Context-Aware Systems: Technologies and Challenges in Internet of Everything Environments

Everton de Matos, Leonardo Albernaz Amaral and Fabiano Hessel

Abstract The Internet of Things (IoT) and Internet of Everything (IoE) paradigms have emerged in the last years, thus generating new challenges in the pervasive computing area. IoT is a computing paradigm that has been recognized for allowing the connection of the physical and virtual worlds by giving processing power to the daily "things". IoE goes beyond the IoT by breaking the barrier of just "things". In IoE, the people, data and processes also make part of the connected world. Context awareness has becoming an important feature in IoT and IoE scenarios. Automatic decision making, sensitivity to context, automatic notification to the user, just to name a few, are some examples of situations where a context-aware system is needed in these environments where the characteristics of the data sources are undergoing constant change. In this chapter we present the context-aware definitions and architecture in IoE and it evolution from IoT. Moreover, we present the context-aware life-cycle phases, which is the process done in order to have context information. In addition, we also analyze the current context-aware approaches of IoT/IoE systems, and present some challenges related to context-aware IoE systems.

1 Introduction

In the last years a computing paradigm called Internet of Things (IoT) has gained significant attention. The basic idea of IoT is the pervasive presence around us of a variety of things or objects (e.g., RFID tags, sensors, etc.) that cooperate with their neighbors to reach common goals [3]. By embedding mobile networking and information processing capability into a wide array of gadgets and everyday items

E. de Matos (✉) · L.A. Amaral · F. Hessel
Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brazil
e-mail: everton.matos.001@acad.pucrs.br

L.A. Amaral
e-mail: leonardo.amaral@acad.pucrs.br

F. Hessel
e-mail: fabiano.hessel@pucrs.br

J.M. Batalla et al. (eds.), *Beyond the Internet of Things*,
Internet of Things, DOI 10.1007/978-3-319-50758-3_1

enabling new forms of communication between people and things, and between things themselves, the Internet of Things has been adding new dimensions to the world of information and communication technology [5]. It promises to create a world where all the objects around us are connected to the Internet and communicate with each other with minimum human intervention.

Beyond the IoT, the concept of Internet of Everything (IoE) has also gained prominence in network and communication scenarios. In addition to the "Things", IoE connects people, data, and processes in networks of billions or even trillions of connections [18]. These connections create vast amounts of data, some of these data that we never had access to before. When these data are analyzed and used intelligently, the possibilities seem endless.

There is a common sense that the data providers in IoE environments will generate a lot of data, and they will only be useful if we could analyze, interpret and understand them [47]. In this sense, context-aware computing has played an important role in tackling this challenge in previous paradigms, such as mobile and pervasive computing, which lead us to believe that it would continue to be successful in the IoE as well [39]. Mobile devices can be part of IoE scenarios and their characteristics are constantly changing (e.g., status, location). Context-aware approaches allow us to discover and store context information linked to these devices. In this sense, context-awareness became a fundamental feature of IoE in order to have a fully automated environment and improve the user's Quality of Experience (QoE) [31].

The concept of context is attached to the information that will be used to characterize the situation of an entity. In this sense, a system becomes context-aware if it uses the context in order to provide new information to user [2]. Taking into account these definitions, an IoE environment needs a context-aware system to be aware of the environment in order to help the user by providing these information in the most useful way. In the context-awareness area, there is a set of methods to build context. These methods are organized in phases that the systems must follow to produce context information that characterizes the context life-cycle of an information [39].

The main contribution of this chapter is to present a discussion about the context-aware systems technologies and challenges in IoE environments in order to provide a view of what can be the best technologies to fit with the necessities of IoE environments and what is the new trends in the area. We will also argue about the context life-cycle, we will show a detailed view of all the phases and the most useful technologies. In addition, we will identify some existing work related to context-awareness in IoE environments and also how we can have a fully functional platform respecting the requirements and challenges of context-aware systems in IoE environments.

The remainder of this paper is organized as follows: Sect. 2 provides the concepts of the IoT evolution into IoE. Section 3 provides a theoretical background about context, we also present the context life-cycle definitions and techniques. Section 4 provides an overview of the characteristics of the systems that produce context information. Section 5 provides a study of some related work. Section 6 shows how context is present in IoE environments, moreover we show the technologies and challenges involving this issue. We conclude this chapter with a summary in Sect. 7.

2 From Internet of Things (IoT) to Internet of Everything (IoE)

Internet of Things (IoT) is a novel computing paradigm that is rapidly gaining space in scenarios of modern communication technologies. The idea of the IoT is the pervasive presence of a variety of things or objects (e.g., RFID tags, sensors, actuators, smart phones, smart devices, etc.), that are able to interact with each other and cooperate with their neighbors to reach common goals through unique addressing schemes and reliable communication media over the Internet [3, 21].

During the past decade, the IoT has gained significant attention in academia as well as industry. The main reasons behind this interest are the capabilities that the IoT will offer [27]. It promises to create a world where all the objects (also called smart objects [30]) around us are connected to the Internet and communicate with each other with minimum human intervention. The ultimate goal is to create “a better world for human beings”, where objects around us know what we like, what we want, what we need, and act accordingly without explicit instructions [17].

The Intranet is being extended to smart things [30] with a higher scalability, pervasiveness, and integration into the Internet Core. This extension is leading to reach a real IoT, where things are first class citizens in the Internet, and they do not need to relay any more on a gateway, middleware, proxy, or broker. IoT drives towards integrating everything into the Internet Core, this trend is the denominated Internet of Everything (IoE). The integration of everything is motivated by the market wish to have all processes remotely accessible through a uniform way [28].

The IoT idea implied other concepts, such as Internet of Service (IoS), Internet of Everything (IoE), Web of Things (WoT), which of course represent the IoT. When we consider the relations M2M (Man to Man), M2T (Man to Thing), M2P (Man to People), P2P (People to People), and D2D (Device to Device), we ultimately reach the IoE [49]. IoE is a new Internet concept that tries to connect everything that can be connected to the Internet, where everything refers to people, cars, televisions (TVs), smart cameras, microwaves, sensors, and basically anything that has Internet-connection capability [1].

The IoE connects people, data, things, and processes in networks of billions or even trillions of connections. These connections create vast amounts of data, some of it data we’ve never had access before. When this data is analyzed and used intelligently, the possibilities seem endless [18].

