A Lightweight IoT Hub for SME Manufacturing Industries

Carlos Agostinho[1], Fabio Lopes[1], Jose Ferreira[1], Sudeep Ghimire[1] and Maria Marques[1]

[1] Centre of Technology and Systems, CTS, UNINOVA
Campus da Caparica, 2829-516 Caparica, Portugal
{ca, fl, japf, sud, mcm}@uninova.pt

Abstract. With the advance of the Internet of Things (IoT), new ways of acquiring, processing and managing collected data from electronic devices are being developed to provide support for more complex systems. This process of transforming the acquired data from the physical world, through the sensors, into viable information on which the applications can make decisions upon, must consider the various implementation scenarios, the business and technical requirements, such as security, privacy and interoperability between heterogeneous devices (which often communicate using different protocols and require a common vocabulary). With the increasing complexity of these requirements, it becomes urgent to develop an infrastructure to handle the associated processes and provide a middle ground layer on which the physical and digital world are connected and translated into each other. This software layer, or middleware, can be described as a hub and aims to fill the gap between devices and information systems. This work contributes with a study of mechanisms and methodologies for the collection of data, interoperability of systems and data filtering, to optimize and automate, using a lightweight approach, the collection and pre-analysis of the data to be used by the various applications of the IoT systems, such as the SME manufacturing industries.

Keywords: Internet of Things, Hub, Middleware, Data Acquisition, Data Filtering, Sensors, Interoperability.

1 Introduction

The Internet of Things (IoT) is an emerging technological concept that aims to combine consumer products, industrial components and other everyday objects with Internet connectivity and powerful data analytic capabilities that can transform the way we work and live [1]. The objective of this is to optimize the functionalities of each object, as every extractable information becomes a mean of analyzing and computing the functioning processes, to enhance its context functioning and to provide new purposes, that emerge when connecting a certain object to an intelligent network [2].

The IoT has the potential to change how people can be connected to objects, that can also be connected among themselves, creating intelligent technological environments, generating interest both in the business and scientific areas. Some of the addressed issues related to the devices, include different manufacturing specifications and the lack of a common vocabulary, due to the different sources of models and semantics [3]. This may pose some barriers to interoperable and scalable IoT networks. The current research efforts involve connecting the machines, equipment, software and devices using internet protocols for allowing communication without human intervention [4]. Hence, a methodology to realize homogeneous communication between applications and devices, and between the devices, should be applied.

One of the solutions that can be adopted is a middleware technology, a layer of software between the two systems that makes it easy for the two to communicate [5]. Figure 1 shows three different approaches for the data collection and integration of heterogenous devices where the proposed one is a middleware-based approach based on fog computing paradigm (b), providing an abstraction layer between devices and applications/services. The advantages consist in the possibility of implementing independent processes that enhance the quality of the methodology, providing more reliable systems, and also diminishing the required data processing in the applications. This solution is modular and applicable to a wider range of scenarios, due to its independency from the other layers (cloud and devices). It is appropriate for SME's, representing a cheaper and easier solution due to the possibility of using low-cost equipment. Some of the useful applicable processes in this situation are data validation and filtering, event detection and security checking.

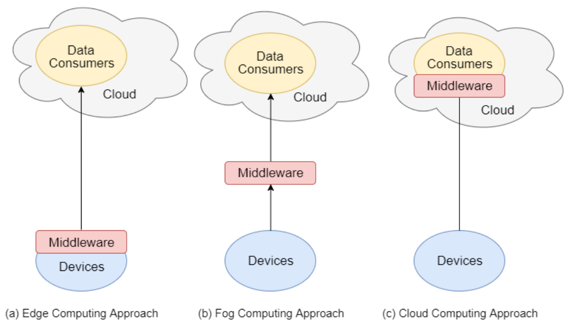


Fig. 1. Middleware-based approaches for data collection through sensor devices.

The other two solutions, provided in Fig. 1, are based on the processing being made directly at the devices (a) applying the edge computing paradigm, or being made in the data consumers (c) following the traditional cloud paradigm. The first, generally, does not allow complex processes due to the simplicity of the devices used in contemporary industry (and when it allows, they are very expensive), and the second, requires processing from the data consumers, increasing complexity in the applications and consuming processing time from other important application processes.

