Computer Networks 166 (2020) 106988

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Context information sharing for the Internet of Things: A survey

Everton de Matos[a,b,*], Ramão Tiago Tiburski[a,c], Carlos Roberto Moratelli[d], Sergio Johann Filho[a], Leonardo Albernaz Amaral[e], Gowri Ramachandran[f], Bhaskar Krishnamachari[f], Fabiano Hessel[a]

[a] Pontifical Catholic University of Rio Grande do Sul (PUCRS), Av. Ipiranga 6681, Porto Alegre, RS, Brazil
[b] Meridional Faculty (IMED), Polytechnic School, Rua Sen. Pinheiro 304, Passo Fundo, RS, Brazil
[c] Federal Institute of Santa Catarina (IFSC), Rodovia SC-480, s/n, Distrito Frederico Wastner, São Lourenço do Oeste, SC, Brazil
[d] Federal University of Santa Catarina (UFSC), Rua João Pessoa 2750, Blumenau, SC, Brazil
[e] FTEC Technology School, Av. Assis Brasil 7765, Porto Alegre, RS, Brazil
[f] University of Southern California, Los Angeles, CA 90089, United States

ARTICLE INFO

Article history:
Received 16 August 2019
Revised 18 October 2019
Accepted 8 November 2019
Available online 9 November 2019

Keywords:
Context sharing
Context-awareness
Internet of Things
Context platform

ABSTRACT

Internet of Things (IoT) technology is starting to make an impact in a wide array of applications, including smart cities and industrial environments. Such real-world applications combine computation, communication, sensing, and in some cases, actuation, to monitor and remotely control the environment. Data is at the core of such real-world IoT applications. Analysis, modeling, and reasoning of data are necessary to gain valuable insights. Application developers employ context-aware systems to translate the data into contextual information, which then allows the applications to act cognitively. Context sharing platforms offer a solution to distribute context information to those who may be interested in it, thus enabling context interoperability among different entities. This survey first examines the requirements for sharing context information. It then reviews the relevant literature for context sharing and classifies them based on their requirements and characteristics. Challenges and future directions are presented to encourage the development of context sharing platforms.

1. Introduction

The Internet of Things (IoT) consists of networked embedded devices equipped with a certain level of intelligence. The rapid advance in underlying technologies provides those heterogeneous devices (e.g., smartphones, vehicles, wearable sensors, actuators) with data processing and networking capabilities. Such devices form IoT environments and becomes IoT smart devices [1]. The IoT encompasses many environments, such as smart home, connected vehicles, smart grid, and smart cities. Gartner states that the amount of IoT devices deployed will reach 20.4 billion by 2020 [2]. From such predictions, it is clear that the IoT devices are expected to generate a large volume of data that will need proper interpretation, analysis, and processing [3].

Context-aware systems are deployed in IoT environments to sense the operational surroundings and to provide a suitable response to the user and the application [4]. These systems analyze the data produced by the IoT devices, giving a high-level sense (i.e., semantic meaning) and turning it into context information. The context information is used to define the status of an environment. The environment typically comprised of users, an application, a place, or a device [5], who produce the context information. The context information is mostly stored locally, not giving access to third-parties. In particular, even the same context gathering systems deployed in different environments rarely share the information [4,6]. However, the context sharing between deployments is one of the essential requirements for IoT since it allows systems to *understand* context information across heterogeneous environments and applications. By sharing the context information among different systems, the developers can reuse it in multiple applications, which is expected to increase the return on investment while increasing the utility [7]. Moreover, the context sharing feature is considered a challenge in the context-awareness research field [4].

The IoT can have environments with devices, users, and applications from different application domains. These environments tend to be heterogeneous, as each application can have different purposes. A smart city is an example where it is possible to have systems for different domains, such as smart traffic, public services, smart homes, and smart industries [13]. All these systems may produce different kinds of context information in different formats,

* Corresponding author.
E-mail address: everton.matos@edu.pucrs.br (E. de Matos).

https://doi.org/10.1016/j.comnet.2019.106988
1389-1286/

Table 1
Summary of Internet of Things and context-aware area surveys.

Survey title	Reference	Year	Summary
The Internet of Things: A survey	[8]	2010	Provides definitions of the Internet of Things computing paradigm, surveys its enabling technologies, and shows the benefits of the IoT implementation in different scenarios
A Survey of Middleware for Internet of Things	[9]	2011	Classifies and compares various IoT-middleware systems into different categories based on data interoperability, privacy and security issues, context-awareness, entities management, and scalability
Context Aware Computing for The Internet of Things: A Survey	[4]	2014	Provides an analysis of context-aware computing technologies and evaluates different context-aware systems
Context-Awareness for Mobile Sensing: A Survey and Future Directions	[10]	2016	Analyzes the existing context-aware mobile systems and how its implementation differs from the traditional context-aware systems
Context-Aware Computing, Learning, and Big Data in Internet of Things: A Survey	[11]	2018	Analyzes context-aware systems and how do they provide context information concerning different methods
The MOM of context-aware systems: A survey	[12]	2019	Compares several context-aware systems by its characteristics, and highlights their components

data types and specifications. The context sharing feature appears as a possible solution to provide interoperability in this scenario. A context sharing platform deployed in a smart city can understand a context produced by one domain and share it with applications in other domains [4,14]. It is also a context sharing platform's responsibility to turn this context understandable for the destination domain entity when necessary. For example, if an event is generating context information at the city (e.g., roadblock), thus changing the environment, that context must be sent to who may be interested in it (e.g., citizen, public services), regardless of the application domain. Therefore, actions can be triggered with that shared context information (e.g., change route).

The context sharing feature enables systems deployed in pervasive computational environments to have a common view of the context information to facilitate interoperability [6]. Thus, context sharing also helps in reducing the processing effort of those systems once they do not need to analyze and reason about the environment for the context information [14].

There are many surveys focusing on the Internet of Things and its applications and characteristics, among other topics. Table 1 summarizes some important surveys related to IoT in general and context-awareness area. Some of the existing surveys cover the context-aware systems, but there is no survey addressing the context sharing or context interoperability issue. Atzori et al. [8] state that interoperability is an issue in IoT environments. However, their paper neither discusses context nor context interoperability. Bandyopadhyay et al. [9] paper has the IoT-middleware and its features as the central point of discussion. They did not mention the provision of context interoperability for the analyzed IoT-middleware systems. Perera et al. [4] and Yürür et al. [10] define context sharing as a challenge in the context-awareness area. However, no further discussion on the state-of-the-art of context sharing or details of how the systems should provide this feature is made. Sezer et al. [11] state that the Web Ontology Language (OWL) can handle the context interoperability issue. However, no comparison was made regarding systems that provide context interoperability or context sharing feature. Pradeep et al. [12] did not present definitions on the context interoperability or context sharing process, either systems classification on such features. Our survey focuses on the context sharing feature, which is a challenge in context-awareness environments [4,10–12,14].

This paper focuses on context sharing platforms, which have different characteristics. Such platforms are being deployed in various application scenarios. This survey discusses the features and requirements of such platforms, named building blocks. It also compares the platforms through the building blocks, discusses how they meet the sharing requirements, and shows possible adaptations alongside novel technologies to enhance the overall quality of the context sharing platforms for IoT, thus enabling context interoperability. Also, to the best of our knowledge, the present survey is the first in the field of context sharing platforms for IoT.

The rest of the paper is structured as follows. Section 2 introduces the background concepts in context-awareness. Section 3 presents details of the context sharing feature. We introduce the building blocks of context sharing systems in Section 4 and present how they are related. Then, Section 5 extensively reviews the context sharing platforms, comparing them by the building blocks. Next, Section 6 discusses the challenges and novel technologies in sharing platforms. Finally, we present the conclusions in Section 7.

2. Context-Awareness and IoT

Context, sometimes referred as, context information, is commonly represented semantically [15]. It is used to define the status of an environmental entity (e.g., person, place, application, or computing device), thus characterizing its situation [5,16]. Context information is highly related to the information that is easily understandable by humans when reading it [4].

Abowd et al. have introduced a way to characterize the situation of the entities, that is used until these days by most of context information management solutions [4,5]. It is based on the "Five Ws" approach, which uses five questions: Who, Where, When, What, and Why. Those questions are made to build the context information. The question "Who" can characterize the identity of the entity. By asking "Where" it is possible to discover the location. The question "When" gives a notion of time. The "What" can characterize an activity. Finally, by asking "Why," it is possible to determine the motivation. In light of this, the information is expected to have at least one of the "Five Ws" within it to be considered a context.

Even with the well-known definitions of what is considered context information, there is no standard format and representation for it [4,15,16]. Different researchers have identified different ways to present context based on different approaches. Abowd et al. introduced one of the most popular ways to define the context (i.e., the "Five Ws" approach) [5]. They defined two types of context: primary and secondary. The primary context is identified

```
{
  "provider_id": 3579,
  "provider_type": "pacemaker_device",
  "provider_name": "pacemaker3579",
  "time" : [11, 31],
  "period" : "PM",
  "date" : [12, 19, 2018],
  "event" : "heart attack"
}
```

Fig. 1. Example of a primary context, generated by a pacemaker device, of a patient having a heart attack in JSON format.

```
{
  "provider_id": 2216758877,
  "provider_type": "smartphone",
  "provider_name": "phone01",
  "time" : [11, 31],
  "period" : "PM",
  "date" : [12, 19, 2018],
  "latitude": [33, 59, 14, 3],
  "longitude": [-118, 13, 32, 0],
  "city" : "Los Angeles",
  "zip" : 90089,
  "event" : "heart attack",
  "domain": "healthcare",
  "name" : "Paul Rodriguez"
}
```

Fig. 2. Example of a secondary context from a patient having a heart attack in JSON format. It represents a primary context enriched with a patient's phone information, thus creating a more complex context.

as location, time, identity, and activity. Further, the secondary context can be achieved by using the primary context [4]. For example, when considering that the primary context is both the GPS (Global Positioning System) coordinates of a user's device (e.g., smartphone) and the time of the day, it is possible to achieve the secondary context as being the events that may occur on that particular area, or the traffic status. Fig. 1 shows an example of this representation of a primary context, representing the data from a patient's pacemaker connected to the patient's smartphone. Fig. 2 shows an example of the secondary context, which is the enriched location information (e.g., city, zip code) that can be used to send an alert to the Emergency Medical Services (EMS).

The FIWARE introduced another example of a popular way of representing context information, the Orion Context Broker[1] project. It organizes the context information more straightforwardly by just defining one type of context, that encompasses both primary and secondary ones when compared with the Abowd et al. representation [5]. Fig. 3 shows an example of FIWARE - Orion Context Broker context representation.

Recently, Casadei et al. [17] stated that the context information is a fundamental piece of the IoT environments. The authors classify the IoT environment in three main classes of entities: IoT Entity, IoT Environment, and IoT Service. The IoT Entity is any subject that either produces or consumes IoT Services. The IoT Environment is the physical place where the IoT Entities are deployed. IoT Services are the cyber-physical services provided by the IoT Entities. The context is stated as the dependencies between those three classes of entities. It can express implicit or explicit information regarding them.

As shown in Figs. 1–3, the context information tends to be presented in an easily understandable form for the final user, for example, in a JSON (JavaScript Object Notation) or in an XML (Extensible Markup Language) format.

```
{
  "id": "3579",
  "location": {
      "metadata": {},
      "type": "Point",
      "coordinates": [ 34.020452, -118.288909 ]
  },
  "pacemaker": {
      "metadata": {},
      "type": "Event",
      "value": "heart atack"
  },
  "type": "Pacemaker"
}
```

Fig. 3. Example of a FIWARE - Orion Context Broker context from a patient having a heart attack in JSON format.

In this work, we define a formal way of representing context information, not considering the metadata format, but the content that it should have. Let's consider a set of Entities $E = \{e_1, e_2, \ldots, e_n\}$ and a set of Status $S_n = \{s_1, s_2, \ldots, s_m\}$. An entity e_n represents a person, place, application, or computing device, and has a set of Status S_n composed of information that characterizes the entity e_n. A common way of characterizing an entity is by using the "Five Ws" approach [5]. Thus, a status s_m should be represented by at least one of the "Five Ws" characterization about the entity e_n. Taking this into account, a context information is defined as ci_{nm}, where n is an entity id and m an information id about an entity e_n. A set of context information is denoted as $C = \{ci_{nm1}, ci_{nm2}, \ldots, ci_{nmk}\}$.

To provide context information, a system must follow some steps. Perera et al. defined Acquisition, Modelling, Reasoning, and Distribution as the steps for a system to provide context information, naming as context life-cycle [4]. The Acquisition refers to gather the raw data from a sensor, database, or from the environment. The Modelling process adjusts the data in a specific format to turn its input for the Reasoning step. There are many different techniques for Modelling already surveyed in existing literature (e.g., key-value pairs, ontology, markup scheme) [18,19]. The Reasoning process is the primary step in the context life-cycle. It transforms the information into a context, denoted as ci_{nm}, turning it understandable to the final users. The Reasoning, also called inferencing, may use different data enrichment techniques (e.g., business rules, ontology, probabilistic, data fusion, aggregation). Perera et al. detail the main Reasoning techniques and show in detail how each technique works. Distribution is a straightforward step [4]. It is responsible for spreading the context information. Usually, it has the option to distribute context by direct query or subscription.

A system can be considered context-aware when it uses the context obtained through the context life-cycle to provide useful services/information to the user [5,20]. In this way, it is indispensable to the IoT environments to have a context-aware system able to reason about the environment to provide such kind of services/information. Thus, context-awareness is considered a must-have feature to IoT systems [15].

IoT environments may have different processing layers, including Fog and Edge computing layers. Fog and Edge Computing are firmly related concepts, but they are not the same [21–23]. According to the OpenFog Reference Architecture [21], Fog computing extends Cloud computing into an intermediate layer close to IoT devices and enables data processing across domains, while Edge computing involves the control and management of a standalone endpoint device individually within the Fog domain, typically within a

[1] https://fiware-orion.readthedocs.io/en/master/.