Today, less than 1% of things that could be connected are connected to the Internet or intelligent systems. Projections show that by 2017, 3.5 billion people will be connected to the Internet, 64% of them via mobile devices [13]. People and connected things will generate massive amounts of data, an estimated 40 trillion gigabytes, that will have a significant impact on daily life [1]. It will enable faster response times to medical or public safety emergencies and save lives, it will improve the quality of citizen life by providing direct and personal services from the government, and it will uncover new information about how our cities work, thus enabling city leaders to use resources more efficiently and save money while providing superior services. There are three key ways in which the IoE will significantly impact our lives, as described in the following examples [13]:

- **The IoE will automate connections**: Today, people must proactively connect to the network or Internet via mobile devices like smartphones and tablets and to other people on the network via social media websites. Citizens must proactively call a certain phone number for an enterprise complaint or for an emergency. Imagine if people were connected automatically to systems of services instead. Wearable computers in clothing or watches, or sensors in pills that are swallowed, could automatically send patient information to doctors and nurses. This would allow a sick or an elderly person to manage his or her healthcare from home rather than a hospital or nursing home, getting automatic reminders to take medicine or immediate preventive care for changes in health status. For example, weight gain in cardiac patients is often an early indicator of returning heart problems. Connected scales from the home can be used to alert a doctor of a change in patient weight so that quick action can be taken to prevent another heart attack.
- **The IoE will enable fast personal communications and decision making**: Now imagine that intelligence is embedded within sensors or devices. This means the device itself will filter out relevant information and even apply analytics, so in the case of the connected scale, only when a certain threshold of weight gain is crossed will doctors and nurses be alerted. This type of data not only will enable faster, better decision making but also will help government workers, doctors, and citizens more efficiently manage their time. Instead of doctors searching through files or ordering a battery of tests, information would be sent to them directly from patients to help make decisions. Patients will have faster response times from doctors based on such highly personalized information. This is another example of how the Internet of Everything will completely change the types of services that are offered and also how they are delivered to citizens.
- **The IoE will uncover new information**: With the deployment of so many sensors and other information-gathering devices, city managers will be able to understand their city as never before. An interesting example is the use of acoustic sensors that are calibrated to detect gunshots. Some cities in the United States have deployed these sensors in areas of gun violence and discovered some shocking information. Police departments had historically assumed that residents called the police 80% of the time when shots were heard. These police departments were operating on highly inaccurate information about the level of gun violence in certain neighborhoods. With this new information, police can now plan their patrols differently and better target areas to reduce gun violence.

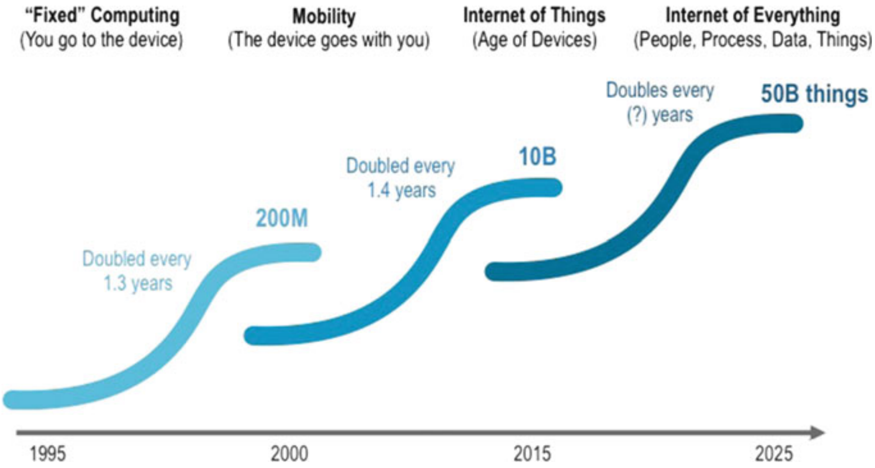

Fig. 1 Internet growth is occurring in waves [32]

As things add capabilities like context awareness, increased processing power, and energy independence, and as more people and new types of information are connected, IoT becomes an Internet of Everything—a network of networks where billions or even trillions of connections create unprecedented opportunities as well as new risks (see Fig. 1, extracted from [32]).

IoE brings together people, process, data, and things to make networked connections more relevant and valuable than ever before—turning information into actions that create new capabilities, richer experiences, and unprecedented economic opportunity for businesses, individuals, and countries (see Fig. 2) [18]. To better understand this definition, we must first break down IoE's individual components.

- **People**: In IoE, people will be able to connect to the Internet in innumerable ways. Today, most people connect to the Internet through their use of devices (such as PCs, tablets, TVs, and smartphones) and social networks. As the Internet evolves toward IoE, we will be connected in more relevant and valuable ways. For example, in the future, people will be able to swallow a pill that senses and reports the health of their digestive tract to a doctor over a secure Internet connection. In addition, sensors placed on the skin or sewn into clothing will provide information about a person's vital signs. According to Gartner [32], people themselves will become nodes on the Internet, with both static information and a constantly emitting activity system.
- **Data**: With IoT, devices typically gather data and stream it over the Internet to a central source, where it is analyzed and processed. As the capabilities of things connected to the Internet continue to advance, they will become more intelligent by combining data into more useful information. Rather than just reporting raw data, connected things will soon send higher-level information back to machines, computers, and people for further evaluation and decision making. This

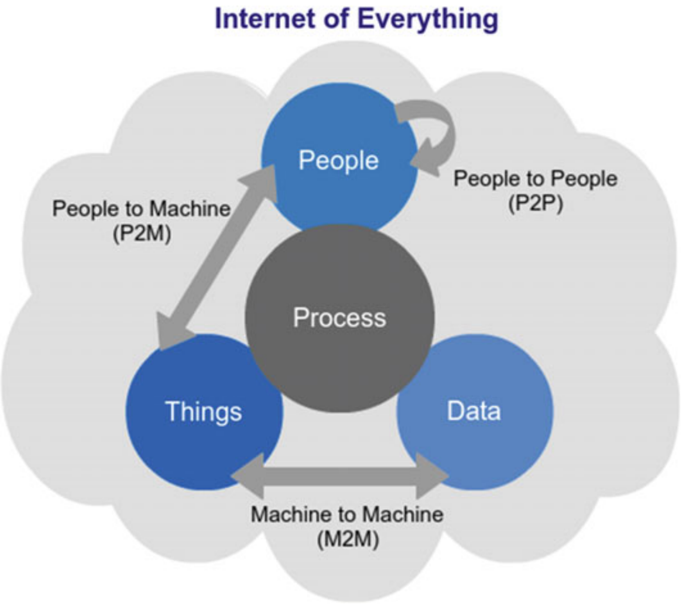

Fig. 2 The what, where, and how of the Internet of Everything

transformation from data to information in IoE is important because it will allow us to make faster, more intelligent decisions, as well as control our environment more effectively.

- **Things**: This group is made up of physical items like sensors, consumer devices, and enterprise assets that are connected to both the Internet and each other. In IoE, these things will sense more data, become context-aware, and provide more experiential information to help people and machines make more relevant and valuable decisions. Examples of "things" in IoE include smart sensors built into structures like bridges, and disposable sensors that will be placed on everyday items such as milk cartons [18].
- **Process**: Process plays an important role in how each of these entities—people, data, and things—work with the others to deliver value in the connected world of IoE. With the correct process, connections become relevant and add value because the right information is delivered to the right person at the right time in the appropriate way.

