Computing (2021) 103:51–73
https://doi.org/10.1007/s00607-020-00848-z

REGULAR PAPER

Notions of architecture in fog computing

Zoltán Ádám Mann[1]

Received: 4 March 2020 / Accepted: 1 October 2020 / Published online: 10 October 2020

Abstract
Fog computing is becoming a popular paradigm for bringing the advantages of the cloud nearer to the network edge. This way, computational tasks can be offloaded from end devices to nearby fog nodes, thus benefiting from high computational power and low latency at the same time. Architecture plays a central role in fog computing. Many papers on fog computing address architectural questions. However, a closer look reveals that different papers use the term "architecture" for very different concepts. This is rooted in the multi-disciplinary nature of the fog computing paradigm. The different communities involved in fog computing—network, hardware, system software, application software—all use the term "architecture," but with different meaning. To facilitate the mutual understanding of architectural issues in fog computing, this paper introduces a conceptual framework for reasoning about architecture in fog computing. This conceptual framework uses three independent dimensions to describe architecture. Based on the three architecture dimensions, several architecture views can be defined to serve the different viewpoints of the involved disciplines, and to highlight different aspects of the architecture. The conceptual framework is validated using a literature mapping study.

Keywords Fog computing · Edge computing · Architecture · Software architecture · Hardware architecture · Network architecture

Mathematics Subject Classification 68M01

1 Introduction

In recent years, fog computing (also called edge computing) has emerged as a promising new paradigm for offering computation and storage services in a distributed way [6]. Fog computing uses, in addition to centralized cloud data centers, a large number

✉ Zoltán Ádám Mann
zoltan.mann@gmail.com

[1] paluno – The Ruhr Institute for Software Technology, University of Duisburg-Essen, Essen, Germany

of resources with smaller capacity near the network edge, called fog nodes [28]. Data processing tasks that require real-time processing of data from end devices can be performed by nearby fog nodes, leading to low transmission latency [62]. Because of its advantages, fog computing has been the subject of intensive research [5,32,45].

Fog computing builds on and combines several existing research areas, including cloud computing, the Internet of Things (IoT), distributed cloud computing, mobile cloud computing, content delivery networks, services computing etc. As a result, researchers of very different background (networks, cloud, software, services, security etc.) are working on fog computing.

Architecture plays a central role in fog computing [27]. Several authors regard fog computing as a specific type of architecture, talking about the "fog architecture" or the "fog computing architecture" [15]. Others propose specific architectures for some aspects of fog computing [61].

Although the term "architecture" is used in the context of fog computing frequently and evidently, different authors sometimes mean completely different things by this term. For example, "fog architecture" is often used to denote the way different devices are organized into a network topology, including concepts like IoT devices, fog nodes, and cloud data centers [69]. However, other publications use completely different decomposition principles, leading to architectural elements like, e.g., "connectivity", "user, device and data management" and "application services" [19].

On the one hand, the wide variety of interpretations of the notion of "architecture" is understandable, as it results from the different background of the involved researchers. On the other hand, given the high importance of architecture in fog computing, the chaotic use of the term "architecture" is problematic. Without a clear common understanding of what architecture means, the probability of misunderstanding is high, and it is hard to compare or combine different "architectures" proposed by different researchers.

In this paper, we analyze the problem of different architectural notions in fog computing in detail, also looking at the origins of the problem, namely the different use of architecture in different disciplines within computer science. On the basis of this analysis, the paper makes the following contributions towards a better common understanding of architecture in fog computing:

- We propose a *conceptual framework* for reasoning about architecture in fog computing. This conceptual framework uses three independent *dimensions* to describe architecture: the device dimension, the system dimension, and the functionality dimension. Based on the three dimensions, several *views* can be defined to highlight different aspects of the architecture.
- To illustrate the usage and the benefits of the proposed framework, we use it for a *literature mapping study*. We categorize about 50 papers from the literature according to the architecture dimensions covered by them to demonstrate the prevalence of different architectural views in the literature and to show how the proposed conceptual framework can be used to structure and categorize architectural approaches.

The paper is organized as follows. Section 2 gives a brief introduction to fog computing. Section 3 explains the problem by showing examples for the widely differing

use of architecture in the fog computing literature. Section 4 analyzes the origins of the problem by looking at the use of architecture in computer science in general. Section 5 presents our proposed conceptual framework of architectural concerns in fog computing. Section 6 shows the application of the proposed framework to structuring the literature. Section 7 presents a critical discussion on the strengths and limitations of our approach. Section 8 discusses related work and Sect. 9 concludes the paper.

2 Fog computing

End devices, such as smart phones, sensors, or devices from the Internet of Things, produce a huge amount of data. To create value, data have to be processed, for instance through some data analytics applications. Data processing can be very resource-intensive, and the resources (such as CPU, memory, and battery) of end devices are often limited. This leads to the necessity of offloading computational tasks from end devices to other, more powerful devices [6].

One possibility is to use cloud services for offloading. That is, data are transferred from the end device to an application running in a cloud data center, and this application performs the data processing. Given the practically unlimited resources available in the cloud, this is a good approach for several use cases.

However, in other cases, uploading all data to the cloud can be problematic [38]. One of the key issues is latency: the cloud data center may be many network hops away from the end device, leading to a significant overhead. This overhead can be prohibitive for some application areas, including road traffic control, augmented reality, and gaming, among other domains. Also, if a huge number of end devices upload large amounts of data to the same cloud data center, the network connection of the data center may become a bottleneck, leading to congestion and possibly to a further increase in latency.

