην προώθηση θετικών προτύπων συμπεριφοράς εντός κοινοτήτων όσον αφορά την κατανάλωση / συνειδητοποίηση της κατανάλωσης ενέργειας με τη χρήση των μετρήσεων της κατανάλωσης ενέργειας σε πραγματικό χρόνο, ενισχυμένων από εγκαταστάσεις Διαδικτύου Αντικειμένων, σε επιλεγμένες σχολικές κοινώτητες της Ιταλίας, της Ελλάδας και της Σουηδίας. Το Organicity είναι μια υπηρεσία πειραματισμού που διερευνά πώς οι πολίτες, οι επιχειρήσεις και οι αρχές μπορούν να συνεργαστούν για τη δημιουργία ψηφιακών λύσεων στις αστικές προκλήσεις χρησιμοποιώντας ένα σύνολο εργαλείων Διαδικτύου Αντικειμένων για να δοκιμάσουν και να αναπτύξουν τις δικές τους ιδέες σε επιτυχημένες εφαρμογές έξυπνων πόλεων.

Βασικές έννοιες και σχετικές ερευνητικές εργασίες

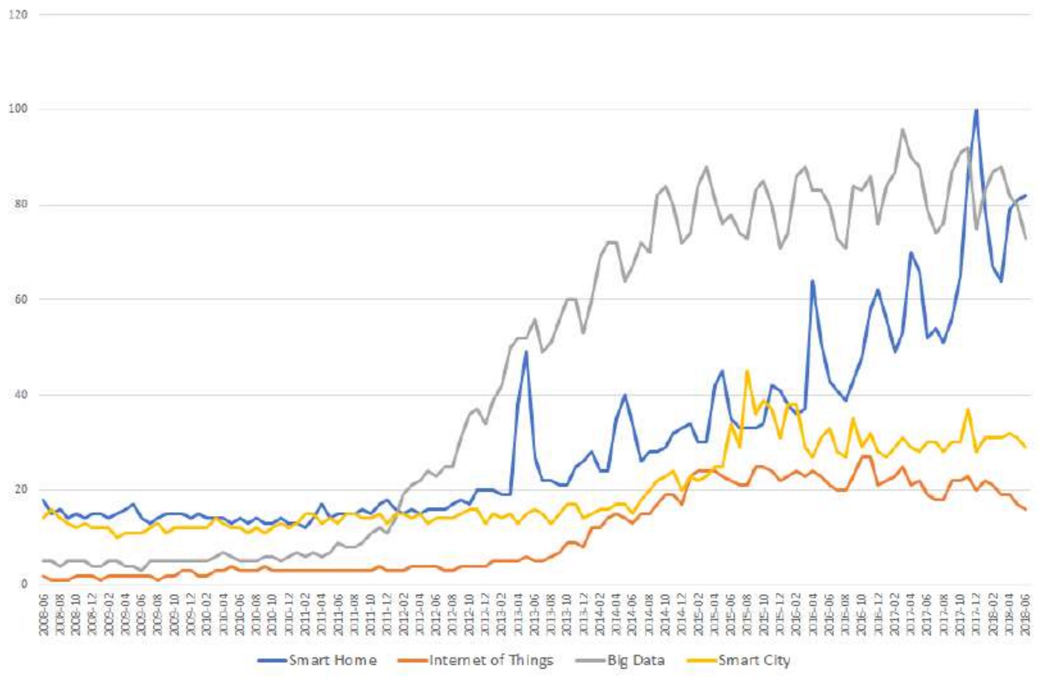
Βασικοί όροι με τους οποίους ασχολούμαστε στα πλαίσια αυτής της ΔΔ είναι μεταξή άλλων τα εξής:

- Έξυπνο Αντικείμενο: μπορεί να αναφέρεται σε ένα άτομο, ζώο ή φυσικό αντικείμεντο στο οποίο έχει εμφυτευθεί ή τοποθετηθεί μια μικρού μεγέθους υπολογισική συσκευή η οποία μπορει να συλλέξει τα απαιτούμενα δεδομένα από το άμμεσο περιβάλλον του αλλά και να τα επεξεργαστεί ή να τα αποστείλει σε γειτονικές συσκευές και στο διαδύκτιο.

- Ροές δεδομένων: Καθώς τα Αντικείμενα που αναφέραμε προηγουμένως συλλέγουν συνεχώς δεδομένα δημιουργούν μια συνεχή ροή πληροφορίας για τα φυσικά μεγέθη που παρακολουθούν. Οι ροές αυτές μπορούν να έχουν μη σταθερούς ρυθμούς αποστολής με αποτέλεσμα να είναι αδύνατον εκ των προτέρων να προβλεφθεί ο ακριβής τους όγκος και η συμπεριφορά τους.

[1] https://gaia-project.eu
[2]https://organicity.eu

- Επεξεργασία Ροής και Επεξεργασία Παρτίδας: Η επεξεργασία αυτών των δεδομένων που περιγράψαμε πριν μπορεί να γίνει με 2 τροπους. Είτε σειριακά καθώς τα δεδομένα φθάνουν στο σύστημα (Επεξεργασία Ροής) αλλά και συνολικά, ανά τακτά χρονικά διαστήματα με βάση το σύνολο των δεδομένων που έχουν συγκετρωθεί μέχρι τότε (Επεξεργασία Παρτίδας). Η δευτερη περίπτωση έχει μελετηθεί αρκετά τα τελευταία χρόνια με την υλοποίηση συστημάτων μεγάλων-δεδομένων και κύριο εκφραστή το Hadoop. Σε αυτή την περίπτωση χρησιμοποιείται μια αλληλουχία κατα ζεύγη επεξεργασίας για την σταδιακή μείωση των δεδομένων σε μια ή ενα μικρό σύνολο τιμών. Το μειονέκτημα μιας τέτοιας τεχνικής είναι κυρίως η καθυστέρηση της επεξεργασίας καθώς απαιτείται η συγκέντρωση όλων των τιμών πριν την εκτέλεσή της. Αντίθετα στην Επεξεργασία Ροής τα στατιστικά εξάγωνται σε πραγματικό χρόνο και ανανεώνωνται με την λήψη νέων δεδομένων.
- Τεχνολογίες Έξυπνων Σπιτιών: Ένας από τους πιο συζητημένους και δημοφιλείς όρους σε σχέση με τις αναζητήσεις στο διαδίκτυο τα τελευταία χρόνια σύμφωνα με τις τάσεις της Google (Σχήμα Β΄.1). Αναφέρεται στην εισαγωγή πολλαπλών συσκευών σε οικιακό ή γραφείο που επεκτείνει τα όρια των παραδοσιακών λύσεων διαχείρισης κτιρίων και προσφέρει πιο εξατομικευμένες πληροφορίες σχετικά με τη λειτουργία και τις παραμέτρους του κτιρίου, καθώς και των κατοίκων του.