2.1 Architecture

Implementation of IoE environments is usually based on a standard architecture derived from IoT. This architecture consists of several layers [5, 18]: from the data

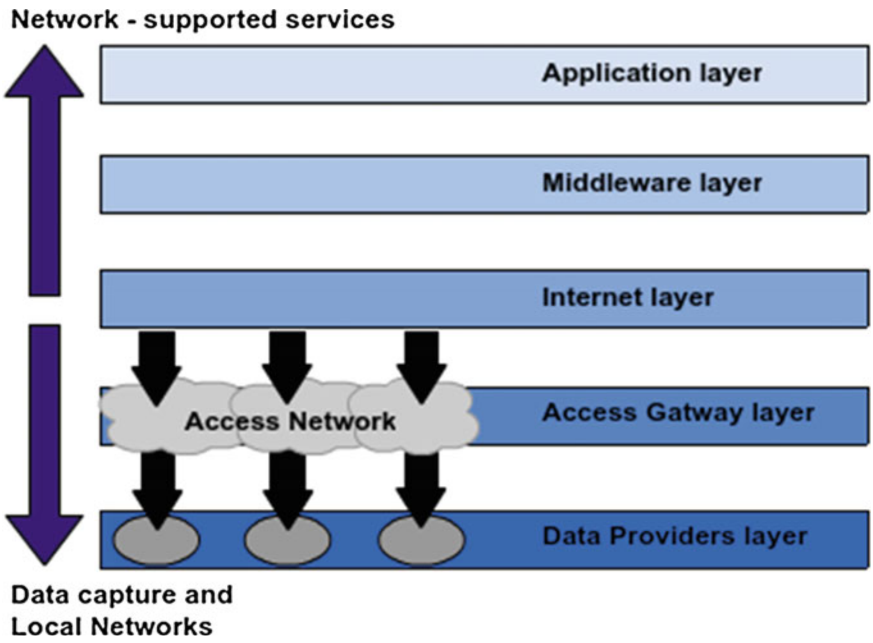

Fig. 3 Layered architecture of Internet of Everything

acquisition layer at the bottom to the application layer at the top. Figure 3 presents the generic architecture for IoE [3].

The two layers at the bottom contribute to data capturing while the two layers at the top are responsible for data utilization in applications. Next, we present the functionality of these layers [5]:

- **Data providers layer**: This layer consists of hardware components such as sensor networks, embedded systems, RFID tags and readers or other IoE devices in different forms. Moreover, in this layer is also present other components, like people information, that is also an IoE entity that provides data to the environment. These entities are the primary data sources deployed in the field. Many of these elements provide identification and information storage (e.g. RFID tags), information collection (e.g. sensor networks), information processing (e.g. embedded edge processors), communication, control and actuation. However, identification and information collection are the primary goals of these entities, leaving the processing activities for the upper layers.
- **Access gateway layer**: The first stage of data handling happens at this layer. It takes care of message routing, publishing and subscribing, and also performs cross platform communication, if required.
- **Middleware layer**: This layer acts as an interface between the hardware layer at the bottom and the application layer at the top. It is responsible for critical functions such as device management and information management, and also

takes care of issues like data filtering, data aggregation, semantic analysis, access control, and information discovery.

- **Application layer**: This layer at the top of the stack is responsible for the delivery of various services to different users/applications in IoE environments. The applications can be from different industry verticals such as: manufacturing, logistics, retail, environment, public safety, healthcare, food and drug, etc.

2.2 Characteristics and Environments

IoT allows communication among very heterogeneous devices connected by a very wide range of networks through the Internet infrastructure. IoT devices and resources are any kind of device connected to Internet, from existing devices, such as servers, laptops, and personal computers, to emerging devices such as smart phones, smart meters, sensors, identification readers, and appliances [28].

In addition to the physical devices, IoT is also enriched with the cybernetic resources and Web-based technologies. For that purpose, IoT is enabled with interfaces based on Web Services such as RESTFul architecture and the novel protocol for Constrained devices Applications Protocol (CoAP) [43]. These interfaces enable the seamless integration of the IoT resources with information systems, management systems, and the humans. Reaching thereby a universal and ubiquitous integration among human networks (i.e., society), appliance networks, sensor networks, machine networks, and, in definitive, everything networks [28].

Beside these devices, the People and Data (see Fig. 2) can also make part of this connection, thus we have the IoE. IoE offers several advantages and new capabilities for a wide range of application areas. For example, nowadays IoE is finding applications for the development of Smart Cities, starting with the Smart Grid, Smart Lighting and transportation with new services such as Smart Parking and the Bicycle Sharing System [20] for building sustainable and efficiently smart ecosystems [28].

The application of the IoE is not limited to high scale deployments such as the locations in Smart Cities, elsewhere it can also be considered for consumer electronics, vehicular communications, industrial control, building automation, logistic, retail, marketing, and healthcare [28].

3 Context-Aware Life-Cycle

Context is considered any information that can be used to characterize the situation of an entity. Entity is a person, place, or computing device (also called thing) that is relevant to the interaction between a user and an application, including the user and the application themselves. A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task [2, 34]. In this way, an IoE ecosystem requires a context-aware mechanism to be aware of the environment situation in order to help the user in the most useful way. In various cases, the context-aware becomes a feature of IoE systems.

Different researchers have identified context types based of different perspectives. Abowd et al. [2] introduced one of the leading mechanisms of defining context types. They identified location, identity, time, and activity as the primary context types. Further, they defined secondary context as the context that can be found using primary context [39]. For example, given primary context such as a person's identity, we can acquire many pieces of related information such as phone numbers, addresses, email addresses, etc. Some examples defined by [39] are:

- **Primary context**: Any information retrieved without using existing context and without performing any kind of sensor data fusion operations (e.g. GPS sensor readings as location information).
- **Secondary context**: Any information that can be computed using primary context. The secondary context can be computed by using sensor data fusion operations or data retrieval operations such as web service calls (e.g. identify the distance between two sensors by applying sensor data fusion operations on two raw GPS sensor values). Further, retrieved context such as phone numbers, addresses, email addresses, birthdays, list of friends from a contact information provider based on a personal identity as the primary context can also be identified as secondary context.

A set of methods is mandatory in order to obtain the context of an entity. Furthermore, there is a set of actions, organized in phases, that characterizes the context life-cycle of an information. Perera et al. [39] proposed a life-cycle and explained how acquisition, modelling, reasoning, and distribution of context should occur.

3.1 Context Acquisition

In acquisition process, context needs to be acquired from various information sources. These sources can be physical or virtual devices. The techniques used to acquire context can vary based on responsibility, frequency, context source, sensor type, and acquisition process [39].

(1) *Based on Responsibility*: Context acquisition can be primarily accomplished using two methods [40]: push and pull.

  - Push: The physical or virtual sensor pushes data to the data consumer which is responsible to acquiring sensor data periodically or instantly. Periodical or instant pushing can be employed to facilitate a publish and subscribe model.

- Pull: The data consumers make a request from the sensor hardware periodically or instantly to acquire data.

(2) *Based on Frequency*: There are two different types: Instant and Interval.

- Instant: These events occur instantly. The events do not span across certain amounts of time. In order to detect this type of event, sensor data needs to be acquired when the event occurs. Both push and pull methods can be employed.
- Interval: These events span in a certain period of time. In order to detect this type of event, sensor data needs to be acquired periodically. Both push and pull methods can be employed.

(3) *Based on Source*: Context acquisition methods can be organized into three categories [12].

- Acquire directly from sensor hardware: In this method, context is directly acquired from the sensor by communicating with the sensor hardware and related APIs. Software drivers and libraries need to be installed locally.
- Acquire through a middleware infrastructure: In this method, sensor (context) data is acquired by middleware solutions. The applications can retrieve sensor data from the middleware and not from the sensor hardware directly.
- Acquire from context servers: In this method, context is acquired from several other context storage types (e.g. databases, web services) by different mechanisms such as web service calls.

(4) *Based on Sensor Types*: In general usage, the term ‘sensor’ is used to refer the tangible sensor hardware devices. However, among the technical community, sensors are referred as any data source that provides relevant context. Therefore, sensors can be divided into three categories [26]: physical, virtual, and logical.

