Fog Computing for the Internet of Things: A Survey

CARLO PULIAFITO, University of Florence, Italy and University of Pisa, Italy
ENZO MINGOZZI, University of Pisa, Italy
FRANCESCO LONGO and ANTONIO PULIAFITO, University of Messina, Italy
OMER RANA, Cardiff University, UK

Research in the Internet of Things (IoT) conceives a world where everyday objects are connected to the Internet and exchange, store, process, and collect data from the surrounding environment. IoT devices are becoming essential for supporting the delivery of data to enable electronic services, but they are not sufficient in most cases to host application services directly due to their intrinsic resource constraints. Fog Computing (FC) can be a suitable paradigm to overcome these limitations, as it can coexist and cooperate with centralized Cloud systems and extends the latter toward the network edge. In this way, it is possible to distribute resources and services of computing, storage, and networking along the Cloud-to-Things continuum. As such, FC brings all the benefits of Cloud Computing (CC) closer to end (user) devices. This article presents a survey on the employment of FC to support IoT devices and services. The principles and literature characterizing FC are described, highlighting six IoT application domains that may benefit from the use of this paradigm. The extension of Cloud systems towards the network edge also creates new challenges and can have an impact on existing approaches employed in Cloud-based deployments. Research directions being adopted by the community are highlighted, with an indication of which of these are likely to have the greatest impact. An overview of existing FC software and hardware platforms for the IoT is also provided, along with the standardisation efforts in this area initiated by the OpenFog Consortium (OFC).

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computer systems organization** → **n-tier architectures**; **Sensors and actuators**;

Additional Key Words and Phrases: Fog computing, internet of things, topological proximity, cloud computing

ACM Reference format:
Carlo Puliafito, Enzo Mingozzi, Francesco Longo, Antonio Puliafito, and Omer Rana. 2019. Fog Computing for the Internet of Things: A Survey. *ACM Trans. Internet Technol.* 19, 2, Article 18 (April 2019), 41 pages.
https://doi.org/10.1145/3301443

1 INTRODUCTION

The *Internet of Things* (IoT) [13] conceives a world in which every single object, from a "smart" one (e.g., a smartphone, a wearable device) to a non-communicating "dumb" thing (e.g., a lamp post, a

Authors' addresses: C. Puliafito, DINFO, University of Florence, Via di S. Marta, 3, 50139, Florence, Italy; email: carlo.puliafito@unifi.it; Department of Information Engineering, University of Pisa, Largo Lucio Lazzarino, 1, 56122, Pisa, Italy; email: carlo.puliafito@ing.unipi.it; E. Mingozzi, Department of Information Engineering, University of Pisa, Largo Lucio Lazzarino, 1, 56122, Pisa, Italy; email: enzo.mingozzi@unipi.it; F. Longo and A. Puliafito, Department of Engineering, University of Messina, Contrada di Dio, S.Agata, 98166, Messina, Italy; emails: {flongo, apuliafito}@unime.it; O. Rana, School of Computer Science and Informatics, Cardiff University, The Parade, 5, CF24 3AA, Cardiff, UK; email: ranaof@cardiff.ac.uk.

1533-5399/2019/04-ART18 $15.00
https://doi.org/10.1145/3301443

dumpster), can join the Internet. Such objects may not only exchange data but may also store and process data, use sensors to collect data from the surrounding environment, and actively intervene on the latter through actuators. Moreover, people are also a part of this ecosystem, consuming and producing data through their smartphones and wearable devices. The number of objects connected to the Internet surpassed the world human population in 2010 [6] and is expected to reach between 50 and 100 billion by 2020 [10]. Furthermore, the McKinsey Global Institute forecasts a potential economic impact for IoT applications of as much as $11.1 trillion per year in 2025 [106].

The IoT is necessary for the implementation of an unprecedented number of innovative services, but it is not sufficient in most cases to host such services directly. The great amount of heterogeneous data (i.e., the Big Data [31]) collected by IoT devices needs to be stored and processed, and the obtained insights need to be retrieved for visualization or actuation. However, all these tasks can rarely be performed on the IoT devices themselves, as such devices typically have limited compute, storage, and networking resources and can be battery-powered [49]. Therefore, the IoT needs support from more powerful resources—the most common being the use of Cloud Computing (CC) resources [20]. It is worth noting that Clouds may be public, private, or an hybrid combination of both [169]. The distinctive feature of public Clouds is that services and resources are made available by a third-party provider to anyone who requires them. Such resources are off-premises and rented according to a pay-per-use pricing model.[1] However, private Clouds are such that services and resources are accessible only by specific users (e.g., the members of an organization). Even though also private Clouds can be off-premises and managed by third-party providers under payment, they typically are on-premises, and their resources are released for free, as in that case users and providers coincide.

However, CC resources are concentrated in few Data Centres (DCs), which are considerably far away from the vast majority of data producers and consumers. This is especially true for public Clouds rather than private ones. Such non-negligible distance from end (user) devices leads to some drawbacks that are not acceptable for several emerging applications and services. Bonomi et al. proposed the *Fog Computing* (FC) [19] paradigm as a means to extend Cloud-based capabilities towards the network edge, distributing resources and services of computing, storage, and networking along the Cloud-to-Things continuum, in closer topological proximity[2] to IoT devices. Using FC, the key benefits of CC should be preserved, including resource virtualization, transparency, and elasticity [42]. Furthermore, as for the Cloud resources and services, also the Fog ones may be provided either for free or under payment. For instance, a municipality can exploit part of its own Fog resources for free and grant upon payment the rest to third-party developers.

In a report commissioned by the OpenFog Consortium (OFC),[3] 451 Research forecasts that the global Fog market opportunity has the potential to be worth $3.7 billion by 2019 and to reach $18.2 billion by 2022 [148]—with significant (and growing) academic and industry literature in this area. Table 1 summarises the most relevant surveys carried out in FC and organizes them by contributions, also highlighting the distinctiveness of the coverage in this article. We do not consider common contributions across these listed papers (e.g., description of FC principles, discussion of use cases for FC, review of the research challenges introduced by FC); we only highlight coverage that is unique in each case.

[1] See https://reasonstreet.co/business-model-pay-per-use/.

[2] Topological proximity means that the communication path between end devices and Fog resources is short. We believe that it is worth distinguishing this concept from that of geographical proximity, which is instead expressed in terms of physical distance. Indeed, while the topological proximity typically entails the geographical one, the opposite is not always true.

[3] See https://www.openfogconsortium.org/.

Table 1. The Main Survey Papers on FC Classified by Contributions

Contribution	Papers
Focus on the IoT	[4, 12, 126, 134, 197], *this article*
Discussion of existing software and hardware platforms	*this article*
In-depth analysis of the state-of-the-art architectures and algorithms	[102, 117]
Focus on resource management and offloading of user tasks	[102, 107]
Standardisation efforts from the OFC	[4], *this article*
Standardisation efforts from ETSI	[4, 102, 107, 164]
Security and privacy issues	[88, 118, 126, 150, 164, 167]
Focus on developers and engineers	[4, 102, 103, 126, 134, 164], *this article*
Historical context & background of FC	*this article*
Summary of recent work (i.e., from 2017 onward)	[4, 12, 88, 102, 107, 117, 118, 126, 164, 197], *this article*

The objective of this article is to provide a comprehensive survey on FC, with a specific focus on its integration with the IoT. However, although FC is tailored to the IoT, it is worth noting that its use is applicable in a number of other contexts, e.g., content delivery, gaming, network control functions. This article extends existing literature in FC in the following ways:

- it provides an overview of existing FC platforms for the IoT. Several software and hardware systems are already available, but to the best of our knowledge, none of the existing surveys discuss them. We believe that such a novel contribution may be of particular interest to engineers and developers. This is a changing landscape, and we provide a representative set of examples of systems;
- it highlights six IoT application domains that can benefit from FC and reports existing literature for these domains;
- it provides the historical background of FC, relating it to earlier proposals and demonstrating how FC is an evolution of these to address the needs of IoT applications. Existing survey papers mainly refer to these other paradigms as "similar concepts."

The rest of the article is organized as follows. For the sake of comprehensiveness, we first provide a general overview of FC. In Section 2, we discuss the limitations of integrating IoT and Cloud systems, which motivate the need for FC; in Section 3, we highlight the principles characterizing FC, whereas, in Section 4, we analyse FC from a historical perspective. Section 5 highlights six IoT application domains that can benefit from FC, identifying existing literature for each domain. In Section 6, we analyse challenges associated with extending Cloud-based systems towards the network edge, summarizing how the research community is addressing these challenges, and pointing out the main open issues and future research directions. Section 7 provides an overview of existing FC platforms for the IoT and outlines standardisation efforts being undertaken by the OFC. Finally, we provide conclusions in Section 8.

2 THE NEED FOR FOG COMPUTING

The integration between CC and the IoT allows resource-constrained IoT devices to offload data and complex computation onto the Cloud, taking advantage of its computational and storage capacity. However, the centralized nature of a Cloud DC can lead to a considerable topological

distance between CC resources/services and the vast majority of end (user) devices. This mostly depends on where the Cloud DC is located and/or on the area it covers. As such, private Clouds are more rarely affected, unless they cover considerably wide areas (e.g., the private Cloud managed by a municipality for Smart City services) and/or are off-premises. On the contrary, public Clouds are aimed at providing global coverage, and it is not rare to be served by public Clouds located in another country or even continent. In this section, we discuss the main shortcomings of the Cloud-IoT integration, which are all due to the great distance separating the Cloud from the IoT devices.

2.1 Latency

Some IoT application domains fall under the Ultra-Reliable Low-Latency Communications (URLLC) category, where extremely low and predictable response times are of utmost importance. According to Reference [160], road safety and autonomous driving services require latencies of less than 50ms, while Smart Grids of up to 20ms; Smart Factories have the most stringent requirements, with latencies varying from 250μs to 10ms. The distance between IoT devices and the Cloud often leads to a high communication latency that makes it difficult to satisfy application time constraints. For instance, the average round trip time between an Amazon Cloud server in Virginia (U.S.A.) and a device in the U.S. Pacific Coast is 66ms; it is equal to 125ms if the end device is in Italy; and reaches 302ms when the device is in Beijing [8].

2.2 Bandwidth Consumption

The number of "smart" objects producing and/or consuming data is projected to exponentially increase within the next few years. ABI Research estimates that data captured by IoT devices in 2014 surpassed 200 exabytes (i.e., 200 billion gigabytes) and is expected to exceed 1.6 zettabytes (i.e., 1,600 billion gigabytes) by 2020 [149]. For example: a smart factory might produce over a thousand terabytes (i.e., one million gigabytes) a day; self-driving cars may generate one gigabyte a second; and smart meters in the United States collect energy consumption data at 53.6 petabytes (i.e., 53.6 million gigabytes) a year [86].

2.3 Privacy and Security

The use of IoT devices leads to the inevitable collection of sensitive data (e.g., health-related data) that needs adequate protection. Transmitting these data over the public Internet to a Cloud DC can incur privacy risk [200]. Due to limited (user) control over identifying a data path from the IoT device to the Cloud DC, and as IoT devices do not have enough resources to encrypt/decrypt data, challenges of confidentiality, integrity and availability (referred to as the C-I-A triad) are important. Legal implications may be raised when sensitive data collected in one country are transmitted to a Cloud DC in another country where regulations are different—an aspect that has become more significant with the recent General Data Protection Regulation (GDPR) legislation in Europe and the California Data Privacy Law (in the US).

2.4 Context Awareness

Context is defined in Reference [2] as "any information that can be used to characterize the situation of an entity." Examples of context information may be: (i) the set of nearby nodes and/or services; and (ii) local network conditions and traffic statistics. Context awareness enables provision of improved services and resources utilization [135]. Due to a disaggregation between a Cloud DC and the sensor/actuator nodes (primarily due to geographical location and lack of proximity), limited context is shared between them. For instance, if a Cloud-hosted service detected a car

accident at an intersection, it would not be able to inform other vehicles in the vicinity of the accident, due to lack of local context.

2.5 Hostile Environments

Some IoT devices are employed in critical domains (e.g., traffic and emergency management) where environment and people's safety are key concerns. In such scenarios, the availability of services and data must be constantly guaranteed. However, there exist contexts referred to as hostile environments (e.g., rural areas or developing countries with a weak networking infrastructure, military settings, areas afflicted by natural or man-made disasters) in which the IoT experiences intermittent or no network connectivity towards the distant Cloud, and in which, as a result, the service gets interrupted, has very low performance, or is simply not available [157].

3 FOG COMPUTING PRINCIPLES AND STRENGTHS

FC was proposed in 2012 by Cisco [19] to overcome limitations of integration between Cloud DCs and the IoT. This section examines the principles and strengths of FC, focusing on the definition from the OFC [42]: "*Fog computing is a horizontal, system-level architecture that distributes computing, storage, control and networking functions closer to the users along a cloud-to-thing continuum.*"

3.1 Closer to the Users Along a Cloud-to-thing Continuum

As outlined in Section 2, drawbacks of Cloud-IoT integration are caused due to centralization of a Cloud DC. When talking about FC, it is worth noting that the expression "toward the network edge" does not mean "only at the network edge," as Fog services may be distributed anywhere along the continuum from Cloud to Things, hosted on nodes known as Fog Nodes (FNs) [108]. Any device that has enough computing, storage, and networking capabilities to run advanced services can be a FN [45]. Hence, FNs may be: (i) resource-rich end devices (e.g., vehicles, smart traffic lights, video surveillance cameras, industrial controllers); (ii) advanced edge nodes (e.g., switches, gateways, Wi-Fi access points, cellular base stations); and (iii) specialized "core" network routers.[4]

Table 2 identifies the advantages of FC over a simple Cloud-IoT integration, which are all consequences of the topological closeness of a Fog service to the associated IoT nodes. It is worth noting that these are all well-known strengths of FC and that the contents in Table 2 are taken from References [37, 154, 157, 162].

3.2 System-level Paradigm

The Fog is a system-level paradigm in the sense that it "*extends from the Things, over the network edges, through the Cloud, and across multiple protocol layers—not just radio systems, not just a specific protocol layer, not just at one part of an end-to-end system, but a system spanning between the Things and the Cloud*" [43]. Hence, FC fosters the development of systems where the overall service is generally not provided by a single resource-rich computer. Instead, the service is typically decomposed and provided by a hierarchy of FNs such that each of them runs a specific portion of the overall service, while cooperating with the other FNs. This pyramidlike organization is one of the guiding principles of the OpenFog Reference Architecture (OFRA) [42], as discussed in Section 7. However, as stated by the OFC in Reference [42], "*computational and system hierarchy is not required for all OpenFog architectures, but it is still expressed in most deployments.*"

[4]The core network, also known as backbone, connects different access networks with one another. Each access network comprises end devices and edge nodes, with the latter providing the former with an entry point to the core network.

Table 2. FC Advantages Over the Simple Cloud-IoT Integration

Cloud-IoT limitation	How the Fog can overcome it
Latency	FNs perform data analytics close to where data are collected and actions should be performed. This enables predictable response times, which are essential to many IoT applications.
Bandwidth consumption	Since a good portion of the data is communicated to nearby FNs, a reduced amount is exchanged with a Cloud DC. Moreover, FNs behave as a broker between the Things and the Cloud, further reducing data transmitted to a Cloud DC. Overall, FC helps to efficiently manage the volume of Big Data, by significantly reducing bandwidth consumption [165].
Privacy and security	Sensitive data can be locally stored and analysed by a FN, instead of being sent over the Internet up to the Cloud. However, the Cloud might need access to (part of the) sensitive data. In this case, such data may pass through the Fog for privacy enforcements that are not feasible for the resource-constrained IoT devices (e.g., extraction and transmission of metadata, complex encryptions). Therefore, the Fog can considerably improve privacy and security in modern applications and services.
Context awareness	FNs are located in closer proximity to IoT devices, improving context awareness. Exploiting context information enables improved services and/or optimizes resource utilization.
Hostile environments	FC proves to be fundamental when a service needs to be always available, but IoT devices experience intermittent or no connectivity to the Cloud. Instead, such a critical service may be provided by a nearby FN to which the IoT devices are able to connect.

As shown in Figure 1, the lowest layer in the hierarchy comprises the Things and the end devices in general, which might themselves behave as FNs if they are powerful enough. The higher layers lead from the network edge up to the core, and their number and composition depends on the actual application domain and purpose [42]. Finally, the topmost layer might be represented by the Cloud. Indeed—and this is of paramount importance—FC does not replace the Cloud but typically coexists and cooperates with it, as many services require the characteristics of both the Fog and the Cloud [19]. Interactions in such hierarchical systems may be of any type, both within the same layer and among nodes belonging to different layers [42]. Each node makes its own contribution to the overall service, and the nature of its role highly depends on its position in the pyramid. This is summarized in Table 3, which is the result of an integration of coverage across [19, 37, 42].

This hierarchical organization, together with proximity to end devices, is the main characteristic of FC, which makes it particularly suitable for the IoT. The IoT domain is often identified by wide-area deployment of sensors and actuators that can cover areas of hundreds or more square miles. Moreover, IoT applications and services are always more complex, as they may involve aspects such as: time-critical control, visualization and reporting, and historical analysis of Big Data. Spanning from the Things up to the Cloud, the Fog hierarchy enables all this. Examples of FC hierarchies applied to transportation systems and to the food processing plant can be found in Reference [42], while Reference [19] reports an example related to Smart Grids.