Σχήμα Β΄.1 Στοιχεία δημοτικότητας όρων από την εφαρμογή Google trends για τους όρους Smart Homes, Smart Cities, Big Data Internet of Things.

- Τεχνολογίες Έξυπνων Πόλεων: καθώς σύμφωνα με πρόσφατες μελέτες το 2050 το 70% του παγκόσμιου πληθυσμού θα ζει σε αστικές περιοχές, ενώ περισσότερο από το ήμισυ του παγκόσμιου πληθυσμού ζει ήδη στις πόλεις οι διάφοροι ενδιαφερόμενοι φορείς (πολεοδόμοι, πολιτικοί, ερευνητές κ.λπ.) εφαρμόζουν πολιτικές που στοχεύουν στη βελτίωση της ποιότητας ζωής σε αστικά περιβάλλοντα με χρήση τεχνολογιών Διαδικτύου Αντικειμένων.

- Εξαγωγή Γνώσης: Μια κεντρική ερώτηση σχετικά με την συλλογή δεδομένων από έξυπνα αντικείμενα και ροές δεδομένω είναι το κατά πόσο μπορεί να υπάρξει εμπιστοσύνη στα παραγόμενα δεδομένα. Και όταν το δεδομένα θεωρούνται αξιόπιστα, πώς μπορούμε να εξάγουμε γνώση απο αυτά, ως κάτι χρήσιμο πέρα από έναν τεχνολογικό επίτευγμα. Επιπλέον, πώς παρέχουμε στους ιδιοκτήτες των δεδομένων αυτό κάποια διευκόληνση στην καθημερινότητα και τη ζωή τους. Στην ουσία, τέτοιες ερωτήσεις φθάνουν στο σημείο να κατανοήσουμε πώς πραγματικά κάνουμε την ανάλυση των δεδομένων πιο έξυπνη για να επιτύχουμε αυτό που χρειαζόμαστε. Παραδείγματος χάρη, στην περίπτωση κυκλοφοριακής συμφόρησης στο κέντρο της πόλης παράγετονται τιμές δεδομένων για την ταχύτητα των αυτοκινήτων, του θορύβου και της συγκέντρωσης αερίων απο τις εκπομπές των οχημάτων. Σε αυτά τα δεδομένα λείπουν χρήσιμοι συσχετισμοί στις ροές τους και δεν ειναι εύκολα εφικτός ο συσχετισμός τους. Η προσθήκη επισημάνσεων και συσχετισμών σε τέτοιου είδους έξυπνα δεδομένα μέσω μηχανισμών μηχανικής μάθησης ή με τη χρήση εθελοντών μπορεί να βοηθήσει να αποκαλύψουμε μια τεράστια κρυφή δυναμική στα δεδομένα που συλλέγουμε.

Αναπαράσταση του Διαδικτύου Αντικειμένων

Προκειμένου να αλληλεπιδράσουμε με μια εγκατάσταση Διαδικτύου Αντικειμένων, απαιτείται μια κατάλληλη αναπαράσταση γι άυτό στον ψηφιακό κόσμο. Κάθε εγκατάσταση περιλαμβάνει πολλά στοιχεία που παράγουν, καταναλώνουν ή μεταφέρουν δεδομένα. Κάθε στοιχείο σε αυτό το οικοσύστημα έχει πολλαπλές σχέσεις με άλλους, δημιουργώντας ένα περίπλοκο σχήμα που μπορεί να είναι δύσκολο να κατανοηθεί ή να απεικονιστεί καθώς η κλίμακα της εγκατάστασης αυξάνεται. Για να ξεπεραστεί αυτό το πρόβλημα και να απλοποιηθεί η αλληλεπίδρασή μας με αυτή την αντιπροσώπευση του ψηφιακού, πρέπει να εφαρμοστούν μορφές πιο κατάλληλες από τις παραδοσιακές σχεσιακές βάσεις δεδομένων. Ταυτόχρονα, αυτά τα μορφότυπα πρέπει να είναι εύκολα κατανοητά και να χρησιμοποιούνται από τους χρήστες με περιορισμένη αντίληψη της τεχνολογίας, όπως καλλιτέχνες ή ακτιβιστές ή απλούς πολίτες.

Για την ανάπτυξη αυτής της αναπαράστασης χρησιμοποιούμε τις πληροφορίες που έχουμε από το ερευνητικό έργο GAIA και εστιάζουμε στη δομή μιας εγκατάστασης που εκτείνεται σε ένα ολόκληρο σχολικό το κτίριο για την παρακολούθηση της κατανάλωσης ενέργειας και των περιβαλλοντικών παραμέτρων. Στόχος της πλατφόρμας είναι να διευκολύνει άμεσα τους χρήστες να συγκρίνουν τα συγκεντρωμένα δεδομένα του σχολείου τους με άλλα παρόμοια κτίρια που συμμετέχουν στο έργο, λαμβάνοντας προσεκτικά υπόψη περιβαλλοντικές παραμέτρους όπως η εποχή του χρόνου, η τοποθεσία ή ο καιρός. Η πλατφόρμα πρέπει επίσης να υποστηρίζει πολλές ομάδες τελικών χρηστών που υπάρχουν στον τομέα της εκπαίδευσης: μαθητές, εκπαιδευτικούς, διαχειριστές κτιρίων και άλλα διοικητικά στελέχη. Σε ένα τέτοιο κτίριο, μια τυπική εγκατάσταση βασισμένη στις γνώσεις του έργου αποτελείται από:

- μια συσκευή μετρητή ισχύος εγκατεστημένη στον κύριο ηλεκτρολογικό πίνακα του κτιρίου,
- 5-10 συσκευές μέτρησης περιβάλλοντος εγκατεστημένες σε ένα υποσύνολο των τάξεων και των κοινόχρηστων χώρων του κτιρίου,
- μια συσκευή μετεωρολογικού σταθμού εγκατεστημένη στην οροφή του κτιρίου

Με βάση αυτή την ανάλυση, μπορούμε να συνοψίσουμε τα εμπλεκόμενα μέλη στις ακόλουθες κατηγορίες:

- **Χρήστες**: Οι χρήστες είναι άνθρωποι που θα αλληλεπιδρούν ή θα ζουν μέσα στην εγκατάσταση.
- **Συσκευές ανίχνευσης**: Οι αισθητήρες είναι οι συσκευές που έχουν εγκατασταθεί και έχουν τον ρόλο των παραγωγών δεδομένων. Οι πληροφορίες τους μπορούν να μεταφερθούν, να καταναλωθούν ή να συλλεχθούν για τη χρήση τους
- **Συσκευές ενεργοποιητών**: Οι συσκευές ενεργοποιητή είναι οι συσκευές που μπορούν να ελεγχθούν για να διεγείρουν την αλλαγή στον φυσικό κόσμο. Περιλαμβάνουν στοιχεία όπως διακόπτες φωτός ή την καθορισμένη τιμή του θερμοστάτη σε ένα σύστημα κλιματισμού
- **Συσκευές πύλης**: Οι συσκευές πύλης είναι συσκευές που έχουν απλώς το ρόλο της μεταφοράς πληροφοριών μεταξύ της εγκατάστασης και του διαδικτύου. Ο ρόλος τους είναι ζωτικής σημασίας σε εγκαταστάσεις όπου οι άλλες εγκατεστημένες συσκευές δεν είναι συνδεδεμένες στο διαδίκτυο και δεν μπορούν να επικοινωνούν απευθείας με το υπόλοιπο σύστημα.
- **Παρατηρηθέντα φαινόμενα**: αναφέρονται στα φυσικά φαινόμενα που μπορούν να παρατηρηθούν από συσκευές ψηφιακής ανίχνευσης ή ενεργοποίησης.
- **Μονάδες μέτρησης**: αναφέρονται στη σύμβαση που χρησιμοποιείται για την ποσοτικοποίηση ενός παρατηρούμενου φαινομένου.
- **Τοποθεσίες**: αναφέρονται στις φυσικές και λογικές ομαδοποιήσεις που μπορούν να χρησιμοποιηθούν εφαρμόστηκε σε όλα τα στοιχεία που παρουσιάστηκαν παραπάνω. Μπορούν να περιγράψουν ένα κτίριο, μια σχολική μονάδα ή ένα εργαστήριο, αλλά και

μια πιο αφηρημένη ομαδοποίηση ενός συνόλου χρηστών μέσα σε πολλαπλές σχολικές κοινότητες.

Με παρόμοιο τρόπο υπάρχουν σχέσεις μεταξύ των στοιχείων που παρουσιάζονται παραπάνω που ορίζονται ως εξής:

- Η σχέση **ιδιοκτησίας** καθορίζει τη σχέση με την οποία οι χρήστες έχουν τον πλήρη έλεγχο και τις αποστολές σε ένα σύνολο αισθητήρων, ενεργοποιητών ή συσκευών πύλης και φυσικών τοποθεσιών.
- Τα δικαιώματα ενός χρήστη να έχει **πρόσβαση** στις πληροφορίες που παράγονται από ένα σύνολο συσκευών στίγματος ή τη φυσική θέση ή να ελέγχει μια διάταξη ενεργοποιητή.
- **Θέση** των συσκευών αισθητήρων και ενεργοποίητών ως μέρος μιας φυσικής ή λογικής θέσης. Για παράδειγμα, οι συσκευές συλλογής περιβαλλοντικών συνθηκών αποτελούν μέρος μιας φυσικής τάξης που ανήκει σε ένα κτίριο
- **Χαρακτηριστικά ανίχνευσης**: Τα χαρακτηριστικά ανίχνευσης αναφέρονται στις συσκευές ανίχνευσης και ενεργοποίησης και χρησιμοποιούνται για τον ορισμό του τύπου πληροφορίας που παράγει η κάθε συσκευή ή του τύπου της συσκευής που ελέγχει. Μπορεί να υπάρχουν πληροφορίες σχετικά με το φυσικό φαινόμενο ή τη μονάδα μέτρησης που μετράται ή ακόμα και αν τα δεδομένα που παράγονται είναι πρώτες τιμές ή χρειάζονται μια μεταγενέστερη επεξεργασία για να είναι έγκυρη.