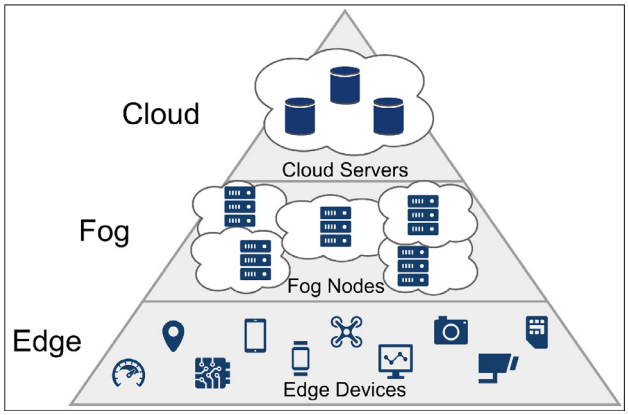

Fig. 4. Overview of different layers present in IoT environments.

close proximity of the device [24]. It is prevalent in context-aware IoT environments to produce the context information at the Edge layer and enrich them in the Fog layer. It may happen because the Fog layer may access data from other nodes, making it possible to fuse different information.

To avoid any possible misunderstanding about the concepts of Fog and Edge, we adopt the definitions from the OpenFog project and Morabito et al. for the contextualization of those concepts in this work [21,22]. Thus, we can define three entities for IoT environments: (i) edge devices, (ii) fog nodes, and (iii) cloud servers. Fig. 4 illustrates these entities divided into layers. The edge devices make part of the Edge layer; It is composed of different IoT devices. These devices usually have sensors attached to it responsible for sensing the environment and for data generation. The edge devices may also have the responsibility of data pre-processing and sometimes local decision making. They are characterized by having limited processing power and in some cases, limited energy supply (i.e., battery). Some examples of edge devices are IoT platforms (e.g., Raspberry Pi, Arduino), sensors, actuators, smartphones. The fog nodes make part of the Fog layer. They have the main function of processing data that could not be processed at the Edge layer, which may occur due to the limited resource capacity of the lower layer (i.e., Edge). Also, fog nodes may store edge devices data. It is usual for the fog nodes to be placed physically close to the edge devices (i.e., from the same room to the same city). Some examples of fog nodes are small servers and personal computers. Even a powerful edge device sometimes may be considered a fog node (e.g., last generation smartphones/tablets connected to an energy supply). In the Edge-Fog approach, the cloud servers, placed at the Cloud layer, work as information storage. They usually do not process data, working as a larger database. The cloud servers may store data (e.g., context information) from a few fog nodes and several edge devices at the same time.

The context information may vary depending on the producer. It can vary in format, size, representation. Most of the context-aware systems produce such kind of context information and use it only locally for decision making or spread it directly to the final user. However, there are some systems that could share context information with whom may be interested. This process is called context sharing and is one of the main challenges of the context-awareness area [4,5,14].

3. Context sharing

Before going into context sharing details, it is important to separate its definition from data interoperability, i.e., data sharing. Although having a similar concept of delivering common understanding information for two different entities, both data interoperability and context sharing differ in some aspects. There are some data interoperability platforms for IoT already studied and developed by the scientific community, such as FIESTA-IOT [25], and IoT-A [26]. Their main goal is to provide a way to make IoT device data interoperable between different applications and users. However, context sharing platforms may work with context, that can be considered a more complex information [4].

Context information sharing needs a more careful process than sharing regular data. This data may be sensitive, making it essential to care about its security. For example, if an attacker gets regular data from a communication channel, it can be tough to understand the meaning of such data without the right context. Differently, the context information represents a specific event, many times in a semantic manner. Thus, it is much more understandable for a possible attacker. Moreover, there are many different ways to provide context information [4,15]. In light of this, it is very common that heterogeneous context information providers act in different ways when generating context, varying in its format, length, data type, representation. Thus, these variations should be considered when providing context sharing feature. Therefore, context sharing platforms tend to have an enormous effort in providing context interoperability, many times by using different techniques (e.g., rules, ontology, decisions trees).

As IoT has many heterogeneous environments with different devices generating context information, it is essential to share context information between entities. More than helping in a common understanding of the context information, the context sharing feature may also help in reducing the effort of the entities. The receiving entity can get the context information without performing the reasoning process, which is considered the most demanding hardware operation in context life cycle [4].

The context sharing feature can be performed embedded in an IoT entity or by a third-party software. The software system that performs the context sharing feature can be called architecture, platform, tool or mechanism. It will be called by platform throughout this paper.

To make clear the organization of a context sharing platform, Fig. 5 shows how the context sharing feature can fit in an Internet of Things environment (e.g., smart cities). It shows an example scenario in which the context information generated in one domain (e.g., traffic) is shared with different domains (e.g., public services and citizens). The context can be produced with data from different devices (e.g., monitoring camera, light pole) and it should be understood by the destination domain and its devices (e.g., ambulances and traffic lights). In Fig. 5, the context sharing feature is provided by a context sharing platform. We use such platform view to represent a software system able to interconnect different domains by sharing its context information.

An event that can occur in a smart city generates context information that is automatically shared by the context sharing platform to whom may be interested in it. The sharing process includes heterogeneous entities. Thus, the received context information can be used in different kinds of processing. For example, the entities can process the received context, create a new context, and share it back with the platform.

A context sharing platform may vary in its characteristics (i.e., features). For example, it can have decentralized or centralized processing. Next section, explains in details the different features (i.e., the building blocks) that a platform must have to share context information.

4. Sharing building blocks

It is well-known that IoT environments have complex application scenarios. A context sharing platform must deal with these scenarios by implementing some specific functions. These func-

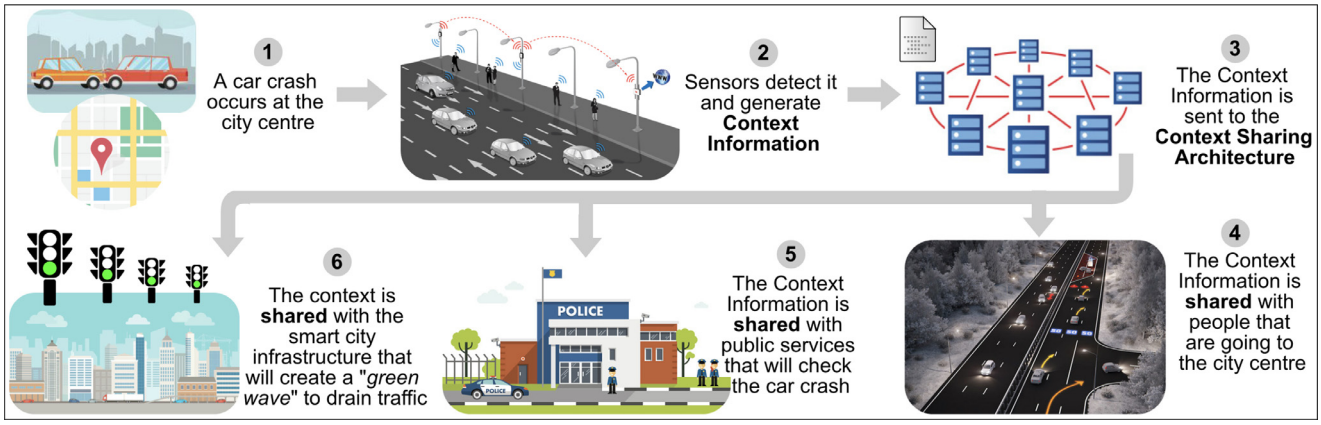

Fig. 5. Context sharing platform usability in a smart city application scenario.

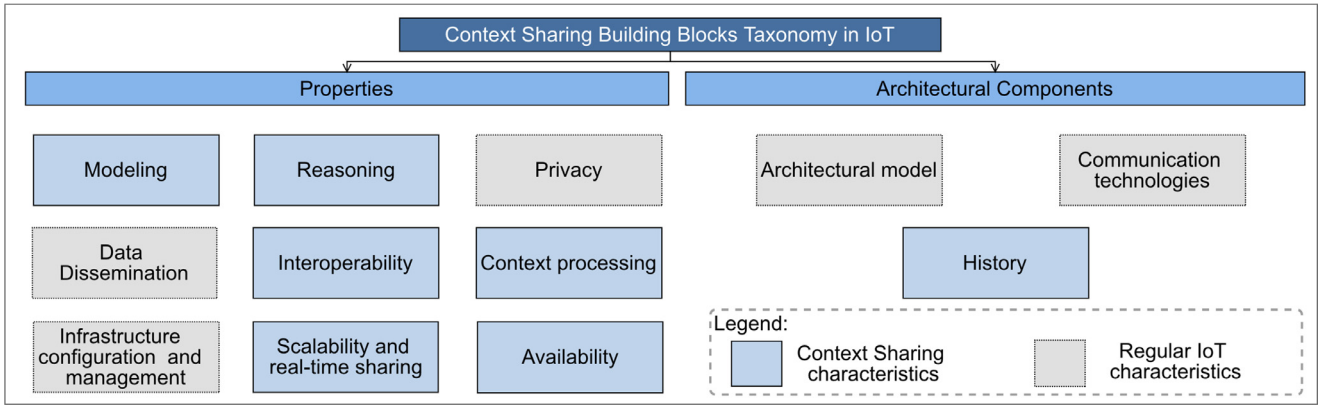

Fig. 6. The context sharing building blocks taxonomy.

tions are also called building blocks. In this survey, we have defined the context sharing building blocks taking into account some past research in the context-awareness and context sharing areas [4,27,28]. We used the following building blocks to compare the different context sharing platforms: Modeling (M), Reasoning (R), Data Dissemination (D), Privacy (P), Interoperability (I), Context Processing (CP), Infrastructure Configuration and Management (ICM), Scalability and Real-time sharing (SRT), Availability (Av), Communication technologies (C), History (Hi), and Architectural model (Ar).

The building blocks can categorized as: (i) *Properties*, and (ii) *Architectural Components*. The *Properties* refer to the ones mainly related to the software side of the context sharing platforms. The *Architectural Components* refers to the architectural decisions when deploying a context sharing platform, mainly related to the hardware side (e.g., communication technologies, storage space, processing layers). We present the building blocks and its divisions in a taxonomy view at Fig. 6. Moreover, as shown in Fig. 6, some building blocks are strictly related to the context sharing characteristics, which means the ones exclusively need in such processing. On the other hand, some building blocks are related to regular IoT characteristics, which are common in IoT systems that do not necessarily originate from the context sharing feature.

Fig. 7 shows all the different building blocks and how they fit and interact together in an Internet of Things Environment. It is common sense that most of the building blocks are a responsibility of the Context Source entity that will share the context information. However, it opens new possibilities when some building blocks are implemented by the Context Destination, that will receive the context information. For example, if the Context Destination has a Reasoning (R) function, it can produce new context information to perform a new task.

The following paragraphs present the context sharing building blocks that a context sharing platform must have. In this sense, this section defines these building blocks and the enabling technologies for each one, while Table 3 (see Section 5) makes a comparison with many systems architectures through the defined building blocks.

4.1. Properties

4.1.1. Modeling (M)

The modeling is the first step for context standardization. It is responsible for converting the context into a predefined format. The modeling process helps in a more straightforward interpretation of the context information. An efficient modeling process is essential for a context sharing platform. Researchers already surveyed the most popular techniques for modeling context information [4,29,30]. These surveys present the techniques and different ways of implementation for each one. To choose for a specific modeling technique will depend on the deployed site characteristics, once each technique may be suitable for a particular situation. A given system may employ one or more modeling technique. In this survey, we classify the works by the following modeling techniques: key-value modeling (Key), markup schemes (Mrk), text-based modeling (Tex), graphical modeling (Gra), object-oriented modeling (Oob), logic-based modeling (Lob), and ontology-based modeling (Onb). The symbol (✓) is used to denote that the work employs the modeling feature, but it does not make clear the specific technique.

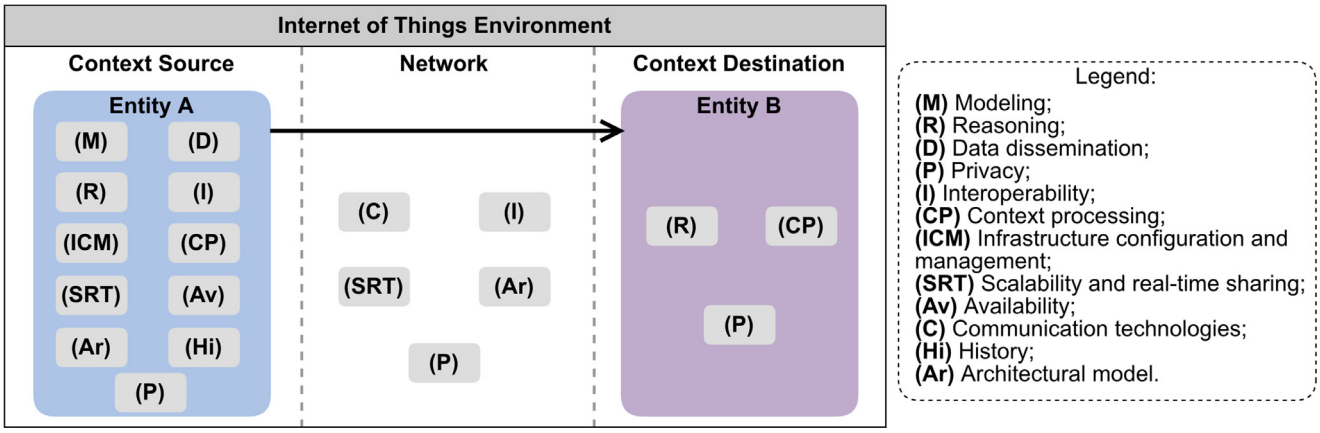

Fig. 7. The context sharing building blocks and its interaction.

4.1.2. Reasoning (R)

It is defined as the process to obtain high-level information from less enriched data, or even raw data. The reasoning process uses the available context to produce a more useful one. The outcome of the reasoning process could be semantic information, being easily understood by final users. Moreover, the reasoning also is defined as the inference process [31]. In the scope of context sharing, the reasoning is used to discover/infer the destination of the context information for sharing. For example, this feature may be used to discover that a context of a patient having a heart attack must be sent to the Emergency Medical Services (e.g., ambulances) of the city. Moreover, the reasoning can be also performed by the entity that will receive the context in order to do a new processing (e.g., decision making, inferencing). The most popular reasoning techniques are surveyed in [4]. In this survey, we classify the works by the following reasoning techniques: supervised learning (Sul), unsupervised learning (Unl), reinforcement learning (Rel), rules (Rul), fuzzy logic (Fuz), ontology-based (Onb), and probabilistic reasoning (Pro). The symbol (✓) is used to denote that the work employs the reasoning feature, but it does not make clear the specific technique.