- Physical sensors: These are the most commonly used type of sensors. These sensors generate data by themselves. Most of the devices we use today are equipped with a variety of physical sensors (e.g. temperature, humidity, microphone, touch).
- Virtual sensors: These sensors do not necessarily generate data by themselves. Virtual sensors retrieve data from many sources and publish it as sensor data (e.g. calendar, contact number directory, twitter statuses, email, and chat applications). These sensors do not have a physical presence.
- Logical sensors (also called software sensors): They combine physical sensors and virtual sensors in order to produce more meaningful information. A web service dedicated to providing weather information can be called a logical sensor.

(5) *Based on Acquisition Process*: Here are three ways to acquire context: sense, derive, and manually provided.

- Sense: The data is sensed through sensors, including the sensed data stored in databases (e.g. retrieve temperature from a sensor, retrieve appointments details from a calendar).
- Derive: The information is generated by performing computational operations on sensor data. These operations could be as simple as web service calls or as complex as mathematical functions running over sensed data (e.g. calculate distance between two sensors using GPS coordinates).
- Manually provided: Users provide context information manually via predefined settings options such as preferences (e.g. understanding that user doesn't like to receive event notifications between 10 pm to 6 am).

3.2 Context Modeling

Context modeling is organized in two steps [7]. First, new context information needs to be defined in terms of attributes, characteristics, and relationships with previously specified context. In the second step, the outcome of the first step needs to be validated and the new context information needs to be merged and added to the existing context information repository. Finally, the new context information is made available to be used when needed.

The most popular context modeling techniques are surveyed in [11, 44]. These surveys discuss a number of systems that have been developed based on the following techniques. Each technique has its own strengths and weaknesses.

(1) *Key-Value Modelling*: In the key-value each data has a key. The key-value technique is an application oriented and application bounded technique that suits the purpose of temporary storage such as less complex application configurations and user preferences. It models context information as key-value pairs in different formats such as text files and binary files. This is the simplest form of context representation among all the other techniques. They are easy to manage when they have smaller amounts of data. However, key-value modelling is not scalable and not suitable to store complex data structures.

(2) *Markup Scheme Modelling (Tagged Encoding)*: It models data using tags. Therefore, context is stored within tags. This technique is an improvement over the key-value modelling technique. The advantage of using markup tags is that it allows efficient data retrieval. Markup schemes such as XML are widely used in almost all application domains to store data temporarily, transfer data among applications, and transfer data among application components. In contrast, markup languages do not provide advanced expressive capabilities which allow reasoning.

(3) *Graphical Modelling*: It models context with relationships. Some examples of this modelling technique are Unified Modelling Language (UML) [45] and Object Role Modelling (ORM) [36]. Actual low-level representation of the graphical modelling technique could be varied. For example, it could be a SQL

database, noSQL database, etc. Further, as we are familiar with databases, graphical modelling is a well-known, easy to learn, and easy to use technique. Databases can hold massive amounts of data and provide simple data retrieval operations, which can be performed relatively quickly. In contrast, the number of different implementations makes it difficult with regards to interoperability.

(4) *Object Based Modelling*: Object based (or object oriented) concepts are used to model data using class hierarchies and relationships. Object oriented paradigm promotes encapsulation and re-usability. As most of the high-level programming languages support object oriented concepts, modelling can be integrated into context-aware systems easily. Object based modelling is suitable to be used as an internal, non-shared, code based, run-time context modelling, manipulation, and storage mechanism. Validation of object oriented designs is difficult due to the lack of standards and specifications.

(5) *Logic Based Modelling*: Facts, expressions, and rules are used to represent information about the context. Rules are primarily used to express policies, constraints, and preferences. It provides much more expressive richness compared to the other models discussed previously. Therefore, reasoning is possible up to a certain level. Logic based modelling allows new high-level context information to be extracted using low-level context.

(6) *Ontology Based Modelling*: The context is organized into ontologies using semantic technologies. A number of different standards and reasoning capabilities are available to be used depending on the requirement. A wide range of development tools and reasoning engines are also available. However, context retrieval can be computationally intensive and time consuming when the amount of data is increased.

3.3 Context Reasoning

Context reasoning can be defined as a method of deducing new knowledge based on the available context [8]. It can also be explained as a process of giving high-level context deductions from a set of contexts [22]. Reasoning is also called inferencing. Broadly the reasoning can be divided into three phases [35].

- Context pre-processing: This phase cleans the collected sensor data. Due to inefficiencies in sensor hardware and network communication, collected data may be not accurate or missing. Therefore, data needs to be cleaned by filling missing values, removing outliers, validating context via multiple sources, and many more.
- Sensor data fusion: It is a method of combining sensor data from multiple sensors to produce more accurate, more complete, and more dependable information that could not be achieve through a single sensor [24].
- Context inference: It is a method of generation of high-level context information using lower-level context. The inferencing can be done in a single interaction or

in multiple interactions. For example in a situation where the context is represented as tuples (e.g. Who: Leonardo, What: walking: 1 km/h, Where: Porto Alegre, When: 2016-01-05:11.30 am). This low-level context can be inferred through a number of reasoning mechanisms to generate the final results. For example, in the first iteration, longitude and latitude values of a GPS sensor may be inferred as Rei do Cordeiro restaurant in Porto Alegre. In the next iteration Rei do Cordeiro restaurant in Porto Alegre may be inferred as Leonardo's favourite restaurant. Each iteration gives more accurate and meaningful information.

In [39], context reasoning techniques are classify into six categories: supervised learning, unsupervised learning, rules, fuzzy logic, ontological reasoning, and probabilistic reasoning.

(1) *Supervised learning*: In this category of techniques, we first collect training examples. Then we label them according to the results we expect. Then we derive a function that can generate the expected results using the training data. Decision tree is a supervised learning technique where it builds a tree from a dataset that can be used to classify data.
(2) *Unsupervised learning*: This category of techniques can find hidden structures in unlabeled data. Due to the use of no training data, there is no error or reward signal to evaluate a potential solution.
(3) *Rules*: This is the simplest and most straightforward method of reasoning. Rules are usually structure in an IF-THEN-ELSE format. Rules are expected to play a significant role in the IoE, where they are the easiest and simplest way to model human thinking and reasoning in machines.
(4) *Fuzzy logic*: This allows approximate reasoning instead of fixed and crisp reasoning. Fuzzy logic is similar to probabilistic reasoning but confidence values represent degrees of membership rather than probability [42]. In traditional logic theory, acceptable truth values are 0 or 1. In fuzzy logic partial truth values are acceptable. It allows real world scenarios to be represented more naturally; as most real world facts are not crisp.
(5) *Ontology based*: It is based on description logic, which is a family of logic based knowledge representations of formalisms. The advantage of ontological reasoning is that it integrates well with ontology modelling. In contrast, a disadvantage is that ontological reasoning is not capable of finding missing values or ambiguous information where statistical reasoning techniques are good at that. Rules can be used to minimize this weakness by generating new context information based on low-level context.
(6) *Probabilistic logic*: This category allows decisions to be made based on probabilities attached to the facts related to the problem. This technique is used to understand occurrence of events. For example, it provides a method to bridge the gap between raw GPS sensor measurements and high level information such as a user destination, mode of transportation, calendar based observable evidence such as user calendar, weather, etc.

3.4 Context Distribution

Finally, context distribution is a fairly straightforward task. It provides methods to deliver context to the consumers. From the consumer perspective this task can be called context acquisition. There are two methods that are commonly used in context distribution [39]:

- Query: Context consumer makes a request in terms of a query, so the context management system can use that query to produce results.
- Subscription (also called publish/subscribe): Context consumer can be allowed to subscribe to a context management system by describing the requirements. The system will then return the results periodically or when an event occurs. In other terms, consumers can subscribe for a specific sensor or to an event.