Προκειμένου να παρέχουμε μια ψηφιακή αναπαράσταση αυτών των στοιχείων, επιλέγουμε να χρησιμοποιήσουμε μια βάση δεδομένων γραφημάτων. Μια τέτοια βάση δεδομένων είναι πιο κατάλληλη για την περίπτωση μας, καθώς μπορεί εύκολα να αντιστοιχίσουμε τα στοιχεία και τις σχέσεις τους με τις κορυφές και τις ακμές ενός γραφήματος. Παρέχει επίσης ως έναν μηχανισμό προβολής και αποθήκευσης για κάθε στοιχείο ή σχέση χρησιμοποιώντας ζεύγη κλειδιού-τιμής για τα χαρακτηριστικά τους και απλοποιεί τις αναζητήσεις μέσα στο γράφημα με τη μορφή ερωτημάτων γραφημάτων. Η υλοποίηση αυτής της αναπαράστασης γίνεται με χρήση της πλατφόρμας Neo4j μιας λύσης επιχειρηματικής ποιότητας για τη δημιουργία εφαρμογών υψηλών απαιτήσεων. Το τελικό σχήμα δεδομένων που χρησιμοποιούμε έχει περιέχει τις εξής οντότητες ως κόμβους του γραφήματος:

- **Χρήστης**: για την αποθήκευση πληροφοριών σχετικών με το χρήστη
- **Τοποθεσία**: για αποθήκευση σχολείων και σχολικών χώρων
- **Πηγή**: για την αποθήκευση σημείων ανίχνευσης
- **Πύλη**: για την αποθήκευση κόμβων πύλης που επικοινωνούν με συσκευές ανίχνευσης
- **Ιδιότητα**: για αποθήκευση των δυνατοτήτων ανίχνευσης και των μεταδεδομένων

Αντίστοιχα περιέχει τις εξής συσχετήσεις τως ακμές του γραφήματος:

- **ResourceProperty**: για τη σύνδεση μιας οντότητας πόρων με τις ιδιότητες που εμφανίζουν τις αισθητήριες παραμέτρους της
- **GatewayProperty**: για τη σύνδεση μιας οντότητας πύλης με τις ιδιότητες που εμφανίζουν τις δυνατότητες συνδεσιμότητάς της
- **IsPartOf**: για τη σύνδεση μιας οντότητας πόρων με τη θέση στην οποία βρίσκεται φυσικά
- **ShareWith**: για την παροχή πρόσβασης στο Σιτε στους χρήστες του συστήματος
- **SubSite** για εύκολη περιγραφή της δομής μέσα στα σχολικά κτίρια

Μέχρι στιγμής έχουν συμμετάσχει στο έργο GAIA συμμετέχουν 18 σχολικά κτίρια σε 3 χώρες (Ελλάδα, Ιταλία, Σουηδία) και καλύπτουν μια σειρά από τοπικές κλιματολογικές συνθήκες και επίπεδα εκπαίδευσης (πρωτοβάθμια, δευτεροβάθμια, γυμνάσια και πανεπιστήμια). Οι μετρητές κατανάλωσης ηλεκτρικής ενέργειας εγκαθίστανται σε όλα αυτά τα κτίρια, μαζί με αισθητήρες που παρακολουθούν εσωτερικές και εξωτερικές συνθήκες λειτουργίας όπως περιγράφεται παραπάνω. Η συντριπτική πλειονότητα των επιτηρούμενων δωματίων χρησιμοποιείται για εκπαιδευτικούς σκοπούς και τα υπόλοιπα για άλλες δραστηριότητες όπως η αίθουσα διδασκόντων, προσωπικού κ.λπ. Το έτος κατασκευής αυτών των κτιρίων κυμαίνεται από το 1950 έως το 2000. Για να αντιπροσωπεύουν όλες τις πληροφορίες του έργου η βάση δεδομένων γραφημάτων περιέχει συνολικά 7332 κόμβους και 47864 ακμές συνολικά για την ανάπτυξη και χρησιμοποιεί συνολικά 113MB στο χώρο του δίσκου. Από αυτές τις κορυφές, τα 4749 σημεία συγκέντρωσης είναι 1012 τοποθεσίες, 681 ιδιότητες και 300 χρήστες του συστήματος.

Συλλογή και Ανάλυση Δεδομένων

Το επόμενο ορόσημο μετά την αναπαράσταση των δεδομένω είναι η ανάπτυξη ενός συστήματος που να μπορεί να επεργάζεται όλα τα δεδομένα που παράγει μια τέτοια εγκατάσταση Διαδικύου Αντικειμένων. Το σύστημα αυτό πρέπει να καλύπτει ένα σύνολο βασικών απαιτήσεων για να είναι επιτυχημένο, όπως να είναι επεκτάσιμο, ευέλικτο, ανοιχτό και ασφαλές ως προς την ιδιωτικότητα των δεδομένω.

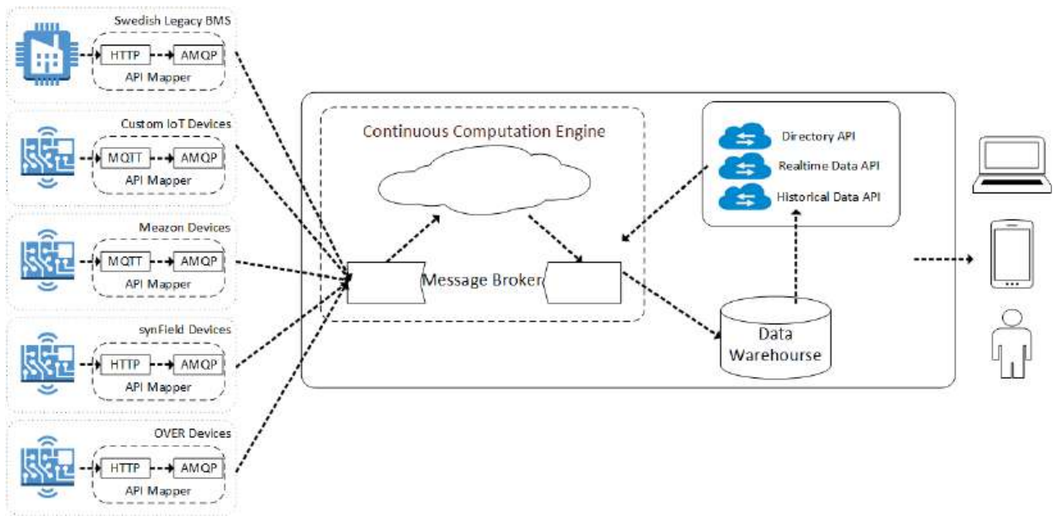
Για να επιτύχουμε μια τέτοια λύση, βασίζουμε τη δουλειά μας στη χρήση πλαισίων ανοιχτού κώδικα, καθιερωμένων εφαρμογών, που χρησιμοποιούνται από πολλούς προγραμματιστές λογισμικού σε όλο τον κόσμο. Αυτές οι τεχνολογίες είναι επίσης εύκολο να επεκταθούν και να υποστηρίξουν τις νέες τεχνολογίες και να λειτουργήσουν σε πολλαπλές υποδομές μειώνοντας τις πιθανότητες δημιουργίας παλαιών συστημάτων που κληρονομούνται χωρίς δυνατότητες αλλαγών. Επίσης, βασίζουμε την υλοποίησή μας σε υπηρεσίες που μπορούν εύκολα να κλιμακωθούν κάθετα ή οριζόντια για να υποστηρίξουν τις αυξανόμενες ανάγκες που θα αναπτυχθούν με την πάροδο του χρόνου λόγω ανάπτυξης της εγκατάστασης, ελαχιστοποιώντας την ανάγκη επανασχεδιασμού οποιουδήποτε τμήματος του συστήματος.