4.1.3. Data dissemination (D)

The context information is shared to who is interested in it. Data dissemination is a fairly straightforward task. Each system has a policy to disseminate the context information. There are only two ways of data dissemination: static (Sta), and dynamic (Dyn). In the static approach, there is a predefined list to whom the context must be sent depending on the specific situation. On the other hand, the dynamic approach needs a specific reasoning function to define the exact destination of the context information. For example, in the static approach, there will be a predefined list of the city ambulances, while in the dynamic, a reasoning method will find the nearest available ambulance. For both approaches, the dissemination may occur by groups, roles, or individually.

4.1.4. Privacy (P)

The context information includes private data in most situations such as location, activities, and preferences. Thus, privacy is an important role. Different approaches can be used to ensure privacy, such as authorization, access control policies, anonymization, cryptography [32]. Concerning privacy protection, it is necessary to specify what information may be disclosed, providing means to trace and destroy the information, if necessary. The symbol (✓) is used to denote works that employ techniques to ensure privacy.

4.1.5. Interoperability (I)

Differently from the most context systems that comprise only an application-specific system (i.e., a vertical domain), shared systems tend to be more heterogeneous. In the context sharing platforms, the interoperability needs to appear in different ways, such as in the context production, format, and interpretation. The interoperability requirement is related to the capability of the platform to manage context information in different aspects, such as format, source, length, and representation. Currently, there is no standard context notation format. Although there is a wide range of applicable context information, it is very difficult to make a one size fits all context format. The complete interoperability only will be possible by the combination of many other factors, such as modeling, data dissemination, communication services, among others. In this survey we categorize the works in two groups: full interoperability (FuI), and partial interoperability (PaI). The full interoperability works try to address interoperability in different ways (e.g., data format and communication technologies) and between heterogeneous entities. Partial interoperability works usually provide interoperability only in a predefined application domain or between entities that may have similar characteristics to define the context.

4.1.6. Context processing (CP)

Considering the development of the IoT and with the increasing number of devices being connected to the Internet, it will not be realistic for those requesting context information to enter an IP (Internet Protocol) address to a specific device. In this sense, the context sharing platform must have the smart capacity to obtain, produce, and share context information from a highier-level request. This feature is similar to the device discovery service already present in IoT environments [33]. In this survey, we classify the works by the following context processing techniques: searching (S), filtering (F), and aggregation (A). The symbol (✓) is used to denote that the work employs the context processing feature, but it does not make clear the specific technique.

4.1.7. Infrastructure configuration and management (ICM)

In an IoT environment, many different devices can connect to the network. It is essential to the context sharing platforms to provide ways for the new devices/applications to connect with the sharing infrastructure. The infrastructure configuration and management feature must facilitate the connection to the sharing platform to different kinds of devices and applications. Besides, to allow the sharing of information, context sharing platforms must be responsible for managing the connected ones in order to know a possible source and destination for context information. The symbol (✓) is used to denote that the work employs techniques to provide the infrastructure configuration and management feature.

4.1.8. Scalability and real-time sharing (SRT)

In IoT environments, with many requests happening at the same time, enormous quantities of data/context information need proper processing with acceptable execution time. Moreover, this also reverberates to massive communication efforts. Taking into account this concept, the context sharing process must minimize the processing and the communication overhead in the sharing platforms. The symbol (✓) is used to denote that scalability and real-time sharing functionality is employed by the analyzed works in some perspective.

4.1.9. Availability (Av)

As the context information can be produced every time that the IoT devices generate new data, the context sharing process may happen anytime in such dynamic IoT environments. Context sharing platforms must be always available for a possible sharing. In this sense, sharing platforms may run the sharing process automatically (Aut), that is, the sharing should occur without the need for setting up the system, and it will happen as soon as new context information is produced and defined as shareable. On the other hand, the sharing process may need to be triggered (Tri), that is, it will require some setup or modification in the system to start the sharing process.

4.2. Architectural components

4.2.1. Communication technologies (C)

The context sharing platform is an entity of the IoT environment. In this sense, it must be able to communicate with many other entities. This communication may happen to gather specific data, or, in most cases, to share the context information. This topic is not directly related to the context sharing process, but the communication services feature refers to which network technologies the system supports, that is essential for the communication between the entities. The systems may offer local communication (Loc) (e.g., Bluetooth, Wi-Fi, NFC), external communication (Ext) (e.g., 3G, 4G, LTE), or both (LE). Moreover, network layer protocol aspects and messaging paradigms are also crucial for communication technologies in such systems. These kinds of communication technologies help to provide interoperability by their different ways of implementation. Some examples are Web services (e.g., REST, SOAP), socket and WebSockets. These kinds of technologies are well-known device-agnostic standards for communication [34,35]. Thus, the work is considered attending the network layer communication criteria (NeC) if it mentions the use of such kinds of technologies.

4.2.2. History (Hi)

The shared context information may be stored in the context sharing platforms. It can be useful for probabilistic reasoning or to access the last record of some information. Also, the history can help in the real-time processing, when it is updated in storage. Context information may be stored locally or in the cloud. The symbol (✓) is used to denote that the work employs techniques to provide the history feature.

4.2.3. Architectural model (Ar)

The approach for the architectural perspective of context sharing platforms may vary depending on the application domain characteristics. There is a wide range of applicable perspective of systems architecture. However, considering the IoT environments, some architectures are more suitable than others. There are three main architectures for sharing platforms appropriate for IoT environments: cloud-based (Clo), centralized-edge (Cen), and decentralized-edge (Dec). In the cloud-based, the most processor demanding computation is executed in the cloud. It has a significant dependency on the network. The centralized-edge brings the computation near to the IoT devices, but still has a central point of computation, even to store information. It works based on groups. Finally, in the decentralized-edge, the computation is divided into devices. It may lack in some features, once the device may be limited in some ways, like processing power and storage capability. On the other hand, some systems do not follow a specific architectural perspective and can adapt (Adp) itself depending on the environment.

5. Context sharing platforms

There are already available systems/platforms that support data processing, data enrichment, and data sharing. Moreover, some systems/platforms perform the sharing of more complex information than raw data, thus having the context sharing feature (see Section 3). In this Section, we analyze different works focused on sharing their contexts in different ways. We select the analyzed works by its capability to provide or facilitate the context sharing in some way. Moreover, we based our selection on the works that could address an IoT environment application. Some works talk directly about IoT, while others talk about pervasive/ubiquitous computing application scenarios.

We analyze the systems based on the requirements presented previously (see Section 4), which are essential to provide a Context Sharing Architecture. Table 2 summarizes all the aforementioned sharing building blocks and its abbreviations, helping in understanding the Table 3, which presents all the analyzed works.

To provide a fair comparison between the works, they were categorized and grouped in two categories: (i) Model, and (ii) Middleware. The Models are lightweight approaches that has the goal of facilitate and assist in the context information sharing process (e.g., ontologies, Bluetooth-based systems, models for formalizing context information). The Models can act in classifying context information to model it, making easy the context interoperability. Also, some Models may serve as a broker, in which context can be posted and queried. Many times, Models may react to a determined event (e.g., generated context information) or situation (e.g., proximity to an entity) triggering the sharing process. On the other hand, a Middleware is a platform that provides diverse kinds of services, and has different methods to manage data [32]. The Middleware matches with the definition of a PaaS (Platform as a Service) [36], offering a platform that users can deploy different services, including the context sharing functionality.

The Models can be considered a more straightforward category for context sharing, offering only that feature or facilitating it in some way. The Middleware are more robust platforms that usually provide context sharing as an option and different features (e.g., data storage, device interconnection) as main goal. Both categories, Model and Middleware, are suitable for deployment in IoT environments. Even though, by its characteristics, Models are more suitable for lightweight environments and Middleware for those with more resource (e.g., processing power, memory). However, it is a common characteristic of both Models and Middleware to provide a stream processing of data (i.e., context information). The stream happens because the context is always changing (especially in IoT), and it is essential to share the different context when the events occur to keep the environment updated. Some approaches may need a previous connection and others stream context automatically. Moreover, some works also provide a batch processing.

We selected different works to compare. We tried to analyze works that differ in scale. We analyze works from small-scale to large-scale projects. Moreover, we tried to analyze both well established works as well as new projects available in the past few years. We also look to compare works in different categories (i.e.

Table 2
Summary of context sharing building blocks.

Context sharing building blocks	Possibilities
Modeling (M)	Key-value modeling (Key), markup schemes (Mrk), text-based modeling (Tex), graphical modeling (Gra), object-oriented modeling (Oob), logic-based modeling (Lob), ontology-based modeling (Onb). (✓) it is employed, but no specific technique is detailed
Reasoning (R)	Supervised learning (Sul), unsupervised learning (Unl), reinforcement learning (Rel), rules (Rul), fuzzy logic (Fuz), ontology-based (Onb), probabilistic reasoning (Pro). (✓) it is employed, but no specific technique is detailed
Data dissemination (D)	Static (Sta), dynamic (Dyn)
Privacy (P)	(✓) it employs privacy techniques
Interoperability (I)	Full interoperability (Ful), partial interoperability (Pal)
Context processing (CP)	Searching (S), filtering (F), and aggregation (A). (✓) it is employed, but no specific technique is detailed
Infrastructure configuration and management (ICM)	(✓) it employs infrastructure configuration and management techniques
Scalability and real-time sharing (SRT)	(✓) it employs scalability and real-time sharing techniques
Availability (Av)	Automatically (Aut), Triggered (Tri)
Communication technologies (C)	Local communication (Loc), external communication (Ext), local and external (LE), network layer communication (NeC)
History (Hi)	(✓) it employs history techniques
Architectural model (Ar)	Cloud-based (Clo), centralized-edge (Cen), decentralized-edge (Dec), adapt (Adp)

Table 3
Evaluation of surveyed works.

Sharing	Ref	Year	Category	Properties									Architectural components		
Platforms				(M)	(R)	(D)	(P)	(I)	(CP)	(ICM)	(SRT)	(Av)	(C)	(Hi)	(Ar)
CONON	[6]	2004	Model	Onb	Rul, Onb	Dyn	–	Ful	✓	✓	–	Aut	–	–	Adp
ACC	[37]	2004	Model	Oob, Lob	Onb	Sta	✓	Pal	✓	✓	–	Tri	–	–	Adp
DJess	[38]	2005	Model	Tex	Rul	Sta	✓	Pal	–	✓	✓	Tri	Ext, NeC	✓	Adp
CoSM	[39]	2009	Model	Onb	Onb	Dyn	–	Ful	–	✓	–	Aut	–	–	Clo
M3	[40]	2014	Model	Onb	Rul, Onb	Dyn	–	Pal	S	✓	–	Tri	–	–	Adp
CS-Sharing	[41]	2016	Model	Key	Rul	Dyn	–	Pal	A	✓	✓	Aut	Loc, NeC	✓	Dec
Bluewave	[42]	2016	Model	Tex	Rul	Sta	✓	Pal	–	✓	✓	Aut	LE, NeC	✓	Cen
PSW	[43]	2017	Model	Onb	Rul, Onb	Dyn	–	Ful	–	✓	✓	Aut	LE, NeC	–	Adp
LiO-IoT	[44]	2018	Model	Onb	Onb	Dyn	–	Ful	–	✓	✓	Tri	–	–	Adp
SCS	[45]	2019	Model	–	Rel	Dyn	–	Ful	–	✓	✓	Aut	NeC	–	Dec
SE-TSDB	[46]	2019	Model	Onb	Rul, Onb	Dyn	✓	Ful	F,A	✓	–	Tri	–	✓	Adp
FRASCS	[47]	2008	Middleware	Key, Mrk	Rul	Dyn	–	Ful	A	✓	–	Aut	–	✓	Clo
SharedLife	[48]	2009	Middleware	Onb	Rul, Onb	Dyn	✓	Ful	✓	✓	–	Aut	–	✓	Adp
ConCon	[49]	2014	Middleware	Key	Onb	Sta	–	Pal	F	✓	✓	Tri	LE, NeC	–	Cen
Grapevine	[50]	2015	Middleware	Key, Tex	Pro	Dyn	–	Ful	F	✓	✓	Aut	–	–	Cen
HEAL	[51]	2015	Middleware	Key	Pro	Dyn	–	Pal	A	✓	–	Tri	Ext, NeC	✓	Clo
Magpie	[52]	2015	Middleware	✓	✓	Dyn	✓	Ful	✓	✓	–	Tri	LE, NeC	–	Dec
OIoT	[53]	2015	Middleware	✓	Rul	Dyn	–	Ful	A	✓	–	Aut	LE, NeC	–	Adp
RCOS	[54]	2016	Middleware	Onb	Onb	Dyn	–	Ful	✓	✓	✓	Aut	Ext, NeC	✓	Cen
Chitchat	[55]	2016	Middleware	Key, Tex	Pro	Dyn	–	Ful	–	✓	✓	Aut	LE, NeC	✓	Adp
C2IoT	[56]	2017	Middleware	✓	✓	Dyn	–	Ful	F,A	✓	–	Aut	NeC	✓	Clo
BigClue	[57]	2018	Middleware	✓	Pro	Dyn	–	Ful	–	✓	✓	Aut	Ext, NeC	✓	Cen
CoaaS	[58]	2018	Middleware	✓	Rul, Pro	Dyn	✓	Ful	A	✓	✓	Tri	NeC	✓	Clo
SCENTS	[59]	2019	Middleware	✓	Rul	Dyn	–	Ful	–	✓	–	Aut	LE, NeC	✓	Cen

Model, and Middleware) which have different ways to provide context sharing. We detail the analyzed works in Sections 5.1 and 5.2. Section 5.3 discusses how the works provide context sharing. Moreover, we discuss the International Efforts in context sharing in Section 5.4. By covering the vast heterogeneity of context sharing works, we provide a large vision of the area.

In Table 3, we make use of the dash (—) symbol when the work does not provide the specific feature, or it is not mentioned in the available publications. Next, we present the definitions of analyzed platforms.

5.1. Context sharing models

5.1.1. CONON

It is an OWL (Web Ontology Language) ontology to model context information in ubiquitous environments. Besides the modeling, as being an ontology, CONON also facilitates the logic-based inferencing process [6]. CONON provides a flexible architecture, once it has a general ontology for modeling broader context concepts and also enables the coupling with other ontologies in a hierarchical manner. The coupled ontologies could be from any specific domain. To CONON, the context information of different domains shares common definitions and concepts. Thus, it uses a generic ontology to model these general concepts, while domain-specific ontologies deal with fine-grain context information.