To overcome the weaknesses of the cloud-based approach, fog computing introduces a middle layer between the end devices and the cloud [28]. As Fig. 1 shows, this middle layer consists of so-called fog nodes. Fog nodes are computational resources deployed in a geographically distributed way, near the network edge. End devices can offload computational tasks to a nearby fog node just as they would to the cloud; however, using decentralized fog nodes for offloading leads to significantly reduced latency and also mitigates the problem of the cloud becoming a bottleneck. Fog computing can leverage existing network equipment (e.g., routers or base stations) by furnishing them with extra compute capacity so that they can act as fog nodes; alternatively, fog nodes can be deployed specifically for the purpose of serving as fog nodes. Either way, fog nodes can be seen as small data centers near the network edge that serve end devices in their vicinity. Similar to clouds, fog nodes can use virtualization and related technologies (e.g., containers) to run applications belonging to different tenants in an isolated way, also fostering easy application management, including the start-up, automatic scaling, and shut-down of applications.

Figure 1 also shows that some data processing steps can still be offloaded to the cloud, particularly those data processing steps that are resource-intensive and not latency-critical. Therefore, fog computing features three layers: end devices, fog nodes, and clouds. End devices have small capacity but they are present in huge numbers.

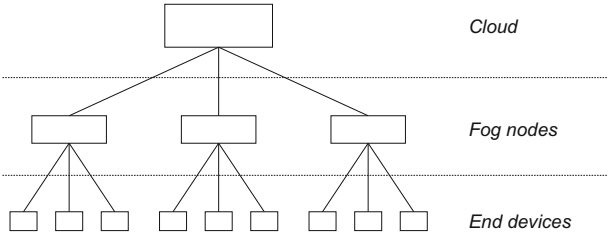


Fig. 1 Abstract model of fog computing

Fog nodes offer higher, but still limited capacity, and they are less numerous. Finally, clouds offer large capacity, while their number is low. From the end devices, fog nodes are just 1–2 hops away and can thus be reached with small latency, whereas data transfer to the cloud is associated with a much higher latency. By combining these three layers, fog computing can effectively provide latency-critical applications with the necessary processing resources [41].

It should be noted that several authors use different terminology—e.g., edge computing, mobile edge computing, or multi-access edge computing—to describe very similar concepts [40]. In this paper the term "fog computing" is used in an abstract way for describing any computing paradigm, in which computational tasks can be offloaded from end devices to nearby computing resources or potentially to the cloud. Thus, our notion of fog computing also encompasses edge computing and other related paradigms.

3 Architecture in fog computing

In recent years, a large number of publications have dealt with fog computing, from a variety of viewpoints. One aspect that often plays a fundamental role in these publications is architecture.

In most papers, when authors talk about "the fog computing architecture", they mean something like the structure shown in Fig. 1; examples include [7,11,25,55,61,63,71,72]. There are some small variations: e.g., some authors also allow communication among end devices [63] or among fog nodes [11], Sharma et al. differentiate between "edge nodes" and "fog nodes" [55], while Zhou et al. [72] extend the three layers of Fig. 1 with corresponding data handling functionalities.

There are some papers that include both a diagram similar to our Fig. 1 and another, different architectural diagram. For example, Tiburski et al. present a second architectural diagram, consisting of the following three layers: hardware, hypervisor, and virtual machines [61]. Another example is the paper of Sharma et al., in which two additional diagrams are presented, one on the architecture of fog nodes and one on the architecture of edge nodes [55]. The fog node architecture consists of the following components: service management, context awareness, attack mitigation, and distributed database. The edge node architecture consists of the following components:

switch information, cache management, network resources, channel monitoring, radio resource, and traffic monitoring.

Somewhat similarly, Alam et al. define for each of the three fog layers (end device, fog node, cloud) a set of components [3]. The components in end devices are: collaborative and autonomous decision-making, micro analytics, self-healing, sensing input, mesh computing, display output, and local caching. The components in fog nodes are: data transformation, network function virtualization, cognitive computing, local systems coordinator, distributed analytics, and distributed data storage. Finally, the components in the cloud are: integration, orchestration, management, monitoring, automation, visualization, stateful services, long-term analytics, and storage. In addition, "mediation" components are foreseen in each layer, as well as "messaging hub" components between the layers.

Chen et al. [8] present multiple different architectures. Their architectural overview diagram, consisting of four "domains" ("device domain", "network domain", "data domain", "application domain"), has some similarity with our Fig. 1, because the device domain consists of the end devices, and the network and data domains together cover the fog nodes. In addition, also an architecture of "edge-enhanced data fusion" is provided. The latter is a combination of hardware devices (sensors, actuators, programmable boards), data fusion functions (data acquisition, semantic interoperability, algorithms, fault detection), and application features (process visibility, fault prediction, digital twins).

The FA^{2}ST approach of Chen et al. [9] uses a different three-layer architecture. Here, the three layers are: infrastructure, platform, and software. In contrast, the Fog of Things approach of Yu et al. [68] is based on an architecture with four planes. Here, the planes are: data plane, virtualization plane, control plane, and application / management plane.