4 Context-Aware Systems

Context-awareness involves acquisition of contextual information, modelling of these information, reasoning about context, and distribution of context. A system for context-awareness would provide support for each of these tasks. It would also define a common model of context, which all agents can use in dealing with context. Moreover, it would ensure that different agents in the environment have a common semantic understanding of contextual information.

4.1 Architecture Overview

In terms of architecture, some authors have identified and comprehensively discussed some design principles related to context-aware systems [39]. We summarize the findings below with brief explanations. This list is not intended to be exhaustive. Only the most important design aspects are considered.

- **Architecture layers and components**: The functionalities need to be divided into layers and components in a meaningful manner. Each component should perform a very limited amount of the task and should be able to perform independently up to a large extent.
- **Scalability and extensibility**: The component should be able to be added or removed dynamically. For example, new functionalities (i.e. components) should be able to be add without altering the existing components (e.g. Open Services Gateway initiative). The component needs to be developed according to standards across the solutions, which improves scalability and extensibility (e.g. plug-in architectures).

- **Application Programming Interface (API)**: All the functionalities should be available to be accessed via a comprehensive easy to learn and easy to use API. This allows the incorporation of different solutions very easily. Further, API can be used to bind context management frameworks to applications. Interoperability among different IoE solutions heavily depends on API and their usability.
- **Mobility support**: In the IoE, most devices would be mobile, where each one has a different set of hardware and software capabilities. Therefore, context-aware frameworks should be developed in multiple versions, which can run on different hardware and software configurations (e.g. more capabilities for server level software and less capabilities for mobile phones).
- **Monitoring and detect event**: Events play a significant role in the IoE, which is complemented by monitoring. Detecting an event triggers an action autonomously in the IoE paradigm. This is how the IoE will help humans carry out their day-to-day work easily and efficiently. Detecting events in real-time is a major challenge for context-aware frameworks in the IoE paradigm.

4.2 Systems Features

The context-aware systems must have several features to deal with the context information production. First we will introduce some of these features, and in the Sect. 5 a comparison table of systems regarding these features will be shown. The most important features are surveyed by Perera et al. [39] and explained in the follow items:

(1) **Modelling**: It has been discussed in detail in Sect. 3.2. The following abbreviations are used to denote the context modeling techniques employed by the system: key-value modelling (K), markup Schemes (M), graphical modelling (G), object oriented modelling (Ob), logic-based modelling (L), and ontology-based modelling (On).

(2) **Reasoning**: It has been discussed in detail in Sect. 3.3. The following abbreviations are used to denote the reasoning techniques employed by the system: supervised learning (S), unsupervised learning (U), rules (R), fuzzy logic (F), ontology-based (O), and probabilistic reasoning (P).

(3) **Distribution**: It has been discussed in detail in Sect. 3.4. The following abbreviations are used to denote the distribution techniques employed by the system: publish/subscribe (P) and query (Q).

(4) **History and Storage**: Storing context history is critical in both traditional context-aware computing and IoE [16]. Historic data allows sensor data to be better understood. Specifically, it allows user behaviors, preferences,

patterns, trends, needs, and many more to be understood. The symbol (✓) is used to denote that context history functionality is facilitated and employed by the system.

(5) **Knowledge Management**: Most of the tasks that are performed by IoE systems solutions require knowledge in different perspectives, such as knowledge on sensors, domains, users, activities, and many more. Knowledge can be used for tasks such as automated configuration of sensors to IoE system, automatic sensor data annotation, reasoning, and event detection. The symbol (✓) is used to denote that knowledge management functionality is facilitated and employed by the system in some perspective.

(6) **Event Detection**: IoE envisions many types of communication. Most of these interactions are likely to occur based on an event. An occurrence of event is also called an event trigger. Once an event has been triggered, a notification or action may be executed. For example, detecting current activity of a person or detecting a meeting status in a room, can be considered as events. Mostly, event detection needs to be done in real-time. However, events such as trends may be detected using historic data. The symbol (✓) is used to denote that event detection functionality is facilitated and employed by the system in some perspective.

(7) **Level of Context Awareness**: Context-awareness can be employed at two levels: low (hardware) level and high (software) level. At the hardware level, context-awareness is used to facilitate tasks such as efficient routing, modelling, reasoning, storage, and event detection [25]. The software level has access to a broader range of data and knowledge as well as more resources, which enables more complex reasoning to be performed. The following abbreviations are used to denote the level of context awareness facilitated and employed by the system: high level (H) and low level (L).

(8) **Data Source Support**: There are different sources that are capable of providing context. The (P) denotes that the solution supports only physical sensors. Software sensors (S) denotes that the solution supports either virtual sensors, logical sensors or both. The (A) denotes that the solution supports all kinds of data sources (i.e. physical, virtual, and logical). The (M) denotes that the solution supports mobile devices.

(9) **Quality of Context**: It denotes the presence of conflict resolution functionality using (C) and context validation functionality using (V). Conflict resolution is critical in the context management domain [19]. Context validation ensures that collected data is correct and meaningful. Possible validations are checks for range, limit, logic, data type, cross-system consistency, uniqueness, cardinality, consistency, data source quality, security, and privacy.

(10) **Data Processing**: Are denoted the presence of context aggregation functionality using (A) and context filter functionality using (F). Context filter functionality makes sure that the reasoning engine processes only important data. Filtering functionality can be presented in different solutions with in

different forms: filter data, filter context sources, or filter events. Aggregation can just collect similar information together. This is one of the simplest forms of aggregation of context.

(11) **Dynamic Composition**: IoE solutions must have a programming model that allows dynamic composition without requiring the developer or user to identify specific sensors and devices. Software solutions should be able to understand the requirements and demands on each situation, then organize and structure its internal components according to them. The symbol (✓) denotes the presence of dynamic composition functionality at the system in some form.

(12) **Real-Time Processing**: Most of the interactions are expected to be processed in real-time in IoE. This functionality has been rarely addressed by the research community in the context-aware computing domain. The symbol (✓) denotes the presence of real-time processing functionality in some form.

(13) **Registry Maintenance and Lookup Services**: The (✓) symbol is used to denote the presence of registry maintenance and lookup services functionality in the system. This functionality allows different components such as context sources, data fusion operators, knowledge bases, and context consumers to be registered.

5 Related Work

Some systems provide context-aware functions to IoT and IoE environments. This Section presents some examples of these systems and a brief review about their context-aware features based on systems features presented at Sect. 4.2.