5.1.2. ACC

Agent Coordination Context (ACC) works to model the application environment context and the interaction among agents and the environment in Multi-Agent Systems (MASs) [37]. ACC provides tools to organize the access to the shared information (i.e., context). It organizes the shared information into “spaces” and regu-

lates the access control in a role-based policy. Thus, ACC has two main functions: (i) it works to model the application environment characteristics to enable context sharing, and consecutively (ii) by creating sharing "spaces" for interaction considering the environment characteristics. In this sense, the ACC can be suitably understood as an infrastructural abstraction.

5.1.3. DJess

It is a Java package that works in a distributed way and provides an infrastructure for context sharing between entities using it [38]. Only middleware that was implemented using Jess [60] for the inferencing/reasoning process can run DJess. Different nodes using DJess can communicate to have a common understanding of the context. DJess creates an abstraction of a single view of the application environment for the distributed systems running it. Thus, even if the systems were placed in different sites, they will share context as if it were in the same local domain.

5.1.4. CoSM

The Context Sharing Message Broker (CoSM) can model the context in a standard way to enable the context sharing process [39]. CoSM helps in facilitating the common understanding of context between different applications even in dynamic environments. Applications can agree in defining a context model to share context information. CoSM's context model intermediates the context sharing processing by providing a common interface for a mutual understanding of different context information. The applications communicate with CoSM to send context information, then CoSM manages it and delivers the context to interested entities/applications based on the defined context model. CoSM acts as a plug-in to the application, making it an independent tool and not modifying the applications that use it.

5.1.5. M3

The Machine-to-Machine Measurement (M3) is an approach that includes an ontology, a hub, and semantic domain rules, that can combine, and reason about IoT devices data that follow the M2M (Machine-to-Machine) standard [40]. The authors have proposed a semantic-based M2M architecture that is used as basis for M3 [61]. The M3 ontology main goal is to classify and unify the heterogeneous data sensed by devices running at the M2M standard. It is an extension of the Semantic Sensor Network ontology (SSN)[2], thus making easy the reuse of common concepts as appears on the M2M standard. The authors claim that the M3 ontology is able to describe more than 30 sensor types (e.g., thermometer, accelerometer) from different domains (e.g., agriculture, health).

5.1.6. CS-Sharing

It is a compressive sensing (CS) based scheme technique to provide context sharing in a decentralized way for vehicular networks. CS techniques enable the search for context information with basic queries [41]. The vehicles monitor the road conditions to acquire context and then share it using CS-Sharing. The vehicles store context locally. To avoid possible network overheads, CS-Sharing offers aggregation operations in the stored vehicle context before the sharing process. Thus, each vehicle can get context information about the road condition from a focused aggregated message that is broadcast by vehicles.

5.1.7. Bluewave

It is defined as a Bluetooth-based technique that makes possible nearby mobile devices to share their context [42]. In Bluewave, the context information first needs to be uploaded to a server for enabling the context sharing, characterizing it as a centralized approach for storing but an edge in processing. For an accurate context sharing process, it also needs to set a sharing URL for each device and broadcast it, along with a temporary credential, to the devices in a 30 feet radius. Bluewave has two main components: (i) client, and (ii) context broker. The client component is embedded into mobile devices and has the functions of upload the context information to a server, and discover new devices by proximity. The context broker component runs in a web server and has the responsibilities to store context and to share it with the authorized clients (i.e., nearby mobile devices).

5.1.8. PSW

The Physical Semantic Web (PSW) is a framework that makes use of Semantic Web technologies to knowledge discovery and sharing in IoT scenarios [43]. PSW has four components: (i) machine-understandable standard languages, (ii) objects exposing semantic annotations, (iii) knowledge discovery and sharing, and (iv) semantic matchmaking. The machine-understandable standard languages main goal is to provide a common language for a uniform communication. It uses an ontology for this goal. The objects exposing semantic annotations component is responsible for exposing the context information for sharing, and also to update this information when needed. The knowledge discovery and sharing components are responsible for discovering the neighbor devices with corresponding semantic annotations as the request. Finally, the semantic matchmaking component main goal is to make a rigorous semantic matching to rank the resources with the request by relevance.

5.1.9. LiO-IoT

The Light-weight Ontology (LiO-IoT) tackles a challenge in IoT ontologies spectrum by considering sensors, actuators, and RFID as IoT concepts [44]. It uses concepts from both SSN and IoT-Lite[3] ontologies to help in the provision of semantic interoperability. The authors have evaluated the LiO-IoT ontology through experiments to verify the round trip time of a query. They have compared LiO-IoT with both IoT-Lite and SSN. The results have shown that it has a similar response time when compared with IoT-Lite and a better performance when compared with SSN. However, it is valid to mention that SSN is a massive ontology that covers different IoT sensors [62].

5.1.10. SCS

The Smart Context Sharing (SCS) is an algorithm designed for facilitate context sharing among Fog nodes [45]. It focuses on sharing context information of different Fog nodes deployed on different IoT domains (e.g., smart agriculture, smart health, smart traffic). It considers a scenario of connected Fog nodes, in which one of the nodes may broadcast a message looking for a specific context. The Fog node with the wanted context information will respond to the broadcast message. SCS also has a load balancing algorithm that uses reinforcement learning techniques to predict the suitable node to migrate context information when sharing.

5.1.11. SE-TSDB

Semantic-Enhanced Time Series Databases (SE-TSDB) is a tool suite that helps in the data management for IoT [46]. It can work either on the Cloud, Fog, or Edge architectural approach. The authors have developed the DS-Ontology as the main part of the SE-TSDB tool. The DS-Ontology works as a semantic model for the specification of data streams from IoT devices. In light of this,

[2] https://www.w3.org/TR/vocab-ssn/.

[3] https://www.w3.org/Submission/2015/SUBM-iot-lite-20151126/.

the SE-TSDB is the DS-Ontology applied to Time Series Databases (TSDB), as the traditional TSDBs process data streams, but they do not offer sufficient semantic processing as needed in heterogeneous IoT environments. On SE-TSDB, the similarity matching, and reasoning processing happens with the DS-Ontology alongside with pre-defined domain-specific rules.

5.2. Context sharing middleware

5.2.1. FRASCS

The Framework Supporting Context Sharing (FRASCS) [47] enables entities to store and/or receive context information. The entities communicate directly with the FRASCS infrastructure. FRASCS has a Context Pool Manager (CPM) model responsible for receiving sensors connection and their raw data. It can deal with both physical (e.g., physical devices, hardware) and virtual (e.g., events from a software system) sensors. FRASCS also has the functionality of claim for context information directly from the context-aware applications when an entity requires it.

5.2.2. SharedLife

It is defined as a framework to share information (i.e., context) of the users between different applications and/or other users [48]. SharedLife acquires the context from various sources related to a user and stores it in a set of knowledge about the specific user. SharedLife also allows the searching for context information about a particular user, taking care of the privacy defined for each user/information. SharedLife shares the data itself and also a meta-information regarding the data, such as information about potential partners for future sharing. SharedLife works in an event-based manner to describe the sharing interactions and the relationships between entities and the environment. The events can be stored to serve as a possible recommendation for future context sharing by the entities connected to SharedLife.

5.2.3. ConCon

It is a middleware system that offers the context-aware feature in its architecture. ConCon [49] works based on the publish/subscribe pattern, enabling different entities to subscribe for specific context information. The entities connected to ConCon can act by producing and/or consuming the data (i.e., context). It can be embedded in everyday devices, such as smartphones. ConCon defines structures of similarity related to the context. Thus, the context information is matched semantically, avoiding possible data duplication. ConCon has two policies to manage context information: (i) time-sensitive, and (ii) quality measure. In the time-sensitive policy (i), the context to be shared receives a prefixed lifetime. Thus, the old context may lose its relevance. In the quality measure policy (ii), the context information is used to improve the quality of entities interactions, one time that a high-quality provider produces more relevant context (i.e., that is widely shared).

5.2.4. Grapevine

It is a framework that works for context sharing in specific networks (i.e., connected peer devices) [50]. Grapevine is designed to work with pervasive devices. It forms groups of devices to perform context sharing. The situation of the entities (i.e., context) is the primary factor for the creation of the sharing groups. Grapevine uses conditions that define whether the entity context sharing occurs and how widely (i.e., for which groups) the context will be shared. Grapevine works to reduce the communication overhead in the context sharing process. As it works with pervasive devices that may have resource-constrained restrictions, a lightweight data structure models the context information.

5.2.5. HEAL

The Healthcare Event Aggregation Lab (HEAL) is a middleware platform focused in smart healthcare environments [51]. It works in a cloud-based approach and allows different entities to connect to it by using REST web service and SPARQL endpoints. The sensors can connect to HEAL to provide context information. On the other side, smart healthcare applications will take benefit of the sensors data. HEAL detects similarity between sensors data, thus providing a filtered data to the applications. The context sharing occurs when HEAL acts as providing interoperability different platforms. It enables the provision of services based on context to heterogeneous applications as third-party systems and developers tools.

5.2.6. Magpie

It is an approach in which context sharing is provided by opportunistically connections between entities [52]. Magpie works in pervasive computing environments, sharing context information of mobile and heterogeneous devices. Magpie connects devices through an opportunistic mobile network. It has policies to concern about the privacy-preserving in devices connections. Magpie allows the entities connected to it for sharing several types of context with different applications/users. The event of sharing context can also generate a log for future trust agreement between the entities. Magpie provides tools to enable an “useful sharing”, that avoids the sharing of all the context information produced by the entity, making that sharing process to occur with only the useful context information. Both Magpie and the entities that send or receive context information can define a context as useful.

5.2.7. OIoT

Opportunistic IoT Platform (OIoT) is a software infrastructure which helps developers in providing data (i.e., context) sharing by the creation of opportunistic IoT devices groups [53]. OIoT adopts a form of mobile ad hoc network concept defined as Opportunistic Mobile Social Networks. It exploits the characteristics of human social interactions (e.g., daily activities, mobility patterns, and interests) to route messages and to share data. In such networks, the communication between mobile devices happens by dynamic (i.e., on-the-fly) social networks.

5.2.8. RCOS

Real Time Context Sharing (RCOS) is a publish/subscribe system that provides context-awareness features [54]. RCOS provides subscribe services to the entities connected to it. RCOS makes possible the creation of a subscription about a specific context. Thus, RCOS notifies the subscribed entities when it has new information posted. The devices can connect to RCOS through a REST interface to send their context. RCOS has functions to enable a contextual semantic matching based on the context information generated by the devices. One example of the semantic match is to relate two contexts by proximity based on the GPS coordinates. RCOS also acts like a plugin, being connected to preexisting publish/subscribe systems. Thus, the preexisting system can take benefit of the context-aware features and maintain a context history.

5.2.9. Chitchat

It is considered a pool of context information. Chitchat takes advantage of the devices communication by the network and inferencing capabilities to provide different views and access to the context information [55]. The entities (e.g., devices, applications) can connect to Chitchat through an Application Programming Interface (API) to specify its filters (i.e., when to receive context). Chitchat introduces probabilistic data structures based on a Bloomier filter [63]. Chitchat introduces two Bloomier filter based structures to further reduce a context information size and add the ability to update the structure on-the-fly. The first one claims to

reduce the size of the structure without impacting the false positive rate. The second one claims to guarantee a zero false positive rate under certain conditions and adds the ability to update context values in an already constructed context information.

5.2.10. C2IoT

It is a cloud-based framework for providing context-aware services on Internet of Things environments. The main application scenario of C2IoT are smart cities [56]. It provides a three layer infrastructure for such kind of context-aware services provision: SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Infrastructure as a Service). The authors claim that with those three layers, it is possible to reach the necessary flexibility and effectiveness that smart city scenarios require. Thus, dealing with the common challenges of those environments, including context information interoperability. On C2IoT, each layer has a specific function: SaaS - Result Visualization, PaaS - Data Management, and IaaS - Data Collection.

5.2.11. BigClue

BigClue is a data processing platform that integrates existing frameworks on its architecture with the primary goal of providing a cross-domain data layer for IoT environments [57]. It has five main layers on its architecture: Processing, Messaging, Storage, Service registry, and Visualization. For the implementation of those layers, BigClue uses the following techniques: Apache Spark for the Processing layer, Apache Kafka for Messaging, Apache Hbase for Storage, and HashiCorp Consul for Service registry. They also use Hadoop Distributed File System for the underlying file system. Finally, BigClue has RESTful exposed APIs for communication, thus helping on the interoperability.

5.2.12. CoaaS

Context-as-a-Service (CoaaS) is a platform able to exchange context information in IoT environments [58]. It shares context information from different context providers to context consumers. However, it does not have efforts in providing the context in a semantically interoperable way. It has four main components: Security and Communication Manager, Context Query Engine, Context Storage Management System, and Context Reasoning Engine. The Security and Communication Manager handles the incoming context and distributes it to the destination in a private way. The Context Query Engine component is responsible to manage the queries for context. The Context Storage Management System maintains a cache with past context information to provide a better response time. Finally, the Context Reasoning Engine component infers new information (i.e., context) from raw data.

5.2.13. SCENTS

Sensing Collaboratively in Everyday Networks (SCENTS) is a framework that supports context sharing by proximity for heterogeneous IoT devices [59]. One of the SCENTS' motivation is the premise that nearby devices tend to have similar values for context information, thus being interested in the nearby context by its geolocation. On an architecture view, SCENTS framework sits between the hardware layer and the application layer. It has two main components: Neighborhood Agent and the Collaboration Agent. The main function of the Neighborhood Agent is to continuously detects whenever a node (i.e., neighborhood device) is arriving or leaving the network. The Collaboration Agent manages the queries looking for context, process it, and delivers the right data (i.e., context information) to the applications.

5.3. Summary of sharing platforms

All the analyzed platforms have a common goal, which is to share context information between entities. However, each platform has its own peculiarities. They may differ in various aspects and even provide shared context information as well. Table 3 sums up all the analyzed platforms and shows the differences between them regarding the context sharing building blocks defined in Section 4. Next, we present how the analyzed platforms address the building blocks described previously. For each feature, we focused on platforms that discuss it in detail in the literature.