Όπως αναφέραμε πριν, σε κάθε ένα από τα κτίρια της εγκατάστασης εγκαταστήθηκαν συσκευές που μετράνε (α) τη συνολική κατανάλωση ισχύος του κτιρίου, (β) την περιβαλλοντική ευελιξία κάθε κατηγορίας (βλ. Παρακάτω για περισσότερες λεπτομέρειες) και (γ) και τα επίπεδα ατμοσφαιρικής ρύπανσης σε κάθε κτίριο. Αυτές οι συσκευές μπορούν να χωριστούν σε τρεις διαφορετικές κατηγορίες ανάλογα με τον τύπο προέλευσης και λειτουργίας τους. Σε γενικές γραμμές, χρησιμοποιούμε (α) προσαρμοσμένες συσκευές που επικοινωνούν χρησιμοποιώντας ένα τοπικό δίκτυο IEEE 802.15.4, (β) έτοιμες συσκευές που επικοινωνούν μέσω ασύρματου δικτύου ή τηλεφωνικών δικτύων σε απομακρυσμένες περιοχές, και (γ) αισθητήρες από παλαιότερα συστήματα διαχείρισης κτιρίων που έχουν ήδη εγκατασταθεί σε ορισμένα σχολικά κτίρια.

Στόχος μας είναι να οικοδομήσουμε ένα σύστημα που να είναι σε θέση να χειρίζεται ένα απεριόριστο αριθμό δεδομένων από αυτές τις εγκαταστάσεις σε πραγματικό χρόνο χωρίς καθυστερήσεις ή διακοπές. Το σύστημα έχει σχεδιαστεί με την προσέγγιση των μικρο-εφαρμογών. Ως εκ τούτου, αποτελείται από μια σειρά από χαλαρά συνδεδεμένες εφαρμογές που επικοινωνούν χρησιμοποιώντας διεπαφές ιστού και μια κεντρική υπηρεσία ανταλλαγής μηνυμάτων. Αυτό το κεντρικό σημείο της αρχιτεκτονικής μας είναι μια υπηρεσία 'μεσίτη' μηνυμάτων που επιτρέπει σε όλες τις άλλες υπηρεσίες να δημοσιεύουν μηνύματα ή να εγγράφονται σε ροές δεδομένων και να λαμβάνουν ειδοποιήσεις από αυτές. Ο ρόλος του είναι να εισάγει τα δεδομένα από τις διάφορες ροές δεδομένων από τις εγκαταστάσεις στην κεντρική υπηρεσία επεξεργασίας που αναλύει τα δεδομένα και προωθεία ξανά προς τον 'μεσίτη' μηνυμάτων τα αποτελέσματα για περαιτέρω επεξεργασία ή αποθήκευση. Το σύστημα αυτό ονομάζεται Μηχανισμός Συνεχούς Υπολογισμού. Κατάλληλες διεπαφές προγραμματισμού εφαρμογών παρέχονται για την ανάκτηση πληροφοριών από το σύστημα (ιστορικά δεδομένα ή πληροφορίες της δομής του Διαδικτύου Αντικειμένων) για τη δημιουργία τελικών εφαρμογών προς τους χρήστες του GAIA(Σχήμα Β΄.2).

Για την ανάλυση των δεδομένων χρησιμοποιούμε το σύστημα επεξεργασίας ροών δεδομένω Apache Storm, ένα δωρεάν και ανοιχτού κώδικα κατανεμημένο σύστημα υπολογισμού για δεδομένα πραγματικού χρόνου. Το σύστημα αυτό μας επιτρέπει να χωρίσουμε όλα τα βήματα της επεξεργασίας των δεδομένων σε απλές εργασίες που εκτελούνται ασύγχρονα αλλά με τη σωστή σειρά, σχηματίζοντας ένα αγωγό μετασχηματισμών που εφαρμόζονται στα αρχικά δεδομένα που εισέρχονται στο σύστημά μας. Αυτός ο σχηματιζόμενος αγωγός ονομάζεται **τοπολογία επεξεργασίας**. Η ανάλυση που εκτελούμε σε κάθε βήμα γίνεται με βάση τον χρόνο για την εξαγωγή στατιστικών αποτελεσμάτων σε διαφορετικές χρονικές περιόδους (5 λεπτά, 1 ώρα, κλπ). Σε κάθε μια από αυτές τις χρονικές περιόδους μπορούμε να εκτελέσουμε μια ή περισσότερες στατιστικές αναλύσεις όπως εξαγωγή μέσου όρου, αθροισμάτων, εύρεση μεγιστων/ελαχίστων κλπ.

