Pervasive and Mobile Computing 52 (2019) 71–99

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

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

Review

A survey on fog computing for the Internet of Things

Paolo Bellavista [a,*], Javier Berrocal [b], Antonio Corradi [a], Sajal K. Das [c], Luca Foschini [a], Alessandro Zanni [a]

[a] Dept. Computer Science and Engineering (DISI), University of Bologna, Viale del Risorgimento, 2 – 40136 Bologna, Italy
[b] Dept. Ingeneria Sistemas Informaticos y Telematicos, University of Extremadura, Av. de la Universidad s/n. 10003, Cáceres, Spain
[c] Dept. Computer Science, Missouri University of Science and Technology, 500 W. 15th Street, Rolla, MO 65409, USA

ARTICLE INFO

Article history:
Received 1 March 2018
Received in revised form 8 September 2018
Accepted 19 December 2018
Available online 21 December 2018

Keywords:
Fog computing
Internet of Things
IoT case studies
Survey
Taxonomy

ABSTRACT

Fog computing has emerged to support the requirements of IoT applications that could not be met by today's solutions. Different initiatives have been presented to drive the development of fog, and much work has been done to improve certain aspects. However, an in-depth analysis of the different solutions, detailing how they can be integrated and applied to meet specific requirements, is still required. In this work, we present a unified architectural model and a new taxonomy, by comparing a large number of solutions. Finally, we draw some conclusions and guidelines for the development of IoT applications based on fog.

Contents

[*] Corresponding author.
E-mail addresses: paolo.bellavista@unibo.it (P. Bellavista), jberolm@unex.es (J. Berrocal), antonio.corradi@unibo.it (A. Corradi), sdas@mst.edu (S.K. Das), luca.foschini@unex.es (L. Foschini), alessandro.zanni@unibo.it (A. Zanni).

https://doi.org/10.1016/j.pmcj.2018.12.007
1574-1192/© 2018 Elsevier B.V. All rights reserved.

1. Introduction

During the last few years we have lived a revolution in how people communicate, interact, work, etc. This revolution has been caused by two key technologies: the smartphone and the cloud computing. The smartphone has erected as the device preferred by people to interact with the Internet, having a penetration rate of 97% [1,2]. Much of this success has been sustained by the use of cloud environments, reducing the computing and storage load required for these devices. This can be seen in the 18-fold growth of global mobile data traffic over the past 5 years [3]. This architecture in two layers (composed by the final devices and the cloud environment), has allowed an unprecedented development of these technologies.

In parallel, we have witnessed the development and diffusion of the Internet of Things (IoT). The IoT concept is changing the way people interact with the physical world, introducing an explosion of connectivity towards it. IoT refers to the deployment of multiple interconnected smart devices supporting everyday task. IoT will introduce new applications with limitless potentiality and massive impact by enabling mass participation of users and, in particular, boosting machines and sensors/actuators communications. It has been predicted that by 2020 there will be 50 to 100 billion of these devices connected to the Internet [4].

The expected huge number of interconnected devices and the significant amount of available data open new opportunities to create services that will bring tangible benefits to the society, but also poses important challenges [5,6]. If we analyse the requirements and behaviour of the IoT applications, we can see that a two-layer architecture (cloud-IoT devices) can hardly support all the communication and data processing required by all these billions of connected devices. If we get it to support them, the scalability, latency, and response time would be very limited. Usually, IoT applications have stringent requirements. Most of them require almost real-time responsiveness while, at the same time, the Quality of Service (QoS), the security and privacy and the location-awareness of the response have to be achieved [7]. A two-tier architecture, with a myriad of devices sensing and sending the gathered data to the cloud to be processed in order to identify how the system should acts would hardly meet the requirements of these systems [8,9]. An effective and efficient integration between IoT and the cloud is challenging but can contribute to increase the overall efficiency.

Different solutions and architectures are proposed to support the processing of all these data and/or the requirements of IoT applications. [10,11] detail new architectures that allow the integration of any lightweight sensors with the cloud, by overcoming typical cloud issues like latency, management of continuous sensing, the ability to support periodic events and the lack of elasticity when numerous wireless sensors transmit data simultaneously. [12] addresses the problems derived from continuous sensing, that raise many challenges with cloud iterations, by proposing that devices should collect data and only sporadically upload them to the cloud but, in this way, this delay-tolerant model of sensor sampling and processing severely limits applications effectiveness and the ability of the system to be aware of its context, adapt and react to situations.

Several research activities propose to increase the number of layers in the architecture of solution in order to carry out part of the computation and storage of the data in intermediate layers [13], thus reducing the data traffic overload, the response time and the response location awareness. Emerging technologies, such as 5G wireless systems will provide high flexibility, low-latency, and high-capacity in order to support the forecasted growth in mobile data traffic [14], but they will also need a global orchestration for the distributed implementation and the management of heterogeneous networks [15,16]. Fog computing, or more shortly fog in the following, is a relatively new concept and already popular term that tries to satisfy the requirements of applications that deal with device ubiquity. Fog can be defined as a programming and communication paradigm that brings the cloud resources closer to the IoT devices, physically and/or computationally. In other words, fog acts as the interface between cloud and IoT, helping them to communicate. Therefore, it gets the best from each technology, extending the application field of cloud computing and increasing the resource availability in IoT.

Nevertheless, fog computing is a paradigm that is still in its infancy. Currently, there are a large number of works focused on improving certain areas or characteristics, such as communication between different devices or among fog nodes [17,18], the security and the privacy of both the stored and the exchanged data [19,20], reducing the size of the transmitted data [21,22] or where and how the information should be processed to meet the responsiveness required by IoT applications [23,24]. In addition, in many cases these solutions must be combined and integrated so that they can be applied in specific environments. Therefore, proposals evaluating the different solutions, how they can be integrated, and proposing some guidelines on how the different solutions can be applied are needed.

Indeed, although there are different surveys on fog computing, most of them are focused on specific characteristics (such as security and privacy [25] or communications among devices [26]) and do not provide a holistic view of fog computing. In [27], different papers are grouped in order to discuss various aspects of fog computing, but again only some specific characteristics are discussed. Other works, such as [28–32], analyse a greater number of works contributing to different areas of fog, but with the main objective of defining addressing a general fog scenario rather than the specific requirements of the IoT scenario. This survey aims to fill the gap proposing a survey of fog computing solutions specifically designed to serve IoT use cases (i.e., *fog for IoT*). In fact, a clear formulation of fog computing for IoT requirements and its core components is still missing due to several reasons. Fog requirements are very dependent on the final IoT applications and most of the surveyed solutions focus on specific requirements of a concrete IoT application only. We claim, instead, that a fog solution should be able to adapt to different IoT applications with different requirements, complexity, and architectural needs.

Differently from surveys already existing in literature, our work analyses some of the most important proposals in fog computing for IoT detailed so far, trying to select those covering the most relevant requirements for both fog computing and IoT worlds (i.e., demanding environments in response time, information gathered, computing requirements, etc.). In particular, our survey addresses this challenging area by proposing three main novelty aspects: (i) a unified architectural model for fog computing, (ii) a new taxonomy to settle the terminology and concepts useful to compare existing solutions in literature, and (iii) a thorough comparison of the surveyed solution together with some open issues and guidelines on how to integrate them for meeting specific requirements.

The paper is organized in sections as follows. Section 2 motivates this work and reports related work and the methodology followed to identify and analyse the surveyed approaches. Section 3 details fog computing, its main requirements, and how different IoT case studies can take advantage of it. Section 4 presents an architectural model for fog for IoT to meet the detailed requirements. Section 5 proposes our original taxonomy of fog for IoT. In Section 6, we present a comparison of the surveyed solutions and we briefly report about additional research directions. Finally, Section 7 contains some concluding remarks.

2. Motivations, related work, and methodology

In this section, first, we explain the motivations that lead to the introduction of a fog computing layer. Then, we provide a description of what fog computing is and the improvements it introduces in the system. Subsequently, we identify the main requirements for the fog computing solutions. Finally, we detail some IoT case studies and how they can take advantage of a fog layer.

2.1. Motivations

The integration of cloud in IoT applications is double-faced and not easy to manage, bringing substantial advantages to both providers and end users on one side, but raising new unsuitableness in the integration with ubiquitous services on the other side. Although cloud can import huge improvements in the system processes with its great amount of resources availability, direct exploitation of cloud resources by ubiquitous IoT devices may introduce several technical challenges and inefficiencies, such as network latency, traffic and communication overhead to the devices, and further costs. Connecting a myriad of sensors directly to the cloud is extremely demanding on cloud resources. The result is that the cloud remains busy per each sensor duty cycle. In addition, the bandwidth cannot support this data load.

Future internet applications, which are rising from the development of IoT environment, are large-scale, latency-sensitive and are no longer created to work alone but to share infrastructure and resources. Those applications require new requirements to be satisfied, like mobility support, large-scale geographic distribution, location awareness and low latency [7]. As a general consideration, it is widely recognized that an architectural model only based on direct interconnection between IoT devices and the cloud is not appropriate for some IoT applications [7,28]. A distributed intelligent intermediate layer is required to add extra functionalities to the system, doing some processing of data when devices gather them and before sending them to the network and, eventually, to the cloud. There are some work proposing to move part of the resources towards the network edge, such as Cloudlet [33], Edge Computing [34], and Follow-Me Cloud [35]. Cloudlet efforts propose to create a cluster of servers near endpoints in order to satisfy real-time and location-awareness requirements. Cloudlet is based on a three-tiers hierarchy (mobile devices, Cloudlet, and cloud) and is completely transparent under normal conditions, giving mobile users the illusion that they are directly interacting with the cloud. Edge Computing aims to move applications, data, and services from cloud towards the edge of the network. Concretely, [36] introduces the concept of Edge-as-a-Service (EaaS), a concept that decouples the strict ownership relationship between network operators and their network infrastructure.

The Fog vision was conceived to address applications and services that do not fit well the paradigm of the cloud. Fog tries to put itself between IoT and cloud, taking the main benefits of both. Currently, there are some standardization efforts trying to improve the interoperability of different proposals and reference architectures in fog and edge computing areas, such as Open Edge Computing [37], OpenFog Computing [38] and Mobile Edge Computing [39]. The Open Edge Computing Consortium is a joint initiative between industry and academia to drive the development of the ecosystem around Edge Computing by providing open and globally standardized mechanisms. The OpenFog Computing Consortium main objective is to define standards to ease the implementation and interoperability of IoT applications: they are working on an architecture emphasizing information processing and intelligence at the logical edge of the network. Mobile Edge Computing (MEC) is a reference architecture and a standardization effort by the European Telecommunication Standards Institute (ETSI). MEC provides an IT service environment and cloud-computing capabilities at the edge of the mobile network. Fog computing and Edge computing are close concepts [40]. Fog includes all the devices from the cloud to the end devices and has a substantial overlap with Edge Computing [41]. Nevertheless, as some authors indicate, edge computing focus more towards the things side, while fog computing focus more on the infrastructure side [42,43] .

Therefore, there are different standards and approaches developing different parts of the fog vision or, even, complete fog architecture definitions. Nevertheless, most of them are focused on specific requirements or concrete applications. This paper aims to propose a unified architecture supporting the IoT applications' requirements and a taxonomy for comparing the different proposal, detailing also some guidelines for the development of a fog platform and some future research trends. From a terminological perspective, to avoid possible confusion due to the different naming conventions in the different standards, in the following we decided to adopt the definitions proposed by OpenFog because it is the driving standardization body in the fog area and the one closest in vision and definition to our effort. Hence, as better detailed in the next section, we foresee: (i) at the top, a *cloud* layer including public/private cloud resources and data centres; (ii) at the bottom, an *edge* layer including sensor/actuator and network edge devices; and (iii) in the middle between those two layers, a *fog* layer meant itself as an N-tier deployment of intermediate levels/nodes, defined as *fog nodes* [38].

2.2. Related work

The relevance of fog computing and its support to IoT is proved by the evidence at several solutions such as the ones analysed in this paper, and some seminal survey activities have been conducted in the last 2–3 years to accommodate fog computing concepts and the related challenges [25–31]. Differently from surveys already existing in literature, our work focuses on fog for IoT to deeply study main requirements, implementation primitives, and identify research challenges.

Following an order of increasing similarity with our work, [25] and [26] represent seminal efforts that address very specific issues in fog computing, namely security and communications among devices; however, just few systems are considered, and fog for IoT issues are not specifically tackled. [28] discusses some of the challenges in IoT scenarios and demonstrate that fog computing is a promising enabler for IoT applications, but the specific focus of this work is on supporting big data scenarios rather than on the specific IoT aspects. [29] and [40] are also seminal works proposing a first definition of fog computing and of related concepts, also introducing some application scenarios and challenges ahead; however, these works are not full-fledged survey papers, but rather present an first survey part, followed by a second part more oriented on novel research. [27] is a collection of papers that discuss various aspects of fog computing with IoT, but again it only hits some specific characteristics without an idea of providing a general view of main issues and/or a reference conceptual architecture.

Let us conclude with the most recent works, also closer to ours in terms of larger coverage of solutions and goal of the analysis. [31] derives from the literature main requirements, key technologies, and characteristics of fog computing trying to distil the main ingredients of a general conceptual architecture, similarly to what we do in Sections 3 and 4; however, the coverage of the literature is more limited than ours in terms of number of surveyed works and, most important, it lacks a thorough comparison of works according to a neatly defined taxonomy (see Sections 5 and 6). [30] is a very recent and comprehensive work that introduces well-defined and well-motivated criteria and provides an exhaustive literature review. On the one hand, it focuses on fog computing and introduces some main *criteria* for two main areas, that are the same abstraction level of the *requirements* we introduce in Section 3.2, and then it provides a very interesting comparison. On the hand, differently from our goal, it does not propose a taxonomy for a specific area (i.e., fog for IoT), but rather uses those general-purpose criteria to analyse a very large collection of papers divided in two very broad areas, namely, architectures and algorithms for fog systems.

2.3. Methodology

Let us conclude this section by presenting the main steps performed to identify and analyse all considered approaches and efforts considered in the remainder of this survey:

- **Step 1. Systematic Mapping Study (SMS)**. A SMS was done to identify the most important approaches contributing or developing a part of the fog paradigm in any of the above defined case studies. From this step, the most important contributions were categorized depending on the domains in which they were applied or if they were general approaches.
  To identify those contributions, different queries were executed in different digital libraries (Scopus (https://www.scopus.com), IEEE Xplore (http://ieeexplore.ieee.org/) and Google Scholar (https://scholar.google.com/)) using different combinations of the following key words "Fog", "IoT", "Smart Traffic Light", "Wind Farm" and "Smart Grid". From the obtained results those that were relevant, provided a new technique or used a technique in a specific domain were studied.
- **Step 2. Conceptual Architecture Design**. The papers identified in the Step 1 were analysed in order to identify the different components and features that were used in the proposals and how they were combined. The rationale behind each component and their interactions was analysed, identifying the components overlapping the different case studies and those that are complementary. From that analysis, the conceptual architecture was conceived.
- **Step 3. Taxonomy.** A taxonomy where each identified feature and component is deeply defined and detailed was created. For this taxonomy, the different techniques that are normally used and were analysed in the surveyed approaches were detailed. For this step, adopting the snowball technique, the execution of more restrictive queries was also used to identify specific approaches for concrete features of the taxonomy applied to specific case studies. The definition of the taxonomy allowed us to establish a set of concepts to better compare each analysed approach.
- **Step 4. Approaches vs Taxonomy**. Finally, based on the taxonomy, a table was generated detailing the approaches contributing to each feature in the taxonomy, identifying also the specific techniques used. This allowed us to identify how the different features are combined and to provide some guidelines on how to integrate them.

3. Deployment model, requirements, and case studies

3.1. Fog for IoT deployment model

Fog Computing refers to a distributed computing paradigm that moves storage and computation usually near the end nodes of the network with the purpose of reducing the network overload and compute the gathered information as soon as possible. So that, the response time and the system performance improve. In addition, fog computing can be profitably introduced to let IoT applications interwork efficiently with cloud resources, acting as an intermediation layer (or set of layers) between the cloud and the edge layers. Therefore, fog can be considered as a significant extension of the cloud computing concept, capable of providing virtualized computation and storage resources and services with the essential difference of the distance from utilizing end-points.

A primary idea emerging from existing fog solutions in the literature is to deploy a common platform supporting a wide range of different applications and, the same support platform with multi-tenancy features, can be used also by a multiplicity of client organizations that anyway should perceive their resources as dedicated, without mutual interference [28].