As presented in Table 3, the ontology-based (Onb) appears as the most used technique for the *modeling (M)* feature. It is considered a trend on the IoT to use ontologies. It can increase the interoperability feature of the platform [4]. Both CONON and RCOS use various ontologies for the semantically represent the context information. The use of different ontologies makes easier the integration with other semantic applications, as well with web applications. M3 and LiO-IoT extract some definitions from traditional ontologies, as SSN, on its approaches, helping on the interoperability with already deployed devices/entities. However, in some application scenarios, context systems may need a more complex approach. In this sense, FRASCS stands out by combining two techniques. FRASCS uses the key-value technique to model the information detected by the sensors (i.e., raw data), which can be considered low-level context information. On the other hand, it uses the markup scheme model (Mrk) technique to model the information from the context-aware applications connected to it. The markup scheme model technique is used in this case as it offers a more flexible structure than key-value pairs. Another technique used by many platforms is the key-value modeling (Key). It is a technique of simple implementation and gives a unique key for each context information. Key-value is also a lightweight approach for context information modeling.

As in the modeling of context information, the ontologies appears as one of the most important techniques for the *reasoning (R)* feature. The reasoning by ontologies (Onb) is facilitated when the platform also models context by ontologies. One example of work that uses ontologies for both modeling and reasoning is CoSM. It makes possible that the reasoning agents can access the modeled context to understand the context of the environment. On the other hand, ontologies may be considered heavy for some application scenarios since IoT environments could be restricted in processing capabilities. In this sense, SharedLife proposes a hybrid approach by using ontologies together with rules (Rul), which is a lightweight technique. SharedLife allows the definition of rules for automatic sharing of predefined context. It also uses ontologies to the creation of a unified user profile representation, making possible that different user profiles to being classified by similar characteristics. DJess uses a Java-based rule engine in the reasoning process. It makes possible the creation of declarative rules for reasoning in Java-based applications. Rules are the most popular reasoning technique of the analyzed platforms because of their suitability for resource-constrained environments. In some platforms, it may appear as Event-Condition-Action (ECA) rules.

The *data dissemination (D)* process is fundamental for the delivery of context information to whom needs it. The majority of the analyzed platforms makes the data dissemination in a dynamic (Dyn) way. It is the most suitable way for sharing context information since it enables sharing accordingly to the characteristics of the environment. For example, an offline or not apt entity will not receive context. CS-Sharing, OIoT, and SCENTS use the concept of *Opportunistic Meeting* to share context information. It defines that entities can share context information when they are physically near each other, in a dynamic way.

Just a few of the analyzed platforms provide the *Privacy (P)* feature. Although crucial in IoT environments, most of the platforms do not mention privacy efforts on their architectures. However, ACC provides access control to environment resources while DJess provides privacy-preserving options in the registering process of

the entities connected to it. Bluewave concerns about the privacy at the communication protocol level, which can reach a certain level of confidentiality, and Magpie provides privacy-preserving options to devices related to the provided context information so that the context sharing does not breach user privacy. In SharedLife, the context information repository is separated into different levels with access control. SE-TSDB makes use of the Time Series Databases (TSDB) that can include policies to secure the stored information.

As we can see in Table 3, most of the analyzed platforms with context sharing feature address *interoperability (I)* in a full (FuI) way. It is an expected result, as providing interoperability is crucial for a context sharing process involving different entities. CONON and PSW are examples of platforms that use ontologies for a common understanding of a set of concepts regarding context information by the entities interacting through them. The use of ontologies is one of the most effective ways to reach full interoperability because it is developed to work with different situations (i.e., inputs). ConCon uses the WordNet [64] tool, that is very useful in achieving interoperability. WordNet is an extensive lexical database of English. It groups a vast set of words, that include verbs, nouns, adjectives, and adverbs, into logical synonyms. With the use of WordNet, it is possible to discover a correlation between two (or more) different words. For instance, with the input of the words *cold, icy*, and *freezing*, WordNet can infer that they may converge in the meaning of *low temperature*. It is very useful in heterogeneous IoT environments, which have many devices with different specifications. The partial (PaI) interoperability addressed by some platforms means that the interoperability is present only on a specific set of entities or in a limited local way. They fail to reach the desired interoperability for context sharing in IoT environments.

Context processing (CP) is addressed by most of the analyzed platforms. ACC executes operations (e.g., modification) in sets of context information. CS-Sharing, HEAL, OIoT, FRASCS, and CoaaS use aggregation (A) techniques. Also, one of CS-Sharing key issues is to aggregate messages to reduce the network overhead and message size in resource-constrained vehicle networks. The filtering (F) technique is addressed by Grapevine, and ConCon. Grapevine filters information based on his contextual social properties (i.e., users, their characteristics, preferences, and the correlation with the environment) and only display the context information that matches with the filter. M3 is the only analyzed platform that provides the searching (S) feature. It uses Sindice[4] as a search engine. SE-TSDB and C2IoT stands out by providing both filtering and aggregation context processing features. C2IoT provides those functions at the Data Collection layer, near to the devices.

We observed that *infrastructure configuration and management (ICM)* is the most addressed feature by the analyzed platforms. This is expected, once the platform must accept the connection of entities to share context information. ACC proposes an approach in which an agent tries to join in the organization and the request can be refused or accepted. Chitchat is focused on providing the entry of devices to the device-to-device (D2D) network to share context. CONON provides a model that a specific ontology can be registered to the main ontology. Bluewave, CS-Sharing, and Grapevine use proximity to form temporary groups.

Scalability and real-time sharing (SRT) are not only linked to the performance of platforms and the time taken to do tasks. This issue is also related to what platforms can make to reduce the amount/size of context information exchanged by the network and provide more optimized communication. Chitchat fulfills this issue by creating a compressed representation of context information to enable a lightweight context sharing. It takes into account size and energy efficiency. DJess makes the inference process faster allowing better performance regarding both throughput and response time. Grapevine focuses on to be extended to real world scenarios where bandwidth and energy limit connectivity. It also minimizes the transmission overhead. Bluewave uses a lightweight communication protocol. Both CS-Sharing and Bluewave use short-range communication protocols for context sharing, avoiding network delay. BigClue adopts standards instead of customized solutions to help in providing scalability.

Regarding *availability (Av)*, most of the analyzed platforms enable context sharing process automatically (Aut), which is the most suitable technique for a dynamic environment as IoT. FRASCS automatically acquires and delivers the devices/applications context to the participant entities. In CS-Sharing, when entities meet each other, the context sharing process happens automatically. This is the same vision adopted by the opportunistically processing of OIoT and SCENTS.

Communication technologies (C) implies on which network technology the sharing platform supports. Some platforms do not present this information because in some cases, the platform is embedded in an entity, thus not caring about the communication services. However, most of the analyzed platforms have the capabilities of sharing context information by local (Loc) and external (Ext) networks. All the analyzed platforms that detail the communication technologies make use of the network layer communication (NeC). It helps on the interoperability by providing a standard communication channel. Bluewave, and ConCon have an architecture associated to smartphones, thus they are capable of sharing by 3G/4G, WiFi, and Bluetooth.

The *history (Hi)* feature is essential to retrieve older context information. It also can be used for probabilistic inference, as on Grapevine, HEAL, Chitchat, BigClue, and CoaaS. Bluewave has a module named *Context Repository* to store all set of context information alongside with *Context Brokers*. Both modules combined act as a database for context information. They play the role of a trusted entity in which the devices can share context information independently and without having relation to external objects. RCOS enables a query request for all context information of an entity. HEAL has cloud-based storage for context information, events, and valuable data to predict future interactions.

The context sharing process occurs regardless of a specific *architectural model (Ar)*. Most of analyzed platforms may adapt (Adp) itself according to the environment. In this case, they do not follow a specific architecture and in most cases are embedded in other entities. Some platforms, like Bluewave and RCOS, store data in a cloud-based server and process context information at the edge of the network, thus being a centralized-edge (Cen). On the other hand, some platforms, as CoSM and HEAL, are developed only in a cloud-based (Clo) approach, in which all the processing occurs in the cloud. HEAL works with low processing power devices, so the most resource demanding processing needs to be made in the cloud. Finally, there are decentralized-edge (Dec) platforms, as Magpie and CS-Sharing, that does not have a central point-of-control. By working with vehicle networks, CS-Sharing distributes the processing by the decentralized nodes.

5.4. International efforts

Interoperability in IoT environments is one of the focus of many research efforts programs. These programs can be composed by different countries making a consortium or even a large enterprise group looking for standardization. Some examples are EU FP7,[5]

[4] https://www.w3.org/2001/sw/wiki/Sindice.

[5] https://ec.europa.eu/research/fp7/index_en.cfm.

Table 4
International efforts regarding context sharing.

Name	Category	Application area	Main goal
EU FP7	Organization	Different technological areas	To support the research in Europe. It has three main projects regarding semantic interoperability: IoT-A, COIN, and IDIRA
IoT-A	Project	Internet of Things	To provide data interoperability by the definition of a reference architecture
COIN	Project	Industry	To provide semantic interoperability by the definition of web interfaces
IDIRA	Project	Crisis management	To facilitate coordination of large-scale disaster situations by improved interoperability
Horizon 2020	Organization	Science and innovation	To support the research in Europe. It has different project related to data interoperability
FIESTA-IoT	Project	Internet of Things	To provide data interoperability across different Internet of Things domains
INTER-IoT	Project	Internet of Things	To provide interoperability for Internet of Things environments by a multi-layered approach
INTER-Health	Project	Smart Healthcare	To share contextual information in a pub/sub manner on smart healthcare domain
Wise-IoT	Project	Internet of Things	To provide interoperability concerning context information
SEMIoTICS	Project	Internet of Things	To provide secure semantic interoperability
BIG IoT	Project	Internet of Things	To provide a unified Web API for IoT platforms. It also provides a way for monetizing shared data
symbIoTe	Project	Internet of Things	To provide an abstract layer for IoT platforms through a Web API.
FIWARE	Organization	Internet of Things	To provide a development platform for Internet of Things
Orion	Project	Internet of Things	To provide a context information broker
ETSI	Organization	Information and communications	To provide standards for Internet technologies and systems in communication, which include mobile networks, radio, fixed, broadcast
ISG CIM	Project	Smart cities	To share data/context between different spheres of a smart city
OMA SpecWorks	Organization	Internet of Things	To act as a "specifications factory" for Internet of Things
LWM2M	Project	Sensor networks	The management of low power devices for secure data transfer
IPSO	Project	Smart objects	To foster the use of Internet Protocol (IP) by smart objects
OCF	Organization	Internet of Things	To provide device interoperability
OneM2M	Organization	Internet of Things and M2M	To provide technical specifications for an interoperable M2M service layer

Horizon 2020,[6] ETSI,[7] FIWARE.[8] These efforts may differ from the ones presented in Table 3 in some aspects. The international efforts usually have a lot of different solutions and outcomes, as they try to meet different IoT challenges. In this sense, it is difficult to compare with platforms developed specifically for context sharing. Even so, as they also care about data/context/semantics interoperability, it is interesting to look deeper into that solutions. Table 4 shows an overview of the analyzed international efforts. Next paragraphs show their aspects that address context sharing in some way.

The IoT-A (Internet of Thing Architecture)[9] of the EU FP7 program tries to achieve data interoperability by providing a data format developed for resource-constrained environments, being able to minimize the traffic and the number of interactions by the network. The COIN [65] and IDIRA,[10] both from EU FP7, also try to achieve interoperability. COIN[11] provides a semantic based interoperability web solution. It uses the ontology-based approach for the semantic processing. COIN also maintains a knowledge-based system that holds information of distinct entities (i.e., devices, resources). IDIRA provides information sharing between various sources in crisis management scenarios. It uses ontologies to model context and discovery the destination of the shared context information. Even that it works with different sensors, it was developed for a specific crisis management domain.

One of the Horizon 2020 goals is making data interoperable allowing exchange and re-use between researchers, institutions, organizations, countries, etc. FIESTA-IoT[12] is an example of a project from Horizon 2020 that primes for interoperability. FIESTA-IoT is an infrastructure to provide data interoperability among already deployed IoT systems, platforms, and testbeds. It uses a common ontology to guarantee semantic conformity among different providers. It also provides a standard API for communication giving access to the information by the IoT systems connected to it. INTER-IoT and INTER-Health are also projects that received funding from Horizon 2020. The INTER-IoT[13] project goal is to provide interoperability on heterogeneous IoT platforms. INTER-IoT encompasses other projects to reach the interoperability in different layers: device level, networking level, middleware level, application service level, data and semantics level, integrated IoT platform level, and at the business level. On the data and semantics level, INTER-IoT developed the Generic Ontology for IoT Platforms (GOIoTP[14]) to support the semantic matching in IoT scenarios, facilitating the context sharing process. INTER-Health is a specific project under the INTER-IoT umbrella that presents an application scenario with context sharing [66]. On INTER-Health, messages and entities are described semantically using domain ontologies, facilitating the interoperability. It also has a specific communication bus for sharing context information to the subscribers.

[6] https://ec.europa.eu/programmes/horizon2020/en/.
[7] https://www.etsi.org/.
[8] https://www.fiware.org/.
[9] https://cordis.europa.eu/project/rcn/95713_en.html.
[10] https://cordis.europa.eu/project/rcn/98968_en.html.
[11] https://cordis.europa.eu/event/rcn/128988/en.
[12] http://fiesta-iot.eu/.
[13] https://inter-iot.eu/.
[14] https://inter-iot.github.io/ontology/.

Wise-IoT,[15] SEMIoTICS,[16] BIG IoT,[17] and symbIoTe[18] are also projects from the Horizon 2020. Wise-IoT proposes a Global IoT Services (GIoTS) layer with semantic interoperability ensuring reliability, and end-to-end security. It has a Morphing Mediation Gateway (MMG) component, which translates different protocols and data representations, working with different ontologies. SEMIoTICS is an in developing project that aims to provide a pattern-driven solution for semantic interoperability in IoT environments. It claims to support cross-layer adaption for heterogeneous smart objects. BIG IoT proposes the BIG IoT API, a Web API to be used by the IoT platforms, thus providing interoperability. Also, it provides the BIG IoT Marketplace, making possible for the platforms to share and monetize their data. The symbIoTe project, an abbreviation for *symbiosis of smart objects across IoT environments*, provides an abstract layer for a unified view of different IoT platforms. It has a standardized API for the interconnection of heterogeneous IoT solutions.

FIWARE provides a modular open source framework to foster the development of IoT solutions. The FIWARE framework acts as a middleware in the IoT environments, making possible the interconnection of devices and applications through different options of communication services. The Orion Context Broker[19] is one of the FIWARE's modules. It has the primary goal of managing context information. Orion acts as a context broker, receiving context from IoT environments and providing options to query or subscribe to a specific context. It also provides mechanisms to query/subscribe to a context by geolocation, type, and format. By acting as a pool of context, Orion may work for context interoperability.