Tables 1 and 2 presents a comparison between systems with context-aware features. The items (features) used for the comparison are: (1) Modelling, (2) Reasoning, (3) Distribution, (4) History and Storage, (5) Knowledge Management, (6) Event Detection, (7) Level of Context Awareness, (8) Data Source Support, (9) Quality of Context, (10) Data Processing, (11) Dynamic Composition,

Table 1 Context life-cycle phases implemented in IoE systems

IoE system	(1)	(2)	(3)
Hydra	K, On, Ob	R, O	Q
COSMOS	Ob	R	Q
SALES	M	R	Q
C-Cast	M	R	P, Q
CoMiHoc	Ob	R, P	Q
MidSen	K	R	P, Q
CARISMA	M	R	Q
ezContext	K, Ob	R	Q
Feel@Home	G, On	O	P, Q

Table 2 Context features implemented in IoE systems

IoE system	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)
Hydra	✓	✓	✓	H	P	V	–	–	–	–
COSMOS	–	–	✓	H	P	–	A	✓	–	✓
SALES	–	–	✓	L	P	–	F	–	–	✓
C-Cast	✓	–	✓	H	A	–	–	–	–	✓
CoMiHoc	–	✓	✓	H	A	V	–	–	–	–
MidSen	–	✓	✓	H	P	–	–	–	–	✓
CARISMA	–	–	–	H	M	C	–	–	–	–
ezContext	✓	✓	✓	H	A	–	A	–	–	✓
Feel@Home	–	✓	✓	H	A	–	–	–	–	✓

(12) Real-Time Processing, and (13) Registry Maintenance and Lookup Services. We only analyzed systems that provide details of these features in the literature. The definition of each item is given at the Sect. 4.2.

Hydra [4] is a system that comprises a Context-Aware Framework that is responsible for connecting and retrieving data from sensors, context management and context interpretation. A rule engine called Drools [29] has been employed as the core context reasoning mechanism. COSMOS [14] is a system that enables the processing of context information in ubiquitous environments. COSMOS consists of three layers: context collector (collects information), context processing (derives high level information), and context adaptation (provides context access to applications). Therefore, COSMOS follows distributed architecture which increases the scalability of the system.

SALES [15] is a context-aware system that achieves scalability in context dissemination. The XML schemes are used to store and transfer context. C-Cast [41] is a system that integrates WSN into context-aware systems by addressing context acquisition, dissemination, representation, recognizing, and reasoning about context and situations. The data history can be used for context prediction based on expired context information.

CoMiHoc [46] is a framework that supports context management and situation reasoning. CoMiHoc architecture comprises six components: context provisioner, request manager, situation reasoner, location reasoner, communication manager, and On-Demand Multicast Routing Protocol (ODMRP). MidSen [37], as C-Cast, is a context-aware system for WSN. MidSen is based on Event-Condition-Action (ECA) rules. The system proposes a complete architecture to enable context awareness in WSN.

CARISMA [9] is focused on mobile systems where they are extremely dynamic. Adaptation is the main focus of CARISMA. Context is stored as application profiles (XML based), which allows each application to maintain meta-data. The framework ezContext [33] provides automatic context life cycle management. The ezContext comprises several components that provides context, retrieves context, modelling and storage context. Feel@Home [23] is a context management framework that supports interaction between different domains. Feel@Home decides what the relevant domain needs to be contacted to answer the user query. Then, the framework redirects the user query to the relevant domain context managers. Feel@Home consists of context management components responsible for context reasoning and store context.

The first feature to be analyzed in Table 1 (first column) is related to systems context modeling feature. The most popular modeling approaches in the comparison were markup schemes, key-value, and object-oriented modeling. Modeling through key-value is made by Hydra for simplicity of use [4]. CARISMA uses markup schemes because the way it models the context can be easily understood, both by machines and by human [9].

In reasoning and distribution (2 and 3 respectively) almost all analyzed systems seem to have a consensus regarding which technologies to use. With respect to reasoning, the most of analyzed systems use rules as a tool. A study by [39] showed that rules is the most popular method of reasoning used by systems. Hydra besides rules also uses ontologies as a promising technology [48]. On the other hand, Feel@Home makes use only of ontologies. To supply context distribution all analyzed systems use query. However, some systems as C-Cast, MidSen, and Feel@Home also offer the possibility of using publish/subscribe as a plus.

In Table 2 the function of history and storage (4) is a differential of the analyzed systems. Only three have this feature. For C-Cast, the history can be used for context prediction based on expired context information [41]. Another differential feature is knowledge management (5). One of the few that provides this functionality is CoMiHoc. In this system, the knowledge is required to overcome the limitations of the environment and to provide reliable support for the applications [46]. Detection of events (6) is a feature provided by almost all systems. When specific context events occur, event detection takes action such as shutting down if the battery is low [4].

In terms of level of context awareness (7), only one system has a low level, which works with the context in hardware. All other analyzed systems work with context in terms of software, which allows a greater capacity for reasoning [39]. Regarding data source support (8), most of analyzed systems support physical sensors. CARISMA supports mobile sensors because it is a specific solution for this area [9]. A better alternative is to support the largest possible range of different sensors, since IoE provides a heterogeneous environment [3].

A comparison was made between systems on quality of context (9). Only three of the analyzed systems control quality of context, and two of them control through validation. In CoMiHoC validation is integrated into the communication protocol [46]. Data processing (10) is another analyzed functionality. Only three systems perform some kind of processing. SALES uses filtering techniques to reduce traffic [15].

Another feature compared between systems was dynamic composition (11). This is only attended by COSMOS [14]. The real-time processing (12) becomes a challenge of future context-aware systems, as none of the analyzed systems had this feature. Finally, the last item used for systems analysis was registry maintenance and lookup services (13). Many of the compared systems have this feature. Through it, the systems can have a history of performed processes, thus making easy future operations [39].

6 Context-Awareness in IoE

Data alone are not very interesting or useful. It is when data can be used and become actionable that it can change processes and have direct positive impact on people's lives. The IoE generates data, and adding analysis turns those data into information. Aggregated data become information that, when analyzed, become knowledge. Knowledge can lead to context and informed decision-making, which at the highest level is wisdom (Fig. 4) [38].

Data for critical decision-making though the IoE can create approximately US $14.4 trillion dollars of added value in the US commercial sector over the next 10 years across a wide range of industries [38]. This opportunity exists in the form of new value created by technology innovation, market share gains, and increasing competitive advantage. Similarly researches indicates that data analytics were responsible for an improvement in business performance of companies. Capturing these gains, however, may require concurrent investment in resources to manage the rise in data [18].

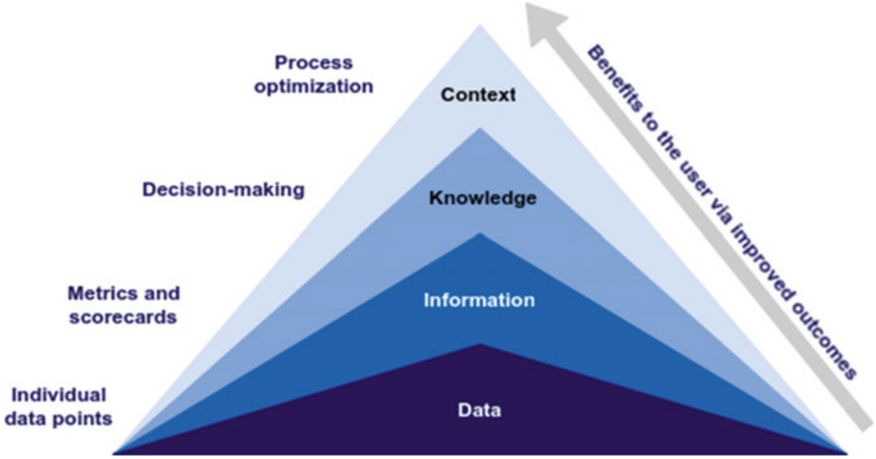

Fig. 4 Turning data into context

6.1 Technologies and Challenges

In this section our objective is to discuss eight unique challenges in the IoE where novel techniques and solution may need to be employed [38, 39].