Adopting the OpenFog deployment model view [38], we consider the *fog layer* as a (potentially) complex deployment consist of multiple (hierarchically organized and coordinated) nodes in between a top *cloud* layer and a bottom *edge* layer. Some of these fog nodes are closer to the end nodes (with lower computing and storage resources, but higher responsiveness) and other (with an increasing computing and storage resources and lower responsiveness) closer to the cloud (see also Fig. 1). This is especially true for fog for IoT applications that typically require different levels of fog and cloud processing at the same time: fog to support different levels of real-time processing and actions, and cloud to store long-term data and to perform long-term analysis. After the advent of the fog concept, at the current stage a lot of work is ongoing to clearly define a deployment model and what capabilities each layer should has, how the IoT requirements are meet, how the workload of each layer/node is balanced, and so on. Moreover, this has also been done on purpose so to leave developers the maximum possible freedom in the design of fog-based applications [38]. In fact, [38] shows a high-level architecture that summarizes

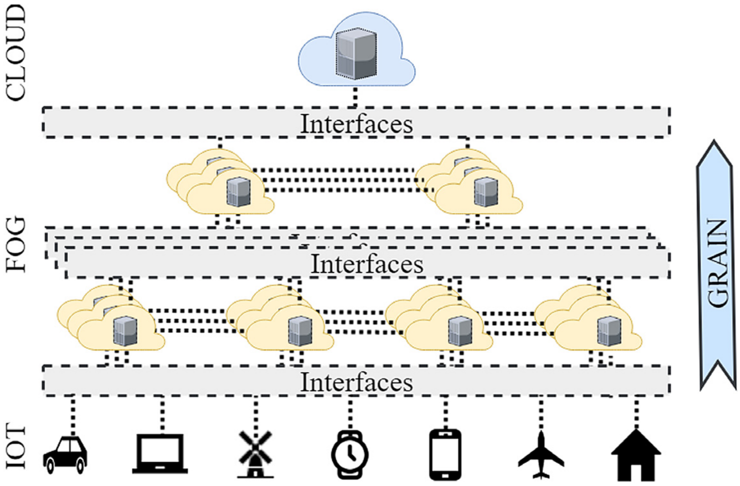

Fig. 1. Cloud-Fog-Edge architecture.

the above vision by positioning the cloud, edge, and fog layers according to the terminology adopted in the remainder of the paper. Let us also underline that although the bottom edge layer could, in general, include any kind of end user device, since we are interested to fog for IoT solutions, in the remainder of the paper (unless specified differently) this layer populated mainly by IoT devices.

3.2. IoT requirements for fog computing

IoT systems present some key requirements that have to be accomplished by the fog and cloud computing environments in order to achieve a correct operation and the user satisfaction. This section briefly details these features and clarify their definitions by motivating their choice. We identified this list of requirements starting from existing standardization proposals and surveys, such as [30–32,38,40]. Then we refined it for our specific fog for IoT scenario by considering different case studies and IoT applications from different domains that have very restricted requirements, requiring the deployment of the Fog computing paradigm to achieve them. Because of that, the advances in these domains have determined the development of a specific research area (that we call fog for IoT) in the development of the fog architecture [28,44].

In particular, we considered some main IoT application domain that range from Smart Traffic Light to Smart Grid, that have been selected in order to identify the most important requirements for all of them, since they present different perspectives and have different needs as better detailed in Section 3.3. With this, a broader vision and taxonomy that covers a greater number of situations have been obtained so to obtain a unified architectural model from the fog for IoT literature.

3.2.1. Scalability

Scalability is a very important requirement connected with both Big Data and the geo-distribution of devices; it is also one of the 8 pillars of the OpenFog reference architecture and recognized among core evaluation criteria by other surveys in the area [30,32,38]. As regard fog for IoT, first, as the number of devices connected to the system increases, it also growths the volume of data to be gathered and processed. Usually, Big Data and geo-distribution are related properties, since obviously a system with a wide-range and a dense distribution generates more data [45]. Big Data scalability is a basic requirement of IoT applications where a growing number of devices has to be connected and the main goal of Big Data approaches to be effectively valuable in IoT environments. Therefore, here we highlight the characteristic of the system to scale in relation to the quantity of information managed. Geo-distribution scalability is a requirement that underlines the property of fog computing to be able to manage distributed services and applications, even highly distributed systems, in contrast with the more centralized cloud. In highly distributed systems, fog must handle large number of nodes widespread in geographic areas, also with varying degrees of density. Therefore, fog for IoT must handle different type of topology and distributed configurations system and, thus, be able to scale and adapt in order to meet the requirements for each system.

3.2.2. Interoperability

Interoperability is one of the main issues in fog computing as widely acknowledged in related standards and surveys [30,31,38]. Focusing on fog for IoT, IoT is by its own nature an extremely heterogeneous environment that operate in real world scenarios, based on a wide range of different devices that collect heterogeneous information from the environment. Sensors range can be various, from wimpy to powerful sensors, with orders of magnitude of differences, in terms of power consumption, data rate, resources availability, etc. Similarly, fog nodes are heterogeneous in nature [28]. For instance, fog nodes range from high-end servers, edge routers, access points, set-top boxes and, even, end devices such as vehicles, sensors, mobile phones, etc., with very different range of resource availabilities. Moreover, inside fog computing sometimes services must be federated because they require the cooperation of different providers [7]. Therefore, fog computing is a highly virtualized platform that needs heterogeneous devices and their running services to be handled in a homogeneous manner, ideally fully automated by software, towards a common goal. In complex systems, heterogeneity can affect both technical and semantic interoperability. Technical interoperability concerns communication standards, elements implementations or components interfaces. Semantic interoperability concerns the information inside data exchanged and the possibility that two elements understand and share the same information differently. A standardized way to describe and exchange information, together with an abstraction layer that hides physical differences among elements, is required to create interoperability [46]. Fog environments could be a suitable place to perform all processes to enable interoperability, in order to create a unique data stream, to expose generic APIs that can be used by diverse applications or an initially unique federation of services, without expensive computations on a Cloud layer.

3.2.3. Real-time responsiveness

Real-time responsiveness (low-latency and real-time interactions) is also a main enabler for IoT applications and their deployment in real-world scenario [31,32]. Real-time interactions force to process continuously fresh data. Nevertheless, sensors are constantly gathering huge amount of data from the environment, usually at high data rates. Therefore, in highly dynamic and real-time scenarios data change very quickly and the exchanged data from IoT to cloud might not be accurate because of the high latency during interactions. Fog Computing is crucial to achieve low-latency requirement because direct interactions Cloud-Edge are not able to be satisfied for multiple reason: (i) fog overtakes distance issues, by moving computation near the edge and decreasing the numbers of network hops; (ii) fog improves temporal accuracy because it senses information, processes it and acts at real-time. Thus, it always uses data that reflect exactly the present situation; (iii) sensors gather huge amount of data from the environment, so, if the whole amount of data is sent to the Cloud, the network may sensibly slow down due traffic congestion and, as consequence, all the system will slow down. For many tasks, which do not require long analysis or high resource consumption, fog should provide real-time execution. In addition, for the former systems, fog nodes should pre-process the information, before uploading it to the cloud, reducing the core network load.

3.2.4. Data quality

Data quality is a relevant requirement in real world IoT applications due to their characteristic of not only sensing, but also acting on scenarios by modifying the physical world, usually in an irreversible mode. This requirement is typically not present in related standards and surveys on general-purpose fog computing, although some authors describe this requirement as part of heterogeneity management [31]. Restricting the analysis on fog for IoT, we deem data quality as a crucial requirement to enable the melting and integrated use of data coming from highly heterogeneous and different IoT sensors/actuators. In fact, increasing the system data quality can lead to relevant improvement during computation and actuation phases and, thus, to a better overall quality of the system. Moreover, differentiate rich-information data, to keep for further processing, from faulty or noisy data without significant information during initial steps, and get rid of those useless data, can decrease the quantity of data and, thus, speed-up the system performance. Data quality is based on the union of different techniques, such as: data filtering, data aggregation, data normalization, etc. The combination of data filtering, data aggregation, data normalization and data analytics is used to perform proactive maintenance and anomaly detection in real time. In ubiquitous environment, faulty data is one of the most serious problem, because it is difficult to discover and affects both performances and reliabilities of the systems. Fog nodes should support data quality also to eliminate the useless data as soon as possible in order to decrease the amount of data to be processed or to be pushed to the cloud.

3.2.5. Security

Security is generally recognized as a crucial cornerstone requirement of any fog computing system [25,31,38]. In fog for IoT, the number of IoT devices interconnected increases the complexity to operate them not only security, but also to obtain safety. This increased complexity requires standards and solutions providing safety, security and privacy, by considering those aspects as tightly interconnected [47]. Safety is a basic requirement since fog for IoT is used in real world applications that act in critical contexts, hence, the presence of unexpected behaviour must be minimized. Security is a key issue that must be faced to support industrial deployments and it concerns the whole systems architecture from IoT devices to cloud. A rich set of security features that enables basic security for each circumstance for the whole system is required to avoid having to implement specific security mechanisms for every node. In addition, privacy is an increasingly concerned issue that is expanding with ubiquitous and pervasive systems and users are becoming more and more sensitive with their personal data. In fog, personal data are not centralized in few components but are distributed in the network. It is important to define the ownership of the data inside the fog because applications must use only data they have access to [48]. One primary challenge in ubiquitous environments and fog computing is to balance security and personal data control with the possibility to access data to provide better services.

3.2.6. Location-awareness

IoT applications should be able to have a widespread knowledge of their location and to understand the external context where they are immersed in. In general, this requirement is acknowledged by some works as a core requirement, such as the hierarchy pillar of OpenFog [38] and the geographical distribution requirement in [31]. In fog for IoT solutions, location-aware supports can strengthen IoT applications, creating systems with a higher degree of consciousness and, thus, a higher degree of resilience to the outside world. In this way, a system can understand if there is an unexpected behaviour or attacks by external agents and react efficiently. Location-awareness leads to enhance the knowledge of the sensed environment towards better adaptability of the system, accuracy of the response and, thus, improvement on its execution and higher quality of applications. Accuracy is provided because the system knows the environment where it is working and, consequently, its responses are more precise and applicable than systems that, vice versa, are not familiar with the information sensed in the scenario and the environment where they are located. Fog should improve adaptability adjusting its behaviour in relation of different events. Therefore, fog nodes should be able to identify the location of the deployed applications and take advantage of this information to improve the data processing and adaptation modules.

3.2.7. Mobility

Mobility is increasingly accepted, especially in more recent research-oriented literature, as a core requirement of fog computing [30–32]. As regards fog for IoT, IoT applications are directly related with mobility and Mobile Internet of Things (MIoT) expands the IoT concept with mobile support and ubiquitous coverage [49,50]. This paradigm has gained a central position due to the massive growth of mobile devices and their generated data while-on-the-move. MIoT is demonstrating to be a technically challenging playground for distributed supports capable of sustaining the execution and run-time requirements of advanced dynamic applications [49,50]. The growing ubiquity of mobile devices and the predominant role of wireless access raise the necessity to introduce in fog computing mobility support. In order to be effective, fog nodes have to adapt themselves to manage high mobility devices. The system knowledge must move around, in particular in data rich mobility applications. If we are able to locate the right data in the fog, we can obtain a better performance, better data models and local caching. Moreover, fog computing has to support the possibility that mobile devices can shift from a fog node authority to another without interrupting system operations or causing any problems.

3.2.8. Reliability

In fog computing, a large number of devices perform in a distributed manner different activities and tasks, hence reliability is recognized as an important requirement. Along this direction, OpenFog introduces the "reliability, availability, and serviceability pillar" as a very broad umbrella including several different issues [38]; other authors consider reliability as a part of the QoS management, typically treating it as part of network/system specifications [30,32]. In the context of fog for IoT, closer to the OpenFog definition, reliability is an essential requirement to be provided at the different layers and spanning different perspectives (see also Section 4): (a) the hardware must be reliable and operate as expected (for example, a sensor providing the expected measurements with the frequency set); (b) communication between all elements of the network must be reliable, supporting data transport and message exchanging ; (c) the different fog nodes have to produce the expected output (processing the data or identifying the action to be performed); and (d) the management of the data centre, the scheduling policy, and the power-consumption model should be reliable [13,51]. In brief, fog for IoT should be reliable, considering the failure of any individual fog node, the failure of the whole network, the failure of the service platform, the failure of the user's interface connected to system etc. Different techniques should be applied to achieve and assure that reliability.

3.3. IoT case studies

Let us conclude this section focusing our main reference IoT application domains through the introduction of considered case studies. Currently, there are several different IoT case studies widely used in the literature to motivate the different requirements of these systems for fog environments. In addition, many researches are working on these scenarios in order to propose innovative solution combining IoT, fog and cloud to address the identified needs [28,52] . In this survey, without any pretence of being exhaustive, we consider three reference case studies that are emblematic of the wide range of IoT applications deployed nowadays. Below a brief description of these case studies is presented. They are from different domain, so that the identified requirements are evaluated in different situations.

Smart Traffic Light (STL) [28,53,54] systems focus on improving how the traffic and the congestion of the city road are handled. These systems rely on video cameras and sensors distributed along the roads, and especially in the crossings, to sense the different vehicles and elements in the roads, detecting the presence of pedestrian, bikers, vehicles or ambulances. These systems, for example, in order to reduce the contaminations are able to know when the different traffic lights have to be turned on, because there is a huge concentration of cars in one direction, and when switch them off as traffic passes. Likewise, when an ambulance flashing lights are detected they can automatically change street lights to open lanes for the vehicle to pass through traffic or, even, to create green traffic waves and deliver warning messages to approaching vehicles. In this scenario, traffic lights could act as fog nodes coordinating the different actions.

Wind Farm [28,55,56] systems try to improve the wind power capture, and preserve windmill structure under adverse condition. To that end, different sensors to identify the turbine speed, the generated power, and the weather conditions are necessary. This information can be provided to the local fog node located in each turbine to tune it in order to increase the efficiency and to reduce the probabilities of damage due to the wind conditions. In addition, wind farms may consist of hundreds of individual turbines that have to be coordinated to get the maximum efficiency. The optimization of one turbine can also reduce the efficiency of other turbines at the rear. Therefore, higher-level fog nodes are also needed to provide a general strategy for the farm in order to increase its efficiency. Finally, more general and long-term analytics about the wind patterns on a yearly and monthly basis should also be generated. These analytics could be performed on the cloud.

Smart Grid [17,45,57,58] systems have been promoted as a solution for minimizing the wastage of electrical energy. These systems are usually based on analysing the energy demand, availability and price in order to automatically switch to alternative energies like solar and wind. In order to do that, different fog nodes with energy load balancing applications may be deployed on the network edge devices, such as metres and micro-grids. In these systems, local decisions and actuations could be taken in the local nodes but also high-level information could be sent to the cloud to generate business intelligence analytics.

Finally, other cases studies have also been analysed. Nevertheless, they have not been included in this paper due to space restriction. These domains in which the deployment of the fog paradigm is also essential are: Smart Connected Vehicle [23] and Smart Building [59].

4. A conceptual architecture for fog for IoT

This section presents a conceptual architecture for Cloud-Fog-IoT applications (see Fig. 2). Its main goal is to clarify its most important components and their interactions. These components will be the basis for presenting the taxonomy about fog computing for IoT. They have been derived from the requirements presented in Section 3.2 and from the surveyed approaches. As detailed before, the surveyed approaches have been used to define a general conceptual architecture covering the different aspects and features presented in the analysed works. Then, the designed architecture was refined in order to better meet these requirements.

Let us start noting that fog computing is a very wide, comprehensive, and application-oriented research area. Hence, deriving a traditional layer-oriented architectural view, commonly used to model communications systems, is often difficult and not viable as recognized by the majority of existing standardization and surveying efforts that typically adopt [30,31,38]: (i) a deployment view to organize the fog node hierarchy; (ii) and then a rather flat and horizontal conceptual architecture that includes all main functional elements that, then, can be composed and deployed at different deployment layers in the distributed fog hierarchy. Along that line, we propose a conceptual architecture consists of a vertical (deployment) dimension and a horizontal (functional) dimension.

The vertical dimension consists the three *deployment layers* (or simply *layers*), namely, cloud, fog, and edge, introduced in Section 3.1: they reflect the different kind of nodes that are responsible for executing the tasks of the components comprised within those areas. Those components between two areas reflect the situation in which a task can be executed by different nodes depending on its granularity, the complexity of the application, and the position and role in the hierarchical fog deployment.

The horizontal dimension, instead, includes six different perspectives that are cross-layer with respect to the deployment model (i.e., it is possible to instantiate and compose these functions at the different deployment layers). In particular, adopting the definition by OpenFog [38], we define *perspective* a group of components or modules highly related because they have a similar goal or role. All the perspectives in the conceptual architecture are vertical, since they affect the three areas of architecture and, depending on the complexity and specific requirements of the IoT application, can be located in different areas. For example, the Communication perspective has to be considered to improve the communication between end devices, fog nodes and cloud. In the following, the different perspectives are further detailed explaining how their components are applied to each specific node.