In this work, a solution for the creation of a middleware is presented, envisioning its industrial application in a real-world scenario related with the European C2NET project [6], to be explained further in this paper. The idea is to create a lightweight IoT Hub which physically integrates the IoT network for communication purposes and provides middleware software processes.

2 Background

To specify and better understand the methodology proposed in this paper and how the solution contributes to IoT, some key concepts are mentioned and briefly explained.

Data Acquisition. This process consists on the collection of information and the system responsible for the regulation of data acquisition, defined as the monitor [7]. In IoT, this acquisition is heavily based on the low-layer devices from the network, as depicted in Fig. 1, and is based on finding, fetching and transferring raw data to be processed and indexed by higher layers of the system [8].

Context Acquisition. An important aspect of the data collection is to clarify the meaning of data sources. Sensors are hardware components that measure environmental information such as temperature, location and processing time, and transform it into a digital signal. In larger networks, the amount of generated data is critically larger, and it is not feasible to process everything that is generated by the devices. Considering that, context-awareness processes play a critical role in deciding what data needs to be processed, which implies the understanding of sensor data, one of the main challenges of IoT. The concept of context management is also very important and essential for the software systems that apply this context modelling and reasoning. Data moves from phase to phase, from the place where it is generated to where it is consumed, creating a data life cycle that starts with the acquisition [9]. The acquisition parameters, such as responsibility (data authoring), frequency, source, sensor type (software, virtual or physical sensor), are thoroughly explained in the works of [10] [11] and considered in this solution.

IoT Protocols. Communication protocols are formal descriptions of digital message formats and define rules that include: packet size, transmission speed, handshaking and synchronization techniques, address mapping, flow control and other important communication aspects to achieve proper information exchange [12]. They are implemented in hardware (communication protocols) and software (message protocols) and used to exchange information between systems.

In IoT, there are various data protocols used for communication, that usually fall into three communication types: Device-to-Server (e.g. MQTT – http://mqtt.org/), Server-to-Server (e.g. AMQP - https://www.amqp.org) and Device-to-Device (e.g. DDS - http://portals.omg.org/dds/). Some communication/transport layer protocols, commonly used in IoT, are Wi-Fi, Bluetooth, ZigBee and Ethernet.

Data Pre-Processing. This is an important aspect of the data collection, using techniques to ignore noisy and unreliable data before the application layer. The objective is to transform raw data, often incomplete or inconsistent, into an understandable format that can be defined by the communication hub or the requesting data consumer. Some techniques used, mentioned in the work of [13], are Data Cleaning (identifying and resolving data inconsistencies), Missing Values (fill probable values that are missing), Noisy Data (corrects random error or variance in a measured variable) and

Inconsistent Data (uses external references for correcting the information, based on aspects such as functional dependencies and attributes).

Middleware Solutions for IoT. The IoT environment holds a wide spectrum of applications, which make use of middleware layers to achieve the previously explained benefits. The main technical aspects to consider when designing middleware systems are the capability of interoperability, scalability, abstraction, interaction, multiplicity, security and privacy. The existing solutions can be categorized according to the involved domains into three categories [14]: Semantic Web and Web Services, RFID and Sensor Networks, and Robotics. Some of the most prominent approaches are Triple Space-Based [15], UBIWARE [16] and TinyREST [17].

3 Lightweight IoT Hub Architecture

In this section, the developed architecture for the Lightweight IoT Hub is presented. It is designed to comply with the functionalities mentioned in the Introduction along with the specifications of the C2NET project, and considers the researched information provided in the Background section.

The Lightweight IoT Hub was designed to provide C2NET SME's with a low processing, but robust, solution for data collection. This low processing requirement, allows the use of a diverse range of readily available hardware to build IoT applications, not requiring a lot of processing capabilities. Examples of these hardware options are the single-board computers called *Raspberry Pi*[1] and the microcontrollers from the *Arduino*[2] series, which are used in the implementation of this solution for the Hub and sensor nodes, respectively. Hence, the Lightweight IoT Hub provides a solution that is platform-independent and supports diverse communication protocols. It is also written in Java for portability purposes and targets Unix systems which provide device drivers for the majority of the ISO standards.