(1) Automated configuration of data providers: In traditional pervasive/ubiquitous computing, we connect only a limited number of data providers to the applications (e.g. smart farm, smart home) [6]. In contrast, the IoE envisions billions of data providers to be connected together over the Internet. As a result, a unique challenge would arise on connection and configuration of data providers to applications. There has to be an automated or at least semi-automated process to connect data providers to applications.

(2) Context discovery: Once we connect data providers to a software solution, there has to be a method to understand the data produced by the data providers and the related context automatically. There are many types of context that can be used to enrich data. However, understanding sensor data and appropriately annotating it automatically in a paradigm such as IoE, where application domains vary widely, is a challenging task.

(3) Acquisition, modelling, reasoning, and distribution: No single technique would serve the requirements of the IoE. Incorporating and integrating multiple techniques has shown promising success in the field. Some of the early work, such as [7, 10], have discussed the process in detail. However, due to the immaturity of the field of IoE, it is difficult to predict when and where to employ each technique. Therefore, it is important to define and follow a standard specification so different techniques can be added to the solutions without significant effort.

(4) Selection of data providers: It is clear that we are going to have access to billions of data providers. In such an environment, there could be many different alternative data providers to be used. For example, in some cases, there will be many similar data providers in a complex environment like a smart city.

(5) Security, privacy, and trust: The advantage of context is that it provides more meaningful information that will help us to understand a situation or data. At the same time, it increases the security threats due to possible misuse of the context (e.g. identity, location, activity, and behavior). In the IoE, security and privacy need to be protected in several layers: sensor hardware layer, sensor data communication (protocol) layer, context annotation and context discovery layer, context modelling layer, and the context distribution layer. IoE is a community based approach where the acceptance of the users (e.g. general public) is essential. Therefore, security and privacy protection requirements need to be carefully addressed in order to win the trust of the users.

(6) Scalability: The growth of mobile data traffic will require greater radio spectrum to enable wireless M2M, as well as people-to-people (P2P) and people-to-machine (P2M), connectivity. Ensuring device connectivity and sufficient bandwidth for all of the uses of wireless sensors will require careful

planning. Moreover, the context to be processing will grow, and the context systems will need to adapt to this scenario keeping the reliability.

(7) Reliability: As more critical processes are conducted as part of the IoE, the need for reliability increases. Healthcare applications that require instant communication between end-users and medical professionals, safety and security applications, utility functions, and industrial uses are some examples where continuous, uninterrupted, real-time communications require reliable and redundant connectivity. The context systems will be present in these fields, and they must work correctly in these critical scenarios.

(8) Context Sharing and Interoperability: This is largely neglected in the context-aware systems domain. Most of the systems solutions or architectures are designed to facilitate applications in isolated factions. Inter-systems communication is not considered to be a critical requirement. However, in the IoE, there would no central point of control. Different systems developed by different parties will be employed to connect to sensors, collect, model, and reason context. Therefore, sharing context information between different kinds of systems or different instances of the same systems is important. Standards and interoperability issues span both the technical and architectural domains. In this sense, an interoperability between systems will be required.

7 Summary

The use of mobile communication networks has increased significantly in the past decades. The proliferation of smart devices (e.g. data providers) and the resulting exponential growth in data traffic has increased the need for higher capacity wireless networks. In addition, new paradigms are emerging, like Internet of Things (IoT) and Internet of Everything (IoE). With these paradigms, billions of data providers will be connected to the Internet in next years. The attention is now shifting toward the next set of innovations in architecture, technologies, and systems that will address the capacity and service demands envisioned for this evolutionary wave. These innovations are expected to form the so called fifth generation of communications systems.

Can be identified through literature that there are significant amount of systems for data management related to IoE, sensor networks, and pervasive/ubiquitous computing. However, unless the system can analyze, interpret, and understand these data, it will keep useless and without meaning for the users and applications. The context is used to give meaning to these data. A context-aware feature is required to the systems in order to address this challenge.

As can be seen during this chapter, there are some systems with different architectures that have context-aware features, thus enabling a sensing-as-a-service platform. The system features can vary in different ways, in addition to the modules that compose it. On the other hand, all the systems may follow the four phases of context life-cycle (acquisition, modelling, reasoning, and distribution) in order to produce context information.

The new requirements imposed by IoE will drive to new context-aware challenges. The systems will aim to produce context in the most efficient way. Moreover, there are many challenges involving the process as well, like: automated configuration of data providers, context discovery, context life-cycle phases, selection of data providers, security issues, scalability, reliability, context sharing and interoperability. These challenges will force new directions to the context-aware systems of the future IoE environments.

Acknowledgments Our thanks to CAPES/CNPq for the funding within the scope of the project number 384843/2015-8.

References

1. Abdelwahab, S., Hamdaoui, B., Guizani, M., Rayes, A.: Enabling smart cloud services through remote sensing: An internet of everything enabler. Internet of Things Journal, IEEE 1 (3), 276–288 (2014)
2. Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing, pp. 304–307. Springer (1999). URL http://dl.acm.org/citation.cfm?id=647985.743843
3. Atzori, L., Iera, A., Morabito, G.: The internet of things: A survey. Computer Networks 54(15), 2787–2805 (2010). doi:http://dx.doi.org/10.1016/j.comnet.2010.05.010
4. Badii, A., Crouch, M., Lallah, C.: A context-awareness framework for intelligent networked embedded systems. In: Advances in Human-Oriented and Personalized Mechanisms, Technologies and Services (CENTRIC), 2010 Third International Conference on, pp. 105–110. IEEE (2010)
5. Bandyopadhyay, D., Sen, J.: Internet of things: Applications and challenges in technology and standardization. Wireless Personal Communications 58(1), 49–69 (2011). doi:10.1007/s11277-011-0288-5. URL http://dx.doi.org/10.1007/s11277-011-0288-5
6. Batalla, J.M., Mastorakis, G., Mavromoustakis, C.X., Z˙ urek, J.: On cohabitating networking technologies with common wireless access for home automation systems purposes. In: IEEE Wireless Communications Magazine (To be published). IEEE (2016)
7. Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., Riboni, D.: A survey of context modelling and reasoning techniques. Pervasive and Mobile Computing 6(2), 161–180 (2010)
8. Bikakis, A., Patkos, T., Antoniou, G., Plexousakis, D.: A survey of semantics-based approaches for context reasoning in ambient intelligence. In: Constructing ambient intelligence, pp. 14– 23. Springer (2008)
9. Capra, L., Emmerich, W., Mascolo, C.: Carisma: Context-aware reflective middleware system for mobile applications. Software Engineering, IEEE Transactions on 29(10), 929–945 (2003)
10. Chellouche, S.A., Négru, D., Arnaud, J., Batalla, J.M.: Context-aware multimedia services provisioning in future internet using ontology and rules. In: Network of the Future (NOF), 2014 International Conference and Workshop on the, pp. 1–5 (2014). doi:10.1109/NOF.2014.7119778