4.1. Communication

The Communication perspective takes care of enabling the communication between the different nodes of the network, and it is widely recognized as an important core functionality [31,40]. In the context of fog for IoT, this perspective contains different techniques for a proper communication between nodes, especially with (typically) poor IoT devices. These techniques involve standardization mechanisms to address several of the requirements introduced in Section 3.2. First of all, the infrastructure should be interoperable to foster open exchange of data. In addition, IoT applications are usually characterized by a high mobility of some of their devices, this perspective must also contain techniques allowing the migration of a device from one subnetwork to another without decreasing the system performance. At the same time, this perspective is crucial to achieve real-time responsiveness. If the communication protocols are not efficient, this requirement will not be achieved. Therefore, it also includes different techniques to reduce the latency of the communications. Finally, a crucial requirement of IoT applications is that the posted data reaches its destination. Most of the surveyed approaches and works have characteristics to assure different levels of network semantics to obtain a reliable communication substrate. Because of that, we defined in this perspective a last component to include methods, or configure the already included techniques, to assure the network semantics, guaranteeing that the information will not be lost in the network. Let us conclude this section noting that although network virtualization (by including here both software defined networking and

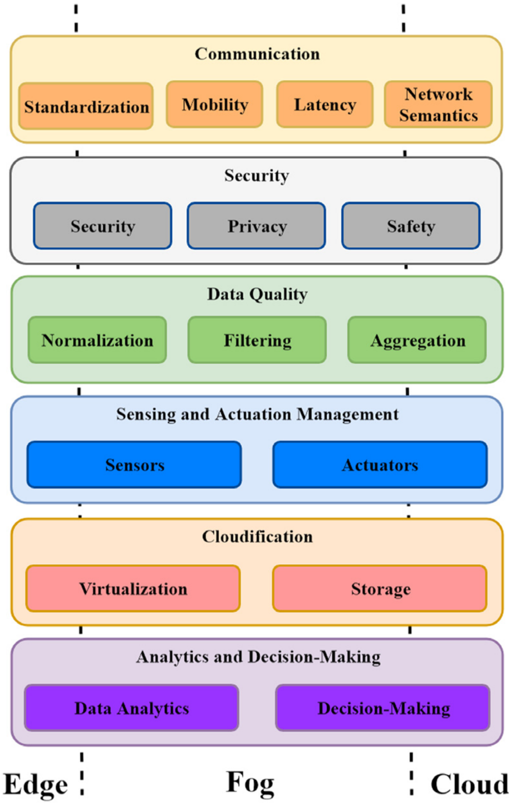

Fig. 2. Conceptual architecture, perspectives, and components.

network function virtualization) is a very hot topic, the use of network virtualization in existing systems in the IoT field, and in fog for IoT, is still in its infancy. This is confirmed by the very few works available in the literature and by the limited importance given to this dimension also by other survey efforts in the fog field, such as [30,32]. At the same time, as explained also in Section 6.7, we expect an increasing trend in network virtualization in fog for IoT during the next years.

4.2. Security

The Security perspective affects the whole architecture, since all communications, data, and actions must be carried out in conditions ensuring systems security in a broad sense according especially to the data quality, security, and reliability requirements (see Section 3.2). The security is a core functionality recognized by standards (e.g., the security perspective in the OpenFog reference architecture [38]) and by surveys on fog computing, although typically with a stronger emphasis on ICT aspects [25,31,32]. In the context of fog for IoT, it has been recognized the importance of enlarging the scope of the security perspective to include not only ICT issues, but also safety issues that may arise due to the use of physical sensors and actuators [47]. Accordingly, our security perspective consists of three different components: security, privacy, and safety. First, security focuses on different techniques to assure the reliability, confidentiality and integrity of the communication between the different nodes. Second, IoT applications usually handle a lot of sensitive data. The privacy of this information is extremely important for users to trust the systems. Therefore, this perspective should include access control mechanism to provide that information only to authorized users. Finally, in some cases, IoT systems act in critical environments in which safety procedures have to be deployed to assure the welfare of the different elements involved in the systems, leveraging also data quality indicators. Fog environments should facilitate the development of such policies. In this perspective these three components have been defined because they are highly related but pursue slightly different sub-goals. In addition, as identified in the analysed works, some of them could be required or not; for instance, an application could require security but not safety.

4.3. Data Quality

As introduced in Section 3.2.4, while this perspective is sometime treated/included as a component of other perspectives (e.g., in the OpenFog "Data, Analytics, and Control" perspective [38]), we this fog for IoT requirement so important to isolate a full-fledged Data Quality perspective. This perspective is in charge of processing all the sensed and gathered data in order to increase their quality, but also to reduce the amount of information to be transmitted by the IoT devices or stored in the fog/cloud nodes. This perspective comprises three different components that sometimes are sequentially executed: data normalization, data filtering and data aggregation. First, the data normalization component gets all the sensed raw data to homogenize and specify it into a common language in order to achieve semantic interoperability (see also Section 3.2.2). Then, different data filtering techniques can be applied to extract only that information useful for the system, throwing away worthless information to not waste unnecessary computational resources. Finally, the Data aggregation component takes the filtered data to create a unique stream of information to improve its analysis. All these techniques can be applied to different areas and nodes in the architecture to reduce the amount of information to transmit and store. Architectural components performing data processing are not particularly new and different architectures use them in different context. They are basic to bridge heterogeneous data to other computational components, increasing its quality, and improving the scalability and responsiveness of the system.

4.4. Sensing and Actuation Management

The Sensing and Actuation Management perspective is comprised by all those devices (both physical and virtual) in charge of sensing the environment and acting in certain situations or under specific orders. This perspective crosses all the areas because the virtual sensors or actuators can be part of the fog and cloud nodes, while the physical devices are part of the network deployed for the correct operation of the application. Moreover, it represents a specialization of the management functionality typically present in all main related fog standards and surveys [30–32,38].

This perspective contains two components: Sensors and Actuators. Sensing is a critical aspect, because directly affects the quality (e.g., in terms of precision, accuracy, confidence level, etc.) of the generated data, which typically is the primary input for successive application steps. Acting is revealing as a major part of IoT systems in many sectors because is always more crucial to create systems actively contributing to improve a context, rather than passively storing incoming events. Fog computing can improve the actuation phase with timely reactions to sensed, aggregated and filtered information.

4.5. Cloudification

Cloudification acts as a small distributed cloud in the fog architecture. This perspective can bring limited cloud services and resources closer to the edge, reducing unnecessary global-scope interactions as widely recognized in fog computing [31,32,38]. In order to be able to deploy a cloud inside a fog node, virtualization techniques are required in order to be able to deploy different applications in the same node. In addition, the different instances and services could be orchestrated with the aim of composing more complex functionalities. Finally, the storage component takes care of managing the distributed information that has to be stored in the node, coordinating its processing and controlling its privacy. Various and non-negligible benefits are expected with that infrastructure because every task can be performed in a location-aware context with better analysis and results (see also Section 3.2.6). Moreover, it is possible to confine traffic near IoT devices, without adding traffic load onto the network. Therefore, this perspective provides important benefits regarding responsiveness and user Quality of Experience (QoE). Depending on the concrete requirements (such as location-awareness and responsiveness) of the IoT applications, and how restrictive they are, these components could be located in the edge, fog or cloud. In addition, this perspective only includes the components required for the correct execution and deployment of IoT applications, while other functionalities such as algorithmic aspects of dynamic distribution of resources and pricing/accounting supports are typically not addressed in fog for IoT solutions. Finally, at the current stage, we consider network virtualization still a marginal part of fog for IoT cloudification, as better detailed in the previous Section 4.1.

4.6. Analytics and Decision-Making

The Analytics and Decision-Making perspective is in charge of analysing the stored data in order to generate different analytics and to detect specific situations, and is widely recognized as a core part of fog computing [30,32,38]. In ubiquitous environments, where a huge numbers of sensors constantly gather information and send it to the fog, the combination of short (in the fog) and long-term analytics (in the cloud) can lead to accomplish both reactive and proactive decision-making, improving the system scalability and covering a wider range of IoT applications (see Sections 3.2.1 and 3.3). The complexity of IoT environments leads to the necessity of a precise initial analysis of the surrounded environment in order to define a valid model to use in the system. On the fog or edge side, there are small data analytics that can be addressed as an extension of Big Data near the devices. These analytics are still related to sensors data and refers to a limited quantity of highly granular data that usually provide valuable information for the system, used to perform real-time decisions and actions. Meanwhile, cloud environments can perform long-term and heavy resources operations, associated to Big Data but easily extendible to any IoT applications. Usually during Big Data analysis and processing, significant resources are used to support data intensive operations that require high computational resources. In addition, these long-term analytics can be used to perform coordinated and proactive decisions and actions.

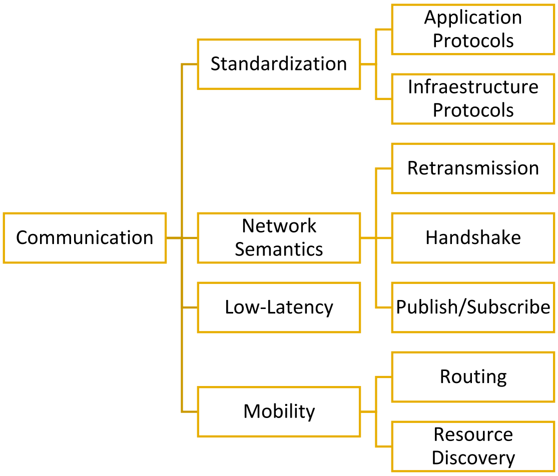

Fig. 3. Taxonomy for the classification of the Communication perspective.

5. Fog computing for IoT: a taxonomy

This section proposes an original taxonomy for clarifying the main characteristics and components used in the fog computing for IoT applications. The goal is to better explain our classification that stems directly from our architectural model, as detailed in Section 4. To facilitate the full understanding of our original proposed taxonomy, together with it, we also present some state-of-the-art proposals that provide support to some specific components or characteristics. In addition, we explain how some of them are applied to the case studies presented in Section 3.3. We believe that this analysis is also of great help to IoT application designers in order to obtain information on reference cases and design guidelines they can follow.

The presentation order of the taxonomy is directly derived from the proposed conceptual architecture. Therefore, below six different parts of the taxonomy are detailed, one for each architectural perspective. Section 5.1 introduces the different communication characteristics. Section 5.2 analyses the different security characteristics required by IoT applications. Section 5.3 classifies how the gathered data are processed. Section 5.4 details the interactions between the fog and the IoT devices. Section 5.5 analyses the aspects that have to be considered for building a distributed cloud using the fog nodes. Finally, Section 5.6 concludes by classifying the taxonomy for the data analytics and decision-making aspects.

5.1. Communication

The Communication perspective offers four different component providing support to the different characteristics and requirements of IoT applications regarding the communication between the devices, and the fog and cloud nodes (see Fig. 3): standardization of the communication among the different nodes, network semantics of the transmitted information, reduction of the communication latency and mobility of the devices. Some communication protocols implement different techniques to support several of the above components. However, as detailed below, each one has its advantages and liabilities, making them more or less suitable for different environments.

5.1.1. Standardization

One of the most critical points for the correct integration and communication between IoT devices and applications is the protocol used. These protocols allow developers to achieve the infrastructure interoperability in IoT systems. Different authors [17,26] divide the infrastructure interoperability into two different set of protocols: application protocols and infrastructure protocols. The former are those protocols and standards used at the application level to ensure messages exchanging and interactions among applications and their devices (CoAP [60], MQTT [61], AMQP [62], DDS [63], ZigBee [64], UPnP [65], DPWS [66]). The later are needed to establish the underlying communication among different networks (RPL [67], 6LoWPAN [68], BLE [69], LTE- Advanced [70], LISP [71]). Each system can adopt a different stack of protocols depending on the requirements and the features of each application. In order to facilitate the adoption of these protocols, currently, there are different libraries and frameworks implementing them. For instance, Mosquitto [72] is a lightweight message broker that implements the MQTT protocol and can be easily integrated within IoT applications [73]. The Kura framework is an OSGi-based open-source framework for IoT application that uses MQTT as its central protocol and, in addition, it implements different functionalities for aggregating and controlling device information [74].

For example, Smart Grid are systems composed of a massive number of distributed and heterogeneous devices, widespread in different networks that have to be able to communicate among them. ZigBee is particularly adopted in

Smart Grid applications due to its short range and robustness under noise conditions [22]. For instance, in [57], ZigBee is applied in Smart Grid applications to connect sensors to smart metres, taking advantages in particular of its low bandwidth requirements and low cost of deployment. In [45], the authors use specific application protocols, such as ZigBee, together with DPWS to improve the devices discovery, interoperability and mobility. Another popular type of WPAN connection used the Bluetooth technology, and specially the BLE version, characterized by a very low transmission range, and a poor data rate, but also with a low power energy consumption [75].

5.1.2. Network semantics

Another important property of the communication protocols is the network semantics. This property ensures the reception of the data transmitted by the different nodes of the network and, therefore, is essential for critical systems. Currently, different techniques can be used to assure the semantics of the network, such as: Retransmission, Handshake and Multicasting. For the retransmission, many application protocols, like CoAP, MQTT, AMQP and DDS focus on network semantics and are based on retransmission schemes that are able to handle the packet loss in the lower layers [18,76], improving the reliability of the communications. For instance, the scheme Per-hop retransmission tries to retransmit a packet several times before the packet is declared lost [76]. CoAP is based on UDP, a not reliable transport layer protocol, but provides the use of confirmable messages [18,77]. The handshaking mechanism allows two nodes to negotiate the parameters of the connection before transmitting data. MQTT and AMQP provide three different layers that can be used depending on the application-specific requirements. Third, the publish/subscribe technique allows publisher devices to post information that is forwarded to subscribers, allowing even to multicast the information to several devices. DDS and DPWS, for instance, use this technique to bring excellent Quality of Service (QoS) and high reliability to its applications [45,77] with the support of numerous QoS policies in relation to a wide range of customizable communication criteria: network scheduling policies (e.g. end-to-end network latency), timeliness policies (e.g. time-based filters to control data delivery rate), temporal policies to determine the rate at which periodic data is refreshed (e.g. deadline between data samples) and other policies that affect how data is treated during the communication in relation to its reliability, urgency, importance, and durability.

In STL, with the tremendous rise in the number of sensors deployed in the road, the number of connected vehicles and their ever-increasing mobility, the support for low latency and uninterrupted communication between the sensors and the fog is crucial to assure the correct operation of the applications [23]. DDS has been used as basis for improving the network semantics and the QoS in these environments [78].

5.1.3. Low-latency

As fog computing is implemented up to the edge of the network, it facilitates the provisioning of low latency responses, of course if coupled with adequate data connection protocols. Different protocols may be used to improve the response between nodes (fog or cloud) or between devices and nodes. Some of the previously analysed protocols have been used and adapted to achieve low-latency. For example, [77] uses MQTT, a publish/subscribe protocol, to realize real-time iterations and low-latency synchronous data streaming in strictly real-time environment based on fog computing solutions. MQTT carries data stream between fog and cloud and MQTT-SN, the lightweight version, transports data from devices to fog nodes (to achieve the low latency between the end-devices and the fog). In [79], the authors propose and extension of Kura framework (an open-source framework that uses MQTT) to reduce the latency, among other aspects. CoAP is another application protocol particularly used in IoT applications to provide low-latency interactions. In addition, [18] discusses performance differences between MQTT and CoAP, highlighting response delay variations in relation to the reliability and quality of service provided for the communication: the lower the packet loss, or bigger the message size, the more MQTT outperforms CoAP, and vice versa. Hence, it is necessary to decide which protocol to use in relation to the type of application, adopting MQTT for reliable communications or for big-size packet communication, and CoAP otherwise, in order to decrease latency and increase system performance. DDS is widely used for real-time M2M communications among constrained devices [78]. [78] addresses DDS as one of the best solutions for real-time distributed industrial systems.

In Smart Grid, most control functions have tight delay requirements and need real-time behaviours. Low-latency actions are basic to improve the system's flexibility on both sides of the electricity market, creating automated demand–response on the user side and aggregating smaller distributed generation on the supply side, and tracking the energy generation/demand [80]. Electricity markets, in smart grid applications, aim to use real-time pricing (demand–response) and charge customers with time-varying prices that reflect the time-varying costs of electricity procurement at the wholesale level [80]. [81] proposes to use DDS for enterprise distributed real-time and embedded systems, such as smart grid applications, because it provides efficient and predictable dissemination of time-critical data.

5.1.4. Mobility

IoT applications are characterized by the high mobility of some of their devices [7]. Different protocols apply routing and resource discovery mechanisms to support such mobility. Routing mechanisms are in charge of constructing and maintaining routes between distant nodes. Some protocols are specialized on maintaining these routes even if the nodes have mobility requirements. For example, the Locator/ID Separation Protocol (LISP) specifies an architecture for decoupling host identity from its location information in the current address scheme. This separation is achieved by replacing the addresses used in the Internet with two separate name spaces: Endpoint Identifiers (EIDs), and Routing Locators (RLOCs). Separating the host identity from its location highly improves its mobility by allowing the applications to bind to a permanent address, the host's

EID. The location of the host can change many times during an ongoing connection. RPL is another routing protocol, created for constrained communications, using minimal routing requirements through building a robust topology over lossy links and supports simple and complex traffic models like multipoint-to-point, point-to-multipoint and point-to-point [26].

In STL, mobility and routing support is one of the main requirements needed by the system to create effective applications due to the high mobility rate of vehicles. In particular, RPL can be adapted to multihop-to-infrastructure architecture, as a network protocol enabling large geographical area coverage of connected vehicles with relatively minimal deployment of infrastructure [82]. In addition, this protocol is emerging as the reference Internet-related routing protocol for advanced metring infrastructure applications [83].