As can be seen from the above examples, there can be big differences in what authors view as architecture in fog computing. It is not only that the architectures are made of different components—this would not be a surprise, since different systems can have different architectures even in the same computing paradigm—but some architectures use completely different decomposition principles than others. Overall, we can conclude that the term "architecture" can have a variety of different meanings in the context of fog computing, emphasizing different aspects of the proposed fog computing approaches. This makes it difficult to compare different fog architectures, to determine the most appropriate architecture for a given environment, or even to unambiguously describe the architecture of a fog computing system.

4 Architecture in different disciplines

The somewhat chaotic combination of different notions of architecture in fog computing is rooted in the fact that the term "architecture" is used in different branches of the Information and Communication Technology (ICT) field, with similar but slightly different meaning, connotations, vocabulary, and formalisms. These differences normally cause no problem because within a specific community a mostly consistent set of notions gets established. However, fog computing combines several disciplines—

network, hardware, software—and this leads to a clash in terminology. Therefore, to obtain a clearer picture of architecture in fog computing, it is important to understand what architecture means in these disciplines.

For a general definition of architecture, we can refer to the ISO/IEC/IEEE 42010:2011 standard. This standard explicitly differentiates between "architecture" as an abstract concept and "architecture description" as a specific artefact that expresses an architecture. The standard defines architecture as follows: "fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution."

This generic definition applies to any kind of system. Regarding architecture in different disciplines, based on the seminal paper of Perry and Wolf [51] the following observations can be made:

- *Building architecture* is the source of the analogy underlying the notion of architecture in ICT. Building architecture has several important properties that may also have analogies in ICT: the connection with engineering principles and properties of materials (e.g., an architectural style requires certain engineering practices and materials with certain properties), the usage of different views to highlight different aspects and foster different levels of details, and the connection to aesthetics.
- *Hardware architecture* describes the configuration of the pieces making up computer hardware and the principles of their interoperation (e.g., showing the components and operation of a processor). Similarly to building architecture, also hardware architecture is constrained by the laws of physics. Compared to software, hardware architecture is characterized by a relatively small number of design elements, which may be replicated many times, thus achieving large scale and complexity.
- *Network architecture* abstracts the elements of a network into nodes and links. The architecture is mainly characterized by the graph structure, or topology, of the nodes and links. In addition, some characteristics of the nodes and links (e.g., bandwidth) may also be of interest.
- *Software architecture* is perhaps the youngest but also most multi-faceted of these notions. In contrast to buildings and hardware, software is not directly constrained by the laws of physics. Also, the flexibility of software leads to an unbounded number of possible design elements, and thus to arbitrarily complex software systems. Software architecture plays a vital role in taming the complexity of software systems.

There are some general notions that play an important role in the architecture of ICT systems. An architecture usually defines *components* that together make up the system—irrespective of whether the components are hardware, software, or a combination of the two. The components are usually connected by some *relationships* (which may be called links, connectors etc.). Typically, the relationships among the components are not allowed to be arbitrary, but must follow a given *architectural style*, which defines what relationships are possible between which pairs of components.

A popular architectural style—used in multiple branches of ICT—is the *layered architecture*. In a layered architecture, the components are organized into a sequence

of layers (also called levels, tiers, planes etc.) on top of one another. Components in a given layer can only rely on and use services of the components in the layer below, while they provide services to the layer above. An example for a layered architecture is the ISO Open Systems Interconnection (OSI) model for telecommunication [73].

A layered architecture may also be accompanied by an *increasing level of abstraction* along the layers. That is, lower layers may abstract away some details for the higher layers, so that the higher layers do not have to bother with these details. For example, in the ISO OSI model, the lowest layer deals with the physical interconnection medium, and it offers the service of the transmission of bits to higher layers. Higher layers can use this service without having to worry about physical details (e.g., voltages or pin layout).

A layered architecture is not a guarantee for an increasing level of abstraction along the layers. For example, many web applications use a three-tier architecture, in which the lowest tier is responsible for data handling, the middle tier for the business logic, and the upper tier for the user interface. This is indeed a layered architecture, since the business logic layer only uses services of the data handling layer, and the user interface layer only uses services of the business logic layer [47]. However, some cross-cutting concerns, e.g. performance optimizations, may make it necessary that the business logic be aware of details of the data handling layer. Thus, the data handling layer cannot fully abstract away the details of data handling.

A further important concept is constituted by *architecture views*. The architecture of complex systems cannot be captured by a single architectural diagram [10]. Attempts to capture the architecture of a complex system with a single architectural diagram usually lead to at least one of the following undesirable effects: (1) important aspects are missing, (2) the diagram becomes prohibitively complicated, (3) the semantics of the diagram is ambiguous (e.g., it is unclear what boxes and arrows exactly represent). To overcome this difficulty, Kruchten suggested to use multiple views to represent different aspects of the architecture of a software system [31]. Specifically, his approach considered four views for describing software architecture:

- *Logical view* decomposition of the software into logical components, which are connected by relationships.
- *Process view* representation as a network of communicating processes, focusing on non-functional properties like performance and reliability, addressing concurrency and synchronization issues.
- *Physical view* mapping of the software onto the available hardware units, thus also encoding distribution.
- *Development view* decomposition of the software in its development environment for supporting the development efforts.

Concepts like architectural styles, layered architectures, and architectural views can be used to describe architectures in fog computing as well. However, what is missing is a conceptual framework that connects and structures the different kinds of architectures used in fog computing. In the next section, we propose such a framework.