11. Chen, G., Kotz, D., et al.: A survey of context-aware mobile computing research. Tech. rep., Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College (2000)
12. Chen, H., Finin, T., Joshi, A., Kagal, L., Perich, F., Chakraborty, D.: Intelligent agents meet the semantic web in smart spaces. Internet Computing, IEEE 8(6), 69–79 (2004). doi:10.1109/MIC.2004.66
13. Clarke, R.Y.: Smart cities and the internet of everything: The foundation for delivering nextgeneration citizen services. Alexandria, VA, Tech. Rep (2013)
14. Conan, D., Rouvoy, R., Seinturier, L.: Scalable processing of context information with cosmos. In: Distributed Applications and Interoperable Systems, pp. 210–224. Springer (2007)
15. Corradi, A., Fanelli, M., Foschini, L.: Implementing a scalable context-aware middleware. In: Computers and Communications, 2009. ISCC 2009. IEEE Symposium on, pp. 868–874. IEEE (2009)
16. Dey, A.K., Abowd, G.D., Salber, D.: A context-based infrastructure for smart environments. In: Managing Interactions in Smart Environments, pp. 114–128. Springer (2000)
17. Dohr, A., Modre-Opsrian, R., Drobics, M., Hayn, D., Schreier, G.: The internet of things for ambient assisted living. In: Information Technology: New Generations (ITNG), 2010 Seventh International Conference on, pp. 804–809. Ieee (2010)
18. Evans, D.: The internet of things: How the next evolution of the internet is changing everything. CISCO white paper 1, 14 (2011)
19. Filho, J., Agoulmine, N.: A quality-aware approach for resolving context conflicts in contextaware systems. In: Embedded and Ubiquitous Computing (EUC), 2011 IFIP 9th International Conference on, pp. 229–236 (2011). doi:10.1109/EUC.2011.9
20. Froehlich, J., Neumann, J., Oliver, N.: Measuring the pulse of the city through shared bicycle programs. Proc. of UrbanSense08 pp. 16–20 (2008)
21. Giusto, D., Iera, A., Morabito, G.: The Internet of Things. Springer (2010)
22. Guan, D., Yuan, W., Lee, S., Lee, Y.K.: Context selection and reasoning in ubiquitous computing. In: Intelligent Pervasive Computing, 2007. IPC. The 2007 International Conference on, pp. 184–187 (2007). doi:10.1109/IPC.2007.102
23. Guo, B., Sun, L., Zhang, D.: The architecture design of a cross-domain context management system. In: Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on, pp. 499–504. IEEE (2010)
24. Hall, D., Llinas, J.: An introduction to multisensor data fusion. Proceedings of the IEEE 85(1), 6–23 (1997). doi:10.1109/5.554205
25. Huaifeng, Q., Xingshe, Z.: Context aware sensornet. In: Proceedings of the 3rd International Workshop on Middleware for Pervasive and Ad-hoc Computing, MPAC'05, pp. 1–7. ACM, New York, NY, USA (2005). doi:10.1145/1101480.1101489. URL http://doi.acm.org/10.1145/1101480.1101489
26. Indulska, J., Sutton, P.: Location management in pervasive systems. In: Proceedings of the Australasian Information Security Workshop Conference on ACSW Frontiers 2003—Volume 21, ACSW Frontiers'03, pp. 143–151. Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2003). URL http://dl.acm.org/citation.cfm?id=827987.828003
27. Institutes, C.: Smart networked objects and internet of things. Carnot Institutes' Information Communication Technologies and Micro Nano Technologies alliance, White Paper (2011)
28. Jara, A.J., Ladid, L., Skarmeta, A.: The internet of everything through ipv6: An analysis of challenges, solutions and opportunities. J. Wirel. Mob. Netw. Ubiq. Comput. Dependable Appl 4, 97–118 (2013)
29. jboss.org: Drools—the business logic integration platformn. http://www.jboss.org/drools (2001). Accessed: 2015-05-15
30. Kortuem, G., Kawsar, F., Fitton, D., Sundramoorthy, V.: Smart objects as building blocks for the internet of things. Internet Computing, IEEE 14(1), 44–51 (2010)

31. Kryftis, Y., Mavromoustakis, C.X., Mastorakis, G., Pallis, E., Batalla, J.M., Rodrigues, J.J.P. C., Dobre, C., Kormentzas, G.: Resource usage prediction algorithms for optimal selection of multimedia content delivery methods. In: 2015 IEEE International Conference on Communications (ICC), pp. 5903–5909 (2015). doi:10.1109/ICC.2015.7249263
32. Mahoney, J., LeHong, H.: Innovation insight: the 'internet of everything'innovation will transform business. Gartner Online, January 3 (2012)
33. Martín, D., Lamsfus, C., Alzua, A.: Automatic context data life cycle management framework. In: Pervasive Computing and Applications (ICPCA), 2010 5th International Conference on, pp. 330–335. IEEE (2010)
34. Matos, E., Amaral, L., Tiburski, R., Lunardi, W., Hessel, F., Marczak, S.: Context-aware system for information services provision in the internet of things. In: Emerging Technologies & Factory Automation, 2015. ETFA 2015. IEEE Conference on, pp. 1–4. IEEE (2015)
35. Nurmi, P., Floree, P.: Reasoning in context-aware systems. position paper. In: Department of Computer Science, University of Helsinki (2004)
36. Ormfoundation.org: The orm foundation (1989). URL http://www.ormfoundation.org
37. Patel, P., Jardosh, S., Chaudhary, S., Ranjan, P.: Context aware middleware architecture for wireless sensor network. In: Services Computing, 2009. SCC'09. IEEE International Conference on, pp. 532–535. IEEE (2009)
38. Pepper, R., Garrity, J.: The internet of everything: How the network unleashes the benefits of big data. The global information technology report pp. 35–42 (2014)
39. Perera, C., Zaslavsky, A., Christen, P., Georgakopoulos, D.: Context aware computing for the internet of things: A survey. Communications Surveys Tutorials, IEEE 16(1), 414–454 (2014). doi:10.1109/SURV.2013.042313.00197
40. Pietschmann, S., Mitschick, A., Winkler, R., Meissner, K.: Croco: Ontology-based, crossapplication context management. In: Semantic Media Adaptation and Personalization, 2008. SMAP'08. Third International Workshop on, pp. 88–93 (2008). doi:10.1109/SMAP.2008.10
41. Reetz, E.S., Tonjes, R., Baker, N.: Towards global smart spaces: Merge wireless sensor networks into context-aware systems. In: Wireless Pervasive Computing (ISWPC), 2010 5th IEEE International Symposium on, pp. 337–342. IEEE (2010)
42. Román, M., Hess, C., Cerqueira, R., Ranganathan, A., Campbell, R.H., Nahrstedt, K.: A middleware infrastructure for active spaces. IEEE pervasive computing 1(4), 74–83 (2002)
43. Shelby, Z., Hartke, K., Bormann, C.: The constrained application protocol (coap) (2014)
44. Strang, T., Linnhoff-Popien, C.: A context modeling survey. In: Workshop Proceedings (2004)
45. Uml.org: Unified modeling language (uml) (2012). URL http://www.uml.org/
46. Wibisono, W., Zaslavsky, A., Ling, S.: Comihoc: A middleware framework for context management in manet environment. In: Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, pp. 620–627. IEEE (2010)
47. Zaslavsky, A., Perera, C., Georgakopoulos, D.: Sensing as a service and big data. arXiv preprint arXiv:1301.0159 (2013)
48. Zhang, W., Hansen, K.M.: Towards self-managed pervasive middleware using owl/swrl ontologies. In: Fifth International Workshop on Modelling and Reasoning in Context. MRC 2008 (2008)
49. Zieliński, J.S.: Internet of Everything (IoE) in smart grid. Przegląd Elektrotechniczny 91(3), 157–159 (2015)