Το βασικό στοιχείο για την αξιολόγηση ενός τέτοιου συστήματος είναι ο χρόνος απόκρισής του και εξαγωγής των αποτελεσμάτων μετά την λήψη των τιμών από τις ροές δεδομένων. Στην



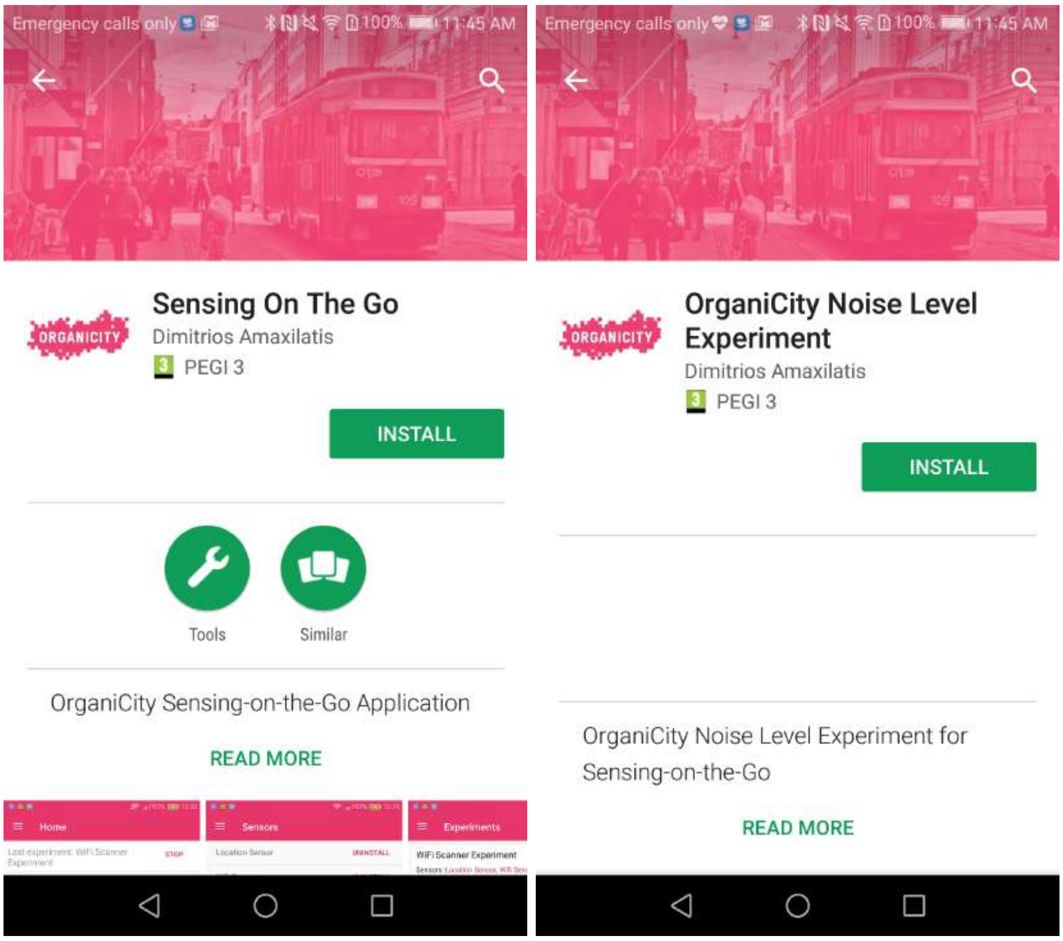
Σχήμα Β΄.2 Αρχιτεκτονική συστήματος συλλογής και επεξεργασίας δεδομένων της εγκατάστασης Διαδικτύου Αντικειμένων GAIA.

δική μας περίπτωση για την ανάλυση όγκου δεδομένων που ανέρχεται μέχρι και στις 500 μετρήσεις ανά δευτερόλεπτο απαιτούνται χρόνοι μικρότεροι του ενός χιλιοστού του δευτερολέπτου (0.6ms για εξαγωγή μέσων όρων και 0.3ms για ανάλυση της κατανάλωσης ενέργειας).

Συλλογή δεδομένων μέσω εθελοντών για έξυπνες πόλεις

Στα πλαίσια μιας πόλης η εγκατάσταση εκτενών δικτύων συσκευών παρακολούθησης περιβαλλοντικών ή άλλων συνθηκών είναι αρκετές φορές ανέφικτο ειδικά σε περιόδους λιτότητας. Στα πλαίσια αυτής της ΔΔ και μέσω το ευρωπαϊκού ερευνητικού έργου OrganiCity στοχεύουμε στην δημιουργία μιας υπηρεσίας συλλογής δεδομένων από εθελοντές με χρήση φορητών συσκευών όπως κινητά τηλέφωνα. Η διαδικασία αυτή ονομάζεται αίσθηση πλήθους και βασίζεται στην ανάθεση εργασιών συλλογής δεδομένων σε ένα πλήθος εθελοντών που τις εκτελούν κατά τη διάρκεια της καθημερινής τους μετακίνησης μέσα στην πόλη, είτε με στόχο την ανιδιοτελή προσφορά προς της κοινότητα της πόλης τους, είτε με στόχο την διεκδίκηση κάποιας ανταμοιβής που μπροεί να προσφέρει ο ενδιαφερόμενος για τα δεδομένα. Τα έξυπνα κινητά τηλέφωνα είναι σήμερα εξοπλισμένα με έναν αριθμό ενσωματωμένων αισθητήρων και διεπαφές δικτύωσης για να επικοινωνούν μεταξύ τους και με συσκευές Διαδικτύου Αντικειμέννω όπως smartwatches, ιχνηλάτες φυσικής κατάστασης ή ειδικά κατασκευασμένες συσκευές (πχ. βασισμένες σε μικροελενκτές Arduino).

Για τον σκοπό αυτό υλοποιήσαμε την εφαρμογή Sensing-on-the-Go, μια εφαρμογή για έξυπνα κινητά τηλέφωνα Android(Σχήμα Β΄.3). Η εφαρμογή αυτή ακολουθεί τις ιδέες των εφαρμογών Διαδικτύου Αντικειμένων, με τα εξυπνα κινητά τηλέφναν να συμμετέχουν σε εκστατείες συλλογής δεδομένων που βρίσκονται διαθέσιμες στην περιοχή τους μέσω των υπηρεσιών του OrganiCity. Πέραν της εφαρμογής κινητών τηλεφώνων έχει υλοποιηθεί και μια διαδικτυακή εφαρμογή η οποία προσφέρει πρόσβαση στα δεδομένα που συλλέγονται από τις εκτελούμενες καμπάνιες αλλά και προσφέρει προγραμματιστικές διεπαφές για την εφαρμογή κινητών τηλεφώνων προκειμένου να αποστέλονται τα δεδομένα που συλλέγονται.