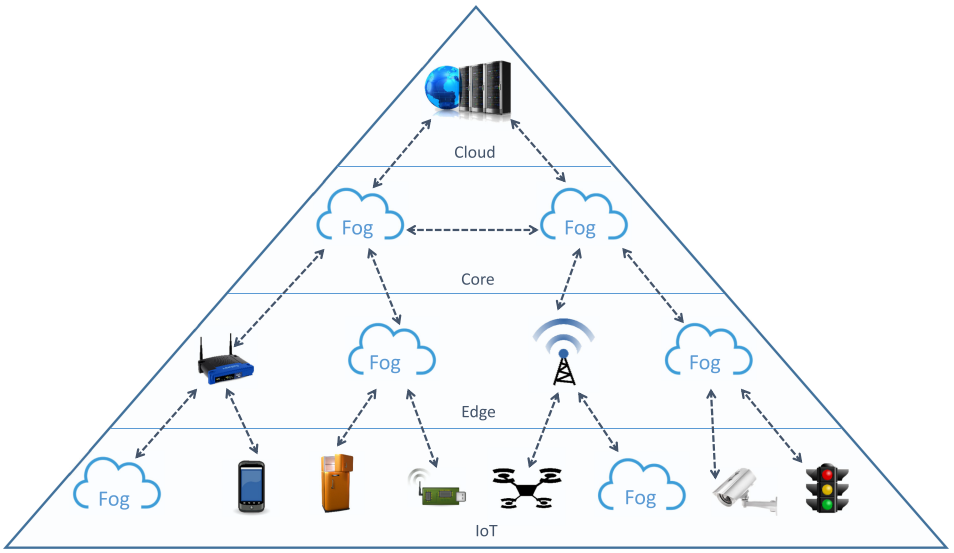

Fig. 1. FC hierarchical organization.

3.3 Horizontal Paradigm

FC can also be viewed "horizontal" in the sense that it is generic enough to be applied in a number of different application scenarios, e.g., content delivery, gaming, network control functions [61, 93, 201]. However, this survey only focuses on the contribution of FC to the IoT.

4 HISTORICAL BACKGROUND

FC is an evolution of early proposals with the objective to best answer the needs of the IoT. This section explores the Fog and the so-called "similar concepts" from an historical perspective, with the purpose to clarify the reasons that led to the characteristics of each of these concepts and focus more on their similarities rather than their differences.

It all began in the early 2000s with a big contradiction in one of the most emerging trends of that period: Mobile Computing. On the one hand, mobile devices have the potential to make emerging services in several fields (e.g., healthcare, gaming, entertainment, social networking) always available; though, on the other hand, they usually have limited computing capabilities, as they have to be often light and small and require a long battery life [51]. Therefore, it is difficult for them to provide resource-intensive services by just relying on their own facilities.

Hence, how is it possible to release the full potential of Mobile Computing despite its limitation? In 2001, Mahadev Satyanarayanan (professor of Computer Science at the Carnegie Mellon University) proposed the concept of *Cyber Foraging* as a possible solution to the problem [153]. This paradigm suggested to offload data and intensive computation from a mobile device onto a more powerful server belonging to the fixed infrastructure. Such a server was supposed to be in close proximity to the associated mobile node, but this assumption was not made explicit by Prof. Satyanarayanan at that time. Although Cyber Foraging is the real ancestor of FC, other paradigms bringing content or computation closer to the end devices, such as Content Delivery Networks (CDNs) [132] and in-network processing [33], were being proposed in those years.

Among the many open issues raised by Cyber Foraging, one was particularly tricky: who and why should have made those servers available? The answer to this question was found few years later with the introduction of CC, whose characteristics have been already discussed in the

Table 3. Nodes have Different Properties and Roles According to Their Position in the Hierarchy

	FNs closest to the IoT	FNs at the core network	Cloud
Fog benefits (see Table 2)	The FC advantages are evident.	They become less evident.	They are null.
Geographical coverage	These FNs are widely distributed to ensure close proximity to the Things. Hence, each of them covers a small area, controlling few IoT devices.	The farther from the true edge, the fewer the FNs. Therefore, each of them covers a rather wide geographical area.	CC resources are highly concentrated in few DCs all over the world. Thus, the Cloud features a global coverage, as each DC has to manage a huge area.
Data persistence	Time-sensitive data are sent to these FNs for instant (i.e., O(ms)) decision-making and actuation. Hence, such data are transient.	Data that can wait (seconds to minutes) from the time of sensing to that of actuation are sent to these FNs.	Data persist in the Cloud for days, weeks, or even months for historical analysis.
Computing power	These FNs are typically the least powerful, as they have to process transient data from a limited area.	The higher the level in the pyramid, the more powerful the nodes. There is therefore a need to process more persisting data from a wider geographical area.	The Cloud is the most powerful. Furthermore, the insights realizable in the Cloud are the greatest due to the size of datasets available.
Contribution	These FNs collect the data, process them, and issue actuation commands. They may also filter the data to be kept locally and transmit the rest to the higher layers. Thus, the only type of interaction at this level is Machine to Machine (M2M).	These nodes typically perform data filtering, compression, and transformation. They may also issue less time-sensitive commands to the actuators. Finally, they can provide visualization and reporting services to end users. Hence, this level features both M2M and Human to Machine (HMI) interactions.	The Cloud collects data from hundreds or thousands of nodes. It performs long-term storage, historical analysis and forecasting, and Big Data analytics. The Cloud typically interacts with the final users for insights delivery, although also IoT nodes might directly communicate with it.

Introduction of this article. The integration between Mobile Computing and CC is referred to as *Mobile Cloud Computing* (MCC) [65].

Although MCC was a promising paradigm, it presented all the limitations discussed in Section 2. Therefore, in 2009 Satyanarayanan et al. [155] suggested to cope with such shortcomings (and in particular with the high and unpredictable latencies) through the concept of *Cloudlet*, which was the de facto birth of a paradigm known as *Mobile Edge Computing* (MEC) [102]. A Cloudlet is defined as "*a trusted, resource-rich computer or cluster of computers that is well-connected to the Internet and available for use by nearby mobile devices.*" A resource-constrained mobile device can behave as a thin client and, rather than relying on the distant Cloud, can offload all the significant

computation onto a nearby Cloudlet located at the network edge. This still provides all the benefits of CC, such as virtualization and efficiency, though without the characteristic delays. If no Cloudlet is present nearby, then the mobile device can temporarily rely on the Cloud as a fallback option or, in the worst case, on its own resources [155]. More in general, the use of Cloudlets to support any type (i.e., either mobile or fixed) of resource-limited end devices or groups of devices is simply referred to as *Edge Computing* (EC) [78].

Since MEC emerged as a worthy solution to enable computation-intensive mobile applications, the European Telecommunications Standards Institute (ETSI) created an Industry Specification Group (ISG) in 2014 with the purpose to define and integrate a standard implementation of MEC into cellular networks, which was called ETSI Mobile Edge Computing (ETSI MEC) [80]. According to the ETSI, such a standard lets operators "*open their Radio Access Network (RAN) edge to authorized third-parties, allowing them to flexibly and rapidly deploy innovative applications and services towards mobile subscribers, enterprises and vertical segments*" [60]. More recently, the ETSI renamed ETSI MEC in *Multi-Access Edge Computing* to emphasize the novel intention to also address non-cellular operators' requirements [59].

Finally, to clarify the last step toward FC, it is fundamental to highlight the following aspect. At least in its infancy, MEC did not consider the overall service to be decomposed and provided by a hierarchy of nodes including also the Cloud; instead, the whole service is entirely provided by a nearby Cloudlet (if available), as we have already mentioned. This is why the OFC states that "*fog works with the cloud, whereas edge is defined by the exclusion of cloud. Fog is hierarchical, where edge tends to be limited to a small number of layers*" [42]. This characteristic of MEC is reasonable in the context of Mobile Computing, where an application typically involves a single user. However, the IoT is often defined by sensors and actuators covering wide areas and by the need for long-term storage and Big Data analytics (i.e., all elements that may require the Cloud). At the same time, proximity is necessary to enable low and predictable response times together with all the other benefits reported in Table 2 (which require resources towards the network edge). As a result, to best suit such requirements, Cisco advanced the FC paradigm in 2012 [19] as a generalization of EC in which it may still happen that a single, closer resource-rich computer provides the overall service, but most of the times any resource in the Cloud-to-Things continuum provides only a portion of the overall service, according to the facilities and position in the pyramid (see Section 3).

Figure 2 illustrates and compares the original definitions of MCC, MEC, and FC. The research community often tends to look for the differences between FC and EC. However, it might be more fruitful to emphasize the several similarities between these two paradigms. Indeed, on the one hand, they were born in different moments and were specifically conceived for different contexts, but, on the other hand, they are evolving over time towards an inevitable convergence [154, 156, 158]. As a proof of this, the ETSI and the OFC recently signed a Memorandum Of Understanding (MOU) with the intent to join forces for the development of Fog-enabled Mobile Edge applications and technologies [58].

5 IOT APPLICATION DOMAINS

As detailed in Section 3, FC proves to be a promising paradigm to support the IoT. Taking inspiration from the classification found in Reference [109], this section organizes the IoT applications into six domains. Overall, we found 45 works proposing an integration between FC and the IoT in one of those categories.[5] We merely report these six domains from the most to the least

[5]We consulted the main scientific literature databases and search engines (i.e., IEEE Xplore, ACM library, ScienceDirect, and Google Scholar) from August 2017 to January 2018. Search queries were formulated to be as comprehensive as possible within each considered application domain. For example, the following is the search query defined for the ITS domain:

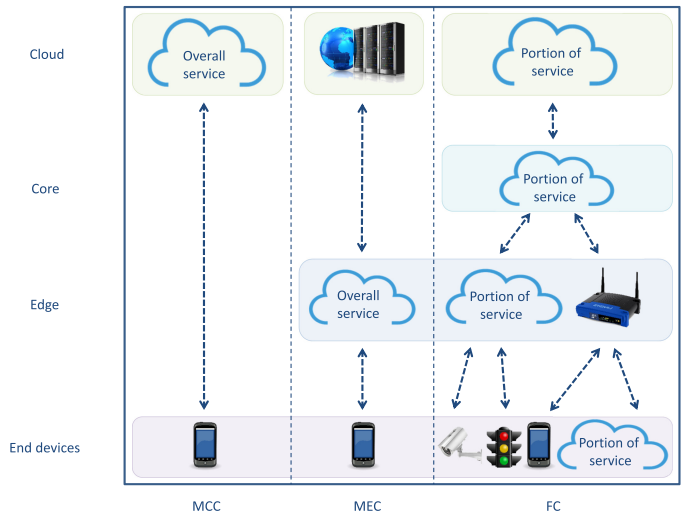

Fig. 2. A comparison among the definitions of MCC, MEC, and FC.

investigated in terms of number of published papers. More specifically, the most examined is the Intelligent Transportation Systems (ITS) domain (12 papers, 26.7%), followed by Smart Healthcare (11 papers, 24.4%). Next, there are the public safety sector (7 papers, 15.5%), Smart Grids (6 papers, 13.3%), and Industry 4.0 (5 papers, 11.1%). Finally, Smart Homes and Buildings conclude the list (4 papers, 8.9%). The objective is to highlight how each of these domains may benefit from FC and provide a comprehensive overview of the state of the art in the employment of the Fog within each of them. Table 4 summarizes the main aspects of each considered work by: (i) outlining the major contribution with keywords; (ii) reporting which devices are employed as FNs; and (iii) pointing out the maturity level of the proposal. The maturity level may be one of the following: *Theory*; *Simulation*; *Prototype*; *Pre-product* (i.e., already available for use but still under active development); and *Product*.

5.1 Intelligent Transportation Systems

The world urban population is dramatically increasing. At present, the number of megacities (i.e., cities with a population exceeding 10 million people) is 28 and is projected to reach 41 by 2030 [183]. As a consequence of this, urban environments are more and more overcrowded with vehicles, and traffic congestions, time losses, accidents, and pollution altogether contribute to a non-negligible reduction in the experienced safety and Quality of Life (QoL). The employment of Information and Communication Technologies (ICT) within the transport domain gives birth to the ITS, where a wide range of services and applications may be conceived to face the aforementioned issues [52]. Hence, ITS allow to considerably improve traffic efficiency, drivers' and passengers' safety, and freight transport.

(Fog Computing OR Edge Computing) AND (ITS OR vehicle OR RSU OR traffic OR road OR transport OR driver OR parking). Through this methodology, we found contributions whose publication years are not earlier than 2014. Finally, with the aim to consider only the most relevant and recent works, we filtered the obtained results as follows: (i) given two similar works from the same group of authors, of which one is a conference paper and the other is a journal article, we selected the latter; (ii) in case these two works are both conference papers or journal articles, we selected the most recent one.

Table 4. Papers Employing the Fog in One of the Considered IoT Application Domains

Domain	Paper	Keywords	FNs	Maturity
ITS	[175]	Look-up service; DHT	n/a	Prototype
	[89]	Parking; Matching theory	n/a	Simulation
	[95]	Architecture for urban traffic management; SDN; 5G	Cellular base stations	Simulation
	[96]	Architecture for urban traffic management; SDN; 5G; IEEE 802.11p	Cellular base stations	Simulation
	[181]	Architecture; SDN; Data streaming; Lane change	Cellular base stations; RSUs; Road Side Unit Controllers (RSUCs)	Theory
	[76]	Architecture for load balancing; SDN	Cellular base stations; RSUs	Simulation
	[69]	Architecture; SDN; 5G	Cellular base stations; RSUs; vehicles; RSUCs	Simulation
	[163]	Architecture for urban traffic management; Pub-Sub; Semantic Web	n/a	Theory
	[23]	Urban traffic management	RSUs	Simulation
	[26]	Stack4Things; Complex Event Processing	Single-board computers	Prototype
	[79]	Vehicular Fog Computing	Vehicles; cellular base stations; RSUs	Simulation
	[195]	Service offloading in bus networks; Genetic algorithm	Vehicles (i.e., buses); RSUs	Simulation
Smart Healthcare	[152]	Zika virus; fuzzy k-nearest neighbor	n/a	Prototype
	[112]	UV radiation measurement; Android	n/a	Prototype
	[29]	Fall detection; Android	Smartphones	Prototype
	[68]	COPD patients; Mild dementia	n/a	Prototype
	[110]	COPD patients; Dynamic adjustment of the oxygen dose	Portable oxygen concentrators; gateways	Prototype
	[198]	Brain monitoring; Semantic Web	Personal Computers; home gateways	Prototype
	[7]	Heart attack; vehicular networks; SDN	Cellular base stations; RSUs; RSUCs	Prototype
	[114]	Parkinson's disease; speech treatments	Embedded systems	Prototype
	[3]	Security and privacy of health-related data; CASB	n/a	Prototype
	[57]	Security and privacy of health-related data	Personal gateways	Prototype
	[145]	Smart e-Health Gateway	Gateways in a Smart Home or hospital	Prototype
Public safety	[151]	Critical events in a Smart City	Cellular base stations	Theory
	[147]	Disaster management; crowdsourcing	n/a	Theory
	[111]	Architecture for social sensing services in hostile environments	n/a	Theory
	[28]	Smart levee monitoring system	Industrial controllers; single-board computers	Prototype
	[116]	Crowd surveillance; UAVs	Cellular base stations	Prototype
	[47]	Intelligent surveillance system	Smart cameras	Prototype
	[32]	Smart urban surveillance; target tracking	Tablets; smartphones; laptops	Prototype

(Continued)

Table 4. Continued

Domain	Paper	Keywords	FNs	Maturity
Smart Grid	[193]	Smart metering infrastructure; Big Data	Smart meters	Prototype
	[121]	Data aggregation for bandwidth efficiency; Power Line Communication	Routers	Simulation
	[16]	Data aggregation for preserving privacy of energy consumption	n/a	Theory
	[75]	Algorithm to detect NTL fraud	n/a	Simulation
	[192]	Power consumption schedule; Demand Side Management	n/a	Simulation
	[176]	V2G; EVs; 5G	EVs; local aggregators; control centres	Simulation
Industry 4.0	[48]	Docker-based service orchestration; oneM2M; P2P communications	n/a	Simulation
	[170]	Energy-efficient FNs for industrial WSNs	Servers in a Wireless Computing System	Simulation
	[133]	Reduction of sensor energy consumption; MQTT	IoT gateways operating as MQTT brokers	Simulation
	[191]	Machine health and process monitoring	Gateways in factory floors	Prototype
	[122]	FC platform tailored to the industrial automation sector	Modular computers	Product
Smart Home and Smart Building	[185]	Awareness of the home context; Device-to-Device	Home gateways; set-top boxes; end user devices	Simulation
	[55]	A single FN for the whole building	Wi-Fi routers	Prototype
	[161]	FNs in multiple rooms of a building	n/a	Theory
	[97]	FC platform tailored to the Smart Home and Smart Building domain	Wi-Fi access points; set-top boxes	Pre-product

FC can play a crucial role in this context [85]. Indeed, as we have already mentioned in Section 2, road safety and autonomous driving services require response times to be lower than 50ms [160], which usually is not achievable with CC. Furthermore, as it is described in Reference [42], FC: (i) saves bandwidth, by avoiding that all the data collected by vehicles and by the fixed infrastructure are sent up to the Cloud; (ii) provides critical ITS services also in the presence of intermittent network connectivity towards the Cloud; and (iii) allows FNs to provide context-aware services to the vehicles in their proximity (e.g., alerting them of bad road conditions in that area). In Reference [175], the authors propose a look-up service for ITS based on a Distributed Hash Table (DHT) to be implemented by FNs. A Fog system to help drivers to find a free parking slot is presented in Reference [89]. Such a system features a pyramid-like organization so that the more toward the network edge a FN, the smaller its coverage area, but the higher its context awareness.

Several works [69, 76, 95, 96, 163, 181] propose distinct Fog-based architectures for ITS. Except for Reference [163], they all employ FC together with Software Defined Networking (SDN), which provides network flexibility and programmability. The resulting architectures are thus organized into four layers: (i) CC; (ii) SDN control; (iii) FC; and (iv) Infrastructure layer, which comprises the sensing and actuation nodes. Moreover, the architectures in References [95, 96, 163] are either validated or specifically envisioned for urban traffic management and control, which is the ITS major concern. Traffic management is the cornerstone also in Reference [23] where the authors propose FOX, a Fog-based system whose objective is to detect and minimize traffic congestions.