5 An architectural framework for fog computing

A fog computing system comprises hardware, software, and network elements. In particular, through recent trends like software defined networking (SDN) and network function virtualization (NFV), software is playing an increasingly important role also in classically hardware and networking dominated domains [16,64]. As a result, software plays multiple roles in fog computing. According to the service layers of cloud computing (Infrastructure-as-a-Service, Platform-as-a-Service, Software-as-a-Service), the following differentiation can be made:

- The infrastructure layer comprises both hardware (physical infrastructure) and software (virtualized infrastructure).
- The platform layer and the application layer are typically implemented by software.

For this reason, a simple differentiation into hardware architecture and software architecture is not sufficient for fog computing.

Inspired by the different architectural aspects described in the fog computing literature, the different architectural concepts of the involved communities, and the idea of architecture views, we now propose a conceptual framework for reasoning about architecture in fog computing. This framework uses *three architecture dimensions*, based on which several *architecture views* can be defined to serve the different viewpoints of the involved disciplines, and to highlight different aspects of the architecture.

5.1 Three dimensions of the fog architecture

To structure the entangled aspects of the fog computing architecture, we propose to differentiate three dimensions of the fog architecture, as follows.

- The *device dimension* focuses on the different types of nodes in the network and the links between them. In fog computing, this means in particular the differentiation between end devices, fog nodes, and the cloud, as well as the communication links between them, as depicted e.g. in Fig. 1. Also questions related to different types of fog nodes (e.g., whether edge nodes should be differentiated from other fog nodes) or whether direct communication between fog nodes is supported belong to this dimension.
- The *system dimension* describes the technology stack that makes up fog computing systems. As with many ICT systems, fog computing systems comprise hardware and different layers of software, including virtualization, operating system, middleware, and applications. The system dimension emphasizes the various layers of the technology stack that must work together to enable the operation of fog computing systems. Also the service layers (Infrastructure-as-a-Service, Platform-as-a-Service, Software-as-a-Service) can be mapped on the system dimension.
- The *functionality dimension* decomposes the functionality of fog computing systems into functional building blocks. The functional building blocks may include components with different management and orchestration functionalities (e.g., resource management, service management, data management), but also domain-specific or application-specific functionalities. Functional building blocks may also offer specific services (e.g., data fusion or security primitives).

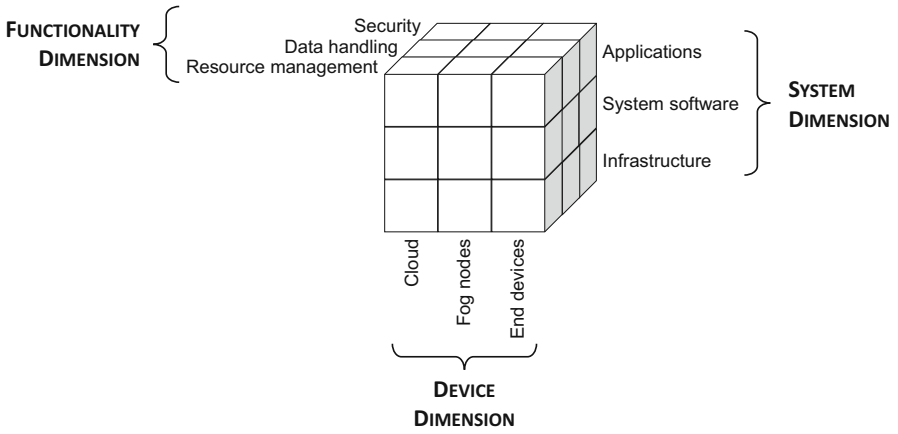


Fig. 2 Three dimensions of the fog architecture, with some example elements for each dimension

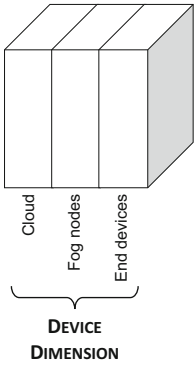
As depicted in Fig. 2, the three dimensions are orthogonal to (i.e., independent from) each other. The device and system dimensions are orthogonal to each other, since each device has its own technology stack and each layer in the technology stack may be present in any of the devices. The device and functionality dimensions are also orthogonal to each other, since each device hosts its own set of functional building blocks and each functional building block may be deployed in any device. And the system and functionality dimensions are also orthogonal to each other, since each functionality can be implemented by various elements of the technology stack and each element of the technology stack may implement various functional building blocks.

Figure 2 also contains three example elements for each dimension (e.g., security, data handling, and resource management for the functionality dimension). These are indeed just examples, i.e., in a particular fog computing system, the specific elements can be different (also their number can be different). However, the presented dimensions—device, system, and functionality—play an important role in all fog computing systems.

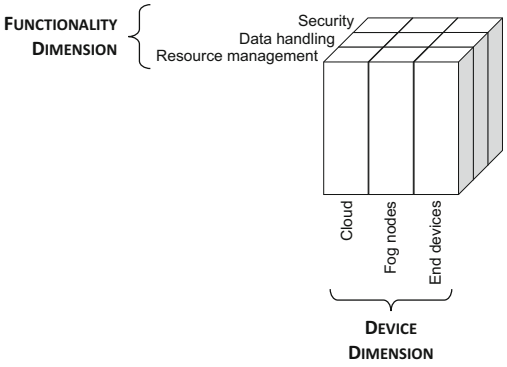
In each dimension, the architecture may follow an established architectural style, independently from the other dimensions. For example, each dimension may or may not follow the layered architecture style, independently from the other dimensions. For the device and system dimensions, layered architectures are quite typical. However, even if both the device and system dimensions use a layered architecture, the two architectures are different since the semantics of the layers is different in the device and the system dimensions.