Resource discovery techniques focus on identifying the neighbour nodes when a device change from one location to another to establish new communication links. For example, CoAP provides a mechanism for resource discovery of nodes in the sub-network, through URI path that define a list of resources provided by the server and visible to clients. Instead, MQTT needs a discovery support because it does not provide a discovery mechanism and clients must know the message format and the topics to allow the communications. UPnP is a discovery protocol used in some application contexts. In particular, some fog solutions use UPnP+ that is an extension for IoT applications [84]. This version includes lightweight protocols and architectural elements (e.g. REST interface, JSON data format instead of XML) to improve the communication with constrained devices. Moreover, in [85], Kim et al. propose an architecture for Smart Grid using UPnP to detect new devices automatically without any user intervention. In [45], Abdullah et al. use DPWS-compliant services to improve services/devices discovery by leveraging a protocol based on IP multicast.

5.2. Security

The Security perspective offers three fundamental components for IoT systems: Safety, Security and Privacy. In the presented use cases, security and privacy must be considered, from computational to physical point of views. In addition, some IoT systems should provide some safety policies to their users. As was detailed above, different security policies can be implemented throughout the data life cycle. For the sake of clarity, Fig. 4 summarizes the proposed taxonomy with the possible choices for any of the components.

5.2.1. Safety

Safety is an essential property for critical IoT systems. Normally, the safety should be part of the rules and business logic of the IoT systems. Nevertheless, fog environments should facilitate the development of such policies. The most commonly used safety techniques are Activity Coordination, coordinating actions to maximize the users or goods safety; Activity Monitoring, controlling the actions carried out at all times to ensure its correct execution; and, Action Planning orchestrating the actions to perform in the event of the identification of hazard situations by using deterministic and stochastic models [20].

Evaluating their application, in [19], the authors apply the Coordinated Activities approach to a STL in order to create green waves to help emergency vehicles to avoid traffic. Or, Action Control techniques to monitor every operation, through images acquisition. Normally, every user action is tracked using targeted surveillance. Finally, in Wind Farm, the system must face weather conditions, relate them with predefined limits and apply a set of planned actions in case of adverse circumstances that may be dangerous for the physical integrity of the system (e.g., stop the turbines in case of strong wind), trying to maximize the economic benefit without sacrificing safety. In [86], the authors review different approaches to address uncertainty in wind power generation in the unit commitment problem, with interesting preliminary results that indicate the presence of models that can effectively balance costs and safety. In [20], the authors apply Action Planning and stochastic models to maximize the wind power penetration without sacrificing safety.

5.2.2. Security

At least four basic pillars usually support the security of IoT systems: confidentiality (making sure that the data arrive to the right place preventing their disclosure by unauthorized entities), data loss (preventing the loss of information during its transmission), integrity (detecting and preventing unauthorized alteration of the information) and intrusion detection (identifying if an unauthorized user is trying to access to the system). First, in order to improve the confidentiality, Data Encryption and the use of Sandboxes to isolate executions, data and communications, are commonly applied [25]. Secondly, for reducing the data loss, specific protocols are used to send information to fog nodes or to cloud environments, by means of Version Control and Configuration Management approaches [87,88]. Third, File Permissions, User Access Controls, Checksum and Hashing methods are applied to increase the data integrity, detecting and preventing unauthorized alteration of the information over its entire life cycle [89]. Finally, Data Analytics and Pattern Detection techniques are used to observe the behaviour of the systems and the users in order to detect anomalies and intrusions [52]. For instance, reputation-based systems and truth discovery approaches forensically analyse the behaviour of each node to identify attacks [90]. Nevertheless, intrusion detection algorithms are difficult to adapt to each IoT system because they require a deep knowledge of the system and its users.

The previously defined protocols apply some of these techniques to improve the security. For instance, MQTT apply SSL/TLS encryption techniques. AMQP extends the security of MQTT with sandboxes for the authentication phase. In addition, AMQP separates the message and the delivery information, providing meta-data management and encrypted message.

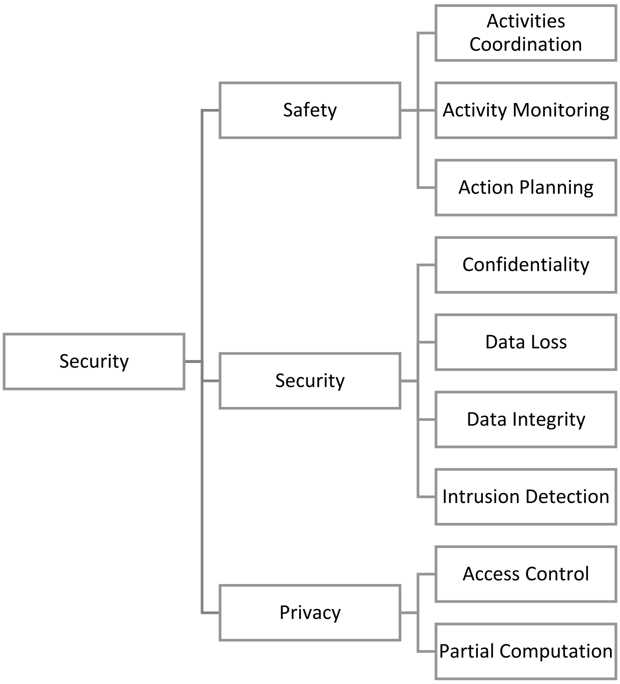

Fig. 4. Taxonomy for the classification of the Security perspective.

Finally, CoAP improves the security/privacy using DTLS (datagram transport layer security) to prevent eavesdropping and tampering.

Some of these techniques are used in the analysed case studies. In Smart Grid, for instance, the required security levels are extremely high, since security issues can produce the disruption of the system, destabilize the demand patterns or, potentially, initiate a blackout [91]. These systems must ensure that devices are well protected, using User Access Control policies, and that the sensitive data cannot be modified during their transmission, using Data Encryption and Sandboxes techniques [17]. In addition, Fault-tolerant and integrity-check methods are deployed in power systems to protect data integrity and, also, to defend and anonymize user's activity and their localizations.

5.2.3. Privacy

The essential methodology to assure the privacy of the data is controlling data access, trying to avoid false or unauthorized users to get the users' information. Moreover, Stojmenovic et al. [52] indicate that authentication at different levels is one of the main security issue of fog environments. Therefore, some mechanism such as User Authentication, Security Token or Air Gapping can be used to increase the privacy of sensitivity applications [92]. Each device has an IP address and a malicious user can tamper his device and send false reading reports or spoof IP addresses. In order to overcome this issue some authentication techniques, based on public key infrastructures or key exchange, could be introduced. To improve the exchange of private information, some works, such as [93], use a Partial Computation technique (i.e., Secure Multi-Party Computing (SMCP) [94]). SMPC consists of two or more parties, where each party has their own secret input. SMPC computes a joint function that receives as input the secret information of each party. At the end of the protocol, each participant will get only the result of the function.

In Smart Grid, the privacy concerns are mainly related to the possible diffusion of detailed users' data (e.g. pricing information, account balance) or information associated to the disclosure of energy information (e.g. voltage/power readings, device running status) to unauthorized entities. This is a very valuable information to both end users and utility companies. [17] classifies the main Smart Grid vulnerabilities: (i) device vulnerabilities, malicious attackers can compromise IEDs; (ii) network vulnerabilities, the adoption of open network architectures can be risky for routing modifications, DNS hacking, different denial-of-service; (iii) data vulnerabilities, data attacks designed to compromise the privacy of customers and understand users' behaviours, activities or habits. Typical Smart Grid applications guarantee strict access control with minimal functionalities performed by each node, usually constraint node. With the introduction of fog computing, Smart Grid nodes can delegate access control to the fog nodes that dispose greater resources and, thus, perform a more accurate analysis.

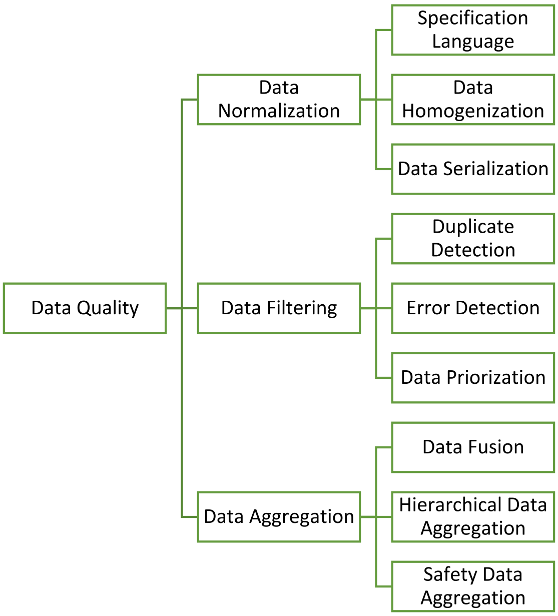

Fig. 5. Taxonomy for the classification of the Data Quality perspective.

5.3. *Data Quality*

The Data Quality perspective is in charge of processing all the gathered data in order to provide a uniform specification to all the information, get rid of the useless data as soon as possible and reduce the amount of data to store in the fog node, or to be transmitted to other nodes or to the cloud. The Data Quality perspective offers three fundamental components for processing the data: Data Normalization, Data Filtering and Data Aggregation. Fig. 5 shows the proposed taxonomy for this perspective.

5.3.1. *Data normalization*

IoT and fog are extremely heterogeneous in nature. Sensors range can be various, from wimpy to powerful sensors, with orders of magnitude of differences, etc. Similarly, fog nodes are heterogeneous in nature, and they can provide different kind of services. Therefore, all the sensed and provided information should be normalized in order to facilitate the data exchange [95]. To that end, the sensed data can pass through different steps: specification language (transforming it into a common format), data homogenization (unifying it by means of semantic data, open standard middleware, etc. [96]) and data serialization (converting and compacting data to different format in order to transmit them efficiently). Normally, these techniques are combined to improve the data normalization. For instance, Zao et al. [77] face the data normalization with a two-level mechanism: first, they use a unified specification language, called Pigi [97]; secondly, they use the Google Protocol Buffers [98] to perform the data serialization.

Smart Grid systems are usually composed of distributed and heterogeneous devices, requiring the use of standards and protocols to achieve the inter-communication among them. [99] provides some guidelines to identify standards and protocol supporting interoperability of the Smart Grid, with the definition of architectures to incorporate and support a broad range of technologies. In addition, standard languages are essential to perform interoperability among smart metres, smart devices, charging interfaces, and to exchange information among all smart grid applications [17]. Finally, Wind Farm is a quite close system that must sense the wind and the turbine power and react with a limited numbers of different typology of actuators, thus, data normalization, homogenization and serialization techniques should be selected in order to efficiently exchange the information [100].

5.3.2. *Data filtering*

Data filtering is a component aimed at reducing the number of information transmitted by eliminating those data that are redundant, erroneous or faulty [101]. Data filtering techniques should be implemented as near to the edge as possible in order to reduce the data traffic as soon as possible. Although sensors may implement light-weight filtering, to remove some noises at the data collection phase, more robust and complex data filtering techniques are still required.

The main data filtering techniques are Duplicate Detection, Errors Detection and Data Prioritization. Duplicate detection techniques are focused on the analysis of the received data in order to identify the redundant data that could be eliminated. Spatial–Temporal and Buffering algorithms can be used to detect these data [21]. For instance, Bloom algorithm [21]

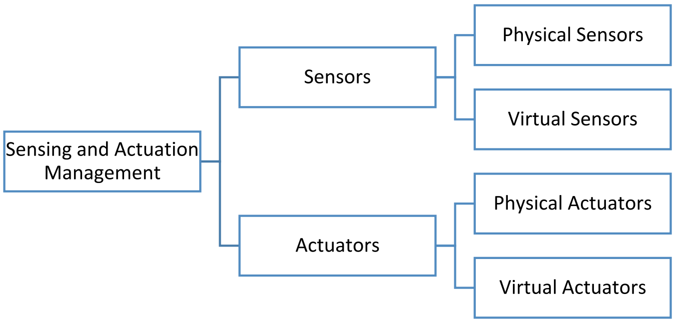

Fig. 6. Taxonomy for the classification of the Sensing and Actuation Management perspective.

identifies the duplicate data by means of a buffer that checks whether a new data is already stored or not. Other algorithms make use of spatial–temporal correlation between some sensors, so there is a high probability that they will gather the same data in a specific moment or in the near future. Error detection mechanism try to identify faulty data produced by incorrect measurement. Arithmetic models (e.g. Hodrick–Prescott, Moving Average, etc. [22]) and Statistical models (e.g. Six Sigma, Lean, etc.) can be used to compare the information or predict the data distribution in order to create models predicting the data distribution and highlighting those data that do not fit the model, i.e. outliers [102]. Finally, data prioritization techniques do not focus on reducing the data set. Instead, their goal is to filter the time-critical data with the aim of processing it as soon as possible. For instance, in [79], the authors extends Kura to include message priorities and propagate the messages to theirs destination as soon as possible.

In the STL, a metric for identifying the outliers and detect the correct speed of the approaching vehicles can be introduced by calculating the standard deviation of the collected speed in relation to the average data received in the same conditions. In Smart Grid, the signals are typically sampled and communicated at high rates, leading to some congestion problems. In [103], the authors propose the combination of Distributed Execution with Filtered data forwarding techniques in order to prioritize the most important data.

5.3.3. *Data Aggregation*

The Data Aggregation component focused on further reducing the collected information. To that end, it uses different complementary mechanisms focused on: fusing data, hierarchical aggregation and improving the system safety through the data aggregation. Data fusion techniques try to merge different kind of data in order to reduce the data set and obtain a unique data flow. To that end, different arithmetic operations (to get more representative values) and spatial–temporal techniques (to aggregate data depending on their location/timing) can be applied to get more stable and representative values of a large sample. Spatial techniques can be used to aggregate data depending on the samples location. Alternatively, temporal techniques aggregate data depending on when they were gathered [55]. Hierarchical techniques propose to aggregate data successively on different nodes. In fog, where may exist intermediate nodes with different capacities, this technique allows the successive application of aggregation techniques to exploit the resources and location-awareness of each node. Safety Data Aggregation focuses on gathering similar information from different sensors to have different point of views of the same situation with the goal of improving the safety of the IoT applications. This perspective could be seen as a mixture of the above perspectives applied to the concrete requirement of safety. These three techniques improve the use of the hardware redundancy technique reducing the amount of information that it generates to improve the behaviour of the system in distributed networks in fault tolerant situations, greatly increasing the reliability of the system [104].

In STL, the control of the different conditions of the vehicles and the roads is critical to provide the desired safety to drivers. These are highly distributed applications with many geo-distributed data collectors that must communicate and aggregate data in order to create efficient traffic policies to route vehicles. Currently, different approaches rely on a combination of Hierarchical and Safety Data Aggregation techniques to identify and track vehicles using surveillance cameras and different sensors. For instance, the BOLO Vehicle Tracking Algorithm [24] forward the recorded video and the sensed information to different nodes in a tree (with different capabilities) to hierarchically processes it and identify specific vehicles to track. In Smart Grid, the information generated by the different elements of the smart grid network is quite large. Therefore, in [45,101], the authors propose the hierarchical aggregation of data using arithmetic and temporal operations at data aggregation points (i.e., distributed stations, substations, etc.).

5.4. *Sensing and Actuation Management*

The Sensing and Actuation Management perspective comprises all those elements sensing information from the context and executing actions to change the environments for achieving the desired goals. This perspective is of central relevance because of the very nature of IoT applications. These applications always require sensors or/and actuators interacting with the environment. Fig. 6 shows the proposed taxonomy with the most important possibilities for these components.

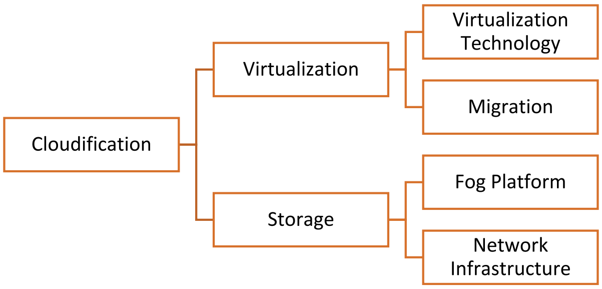

Fig. 7. Taxonomy for the classification of the Cloudification perspective.

5.4.1. Sensors

Sensing is a property that most IoT systems perform in order to understand the environment and identify if the business goals are being achieved. This component can take care of two different kind of sensors: physical sensors, which gather information directly from the environment by means of specific hardware, and virtual sensors, which obtain the information through other sources (a web service of a third party system, for instance) [105,106]. Even, different works replicate the sensing elements to obtain more information in order to make tolerant to faults and more reliable systems [104]. To obtain a greater benefit, some proposals mix the information from both kind of sensors. For instance, in STL, the physical sensors are normally used to obtain information on the traffic flow and the virtual sensors to get information on meteorological forecast or traffic alerts from the traffic authority [53]. Likewise, in Wind Farm, physical sensors are fundamental to get real-time information on the weather conditions (wind strength or electrical power generated) [28]. Instead, the virtual sensors are used to get the forecasted weather conditions [55]. Thus, different reports can be generated to compare the weather conditions with the generated power or deviation from the forecast loads.