Finally, in Reference [26], the authors propose Stack4Things as a FC platform for Smart City applications. They exploit Cloud-based network virtualization functionalities to implement a smart mobility use case in which smart cars can interact with Smart City objects to implement geolocalised services. For example, smart cars approaching intersections are able to communicate with smart traffic lights to acquire a certain level of priority with respect to other cars.

In Reference [42], traffic control is one of the reported use cases for FC. This article, unlike the others that have just been introduced, points out an interesting aspect: the vision of vehicles as FNs and not only as sensing and actuation devices. This is further discussed in Reference [79] where the authors present Vehicular Fog Computing (VFC) to exploit and aggregate the great amount of underutilized resources in nearby vehicles, together with those belonging to the fixed infrastructure, such as cellular base stations and Road Side Units (RSUs), to provide services of computation, storage, and networking. As a result, parked and slow-moving vehicles form a FC layer to enable several vehicular services and applications. To conclude, Reference [195] might be considered as a particular case of Reference [79], as the authors propose to extend the computing capability of the fixed infrastructure at the network edge by utilizing buses and bus networks. The main reason for this is that the fixed trajectories and strong periodicity of buses are ideal in this direction.

5.2 Smart Healthcare

The healthcare domain is one of the toughest and most delicate as it deals with people's lives. The Internet of Healthcare Things (IoHT), together with CC, allows to envision several services for the improvement of patients' QoL. However, a simple sensor-to-Cloud architecture proves to be often too reductive and unsuitable for many emerging healthcare applications with critical requirements. FC can be the solution to the problem [63, 90], especially but not only in the following three ways: (i) it enables low and predictable response times, which can often make the difference between life and death for patients; (ii) it ensures that at least the most critical portion of the overall service is always available to the patient, also in the presence of hostile environments with intermittent or no network connectivity to the Cloud; and (iii) it protects the health-related sensitive data by keeping them locally (e.g., in a FN located within the hospital or the patient's house) rather than sending them to the Cloud through the Internet.

Sareen et al. [152] propose a Fog-based system for predicting and preventing the Zika virus outbreak. The Fog layer performs real-time processing of environmental sensor data as well as symptoms data collected by the users' smartphones. In Reference [112], the authors conceive a service that takes advantage of the FC context awareness due to the proximity to users' smartphones to provide accurate and localized measurements of Ultraviolet (UV) radiations. Falls are among the major causes of mortality for stroke patients. Therefore, it is of vital importance to promptly detect falls and intervene. U-Fall [29] is a FC system to achieve this objective: The patient's smartphone behaves as the FN for a quick fall detection; sensor data are also transmitted to the distant Cloud for long-term storage and analysis. Some works propose to adopt FC to improve the QoL of Chronic Obstructive Pulmonary Disease (COPD) patients. In this context, Fratu et al. [68] extend the eWALL Cloud-IoT system[6] by implementing the Fog layer. A further contribution in this direction is made in Reference [110], where the authors propose to assist COPD patients also when these are performing physical exercises. To this aim, the oxygen dose is dynamically adjusted also to the patient's context and needs; hence, FC context awareness is required. The Fog can be similarly applied to enable services that monitor brain activity in, for example, stressed or Parkinson's disease patients [198]. In Reference [7], the authors propose a service exploiting

[6]See http://ewallproject.eu/.

resources at the network edge and SDN for the real-time detection of heart attacks in drivers. FIT is a FN conceived in Reference [114] that preprocesses the speech data of a patient with speech impairments and forwards speech features to the Cloud to reduce the required bandwidth and computational burden on the Cloud.

Patients' health-related sensitive data need to be preserved and protected: FNs may behave as privacy and security enforcement points. In this direction, a Cloud Access Security Broker (CASB) may be executed at the Fog layer as in Reference [3]. Similarly, the authors in Reference [57] develop an Enhanced Middleware for Collaborative Privacy (EMCP) to be hosted on FNs. More in general, Rahmani et al. [145] present UT-GATE, the prototype of a FN that provides the healthcare domain with all the benefits typical of FC.

5.3 Public Safety

FC and the IoT are relevant paradigms also from the viewpoint of public safety and well-being. For example, in Reference [151] the authors present a Fog-IoT architecture with this purpose. To guarantee public safety, two tasks have to be effectively performed: disaster management and crowd surveillance.

Natural or man-made disasters usually cause significant human, economic, and environmental damages. According to Reference [82], more than 6,000 disasters happened in the past 10 years, causing almost 772,000 people killed, 1,917,557 somehow affected, and a total estimated damage equal to $1,424,814 million. Therefore, properly managing these situations is of vital importance. In this direction, Rauniyar et al. [147] propose a Fog-based architecture where crowdsourced data are communicated to the Fog for quick processing and decision-making. FNs store emergency contact numbers and are accessible by the local public safety authorities that can plan rescue actions according to the produced insights. At the same time, affected people may contact the nearby FN to efficiently obtain crowdsourced pictures and videos, thus to have an idea of the current situation. Both natural and man-made disasters may cause Internet connectivity to be unstable. Despite this, having uninterrupted access at least to the most critical part of the service is a must in such delicate situations. As we reported in Section 3, FC provides this important feature [111]. Brzoza-Woch et al. [28] present a levee monitoring use case involving the Fog. They conceive a three-layered architecture where edge nodes may: (i) locally make decisions; (ii) collaborate with one another; and (iii) optionally return preprocessed (i.e., filtered and/or compressed) results to the Cloud for further analysis and forecasting. Different versions of this system exist to best suit diverse environmental, infrastructure, and economic conditions.

Crowd surveillance is essential to guarantee public safety. Indeed, it allows, for example, to: (i) identify non-authorized accesses and suspicious activities; (ii) detect the fall of an elderly or infirm person; (iii) pinpoint a terrorist or criminal; and (iv) find a missing person. The suitability of FC to crowd surveillance is evident, as the Fog grants low and predictable response times, bandwidth efficiency, and privacy preservation [42]. Taking this into consideration, the authors in Reference [116] propose an Unmanned Aerial Vehicle (UAV)-based IoT platform and present a use case where drones transmit surveillance videos to edge nodes that locally perform face recognition tasks. Similarly, in Reference [47], the authors present a case study based on a distributed intelligent surveillance system scenario in a crowded area, implemented on clustered Fog devices that are able to horizontally offload tasks among themselves. To conclude, Reference [32] discusses an urban speeding traffic monitoring system using FC. A drone monitors moving vehicles by recording a surveillance video that is sent back to the drone controller on the ground and displayed on a screen. If the police officer finds a vehicle moving at a suspicious speed, then the system forwards the next video frames to a FN to track that vehicle.

5.4 Smart Grid

The traditional electrical grids distribute energy from few central power generators to a very large number of final customers. The Smart Grid [62] is an evolution of the traditional power grid, as it is the result of the integration between the latter and the ICT. In a Smart Grid, energy is generated by several widely distributed stations, and smart meters and other sensor nodes are employed to monitor and control the energy consumption. As a result, there is a continuous, bi-directional flow of both electricity and data that allows to conceive services for a more efficient, reliable, and secure energy management. Such services may greatly benefit both: (i) the electricity suppliers, e.g., to efficiently deliver and manage energy; (ii) the final customers, e.g., to easily monitor and/or reduce their energy consumption.

As it has been just mentioned, Smart Grids are characterized by a strong distribution of power generators, energy transformers, sensors, and actuators: it is not uncommon for a Smart Grid to cover an area of hundreds square miles. Moreover, Smart Grid sensors produce a vast amount of data, which can easily saturate network, storage, and processing resources. To further complicate matters, smart meters data may be exploited to deduce personal information (e.g., the number of people in a specific area, the habits of a family); therefore, privacy in Smart Grids is an important issue [127]. Last but not least, many Smart Grid services require quick and predictable response times, typically between three and 20ms [160]. All these features make Smart Grids an ideal domain where to apply FC.

Several works employ FC in Smart Grids. The authors of Reference [193] propose a Fog-based Smart Grid solution where smart meters are grouped to form computing and storage clusters, thus realizing a Fog layer at the extreme network edge. A hierarchy of FNs in the Smart Grid context may perform data aggregation (i.e., data are gathered and expressed in a summary form) to reduce the amount of data transmitted to the Cloud and thus save bandwidth [121]. Data aggregation carried out by FNs can also preserve the privacy of customers' energy-related data [16]. Han et al. [75] propose a security analytic algorithm to be executed by cooperating FNs for the detection of Non-Technical Loss (NTL) fraud in Smart Grids. An attacker performs NTL fraud by tampering with a smart meter so that it reports fake energy consumption values. The proposed iterative algorithm divides the overall problem in sub-problems and assigns each of them to a FN; the solution to the overall problem is given by the local solutions of the sub-problems.

Some works are more application-oriented. The authors of Reference [192] present a Fog-based approach for the optimization of the power consumption schedule in Smart Grids, which results in an optimization of both customers' and electricity supplier's costs. In more detail, the Smart Grid is organized in regions, and each region is managed by an edge node that finds the optimal power consumption schedule for its region, based on the collected data. The centralized Cloud is then responsible for the optimization of the energy consumption schedule at a multiregional level. To conclude, Vehicle to Grid (V2G) is an emerging set of services that allows Electric Vehicles (EVs) to both consume and return back electricity from/to the Smart Grid. Foud [176] is a computing model integrating the Cloud, the Fog, and 5G technologies to improve V2G services. In Foud, EVs may be both final users and components of the Fog layer, which is said to be temporary due to the vehicles mobility.

5.5 Industry 4.0

Since its very beginning in the late 18th Century, industrial production has experienced several revolutions that have deeply changed its nature. First, mechanization driven by steam power made its entrance. The second industrial revolution consisted in electrification and mass production, while the third era of industry started in the 1960s with the digital programming of automation

systems. Nowadays, we are undergoing the fourth industrial revolution, which is either known as Industry 4.0, Smart Factory, or Smart Manufacturing. All these terms identify the same revolutionary trend: the employment of the IoT and, more generally, Cyber-Physical Systems (CPSs) in industrial automation for a smarter production [53].

Thanks to its advantages, FC may be the solution to several challenges raised in this context [22]. In particular, the Fog is very useful within a Smart Factory to satisfy the latency requirements that characterize such a context. Typically, these requirements are the most stringent among all the investigated domains, as they vary from from 250μs to 10ms. An exclusive reliance on the Cloud would not allow to respect such stringent latency requirements.

De Brito et al. [48] propose a solution based on the oneM2M technical specifications[7] that enhances peer-to-peer (P2P) communications between FNs and implements a Docker-based service orchestration mechanism in the industrial domain. The authors in Reference [170] present a system for industrial Wireless Sensor Networks (WSNs) that minimizes the power consumption, by controlling the FNs sleep scheduling and network connectivity, while satisfying the time constraints imposed by Smart Manufacturing applications. A Fog architecture is described in Reference [133] where FNs are IoT gateways operating as Message Queue Telemetry Transport (MQTT)[8] brokers able to predict future sensor measurements. As a result, sensors need to publish their data only in case of wrong predictions by the broker; this helps to reduce their power consumption while keeping latencies low. The authors in Reference [191] discuss a Fog-based architecture for machine health and process monitoring in cyber-manufacturing systems. To conclude, Nebbiolo Technologies [122] launched a FC platform for the industrial automation sector; we will discuss it in Section 7.

5.6 Smart Home and Smart Building

FC is progressively entering the home context; in-home devices (e.g., home gateways, set-top boxes, end user devices) may behave as FNs, as they are becoming increasingly powerful, and virtualization techniques are more and more efficient [185]. Smart Homes will enormously benefit from this trend. Indeed, response times would be further reduced, which is essential for time-sensitive Smart Home systems such as those who deal with surveillance and access control. Moreover, the presence of a FN in the house would ensure resilience when there is no Internet connectivity to the Cloud. Last but not least, privacy and bandwidth efficiency would be both improved, as the many (sensitive) data collected would be mainly kept within the house.

Similarly, FNs may be also present inside buildings to enable improved Smart Building services. Depending on the actual needs, there could be a single FN for the whole building, or there could be an internal hierarchy with a FN for each floor or even one for each room [42]. Dutta et al. [55] propose a Smart Building system with a single FN for the whole building. The FRODO architecture proposed in Reference [161] is more sophisticated, as FNs may be deployed in multiple rooms of a building for decentralized decision-making. Each of them provides highly context-aware services to the occupants of its room, taking into account their personal preferences together with objective, room-related parameters (e.g., the type of sensors and actuators present). Last but not least, Liu et al. [97] present ParaDrop, a FC platform that allows to manage and deploy services on wireless gateways (e.g., Wi-Fi access points, set-top boxes). This platform, which particularly suits the Smart Home and Smart Building domain, will be further detailed in Section 7.

[7] See http://www.onem2m.org/.
[8] MQTT is a Publish-Subscribe lightweight messaging protocol. See http://mqtt.org/.

Table 5. Characteristics Needed to Be Considered when Extending the Cloud Toward the Network Edge

Characteristic	Description	Introduced or influenced challenges
Geographical distribution	FC leads from a situation in which resources and services are all concentrated in a Cloud DC to one in which they are distributed over a potentially wide area.	Mobility support; orchestration; deployment models; security and privacy
Higher heterogeneity	While Cloud servers are all very alike, FNs are usually heterogeneous, as they might feature different hardware specifications and capabilities, operating systems, or protocol suites [186].	Orchestration; deployment models; security and privacy
Computing power	As reported in Table 3, FNs are in general less powerful than Cloud servers. However, there exists a wide range of diverse FNs with very different hardware capabilities, as outlined in Table 9.	Mobility support; orchestration; security and privacy
Network performance	While a Cloud DC relies on a high-bandwidth and low-latency LAN, FNs are typically interconnected with each other through a WAN and hence experience higher latencies with respect to those within a Cloud DC and an average bandwidth of 13Mbps [73].[9]	Mobility support; orchestration
Vulnerable environment	With the aim to be closer to IoT devices, FNs are usually located in environments that are more vulnerable and less protected than Cloud DCs [35].	Deployment models; security and privacy

6 RESEARCH CHALLENGES

New system, network, and environmental characteristics need to be considered when extending the Cloud toward the network edge—see Table 5. This section specifically focuses on challenges associated with these characteristics, identifying how the research community is addressing them.

6.1 Mobility Support

The Internet of Mobile Things (IoMT) [119] is an ever-growing phenomenon—according to Reference [38], wearable devices are expected to reach 930 million by 2021. These resource-constrained mobile IoT devices require topologically close resources and services (e.g., located at the network edge) for enabling value-added applications that can benefit from FNs.

The objective is to utilise FNs when IoT devices move from one place to another. Device mobility limits FC benefits, as when a device moves, the topological distance between it and the associated FN increases. It is worth highlighting that this issue does not exist in Cloud-only environments,

[9]This does not change the fact that the topological distance between one or more IoT devices and a FN is much shorter than the one between the same IoT devices and the Cloud.

Table 6. Comparison Among the FC Platforms Targeting Mobility Support

Platform	Migrates	Aimed at the IoT	Maturity
FMF [15]	Pending jobs	✓	Prototype
FMC/FME [171, 172]	Content and session		Prototype
Foglets [159]	Execution state at a coarse granularity	✓	Simulation
Bellavista et al. [18]	VMs		Prototype
Farris et al. [64]	Containers (stateless replication)		Prototype
Cloud4IoT [54]	Containers (stateless destruction and re-instantiation)	✓	Prototype
CFP [141]	Containers (stateful)	✓	Prototype

as a Cloud service is generally distant from an end device irrespective of the position of the latter. What has to be done to enable mobility support is to migrate the Fog service from one FN to another, keeping it close enough to the associated application component of the mobile IoT device. This leads to novel applications such as: (i) an autonomous drone whose flying logic runs as a Fog service; (ii) automotive services and automated driving in the IoV context [5]; (iii) Augmented Reality (AR) and Virtual Reality (VR) mobile applications; and (iv) smart healthcare applications employing wearable devices and FC. A more detailed discussion of mobility support in a Fog environment can be found in Reference [140].

In what follows, we provide an overview of the state-of-the-art platforms and policies that have been proposed in literature to support mobility in a Fog environment and then conclude with the main open issues in the field. While migration policies are extensively debated in other surveys [102, 107], we did not find any article reporting the FC platforms that specifically provide mobility support. We highlight that some of the literature referenced below does not specifically relate to IoMT, as it considers mobile devices, in general. However, the adopted approaches and techniques are very similar to those employed within an IoT context.

6.1.1 Platforms. Literature proposes FC platforms to support the mobility of end devices. Table 6 provides a summary and comparison of such platforms. In Reference [15], the authors present Follow Me Fog (FMF), a platform in which a Software as a Service (SaaS) server is hosted on each access point and provides resource-intensive services to mobile IoT devices. What is migrated here are the pending jobs offloaded by the mobile device. This platform migrates jobs any time that a handover occurs, which is not always necessary, and does not handle common scenarios in which there are two or more potential FNs given a specific access point. Follow Me Cloud (FMC) [172] and Follow Me Edge (FME) [171] mainly focus on content and session migration across FNs and heavily exploit elements and functionalities available in 3G, 4G, or 5G cellular networks. With the aim to make their proposal more generic, the authors of FMC improve it in Reference [92], adapting it to support mobile users connected also from networks other than the cellular one (e.g., Wi-Fi). Furthermore, the authors express concern about threats to service continuity, which is raised by the change of the IP addresses after node relocation(s). In Reference [92], the FMC concept is implemented exploiting the Locator/ID Separation Protocol (LISP),[10] while an SDN-based implementation is proposed in Reference [173].