5.2 Architecture views

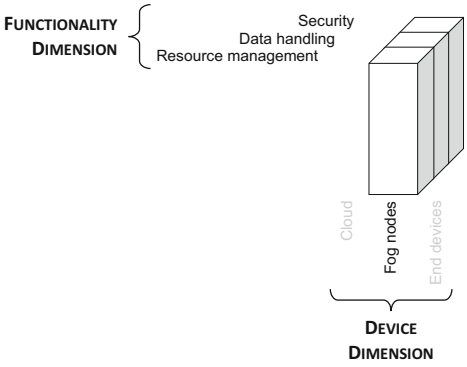
On the basis of the presented dimensions, several different views on the fog architecture can be defined. This explains the large variety of fundamentally different architecture descriptions (and architecture diagrams) used in the literature. Some example views are shown in Fig. 3.



(a) View corresponding to the device dimension



(b) View combining the device and functionality dimensions



(c) View combining one component of the device dimension with the functionality dimension

Fig. 3 Examples for architecture views

The simplest architecture views consist of exactly one of the three architecture dimensions presented above. Hence, there is a device view on the fog architecture, there is a system view, and a functionality view. For example, our Fig. 1 is a device view on the fog architecture, while Cui et al. use also a device view, although slightly different [11]. On the other hand, the FA^{2}ST approach of Chen et al. focuses on a system view [9], and Gedawy et al. present a functional view [21]. Figure 3a shows schematically how for example the device view fits into the proposed framework.

An architecture view can also combine two architecture dimensions. For example, Alam et al. combine the device and functionality dimensions by defining for each of the three device layers (end device, fog node, cloud) a set of functional building blocks [3]. Figure 3b shows schematically how such a view fits into the proposed framework.

It is also possible to combine one dimension with a single element of another dimension. An example can be found in the work of Sharma et al., specifying an architecture view for fog nodes using a decomposition into functional building blocks [55]. Hence, this example combines the functional dimension with a single element

(fog node) of the device dimension. Figure 3c shows schematically how such a view fits into the proposed framework.

6 Applying the framework

We have used the proposed architectural framework to categorize the architectures described in the fog computing literature. To obtain a representative sample of the relevant literature, we selected about 50 papers from several different journals (Future Generation Computer Systems, IEEE Communications Magazine, Journal of Parallel and Distributed Computing) and conferences (IEEE International Conference on Edge Computing, IEEE International Conference on Fog and Edge Computing, International Conference on Service-Oriented Computing, ACM Symposium on Applied Computing) that contain the description of an architecture in the context of fog computing, and mapped these architectures on the dimensions introduced in Sect. 5. Including such a varied set of sources in our study allows the accommodation of the views of the different communities involved in fog computing research. For example, papers in IEEE Communications Magazine have a focus on networking and communication, whereas the International Conference on Service-Oriented Computing represents the point of view of the services community. On the other hand, Future Generation Computer Systems and the ACM Symposium on Applied Computing cover a heterogeneous set of communities by themselves.

The results of this mapping study are summarized in Tables 1, 2, 3 and 4 in the Appendix. If a paper describes multiple architectures, these are shown in multiple corresponding rows in the table. For example, the first two rows of Table 1 after the heading show that paper [2] describes two architectures: first a device architecture consisting of 4 layers, followed by a functional decomposition of one of the device layers into 5 components.

Based on the mapping study, we can identify the following lessons learned:

- Each paper could successfully be categorized based on our architectural framework. Moreover, each of the three dimensions of the framework was used. This suggests that the framework is indeed useful for categorizing work on architecture in fog computing.
- As shown in Fig. 4, the device dimension is the most frequently used architectural dimension. 39 papers contained information about the device dimension, followed by the functionality dimension with 23 papers. This is in line with the high impact of the specifics of the device architecture on fog computing. The system dimension was used in only 12 papers, which may be attributed to the fact that fog computing is not specific in its system dimension: it uses a similar system stack as most other virtualized computer systems.
- Several papers contain multiple architectures. A frequent pattern (used in 15 papers) is that the paper first describes a device architecture, followed by some other architectures (in most cases, using the functionality dimension) for some layers of the device architecture. Other papers simply contain multiple different

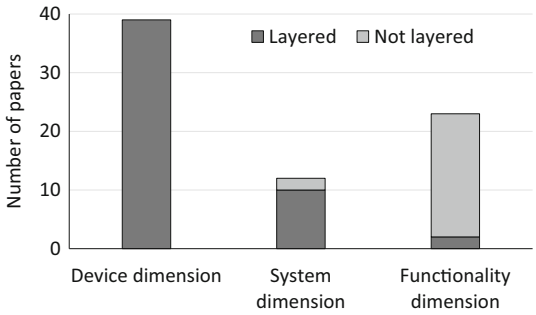


Fig. 4 Frequency of the architecture dimensions in the investigated papers, also differentiating between layered and non-layered architectures

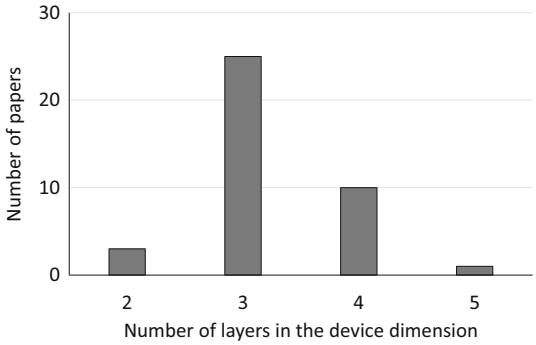


Fig. 5 Histogram of the number of layers in the device dimension

architectures. All these cases can be seen as showing different architectural views, as presented schematically in Fig. 3.