Σχήμα Β΄.3 Δημοσίευση εφαρμογής Sensong-on-the-Go στο Google Play Store.

Η υλοποίηση της συλλογής δεδομένων γίνεται με βάση τις δυνατότητες που προσφέρει το λειτουρικό σύστημα Android για την εκτέλεση εξωτερικών διεργασιών μέσα από μια ανεξάρτητη εφαργμογή. Οι εξωτερικές αυτές διεργασίες, αναλαμβάνουν τη συλλογή των δεδομένων και την αποστολή τους και την αποστολή τους στην κεντρική εφαρμογή. Η δομή αυτή, μας δίνει τη δυνατότητα να εμπλουτίζουμε την λειτουργικότητα της εφαρμογής για την συλλογή όλο και περισσότερων τύπων μετρήσεων χωρίς να χρειαζόμαστε αλλαγές στην βασική μας εφαρμογή. Επίσης δίνει την δυνατότητα, σε εξωτερικούς χρήστες και προγραμματιστές να υλοποιήσουν μια δική τους διεργασία συλλογής δεδομένων και να χρησιμοποιήσουν το Sensing-on-the-Go για να αποκτήσουν πρόσβαση σε όλη την υποδομή συλλογής δεδομένων και στο πλήθος των εθελοντών της. Η ανάπτηξη των διεργασιών συλλογής δεδομένων γίνεται σε περιβάλλον Android με αποτέλεσμα την όσο το δυνατόν πιο απλή διαδικασία υλοποίησης και δοκιμής από τους ίδιους τους προγραμματιστές. Η εφαρμογή είναι διαθέσιμη σε όλους μέσω του Google Play Store και η συλλογή των δεδομένων μπορεί να γίνει είτε μέσω ενός προσωπικού λογαριασμού στην πλατφόρμα του OrganiCity είτε ανώνυμα, δίχως την απαίτηση για την αποστολή οποιονδήποτε προσωπικών στοιχείων.

Στα πλαίσια του έργου OrganiCity η εφαρμογή αυτή χρησιμοποιήθηκε από 4 διαφορετικές ομάδες εθελοντών οι οποίες ανέπτυξαν δικές τους καμπάνιες συλλογής δεδομένων και της πραγματοποίησαν στις πόλεις τους σε πραγματικές συνθήκες με τη συμβολή εθελοντών πολιτών. Σε μια από αυτές τις καμπάνιες συγκεντρώθηκαν δεδομένα από τα διαθέσιμα WiFi access points για 3 ευρωπαικές πόλεις (Πάτρα, Λονδίνο και Σαντεντέρ) μέσα σε 5 ημέρες και από 14 διαφορετικούς χρήστες. Το σύνολο των δεδομένων που συγκεντρώθηκε ήταν 7500 μετρήσεις και για τις 3 πόλεις (κατά μέσο όρο 500 μετρήσεις ανά συμμετέχοντα).

Εξαγωγή γνώσης από δεδομένα Διαδικτύου Αντικειμένων

Τα 2 βασικότερα ερωτήματα σχετικά με τη συλλογή των δεδομένων από εγκαταστάσεις Διαδικτύου Αντικειμένων είναι τα εξής:

- Τι κάνουμε με όλα αυτά τα δεδομένα·
- Πως μπρούμε να μάθουμε πράγματα από αυτά τα δεδομένα·

Μια απάντηση και στις δύο ερωτήσεις είναι η εξαγωγή **γνώσης**. Ο όρος **γνώση** αναφέρεται σε κάτι πραγματικά χρήσιμο, ξεπερνώντας κάποια απλά νούμερα. Θα μπορούσε να περιγραφεί επίσης και ως κάτι που παρέχει αληθινή χρησιμότητα στους χρήστες ενός συστήματος. Για παράδειγμα, ορισμένα συμβάντα μπορούν να παράγουν δεδομένα από την υποδομή Διαδικτύου Αντικειμένων μιας πόλης, αλλά χωρίς αξιόπιστες επισημάνσεις πάνω από αυτά είναι αδύνατον να εντοπίσουμε το ενδεχόμενο κυκλοφοριακής συμφόρησης στο κέντρο της πόλης. Μπορούμε να παρατηρήσουμε χαμηλές τιμές για την ταχύτητα των αυτοκινήτων, του θορύβου και της συγκέντρωσης εκπομπών ρύπων αλλά χωρίς τον συνδυασμό τους είναι αδύνατον να το εντοπίσουμε με βεβαιώτητα. Οι επισημάνσεις αυτές μας δίνουν την δυνατότητα να προσπαθήσουμε να καταλάβουμε τι είναι αυτό που κρύβεται πίσω από τις αριθμητικές τιμές που βλέπουμε αρχικά.

Στα πλαίσια αυτής της ΔΔ υλοποιήσαμε ένα σύστημα το οποίο μπορεί με ημι-αυτόματο τρόπο να αναγνωρίσει και να εξάγει αυτές τις επισημάνσεις από τα αριθμητικά δεδομένα με χρήση τεχνικών μηχανικής μάθησης το οποίο ονομάζουμε JAMAiCA. Το σύστημα αυτό μπορεί να εκτελέσει 2 διαφορετικές κατηγορίες ανάλυσης δεδομένω: εύρεση ανωμαλιών και ταξινόμηση ωστόσο η δομή του συστήματος είναι παραμετροποιήσιμη έτσι ώστε να μπορεί εύκολα να υποστηρίξει περισσότερες εργασίες. Για την εκτέλεση των αλγορίθμων μηχανικής μάθησης χρησιμοποιούμε 2 πλατφόρμες που εξυπηρετούν αυτόν τον σκοπό: το Jubatus και την JavaML.