In Reference [159], the authors present Foglets. This platform makes use of mobile agents [139] to implement service migration. The runtime state of a service is captured at a high level by the

[10] See http://lisp.cisco.com/lisp_over.html.

application itself and then migrated to a new node. This mechanism only captures the execution state at a coarse granularity (i.e., *weak mobility*), as it does not allow the destination node to restore the state of a thread at the exact instant of checkpointing. Furthermore, it is a responsibility of the application developer to implement such mechanisms. On the contrary, Bellavista et al. [18] present a platform capable of proactively migrating the whole runtime state of a service (i.e., *strong mobility*), by actually migrating Virtual Machines (VMs). More specifically, their proposal extends the Openstack++[11] platform to enable mobility support. As in Reference [15], the authors of this work only consider situations in which there is a single FN given a specific access point. Furthermore, they do not contemplate parameters such as the state of hardware resources in their migration decision-making.

Even though the choice between VMs and containers depends on the actual context and need, the latter are preferred more often to address the requirements of a Fog environment. Indeed, while VMs may represent a better choice for concerns such as multi-tenant isolation and software compatibility [73], containers are more lightweight and in general perform better [87, 146]. Such differences between these two technologies are mainly due to the fact that each VM has its own kernel, whereas all containers share the same kernel of the host system. Both References [54, 64] deal with containers but do not perform stateful migrations (i.e., those allowing both the whole runtime state and the persistent one to be available on the target node once the migration ends). Indeed, the first proposes a replication of stateless containers across FNs, while the second, which is called Cloud4IoT, destroys a container on the source node and statelessly re-instantiates it on the target node. The authors in Reference [141], instead, propose Companion Fog Platform (CFP), which performs stateful container migrations employing Docker[12] as containerization technology. Several techniques and relative implementations exist in literature to perform stateful migrations of VMs or containers. Nonetheless, it is worth noting that some approaches that are well established in a Cloud DC may not be equally appropriate going towards the network edge. This is caused by some of the aspects characterizing a FC environment, such as: (i) a reduced computing power of FNs with respect to Cloud servers; (ii) limited network performance of WANs; and (iii) the fact that the total migration time is of paramount importance, while in a Cloud DC it is only secondary to service downtime [72]. A comprehensive overview and comparison of stateful migration techniques together with the analysis of their aptness for a Fog environment can be found in Reference [142].

6.1.2 Policies. Another group of works focuses on the definition of a migration policy (i.e., when and where to migrate the Fog service). Markov Decision Process (MDP) is a commonly used framework for this purpose. In Reference [91], the authors of FMC model the service migration procedure as a distance-based MDP. In this first proposal, the user's mobility, which is not deterministic, is modelled and predicted through a one-dimensional (1D) mobility pattern. However, References [173, 187] formulate the service migration problem as a distance-based MDP where 2D mobility scenarios are captured. The same authors of Reference [187] advance an alternative solution method in Reference [184]. They establish a decoupling property of their initial MDP, which transforms it into two independent MDPs on disjoint state spaces. Lyapunov optimization can then be applied so that what is obtained is a simple deterministic (rather than stochastic) optimization problem. Another work from the same authors [36] contextualizes the mobility support issue in military environments rather than in commercial ones. Since military environments demand stronger security guarantees, a new parameter (i.e., security cost) is considered to make

[11] See https://github.com/OpenEdgeComputing/elijah-openstack.
[12] See https://www.docker.com/.

MDP-based migration decisions, together with the usual parameters (i.e., transmission and migration costs). The security cost of a migration increases when services of different users are hosted on the same physical node. In Reference [199], the authors employ an MDP to decide where (and not when) to migrate the Fog service; this work is worth mentioning because, unlike the aforesaid contributions, it also considers the network and FN states as parameters on which to base migration decisions. In Reference [138], which also uses MDPs, the authors propose to handle user mobility by either migrating the Fog service or by finding a new, more suitable communication path between it and the mobile node. Although MDPs are by far the most common way to define a migration policy, they are not the only one. For example, in Reference [194], the authors suggest making migration decisions to minimize the overall bandwidth consumption in a Fog-enabled Vehicular Cloud Computing (VCC) context; the problem is herein formulated as a Mixed-Integer Quadratic Programming (MIQP) problem.

6.1.3 Open Issues. Even though there are several contributions to mobility support in a Fog environment, there are still unexplored possibilities and room for further improvements. A first research direction may be to conceive new virtualization and migration techniques that are specifically tailored to the characteristics of a FC environment rather than a Cloud DC. Another possibility is to formulate optimal migration policies through uninvestigated frameworks, such as multi-objective genetic algorithms. To conclude, it might be beneficial to conceive mobility support solutions that exploit a federation among Fog providers, namely, the possibility to utilize computing resources of other providers on the basis of pre-established Service Level Agreements (SLA). Indeed, mobility support may be significantly improved if considering a federated Fog environment. For example, let us suppose that a mobile IoT device moves to an area in which there are no FNs belonging to its Fog provider, or where those available do not satisfy its requirements. In case of federation, the mobile device could rely on a suitable FN owned by a federated Fog provider.

6.2 Orchestration

To orchestrate computing resources and services means to coordinate, arrange, and jointly manage them to satisfy specific functional and non-functional requirements. For instance, service deployment and resource allocation, service coordination, and load balancing are all orchestration activities. A suitable orchestration is essential in every complex system to ensure efficiency and efficacy.

Resource and service orchestration is influenced by some of the distinguishing characteristics of FC (see Table 5). As a result, the orchestration techniques that are widely adopted in a Cloud DC cannot be always applied "as is" in the Fog but need to be customized for it [84]. Let us begin by examining the impact of FC distinctive features on orchestration. First, the heterogeneity of FNs imposes non-trivial orchestration issues [190]. It is fundamental to consider this diversity when deploying and coordinating services, since not all FNs are able to run all services. Two FNs that are identical in terms of hardware and software capabilities may be very different from one another due to their geographical distribution featured by FC and the requirement of topological proximity. For instance, it may happen that only one of them is suitable to host a specific Fog service, as the other may be not close enough to the IoT devices requiring that service. To further complicate matters, the high distribution featured by the Fog and its hierarchical nature imposes other challenges, namely, those regarding the management of large data volumes. Indeed, these are not only exchanged with a centralized Cloud DC but typically need to be orchestrated among the nodes along the continuum from Cloud to Things according to the nature and purpose of these data, the specific application scenario, and its requirements. Furthermore, the potentially wide-area distribution of Fog services and resources, together with the need for scalability and

the strict requirements of the IoT, naturally arises from centralized orchestration as in the Cloud DC to a distributed one where multiple orchestrators are arranged according to a hierarchical or flat architecture [84], and where each of them directly controls only a subset of nodes. Such orchestrators have to strongly coordinate with one another for the joint management of complex and distributed IoT applications. Moreover, the introduction of a great number of distributed FNs composing the Fog layer causes FC environments to be in general less energy-efficient than Cloud-only environments [46]. More specifically, the energy consumed by FNs represents the 60–80% of the overall energy consumed by systems that span from the things up to the Cloud [120]. To conclude, orchestration should be dynamic, i.e., should perform smart reconfigurations to adapt to the continuous changes that occur in the system. FC environments are highly dynamic [190] due to their intrinsic limitations in terms of computing power and network performance with respect to those in the Cloud DC and because of the strict requirements of IoT applications in terms of Quality of Service (QoS) and Quality of Experience (QoE). In what follows, we identify Fog orchestration architectures and policies that have been proposed in literature and conclude with the main open issues in the field.

6.2.1 Architectures. Hoque et al. [77] first analyse how the existing container orchestration tools address the requirements of FC and the IoT. Based on the obtained insights, they propose a container orchestration framework that bridges the found gap. Such a framework extends Docker Swarm,[13] which is extremely lightweight and rather complete, with an additional component called OpenIoTFog Agent, which is part of the OpenIoTFog toolkit.[14] The same authors detail an orchestration architecture for Fog environments in Reference [24]. This architecture is based on two essential components, namely, the Fog Orchestrator (FO), which runs on a central node, and the Fog Orchestration Agent (FOA), which runs on every FN. It is worth highlighting that in those cases in which there is no connectivity towards the FO, a FOA can become a FO for a subset of FNs. It will return a simple FOA if and when the connection to the central node resumes. Moreover, this architecture presents two main strengths. The first is the compliance with the recently released OFRA, which is discussed in Section 7. The second is its conformity with Topology and Orchestration Specification for Cloud Applications (TOSCA),[15] which is the de facto standard for modeling service orchestration. Yigitoglu et al. [196] propose Foggy, a framework for dynamic resource provisioning and IoT applications deployment in FC environments. The orchestration server, which runs on a central node and manages the whole system, obtains each application module requirements (i.e., priority, privacy, computation, latency, output) in JSON format and continuously monitors the system to capture every dynamic change and adapt the module placement accordingly. On the contrary, Reference [84] proposes a distributed orchestration architecture where each orchestrator controls a subset of resources and services. All these orchestrators are equally important (i.e., flat architecture) and coordinate with one other for the orchestration of the overall system.

With regard to Fog orchestration architectures, SDN plays a fundamental role [14]. Indeed, by separating the control plane (i.e., where the network control logic resides) from the data plane (i.e., the set of network devices forwarding packets), this technology enables a great network programmability and flexibility [180]. More specifically, SDN controllers have a comprehensive and constantly updated view of the dynamically changing network and computing resources and expose a northbound programming interface to network management applications to adaptively orchestrate resources, services, and network traffic according to QoS/QoE requirements and current system conditions [14]. The communication between the SDN controller and the

[13] See https://docs.docker.com/engine/swarm/.
[14] See https://openiotfog.org/en/.
[15] See http://docs.oasis-open.org/tosca/tosca-primer/v1.0/cnd01/tosca-primer-v1.0-cnd01.pdf.

data plane devices is commonly achieved through the OpenFlow[16] protocol. The authors in Reference [180] propose a Fog-IoT architecture where SDN is exploited as an orchestration and network control facility. This architecture conceives a cooperation among different SDN controllers, and FNs expose Application Programming Interfaces (APIs) to allow the remote monitoring and management of their resources. To conclude, References [76, 181] present SDN-based architectures for Fog orchestration in an IoV domain. In particular, Reference [76] focuses on load-balancing strategies.

6.2.2 Policies. In Reference [190], the authors present the preliminary results of their genetic algorithm for Fog orchestration. Although their proposal features some scalability limitations, it originally characterizes security risks as a cost to be minimized when performing orchestration. Instead, in Reference [174], the authors focus on the efficient utilization of computing resources as the primary concern in the formulation of their solution of Fog orchestration. Skarlat et al. [166] model Fog orchestration as an optimization problem and propose a genetic algorithm to solve it. This work envisions a distributed orchestration architecture where each orchestrator controls a subset of FNs (i.e., a Fog colony) and can be in turn controlled by another orchestrator that resides at a higher level in the hierarchy. The top-most orchestrator is in the Cloud. Both References [104, 105] particularly focus on QoS- and QoE-aware orchestration policies. More specifically, Reference [105] is based on Fuzzy logic, and Reference [104] also performs energy-aware Fog orchestration. Indeed, it proposes to re-locate application modules to optimize the number of active FNs and thus minimize energy consumption. It is worth noting that References [104, 105, 166] all simulate their solutions in iFogSim, which indeed is the most utilized tool to simulate resource management techniques in Fog-IoT environments [71]. As in Reference [104], Reference [120] also proposes an orchestration algorithm to find the optimal compromise between QoS and energy efficiency. Going into details, the authors propose to power FNs first via green energy (e.g., produced by sun or wind) and, when this is not available due to inappropriate weather conditions, via brown energy (e.g., produced through fossil fuels). The energy cost is represented only by brown energy consumption. To conclude, Reference [1] defines a scheme that incorporates service usage patterns and the history of service customers to dynamically estimate resources and adapt resource orchestration accordingly. The dynamic deployment of multicomponent IoT applications in Fog infrastructures is also addressed in Reference [25], where the authors propose a system model and present algorithms to determine eligible deployments.

6.2.3 Open Issues. At the moment of writing, there exist several northbound interfaces in between SDN controllers, on the one hand, and network management applications, on the other hand. Therefore, the definition of a vendor-independent northbound interface as a result of a standardisation effort would be a significant contribution [14]. Similarly, the research on the communication among peer SDN controllers is still at its beginning and thus is worth of investigation [14]. Finally, another research direction is the application of predictive analytics for a dynamic and proactive orchestration.

6.3 Deployment Models & Revenue Scenarios

An important challenge that needs to be faced to get a wider adoption of Fog-based systems consists in understanding the potential revenue and incentive models that can be supported through different deployment scenarios. Such models are needed to better understand why: (i) infrastructure providers would offer their resources to act as FNs; (ii) users would want to make use of these FC resources. We can consider FC deployments to be somewhat similar to the

[16] See https://www.opennetworking.org/wp-content/uploads/2013/04/openflow-spec-v1.3.1.pdf.

deployment of other types of edge infrastructures that currently exist, such as Wi-Fi deployments within cities, which may be operated and managed by a variety of different organizations, ranging from universities, coffee chains, transport operators/city councils, and so on. It is useful to note that not all such infrastructure deployments require payment from the end user. Understanding potential incentive models that encourage restaurant and café owners to operate Wi-Fi access points can be useful to understand this next generation of services, which are operated towards the network edge. However, this is still an open issue and is likely to grow as the FC infrastructure becomes more resilient and mature [136, 189].

Revenue models can be related to the characteristics identified in Table 5, where geographical distribution, node heterogeneity, and security requirements influence how FNs can generate a potential revenue stream for providers. More importantly, without an adequate number of FNs being available, sustaining a suitable infrastructure that provides suitable computing power and network performance will be unrealistic. Providing incentive models for provision and maintenance of FNs is essential. We consider the following four types of deployment models. The description below attempts to provide context for the deployment model based on the particular deployment approach being used:

- **Dynamic FN discovery supported revenue model**: This model involves dynamic discovery of a FN as a user moves from one location to another. The user device attempts to discover a FN in its "vicinity" using the advertised profile of the node (which can include: availability statistics, security credentials, and types of available services). Using this approach, the user does not have any guarantee that a suitable FN will be discovered to sustain an application session, but a negotiation can take place if multiple FNs are found. A user device can also cache previously seen FNs. The incentive for the provider is to gain revenue from each user session that is sustained using that FN. A user can purchase a subscription with particular FN types a priori (i.e., before discovery). A user is charged based on connection time, size of data, or range of services utilized. The deployment model in this case is the incentive for FN operators/owners to make services discoverable by IoT devices (including those that are mobile). The revenue earned by undertaking this would be the basis for the deployment model. Conversely, users/ owners of IoT devices need to determine whether a discovered service is suitable for their needs (taking account of a subscription cost to use the service). Discovering suitable services is akin to finding a service description match within a registry.
- **Pre-agreed contracts with Fog providers**: This deployment model involves generating pre-agreed contracts with operators of specific FNs—negotiated at a set price. Hence, there would be a preferential selection of particular nodes by a user if multiple choices are found. This also reduces risks for users, as security credentials would be included in these pre-agreed contracts and could be configured (e.g., use of particular encryption keys) beforehand. These pre-agreed contracts would need to comply with service level objectives (e.g., an availability profile) that an operator needs to meet. It is therefore possible that a FN operator may outsource their task to a Cloud provider. The incentive for the provider is to increase the number of potential subscribers by developing pre-agreed contracts. Capacity planning associated with such FNs is therefore dependent on accurately predicting potential future demand. The deployment model in this case involves agreeing a cost for entering into a contract with a Fog provider. This contract also allows preferential access to FNs owned by the provider.
- **FNs federation**: This deployment model involves multiple FN operators collaborating to share workload. To sustain potential revenue, this would imply federation between FNs

that exist within a particular geographical area. There would be a preferred cost for sharing workload with other providers, enabling revenue sharing between providers. To enable such an exchange to take place, it is necessary to identify how workload "units" can be characterized. This is equivalent to alliances set up between airline companies, for instance, where specialist capability (and capacity) available along a particular route can be shared across multiple operators. In the same way, if an operator deploys specialist GPUs or video analytics capability within a FN at a particular location, other operators could also make use of this in a seamless way and similarly share other capabilities in other locations. This type of geographic-centric specialization could enable localized investment within particular areas by operators.

- **Fog-Cloud exchange**: This deployment model involves a user device not being aware of the existence of any FN. Instead, the user device interacts with a Cloud operator who then attempts to find a FN in the vicinity of the user. Therefore, the Cloud operator needs to keep a track of the user location and discover suitable FN operators that could be used to support the session at a particular location. In this instance, the Cloud operator will always try to complete the user request first; however, if a QoS target is unlikely to be met due to latency constraints, it can outsource the user request to a regional FN. The incentive in this instance is to enable Fog-Cloud exchange contracts to be negotiated between providers [56].

Some of the above deployment and revenue generation scenarios are not unique to FC and closely relate to other similar efforts in service-oriented systems. We identify three **open issues** that could have an impact on realizing some of these deployment models in practice:

- The recent emergence of regulations such as the GDPR, which is being introduced in Europe, could have a significant impact on these deployment models. GDPR necessitates all external service providers who hold data about users to seek consent from users and state: (i) which data they hold; (ii) how these data are being used by the provider. More significantly, the user has the ability to revoke access to their data at any time. With the use of FNs, user data may be fragmented across different providers, depending on the mobility pattern of the user. Understanding how a group of FNs, which may not be part of a federated infrastructure, may seek consent of users remains a challenge.
- Vendors who own and operate an infrastructure at the network edge (e.g., cellular base stations) could become potential Fog providers in the future, as they are likely to provide the FN that a user interacts with. Deployment models that require interaction between such network operators and Cloud providers remain unclear at present.
- There is also potential for auction models that could operate in a FC environment when multiple FNs are available for a user to choose from. Understanding the metrics (other than price) that influence such auctions remains a challenge. Additionally, such auctions should not cause detrimental overhead on the performance of the application that makes use of the FC infrastructure. The definition of services that manage and operate such algorithms is also an open issue.