European Telecommunications Standards Institute (ETSI) has established a special interest group to develop context management systems standards [67]. The group is called Industry Specification Group on Context Information Management (ISG CIM)[20] and focuses on smart city applications. The ISG CIM specifics a standard API to provide access for a context management system that focuses on smart cities environments containing heterogeneous data sources. They claim that their approach works for providing real-time access to the context information [68]. The ETSI specification does not try to replace the ways to exchange data between software platforms but offers standards to facilitate the cooperation among different platforms.

Looking deeper into the standardization for IoT, some efforts are made having the goal of standardizing the interactions with IoT devices. Some examples are projects and organizations such as LWM2M,[21] IPSO,[22] OCF,[23] OneM2M.[24] They all make attempts to come up with well-defined protocol stacks, data models and data representations, often using REST as messaging paradigm. They come up with models that describe how to model lights, buttons, inputs, outputs in IoT environments. By doing so, the meaning of data can be easily understood, or additional context can be retrieved from the device itself. This helps in achieve a better level of interoperability and facilitates further processing.

Lightweight M2M (LWM2M) is protocol created for remote managing lightweight and low power devices on a variety of networks [69]. It has an architectural design based on REST, and builds on the Constrained Application Protocol (CoAP) data transfer standard. The Internet Protocol for Smart Objects (IPSO) Alliance focuses on popularizing and incentive the use of Internet Protocol (IP) by smart entities (i.e., devices, systems). It also works to the definition of a framework considering privacy-preserving issues. It makes part of the OMA (Open Mobile Alliance) SpecWorks[25] organization that has efforts on interoperability for IoT. OMA SpecWorks also coordinate the LWM2M protocol.

The Open Connectivity Foundation (OCF) is a group whose solutions are to minimize communication effort between IoT devices. It provides a standard communication platform and data models that allow the communication among devices regardless of their characteristics, such as transport layer technology, application environment, operating system. The oneM2M is a well know global standards initiative for Machine-to-Machine (M2M) communications that also considers IoT environments. It provides technical specifications for requirements, architectures, APIs, security solutions and interoperability in IoT technologies. The oneM2M has the primary objective of providing technical specifications to achieve the demand for an interoperable M2M service layer to be used by different systems and devices.

The presented standardization efforts for IoT may help in reaching the context sharing feature. However, even with the growing amount of different IoT devices generating heterogeneous data, there is no standard for context information description [4,14,15]. Moreover, it is hard to impose a context format, one time that it may have different characteristics and providers. Thus, it remains a necessity for the implementation of a context sharing platform.

6. Challenges and future directions

In IoT, there is always a need for interoperability in different aspects. It is common in IoT environments to have various entities such as software systems and physical components from different producers. A platform to provide the horizontal integration of those heterogeneous entities is essential for the proper function of IoT environments. The context information plays a significant role in IoT. It is desired a platform able to provide context information interoperability in such environments [4,14]. Despite some of the analyzed works try to overcome the context interoperability issues providing the context sharing feature, there is no a platform that meets all the context sharing building blocks. Most of the analyzed works focus on a specific application domain/scenario. Thus, they fail in addressing some essential features to provide the complete context interoperability for IoT.

The development and strengthening of the context sharing field in IoT will only occur with the continuation of the ongoing research efforts. Thus, the next items present the major challenges to be addressed in the context sharing for the consolidation of the area. We also give future directions allowing readers to know which are the next steps in developing context sharing platforms for Internet of Things environments.

1) Interoperability: It remains a challenge to overcome in the context sharing field. In a significant amount of works, context sharing occurs locally or within a small group of similar entities. They try to care only about local sharing, not concerning the vast heterogeneity present in IoT environments. In most cases, they fail to provide an inter-domain context interoperability. Even that ontology works for the standardization by mitigating the interoperability challenge, and other technologies need to be employed in order to tackle such complex issue. Web services can be used to hide context systems patterns in order to standardize the commu-

[15] http://wise-iot.eu/en/home/.
[16] https://www.semiotics-project.eu/.
[17] http://big-iot.eu/.
[18] https://www.symbiote-h2020.eu/.
[19] https://fiware-orion.readthedocs.io/en/master/.
[20] https://portal.etsi.org/tb.aspx?tbid=854&SubTB=854.
[21] https://www.omaspecworks.org/what-is-oma-specworks/iot/lightweight-m2m-lwm2m/.
[22] https://www.omaspecworks.org/ipso-alliance/.
[23] https://openconnectivity.org/.
[24] http://www.onem2m.org/.
[25] https://www.omaspecworks.org/.

nication channel used for context information transport. As RESTful technology is commonly used by IoT middleware [70], it can also be used by context sharing platforms as well. Also, WordNet [64] can be more explored to mitigate interoperability issues. WordNet is a lexical database and an open source tool that works with the English language. It can correlate words and expressions semantically. By using WordNet, the context sharing platforms can relate two different entities and their events (i.e., context) [71]. Another concept that can be used to enhance interoperability in such scenarios is the Virtual Object. It appears in the IoT as the digital/virtual representation of the service(s) of a cyber-physical object. The Virtual Object can provide interoperability among heterogeneous objects by using semantic descriptions and context sharing techniques [72]. Moreover, Virtual Objects are gaining momentum in the IoT scenarios as secure lightweight virtualization solutions are being proposed [73,74].

2) Communication Technologies: The use of Low-Power Wide-Area (LPWA) networks must be considered in IoT environments and for context sharing as well. LPWA represents a set of technologies able to provide network communication for a considerable distance with low energy consumption [75]. LPWA networks support (i) low power devices such as the ones that can last for several years on battery, (ii) devices with low data throughput requirements, and (iii) long range operation [76]. The use of LPWA networks by the context sharing platforms, such as SigFox, LoRa, and NB-IoT, has potential to enable new forms of communication, making possible the sharing of context information in most IoT scenarios.

3) Fog and Edge Computing: Manashty et al. discuss how cloud-based context sharing platforms lack in fulfill context sharing requirements [77]. In this sense, we believe the adoption of Fog and Edge computing paradigms towards the architectural perspective can help to cover these requirements. Fog Computing paradigm brings the cloud applications physically closer to the IoT devices. It works in a distributed way. Fog Computing leverages cloud and edge resources along with its own infrastructure [78]. There is an urgency to minimize network communication in IoT by optimizing the systems that exchange data, and also context. The Fog and Edge computing paradigms can tackle this issue as well by improving the scalability of the systems. The Edge Computing paradigm reduces the amount of data (i.e., context) exchanged by the entities. It is related to perform some processing tasks of the systems at the final node of the communication layer (i.e., edge device), directly embedded into the devices themselves minimizing the communication with cloud instances [79]. Context sharing platforms can take advantage of both Fog and Edge computing paradigms to decrease latency and network overhead. Moreover, the decentralization provided by these paradigms helps in solving the heterogeneity problems of IoT environments in terms of systems and devices.

4) Hybrid Reasoning: The implementation of both Fog and Edge Computing paradigms facilitate a hybrid reasoning feature. The use of different reasoning techniques, accordingly to the environment, is a challenge for context sharing platforms. IoT environments tend to be heterogeneous by varying its characteristics (e.g., processing power, entities manufacturer). They can take benefit of a hybrid reasoning approach by adapting easier for each situation [4,80]. Following the hybrid Fog and Edge approach, a lightweight reasoning mechanism (e.g., rule-based system) can be embedded directly into IoT devices (edge) with resource-constrained characteristics. On the other hand, fog devices have more resource capabilities (e.g., processing power, unlimited energy) and can implement more complex reasoning (e.g., machine learning, ontologies).

5) Security and Privacy: Considered a leading challenge towards the definition of a context sharing platform. First, there is a need to protect contextualized information exchanged between entities. In this case, the use of lightweight communication protocols is seen as a viable choice [74]. Also, the context may include private information, such as medical and location data. In this sense, there are some efforts in defining a security architecture for IoT systems [32]. Moreover, there is still a lot to be explored to provide security at the hardware level, especially regarding the context-generating devices [81]. They are the primary targets for attacks and must be protected from the low level of hardware to the high level of software to ensure the integrity of the generated context.

6) Context-Aware Security: Besides the provision of privacy to the platform, context information may also be used for context-aware security decisions [82,83]. Shared context information is often used to improve the knowledge of the receiver, but context information should be used as well in smart environments for authentication, authorization, access control, and privacy-preserving services provision [84]. As analogy, let's consider the scenario of a door. A simple door will open with a key. A door using context-aware security functionalities will adapt itself depending on the environment (i.e., context). For example, a door may require different access control policies depending on the geographical place. While in a specific country, the door will need a physical key for the access, in another country it may require a secret password. Bringing this scenario to the IoT, we can replace the analogy of a door for a computing device, or a system. Furthermore, the context information can improve the communication channel security, by strengthening the network security, sending reports to a manager, or making data anonymous when some event is detected (e.g., an intruder on the network). Context-aware security ensures decision to be made according to the actual environmental context. Some efforts provide context-aware security services to IoT application scenarios [85–87]. However, those solutions do not care in providing the context-aware security feature considering the use of context information from heterogeneous application domains. The union of context sharing with context-aware security can leverage new frontiers in the development of security solutions to IoT environments.

7) Context Economy: An emerging trend is the development of marketplaces for IoT data. These marketplaces are community-driven software systems that allow device owners to sell their sensor or the actuator data for a monetary benefit [88]. There are some recent researches that present different kinds of IoT marketplaces [89–92]. However, most of these works only deal with raw IoT data. Even when the IoT marketplace uses a kind of context [91], it only uses the information of one domain and does not use it to provide new decisions. No IoT marketplace allows sellers and buyers to deal with high-level information (i.e., context) from different domains. To be able to perform such function, the IoT marketplace needs to add a layer of semantic interoperability, to deal with the shared context information. More than commercializing the high-level information, such Context Marketplace can avoid buyers' entities to perform a reasoning process, that is considered a performance demanding task. Moreover, such Context Marketplace has the potential to leverage the development of the area.

Fig. 8 presents an architecture with the typical features deployed in existing context sharing works (i.e., Building Blocks) together with the challenging functionalities that a complete context sharing platform for IoT should have. In our vision, one of the main things that help in achieve an ideal context sharing platform is the Edge and Fog computing approach. It helps with the scalability by providing decentralization, and it is possible to deal with the hybrid reasoning challenge. This two-layered approach gives the possibility of implement the same module in different scopes. For example, in our vision, it is essential to provide **Interoperability** features to the platform both in Fog and Edge layers. However, the

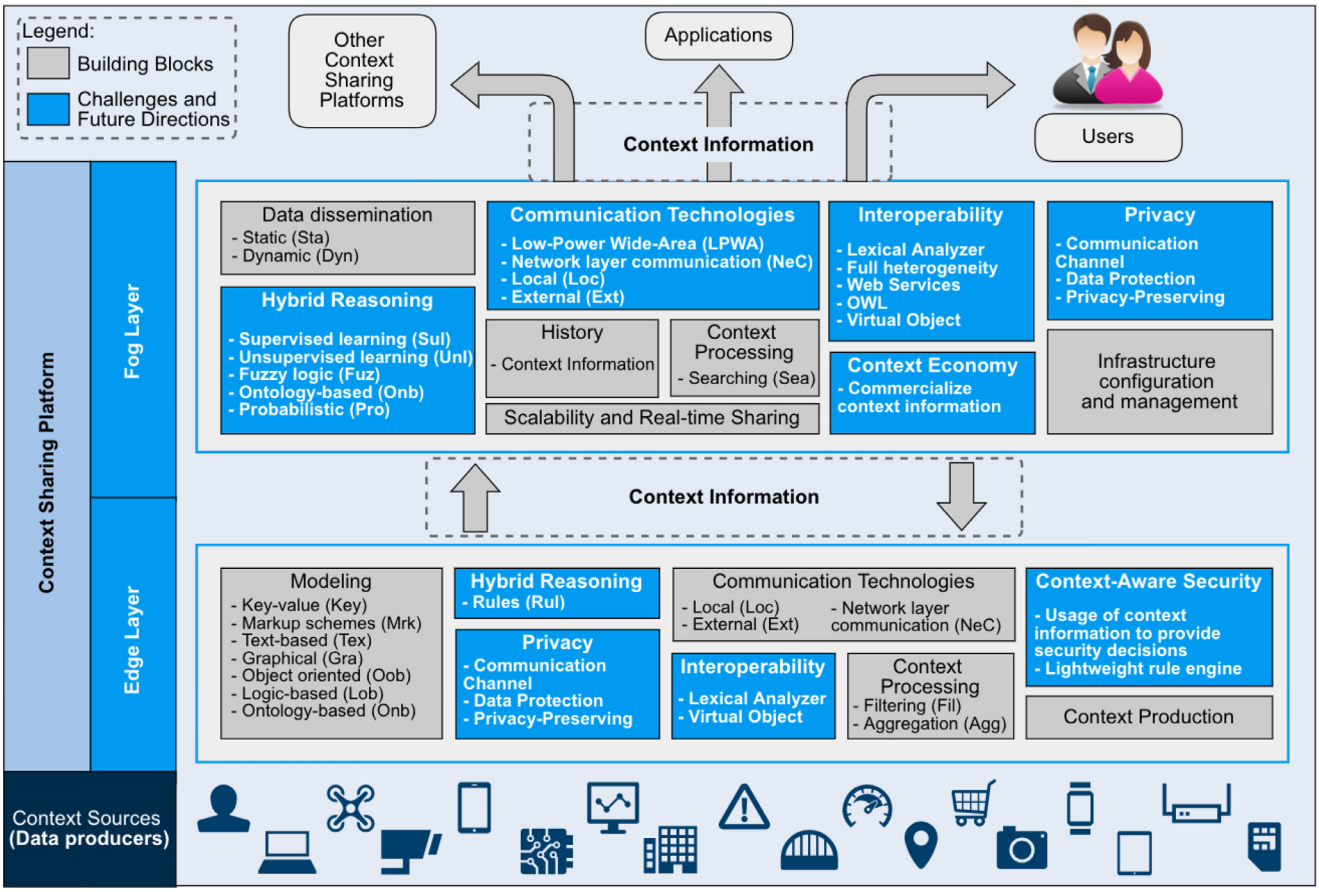

Fig. 8. A context sharing platform that compresses all the building blocks and challenges.

heavier techniques should be deployed at a Fog layer, which consists of more powerful devices (i.e., processing power, storage) than the Edge layer. In this context, the Edge layer being responsible for deploying lightweight techniques. The same logic can be used for the deployment of **Hybrid Reasoning** techniques.