Some systems only have a sensing component, because they only have to observe the environment without focusing on any actuation activity. Business Intelligence (BI) applications are a notable example of this kind of systems. BI applications can use techniques such as data discovery [107], data mining [108], business performance, analytics and processing, to turn sensed raw data to valuable information for later decision-making processes, for generating reports or for visualizing results [109,110]. In particular, due to the pace of the real-world environment, it is possible to use BI in real-time scenarios to support information delivery, data modelling, data analysis, and propagate the generated results on real time [111], improving the decisions-making process and maximizing enterprise resources.

5.4.2. Actuators

On the other hand, other systems are based on a sensing phase along with a strong actuation phase. Thus, the actuators can change the environments with the goal of automatically or semi-automatically achieve the desired goals. This component is divided into two different kind of actuators: physical actuators, which can physically produce a change in the environment by means of specific hardware; and virtual actuators, which can be used to: control a set of actuators [112], to hide the low-level information to interact with the physical actuators [113] or to substitute faulty actuators in order to allow systems to continue with their usual execution [56].

In STL, the actuation is a key component to prevent accidents and maintain a fluid traffic. For instance, physical actuators can trigger alarms or change the traffic light from green to red to slow down approaching vehicles. Likewise, virtual actuators can be used to better control big areas and create green waves of traffic lights in order to decrease pollution or for emergency vehicles [46,114]. In Wind Farm, physical actuators are used to start and stop turbines in relation to the prediction forecast and the wind strength, and prevent part of the systems to break [28]. In addition, recent studies in wind turbines replace the real faulty actuator by activating the corresponding virtual actuator [56]. Finally, in Smart Grid, operational planning and optimization actors perform simulation of network operations, schedule switching actions, dispatch repair crews, inform affected customers, and schedule the importing of power [99].

5.5. Cloudification

The Cloudification perspective allows the execution and deployment of different IoT systems in edge, fog or cloud nodes, converting the fog into a small-distributed cloud. This perspective offers two essential components (see Fig. 7): Virtualization and Storage. First, the Virtualization module allows developers to encapsulate IoT applications and deploy them in the edge, fog or cloud nodes. Secondly, the Storage component supports the persistent storage of information in order to increase the system responsiveness. Note that, for the sake of specialization and focalization, in this classification we only include the components required for the correct execution and deployment of IoT applications, but not for the distribution of resources among different IoT applications nor for the identification of different pricing models.

5.5.1. Virtualization

The Virtualization allows fog nodes to create virtual machines (VM) to support specific IoT applications, providing isolated environments. Thus, a fog node, for instance, can have deployed different VMs supporting different systems. Virtualization considers two main characteristics that have to be taken into account: the technology used to encapsulate the IoT system and how the virtual images are migrated from one node to another, supporting the users and system mobility requirement and the system reliability. Currently, the main technologies to create virtual images are hypervisor and container. Hypervisor (e.g. OpenStack [115] and OpenNebula [116]) is a flexible virtualization solution, since the virtual image not only contains the final application but, also, the operating system required to execute it. Instead, container (such as LinuX Containers (LXC) [117] or Docker [118]) is a lightweight solution since the operating system is not virtualized. Not having to support the emulation of different operating systems improves the performance and the migration of the containers. Migration is another key property of the virtualization to address the mobility requirement. When a user leaves the area covered by the current fog node, or when the node efficiency is reduced, the VMs or the container may need to be migrated to another node covering the system's requirements. The migration of VMs or the container should be fast enough to maintain the real-time, location-awareness and reliability requirements of IoT applications. Two main migration techniques are used in the surveyed solution: a complete migration (using Internet Suspend/Resume (ISR) [119] or Xen live migration [120], for instance) or partial migration (using, for example, variants of the previous techniques [121]).

There are several proposals applying these virtualization techniques to fog. Cloudlet, for instance, proposes a three-layer architecture. The bottom layer contains the operating system (Linux) and the data cache. The middle layer includes a hypervisor to encapsulate and separate the transient guest software from the cloudlet infrastructure's permanent host software environment (concretely, it uses OpenStack++ [122]). The third layer contains the applications isolated by different virtual machine instances. Finally, Cloudlets also implements a specific technique for the partial migration of the VM instances, called dynamic VM synthesis. Each Cloudlet node contains a base VM and each mobile device contains a small VM overlay. Therefore, when a mobile device change from one node to another, the source node suspends the overlay and stores it in the mobile device. When the mobile device is in the destination, it transmits the VM overlay to the target node, applying it to the base and starting its execution in the precise state in which it was suspended.

In [29], the authors define an experimental fog computing platform. This platform uses a hypervisor virtualization technique. Concretely, they make use of OpenStack together with the Glance module for the management of VM images. In addition, to support service continuity, they also implement the two different migration schemes. In the first method, they take a snapshot of the VM to be migrated, compresses it, and then transfers the compressed data to the destination Fog. In the second method, the VM has a base snapshot stored on both fog nodes, so that they only transfer the incremental part of the VM's snapshot.

Instead, IOx [123], the Cisco implementation of fog computing, implements both virtualization techniques. IOx works by hosting applications in a Guest Operating System. The platform also supports developers to run applications encapsulated on Docker or Linux Containers, packaged as a virtual machine, or to compile and run Java SE or python scripts.

5.5.2. Storage

Data can be initially stored on the edge or fog nodes in order to speed up their processing, reduce data transfer latency and increase the system reliability. Different approaches are working on storing this information on fog nodes or on different element of the network infrastructure.

On the one hand, the fog platform can handle a local repository storing the data in a non-volatile memory. This repository can store the information on a given node (following a semi-centralized model) or on several nodes (following a distributed model). Each virtualization technology and concrete framework can implement one or both models. OpenStack, for instance, can be complemented with the Cinder and Swift modules to allow the storage of data. Cinder provides persistent block storage to guest virtual machines. This module facilitates the storage of data on a given node, using a centralized model. Cinder virtualizes the management of block storage devices and provides end users with a self-service API to request and consume those resources. Swift functions as a distributed, API-accessible storage platform that can be integrated directly into IoT applications or used to store VM images or archives. It automatically stores redundant copies of each object to maximize availability and scalability. Cloudlet is based on OpenStack++, which is an extension of OpenStack, so that it supports the inclusion of both Cinder and Swift modules. In addition, in [124], the authors present CoSMiC, a cloudlet-based implementation of a hierarchical cloud storage system for mobile devices based on multiple I/O caching layers. The solution relies on Memcached as a cache system, preserving its powerful capacities such as performance, scalability, and quick and portable deployment. Containers, such as Docker or LXC, also provide specific functionalities for storing and caching data. For instance, in [125], each Docker container is isolated and consists of its own independent subsystem of network, memory and file system. For storing data, Docker uses a lightweight file system called UnionFS, improving the overall application performance. Concretely, Flocker is a container data volume manager that can be used by almost any container [125]. Finally, other works, such as Enigma [93], propose the use of a distributed peer-to-peer network to store and run computations on data using blockchain technology.

On the other hand, there are other researches working on caching the users' information on the network infrastructure. This information can be stored reactively, caching the information once the users have asked for it; or proactively, analysing the users' demands on information and pre-caching it. The Content Delivery Network (CDN) [126] represents the most mature catch networks. CDN is the Internet-based cache network by deploying cache servers at the edge of Internet to reduce the download delay of contents from remote sites. Information Centric Network (ICN) [127] is a wireless cache infrastructure which provides content distribution services to mobile users with distributed cache servers. Different from the cache servers, in ICN, the fog servers are intelligent computing unit [7].

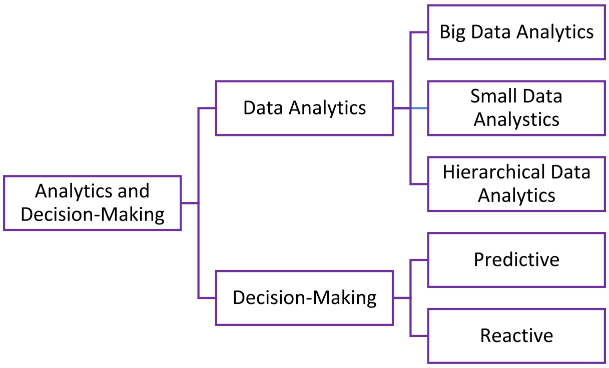

Fig. 8. Taxonomy for the classification of the Analytics and Decision-Making perspective.

5.6. *Analysis and Decision-Making*

The Analytics and Decision-Making perspective offers two fundamental components for the decision-making: Data Analytics and Decision Management. The former analyses all the gathered data in order to identify the different trends and situations. The latter reuses the generated reports in order to identify what business rules and decisions should be made. For the sake of clarity, Fig. 8 summarizes the proposed taxonomy with the possible choices for any of the components.

5.6.1. *Data Analytics*

Data Analytics is the application of advance analytics techniques to data sets in order to identify specific situations [23]. By focusing on where data are analysed, we can divide this component into Big Data Analytics, Small Data Analytics and Hierarchical Data Analytics. Big Data Analytics relies on the computing and storage capabilities of cloud environments to execute complex analytics in big data sets [128]. Small Data Analytics refers to a limited quantity of highly granular data that usually provide valuable information for the system, used to perform real-time decisions and actions, suited to be handled by edge or fog nodes. In Hierarchical Data Analytics, the edge and fog nodes store and analyse the gathered data. Then, the relevant and complex information can be aggregated and posted to other nodes with higher capabilities or, even, to the cloud environment to perform medium or long-term analysis [13].

These methods are applied in the presented case studies. For example, in STL, small data analytics are applied for creating a perfect picture of the current situation and helping the decision-maker component to react in real-time. The system collects environmental information about traffic density, vehicle specific data, movements of other vehicles or pedestrians or bikers on the road, pre-emptive emergency routing, and so on. etc. The fog node should store all this information and execute quick analytics techniques to identify certain movements on the road, trying to understand which movements vehicles are performing and, then, predict where they will probably move. In this sense, Hong et al. [24] introduced the MCEP system for monitoring the traffic through several patterns (e.g. movement, acceleration).

In Wind Farm case, Big Data Analytics are more used, since the real-time is important but also the long-term analytics. In particular, it is important to guarantee a certain accuracy level with wind forecasting techniques, especially short-term forecasting techniques, in order to improve the quality of wind power generators and to schedule appropriate operating levels according to the different regulation tasks [20].

In Smart Grid, hierarchical data analytics models are basic to ensure that the network operates in the right way and to correctly manage dynamic end-user demand and distributed generation sources, favouring promptly reactions in case of unexpected events. Data analytics are key to perform autonomous data control/selection in order to give a consistent feedback on energy usage that can lead to behavioural changes by energy users [59]. In particular, hierarchical data analytics are central to face renewable energy supply unpredictability that may be highly variable in relation to weather conditions, since every intermediate node can act as an active control unit [129].

5.6.2. *Decision-Making*

As the speed on which the gathered data have to be transmitted and processed, the agility on making the decisions to trigger specific business processes and rules in the right moment is crucial and it clearly affects the resource utilization and the customer satisfaction [130]. By focusing on the quickness with which the decisions have to be made, we can divide the Decision-Making component into Predictive and Reactive models. In many applications, the interplay between edge, fog and cloud is directly related with the decisions-making model followed.

Predictive models store all the data gathered in order to get a deep knowledge of the environment and the system and to trigger the most appropriate solutions to each situation or to infer possible evolutions of the system. These models focus on data computation and analytics techniques to find interesting patterns, build descriptive and predictive models. Predictive Systems usually rely more on the cloud in order to collect a great amount of data and perform long-term analysis to identify the different policies that should be executed, to evaluate the results and to improve the predictive analysis. Wind Farm, Smart Building and Smart Grid are scenarios where the prediction is prevalent on reaction phase. Some of the most important techniques applied in these models rely on evolutionary or genetic algorithms. In Wind Farm, for instance, prediction has been identified as an important tool to address the increasing variability and uncertainty. Unit commitment components relay on evolutionary techniques [131] to minimize the operating costs while meeting the total demand bid into the market. This is usually done by a controller that determines global or personalized (for the individual state) policies and pushes them for each sub-system. Of course, they also have some reactive approach in order to increase efficiency and to prevent damage, shutting down the turbines if wind is too low or too strong. Other works make use of Agent Based Model, Multivariate Gaussian Model, Hidden Markov Model and Neural Networks as predictive strategies to foretell the behaviour of the different parts of the system [43]. For instance, in Smart Grid, in [132], Erickson et al. make use of Agent Based and Multivariate Gaussian model to estimate the occupancy in a large multi-function building and for predicting user mobility patterns in order to efficiently control energy usage. Reactive methods can be used simultaneously with the predictive approaches in order to refine the system in case of events and get the best from each situation. In this context, [58] proposes a methodology, based on Neural Networks techniques, that combines distributed generation, distributed storage, and demand-side load management techniques, achieving a better matching of demand and supply.

On the contrary, Reactive models respond in the shortest possible time to different events happened in the environment in order to try to produce corrections as soon as possible. These models act in order to achieve a desired goal interacting with the environment but without predicting the future systems evolutions and solely responding to the present behaviour. Real-time support is a key characteristic of fog that is particularly critical in those systems that require an immediate reaction. To obtain an adequate response time, the closer the fog node is to the edge, the better the response time. Therefore, these systems make use of the Close to the Edge [133] and Location-Awareness [7] strategies. For instance, in STL, low-latency reaction is one of the most important requirements and is crucial to ensure safety. [27] estimates the reaction time must be within a few milliseconds and, in particular, less than 10 ms to be really effective and compliant with safety requirements. In a context like this, the role of fog is crucial to sense the situation, process the data and identify the required actuation in a so limited time. Hence, these systems should exploit the characteristic of fog moving close to sensors/actuators in order to cut the latency [79]. At the same time, the Location-Awareness strategy improves the responsiveness of the system, providing an advance knowledge of the environment during the applications execution; for instance, to react to the nearby traffic light cycle to change the situation or warn the driver. Finally, great amount of information is sent to the Cloud for long-term analytics in order to evaluate the impact on traffic, to monitor city pollution, and the traffic patterns [27].

Most systems have different levels in the decision-making process combining at different degree both models, but our distinction proposal is based on which part is more developed and on which functionality the system is more focused on. Some systems work in context where predictions are key and, thus, they perform intensive data computation and analytics. Instead, for others, the predictions are not relevant and reduce the resources consumption excluding data analytics methods.

6. Comparisons of surveyed solutions

In this section, we compare the surveyed approaches in order to provide some guidelines for building effective fog environments for IoT applications. As in the previous section, we organize our comparison according to our conceptual architecture (see Fig. 2). In addition, Table 1 summarizes the main focus and characteristics supported by the most important surveyed techniques. They can also support other characteristics due to their combination with other solutions, but that characteristics have not been detailed in the table because they are not their main contribution. This table does not include the analysed case studies in which they have being applied, first, in order to improve the readability and to better compare the focus of each solution and, second, because that information is further explained in the following subsections.

6.1. Communication

The first step is the selection of a communication protocol. To that end, four different characteristics should be considered according to the IoT requirements: standardization, reliability, low-latency and mobility. As Table 1 details, different protocols can be selected to standardize the communication either at the network level, or between the different devices or parts of the system. The most important characteristics leading to the selection of one protocol or another are: the heterogeneity of the devices, the communication range, their behaviour under noise conditions and the power consumption. Currently, different protocols can be selected to improve the communication either at the network level, or between the different devices or parts of the system. In fact, many protocols at the application level are based on specific protocols at the infrastructure level. For example, the ZigBee protocol is based on the IEEE 802.15.4 standard. Specifically, by analysing the case studies, we have identified that ZigBee is especially widespread due to its short range and robustness under noise conditions. In addition, BLE has also gained importance recently due to its low power consumption energy, communication range and flexibility.

The selected communication protocol should also meet the reliability requirements in order to ensure the correct operation of the system. An environment capable of ensuring that adequate data will be received, and not lost, guarantee the correct operation of the system in most situations. In this sense, achieving uninterrupted communications among end-devices, fog nodes and cloud is crucial in mobility scenarios. CoAP, MQTT, AMQP and DDS are based on retransmission

Table 1
Comparison between the most important surveyed techniques.