The Hub is composed by a set of sub-modules, presented in Fig. 2 and further detailed in section 3.1, that focus on the system's communication, management of devices and data handling. The **Communication Module** is responsible for the communication between the IoT systems and the C2NET platform, guaranteeing the external components' interoperability, protocol abstraction and connection between consumers and the data providers. The **Device Management Module** manages the different types of IoT devices in the IoT network, ensuring their security and connectivity. The **Data Handling Module** allows the manipulation of data at the source, filtering and reducing the data flow before it reaches the application, the C2NET platform in this case. This improves performance and reduces inaccuracy in the retrieved data.

The IoT Hub is an external component relatively to the cloud platform. It communicates with it using a publish/subscribe (Pub/Sub) message queue, feeding the C2NET cloud platform with real-time data coming from the different IoT data sources (devices). The communication is bidirectional, enabling not only data input but also a

[1] https://www.raspberrypi.org

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

configuration flow from the platform to the various possible existing Hubs, virtualizing the IoT network. Following this approach, the user (e.g. company IT) can register different devices and allocate specific configurations for how to receive the data.

3.1 Developed Architecture

The development of the architecture for the Lightweight IoT Hub started at the lower layer, with the scrutiny of the specifications for the contemporary sensor networks and the analysis of which protocols are best suited for the intended deployment of this work's solution. With this, considering the necessities of the C2NET project pilots, it was clear the need for event-driven processes, because these networks are typically intensively active for short periods of time and remain idle for the remaining periods. Hence, the Lightweight IoT Hub considers that sensors broadcast results immediately and the hub is responsible for managing thresholds, filter unwanted data and forward the information to the C2NET platform in a predefined time frame. This allows for a larger variety of sensors to be used as it reduces the need for processing capability of each node while also reducing power consumption. Also, the volume of data doesn't require huge amounts of memory in the Hub, achieving one objective of the Hub that is to provide a reliable solution with low memory provisions.

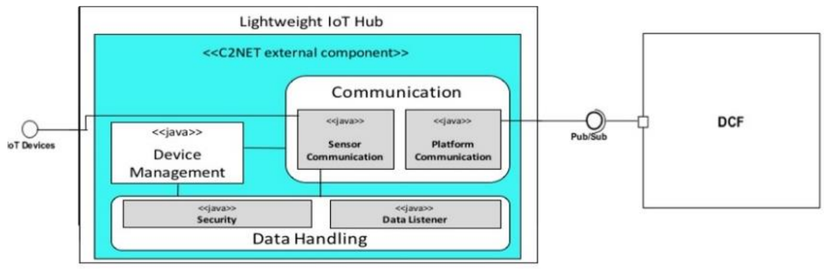


Fig. 2. Lightweight IoT Hub deployment view.

Infrastructural approaches based on communication layer protocols such as HTTP and ZigBee, with a shared medium and limited spectrum, are supported and fit perfectly the needs of home/office wireless sensor networks, while dedicated Can-Bus and Serial communication approaches are more appropriate for heavy industrial environments (as used in the implementation of this work), enabling lower bit error rates.

Fig. 2 shows a technical version of the Hub, where the sub-modules internal content is depicted. This sub-section, the information from the previous sub-section is completed with the specifications of each sub-module.

The **Device Management Module** allows the discovery and connection of several different sensors to the Hub, handling the different communication layer protocols according to the user configuration. It is responsible for monitoring the connectivity and availability of the devices, with the support of a simple database.

The **Communication Module** is divided into two sub-modules: **Sensor Communication** and **Platform Communication**. The Sensor Communication is related to the device management, enabling the interaction with devices coupled via one of the possible protocols. The Platform Communication is responsible for all interactions between the IoT Hub and the platform. It is used to receive the configuration of protocols and devices from the platform and forwarding data from sensors to the platform.

Each **Data Listener** of the **Data Handling Module** is an implementation of a protocol used to communicate with sensors. The Data Listener also performs data filtering activities when specified by the user during the resource network configuration. The Data Listener must be compliant with the requirements of the Hub and the platform, but developers can write their own Data Listeners according to their needs.