6.4 Security and Privacy

As reported in Table 2, one of the main advantages of FC over other approaches to Cloud-IoT integration is represented by security and privacy enforcements, especially with regard to the protection of sensitive data. Nevertheless, this advantage comes at the cost of new security and privacy challenges that are raised by some of the intrinsic characteristics of the Fog (see Table 5). More specifically, distributed systems are in general more vulnerable to attacks than centralized ones. Moreover, with the purpose to provide a better QoS/QoE and enable the distinguishing advantages

of FC, FNs are usually deployed in environments that are less protected than Cloud DCs [35]. To conclude, both the heterogeneity among FNs and their limited computing capabilities, if compared to Cloud servers, further complicate the situation. The security and privacy challenges afflicting FC have been significantly drawing the attention of the research community. This is demonstrated by the great number of works that have been proposed to face such challenges and, as a consequence, by the considerable number of surveys focusing on this topic [88, 118, 126, 150, 164, 167]. Among these surveys, Reference [126] is the only one that specifically discusses these challenges within the IoT context. Given this abundance of survey papers and for space reasons, what follows is a high-level overview of the main security and privacy concerns in a Fog-IoT environment.

The OFC dedicated an appendix of its OFRA document [42] to a detailed discussion about several security aspects in a FC environment. According to this appendix, security is the largest cross-cutting technical concern within critical IoT systems, which necessitate common baseline and interoperable standards to address security challenges within both hardware and software. Particularly interesting is the analysis of the hardware/firmware precautions that the Consortium suggests to implement a full-stack secure Chain of Trust comprised of trusted components. Among such components, IoT devices represent the most vulnerable elements of the FC hierarchy. Securing this part of the infrastructure is a promising research direction that has been only preliminary explored up to now, mainly relying on remote attestation techniques [21, 30].

With regard to the possible attacks against FNs, man-in-the-middle is one of the most important and urgently needs effective countermeasures. Being deployed in the field, FNs are vulnerable to this type of attack that consists of compromising a FN with malicious code [188] or even in replacing it with a fake FN [108].

From the point of view of the end users, privacy is beyond any doubt one of the most prominent requirements. An interesting research challenge (strictly connected to that of mobility support) in this field is related to the design and implementation of techniques able to guarantee the privacy of location and mobility data. As FC enables end users to offload their tasks to the nearest FNs, their location and trajectory can be retrieved by an attacker [118] (e.g., a malicious FNs administrator). This could even be the result of internal policies of Cloud/Fog providers that might act in an "honest-but-curious" way [126].

Finally, it is worth mentioning that security and privacy solutions in FC also have to take into consideration the complex combination of regional and governmental requirements that must be satisfied due to the widespread distribution of the nodes in a Fog hierarchy, as also explicitly stated in Reference [42]. This, however, is out of the scope of the present work.

7 FOG COMPUTING PLATFORMS FOR THE IOT

As a proof of the increasing maturity of the Fog paradigm, several software and hardware systems are already available for use. In this section, we provide an overview of existing FC platforms for the IoT. To the best of our knowledge, we are the first to make this novel contribution, which we believe may draw the attention of engineers and developers. Going into more details, we classify the discussed platforms into three categories, namely: (i) software platforms; (ii) development frameworks; and (iii) hardware platforms. The section then concludes with a discussion of OFC efforts towards a standardisation process that involves both FC software and hardware platforms.

7.1 Software Platforms

We define a FC software platform for the IoT as "*a software environment providing at least the basic functionalities and mechanisms that are necessary for the deployment and execution of IoT applications over a Fog infrastructure.*" We first discuss software platforms started as industrial initiatives, and then focus on open-source systems. Table 7 summarizes and compares these

Table 7. Comparison Among the FC Software Platforms for the IoT

Platform	Open-source	Extension of a Cloud platform	Only runs on specific hardware	Maturity
Nebbiolo			✓	Product
FogHorn Lightning				Product
Cisco IOx			✓	Product
Dell Edge Device Manager			✓	Product
IBM Watson IoT		✓		Product
AWS Greengrass		✓		Product
Microsoft Azure IoT Edge	✓[17]	✓		Pre-product
FogFlow	✓[18]			Pre-product
ParaDrop	✓[19]			Pre-product
OpenStack++	✓[20]	✓		Pre-product
Stack4Things	✓[21]	✓		Pre-product
OpenVolcano	✓[22]	✓		Pre-product

platforms on the basis of a set of features—we do not include features such as orchestration, as these are common across all platforms. Furthermore, we only discuss those platforms that are already available for use, namely, those whose maturity level is either *Pre-product* or *Product* (see Section 5). Nonetheless, there exist ongoing research activities likely to produce platforms in the near future [94, 115, 129].

7.1.1 Commercial Platforms. Nebbiolo Technologies was founded by Flavio Bonomi, who first advanced the concept of FC in 2012 (when he was with Cisco). The Nebbiolo Technologies FC platform [123] is a commercial platform consisting of a closed-source software stack that runs on a proprietary hardware solution and particularly tailored to the industrial automation sector. The platform allows a Cloud-like centralized management of distributed mini DCs deployed at the network edge. Such mini DCs comprise computing, networking, and storage resources in the form of purpose-built hardware nodes called fogNodes. This software platform includes the fogOS software stack, a custom operating system providing virtualization, SDN, data analytics, and security features. Moreover, the fogSM is a system manager, deployed in the Cloud or on-premises, which allows remote management of the fogNodes and assisted deployment of IoT applications.

FogHorn Lightning by FogHorn Systems [66] includes the FogHorn Manager that allows remote management, monitoring, and configuration of edge nodes, and deployment of IoT applications. Moreover, as described in Section 7.2, the company provides a powerful analytics framework enabling real-time and on-site stream processing of data coming from IoT devices. FogHorn Lightning does not exclusively run on a specific hardware.

As the biggest network appliance manufacturer, Cisco proposes a wide range of both FC software and hardware products for the IoT. The Cisco IOx [40] ecosystem provides uniform and consistent hosting capabilities for Fog applications across Cisco network infrastructure products.

[17] See https://github.com/Azure/iot-edge.
[18] See https://github.com/smartfog/fogflow.
[19] See https://github.com/ParadropLabs/Paradrop.
[20] See https://github.com/OpenEdgeComputing/elijah-openstack.
[21] See https://github.com/MDSLab/stack4things.
[22] See http://openvolcano.org/dokuwiki/doku.php?id=ov:download.

In particular, the Cisco IOx Fog Director provides users with the possibility to deploy, run, and monitor applications across the Fog infrastructure, while the Cisco IOx Client is a command-line utility for developers to control application lifecycle tasks within typical developer systems.

Similarly, Dell Technologies entered the market by proposing Dell Edge Device Manager [177], which enables secure registration of Dell hardware products and their remote management with automation of upgrades, task scheduling, real-time monitoring, and configuration.

Two other platforms are: (i) IBM Watson IoT [81], which extends IBM Cloud; and (ii) AWS Greengrass [9], which extends AWS Cloud. Both extend pre-existing proprietary Cloud platforms towards the network edge and provide support for deploying and running application components on IoT devices, edge nodes, and the Cloud.

7.1.2 Open-source Platforms. Similar to IBM and Amazon, Microsoft has recently released its FC software platform for the IoT, namely, Microsoft Azure IoT Edge [113], which extends Microsoft Azure Cloud. This platform is open-source but, at the moment of writing, is still in a preview phase.

FogFlow [34] is a FC software platform that is able to automatically and dynamically compose multiple tasks into high-level IoT services. Each task is represented by a Docker container hosting the data processing logic and needs to be described by the software developer through NGSI, the standard exploited within the FIWARE European project[23] for context information management. Based on such a description, FogFlow performs the orchestration in an optimized way, deploying tasks anywhere along the continuum from Cloud to Things, only when actually required, and based on the locality of data producers and consumers. Availability and mobility criteria are also taken into consideration by the system for task deployment.

Another FC software platform, which specifically targets nodes at the extreme edge of the wireless networks (i.e., home Wi-Fi routers and wireless gateways), is ParaDrop [98]. The attention to this specific kind of nodes is mainly motivated by their peculiar contextual knowledge about end user devices that are directly attached to them (e.g., proximity, characteristics of the channel). This knowledge is useful for making decisions about application placement and orchestration. Specifically, this platform can "paradrop" services from the Cloud to the network edge in the form of self-contained units, called "chutes," that are deployed as near as possible to the IoT devices requiring them (e.g., sensors, actuators, end user mobile devices). As common in many of these kinds of platforms, chutes are implemented as Docker containers. Due to such specific design choices, ParaDrop is particularly suitable for Smart Home and Smart Building scenarios. Although the range of currently supported nodes is still limited to a custom Wi-Fi access point based on the PC Engines APU2 single-board computer and few more nodes belonging to the Intel NUC family, there is the possibility to deploy a "ParaDrop router" as a QEMU/KVM VM.

To conclude, three open-source platforms integrate the Fog in OpenStack,[24] which is the most prominent open-source Cloud platform. The OpenStack project initiated in 2010 as a joint initiative of Rackspace Hosting and the NASA and is currently managed by the OpenStack Foundation, a non-profit corporate entity established in September 2012. More than 500 companies have joined the project since then, and the OpenStack development community currently counts more than 82,000 members from 187 countries around the world [130]. The first FC software platform extending OpenStack with Cloudlets support is OpenStack++ [74]. It is the output of the Open Edge Computing [128] initiative, which was launched in June 2015 by Vodafone, Intel, and Huawei in partnership with the Carnegie Mellon University. The second platform extending OpenStack with FC capabilities is Stack4Things [99], which was initially developed by the University of Messina

[23] See https://www.fiware.org/.
[24] See https://www.openstack.org/.

and is now commercialized by SmartME.io Srl. It provides functionalities for the remote management of IoT device fleets irrespective of their physical location, their network configuration, and their underlying technology. It is a Cloud-oriented horizontal solution providing IoT objects virtualization, customization, and orchestration. Last but not least, OpenVolcano [27, 131] is an open-source platform, conceived in the context of the Horizon 2020 INPUT project,[25] that specifically aims at supporting FC services in 5G-ready infrastructures. Besides extending OpenStack, it applies Network Functions Virtualization (NFV) and SDN through the OpenFlow protocol, thus enabling great network programmability and flexibility.

7.2 Development Frameworks

We define a FC development framework for the IoT as *"a set of tools (e.g., libraries, microservices, abstraction layers) easing the development of Fog applications for the IoT and assisting the developer in focusing on the application logic rather than on the distributed nature of the Fog infrastructure on top of which the application will be deployed."* Table 8 reports a comparison among the FC development frameworks for the IoT. Specifically, we report the information that we believe is more interesting from the point of view of the application developer, namely, if the framework is released under an open-source license, the supported programming languages, and the deployment model. Prior to starting, we point out that most of the development frameworks are tightly coupled with a FC software platform discussed in Section 7.1. To the best of our knowledge, only two frameworks are completely independent of the underlying FC software platform, thus totally decoupling application development from FN management and service deployment.

The most important initiative within this second category of development frameworks is the EdgeX Foundry project [67], which is hosted by the Linux Foundation. In April 2017, Dell Technologies, in conjunction with several partners and customers, launched the EdgeX Foundry project with the donation of about over 125,000 lines of code. The project is currently being actively developed by tens of companies including Samsung, Analog Devices, Toshiba, and others. EdgeX Foundry is a vendor-neutral open-source interoperability framework that allows developers to implement IoT applications in a hardware, Operating System (OS), and programming language agnostic way. It is composed of an ecosystem of microservices that can be combined and plugged together according to the application logic and/or easily replaced with open-source or proprietary solutions. The reference language is Java and at the core of the architecture lies a MongoDB database, which is used as a persistence mechanism for both the data collected by sensors and the metadata about the connected devices. A key aspect of the project is the certification program that aims at guaranteeing an overall ecosystem compatibility. Indeed, to be authorized to use the EdgeX trademark, vendors need the Project board to certify any commercial value-add that they build within the core framework, so that the core APIs are always supported [168].

macchina.io [70] is a toolkit that allows IoT developers to easily implement embedded applications on top of the most commonly used Linux-based single-board computers such as Raspberry Pi. It is based on a JavaScript and C++ runtime environment and provides several bundles implementing interfaces to devices and sensors, network protocols such as MQTT or COAP, interfaces to Cloud services (e.g., for sending SMS or Twitter messages), and a Web-based user interface. The core of the platform is represented by the POCO C++ libraries that implement essential features, e.g., platform abstraction, multithreading, stream, datagram and multicast sockets, HTTP server and client, SSL/TLS. macchina.io is released under the Apache 2.0 License.

[25] See https://www.input-project.eu/.

Table 8. Comparison Among the FC Development Frameworks for the IoT

Framework	Open-source	Coupled with a platform in 7.1	Supported languages	Deployment model
EdgeX Foundry	✓[26]		Java (officially supported) + others (from the community)	Docker containers
macchina.io	✓[27]		C++	Custom C-based runtime environment
Nebbiolo SDK		✓	Python	Docker containers
FogHorn Lightning SDK		✓	C++ (micro edition) + other not specified languages (standard edition)	n/a
Cisco IOx SDK		✓	C/C++, Python, Ruby, Nodejs	Custom containers, Docker containers, KVM/QEMU VMs

What follows is the set of FC development frameworks for the IoT that are part of a software platform discussed in Section 7.1. Within its proprietary ecosystem, Nebbiolo Technologies provides an SDK for the development of native applications on top of the fogOS software stack [125]. The reference language is Python, and the developers are provided with a set of tools that allow an application to be packaged within a Docker container and deployed onto the system in the form of a fogLet. A set of libraries are available to interact with the fogOS Pub/Sub Databus for data, events, and alarms propagation.

Similarly, the FogHorn Lightning platform provides developers with specific SDKs. This development framework is available in two editions, namely, standard and micro, which primarily differ from one another for their footprint. In the micro edition, a C++ SDK allows custom applications to implement data preprocessing, data visualization, and machine learning features at the edge. In the standard edition, a polyglot SDK further provides support for multiple industrial protocols (e.g., MQTT, Modbus). No open documentation is available on the system architecture; therefore, no details about the supported deployment methods can be provided.

Within the IOx [40] ecosystem, Cisco provides the Cisco IOx SDK and other development tools, which help developers to correctly package their applications for execution on Cisco IOx. The SDK allows developers to use several high-level languages, e.g., C/C++, Python, Ruby, Node.js and supports different categories of applications. Specifically, both containerized applications and VM-packaged applications are supported. The developer can use either an ad hoc LXC-compliant format or the Docker tooling to containerize applications. A KVM/QEMU hypervisor infrastructure is available for VM-packaged applications. Finally, the "IOx middleware services" provide high-level abstractions and APIs to facilitate the development of IOx applications.

7.3 Hardware Platforms

In this section, we report the hardware solutions that are provided by the most prominent hardware manufacturers on the market and that can play the role of FNs. Table 9 reports a comparison among such FC hardware platforms on the basis of those features that we believe are of particular interest in an IoT context. Specifically, besides some information about the hardware resources and the approximate price, we include details on: (i) the network connectivity; (ii) the additional interfaces that can be used to connect with external sensors and actuators (which represents the

[26] See https://github.com/edgexfoundry.
[27] See https://github.com/macchina-io/macchina.io.

Table 9. Comparison Among the FC Hardware Platforms for the IoT

Manufacturer	Model	Hardware resources	Network connectivity	Interfaces for external sensors and actuators	Hardware-based security	Price
Nebbiolo Technologies	fogNode	4–8 cores x86 i5/i7 CPUs, 8-16GB RAM	Ethernet (Wi-Fi and LTE are optional)	No	✓	n/a
TTTech	MFN 100	Intel Atom 4 cores 1.8GHz CPUs, 4–8GB RAM	Ethernet	2 USB ports		n/a
Cisco	800 Series Industrial Integrated Services Routers	Intel Atom 2 cores 1250MHz CPU, 2GB RAM	Ethernet, LTE (Wi-Fi is optional)	2 asynchronous serial interfaces	✓	2000$
Cisco	Compute Modules for the 1000 Series	AMD GX-410VC 4 cores 800MHz CPU, 4GB RAM	Ethernet	1 USB port		2000$
Dell	Edge Gateway 5000	Intel Atom E3825 CPU, 2GB RAM	Ethernet, Wi-Fi, BLE, LTE	6 different serial interfaces	✓	1000$
Dell	Embedded Box PCs	4 cores x86 i5/i7 CPU, 4–32GB RAM	Ethernet, Wi-Fi, BLE, LTE	5 USB ports, 3 different serial interfaces, GPIO	✓	1000$
HPE	GL20 IoT Gateway	Intel 4300U 2 cores i5 CPU, 8GB RAM	Ethernet, Wi-Fi (LTE is optional)	5 USB ports, 2 different serial interfaces		2000$
HPE	Edgeline EL1000/4000	1-4 Intel Xeon D 8-16 cores each, up to 128GB RAM	Ethernet	via PCIe expansion slots		3800$
Raspberry Pi Foundation	Raspberry Pi 3 Model B+	1.4GHz 4 cores ARM Cortex-A53 CPU, 1GB RAM	Ethernet, Wi-Fi, BLE	4 USB ports, 40 GPIO pins, Camera Serial Interface		35$
Qualcomm	DragonBoard 820c	2.35GHz 4 cores CPU, Adreno 530 GPU, 3GB RAM	Wi-Fi, Bluetooth	3 USB ports, pins, Camera Serial Interface		200$
Qualcomm	DragonBoard 410c	1.2GHz 4 cores ARM Cortex-A53 CPU, Adreno 306 GPU, 1GB RAM	Wi-Fi, Bluetooth	3 USB ports, pins, Camera Serial Interface		75$
Intel	Edison	500MHz 2 cores CPU, 100MHz MCU, 1GB RAM	Wi-Fi, Bluetooth	Total of 40 GPIO pins		50$

main difference between this kind of hardware products and the standard Cloud DC solutions); and (iii) the presence of hardware-based security solutions, such as Trusted Platform Module (TPM). By looking at Table 9, it is evident that FNs are very heterogeneous, especially in terms of hardware capabilities and therefore price.