System	Communication				Security			Data Quality			Sen. & Act. Mgmt.		Cloudific.		Analy & DM	
	Standardization	Net. Semantics	Low-Latency	Mobility	Safety	Security	Privacy	Normalization	Filtering	Aggregation	Sensors	Actuators	Virtualization	Storage	Analytics	Decision-Making
CoAP [60]	✔	✔	✔	✔		✔										
MQTT [61]	✔	✔	✔			✔										
AMQP [62]	✔	✔				✔										
DDS [63]	✔	✔	✔													
ZigBee [64]	✔															
UPnP [65]	✔			✔												
UPnP+ [84]	✔			✔												
RPL [67]	✔			✔												
Mosquitto [72]	✔	✔	✔			✔										
Kura [74]	✔	✔	✔			✔										
FCD [79]	✔	✔	✔	✔		✔			✔	✔						✔
Per-Hop [76]	✔	✔														
6LoWPAN [68]	✔															
BLE [69]	✔															
LTE-Adv [70]	✔															
LISP [71]	✔			✔												
DPWS [66]	✔	✔	✔	✔		✔										
Act. Coord [20]					✔											
Act. Monit [20]					✔											
Act. Planning [20]					✔											
Access Contr. [92]							✔									
Data Encript [25]						✔										
Conf. Mang. [87,88]						✔										
Rept. Systems [90]						✔										
Role Commu. [17]						✔	✔	✔								
Enigma [93]						✔	✔							✔		
Virt. Sensors [105,106]											✔					
Dyn. Resour. [105,106]											✔					
Data discov. [107]											✔					
Data mining [108]											✔					
Virtual Res. [112]											✔	✔				
Fault control [113]											✔	✔				
FDI and FTC [56]											✔	✔				
SVM [95]								✔								
Pigi [97]								✔								
G. Buffers [98]								✔								
NIST [99]								✔								
Spatial-Temp [21]									✔	✔						
Arithmet. Opt [22]									✔	✔						
Data Forw. [103]									✔							
Saf. Data Agg [24]					✔					✔					✔	
WNA – WT [24,55,101]							✔			✔						
LXC [117]													✔	✔		
Cloudlet [33]													✔	✔		
Zi. Et al. [28]													✔	✔		
IOx [123]													✔	✔		
CoSMiC [124]														✔		
CDN [126]														✔		
ICN [127]														✔		
Big data ana [128]															✔	
MCEP [24]															✔	
DQA [129]										✔					✔	
Evo. Tech. [131]															✔	✔
Neur. Net. [58]														✔	✔	✔
EBE [132]															✔	✔
C2E [133]															✔	✔

techniques. Algorithms such as MQTT and AMPQ also incorporate handshaking techniques to ensure such network semantics. DDS also implements multicasting techniques. In addition, frameworks and libraries such as Mosquito, Kura or FCD implement and integrate some of these algorithms to improve the communication. These techniques are specially used in critical cases studies, such as Smart Traffic Light, being DDS one of the most used protocols according the surveyed solutions. Nevertheless, as Table 1 shows, all the network semantics techniques identified in the surveyed solutions are applied by application protocols.

The low-latency not only have to be achieved between the IoT devices and the fog nodes, but also between the fog and the cloud in order to achieve highly responsive applications. CoAP, MQTT and DDS are protocols supporting a low-latency communication between the different nodes. In addition, MQTT proposes to use different versions, one for the communication between the devices and the fog and another interacting with the cloud. Finally, DDS has been proposed and evaluated in different environments related to Smart Grid and Smart Traffic. Like network semantics, in the surveyed approaches, low-latency is mainly considered by the application protocols.

The protocols and frameworks detailed above can incorporate different techniques to facilitate the devices mobility. Mobility support must be provided by both the infrastructure and the application protocols. RPL and LISP provide routing capabilities. In addition, CoAP, UPnP and DPWS facilitate devices to discover the services and resources available in any new context. Analysing the case studies, this property is crucial in STL environments. RPL has already been applied in these environments and it is spreading to other environments.

From the analysis of Table 1, we can identify that frameworks, such as FCD and Kura, are the ones covering a higher number or characteristics. Nevertheless, the most used protocols in the surveyed case studies are ZigBee, DDS and RPL.

Therefore, protocols supporting all the identified characteristics or the application of the surveyed frameworks to real cases are still needed.

6.2. Security

Secondly, techniques ensuring the security, privacy and safety of the system and its data are required. The security of the communications among devices, fog nodes and the cloud are initially provided by the communication protocols. MQTT, AMQP and ADQP (or the libraries implementing them) support different techniques to encrypt the data. Other protocols, such as CoAP, also include techniques to improve the data integrity. On the other hand, security also has to be provided by each system. [17] details a smart grid application applying confidentiality and integrity techniques. As Table 1 shows, the approaches and case studies evaluated mainly focus on the confidentiality and integrity of the data; first, to ensure that all communications are made by authorized persons; and, second, to control that the exchanged information is not modified. Therefore, more distributed and internetworked security approaches are required to create more complete and responsive solutions.

In all case studies, privacy plays a fundamental role. They store and analyse very sensitive information and any unauthorized access to it entails a great risk for the IoT systems and for their users. From the analysed solutions, [17] includes mechanisms for the data privacy control, focusing mainly on the security and the safety of the applications. In [93], the authors propose a platform that combines SMPC and Blockchain to securely store and compute the gathered information. Nonetheless, only specific approaches address the security and privacy characteristics together. Novel works contributing to these two characteristics and oriented to fog are still needed.

Finally, some IoT applications operate in critical environments. The fog should provide mechanisms for implementing safety policies and procedures ensuring the correct operation in anomalous situations. Once analysed the surveyed approaches, we have identified that those case studies requiring a real-time actuation usually apply two techniques: activity monitoring and activity coordination, to exactly know the state of each element and trigger coordinated actions to meet the system's goals. However, in case studies where the response should not be in real-time, such as Wind Farm [20,86], there is a greater emphasis on the use of planned action techniques, since it allows a fine planning of every action. Nonetheless, as can be seen in Table 1, among the surveyed approaches we have not identified works contributing to the three components of these perspective. These characteristics are highly related and frameworks supporting all of them are still needed.

6.3. Data Quality

The Data Quality perspective provides a uniform specification to all the information, gets rid of the useless data as soon as possible and reduces the amount of data stored in the fog or transmitted to other nodes or to the cloud. IoT systems typically involve a large number of sensors, actuators, nodes, etc. (in many cases these elements are redundant in order to improve the system reliability). Therefore, the normalization of the exchanged data is a key step for all these systems. From the case studies analysed, we have identified that current solutions [17,59,77,99] are usually based on proposing common languages (improving the communication between different devices or nodes) or, even, some serialization mechanisms (reducing the resource consumption). Few solutions include in this component the data homogenization techniques. This is probably because this step can also be relegated to later phases (during data filtering and aggregation).

To get rid of the useless or wrong data, data filtering is one of the most commonly used mechanism. Currently, there are a large number of solutions applying techniques for identifying duplicates and, above all, for detecting outliers [22]. This is not the case with data prioritization since only a couple of works implement it [79,103]. It is specifically exploited in environments where different parts of the system may have a very different priority [103]. Nevertheless, the ability to filter critical data could also be provided by the fog, improving data communication and the responsiveness of the deployed applications. In a multi-perspective and hierarchical fog architecture, like the presented in this paper, the capability of filtering the data to be processed in other nodes depending on their criticality, or the resources they require, is crucial for improving the system's latency, responsiveness, reliability and scalability.

In addition, to further reduce the amount of stored and transmitted information, data aggregation techniques are used. The vast majority of the techniques analysed [24,55,101] (Table 1) focus on data fusion, since it is the central part of this component. Different techniques, such as [101], also make use of a hierarchical aggregation, but this implies that either general aggregation techniques are deployed in every node or the systems' managers have to perfectly know the resources of each node and its location. Finally, the safety data aggregation is another technique widely extended in critical systems [24].

Therefore, as can be seen in Table 1, approaches exploiting the characteristic of a hierarchical fog architecture and facilitating the standardization, filtering and aggregation of data in different nodes depending on their capabilities are necessary.

6.4. Sensing and actuation management

Fourth, an essential part of every IoT application is sensing the environment and acting according to its status. As Table 1 shows, sensing is a property performed by all the analysed case studies. All systems need to gather information from the environment to obtain results, analyse the situation and make decisions. Typically, these case studies use information from both physical sensors and virtual sensors. Some systems, such as [107] and [72], only need the sensing component, since they do not have an actuation phase. Typically, these systems require such information to perform a thorough analysis of the environment, for generating reports or for visualizing results.

Finally, those systems that require real-time performance mix the sensing phase with a strong actuation phase. Thus, the actuators can automatically or semi-automatically change the environments to achieve the desired goals. All these systems obviously require physical actuators. Nevertheless, some of them, [56,113], also use virtual actuators to hide the low-level information to interact with the physical actuators or to substitute faulty actuators in order to allow systems to continue with their usual execution.

As the number of deployed internet-connected devices increases, a higher number of techniques for their coordination would be required, some of the analysed approaches provide these functionalities but some work is still needed to be able to coordinate heterogeneous devices.

6.5. Cloudification

Fifth, in order for the fog to act as a small-distributed cloud, different virtualization and storage capabilities should be offered. Virtualization techniques allow the deployment of different IoT applications. Hypervisor is the most extended technology [29,33,123], because of its flexibility and the number of IoT applications that can be deployed using it, since the required operating system could be included in the VM. Nonetheless, for environments in which the fog nodes have fewer computing capabilities or the kind of applications to deploy are known, the containers technology presents additional advantages [123]. Analysing the surveyed solutions, the general approaches usually implement the hypervisor technology or, even, both. Approaches that are more specific implement containers. In addition, almost every platform allows the migration of VM instances. Usually, the containerized applications are completely migrated, while the platforms implementing the hypervisor technology usually allow a complete and a partial migration. This is due to the larger size of the hypervisor VM images. It should be highlighted that some approaches even propose some efficient algorithms to migrate the images using the IoT devices to transmit the information without overloading the network.

Regarding the storage of information, all fog platforms allow data storage. Normally, the data storage approaches implemented follows a centralized model, storing the data in the fog node. Nevertheless, different approaches [33,93,124] also implements distributed information storage. This improves the mobility of the users along the network and the storage capacity, but increases the network overload. Moreover, other approaches are working on directly store data on the network infrastructure [126,127]. These solutions allow the deployment of a large number of servers to cache the information in a distributed way. These approaches usually do not implement any method for the deployment of IoT applications, but they can be used to store the information produced by IoT applications.

These techniques allow the deployment, migration and storage of IoT applications and their data. Approaches orchestrating and composing the different services provided by these applications, depending on the context, location and status of the application, and oriented to fog architectures are still needed.

6.6. Analysis and Decision-Making

Finally, the Data Analysis and Decision-Making is an essential part to any IoT application for identifying the correct processes, rules or tasks to trigger. To that end, first, every IoT system must perform an analysis of the data obtained. Depending on the volume of data to be analysed and the complexity of the techniques to apply, the geo-distribution of the data analysis should be considered, taking into account if it is going to be executed in the fog, in the cloud or in both [13,128]. Regarding the case studies, we have confirmed that environments requiring real-time responsiveness, perform a higher number of data analytics in the fog [24]. Instead, those systems, such as Wind Farms, needing long-term analysis and forecasts, relegate the data analytic to the cloud environments [59]. Nevertheless, in most cases, the combination of both, Small Data Analytics and Big Data Analytics, is essential to obtain an optimum performance in any situation [129].

Secondly, once the gathered data have been analysed, different decisions should be made. Again, where to execute this component largely depends on the required response time. Those systems with a stringent response time would perform the most important part of the decision-making process in the fog nodes. In contrast, for those systems with more relaxed responsiveness constraints, this component will be relegated to the cloud [130]. Wind Farm and Smart Grid are scenarios where the temporal requirements are more or less relaxed [131]. Instead, in critical systems, such as STL, low-latency reactions is one of the most important requirements and is essential to ensure safety [27]. Therefore, the data analytics and the decision-making processes are usually executed in fog nodes close to the edge. Nevertheless, most systems have different levels in the decision-making process combining both models at different degree [133].

6.7. Additional research directions

Fog is a powerful computational paradigm that is able to boost IoT applications, but many challenges, both application-specific and general-purpose across applications, still have to be addressed to turn effective fog solutions into reality. From the surveyed approaches, some trends and novel works that should be further researched are:

- Multi-levels organization. These are groups of nodes densely connected. They may also have an internal organization of sub-groups of nodes. In such a multi-level organization, each node may have a specific role and responsibility. Usually, in real-world applications, fog nodes should be structured into a hierarchical organization or into a mesh/cluster of nodes, with associated load balancing and consequent stronger scalability and reliability. Some case studies are already applying this trend is Smart Grid and Smart Traffic.
- Node specialization. In Multi-level organization, nodes can be specialized and optimized to perform a specific work. Each application must design fog nodes to optimize the overall system operations. For instance, in Smart Traffic, fog nodes have to manage fast mobility in wide geographical areas and, in contrast, high level nodes controlling the traffic lights in an area should be able to manage a huge amount of information.
- Context-awareness. IoT application and fog environment should be able to identify the context and adapt their behaviours to the specific situation. Different behaviour and reactions could be defined for different contexts, even taking into account the fog–cloud interplay (choosing the type of interactions or the communication algorithm that better suits the specific situation).
- Efficient load balancing. Fog computing must manage a huge amount of data that have to be processed with multiple components in a cost-effective way. Different load and computation balancing techniques could be used to distribute or delegate some computation/storage tasks to more powerful nodes.
- Interworking of different fog localities. A current challenge is to define the way fog localities should coordinate and interwork to achieve more global objectives, by leveraging also virtual networking techniques and support. Through the interconnection of different networks or nodes spread in different locations, it could be possible to extend the sensing/actuation/computation phases within a wider area, providing services in a more pervasive manner.

In conclusion, we believe fog is a promising concept that has the potentiality to be an enabler and a significant driver for IoT environments. Further research is needed and many challenges have to be solved to support the deployment of critical and dynamic real-world IoT applications.

6.8. A quantitative and summarized comparison

In this paper we evaluated 56 techniques applied to more than 35 case studies from three different domains (Smart Traffic Light, Wind Farm and Smart Grid). From these techniques, 16 are focused on the communication perspective. This is the perspective with the largest number of mechanisms surveyed, by showing the primary role that pervasive and mobile communications play a basic feature for any IoT system. All these proposals push towards the suitability and need for communication standardization; a lower number of them contribute to the network semantics and low-latency (it should be highlighted that, in the evaluated works, both characteristics are usually supported together); and only six of them concentrate their effort on the support of device mobility.

For the Security perspective, we analysed nine approaches, which are mainly focused on Security and Privacy. Security mechanisms are often covered also by a larger number of communications mechanisms (six from the sixteen evaluated and discussed above) in order to achieve secure data transmission.

Nine approaches focused on the Data Quality perspective were surveyed. Between these techniques, there is a clear separation between those supporting data normalization, and those covering data filtering and aggregation. Two of these approaches also contribute to the Security perspective, since data management and its security and privacy characteristics are often highly related. Some of these works also support the Data Analytics category of our taxonomy, from the Analysis and Decision-Making perspective, due to the narrow gap between the two perspectives. Note that these relations among different perspectives prove that specific characteristics of different perspectives can be distributed between different nodes.

For the Sensing and Actuation Management, we analysed seven different techniques. All of them cover the sensing characteristic, thus showing that this is widely considered a basic functionality for every IoT system. Instead, the actuation phase is only supported by half of the evaluated approaches. In addition, we evaluated seven mechanisms for the Cloudification perspective. All of them support the data storage in the fog nodes but only half of them allow developers to virtualize their applications in these nodes. This is reasonable since the storage and treatment of the sensed data can be done in more or less powerful fog nodes and reduces network load in several applications scenarios; however, the deployment of virtualized applications is associated with higher complexity in computing management and computationally richer participating nodes. Finally, for the Analysis and Decision-Making perspective, seven different approaches were surveyed, from which two works also support some characteristics of the Cloudification and Data Quality perspective due to the need to store and process the data at different levels of the architecture.

As illustrated by Table 1, the vast majority of the surveyed research proposals are focused on a concrete perspective. Only thirteen of the analysed works integrate multiple perspectives, and only two of them provide support to more than two perspective. The distribution of the surveyed works and their relationship with the presented conceptual architecture and taxonomy re-inforce the suitability/correctness of our architecture and taxonomy proposals and show a reliable design and categorization of the components and characteristics usually required by IoT solutions. Nevertheless, it also shows that mechanisms integrating and supporting multiple perspective, or even our complete architecture, are strongly recommendable and needed to leverage the widespread industrial adoption of fog computing techniques in the IoT application domain.

7. Conclusion and ongoing research works

Cloud computing has led to a revolution in how devices interact with the Internet, allowing almost any device to interact with the environment, adapt their behaviour, obtain complex information, and so on. This revolution has enabled the development of the IoT paradigm and the deployment of a myriad of internet-connected devices with enough capabilities to be constantly sensing or acting according to the users' needs. However, the cloud environments and the network infrastructure cannot withstand the increasing communication and processing load that these systems require. In the last few years, different approaches have been proposed to overcome these limitations.

Fog Computing has been one of the paradigms that more importance and relevance has acquire. Currently, there are a lot of solutions improving the communication between devices, the data security and privacy, the data quality or, even, how the applications react to the environment. Nevertheless, the majority of these solutions are oriented to improve a specific characteristic of the fog vision, or are adapted to specific environments. In this paper, we have analysed the main IoT applications requirements. We have defined a unified model of a fog platform meeting the analysed requirements and a taxonomy in order to be able to compare different solutions and how they are applied by IoT applications to specific domains. This allowed us, first, to identify some areas and characteristics in which new proposals are needed; second, to provide a complete overview of the different proposals and how they can be integrated; and, third, to establish some guidelines on what kind of solutions can be used depending on the requirements and the specific environment of an IoT application. Therefore, the outcomes of this work can be reused by researches, and by developers that are designing IoT applications based on fog computing.