Για την αξιολόγηση του συστήματος Jubatus εκτελέσαμε πολλαπλά πειράματα με χρήση δεδομένων και από τις 2 πηγές που είχαμε στην διάθεσή μας (OrganiCity και GAIA). Για την πρώτη περίπωση, αναλύσαμε περισσότερες από 40000 μετρήσεις συγκεντώσεων μικροσωματιδίων PM_10 μέσα σε μια χρονική περίοδο 20 ημερών. Το σύστημά μας ήταν ικανό να εντοπίσει σε πραγματικό χρόνο προβληματικές τιμές (τιμές εκτός επιτρεπτών ορίων, και τιμές προβληματικών αισθητήρων) χωρίς προβλήματα. Ένα άλλο πείραμα εκτελέστηκε με βάση τα δεδομένα του έργου GAIA με σκοπό την κατάταξη των συνθηκών σε αίθουσες σχολείων του προγράμματος σε 4 κατηγορίες με βάση τιμές θερμοκρασίας και υγρασίας που είχαν συλλεχθεί για αυτές.

Συμπεράσματα

Στο πλαίσιο της παρούσας ΔΔ αντιμετωπίσαμε τα προβλήματα που προκύπτουν από την εισαγωγή ολοένα και περισσότερων έξυπνων συνδεδεμένων συσκευών στο καθημερινό μας περιβάλλον. Αυτή η αυξανόμενη χρήση τέτοιων συσκευών κάνει επιτακτική την ανάγκη εύρεσης μιας βελτιωμένης μεθοδολογίας ανάλυσης των δεδομένωνπου παράγονται από τις συσκευές αυτές. Η δουλειά μας βασίστηκε σε δύο πραγματικές περιπτώσεις χρήσης εγκαταστάσεων Διαδικτύου Αντικειμένων ως μέρος 2 ερευνητικών προγραμμάτων της ευρωπαικής ένωσης. Τα προγράμματα μας παρείχαν ένα πεδίο δοκιμών για τις εφαρμογές μας σε πραγματικές συνθήκες. Ως αποτέλεσμα, φτιάξαμε εφαρμογές που είναι ικανές να συλλέγουν και να αναλύουν δεδομένα αποτελεσματικά από αυτές τις πηγές και να τα παρέχουν στους τελικούς χρήστες και τους προγραμματιστές εφαρμογών σε σχεδόν πραγματικό χρόνο. Είμαστε επίσης σε θέση να δημιουργήσουμε πρόσθετη γνώση από τα πρωτογενή δεδομένα που λαμβάνονται. Αποδείξαμε επίσης ότι η αρχιτεκτονική του συστήματος μας είναι ικανή να χειρίζεται διάφορους τύπους και όγκους δεδομένων και με ελάχιστες αλλαγές σε επίπεδο υλοποίησης, με την αρθρωτή σχεδίασή του. Από την ενασχόλησή μας αυτή λάβαμε τα εξής μαθήματα:

- Οι συσκευές Διαδικτύου Αντικειμένων τείνουν να είναι εξαιρετικά αναξιόπιστες, ειδικά όταν εγκαθίστανται σε εξωτερικούς χώρους έξω από τον άμεσο έλεγχό μας. Μερικοί από τους χρήστες τείνουν επίσης να προκαλούν προβλήματα στις συσκευές, με πρόθεση ή όχι, προκαλώντας έτσι προβλήματα στη διατήρηση των ροών δεδομένων από τις εγκαταστάσεις.

- Επίσης, οι αποτυχίες των αισθητήρων ή τα προβλήματα συνδεσιμότητας είναι μια συνηθισμένη περίπτωση και θα εμφανίζονται σε οποιαδήποτε εγκατάσταση ανεξάρτητα από την ποιότητα των υλικών ή τον κατασκευαστή τους.

- Τέλος, στην περίπτωση του της συλλογής δεδομένων μέσω εθελοντών, είναι σημαντικό να σημειωθεί ότι οι διεπαφές χρήστη που αναπτύσσονται είναι εξαιρετικά σημαντικές για την επιτυχία του συστήματος. Στην περίπτωση που η διεπαφή χρήστη είναι πιο περίπλοκη από ό, τι απαιτείται, οι χρήστες τείνουν να χάνουν το ενδιαφέρον και την θέληση να την χρησιμοποιήσουν περισσότερο.

Μια σημαντική επέκταση στο έργο που παρουσιάζεται σε αυτή τη ΔΔ θα ήταν η επιδίωξη της δυνατότητας διαίρεσης της ανάλυσης των συλλεγόμενων δεδομένων σε περισσότερα από ένα στρώματα. Στην περίπτωση αυτή, κάποια επεξεργασία μπορεί να συμβεί στις εγκαταστάσεις του χρήστη, μειώνοντας έτσι την ανάγκη μεταφοράς δεδομένων μέσω του Διαδικτύου. Αυτή η περίπτωση θα μας βοηθήσει επίσης να μειώσουμε τυχόν απώλειες δεδομένων από προβλήματα δικτύωσης και συνδεσιμότητας, τα οποία είναι αρκετά κοινά με τις εμπορικές διαδικτυακές συνδέσεις. Αυτή η τάση, που ονομάζεται υπολογιστική άκρη ή ομίχλη, είναι ένα εξαιρετικά ενδιαφέρον και ελπιδοφόρο πεδίο έρευνας που μπορεί να αυξήσει σημαντικά την αποδοτικότητα του συστήματος. Οι περισσότερες επεκτάσεις θα μπορούσαν να επικεντρωθούν στην προσθήκη περισσότερων τύπων δεδομένων στις ενότητες ανάλυσης του συστήματός μας, όπως το βίντεο ή το ήχο, καθώς η φύση τους θα μπορούσε να αποκαλύψει πιο περίπλοκες απαιτήσεις που δεν διερευνήθηκαν.