Finally, the **Security Module** is responsible for providing authentication and authorization for devices connected to the Lightweight IoT Hub. All communications with the C2NET platform must be authenticated, obtained via secured channels and immune to repetition and violation of data integrity. For this, HTTP over SSL with symmetric key cryptography is used and the Hub will request a public key via the Pub/Sub component. Within the platform, ACS will hold the private key that authenticates the Hub. This module provides features that allow the monitoring of the hub's hardware and informs the user about the global behaviour of the system.

3.2 Communication

The communication of the Hub with the sensor nodes and the C2NET platform (Fig. 3) is a very important process of the Hub's functioning, allowing the essential interoperability that makes it an IoT system element, and also providing certain functionalities such as the management of sensors and data collecting procedures, configuration exchange and data forwarding (to the cloud platform).

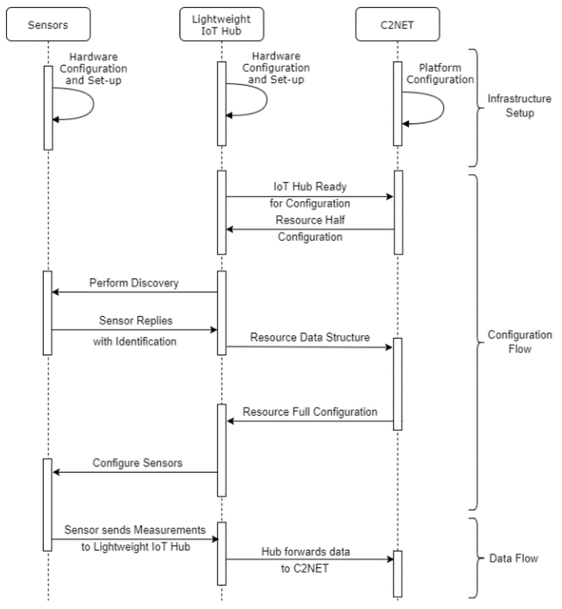

Sensors
Lightweight IoT Hub
C2NET
Hardware Configuration and Set-up
Hardware Configuration and Set-up
Platform Configuration
Infrastructure Setup
IoT Hub Ready for Configuration
Resource Half Configuration
Perform Discovery
Sensor Replies with Identification
Resource Data Structure
Configuration Flow
Resource Full Configuration
Configure Sensors
Sensor sends Measurements to Lightweight IoT Hub
Hub forwards data to C2NET
Data Flow

Fig. 3. Lightweight IoT Hub communication.

The sensors, pre-configured for communication, send data over the network, from the node to the hub, where the data is collected for further processing (on the hub or the platform). The hub is responsible for actuating on the devices, enabling them or disabling them, and is responsible for the mechanisms to detect new devices, or nodes, automatically (as seen in Fig. 3). The data retrieved from the sensors is then forward to the platform for further analysis and display.

3.3 Messaging between Hub and Devices

The messages that allow the management mentioned in the previous sub-section are also important to consider. In this case, the structure of the messages is designed

to be constituted by a maximum of 8 bytes, where each one of them contains specific data. The first byte is always used for the message type, which defines the use of the remaining bytes. Typically, the second byte denotes the node ID, the third defines the sensor ID (within each node), the fourth denotes the type of sensor (useful for messages from the node to the hub, where the hub acknowledges the existence of that sensor and what type of data it measures) and the remaining bytes are usually for values of measurements, thresholds and frequencies (for communication to happen). Those values may be very high and need to be represented by more than two bytes.
An example of this structure is presented in Fig. 4. It's a message intended for the actuation of a warning light. The actuator is in node 3 and its ID is 10. To activate it, the Hub should make the value 1 reach the correspondent node. So, the first byte has the pre-defined message type code "0x17" that denotes the message for actuation, "0x03" is the node ID and "0x10" is the actuator ID, and occupy the bytes 2 and 4, respectively. This type of actuator has a pre-determined value for its designation, "0x30", and that information is inserted into the third byte of the message. The value for actuation purposes occupies the fifth byte and to activate this warning light "0x01" is written. The three remaining bytes are not used for this type of message.