Nebbiolo Technologies offers a series of modular hardware solutions, fully compliant with their FC software platform, called fogNodes [124]. fogNodes exist with a wide range of form factors and different computing capabilities, including standard x86 CPUs, FPGAs, and GPUs. Ethernet connectivity is available by default, while Wi-Fi and LTE interfaces come with optional modules. Particularly interesting is the presence of a TPM device onboard to provide hardware security capabilities. TTTech produces the MFN 100 [182], a device that can be employed as FN in industrial environments within the Nerve platform, which integrates the fogOS and the fogSM from Nebbiolo Technologies.

Cisco provides a series of network infrastructure products fully supporting the IOx ecosystem and thus allowing seamless deployment and execution of Fog applications. Specifically, the Cisco 800 Series Industrial Integrated Services Routers [39] are compact routers providing IoT gateway functionalities. They offer integrated 4G LTE connectivity, Ethernet ports, and a couple of asynchronous serial interfaces for sensors/actuators. The Cisco Compute Modules for the Cisco 1000 Series Connected Grid Routers [41] are field-replaceable modules that bring FC capabilities to already operational networks. They are specifically tailored to industrial IoT markets such as utilities, manufacturing, and Smart Cities.

Being primarily a hardware manufacturer, Dell Technologies provides enterprises with a portfolio of IoT-focused infrastructure products that allow them to build and deploy complete, secure, and scalable solutions from end IoT devices, to the network edge, and up to the Cloud [50]. In this regards, Dell Technologies portfolio includes the following products. On the one hand, the Dell Edge Gateway 5000 [178] is the flagship product of a family of IoT gateways that is equipped with a wide range of I/O connectors to bridge both legacy systems and modern sensors to the Internet but also provides enough computing/storage power to aggregate data and perform local analytics. On the other hand, the Dell Embedded Box PCs [179] seem to prioritize performance and adaptability to different use cases, rather than I/O connectivity. They are highly reliable devices for a variety of use cases, including process and discrete manufacturing, fleet management, kiosks, digital signage, surveillance, and automated retail solutions. Dell also provides Cloud DC solutions for advanced analytics, data management, storage, and computation, but these are out of the scope of this survey. Among the main hardware manufacturers, also HPE provides customers with a set of products that are specifically designed with the FC use case in mind. The HPE GL20 IoT Gateway [101] is a compact solution targeting verticals such as manufacturing, Smart Cities, oil and gas. Similarly to other manufacturers' IoT gateways, it comes with a set of I/O interfaces for connecting to IoT devices and with enough power to quickly elaborate data and react to critical situations. Products belonging to the HPE Edgeline family [100], such as the EL1000 and EL4000, instead, feature a reduced set of I/O interfaces but possess an expansible amount of hardware resources, which makes them similar to standard DC solutions.

There exists also a considerable number of less powerful FNs, which can be therefore employed in a more limited range of scenarios but are much cheaper than the previously discussed solutions. For instance, in Table 9, we report information on the Raspberry Pi 3 Model B+ single-board computer [137], which is powerful enough to behave as a FN. Indeed, the authors of Reference [17] demonstrate the feasibility of deploying Fog-IoT services as Docker containers on a Raspberry Pi. Other single-board computers that are worth mentioning as potential FNs are the Qualcomm DragonBoard 820c [144], the Qualcomm DragonBoard 410c [143], and the Intel Edison board [83].

7.4 Toward a Standardisation

The proliferation of proprietary solutions in ICT inevitably leads to delays in innovation and development and to strong limitations to the potential economic impact that ICT might have. Looking at the IoT, the McKinsey Global Institute states that interoperability is required on average for 40% of the total potential economic value that the IoT enables [106]. Therefore, it is time for technology suppliers to give birth to interoperable ecosystems by cooperating on the definition of standard technologies, protocols, and architectures.

Following this direction within the FC field, the OFC was founded in 2015 by ARM, Cisco, Dell, Intel, Microsoft, and the Princeton University and currently has 62 members throughout the world [44]. The stated objectives of the Consortium are: (i) to create an open, comprehensive reference architecture for the Fog; (ii) to promote the adoption of the Fog in the several application domains that may benefit from it; and (iii) to influence Fog standards development through liaisons with

standardisation bodies. In February 2017, the Consortium released the OFRA, thus paving the way to a multi-vendor interoperable FC ecosystem. More recently, in June 2018, the IEEE Standards Association (IEEE-SA) adopted the OFRA as an official standard [11], namely, the IEEE 1934.

We now provide a high-level overview of the salient characteristics of the above-mentioned architecture; further details may be found in Reference [42]. Eight core principles, known as pillars, guided the definition of the entire OFRA; they are: (i) security; (ii) scalability; (iii) openness; (iv) autonomy; (v) reliability, availability, and serviceability (RAS); (vi) agility; (vii) hierarchy; and (viii) programmability. Basically, the OFRA consists of five vertical perspectives and three horizontal views. Each perspective represents a cross-cutting concern that involves all the layers of the architecture. In other words, perspectives are the OpenFog pillars made integral part of the architecture itself. However, each of the three views is a set of layers that represents one or more specific aspects of the architecture. To be more precise, the Node View includes all the aspects of interest to the chip designers and the silicon manufacturers, as it clarifies the generic characteristics (e.g., computation, storage, networking) that a chip in a FN should possess. The actual FN is a composition of one or more chips (i.e., Node Views) with some additional elements. The higher the number of chips in a FN, the higher its expected positioning within the Fog hierarchy due to its greater capabilities. The OpenFog view that represents a FN is called System View, and typically the stakeholders interested in it are the system architects and the hardware Original Equipment Manufacturers (OEM). To conclude, the Software View characterizes the software running on a FN. It includes the software for the management of the node and its communications, the application services, and the software required to support them (e.g., VMs and containers, software libraries, databases, message brokers). As such, this view is of interest to the software architects and the application developers.

8 LESSONS LEARNT AND CONCLUSIONS

The Fog is a Cloud closer to the ground. As such, FC extends the Cloud toward the network edge (which does not mean only at the network edge), distributing resources and services of computing, storage, and networking anywhere along the Cloud-to-Things continuum. The resulting topological proximity to the end devices is the key enabler of innovative applications and services that were not conceivable when relying only on the distant Cloud. Moreover, although FC is tailored to the IoT, it is easily applicable in many other industry verticals that do not fall under the definition of IoT. In this article, we have provided a comprehensive survey on FC, with a specific focus on its employment within the IoT context. In what follows, we report the lessons learnt from this work by grouping them in two main categories.

FC is no more in its early stages. Since its very beginning, the Fog has been drawing the attention of both academia and industry. This growing interest toward FC has been contributing to a significant technological advancement in the field. Indeed, as we have shown in this survey, several scientific papers have proposed to employ the Fog in the most diverse IoT vertical domains, although its application within the ITS and Smart Healthcare has been investigated the most. Moreover, this survey has clearly highlighted how several ready-to-use software and hardware products already exist to realize FC environments for the IoT. Most of the open-source software platforms are an extension toward the network edge of a pre-existing Cloud platform (typically OpenStack), while commercial platforms are mostly independent solutions. Besides, the analysis of the available hardware platforms has clearly shown that these products greatly differ from one another, especially in terms of hardware resources and therefore price. This result confirms how the heterogeneity among FNs is one of those characteristics that distinguish FC the most from a Cloud-only environment. Last but not least, FC is experiencing significant standardisation efforts and promising collaborations, which are fundamental for a wider and quicker adoption of this

paradigm. In June 2018, the IEEE-SA officially adopted the OFRA as the new IEEE 1934 standard, while ETSI MEC will be a key feature of the next 5G networks. In addition, the recently signed MOU between the ETSI and the OFC is a first step toward further advancements in the field.

FC is far from complete and established. Although FC is an extension of the Cloud and resembles it in many respects, a Fog environment presents several characteristics (e.g., distribution, heterogeneity) that distinguish it from a Cloud-only one. The research community has been dealing with the challenges that derive from these distinctive characteristics of FC, and many results have been actually achieved over the years. Nevertheless, several open issues and research directions are still worthy of investigation for the final solution of such challenges. To conclude, in this survey, we have outlined how the application of FC within some IoT vertical domains has been less investigated than in others. For instance, there is still a lot of work that should be carried out to integrate the Fog into Smart Homes and Buildings.

REFERENCES

[1] M. Aazam, M. St-Hilaire, C. Lung, I. Lambadaris, and E. Huh. 2018. IoT resource estimation challenges and modeling in Fog. In *Fog Computing in the Internet of Things: Intelligence at the Edge*. Springer International Publishing, 17–31.

[2] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. 1999. Towards a better understanding of context and context-awareness. In *Proceedings of the International Symposium on Handeld and Ubiquitous Computing (HUC)*. 304–307.

[3] M. Ahmad, M. B. Amin, S. Hussain, B. H. Kang, T. Cheong, and S. Lee. 2016. Health fog: A novel framework for health and wellness applications. *Springer J. Supercomput.* 72, 10 (Oct. 2016), 3677–3695.

[4] Y. Ai, M. Peng, and K. Zhang. 2018. Edge cloud computing technologies for Internet of Things: A primer. *Dig. Commun. Netw.* 4, 2 (Apr. 2018), 77–86.

[5] A. Aissioui, A. Ksentini, A. Gueroui, and T. Taleb. 2018. On enabling 5G automotive systems using follow me edge-cloud concept. *IEEE Trans. Vehic. Technol.* 67, 6 (June 2018), 5302–5316.

[6] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. 2015. Internet of Things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surveys Tutor.* 17, 4 (2015), 2347–2376.

[7] S. Ali and M. Ghazal. 2017. Real-time heart attack mobile detection service (RHAMDS): An IoT use case for software defined networks. In *Proceedings of the 30th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE'17)*. 1–6.

[8] Amazon. 2017. AWS Network Latency Map. Retrieved from https://datapath.io/resources/blog/aws-network-latency-map/.

[9] Amazon. 2018. AWS Greengrass. Retrieved from https://aws.amazon.com/greengrass/.

[10] E. Amiot. 2015. *The Internet of Things: Disrupting Traditional Business Models*. Technical Report. Oliver Wyman. Retrieved from http://www.oliverwyman.com/content/dam/oliver-wyman/global/en/2015/jun/Internet-of-Things_Report.pdf.

[11] IEEE Standards Association. 2018. 1934—IEEE Approved Draft Standard for Adoption of OpenFog Reference Architecture for Fog Computing. Retrieved from https://standards.ieee.org/develop/project/1934.html.

[12] H. F. Atlam, R. J. Walters, and G. B. Wills. 2018. Fog Computing and the Internet of Things: A review. *J. Big Data Cogn. Comput.* 10, 2 (Apr. 2018).

[13] L. Atzori, A. Iera, and G. Morabito. 2010. The Internet of Things: A survey. *Comput. Netw.* 54, 15 (Oct. 2010), 2787–2805.

[14] A. C. Baktir, A. Ozgovde, and C. Ersoy. 2017. How can edge computing benefit from software-defined networking: A survey, use cases, and future directions. *IEEE Commun. Surveys Tutor.* 19, 4 (2017), 2359–2391.

[15] W. Bao, D. Yuan, Z. Yang, S. Wang, W. Li, B. B. Zhou, and A. Y. Zomaya. 2017. Follow me fog: Toward seamless handover timing schemes in a Fog Computing environment. *IEEE Commun. Mag.* 55, 11 (Nov. 2017), 72–78.

[16] F. Beligianni, M. Alamaniotis, A. Fevgas, P. Tsompanopoulou, P. Bozanis, and L. H. Tsoukalas. 2016. An Internet of Things architecture for preserving privacy of energy consumption. In *Proceedings of the Mediterranean Conference on Power Generation, Transmission, Distribution and Energy Conversion (MedPower'16)*. 1–7.

[17] P. Bellavista and A. Zanni. 2017. Feasibility of Fog Computing deployment based on docker containerization over RaspberryPi. In *Proceedings of the 18th International Conference on Distributed Computing and Networking (ICDCN'17)*.

[18] P. Bellavista, A. Zanni, and M. Solimando. 2017. A migration-enhanced edge computing support for mobile devices in hostile environments. In *Proceedings of the 13th International Wireless Communications and Mobile Computing Conference (IWCMC'17)*. 957–962.

[19] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. 2012. Fog Computing and its role in the Internet of Things. In *Proceedings of the 1st Workshop on Mobile Cloud Computing (MCC'12)*. 13–16.
[20] A. Botta, W. de Donato, V. Persico, and A. Pescapè. 2016. Integration of cloud computing and Internet of Things: A survey. *Future Gen. Comput. Syst.* 56 (Mar. 2016), 684–700.
[21] F. Brasser, K. B. Rasmussen, A. R. Sadeghi, and G. Tsudik. 2016. Remote attestation for low-end embedded devices: The prover's perspective. In *Proceedings of the 53rd ACM/EDAC/IEEE Design Automation Conference (DAC'16)*. 1–6.
[22] H. P. Breivold and K. Sandstrom. 2015. Internet of Things for industrial automation—Challenges and technical solutions. In *Proceedings of the IEEE International Conference on Data Science and Data Intensive Systems (DSDIS'15)*. 532–539.
[23] C. A. R. L. Brennand, F. D. da Cunha, G. Maia, E. Cerqueira, A. A. F. Loureiro, and L. A. Villas. 2016. FOX: A traffic management system of computer-based vehicles FOG. In *Proceedings of the 21st IEEE Symposium on Computers and Communications (ISCC'16)*. 982–987.
[24] M. S. De Brito, S. Hoque, T. Magedanz, R. Steinke, A. Willner, D. Nehls, O. Keils, and F. Schreiner. 2017. A service orchestration architecture for Fog-enabled infrastructures. In *Proceedings of the 2nd International Conference on Fog and Mobile Edge Computing (FMEC'17)*. 127–132.
[25] A. Brogi and S. Forti. 2017. QoS-aware deployment of IoT applications through the fog. *IEEE Internet Things J.* 4, 5 (Oct. 2017), 1185–1192.
[26] D. Bruneo, S. Distefano, F. Longo, G. Merlino, A. Puliafito, V. D'Amico, M. Sapienza, and G. Torrisi. 2016. Stack4Things as a Fog Computing platform for smart city applications. In *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM'16)*. 848–853.
[27] R. Bruschi, P. Lago, G. Lamanna, C. Lombardo, and S. Mangialardi. 2016. OpenVolcano: An open-source software platform for Fog Computing. In *Proceedings of the 28th International Teletraffic Congress (ITC'16)*. 22–27.
[28] R. Brzoza-Woch, M. Konieczny, P. Nawrocki, T. Szydlo, and K. Zielinski. 2016. Embedded systems in the application of Fog Computing—Levee monitoring use case. In *Proceedings of the 11th IEEE Symposium on Industrial Embedded Systems (SIES'16)*. 1–6.
[29] Y. Cao, S. Chen, P. Hou, and D. Brown. 2015. FAST: A Fog Computing assisted distributed analytics system to monitor fall for stroke mitigation. In *Proceedings of the IEEE International Conference on Networking, Architecture and Storage (NAS'15)*. 2–11.
[30] A. Celesti, M. Fazio, F. Longo, G. Merlino, and A. Puliafito. 2017. Secure registration and remote attestation of IoT devices joining the cloud: The Stack4Things case of study. In *Security and Privacy in Cyber-Physical Systems*. Wiley-Blackwell, Chap. 7, 137–156.
[31] M. Chen, S. Mao, and Y. Liu. 2014. Big data: A survey. *Mobile Netw. Appl.* 19, 2 (Apr. 2014), 171–209.
[32] N. Chen, Y. Chen, Y. You, H. Ling, P. Liang, and R. Zimmermann. 2016. Dynamic urban surveillance video stream processing using Fog Computing. In *Proceedings of the 2nd IEEE International Conference on Multimedia Big Data (BigMM'16)*. 105–112.
[33] Y. Chen, H. V. Leong, M. Xu, J. Cao, K. C. C. Chan, and A. T. S. Chan. 2006. In-network data processing for wireless sensor networks. In *Proceedings of the 7th International Conference on Mobile Data Management (MDM'06)*. 26–26.
[34] B. Cheng, G. Solmaz, F. Cirillo, E. Kovacs, K. Terasawa, and A. Kitazawa. 2018. FogFlow: Easy programming of IoT services over cloud and edges for smart cities. *IEEE Internet Things J.* 5, 2 (Apr. 2018), 696–707.
[35] M. Chiang and T. Zhang. 2016. Fog and IoT: An overview of research opportunities. *IEEE Internet Things J.* 3, 6 (Dec. 2016), 854–864.
[36] E. N. Ciftcioglu, K. S. Chan, R. Urgaonkar, S. Wang, and T. He. 2015. Security-aware service migration for tactical mobile micro-Clouds. In *Proceedings of the IEEE Military Communications Conference (MILCOM'15)*. 1058–1063.
[37] Cisco. 2015. *Fog Computing and the Internet of Things: Extend the Cloud to where the Things are*. Technical Report. Retrieved from https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf.
[38] Cisco. 2017. *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021*. Technical Report. Retrieved from https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf.
[39] Cisco. 2018. Cisco 800 Series Industrial Integrated Services Routers. Retrieved from https://www.cisco.com/c/en/us/products/routers/800-series-industrial-routers/index.html.
[40] Cisco. 2018. Cisco IOx. Retrieved from https://www.cisco.com/c/en/us/products/cloud-systems-management/iox/index.html.
[41] Cisco. 2018. Compute Modules for the Cisco 1000 Series Connected Grid Routers. Retrieved from https://www.cisco.com/c/en/us/products/collateral/routers/1000-series-connected-grid-routers/datasheet-c78-739683.html.
[42] OpenFog Consortium. 2017. *OpenFog Reference Architecture for Fog Computing*. Retrieved from https://www.openfogconsortium.org/wp-content/uploads/OpenFog_Reference_Architecture_2_09_17-FINAL.pdf.