As future work, currently, we work on increasing the number of analysed solutions and IoT environments in which they are applied. Three very important environment in which a lot of IoT applications are been develop are Smart Connected Vehicle and Smart Building, for instance. We are currently analysing how the different solutions, and other proposals, are being applied in these environments.

Acknowledgements

This research was supported by the Emilia Romagna POR-FESR 2014–20 (Sacher project, no. J32I16000120009), by the Spanish Ministry of Economy, Industry and Competitiveness (TIN2015-69957-R (MINECO/FEDER)), by the Interreg V-A España-Portugal (POCTEP) 2014–2020 program (4IE project, 0045-4IE-4-P), by the Department of Economy and Infrastructure of the Government of Extremadura (GR15098), and by the European Regional Development Fund.

References

[1] ITU Telecommunication Development Bureau, ICT facts and figures, 2015.

[2] ITU Telecommunication Development Bureau, ICT facts and figures, 2017.

[3] Cisco, Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017, pp. 2016–2021.

[4] C. Perera, C.H. Liu, S. Jayawardena, M. Chen, Context-aware Computing in the Internet of Things: A survey on Internet of Things from industrial market perspective, CoRR, 2015.

[5] L. Atzori, A. Iera, G. Morabito, The internet of things: a survey, Comput. Netw. 54 (2010) 2787–2805, http://dx.doi.org/10.1016/j.comnet.2010.05.010.

[6] E. Borgia, The internet of things vision: key features, applications and open issues, Comput. Commun. 54 (2014) 1–31, http://dx.doi.org/10.1016/j.comcom.2014.09.008.

[7] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proc. First Ed. MCC Workshop Mob. Cloud Comput, ACM, New York, NY, USA, 2012, pp. 13–16, http://dx.doi.org/10.1145/2342509.2342513.

[8] P. Bellavista, A. Corradi, E. Magistretti, REDMAN: An optimistic replication middleware for read-only resources in dense MANETs, Pervasive Mob. Comput. 1 (2005) 279–310, http://dx.doi.org/10.1016/j.pmcj.2005.06.002.

[9] I. Stojmenovic, Fog computing: A cloud to the ground support for smart things and machine-to-machine networks, in: 2014 Australas. Telecommun. Netw. Appl. Conf. ATNAC, 2014, pp. 117–122. http://dx.doi.org/10.1109/ATNAC.2014.7020884.

[10] W. Wang, K. Lee, D. Murray, Integrating sensors with the cloud using dynamic proxies, in: 2012 IEEE 23rd Int. Symp. Pers. Indoor Mob. Radio Commun. - PIMRC, 2012, pp. 1466–1471.http://dx.doi.org/10.1109/PIMRC.2012.6362579.

[11] I. Podnar Zarko, A. Antonic, K. Pripužic, Publish/Subscribe middleware for energy-efficient mobile crowdsensing, in: Proc. 2013 ACM Conf. Pervasive Ubiquitous Comput. Adjun. Publ, ACM, New York, NY, USA, 2013, pp. 1099–1110, http://dx.doi.org/10.1145/2494091.2499577.

[12] N.D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, A.T. Campbell, A survey of mobile phone sensing, IEEE Commun. Mag. 48 (2010) 140–150, http://dx.doi.org/10.1109/MCOM.2010.5560598.

[13] M. Yannuzzi, R. Milito, R. Serral-Gracià, D. Montero, M. Nemirovsky, Key ingredients in an IoT recipe: Fog computing, cloud computing, and more Fog Computing, in: 2014 IEEE 19th Int. Workshop Comput. Aided Model. Des. Commun. Links Netw. CAMAD, 2014, pp. 325–329. http://dx.doi.org/10.1109/CAMAD.2014.7033259.

[14] J.G. Andrews, S. Buzzi, W. Choi, S.V. Hanly, A. Lozano, A.C.K. Soong, J.C. Zhang, What will 5G be, IEEE J. Sel. Areas Commun. 32 (2014) 1065–1082, http://dx.doi.org/10.1109/JSAC.2014.2328098.

[15] R. Vilalta, A. Mayoral, R. Casellas, R. Martínez, R. Muñoz, Experimental demonstration of distributed multi-tenant cloud/fog and heterogeneous SDN/NFV orchestration for 5G services, in: 2016 Eur. Conf. Netw. Commun. EuCNC, 2016, pp. 52–56. http://dx.doi.org/10.1109/EuCNC.2016.7561003.

[16] F. van Lingen, M. Yannuzzi, A. Jain, R. Irons-Mclean, O. Lluch, D. Carrera, J.L. Perez, A. Gutierrez, D. Montero, J. Marti, R. Maso, a.J.P. Rodriguez, The unavoidable convergence of NFV, 5G, and Fog: A model-driven approach to bridge cloud and edge, IEEE Commun. Mag. 55 (2017) 28–35, http://dx.doi.org/10.1109/MCOM.2017.1600907.

[17] E. Ancillotti, R. Bruno, M. Conti, The role of communication systems in smart grids: architectures, technical solutions and research challenges, Comput. Commun. 36 (2013) 1665–1697, http://dx.doi.org/10.1016/j.comcom.2013.09.004.

[18] D. Thangavel, X. Ma, A. Valera, H.X. Tan, C.K.Y. Tan, Performance evaluation of MQTT and CoAP via a common middleware, in: 2014 IEEE Ninth Int. Conf. Intell. Sens. Sens. Netw. Inf. Process. ISSNIP, 2014, pp. 1–6. http://dx.doi.org/10.1109/ISSNIP.2014.6827678.

[19] H. Gupta, S. Chakraborty, S.K. Ghosh, R. Buyya, Fog computing in 5G networks: an application perspective, 2017, pp. 23–56, http://dx.doi.org/10.1049/PBTE070E_ch2.

[20] F. Bouffard, F.D. Galiana, Stochastic security for operations planning with significant wind power generation, IEEE Trans. Power Syst. 23 (2008) 306–316, http://dx.doi.org/10.1109/TPWRS.2008.919318.

[21] S. Tyagi, A.Q. Ansari, M.A. Khan, Dynamic threshold based sliding-window filtering technique for RFID data, in: 2010 IEEE 2nd Int. Adv. Comput. Conf. IACC, 2010, pp. 115–120. http://dx.doi.org/10.1109/IADCC.2010.5423025.

[22] M. Gupta, K.R. Krishnanand, H.D. Chinh, S.K. Panda, Outlier detection and data filtering for wireless sensor and actuator networks in building environment, in: 2015 IEEE Int. Conf. Build. Effic. Sustain. Technol. 2015, pp. 95–100. http://dx.doi.org/10.1109/ICBEST.2015.7435872.

[23] Varun.G. Menon, Varun g menon moving from vehicular cloud computing to vehicular fog computing: issues and challenges, Int. J. Comput. Sci. Eng. 9 (2017) 14–18.

[24] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, B. Koldehofe, Mobile Fog: A programming model for large-scale applications on the internet of things, in: Proc. Second ACM SIGCOMM Workshop Mob. Cloud Comput, ACM, New York, NY, USA, 2013, pp. 15–20, http://dx.doi.org/10.1145/2491266.2491270.

[25] S. Yi, Z. Qin, Q. Li, Security and privacy issues of fog computing: a survey, in: K. Xu, H. Zhu (Eds.), Wirel. Algorithms Syst. Appl. 10th Int. Conf. WASA 2015 Qufu China August 10-12 2015 Proc, Springer International Publishing, Cham, 2015, pp. 685–695, http://dx.doi.org/10.1007/978-3-319-21837-3_67.

[26] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, Internet of things: a survey on enabling technologies, protocols, and applications, IEEE Commun. Surv. Tutor. 17 (2015) 2347–2376, http://dx.doi.org/10.1109/COMST.2015.2444095.

[27] A. Rahmani, P. Liljeberg, J.-S. Preden, A. Jantsch (Eds.), Fog Computing in the Internet of Things: Intelligence at the Edge, Springer International Publishing, 2018, www.springer.com/gp/book/9783319576381 (Accessed 17 July 2018).

[28] F. Bonomi, R. Milito, P. Natarajan, J. Zhu, Fog Computing: A platform for internet of things and analytics, in: Big Data Internet Things Roadmap Smart Environ, Springer, Cham, 2014, pp. 69–186, http://dx.doi.org/10.1007/978-3-319-05029-4_7.

[29] S. Yi, Z. Hao, Z. Qin, Q. Li, Fog Computing: Platform and Applications, in: 2015 Third IEEE Workshop Hot Top. Web Syst. Technol. HotWeb, 2015, pp. 73–78. http://dx.doi.org/10.1109/HotWeb.2015.22.

[30] C. Mouradian, D. Naboulsi, S. Yangui, R.H. Glitho, M.J. Morrow, P.A. Polakos, A comprehensive survey on fog computing: state-of-the-art and research challenges, IEEE Commun. Surv. Tutor. 20 (2018) 416–464, http://dx.doi.org/10.1109/COMST.2017.2771153.

[31] P. Hu, S. Dhelim, H. Ning, T. Qiu, Survey on fog computing: architecture, key technologies, applications and open issues, J. Netw. Comput. Appl. 98 (2017) 27–42, http://dx.doi.org/10.1016/j.jnca.2017.09.002.

[32] R.K. Naha, S. Garg, D. Georgakopoulos, P.P. Jayaraman, L. Gao, Y. Xiang, R. Ranjan, Fog computing: Survey of trends, architectures, requirements, and research directions, ArXiv180700976 Cs. 2018, http://arxiv.org/abs/1807.00976 (Accessed 6 September 2018).

[33] R. Al Ali, I. Gerostathopoulos, I. Gonzalez-Herrera, A. Juan-Verdejo, M. Kit, B. Surajbali, An architecture-based approach for compute-intensive pervasive systems in dynamic environments, in: Proc. 2Nd Int. Workshop Hot Top. Cloud Serv. Scalability, ACM, New York, NY, USA, 2014, pp. 3:1–3:6, http://dx.doi.org/10.1145/2649563.2649577.

[34] M. Firdhous, O. Ghazali, S. Hassan, Publications, Fog computing: Will it be the Future of Cloud Computing?, in: 2014.

[35] R. Bifulco, M. Brunner, R. Canonico, P. Hasselmeyer, F. Mir, Scalability of a mobile cloud management system, in: Proc. First Ed. MCC Workshop Mob. Cloud Comput, ACM, New York, NY, USA, 2012, pp. 17–22, http://dx.doi.org/10.1145/2342509.2342514.

[36] S. Davy, J. Famaey, J. Serrat, J.L. Gorricho, A. Miron, M. Dramitinos, P.M. Neves, S. Latre, E. Goshen, Challenges to support edge-as-a-service, IEEE Commun. Mag. 52 (2014) 132–139, http://dx.doi.org/10.1109/MCOM.2014.6710075.

[37] Open Edge Computing, (n.d.). http://openedgecomputing.org/ (Accessed 29 October 2017).

[38] OpenFog Consortium. OpenFog Reference Architecture for Fog Computing. (n.d.). https://www.openfogconsortium.org/wp-content/uploads/OpenFog_Reference_Architecture_2_09_17-FINAL.pdf (Accessed 5 September 2018).

[39] S. Dahmen-Lhuissier, Multi-access Edge Computing, ETSI. (n.d.). http://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing (Accessed 29 October 2017).

[40] S. Yi, C. Li, Q. Li, A survey of fog computing: concepts, applications and issues, in: Proc. 2015 Workshop Mob. Big Data, ACM, New York, NY, USA, 2015, pp. 37–42, http://dx.doi.org/10.1145/2757384.2757397.

[41] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, E. Riviere, Edge-centric computing: vision and challenges, SIGCOMM Comput. Commun. Rev. 45 (2015) 37–42, http://dx.doi.org/10.1145/2831347.2831354.

[42] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, Edge computing: vision and challenges, IEEE Internet Things J. 3 (2016) 637–646, http://dx.doi.org/10.1109/JIOT.2016.2579198.

[43] K. Tammemäe, A. Jantsch, A. Kuusik, J.-S. Preden, E. Õunapuu, Self-aware fog computing in private and secure spheres, in: Fog Comput. Internet Things, Springer, Cham, 2018, pp. 71–99, http://dx.doi.org/10.1007/978-3-319-57639-8_5.

[44] M. Aazam, M. St-Hilaire, C.-H. Lung, I. Lambadaris, E.-N. Huh, IoT resource estimation challenges and modeling in Fog, in: Fog Comput. Internet Things, Springer, Cham, 2018, pp. 17–31, http://dx.doi.org/10.1007/978-3-319-57639-8_2.

[45] K. Vatanparvar, M.A.A. Faruque, Control-as-a-service in cyber-physical energy systems over Fog computing, in: Fog Comput. Internet Things, Springer, Cham, 2018, pp. 123–144, http://dx.doi.org/10.1007/978-3-319-57639-8_7.

[46] S. Diallo, H. Herencia-Zapana, J.J. Padilla, A. Tolk, Understanding interoperability, 2011, pp. 84–91.

[47] K. Fu, T. Kohno, D. Lopresti, E. Mynatt, K. Nahrstedt, S. Patel, D. Richardson, B. Zorn, Safety, Security, and Privacy Threats Posed by Accelerating Trends in the {Internet of Things}, Computing Community Consortium, 2017. http://cra.org/ccc/wp-content/uploads/sites/2/2017/02/Safety-Security-and-Privacy-Threats-in-IoT.pdf. (Accessed 5 February 2018).

[48] L.M. Vaquero, L. Rodero-Merino, Finding your way in the fog: towards a comprehensive definition of fog computing, SIGCOMM Comput. Commun. Rev. 44 (2014) 27–32, http://dx.doi.org/10.1145/2677046.2677052.

[49] A. Toninelli, S. Pantsar-Syväniemi, P. Bellavista, E. Ovaska, Supporting Context Awareness in Smart Environments: A Scalable Approach to Information Interoperability, in: Proc. Int. Workshop Middlew. Pervasive Mob. Embed. Comput, ACM, New York, NY, USA, 2009, pp. 5:1–5:4, http://dx.doi.org/10.1145/1657127.1657134.

[50] P. Bellavista, M. Cinque, D. Cotroneo, L. Foschini, Integrated support for handoff management and context awareness in heterogeneous wireless networks, in: Proc. 3rd Int. Workshop Middlew. Pervasive Ad-Hoc Comput, ACM, New York, NY, USA, 2005, pp. 1–8, http://dx.doi.org/10.1145/1101480.1101495.

[51] H. Madsen, B. Burtschy, G. Albeanu, F. Popentiu-Vladicescu, Reliability in the utility computing era: Towards reliable Fog computing, in: 2013 20th Int. Conf. Syst. Signals Image Process. IWSSIP, 2013, pp. 43–46. http://dx.doi.org/10.1109/IWSSIP.2013.6623445.

[52] I. Stojmenovic, S. Wen, The fog computing paradigm: scenarios and security issues, in: Comput. Sci. Inf. Syst. FedCSIS 2014 Fed. Conf. On, IEEE, 2014, pp. 1–8, http://dx.doi.org/10.15439/2014f503.

[53] Ronnie Burns, Method and system for providing personalized traffic alerts, US6590507 B2, n.d. http://patft.uspto.gov/netacgi/nph-Parser?Sect2=PTO1{&}Sect2=HITOFF{&}p=1{&}u=/netahtml/PTO/search-bool.html{&}r=1{&}f=G{&}l=50{&}d=PALL{&}RefSrch=yes{&}Query=PN/6590507.

[54] T. Semertzidis, K. Dimitropoulos, A. Koutsia, N. Grammalidis, Video sensor network for real-time traffic monitoring and surveillance, IET Intell. Transport. Syst. 4 (2010) 103–112, http://dx.doi.org/10.1049/iet-its.2008.0092.

[55] M.A. Ahmed, Y.-C. Kim, Wireless communication architectures based on data aggregation for internal monitoring of large-scale wind turbines, Int. J. Distrib. Sens. Netw. 12 (2016) 1550147716662776, http://dx.doi.org/10.1177/1550147716662776.

[56] J. Blesa, D. Rotondo, V. Puig, F. Nejjari, FDI and FTC of wind turbines using the interval observer approach and virtual actuators/sensors, Control Eng. Pract. 24 (2014) 138–155, http://dx.doi.org/10.1016/j.conengprac.2013.11.018.

[57] V.C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, G.P. Hancke, Smart grid technologies: communication technologies and standards, IEEE Trans. Ind. Inform. 7 (2011) 529–539, http://dx.doi.org/10.1109/TII.2011.2166794.

[58] A. Molderink, V. Bakker, M.G.C. Bosman, J.L. Hurink, G.J.M. Smit, Management and control of domestic smart grid technology, IEEE Trans. Smart Grid 1 (2010) 109–119, http://dx.doi.org/10.1109/TSG.2010.2055904.

[59] H. Chen, P. Chou, S. Duri, H. Lei, J. Reason, The design and implementation of a smart building control system, in: 2009 IEEE Int. Conf. E-Bus. Eng. 2009, pp. 255–262. https://dx.doi.org/10.1109/ICEBE.2009.42.