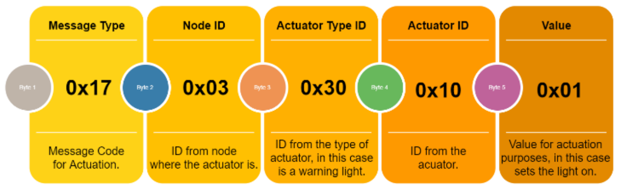

Message Type
Byte 1
0x17
Message Code for Actuation.
Node ID
Byte 2
0x03
ID from node where the actuator is.
Actuator Type ID
Byte 3
0x30
ID from the type of actuator, in this case is a warning light.
Actuator ID
Byte 4
0x10
ID from the acuator.
Value
Byte 5
0x01
Value for actuation purposes, in this case sets the light on.

Fig. 4. Communication between Hub and Node message example.

4 Implementation of the Use Case Scenario

As it was mentioned, this work is related to the C2NET project and aims for the deployment of the Lightweight IoT Hub in the SME's manufacturing industries. The goal of the C2NET project is the creation of cloud-enabled tools for supporting the SME's supply network optimization of manufacturing and logistic assets based on real-time collaborative demand, production and delivery plans [6]. Considering this, the Hub stands as the centralized element which provides the capability to communicate with the cloud-enabled tools and managing the factory's network.

4.1 Infrastructure Set-up

For testing and validation purposes, the resultant solution of this work had a direct impact on the processes a metalworking factory in Portugal. The resultant IoT implementation of this deployment is illustrated in Fig. 5. The Hub is implemented in a *Raspberry Pi* single-board computer and the sensor nodes (C1, E1, 31, A1 and D1) are implemented using *Arduino* devices, with the exception of B1 which is connected to a thickness gauge sensor that requires more processing capabilities. Hence B1 is implemented in a *Raspberry Pi*.

4.2 Configuration

An example of the resulting configuration settings for a specific node, named “Painting Station”, is presented in Fig. 6. These settings represent the message structure sent by the platform to the hub for defining from which sensors to start collecting data. For this to happen, the hub has already had to discover what sensors are present in the network and query for their information, such as type of sensor, data type, range, data units, etc. This information is passed to the hub and only after that, the platform user sends the pre-defined settings for the data collection. It is important to mention that the Device Management module requires the information about the protocol to be used before performing the device discovery processes. This is possible by configuring the internal config.xml file and activating the appropriate Data Listener. The configuration presented in Fig. 6 is stored and monitored by the device management.

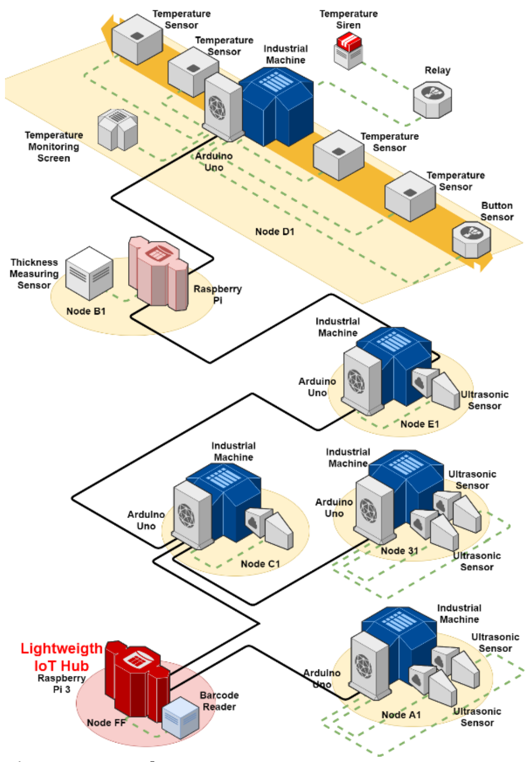
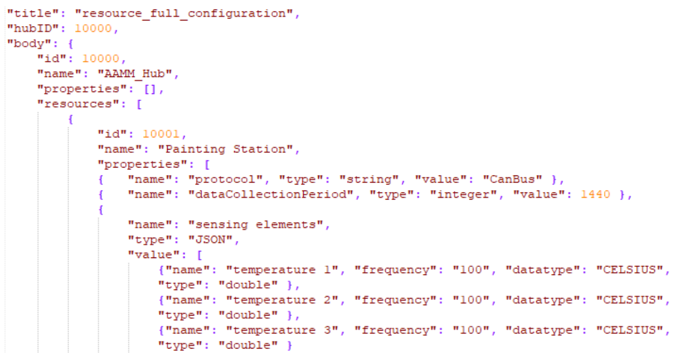


Fig. 5. IoT Implementation.