The context information can play a significant role in providing security services as it has data that characterize the behavior of IoT entities (e.g., users, devices, systems) [93]. Following the proposed ideal architecture, the **Context-Aware Security** feature must be provided at the Edge layer. This makes possible the smart objects (Edge devices/Data producers) to provide security decisions/services based on the context, making it securely adaptive to the context of the deployed environment. The adoption of a lightweight rule engine by this module fits with the characteristics of Edge layer devices. Traditional **Privacy** also must be provided in both in Fog and Edge layers as well. Context information is sensitive information many times containing private data, so it is essential to make it private. Many solutions may opt to develop end-to-end privacy solutions.

The Fog module of **Communication Technologies** should implement LPWA technologies for long-range communications. We believe in that approach because some Fog-to-Fog communication may occur in a considerable distance between nodes, as most authors define Fog as a central point of communication for different Edge devices [21–24]. In this context, in most scenarios, the communication Edge-to-Fog may happen in shorter distances. The Fog layer may communicate directly with its Edge devices and also with different Fogs from other domains, acting as a pool for different connections. In light of these broader communication opportunities, the **Context Economy** should be deployed at the Fog layer to facilitate the process of monetizing and commercialize context information.

Regarding the implementation and deployment of the Building Blocks, the Edge layer should encompass Context Production, as it needs a direct connection with the IoT devices that act as context sources (i.e., data producers). As well as produce context information, the Edge layer should be responsible for the Modeling process, turning the context more understandable. This processing can be done by different techniques, depending on the deployment site. The Context Processing may happen both at the Edge and Fog layers. It is essential to aggregate different contexts produced at the Edge, thus generating an enriched context. Also, some filtering methods for the produced context information should be available. On the Fog layer side, the Context Processing technique of searching for a context should be implemented, as it has connections with different Edges and Fogs. The Communication Technologies should be present both in Edge and Fog layers for data transfer.

The Fog layer has the responsibility to implement some Building Blocks exclusively. The Infrastructure configuration and management should be performed at this layer because of its power of control and communication reachability to different Edge instances. Scalability and Real-time Sharing techniques should be used by the Fog layer to minimize the processing and the communication overhead in the whole sharing platforms and between the Edge layers. It could act as a processing distribution manager. The History feature takes place at the Fog layer by its larger storage capacity when compared with the Edge layer. Finally, as the Fog layer can send context to Fog instances of different domains, it is natural that the Data dissemination process takes place in such layer.

7. Conclusion

Context sharing platforms play a key role in providing context information interoperability in IoT environments. Even that there are various sharing platforms deployed with different characteristics, there are challenges to be overcome. In this survey, we presented essential building blocks towards the development of a context sharing platform. We also introduced various existing context information sharing platforms and discussed their features in detail. Finally, we reviewed the challenges and open issues for such platforms, the potential enhancements for them alongside with an ideal context sharing architecture that encompasses all the building blocks and the discussed challenges. In conclusion, we believe this paper can contribute to the research community by comparing context sharing platforms and helping readers to develop new solutions which address context sharing in IoT.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nivel Superior - Brasil (CAPES) - Finance Code 001. This work was supported in part by the USC Viterbi School of Engineering's Center for Cyber-Physical Systems and the Internet of Things.

References

[1] F. Xia, L.T. Yang, L. Wang, A. Vinel, Internet of things, Int. J. Commun. Syst. 25 (9) (2012) 1101.
[2] Gartner, Gartner Says 8.4 Billion Connected "Things" Will Be in Use in 2017, Up 31 Percent From 2016, 2017.
[3] A.B. Zaslavsky, C. Perera, D. Georgakopoulos, Sensing as a service and big data, CoRR abs/1301.0159 (2013) 1–8.
[4] C. Perera, A. Zaslavsky, P. Christen, D. Georgakopoulos, Context aware computing for the internet of things: a survey, Commun. Surv. Tut., IEEE 16 (1) (2014) 414–454, doi:10.1109/SURV.2013.042313.00197.
[5] G.D. Abowd, A.K. Dey, P.J. Brown, N. Davies, M. Smith, P. Steggles, Towards a better understanding of context and context-awareness, in: Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing, in: HUC '99, Springer-Verlag, London, UK, UK, 1999, pp. 304–307.
[6] X.H. Wang, D.Q. Zhang, T. Gu, H.K. Pung, Ontology based context modeling and reasoning using OWL, in: Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on, Ieee, 2004, pp. 18–22.
[7] G.S. Ramachandran, B. Krishnamachari, Towards a large scale IoT through partnership, incentive, and services: A Vision, architecture, and future directions, Open J. Int. Things (OJIOT) 5 (1) (2019) 80–92.
[8] L. Atzori, A. Iera, G. Morabito, The internet of things: a survey, Comput. Networks 54 (15) (2010) 2787–2805, doi:10.1016/j.comnet.2010.05.010.
[9] S. Bandyopadhyay, M. Sengupta, S. Maiti, S. Dutta, Recent Trends in Wireless and Mobile Networks: Third International Conferences, WiMo 2011 and CoNeCo 2011, Ankara, Turkey, June 26–28, 2011. Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 288–296. 10.1007/978-3-642-21937-5_27
[10] O. Yürür, C.H. Liu, Z. Sheng, V.C.M. Leung, W. Moreno, K.K. Leung, Context-awareness for mobile sensing: A Survey and future directions, IEEE Commun. Surv. Tut. 18 (1) (2016) 68–93, doi:10.1109/COMST.2014.2381246.
[11] O.B. Sezer, E. Dogdu, A.M. Ozbayoglu, Context-aware computing, learning, and big data in internet of things: A Survey, IEEE Int. Things J. 5 (1) (2018) 1–27, doi:10.1109/JIOT.2017.2773600.
[12] P. Pradeep, S. Krishnamoorthy, The MOM of context-aware systems: a survey, Comput. Commun. 137 (2019) 44–69, doi:10.1016/j.comcom.2019.02.002.
[13] T. Nam, T.A. Pardo, Conceptualizing smart city with dimensions of technology, people, and institutions, in: Proceedings of the 12th Annual International Digital Government Research Conference: Digital Government Innovation in Challenging Times, in: dg.o '11, ACM, New York, NY, USA, 2011, pp. 282–291, doi:10.1145/2037556.2037602.
[14] F. Boavida, A. Kliem, T. Renner, J. Riekki, C. Jouvray, M. Jacovi, S. Ivanov, F. Guadagni, P. Gil, A. Triviño, People-Centric Internet of Things–Challenges, Approach, and Enabling Technologies, Springer International Publishing, Cham, pp. 463–474. 10.1007/978-3-319-25017-5_44
[15] E. de Matos, L.A. Amaral, F. Hessel, Context-Aware Systems: Technologies and Challenges in Internet of Everything Environments, Springer International Publishing, Cham, pp. 1–25. 10.1007/978-3-319-50758-3_1
[16] L.D. Xu, W. He, S. Li, Internet of things in industries: a survey, IEEE Trans. Industr. Inf. 10 (4) (2014) 2233–2243, doi:10.1109/TII.2014.2300753.
[17] R. Casadei, G. Fortino, D. Pianini, W. Russo, C. Savaglio, M. Viroli, A development approach for collective opportunistic edge-of-Things services, Inf. Sci. 498 (2019) 154–169, doi:10.1016/j.ins.2019.05.058.
[18] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, D. Riboni, A survey of context modelling and reasoning techniques, Pervasive Mob. Comput. 6 (2) (2010) 161–180, doi:10.1016/j.pmcj.2009.06.002. Context Modelling, Reasoning and Management
[19] L. Snidaro, J. García, J. Llinas, Context-based information fusion: a survey and discussion, Inf. Fusion 25 (2015) 16–31, doi:10.1016/j.inffus.2015.01.002.
[20] E. Matos, L. Amaral, R. Tiburski, W. Lunardi, F. Hessel, S. Marczak, Context-aware system for information services provision in the Internet of Things, in: 2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA), 2015, pp. 1–4, doi:10.1109/ETFA.2015.7301624.
[21] OpenFog Consortium, OpenFog Reference Architecture for Fog Computing, Technical Report, February, 2017.
[22] R. Morabito, R. Petrolo, V. Loscrí, N. Mitton, Enabling a lightweight Edge Gateway-as-a-Service for the Internet of Things, in: 2016 7th International Conference on the Network of the Future (NOF), 2016, pp. 1–5, doi:10.1109/NOF.2016.7810110.
[23] A. Munir, P. Kansakar, S.U. Khan, IFCIoT: integrated fog cloud IoT: a novel architectural paradigm for the future internet of things, IEEE Consumer Electron. Mag. 6 (3) (2017) 74–82, doi:10.1109/MCE.2017.2684981.
[24] L. Lu, L. Xu, B. Xu, G. Li, H. Cai, Fog computing approach for music cognition system based on machine learning algorithm, IEEE Trans. Comput. Social Syst. (2018) 1–10, doi:10.1109/TCSS.2018.2871694.
[25] R. Agarwal, D.G. Fernandez, T. Elsaleh, A. Gyrard, J. Lanza, L. Sanchez, N. Georgantas, V. Issarny, Unified IoT ontology to enable interoperability and federation of testbeds, in: 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), 2016, pp. 70–75, doi:10.1109/WF-IoT.2016.7845470.
[26] S. De, G. Cassar, B. Christophe, S.B. Fredj, M. Bauer, N. Santos, T. Jacobs, E. Zeybek, R. de las Heras, G. Martín, et al., Concepts and solutions for entity-based discovery of IoT resources and managing their dynamic associations, EC FP7 IoT-A Deliverable 4 (2012).
[27] K. Nihei, Context sharing platform, NEC J. Adv. Technol. 1 (3) (2004) 200–204.
[28] A. Kansal, S. Nath, J. Liu, F. Zhao, Senseweb: an infrastructure for shared sensing, IEEE MultiMedia 14 (4) (2007) 8–13, doi:10.1109/MMUL.2007.82.
[29] G. Chen, D. Kotz, A Survey of Context-Aware Mobile Computing Research, Technical Report, Dartmouth College, Computer Science, Hanover, NH, 2000. TR2000-381
[30] T. Strang, C. Linnhoff-Popien, A context modeling survey, in: In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England, 2004, p. 8.
[31] A. Bikakis, T. Patkos, G. Antoniou, D. Plexousakis, Constructing Ambient Intelligence: AmI 2007 Workshops Darmstadt, Germany, November 7–10, 2007 Revised Papers, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 14–23. 10.1007/978-3-540-85379-4_3
[32] R.T. Tiburski, L.A. Amaral, E.D. Matos, F. Hessel, The importance of a standard security architecture for SOA-based IoT middleware, IEEE Commun. Mag. 53 (12) (2015) 20–26, doi:10.1109/MCOM.2015.7355580.
[33] W. Lunardi, E. Matos, R. Tiburski, L. Amaral, S. Marczak, F. Hessel, Context-based search engine for industrial IoT: discovery, search, selection, and usage of devices, in: IEEE Conference on Emerging Technology & Factory Automation (ETFA), IEEE, 2015, pp. 1–8.
[34] H.R.M. Nezhad, B. Benatallah, F. Casati, F. Toumani, Web services interoperability specifications, Computer 39 (5) (2006) 24–32, doi:10.1109/MC.2006.181.
[35] C. Doukas, L. Capra, F. Antonelli, E. Jaupaj, A. Tamilin, I. Carreras, Providing generic support for IoT and M2M for mobile devices, in: The 2015 IEEE RIVF International Conference on Computing Communication Technologies - Research, Innovation, and Vision for Future (RIVF), 2015, pp. 192–197, doi:10.1109/RIVF.2015.7049898.
[36] A. Azeez, S. Perera, D. Gamage, R. Linton, P. Siriwardana, D. Leelaratne, S. Weerawarana, P. Fremantle, Multi-tenant SOA middleware for cloud computing, in: 2010 IEEE 3rd International Conference on Cloud Computing, 2010, pp. 458–465, doi:10.1109/CLOUD.2010.50.
[37] A. Ricci, M. Viroli, A. Omicini, Agent coordination context: from theory to practice, in: R. Trappl (Ed.), Cybernetics and Systems 2004, 2, Austrian Society for Cybernetic Studies, Vienna, Austria, 2004, pp. 618–623.
[38] F. Cabitza, B. Dal Seno, DJess-a knowledge-sharing middleware to deploy distributed inference systems, in: WEC (2), 2005, pp. 66–69.
[39] J. Yamamoto, H. Nakagawa, K. Nakayama, Y. Tahara, A. Ohsuga, A context sharing message broker architecture to enhance interoperability in changeable environments, in: 2009 Third International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, 2009, pp. 31–39, doi:10.1109/UBICOMM.2009.48.
[40] A. Gyrard, C. Bonnet, K. Boudaoud, Enrich machine-to-machine data with semantic web technologies for cross-domain applications, in: 2014 IEEE World Forum on Internet of Things (WF-IoT), 2014, pp. 559–564, doi:10.1109/WF-IoT.2014.6803229.
[41] K. Xie, W. Luo, X. Wang, D. Xie, J. Cao, J. Wen, G. Xie, Decentralized context sharing in vehicular delay tolerant networks with compressive sensing, in: In-