[43] OpenFog Consortium. 2018. Definition of Fog Computing. Retrieved from https://www.openfogconsortium.org/resources/#definition-of-fog-computing.
[44] OpenFog Consortium. 2018. OpenFog Consortium—Member Companies. Retrieved from https://www.openfogconsortium.org/membership-information/#member-companies.
[45] OpenFog Consortium. 2018. Top 10 Myths of Fog Computing. Retrieved from https://www.openfogconsortium.org/top-10-myths-of-fog-computing/.
[46] A. V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, and R. Buyya. 2016. Fog Computing: Principles, architectures, and applications. Retrieved from arXiv:1601.02752.
[47] R. Dautov, S. Distefano, G. Merlino, D. Bruneo, F. Longo, and A. Puliafito. 2017. Towards a global intelligent surveillance system. In *Proceedings of the 11th International Conference on Distributed Smart Cameras (ICDSC'17)*. 119–124.
[48] M. S. de Brito, S. Hoque, R. Steinke, A. Willner, and T. Magedanz. 2017. Application of the Fog Computing paradigm to smart factories and cyber-physical systems. *Trans. Emerg. Telecommun. Technol.* 29, 4 (May 2017), 1–14.
[49] F. C. Delicato, P. F. Pires, and T. Batista. 2017. The resource management challenge in IoT. In *Resource Management for Internet of Things*. Springer International Publishing, 7–18.
[50] Dell Technologies. 2018. Gateways & Embedded Computing. Retrieved from http://www.dell.com/en-us/work/shop/cty/sc/gateways-embedded-pcs?stp_redir=false& ck=mn.
[51] N. Dhingra. 2014. Challenges, limitation and security issues on mobile computing. *Int. J. Curr. Eng. Technol.* 4, 5 (Oct. 2014), 3459–3462.
[52] G. Dimitrakopoulos and P. Demestichas. 2010. Systems based on cognitive networking principles and management functionality. *IEEE Vehic. Technol. Mag.* 5, 1 (Mar. 2010), 77–84.
[53] R. Drath and A. Horch. 2014. Industrie 4.0: Hit or hype? [Industry Forum]. *IEEE Industr. Electron. Mag.* 8, 2 (June 2014), 56–58.
[54] C. Dupont, R. Giaffreda, and L. Capra. 2017. Edge computing in IoT context: Horizontal and vertical linux container migration. In *Proceedings of the Global Internet of Things Summit (GIoTS'17)*. 1–4.
[55] J. Dutta and S. Roy. 2017. IoT-fog-cloud-based architecture for smart city: Prototype of a smart building. In *Proceedings of the 7th International Conference on Cloud Computing, Data Science & Engineering (CONFLUENCE'17)*. 237–242.
[56] A. Eivy. 2017. Be wary of the economics of "Serverless" cloud computing. *IEEE Cloud Comput.* 4, 2 (Mar. 2017), 6–12.
[57] A. M. Elmisery, S. Rho, and D. Botvich. 2016. A fog-based middleware for automated compliance with OECD privacy principles in Internet of Healthcare Things. *IEEE Access* 4 (Oct. 2016), 8418–8441.
[58] ETSI. 2017. ETSI and OpenFog Consortium collaborate on Fog and Edge Applications. Retrieved from http://www.etsi.org/news-events/news/1216-2017-09-news-etsi-and-openfog-consortium-collaborate-on-fog-and-edge-applications.
[59] ETSI. 2017. ETSI Multi-access Edge Computing Starts Second Phase and Renews Leadership Team. Retrieved from http://www.etsi.org/news-events/news/1180-2017-03-news-etsi-multi-access-edge-computing-starts-second-phase-and-renews-leadership-team.
[60] ETSI. 2018. Multi-access Edge Computing. Retrieved from http://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing.
[61] C. Fan, Z. Wu, C. Chang, and S. M. Yuan. 2016. Web resource cacheable edge device in Fog Computing. In *Proceedings of the 15th International Symposium on Parallel and Distributed Computing (ISPDC'16)*. 432–439.
[62] X. Fang, S. Misra, G. Xue, and D. Yang. 2012. Smart grid - the new and improved power grid: A survey. *IEEE Commun. Surveys Tutor.* 14, 4 (2012), 944–980.
[63] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, N. Constant, and K. Mankodiya. 2017. Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare. *Future Gen. Comput. Syst.* 78, 2 (May 2017), 659–676.
[64] I. Farris, T. Taleb, A. Iera, and H. Flinck. 2017. Lightweight service replication for ultra-short latency applications in mobile edge networks. In *Proceedings of the IEEE Int. Conf. Commun. (ICC'17)*. 1–6.
[65] N. Fernando, S. W. Loke, and W. Rahayu. 2013. Mobile cloud computing: A survey. *Future Gen. Comput. Syst.* 29, 1 (Jan. 2013), 84–106.
[66] FogHorn. 2018. FogHorn Lightning. Retrieved from https://www.foghorn.io/products/.
[67] Linux Foundation. 2018. EdgeX Foundry. Retrieved from https://www.edgexfoundry.org/.
[68] O. Fratu, C. Pena, R. Craciunescu, and S. Halunga. 2015. Fog computing system for monitoring mild dementia and COPD patients - romanian case study. In *Proceedings of the 12th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services (TELSIKS'15)*. 123–128.
[69] X. Ge, Z. Li, and S. Li. 2017. 5G software defined vehicular networks. *IEEE Commun. Mag.* 55, 7 (July 2017), 87–93.
[70] Applied Informatics Software Engineering GmbH. 2018. macchina.io. Retrieved from http://macchina.io.

[71] Harshit Gupta, Amir Vahid Dastjerdi, Soumya K. Ghosh, and Rajkumar Buyya. 2017. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, edge and Fog computing environments. *Softw.: Pract. Exper.* 47, 9 (June 2017), 1275–1296.

[72] K. Ha, Y. Abe, Z. Chen, W. Hu, B. Amos, P. Pillai, and M. Satyanarayanan. 2015. *Adaptive VM Handoff Across Cloudlets.* Technical Report. CMU School of Computer Science. Retrieved from http://elijah.cs.cmu.edu/DOCS/CMU-CS-15-113.pdf CMU-CS-15-113.

[73] K. Ha, Y. Abe, T. Eiszler, Z. Chen, W. Hu, B. Amos, R. Upadhyaya, P. Pillai, and M. Satyanarayanan. 2017. You can teach elephants to dance: Agile VM handoff for edge computing. In *Proceedings of the 2nd ACM/IEEE Symposium on Edge Computing (SEC'17).*

[74] K. Ha and M. Satyanarayanan. 2015. *OpenStack++ for Cloudlet Deployment.* Technical Report. CMU School of Computer Science. Retrieved from http://elijah.cs.cmu.edu/DOCS/CMU-CS-15-123.pdf.

[75] W. Han and Y. Xiao. 2016. Big data security analytic for smart grid with fog nodes. In *Proceedings of the 9th International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage (SpaCCS'16).* 59–69.

[76] X. He, Z. Ren, C. Shi, and J. Fang. 2016. A novel load-balancing strategy of software-defined cloud/fog networking in the Internet of Vehicles. *China Commun.* 13 (2016), 140–149.

[77] S. Hoque, M. S. d. Brito, A. Willner, O. Keil, and T. Magedanz. 2017. Towards container orchestration in Fog Computing infrastructures. In *Proceedings of the 41st IEEE International Computer Software and Applications Conference (COMPSAC'17).* 294–299.

[78] Lauren Horwitz. 2017. Edge Computing Technology Key to Future Enterprise, Gartner Says. Retrieved from https://www.cisco.com/c/en/us/solutions/internet-of-things/edge-computing-technology-gartner.html.

[79] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen. 2016. Vehicular Fog Computing: A viewpoint of vehicles as the infrastructures. *IEEE Trans. Vehic. Technol.* 65, 6 (June 2016), 3860–3873.

[80] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young. 2015. Mobile Edge Computing: A Key Technology Towards 5G. Retrieved from http://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf.

[81] IBM. 2018. IBM Watson IoT Platform. Retrieved from https://www.ibm.com/cloud/internet-of-things.

[82] IFRC. 2016. *World Disasters Report 2016.* Technical Report. Retrieved from http://www.ifrc.org/Global/Documents/Secretariat/201610/WDR%202016-FINAL_web.pdf.

[83] Intel. 2018. Intel Edison Development Platform. Retrieved from https://www.intel.com/content/dam/support/us/en/documents/edison/sb/edison_pb_331179002.pdf.

[84] Y. Jiang, Z. Huang, and D. H. K. Tsang. 2018. Challenges and solutions in Fog Computing orchestration. *IEEE Netw.* 32, 3 (May/June 2018), 122–129.

[85] K. Kai, W. Cong, and L. Tao. 2016. Fog computing for vehicular ad hoc networks: Paradigms, scenarios, and issues. *J. China Univ. Posts Telecommun.* 23, 2 (Apr. 2016), 56–65.

[86] M. Kanellos. 2016. How to Keep the Internet of Things from Breaking the Internet. Retrieved from https://www.forbes.com/sites/michaelkanellos/2016/06/16/how-to-keep-the-internet-of-things-from-breaking-the-internet/#5d210e2e6a7c.

[87] M. B. A. Karim, B. I. Ismail, W. M. Tat, E. M. Goortani, S. Setapa, J. Y. Luke, and H. Ong. 2016. Extending cloud resources to the edge: Possible scenarios, challenges, and experiments. In *Proceedings of the International Conference on Cloud Computing Research and Innovations (ICCCRI'16).* 78–85.

[88] S. Khan, S. Parkinson, and Y. Qin. 2017. Fog Computing security: A review of current applications and security solutions. *J. Cloud Comput.* 6, 1 (Aug. 2017).

[89] O. T. T. Kim, N. D. Tri, V. D. Nguyen, N. H. Tran, and C. S. Hong. 2015. A shared parking model in vehicular network using fog and cloud environment. In *Proceedings of the 17th Asia-Pacific Network Operations and Management Symposium (APNOMS'15).* 321–326.

[90] F. A. Kraemer, A. E. Braten, N. Tamkittikhun, and D. Palma. 2017. Fog Computing in healthcare—A review and discussion. *IEEE Access* 5 (May 2017), 9206–9222.

[91] A. Ksentini, T. Taleb, and M. Chen. 2014. A Markov decision process-based service migration procedure for follow me cloud. In *Proceedings of the IEEE International Conference on Communications (ICC'14).* 1350–1354.

[92] A. Ksentini, T. Taleb, and F. Messaoudi. 2014. A LISP-based implementation of follow me cloud. *IEEE Access* 2 (September 2014), 1340–1347.

[93] C. Lai, D. Song, R. Hwang, and Y. Lai. 2016. A QoS-aware streaming service over Fog Computing infrastructures. In *Proceedings of the Digital Media Industry & Academic Forum (DMIAF'16).* 94–98.

[94] A. Lebre, J. Pastor, A. Simonet, and F. Desprez. 2017. Revising OpenStack to operate Fog/Edge Computing infrastructures. In *Proceedings of the IEEE International Conference on Cloud Engineering (IC2E'17).* 138–148.

[95] J. Liu, J. Wan, D. Jia, B. Zeng, D. Li, C. Hsu, and H. Chen. 2017. High-efficiency urban-traffic management in context-aware computing and 5G communication. *IEEE Commun. Mag.* 55, 1 (Jan. 2017), 34–40.

[96] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu. 2017. A scalable and quick-response software defined vehicular network assisted by mobile edge computing. *IEEE Commun. Mag.* 55, 7 (July 2017), 94–100.

[97] P. Liu, D. Willis, and S. Banerjee. 2016. ParaDrop: Enabling lightweight multi-tenancy at the network's extreme edge. In *Proceedings of the IEEE/ACM Symposium on Edge Computing (SEC'16)*. 1–13.

[98] P. Liu, D. Willis, and S. Banerjee. 2016. ParaDrop: Enabling lightweight multi-tenancy at the network extreme edge. In *Proceedings of the IEEE/ACM Symposium on Edge Computing (SEC'16)*. 1–13.

[99] F. Longo, D. Bruneo, S. Distefano, G. Merlino, and A. Puliafito. 2017. Stack4Things: A sensing-and-actuation-as-a-service framework for IoT and cloud integration. *Ann. Telecommun.* 72, 1 (Feb. 2017), 53–70.

[100] Hewlett Packard Enterprise Development LP. 2018. HPE Edgeline EL1000 and EL4000. Retrieved from https://h20195.www2.hpe.com/v2/GetPDF.aspx/4AA6-6095ENN.pdf.

[101] Hewlett Packard Enterprise Development LP. 2018. HPE GL20 IoT Gateway. Retrieved from https://www.hpe.com/us/en/product-catalog/servers/edgeline-systems/pip.hpe-edgeline-el20-intelligent-gateway.1008670391.html.

[102] P. Mach and Z. Becvar. 2017. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys & Tutorials* 19, 3 (2017), 1628–1656.

[103] R. Mahmud, R. Kotagiri, and R. Buyya. 2017. Fog Computing: A taxonomy, survey and future directions. In *Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives*. Springer, Singapore, 103–130.

[104] R. Mahmud, K. Ramamohanarao, and R. Buyya. 2018. Latency-aware application module management for Fog Computing environments. *ACM Trans. Internet Technol.* 19, 1, Article 9 (Nov. 2018).

[105] R. Mahmud, S. N. Srirama, K. Ramamohanarao, and R. Buyya. 2018. Quality of experience (QoE)-aware placement of applications in Fog Computing environments. *J. Parallel Distrib. Comput.* (Mar. 2018). Article in press.

[106] J. Manyika, M. Chui, P. Bisson, J. Woetzel, R. Dobbs, J. Bughin, and D. Aharon. 2015. *The Internet of Things: Mapping the Value Beyond the Hype*. Technical Report. McKinsey Global Institute. Retrieved from http://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world.

[107] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. 2017. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surveys Tutor.* 19, 4 (2017), 2322–2358.

[108] E. Marìn-Tordera, X. Masip-Bruin, J. Garcìa-Alminana, A. Jukan, G-J. Ren, and J. Zhu. 2017. Do we all really know what a fog node is ? Current trends towards an open definition. *Comput. Commun.* 109 (Sep. 2017), 117–130.

[109] MarketsandMarkets. 2016. *Fog Computing Market by Offering (Hardware, Software), Application (Building & Home Automation, Smart Energy, Smart Manufacturing, Transportation & Logistics, Connected Health, Security & Emergencies), and Geography—Global Forecast to 2022*. Technical Report. Retrieved from https://www.marketsandmarkets.com/pdfdownload.asp?id=28314581.

[110] X. Masip-Bruin, E. Marìn-Tordera, A. Gòmez, V. Barbosa, and A. Alonso. 2016. Will it be cloud or will it be fog? F2C, a novel flagship computing paradigm for highly demanding services. In *Proceedings of the Future Technologies Conference (FTC'16)*. 1129–1136.

[111] R. Mayer, H. Gupta, E. Saurez, and U. Ramachandran. 2017. The fog makes sense: Enabling social sensing services with limited Internet connectivity. In *Proceedings of the 2nd International Workshop on Social Sensing (SocialSens'17)*. 61–66.

[112] B. Mei, R. Li, W. Cheng, J. Yu, and X. Cheng. 2017. Ultraviolet radiation measurement via smart devices. *IEEE Internet Things J.* 4, 4 (June 2017), 934–944.

[113] Microsoft. 2018. Microsoft Azure IoT Edge. Retrieved from https://azure.microsoft.com/en-us/services/iot-edge/.

[114] A. Monteiro, H. Dubey, L. Mahler, Q. Yang, and K. Mankodiya. 2016. FIT: A Fog Computing device for speech tele-treatments. In *Proceedings of the 2nd IEEE International Conference on Smart Computing (SMARTCOMP'16)*. 1–3.

[115] R. S. Montero, E. Rojas, A. A. Carrillo, and I. M. Llorente. 2017. Extending the cloud to the network edge. *Computer* 50, 4 (Apr. 2017), 91–95.

[116] N. H. Motlagh, M. Bagaa, and T. Taleb. 2017. UAV-based IoT platform: A crowd surveillance use case. *IEEE Commun. Mag.* 55, 2 (Feb. 2017), 128–134.

[117] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos. 2018. A comprehensive survey on Fog Computing: State-of-the-art and research challenges. *IEEE Commun. Surveys Tutor.* 20, 1 (2018), 416–464.

[118] M. Mukherjee, R. Matam, L. Shu, L. Maglaras, M. A. Ferrag, N. Choudhury, and V. Kumar. 2017. Security and privacy in Fog Computing: Challenges. *IEEE Access* 5 (Sep. 2017), 19293–19304.

[119] K. Nahrstedt, H. Li, P. Nguyen, S. Chang, and L. Vu. 2016. Internet of mobile things: Mobility-driven challenges, designs and implementations. In *Proceedings of the 1st International Conference on Internet-of-Things Design and Implementation (IoTDI'16)*. 25–36.