- As shown in Fig. 4, the device and system dimensions use almost exclusively the layered architectural style. The functional dimension is in most cases not layered, nor does it follow another common architectural style.
- As shown in Fig. 5, the device architectures consist in most cases of 3 or 4 layers. Although there is large variety in the naming of these layers, the differences in meaning are rather small. All these architectures can be seen as variations of our Fig. 1. Deviations from Fig. 1 include the absence of a layer (e.g., [33] does not include the cloud), or splitting a layer into two (e.g., [23] differentiates between a layer of sensors / actuators and a layer of devices).
- In contrast, the functionality dimension shows large variance, not only in the number of components and their names but also regarding their semantics. This is plausible since there is a wide array of possible applications of fog computing, and also many ways to decompose the functionality of an application into components.
- Some papers combine multiple architecture views into a single architecture diagram. For example, [67] combines several architecture views corresponding to different combinations of all three architecture dimensions into a single diagram. This leads to highly complex diagrams that are hard to understand. Using separate diagrams for different architecture views makes it easier to cope with the complexity of a larger architecture.

A further general experience with architectures in the fog computing literature is that architectures are described in most cases with natural-language text and using ad-hoc diagrams. Architecture description languages or other modeling languages with well-defined syntax and semantics are hardly used.

7 Discussion

The aim of this section is to provide a critical discussion of the strengths and limitations of the contributions of this paper. In particular, we discuss the simplicity of the proposed conceptual model, the extent to which the conceptual model is specific to fog computing, and the potential benefits and applications of our approach.

7.1 Potential limitations

The proposed conceptual model, consisting of only three simple dimensions, could be perceived as simplistic or obvious. However, there are two reasons why we do not see this as a problem.

The first reason is *usability*. A conceptual model must be simple to be useful. A model abstracts from all unessential details to direct the focus on some essential characteristics. If a model contains too many details, then the model may not fulfill its goal of supporting abstraction. In addition, if a model is too detailed, this may overly constrain its applicability. It is no surpise that many successful models in ICT (for example, the already mentioned ISO OSI reference model [73], the MapReduce programming model for distributed data processing [14] or the MAPE-K model of autonomic systems [29]) are rather simple.

The second reason is related to *expressivity* and *extensibility*. While our conceptual model only consists of three dimensions and each dimension on its own may be quite simple, the three dimensions can be arbitrarily combined. This leads to a large number of possible architectural views, some of which were shown in Sect. 5.2. Using different combinations of the three dimensions, quite complicated architectures can be described and structured, making the conceptual model rather powerful.

Another concern is *specificity*. It could be questioned to what extent the proposed conceptual model is specific to fog computing. Indeed, the three dimensions of the conceptual model are quite generic and could be applied to any ICT system.

The specificity of the conceptual model to fog computing arises through the concrete elements along the dimensions, particularly along the device dimension. As shown in Sect. 6, most papers on fog computing emphasize a similar set of layers along the device dimension (e.g., cloud, fog nodes, end devices). This structure along the device dimension seems to be the common denominator across the investigated papers on fog computing. This finding is in line with the experience that the fog computing literature covers an exceptionally wide-ranging set of topics, where the main commonality is the structure according to the cloud, fog, and end device layers.

7.2 Benefits and applicability

The contribution of this paper is intrinsically intangible, since the paper aims at a better understanding of the different notions of architecture in fog computing. Nevertheless, the insights of the paper can be applied in several important practical scenarios, such as the following ones:

- When *evaluating* the suitability of an architecture for a fog computing system, the conceptual framework proposed in this paper can be used to determine which dimensions the given architecture addresses and to what extent, and which ones it does not address. This way, our conceptual framework can help to identify gaps in architecture descriptions that would otherwise be hard to detect.
- When *developing* an architecture for a fog computing system, the conceptual framework proposed in this paper can be used as a checklist to ensure that the architecture covers all important aspects. In addition, the results of Sect. 6 can be used to ensure that the architecture corresponds to the state of the art in each considered dimension.
- When *comparing* two architectures for fog computing systems, the conceptual framework proposed in this paper can be used to decide to which extent the two architectures are comparable. For example, we can identify situations in which no meaningful comparison is possible since the two architectures address different architectural dimensions.

Thus, the contributions of this paper can be leveraged to make better architectural decisions, which will lead to better fog computing systems.

8 Related work

Although architecture plays an important role in many publications on fog computing (as can also be seen in Sect. 6), to the best of our knowledge we are the first to discuss the different notions of architecture in fog computing and to provide a general framework for conceptualizing different architectures. In the following we review related efforts on architecture in general, reference architectures for fog computing and on the comparison of different architectures.