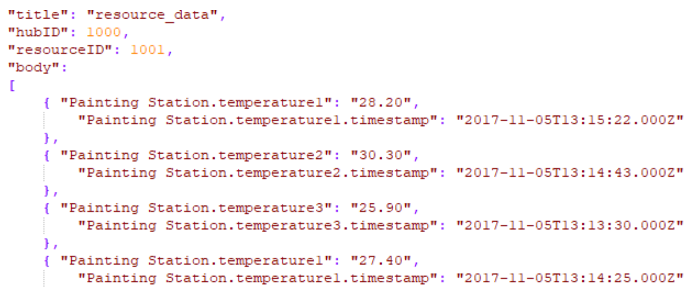


```
"title": "resource_full_configuration",
"hubID": 10000,
"body": {
    "id": 10000,
    "name": "AAMM_Hub",
    "properties": [],
    "resources": [
        {
            "id": 10001,
            "name": "Painting Station",
            "properties": [
            {   "name": "protocol", "type": "string", "value": "CanBus" },
            {   "name": "dataCollectionPeriod", "type": "integer", "value": 1440 },
            {
                "name": "sensing elements",
                "type": "JSON",
                "value": [
                    {"name": "temperature 1", "frequency": "100", "datatype": "CELSIUS",
                    "type": "double" },
                    {"name": "temperature 2", "frequency": "100", "datatype": "CELSIUS",
                    "type": "double" },
                    {"name": "temperature 3", "frequency": "100", "datatype": "CELSIUS",
                    "type": "double" }
```

Fig. 6. Lightweight IoT Hub message structure for the settings of data collecting.

4.3 Data Collection

After the configuration, the selected IoT devices start to transmit data. The sensor data is temporarily kept, with a timestamp, in memory storage. To forward the data to the C2NET cloud platform, as intended, the Platform Communication uses the message structure presented in Fig. 7. This information is sent to the platform, for displaying the collected data. The platform, accessible to the company’s IT personnel, allows to perform analysis on the collected data and perform configurations on various aspects of the overall IoT process.



```
"title": "resource_data",
"hubID": 1000,
"resourceID": 1001,
"body":
[
    { "Painting Station.temperature1": "28.20",
        "Painting Station.temperature1.timestamp": "2017-11-05T13:15:22.000Z"
    },
    { "Painting Station.temperature2": "30.30",
        "Painting Station.temperature2.timestamp": "2017-11-05T13:14:43.000Z"
    },
    { "Painting Station.temperature3": "25.90",
        "Painting Station.temperature3.timestamp": "2017-11-05T13:13:30.000Z"
    },
    { "Painting Station.temperature1": "27.40",
        "Painting Station.temperature1.timestamp": "2017-11-05T13:14:25.000Z"
```

Fig. 7. Lightweight IoT Hub message structure for the settings of data collecting.

5 Conclusion and Future Work

In an IoT paradigm, the distributed scenarios prevail where the data sources are physically separated and are often directed to autonomous data consumers, which are often the higher-level applications with functionality for high-end data analytics and event detection. Still, this type of distributed scenarios, typically has issues regarding scalability and maintenance if the data sources and data consumers are tightly coupled. That means that if there are changes in the data sources or consumers, the overall systems need to also perform changes to deal with the new requirements. So, a viable industrial solution needs to have a system ready for seamless integration of data sources with a high level of abstraction between the data sources and consumers. Middleware aims to reduce the complexity of such systems and processes, providing a fog computation approach where the processing occurs closer to the devices and independently from the applications. The direction of IoT is to have an edge computation approach where the processing occurs in the devices.

In this work, a Lightweight IoT Hub middleware solution is presented, and a prototype has been developed and installed in the premises of a Portuguese metalworking company. The Hub is responsible for addressing interactions made between the C2NET cloud platform and the IoT devices, managing the real-world data collection process. It is a simple, robust, low-processing and low-cost IoT middleware, for easy integration in SME’s. It supports several protocols and implements some basic security mechanisms based on authentication and authorization.