[120] Y. Nan, W. Li, W. Bao, F. C. Delicato, P. F. Pires, Y. Dou, and A. Y. Zomaya. 2017. Adaptive energy-aware computation offloading for cloud of things systems. *IEEE Access* 5 (Oct. 2017), 23947–23957.
[121] M. S. H. Nazmudeen, A. T. Wan, and S. M. Buhari. 2016. Improved throughput for power line communication (PLC) for smart meters using Fog Computing-based data aggregation approach. In *Proceedings of the 2nd IEEE International Smart Cities Conference: Improving the Citizens Quality of Life (ISC'16)*. 1–4.
[122] Nebbiolo. 2017. *Fog Computing: Keystone of Industrial IoT and Industry 4.0*. Technical Report. Retrieved from https://www.nebbiolo.tech/wp-content/uploads/Nebbiolo-Technologies-solutions-brief.pdf.
[123] Nebbiolo. 2018. Nebbiolo Fog Computing Platform. Retrieved from https://www.nebbiolo.tech/.
[124] Nebbiolo. 2018. Nebbiolo fogNode. Retrieved from https://www.nebbiolo.tech/wp-content/uploads/fogNode-OVERVIEW-rev3.pdf.
[125] Nebbiolo. 2018. Nebbiolo SDK. Retrieved from https://docs.nebbiolo.io/latest/sdk-guide/services/installSDK/.
[126] J. Ni, K. Zhang, X. Lin, and X. S. Shen. 2018. Securing Fog Computing for Internet of Things applications: Challenges and solutions. *IEEE Commun. Surveys Tutor.* 20, 1 (Oct. 2018), 601–628.
[127] F. Y. Okay and S. Ozdemir. 2016. A Fog Computing-based smart grid model. In *Proceedings of the International Symposium on Networks, Computers and Communications (ISNCC'16)*. 1–6.
[128] OpenEdgeComputing. 2018. OpenEdgeComputing—Home Page. Retrieved from http://openedgecomputing.org.
[129] OpenStack. 2018. Fog Edge Massively Distributed Clouds Group of Interest—Home Page. Retrieved from https://wiki.openstack.org/wiki/Fog_Edge_Massively_Distributed_Clouds.
[130] OpenStack. 2018. OpenStack Foundation—Home Page. Retrieved from https://www.openstack.org/foundation.
[131] OpenVolcano. 2018. OpenVolcano—Home Page. Retrieved from http://openvolcano.org/.
[132] A. K. Pathan and R. Buyya. 2007. *A Taxonomy and Survey of Content Delivery Networks*. Technical Report. Retrieved from http://www.cloudbus.org/reports/CDN-Taxonomy.pdf.
[133] G. Peralta, M. Iglesias-Urkia, M. Barcelo, R. Gomez, A. Moran, and J. Bilbao. 2017. Fog Computing-based efficient IoT scheme for the industry 4.0. In *Proceedings of the IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM'17)*. 1–6.
[134] C Perera, Y Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos. 2017. Fog Computing for sustainable smart cities: A survey. *Comput. Surveys* 50, 3 (June 2017).
[135] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. 2014. Context aware computing for the Internet of Things: A survey. *IEEE Commun. Surveys Tutor.* 16, 1 (2014), 414–454.
[136] I. Petri, O. F. Rana, J. Bignell, S. Nepal, and N. Auluck. 2017. Incentivising resource sharing in edge computing applications. In *Proceedings of the International Conference on the Economics of Grids, Clouds, Systems, and Services (GECON'17)*. 204–215.
[137] Raspberry Pi. 2018. Raspberry Pi 3 Model B+. Retrieved from https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/.
[138] J. Plachy, Z. Becvar, and E. C. Strinati. 2016. Dynamic resource allocation exploiting mobility prediction in mobile edge computing. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'16)*. 1–6.
[139] A. Poggi and M. Tomaiuolo. 2011. Mobile agents: Concepts and technologies. In *Handbook of Research on Mobility and Computing: Evolving Technologies and Ubiquitous Impacts*. IGI Global, 343–355.
[140] C. Puliafito, E. Mingozzi, and G. Anastasi. 2017. Fog Computing for the Internet of mobile things: Issues and challenges. In *Proceedings of the 3rd IEEE International Conference on Smart Computing (SMARTCOMP'17)*. 1–6.
[141] C. Puliafito, E. Mingozzi, C. Vallati, F. Longo, and G. Merlino. 2018. Companion Fog Computing: Supporting things mobility through container migration at the edge. In *Proceedings of the 4th IEEE International Conference on Smart Computing (SMARTCOMP'18)*. 97–105.
[142] C. Puliafito, E. Mingozzi, C. Vallati, F. Longo, and G. Merlino. 2018. Virtualization and migration at the network edge: An overview. In *Proceedings of the 4th IEEE International Conference on Smart Computing (SMARTCOMP'18)*. 368–374.
[143] Qualcomm. 2018. DragonBoard 410c Development Board. Retrieved from https://developer.qualcomm.com/hardware/dragonboard-410c.
[144] Qualcomm. 2018. DragonBoard 820c Development Board. Retrieved from https://developer.qualcomm.com/hardware/dragonboard-820c.
[145] A. M. Rahmani, T. N. Gia, B. Negash, A. Anzanpour, I. Azimi, M. Jiang, and P. Liljeberg. 2017. Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A Fog Computing approach. *Future Gen. Comput. Syst.* 78, 2 (Feb. 2017), 641–658.
[146] F. Ramalho and A. Neto. 2016. Virtualization at the network edge: A performance comparison. In *Proceedings of the 17th IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM'16)*. 1–6.

[147] A. Rauniyar, P. Engelstad, B. Feng, and D. V. Thanh. 2016. Crowdsourcing-based disaster management using Fog Computing in Internet of things paradigm. In *Proceedings of the 2nd IEEE International Conference on Collaboration and Internet Computing (CIC'16)*. 490–494.
[148] 451 Research. 2017. *Size and Impact of Fog Computing Market*. Technical Report. Retrieved from https://www.openfogconsortium.org/growth/.
[149] ABI Research. 2015. Data Captured by IoT Connections to Top 1.6 Zettabytes in 2020, as Analytics Evolve from Cloud to Edge. Retrieved from https://www.abiresearch.com/press/data-captured-by-iot-connections-to-top-16-zettaby/.
[150] R. Roman, J. Lopez, and M. Mambo. 2016. Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges. *Future Gen. Comput. Syst.* 78, 2 (Nov. 2016), 680–698.
[151] M. Sapienza, E. Guardo, M. Cavallo, G. La Torre, G. Leombruno, and O. Tomarchio. 2016. Solving critical events through mobile edge computing: An approach for smart cities. In *Proceedings of the 2nd IEEE International Conference on Smart Computing (SMARTCOMP'16)*. 1–5.
[152] S. Sareen, S. K. Gupta, and S. K. Sood. 2017. An intelligent and secure system for predicting and preventing Zika virus outbreak using Fog Computing. *Enterpr. Info. Syst.* 11, 9 (Jan. 2017), 1436–1456.
[153] M. Satyanarayanan. 2001. Pervasive computing: Vision and challenges. *IEEE Person. Commun.* 8, 4 (Aug. 2001), 10–17.
[154] M. Satyanarayanan. 2017. The emergence of edge computing. *Computer* 50, 1 (Jan. 2017), 30–39.
[155] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. 2009. The case for VM-based cloudlets in mobile computing. *IEEE Pervas. Comput.* 8, 4 (Oct. 2009), 14–23.
[156] M. Satyanarayanan, Z. Chen, K. Ha, W. Hu, W. Richter, and P. Pillai. 2014. Cloudlets: At the leading edge of mobile-cloud convergence. In *Proceedings of the 6th International Conference on Mobile Computing, Applications and Services (MobiCASE'14)*. 1–9.
[157] M. Satyanarayanan, G. Lewis, E. Morris, S. Simanta, J. Boleng, and K. Ha. 2013. The role of cloudlets in hostile environments. *IEEE Pervas. Comput.* 12, 4 (Oct. 2013), 40–49.
[158] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos. 2015. Edge analytics in the Internet of Things. *IEEE Pervas. Comput.* 14, 2 (Apr. 2015), 24–31.
[159] E. Saurez, K. Hong, D. Lillethun, U. Ramachandran, and B. Ottenwalder. 2016. Incremental deployment and migration of geo-distributed situation awareness applications in the Fog. In *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems (DEBS'16)*. 258–269.
[160] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. A. Ashraf, B. Almeroth, J. Voigt, I. Riedel, A. Puschmann, A. Mitschele-Thiel, M. Muller, T. Elste, and M. Windisch. 2017. Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture. *IEEE Commun. Mag.* 55, 2 (Feb. 2017), 70–78.
[161] A. Seitz, J. O. Johanssen, B. Bruegge, V. Loftness, V. Hartkopf, and M. Sturm. 2017. A Fog architecture for decentralized decision making in smart buildings. In *Proceedings of the 2nd International Workshop on Science of Smart City Operations and Platforms Engineering (SCOPE'17)*. 34–39.
[162] W. Shi and S. Dustdar. 2016. The promise of edge computing. *Computer* 49, 5 (May 2016), 78–81.
[163] S. Shin, S. Seo, S. Eom, J. Jung, and K. H. Lee. 2016. A Pub/Sub-based Fog Computing architecture for Internet-of-Vehicles. In *Proceedings of the IEEE International Conference on Cloud Computing Technology and Science (CloudCom'16)*. 90–93.
[164] S. N. Shirazi, A. Gouglidis, A. Farshad, and D. Hutchison. 2017. The extended cloud: Review and analysis of mobile edge computing and Fog from a security and resilience perspective. *IEEE J. Select. Areas Commun.* 35, 11 (Nov. 2017), 2586–2595.
[165] Y. Simmhan. 2017. Big data and Fog Computing. Retrieved from arXiv:1712.09552.
[166] O. Skarlat, M. Nardelli, S. Schulte, M. Borkowski, and P. Leitner. 2017. Optimized IoT service placement in the Fog. *Serv. Orient. Comput. Appl.* 11, 4 (Dec. 2017), 427–443.
[167] I. Stojmenovic, S. Wen, X. Huang, and H. Luan. 2015. An overview of Fog Computing and its security issues. *Concurr. Comput.: Pract. Exper.* 28, 10 (Apr. 2015), 2991–3005.
[168] Moor Insights & Strategy. 2017. *Cleaning up the Industrial IoT Edge*. Technical Report. Retrieved from http://www.moorinsightsstrategy.com/wp-content/uploads/2017/04/CLEANING-UP-THE-INDUSTRIAL-IOT-IIOT-EDGE-By-Moor-Insights-and-Strategy.pdf.
[169] G. Suciu, E. G. Ularu, and R. Craciunescu. 2012. Public versus private Cloud adoption—A case study based on open source cloud platforms. In *Proceedings of the 20th Telecommunications Forum (TELFOR'12)*. 494–497.
[170] K. Suto, H. Nishiyama, N. Kato, and C. W. Huang. 2015. An energy-efficient and delay-aware wireless computing system for industrial wireless sensor networks. *IEEE Access* 3 (June 2015), 1026–1035.
[171] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck. 2017. Mobile edge computing potential in making cities smarter. *IEEE Commun. Mag.* 55, 3 (Mar. 2017), 38–43.
[172] T. Taleb and A. Ksentini. 2013. Follow me cloud: Interworking distributed clouds & distributed mobile networks. *IEEE Netw.* 27, 5 (Oct. 2013), 12–19.

[173] T. Taleb, A. Ksentini, and P. Frangoudis. 2018. Follow-me cloud: When cloud services follow mobile users. *IEEE Trans. Cloud Comput.* (Feb. 2016). Article in press.

[174] M. Taneja and A. Davy. 2017. Resource aware placement of IoT application modules in Fog-cloud computing paradigm. In *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM'17)*. 1222–1228.

[175] G. Tanganelli, C. Vallati, and E. Mingozzi. 2017. A Fog-based distributed look-up service for intelligent transportation systems. In *Proceedings of the 18th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'17)*. 1–6.

[176] M. Tao, K. Ota, and M. Dong. 2017. Foud: Integrating Fog and cloud for 5G-enabled V2G networks. *IEEE Netw.* 31, 2 (Mar./Apr. 2017), 8–13.

[177] Dell Technologies. 2018. Dell Edge Device Manager. Retrieved from http://delliotpartners.com/#!/edgedevicemanager/overview.

[178] Dell Technologies. 2018. Dell Edge Gateway 5000. Retrieved from http://www.dell.com/en-us/work/shop/gateways-embedded-computing/edge-gateway-5000/spd/dell-edge-gateway-5000/xctoi5000us.

[179] Dell Technologies. 2018. Dell Embedded Box PCs. Retrieved from http://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/specsheet-dell-embedded-box-PC-3000-5000.pdf.

[180] S. Tomovic, K. Yoshigoe, I. Maljevic, and I. Radusinovic. 2017. Software-defined Fog network architecture for IoT. *Wireless Person. Commun.* 92, 1 (Jan. 2017), 181–196.

[181] N. B. Truong, G. M. Lee, and Y. Ghamri-Doudane. 2015. Software defined networking-based vehicular adhoc network with fog computing. In *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM'15)*. 1202–1207.

[182] TTTech. 2018. MFN 100 Edge Computing Device. Retrieved from https://www.tttech.com/fileadmin/content/industrial/files/secure/flyer/TTTech_MFN-100.pdf.

[183] United Nations, Department of Economic and Social Affairs, Population Division. 2014. *World Urbanization Prospects: The 2014 Revision, Highlights.* Technical Report. Retrieved from https://esa.un.org/unpd/wup/publications/files/wup2014-highlights.Pdf.

[184] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung. 2015. Dynamic service migration and workload scheduling in edge-clouds. *Perform. Eval.* 91 (Sept. 2015), 205–228.

[185] C. Vallati, A. Virdis, E. Mingozzi, and G. Stea. 2016. Mobile-edge computing come home - connecting things in future smart homes using LTE device-to-device communications. *IEEE Consum. Electron. Mag.* 5, 4 (Oct. 2016), 77–83.

[186] B. Varghese, N. Wang, D. S. Nikolopoulos, and R. Buyya. 2017. Feasibility of Fog Computing. Retrieved from arXiv:1701.05451.

[187] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung. 2015. Dynamic service migration in mobile edge-clouds. In *Proceedings of the IFIP Networking Conference*. 1–9.

[188] Y. Wang, T. Uehara, and R. Sasaki. 2015. Fog Computing: Issues and challenges in security and forensics. In *Proceedings of the 39th IEEE Conference on Computers, Software and Applications (COMPSAC'15)*. 53–59.

[189] J. Weinman. 2018. The 10 laws of fogonomics. *IEEE Cloud Comput.* 4, 6 (Nov. 2018), 8–14.

[190] Z. Wen, R. Yang, P. Garraghan, T. Lin, J. Xu, and M. Rovatsos. 2017. Fog orchestration for Internet of Things services. *IEEE Internet Comput.* 21, 2 (Mar. 2017), 16–24.

[191] D. Wu, S. Liu, L. Zhang, J. Terpenny, R. X. Gao, T. Kurfess, and J. A. Guzzo. 2017. A Fog Computing-based framework for process monitoring and prognosis in cyber-manufacturing. *J. Manufactur. Syst.* 43, 1 (Apr. 2017), 25–34.

[192] M. H. Yaghmaee, M. Moghaddassian, and A. Leon-Garcia. 2017. Autonomous two-tier cloud-based demand side management approach with microgrid. *IEEE Trans. Industr. Info.* 13, 3 (June 2017), 1109–1120.

[193] Y. Yan and W. Su. 2016. A Fog Computing solution for advanced metering infrastructure. In *Proceedings of the IEEE/PES Transmission and Distribution Conference and Exposition (T&D'16)*. 1–4.

[194] H. Yao, C. Bai, D. Zeng, Q. Liang, and Y. Fan. 2015. Migrate or not ? Exploring virtual machine migration in roadside cloudlet-based vehicular cloud. *Concurr. Comput.: Pract. Exper.* 27, 18 (Dec. 2015), 5780–5792.

[195] D. Ye, M. Wu, S. Tang, and R. Yu. 2016. Scalable Fog Computing with service offloading in bus networks. In *Proceedings of the 3rd IEEE International Conference on Cyber Security and Cloud Computing (CSCloud'16)*. 247–251.

[196] E. Yigitoglu, M. Mohamed, L. Liu, and H. Ludwig. 2017. Foggy: A framework for continuous automated IoT application deployment in Fog Computing. In *Proceedings of the 6th IEEE International Conference on AI and Mobile Services (AIMS'17)*. 38–45.

[197] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang. 2017. A survey on the edge computing for the Internet of Things. *IEEE Access* 6 (Nov. 2017), 6900–6919.

[198] J. K. Zao, T. Gan, C. You, C. Chung, Y. Wang, S. Rodrìguez Mèndez, T. Mullen, C. Yu, C. Kothe, C. Hsiao, S. Chu, C. Shieh, and T. Jung. 2014. Pervasive brain monitoring and data sharing based on multi-tier distributed computing and linked data technology. *Front. Hum. Neurosci.* 8, 370 (June 2014).

[199] W. Zhang, Y. Hu, Y. Zhang, and D. Raychaudhuri. 2016. SEGUE: Quality of service aware edge cloud service migration. In *Proceedings of the International Conference on Cloud Computing Technology and Science (CloudCom'16)*. 344–351.

[200] J. Zhou, Z. Cao, X. Dong, and A. V. Vasilakos. 2017. Security and privacy for cloud-based IoT: Challenges. *IEEE Commun. Mag.* 55, 1 (Jan. 2017), 26–33.

[201] J. Zhu, D. S. Chan, M. S. Prabhu, P. Natarajan, H. Hu, and F. Bonomi. 2013. Improving web sites performance using edge servers in Fog Computing architecture. In *Proceedings of the 7th IEEE International Symposium on Service-Oriented System Engineering (SOSE'13)*. 320–323.

Received January 2018; revised October 2018; accepted November 2018