As already mentioned in Sect. 4, the study of architecture in general has a long tradition. In particular, researchers on software architecture investigated comprehensively what architecture means in different disciplines and how software architecture differs from other types of architecture [31,51,56]. According to Taylor et al., software architecture is the centerpiece of development, encompassing the principal design decisions [60]. A recurring topic in software architecture is that architecture is more than just a boxes-and-arrows diagram. Indeed, the architecture of a complex system must define several different types of decompositions. Kruchten uses different views for this purpose [31]. Bass et al. [4] talk about different *structures* of the system, for example corresponding to structuring the system as a set of code units or as a set of run-time entities and their run-time interactions. Our work uses a set of dimensions and views for

this purpose. In contrast to previous work on software architecture, our aim is to cover the aspects relevant for fog computing, and these aspects are not limited to software.

The OpenFog Consortium published a reference architecture (RA) for fog computing in 2017 to help create and maintain fog computing systems [48], which was later adopted as standard IEEE 1934–2018 by the IEEE Standards Association.[1] The OpenFog RA codifies relevant best practices—for example, it contains a long list of the types of security techniques that should be applied to mitigate common security concerns in fog computing systems. In contrast, our work does not aim at prescribing the use of specific techniques, but rather to provide a conceptual framework in which many different architectures for fog computing can be formulated, studied, and compared to each other.

The ETSI (European Telecommunications Standards Institute) Industry Specification Group on Multi-access Edge Computing also defined a reference architecture for multi-access edge computing (MEC) systems [17]. The ETSI MEC RA is much more concrete than the OpenFog RA, and it describes one specific architecture. In contrast to our work, it does not aim at providing a framework for describing different architectures.

Mach and Becvar describe and compare several fog computing approaches, also including architectural aspects [37]. However, their comparison is not focused on different architectural notions, but rather on the device dimension of the architectures and different further technical aspects. In contrast, our approach aims at reasoning about different architectural notions.

9 Conclusion

This paper has highlighted the large variety of fundamentally different architectures used in publications on fog computing. We have argued that this is a consequence of the different architectural notions used in the different communities that play an important role in fog computing, including networks, hardware, virtualization, system software, and applications. We have proposed a conceptual framework for reasoning about architecture in fog computing, consisting of three orthogonal dimensions: device, system, and functionality. On the basis of these three architecture dimensions, several different architecture views can be defined, and the notions of architecture used in different publications can be categorized. This way, our work can contribute to a better understanding of the different facets of architecture in fog computing, which is of central importance to the future development of this field.

[1] https://standards.ieee.org/standard/1934-2018.html.

Acknowledgements This work has been supported by the European Union's Horizon 2020 research and innovation program under Grants 731678 (RestAssured) and 871525 (FogProtect).

Funding Open Access funding enabled and organized by Projekt DEAL.

Appendix

This appendix contains the results of the literature mapping study described in Sect. 6 in the form of Tables 1, 2, 3 and 4.

Table 1 Architecture dimensions used in the fog literature, part 1

	Device dimension	System dimension	Functionality dimension
[2]	4 layers: mobile stations, mobile fog, evolved packet core, public cloud		
	1 layer: mobile stations		5 components: device manager, network manager, application manager, execution analyzer, compiler
[19]	3 layers: devices, fog, cloud		
	1 layer: fog	3 layers: hardware, operating system, software	
	1 layer: cloud		3 layers: connectivity, user & device & data management, application services
[20]	4 layers: IoT, mobile clouds, edge nodes, remote clouds		
[22]	3 layers: sensor, fog, cloud		
	1 layer: fog		6 components: fall detection, QT interval extraction, interoperability, heart rate, activity status, security
[24]	3 layers: mobile agents, controllers, IoT devices		
	1 layer: IoT device		3 components: communication, core functions, Aura sandbox
[26]		3 layers: infrastructure, resources, data	
	3 layers: devices, network, cloud	1 layer: infrastructure	
[34]	3 layers: nodes, edge, cloud		
[35]	3 layers: things, fog, cloud		
[42]	3 layers: things, fog, cloud		
			3 components: data collection, data transfer, data storage
[43]	4 layers: edge, basic nodes, smart nodes, cloud		
			23 components: monitoring, routing, allocation, security, privacy, profiling, categorization etc.
[44]	3 layers: devices, fog, cloud		
[46]	3 layers: edge, gateways, cloud		
			2 components: intermittent security module, resource-aware security module

Table 2 Architecture dimensions used in the fog literature, part 2

	Device dimension	System dimension	Functionality dimension
[50]	4 layers: sensors, edge, intermediate levels, cloud		
	1 layer: edge		3 components: analytics engine, control plane, synch module
	1 layer: cloud		4 components: control plane, data store, distributed system state, analytics engine
[52]	3 layers: devices, edge/fog, cloud		
	3 layers: sensors/actuators, e-health gateways, back-end		
	1 layer: e-health gateways	4 layers: hardware, operating system, local database, software	
[58]	3 layers: physical, fog, cloud		
	1 layer: fog		2 components: keyword extraction, classification
	1 layer: cloud		2 components: storage, outbreak prevention
[67]	4 layers: smart home, IoT gateways, fog, cloud		
	1 layer: smart home	3 layers: cyber-physical, connectivity, context-aware	
	1 layer: smart home	1 layer: cyber-physical	6 components: metering, vehicle charging, energy management etc.
	1 layer: smart home	1 layer: context-aware	6 components: behavior, prediction, event detection etc.
	2 layers: fog, cloud		4 components: authentication, service registry, requests handler, rules engine
	1 layer: fog		6 components: prediction, visualization, pattern mining etc.
[59]	3 layers: fog 1, fog 2, cloud		
[66]	3 layers: extreme edge, edge, cloud		
[65]		3 layers: hardware, hypervisor, unikernels	2 components: dispatcher, image warehouse
[69]	3 layers: IoT devices, fog, cloud		+1 component: trusted authority