ternational Conference on Distributed Computing Systems, 2016, pp. 169–178, doi:10.1109/ICDCS.2016.83.
[42] A.A. de Freitas, M. Nebeling, A.S.K.K. Ranithangam, J. Yang, A.K. Dey, Bluewave: enabling opportunistic context sharing via bluetooth device names, in: Symposium on Engineering Interactive Computing Systems, 2016, pp. 38–49, doi:10.1145/2933242.2933248.
[43] M. Ruta, F. Scioscia, S. Ieva, G. Loseto, F. Gramegna, A. Pinto, Knowledge discovery and sharing in the IoT: the physical semantic web vision, in: Proceedings of the Symposium on Applied Computing, in: SAC '17, ACM, New York, NY, USA, 2017, pp. 492–498, doi:10.1145/3019612.3019701.
[44] H. Rahman, M. Hussain, et al., LiO-IoT: a light-weight ontology to provide semantic interoperability in internet of things, Int. J. Comput.Intell. IoT 2 (4) (2018).
[45] D. Sinha Roy, R.K. Behera, K.H.K. Reddy, R. Buyya, A context-aware fog enabled scheme for real-time cross-vertical IoT applications, IEEE Int. Things J. 6 (2) (2019) 2400–2412, doi:10.1109/JIOT.2018.2869323.
[46] W. Zeng, S. Zhang, I. Yen, F. Bastani, Invited paper: semantic IoT data description and discovery in the IoT-edge-fog-cloud infrastructure, in: 2019 IEEE International Conference on Service-Oriented System Engineering (SOSE), 2019, pp. 106–115, doi:10.1109/SOSE.2019.00024.
[47] T. Lai, W. Li, H. Liang, X. Zhou, FRASCS: a framework supporting context sharing, in: 2008 The 9th International Conference for Young Computer Scientists, 2008, pp. 919–924, doi:10.1109/ICYCS.2008.227.
[48] A. Kroner, M. Schneider, J. Mori, A framework for ubiquitous content sharing, IEEE Pervasive Comput. 8 (4) (2009) 58–65, doi:10.1109/MPRV.2009.65.
[49] M. Madhukalya, M. Kumar, ConCon: context-aware middleware for content sharing in dynamic participating environments, in: 2014 IEEE 15th International Conference on Mobile Data Management, 1, 2014, pp. 156–161, doi:10.1109/MDM.2014.25.
[50] S. Cho, C. Julien, The grapevine context processor: application support for efficient context sharing, in: 2015 2nd ACM International Conference on Mobile Software Engineering and Systems, 2015, pp. 68–71, doi:10.1109/MobileSoft.2015.18.
[51] A. Manashty, J. Light, U. Yadav, Healthcare event aggregation lab (HEAL), a knowledge sharing platform for anomaly detection and prediction, in: 2015 17th International Conference on E-health Networking, Application Services (HealthCom), 2015, pp. 648–652, doi:10.1109/HealthCom.2015.7454584.
[52] C. Liu, C. Julien, Pervasive Context Sharing in Magpie: Adaptive Trust-Based Privacy Protection, vol. 162, Springer Verlag, Germany, pp. 122–139. 10.1007/978-3-319-29003-4_8
[53] D.A.L. Nuevo, D.R. Valles, E.M. Medina, R.M. Pallares, OIoT: a platform to manage opportunistic IoT communities, in: 2015 International Conference on Intelligent Environments, 2015, pp. 104–111, doi:10.1109/IE.2015.22.
[54] J. Dhallenne, P.P. Jayaraman, A. Zaslavsky, RCOS: Real Time Context Sharing Across a Fleet of Smart Mobile Devices, Springer International Publishing, Cham, pp. 87–100. 10.1007/978-3-319-46301-8_8
[55] S. Cho, C. Julien, Chitchat: navigating tradeoffs in device-to-device context sharing, in: International Conference on Pervasive Computing and Communications, 2016, pp. 1–10, doi:10.1109/PERCOM.2016.7456512.
[56] S. Faieq, R. Saidi, H. Elghazi, M.D. Rahmani, C2iot: a framework for cloud-based context-aware internet of things services for smart cities, Procedia Comput. Sci. 110 (2017) 151–158, doi:10.1016/j.procs.2017.06.072. 14th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017) / 12th International Conference on Future Networks and Communications (FNC 2017) / Affiliated Workshops
[57] D. Huru, C. Leordeanu, E. Apostol, V. Cristea, BigClue: towards a generic IoT cross-domain data processing platform, in: 2018 IEEE 14th International Conference on Intelligent Computer Communication and Processing (ICCP), 2018, pp. 427–434, doi:10.1109/ICCP.2018.8516597.
[58] A. Hassani, A. Medvedev, P.D. Haghighi, S. Ling, M. Indrawan-Santiago, A. Zaslavsky, P.P. Jayaraman, Context-as-a-service platform: exchange and share context in an IoT ecosystem, in: 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2018, pp. 385–390, doi:10.1109/PERCOMW.2018.8480240.
[59] C. Liu, J. Hua, C. Julien, SCENTS: collaborative sensing in proximity IoT networks, in: 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2019, pp. 189–195, doi:10.1109/PERCOMW.2019.8730863.
[60] E. Friedman-Hill, Jess, the Java expert system shell, 1997, 10.2172/565603
[61] A. Gyrard, A machine-to-machine architecture to merge semantic sensor measurements, in: Proceedings of the 22Nd International Conference on World Wide Web, in: WWW '13 Companion, ACM, New York, NY, USA, 2013, pp. 371–376, doi:10.1145/2487788.2487945.
[62] M. Compton, P. Barnaghi, L. Bermudez, R. GarcíA-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog, et al., The SSN ontology of the W3C semantic sensor network incubator group, Web Semantics 17 (2012) 25–32.
[63] B. Chazelle, J. Kilian, R. Rubinfeld, A. Tal, The bloomier filter: an efficient data structure for static support lookup tables, in: Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, in: SODA '04, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2004, pp. 30–39.
[64] G.A. Miller, Wordnet: a lexical database for english, Commun. ACM 38 (11) (1995) 39–41, doi:10.1145/219717.219748.
[65] F.M. Facca, S. Komazec, C. Guglielmina, S. Gusmeroli, COIN: platform and services for saas in enterprise interoperability and enterprise collaboration, in: 2009 IEEE International Conference on Semantic Computing, 2009, pp. 543–550, doi:10.1109/ICSC.2009.72.
[66] G. Fortino, C. Savaglio, C.E. Palau, J.S. de Puga, M. Ganzha, M. Paprzycki, M. Montesinos, A. Liotta, M. Llop, Towards Multi-layer Interoperability of Heterogeneous IoT Platforms: The INTER-IoT Approach, Springer International Publishing, Cham, pp. 199–232. 10.1007/978-3-319-61300-0_10
[67] ETSI, ETSI launches new group on Context Information Management for smart city interoperability, 2017, (http://goo.gl/PLAoHb), [Online; accessed 04-September-2018].
[68] ETSI, ETSI ISG CIM group releases first specification for context exchange in smart cities, 2018, (http://goo.gl/QgTRdE), [Online; accessed 04-September-2018].
[69] C.A. Putera, F.J. Lin, Incorporating OMA Lightweight M2M protocol in IoT/M2M standard architecture, in: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)(WF-IOT), 00, 2015, pp. 559–564, doi:10.1109/WF-IoT.2015.7389115.
[70] A.H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, Q.Z. Sheng, IoT middleware: a survey on issues and enabling technologies, IEEE Int. Things J. 4 (1) (2017) 1–20, doi:10.1109/JIOT.2016.2615180.
[71] E. Mingozzi, G. Tanganelli, C. Vallati, B. Martínez, I. Mendia, M. González-Rodríguez, Semantic-based context modeling for quality of service support in IoT platforms, in: 2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016, pp. 1–6, doi:10.1109/WoWMoM.2016.7523563.
[72] M. Nitti, V. Pilloni, G. Colistra, L. Atzori, The virtual object as a major element of the internet of things: a survey, IEEE Commun. Surv. Tut. 18 (2) (2016) 1228–1240, doi:10.1109/COMST.2015.2498304.
[73] R.T. Tiburski, C.R. Moratelli, S.F. Johann, M.V. Neves, E. d. Matos, L.A. Amaral, F. Hessel, Lightweight security architecture based on embedded virtualization and trust mechanisms for IoT edge devices, IEEE Commun. Mag. 57 (2) (2019) 67–73, doi:10.1109/MCOM.2018.1701047.
[74] R.T. Tiburski, L.A. Amaral, E. de Matos, D.F.G. de Azevedo, F. Hessel, The role of lightweight approaches towards the standardization of a security architecture for IoT middleware systems, IEEE Commun. Mag. 54 (12) (2016) 56–62, doi:10.1109/MCOM.2016.1600462CM.
[75] R.S. Sinha, Y. Wei, S.-H. Hwang, A survey on LPWA technology: LoRa and NB-IoT, ICT Expr. 3 (1) (2017) 14–21, doi:10.1016/j.icte.2017.03.004.
[76] M. Taneja, LTE-LPWA networks for IoT applications, in: 2016 International Conference on Information and Communication Technology Convergence (ICTC), 2016, pp. 396–399, doi:10.1109/ICTC.2016.7763505.
[77] A. Manashty, J.L. Thompson, Cloud Platforms for IoE Healthcare Context Awareness and Knowledge Sharing, Springer International Publishing, Cham, pp. 303–322. 10.1007/978-3-319-50758-3_12
[78] A.V. Dastjerdi, R. Buyya, Fog computing: helping the internet of things realize its potential, Computer 49 (8) (2016) 112–116, doi:10.1109/MC.2016.245.
[79] W. Shi, S. Dustdar, The promise of edge computing, Computer 49 (5) (2016) 78–81, doi:10.1109/MC.2016.145.
[80] A.I. Maarala, X. Su, J. Riekki, Semantic reasoning for context-Aware internet of things applications, IEEE Internet Things J. 4 (2) (2017) 461–473, doi:10.1109/JIOT.2016.2587060.
[81] C. Moratelli, S. Johann, M. Neves, F. Hessel, Embedded virtualization for the design of secure IoT applications, in: 27th International Symposium on Rapid System Prototyping: Shortening the Path from Specification to Prototype, ACM, 2016, pp. 2–6, doi:10.1145/2990299.2990301.
[82] P.K. Das, S. Narayanan, N.K. Sharma, A. Joshi, K. Joshi, T. Finin, Context-sensitive policy based security in internet of things, in: 2016 IEEE International Conference on Smart Computing (SMARTCOMP), 2016, pp. 1–6, doi:10.1109/SMARTCOMP.2016.7501684.
[83] F. Al-Turjman, S. Alturjman, Context-sensitive access in industrial internet of things (IIot) healthcare applications, IEEE Trans. Industr. Inform. 14 (6) (2018) 2736–2744, doi:10.1109/TII.2018.2808190.
[84] Y.J. Jia, Q.A. Chen, S. Wang, A. Rahmati, E. Fernandes, Z.M. Mao, A. Prakash, S.J. Unviersity, ContexIoT: towards providing contextual integrity to appified IoT platforms, in: Proceedings of the 21st Network and Distributed System Security Symposium (NDSS'17), 2017, pp. 1–15.
[85] F. Al-Turjman, S. Alturjman, Confidential smart-sensing framework in the IoT era, J. Supercomput. 74 (10) (2018) 5187–5198, doi:10.1007/s11227-018-2524-1.
[86] E. de Matos, R.T. Tiburski, L.A. Amaral, F. Hessel, Providing context-aware security for IoT environments through context sharing feature, in: 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom), 2018, pp. 1711–1715, doi:10.1109/TrustCom/BigDataSE.2018.00257.
[87] L. Zhang, Y. Li, L. Wang, J. Lu, P. Li, X. Wang, An efficient context-aware privacy preserving approach for smartphones, Secur. Commun. Networks 2017 (2017) 1–11, doi:10.1155/2017/4842694.
[88] D. Niyato, X. Lu, P. Wang, D.I. Kim, Z. Han, Economics of internet of things: an information market approach, IEEE Wireless Commun. 23 (4) (2016) 136–145, doi:10.1109/MWC.2016.7553037.
[89] European Commission - Horizon 2020, AutoMat - automotive Big Data marketplace for innovative cross-sectorial vehicle data services, 2018,
[90] M. Travizano, M. Minnoni, G. Ajzenman, C. Sarraute, N. Della Penna, Wibson: adecentralized marketplace empowering individuals to safely monetize their personal data, 2018.
[91] K. Nagorny, S. Scholze, M. Ruhl, A.W. Colombo, Semantical support for a CPS data marketplace to prepare Big Data analytics in smart manufacturing

environments, in: 2018 IEEE Industrial Cyber-Physical Systems (ICPS), 2018, pp. 206–211, doi:10.1109/ICPHYS.2018.8387660.

[92] B. Krishnamachari, J. Power, S.H. Kim, C. Shahabi, I3: an IoT marketplace for smart communities, in: Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services, ACM, 2018, pp. 498–499.

[93] J.L.H. Ramos, J.B. Bernabe, A.F. Skarmeta, Managing context information for adaptive security in IoT environments, in: 2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops, 2015, pp. 676–681, doi:10.1109/WAINA.2015.55.

Everton de Matos is a Ph.D. candidate in the area of context-awareness for Internet of Things, at the Computer Science School at the Pontifical Catholic University of Rio Grande do Sul (PUCRS), acting at Embedded System Group (GSE). He is an adjunct professor at Meridional Faculty (IMED), Brazil. He has experience in computer science and his research interests are IoT, Middleware for IoT, and context-awareness in IoT environments.

Ramão Tiago Tiburski received his M.S. degree in computer science from PUCRS. He is a Ph.D. student of computer science at PUCRS and a professor at Federal Institute of Santa Catarina (IFSC). His research interests are IoT, fog and edge computing, and security for IoT resource-constrained devices.

Carlos Roberto Moratelli received his Ph.D. in computer science from PUCRS. He is an adjunct professor at UFSC. He worked ten years in the telecommunication industry, acting on software engineering related to embedded systems. His research interests are embedded real-time systems, Linux Embedded, and virtualization for embedded systems.

Sergio Johann Filho received his Ph.D. degree in computer science from PUCRS. He is an adjunct professor at PUCRS, Brazil. He has experience in computer architecture design and organization, operating systems, embedded systems (design and integration), embedded software support, real-time systems and control systems.

Leonardo Albernaz Amaral has received his M.Sc. and Ph.D. degrees in computer science from PUCRS. He has experience as R&D research leader and project manager. He is an adjunct professor at FTEC Technology School, Brazil. He has experience in computer science with an emphasis in Middleware systems, RFID, IoT, Smart Cities, and pervasive systems. He has publications in prestigious conferences, journals, and books.

Gowri Ramachandran is a postdoctoral researcher at Center for Cyber-Physical Systems and the Internet-of-Things (CCI) at University of Southern California. He received his Ph.D. from imec-DistriNet, KU Leuven, Belgium. His research interests include Internet-of-Things (IoT), smart cities, and blockchain.

Bhaskar Krishnamachari received his Ph.D. degree from Cornell University, Ithaca, NY, USA, in 2002. He is currently a Professor with the Department of Electrical Engineering, Viterbi School of Engineering, University of Southern California, Los Angeles, CA, USA. His primary research interest include the design and analysis of algorithms and protocols for next-generation wireless networks and the Internet of Things.

Fabiano Hessel is Full Professor of Computer Science at PUCRS. He received his Ph.D. in computer science from UJF, France (2000). He has experience as General and Program Chair in several committees of prestigious conferences and journals. His research interests are embedded real-time systems, RTOS and MPSoC systems applied to IoT/SmartCities.