[60] CoAP — Constrained Application Protocol, (n.d.) http://coap.technology/ (Accessed 24 October 2017).

[61] MQTT - Message Queue Telemetry Transport, (n.d.). http://mqtt.org/ (Accessed 24 October 2017).

[62] AMQP - Advanced Message Queuing Protocol, (n.d.) https://www.amqp.org/ (Accessed 24 October 2017).

[63] DDS – Data Distribution Services, (n.d.) http://portals.omg.org/dds/ (Accessed 24 October 2017).

[64] Zigbee, (n.d.). http://www.zigbee.org/ (Accessed 24 October 2017).

[65] ISO, UPnP - ISO/IEC 29341-1:2011 Device Architecture, (n.d.). https://www.iso.org/standard/57195.html (Accessed 24 October 2017).

[66] OASIS, Devices Profile for Web Services Version 1.1, (n.d.). http://docs.oasis-open.org/ws-dd/dpws/wsdd-dpws-1.1-spec.html (Accessed 24 July 2018).

[67] Tim Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.P. Vasseur, R. Alexander, RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, (n.d.). https://tools.ietf.org/html/rfc6550 (Accessed 24 October 2017).

[68] IEFT WG, 6LoWPAN - IPv6 over Low-Power Wireless Area Networks, (n.d.). http://6lowpan.tzi.org/ (Accessed 24 October 2017).

[69] BLE - Bluetooth Low Energy, (n.d.). https://www.bluetooth.com (Accessed 24 October 2017).

[70] LTE Advanced, Qualcomm. (2014). https://www.qualcomm.com/invention/technologies/lte/advanced (Accessed 24 October 2017).

[71] D. Farinacci, D. Lewis, D. Meyer, V. Fuller, LISP - The Locator/ID Separation Protocol, (n.d.). https://tools.ietf.org/html/rfc6830 (Accessed 24 October 2017).

[72] Mosquitto, An Open Source MQTT v3.1 Broker, (n.d.). https://mosquitto.org/ (Accessed 7 February 2018).

[73] P. Bellavista, C. Giannelli, R. Zamagna, The pervasive environment sensing and sharing solution, Sustainability 9 (2017) 585, http://dx.doi.org/10.3390/su9040585.

[74] Kura, Eclipse Kura TM - Open Source framework for IoT, (n.d.). https://eclipse.org/kura/ (Accessed 7 February 2018).

[75] R. Frank, W. Bronzi, G. Castignani, T. Engel, Bluetooth Low Energy: An alternative technology for VANET applications, in: 2014 11th Annu. Conf. Wirel. -Demand Netw. Syst. Serv. WONS, 2014, pp. 104–107. http://dx.doi.org/10.1109/WONS.2014.6814729.

[76] O. Gnawali, M. Yarvis, J. Heidemann, R. Govindan, Interaction of retransmission, blacklisting, and routing metrics for reliability in sensor network routing, in: 2004 First Annu. IEEE Commun. Soc. Conf. Sens. Ad Hoc Commun. Netw. 2004 IEEE SECON 2004, 2004, pp. 34–43. http://dx.doi.org/10.1109/SAHCN.2004.1381900.

[77] J.K. Zao, T.T. Gan, C.K. You, S.J.R. Méndez, C.E. Chung, Y.T. Wang, T. Mullen, T.P. Jung, Augmented brain computer interaction based on fog computing and linked data, in: 2014 Int. Conf. Intell. Environ. 2014, pp. 374–377. http://dx.doi.org/10.1109/IE.2014.54.

[78] B.N. Amel, B. Rim, J. Houda, H. Salem, J. Khaled, Flexray versus Ethernet for vehicular networks, in: 2014 IEEE Int. Electr. Veh. Conf. IEVC, 2014, pp. 1–5. http://dx.doi.org/10.1109/IEVC.2014.7056123.

[79] P. Bellavista, A. Zanni, Feasibility of Fog computing deployment based on docker containerization over RaspberryPi, in: Proc. 18th Int. Conf. Distrib. Comput. Netw. ACM, New York, NY, USA, 2017, pp. 16:1–16:10. http://dx.doi.org/10.1145/3007748.3007777.

[80] S. Rusitschka, K. Eger, C. Gerdes, Smart grid data cloud: A model for utilizing cloud computing in the smart grid domain, in: 2010 First IEEE Int. Conf. Smart Grid Commun. 2010, pp. 483–488. http://dx.doi.org/10.1109/SMARTGRID.2010.5622089.

[81] A. Hakiri, P. Berthou, A. Gokhale, D.C. Schmidt, G. Thierry, Supporting SIP-based end-to-end data distribution service QoS in WANs, J. Syst. Softw. 95 (2014) 100–121, http://dx.doi.org/10.1016/j.jss.2014.03.078.

[82] K.C. Lee, R. Sudhaakar, J. Ning, L. Dai, S. Addepalli, J.P. Vasseur, M. Gerla, A comprehensive evaluation of RPL under mobility, Int. J. Veh. Technol. (2012) http://dx.doi.org/10.1155/2012/904308.

[83] E. Ancillotti, R. Bruno, M. Conti, The role of the RPL routing protocol for smart grid communications, IEEE Commun. Mag. 51 (2013) 75–83, http://dx.doi.org/10.1109/MCOM.2013.6400442.

[84] UPNP Forum, Leveraging UPNP+. The next generation of universal interoperability, 2015.

[85] J.E. Kim, G. Boulos, J. Yackovich, T. Barth, C. Beckel, D. Mosse, Seamless integration of heterogeneous devices and access control in smart homes, in: 2012 Eighth Int. Conf. Intell. Environ. 2012, pp. 206–213. http://dx.doi.org/10.1109/IE.2012.57.

[86] A. Botterud, J. Wang, C. Monteiro, V. Mir, Wind power forecasting and electricity market operations, 2009.

[87] J. j Wang, S. Mu, Security issues and countermeasures in cloud computing, in: Proc. 2011 IEEE Int. Conf. Grey Syst. Intell. Serv. 2011, pp. 843–846. http://dx.doi.org/10.1109/GSIS.2011.6043978.

[88] Meena Kumari, Rajender Nath, Security concerns and countermeasures in cloud computing paradigm, 00 (2015) 534–540, http://dx.doi.org/10.1109/ACCT.2015.80.

[89] G. Sivathanu, C.P. Wright, E. Zadok, Ensuring data integrity in storage: techniques and applications, in: Proc. 2005 ACM Workshop Storage Secur. Surviv, ACM, New York, NY, USA, 2005, pp. 26–36, http://dx.doi.org/10.1145/1103780.1103784.

[90] C. Huang, R. Lu, K.K.R. Choo, Vehicular fog computing: architecture, use case, and security and forensic challenges, IEEE Commun. Mag. 55 (2017) 105–111, http://dx.doi.org/10.1109/MCOM.2017.1700322.

[91] W. Wang, Z. Lu, Cyber security in the smart grid: survey and challenges, Comput. Netw. 57 (2013) 1344–1371, http://dx.doi.org/10.1016/j.comnet.2012.12.017.

[92] I. Stojmenovic, S. Wen, X. Huang, H. Luan, An overview of fog computing and its security issues, Concurr Comput. Pract. Exp. 28 (2016) 2991–3005, http://dx.doi.org/10.1002/cpe.3485.

[93] P.R. Sousa, L. Antunes, R. Martins, The present and future of privacy-preserving computation in fog computing, in: Fog Comput. Internet Things, Springer, Cham, 2018, pp. 51–69, http://dx.doi.org/10.1007/978-3-319-57639-8_4.

[94] P. Chen, S. Narayanan, J. Shen, Using secure MPC to play games, Mass. Inst. Technol. (2015).

[95] J. Pan, Y. Zhuang, S. Fong, The impact of data normalization on stock market prediction: Using SVM and technical indicators, in: Soft Comput. Data Sci, Springer, Singapore, 2016, pp. 72–88, http://dx.doi.org/10.1007/978-981-10-2777-2_7.

[96] M. Díaz, C. Martín, B. Rubio, State-of-the-art, and challenges, and open issues in the integration of Internet of things and cloud computing, J. Netw. Comput. Appl. 67 (2016) 99–117, http://dx.doi.org/10.1016/j.jnca.2016.01.010.

[97] Pigi - The Piqi Project, (n.d.). http://piqi.org/ (Accessed 25 October 2017).

[98] Google, Protocol Buffers, Google Dev. (n.d.). https://developers.google.com/protocol-buffers/ (Accessed 25 October 2017).

[99] C. Greer, D.A. Wollman, D.E. Prochaska, P.A. Boynton, J.A. Mazer, C.T. Nguyen, G.J. FitzPatrick, T.L. Nelson, G.H. Koepke, A.R.H. Jr, V.Y. Pillitteri, T.L. Brewer, N.T. Golmie, D.H. Su, A.C. Eustis, D.G. Holmberg, S.T. Bushby,

[100] D. Adams, J. White, M. Rumsey, C. Farrar, Structural health monitoring of wind turbines: method and application to a HAWT, Wind Energy 14 (2011) 603–623, 10.1002/we.437.

[101] Q.D. Ho, Y. Gao, T. Le-Ngoc, Challenges and research opportunities in wireless communication networks for smart grid, IEEE Wirel. Commun. 20 (2013) 89–95, http://dx.doi.org/10.1109/MWC.2013.6549287.

[102] L.-A. Tang, J. Han, G. Jiang, Mining sensor data in cyber-physical systems, Tsinghua Sci. Technol. 19 (2014) 225–234, http://dx.doi.org/10.1109/TST.2014.6838193.

[103] K. Khandeparkar, K. Ramamritham, R. Gupta, QoS-Driven data processing algorithms for smart electric grids, ACM Trans. Cyber-Phys. Syst. 1 (2017) 14:1–14:24, http://dx.doi.org/10.1145/3047410.

[104] Y. Xiao, Z. Ren, H. Zhang, C. Chen, C. Shi, A novel task allocation for maximizing reliability considering fault-tolerant in VANET real time systems, in: 2017 IEEE 28th Annu. Int. Symp. Pers. Indoor Mob. Radio Commun. PIMRC, 2017, pp. 1–7. http://dx.doi.org/10.1109/PIMRC.2017.8292511.

[105] S. Kabadayi, A. Pridgen, C. Julien, Virtual sensors: abstracting data from physical sensors, in: 2006 Int. Symp. World Wirel. Mob. Multimed. NetworksWoWMoM06, 2006, 6 pp. – 592. http://dx.doi.org/10.1109/WOWMOM.2006.115.

[106] M. Aazam, E.N. Huh, Dynamic resource provisioning through Fog micro datacenter, in: 2015 IEEE Int. Conf. Pervasive Comput. Commun. Workshop PerCom Workshop, 2015, pp. 105–110. http://dx.doi.org/10.1109/PERCOMW.2015.7134002.

[107] Liping Di, Geospatial Sensor Web and Self-adaptive Earth Predictive Systems (SEPS), in: Proc. Earth Sci. Technol. Off. ESTOAdvance Inf. Syst. Technol. AIST Sens. Web Princ. Investig. PI, San Diego, 2007, pp. 1–4.

[108] A.N. Srivastava, N.C. Oza, J. Stroeve, Virtual sensors: using data mining techniques to efficiently estimate remote sensing spectra, IEEE Trans. Geosci. Remote Sens. 43 (2005) 590–600, http://dx.doi.org/10.1109/TGRS.2004.842406.

[109] L. Duan, L.D. Xu, Business intelligence for enterprise systems: A survey, IEEE Trans. Ind. Inform. 8 (2012) 679–687, http://dx.doi.org/10.1109/TII.2012.2188804.

[110] H. Baars, H.G. Kemper, H. Lasi, M. Siegel, Combining RFID technology and business intelligence for supply chain optimization scenarios for retail logistics, in: Proc. 41st Annu. Hawaii Int. Conf. Syst. Sci. HICSS 2008, 2008, pp. 73–73. http://dx.doi.org/10.1109/HICSS.2008.93.

[111] H.J. Watson, B.H. Wixom, The current state of business intelligence, Computer 40 (2007) 96–99, http://dx.doi.org/10.1109/MC.2007.331.

[112] A. Azzara, L. Mottola, Virtual resources for the Internet of Things, in: 2015: 245–250. http://dx.doi.org/10.1109/WF-IoT.2015.7389060.

[113] D. Rotondo, F. Nejjari, V. Puig, A virtual actuator and sensor approach for fault tolerant control of LPV systems, J. Process Control. 24 (2014) 203–222, http://dx.doi.org/10.1016/j.jprocont.2013.12.016.

[114] S. Al-Sultan, M.M. Al-Doori, A.H. Al-Bayatti, H. Zedan, A comprehensive survey on vehicular ad hoc network, J. Netw. Comput. Appl. 37 (2014) 380–392, http://dx.doi.org/10.1016/j.jnca.2013.02.036.

[115] OpenStack open source cloud computing software, OpenStack. (n.d.). https://www.openstack.org/ (Accessed 26 October 2017).

[116] OpenNebula – Flexible Enterprise Cloud Made Simple, (n.d.). https://opennebula.org/ (Accessed 26 October 2017).

[117] Linux Containers, (n.d.). https://linuxcontainers.org/ (Accessed 26 October 2017).

[118] Docker, Docker. (n.d.). https://www.docker.com/ (Accessed 26 October 2017).

[119] M. Kozuch, M. Satyanarayanan, The internet suspend/resume (ISR), in: 4th IEEE Workshop Mob. Comput. Syst. Appl, IEEE CS Press, 2002, http://isr.cmu.edu/.

[120] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, Live migration of virtual machines, in: Proc. 2Nd Conf. Symp. Networked Syst. Des. Implement. - 2, USENIX Association, Berkeley, CA, USA, 2005: 273–286. http://dl.acm.org/citation.cfm?id=1251203.1251223 (Accessed 26 October 2017).

[121] M. Kozuch, M. Satyanarayanan, T. Bressoud, C. Helfrich, S. Sinnamohideen, Seamless mobile computing on fixed infrastructure, Computer 37 (2004) 65–72, http://dx.doi.org/10.1109/MC.2004.66.

[122] Kiryong. Ha, Mahadev Satyanarayanan, OpenStack++ for cloudlet deployment, 2015.

[123] Cisco, IOx, (n.d.). https://developer.cisco.com/site/iox/docs/ (Accessed 26 October 2017).

[124] F. Rodrigo Duro, J. Garcia Blas, D. Higuero, O. Perez, J. Carretero, CoSMiC: a hierarchical cloudlet-based storage architecture for mobile clouds, Simul. Model. Pract. Theory. 50 (2015) 3–19, http://dx.doi.org/10.1016/j.simpat.2014.07.007.

[125] B.I. Ismail, E.M. Goortani, M.B.A. Karim, W.M. Tat, S. Setapa, J.Y. Luke, O.H. Hoe, Evaluation of Docker as Edge computing platform, in: 2015 IEEE Conf. Open Syst. ICOS, 2015, pp. 130–135. https://dx.doi.org/10.1109/ICOS.2015.7377291.

[126] P. Bellavista, A. Corradi, M. Fanelli, L. Foschini, A survey of context data distribution for mobile ubiquitous systems, ACM Comput. Surv. 44 (2012) 24:1–24:45, http://dx.doi.org/10.1145/2333112.2333119.

[127] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, B. Ohlman, A survey of information-centric networking, IEEE Commun. Mag. 50 (2012) 26–36, http://dx.doi.org/10.1109/MCOM.2012.6231276.

[128] W. Raghupathi, V. Raghupathi, Big data analytics in healthcare: promise and potential, Health Inf. Sci. Syst. 2 (2014) http://dx.doi.org/10.1186/2047-2501-2-3.

[129] M. Allalouf, G. Gershinsky, L. Lewin-Eytan, J. Naor, Data-quality-aware volume reduction in smart grid networks, in: 2011 IEEE Int. Conf. Smart Grid Commun. SmartGridComm, 2011, pp. 120–125. https://dx.doi.org/10.1109/SmartGridComm.2011.6102302.

[130] M.T. Isaai, N.P. Cassaigne, Predictive and reactive approaches to the train-scheduling problem: a knowledge management perspective, IEEE Trans. Syst. Man Cybern. C 31 (2001) 476–484, http://dx.doi.org/10.1109/5326.983931.

[131] C.A. Georgopoulou, K.C. Giannakoglou, Metamodel-assisted evolutionary algorithms for the unit commitment problem with probabilistic outages, Appl. Energy 87 (2010) 1782–1792, http://dx.doi.org/10.1016/j.apenergy.2009.10.013.

[132] V.L. Erickson, Y. Lin, A. Kamthe, R. Brahme, A. Surana, A.E. Cerpa, M.D. Sohn, S. Narayanan, Energy efficient building environment control strategies using real-time occupancy measurements, in: Proc. First ACM Workshop Embed. Sens. Syst. Energy-Effic. Build, ACM, New York, NY, USA, 2009, pp. 19–24, http://dx.doi.org/10.1145/1810279.1810284.

[133] T.H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, L. Sun, Fog Computing: Focusing on Mobile Users at the Edge, ArXiv150201815 Cs. 2015, http://arxiv.org/abs/1502.01815 (Accessed 26 October 2017).