One of the possible enhancements for this work is the integration more elaborate security methods, as for instance the ones described in [18]. Given the tendency to move computing to the edge, it is important to analyze if the middleware software and procedure proposed for the Lightweight IoT Hub fit within the capacity of contemporary smart devices. Finally, when analyzing the RAMI 4.0 architecture [19], it is also possible to identify other future research direction, namely the integration with the digitalization process (transition from the real to digital world) of the architecture, more specifically with the layers of integration and communication, allowing a contextualization in the Industry 4.0 paradigm.

Acknowledgments

The research leading to these results has received funding from the EC HORIZON2020 Program under grant agreement n° C2NET 636909 (http://www.c2net-project.eu/) and nº VF-OS 723710 (http://vf-os.eu/), and from Faculty of Science and Technology of Nova University of Lisbon.

References

1. Rose Jaren, Eldridge Scott, C.L.: The internet of things: an overview - Understanding the issues and challenges of a more connected world. Internet Soc. (2015).
2. Lopes, F., Ferreira, J., Jardim-goncalves, R., Agostinho, C.: Semantic Maps for IoT Network Reorganization in face of Sensor Malfunctioning, 2017.
3. Agostinho, C., Sarraipa, J., Goncalves, D., Jardim-Goncalves, R.: Tuple-based semantic and structural mapping for a sustainable interoperability. In: IFIP Advances in Information and Communication Technology. (2011).
4. Huang, Y., Li, G.: Descriptive models for Internet of things. In: Proceedings of 2010 International Conference on Intelligent Control and Information Processing, 2010.
5. Rosenblum, D.S.: Interoperability & Middleware. 1–5 (2001).
6. C2NET: C2NET Project, http://c2net-project.eu/.
7. Ojha, S., Bandil, P.L.: IoT based Data Acquisition system using Raspberry Pi. (2016).
8. Fersi, G.: Middleware for internet of things: A study. In: Proceedings - IEEE International Conference on Distributed Computing in Sensor Systems, DCOSS 2015.
9. Fox, P.: Data Management Considerations for the Data Life Cycle. NRC STS 2011.
10. Pietschmann, S., Mitschick, A., Winkler, R., Meißner, K.: CROCO: Ontology-based, cross-application context management, SMAP 2008. (2008).
11. Chen, H., Finin, T., Joshi, A., Kagal, L., Perich, F., Chakraborty, D.: Meet the Semantic Web in Smart Spaces. IEEE Internet Comput. 8, 69–79 (2004).
12. Salman, T., Jain, R.: Networking protocols and standards for internet of things. In: Internet of Things and Data Analytics Handbook. pp. 215–238 (2017).
13. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques. (2012).
14. Chaqfeh, M.A., Mohamed, N.: Challenges in middleware solutions for the internet of things. Proc. 2012 Int. Conf. Collab. Technol. Syst. CTS 2012,. (2012).
15. Gómez-Goiri, A., López-De-Ipiña, D.: A triple space-based semantic distributed middleware for internet of things. In: Lecture Notes in Computer Science (2010).
16. Katasonov, A., Kaykova, O., Khriyenko, O., Nikitin, S., Terziyan, V.: Smart Semantic Middleware for the Internet of Things. ICINCO-ICSO. 169–178 (2008).
17. Luckenbach, T., Gober, P., Arbanowski, S., Kotsopoulos, A., Kim, K.: TinyREST: A protocol for integrating sensor networks into the internet. Proc. REALWSN. (2005).
18. Mozzaquatro, B.A., Jardim-Goncalves, R., Melo, R., Agostinho, C.: The application of security adaptive framework for sensor in industrial systems. In: SAS 2016 (2016).
19. Adolphs, P., Bedenbender, H., Dirzus, D., Ehlich, M., Epple, U., Hankel, M., Heidel, R., Hoffmeister, M., Huhle, H., Kärcher, B., Koziolek, H., Pichler, R., Pollmeier, S., Schewe, F., Walter, A., Waser, B., Wollschlaeger, M.: RAMI Industrie 4.0, (2015).