Table 3 Architecture dimensions used in the fog literature, part 3

	Device dimension	System dimension	Functionality dimension
[3]	3 layers: edge, fog, cloud		
	1 layer: edge		7 components: sensing, caching, self-healing etc.
	1 layer: fog		6 components: data transformation, cognitive computing, distributed analytics etc.
	1 layer: cloud		9 components: orchestration, monitoring, visualization etc.
[7]	3 layers: devices, MEC servers, cloud		
[8]		4 components: devices, network, data, application	
[9]		3 layers: infrastructure, platform, software	
[11]	4 layers: devices, access points, edge, cloud		
	1 layer: edge	3 components: edge server handler, data repository, VM sets	
[25]	3 layers: IoT, edge gateways, cloud		
[55]	4 layers: data producer, edge, fog, cloud		
	1 layer: fog		4 components: context awareness, service management, distributed database, attack mitigation
	1 layer: edge		6 components: radio resources, traffic monitoring, channel monitoring etc.
[61]	3 layers: edge, fog, cloud		
	1 layer: edge	3 layers: hardware, hypervisor, VMs	
[63]	4 layers: user devices, MEC servers, base stations, cloud		
[68]		4 layers: data, virtualization, control, application	
		1 layer: control	5 components: service provisioning, dynamics handling, system monitoring, resource optimization, energy management
[71]	3 layers: access cloud, edge cloud, core cloud		
[72]	2 layers: edge, cloud		
			4 layers: data source, data collection, data storage, data analysis
[13]			3 components: frontend, manager, worker node

Table 4 Architecture dimensions used in the fog literature, part 4

	Device dimension	System dimension	Functionality dimension
[18]	3 layers: users, edge, cloud		
[21]	1 layer: edge femtocloud		2 components: controller, worker
[23]	5 layers: sensors/actuators, devices, gateways, internet, cloud		
[30]		6 layers: edge servers, connectivity, EdgeStore, file system, I/O, applications	
		1 layer: EdgeStore	5 components: request placement, resource monitor, metadata manager etc.
[36]	3 layers: user device, edge, data center		
[49]	4 layers: field, edge, cloud, environment		
[54]	3 layers: edge, cloud, dashboard		
	1 layer: edge		4 components: data service, monitoring, updater, scheduling
	1 layer: cloud		10 components: object storage, location database, log etc.
	1 layer: dashboard		2 components: occupancy insights, context-aware services
[1]	3 layers: robots, cloudlets, cloud		
	1 layer: cloudlet	2 layers: virtualized resources, resource manager	
[12]	2 layers: edge, cloud		
[33]	2 layers: users, edge servers		
[53]			4 components: resource directory, device service, platform master, platform service
[70]	3 layers: things, fog, cloud		
[39]	4 layers: IoT, fog gateways, fog compute, cloud		
	1 layer: fog gateway		4 components: application initiation, expectation rating, application placement, data container
	1 layer: fog compute		3 components: communication, computation, controller
[57]	3 layers: sensors/actuators, fog colonies, cloud		

References

1. Afrin M, Jin J, Rahman A (2018) Energy-delay co-optimization of resource allocation for robotic services in cloudlet infrastructure. In: International conference on service-oriented computing. Springer, pp 295–303
2. Alam MGR, Hassan MM, Uddin MZ, Almogren A, Fortino G (2019) Autonomic computation offloading in mobile edge for IoT applications. Future Gener Comput Syst 90:149–157
3. Alam M, Rufino J, Ferreira J, Ahmed SH, Shah N, Chen Y (2018) Orchestration of microservices for IoT using Docker and edge computing. IEEE Commun Mag 56(9):118–123
4. Bass L, Clements P, Kazman R (2003) Software architecture in practice. Addison-Wesley Professional, Boston
5. Bellendorf J, Mann ZÁ (2020) Classification of optimization problems in fog computing. Future Gener Comput Syst 107:158–176
6. Bonomi F, Milito R, Zhu J, Addepalli S (2012) Fog computing and its role in the internet of things. In: Proceedings of the first edition of the MCC workshop on Mobile cloud computing. ACM, pp 13–16
7. Cao B, Zhang L, Li Y, Feng D, Cao W (2019) Intelligent offloading in multi-access edge computing: a state-of-the-art review and framework. IEEE Commun Mag 57(3):56–62
8. Chen B, Wan J, Celesti A, Li D, Abbas H, Zhang Q (2018) Edge computing in IoT-based manufacturing. IEEE Commun Mag 56(9):103–109
9. Chen N, Yang Y, Zhang T, Zhou MT, Luo X, Zao JK (2018) Fog as a service technology. IEEE Commun Mag 56(11):95–101
10. Clements P, Garlan D, Bass L, Stafford J, Nord R, Ivers J, Little R (2002) Documenting software architectures: views and beyond. Pearson Education, London
11. Cui Q, Gong Z, Ni W, Hou Y, Chen X, Tao X, Zhang P (2019) Stochastic online learning for mobile edge computing: learning from changes. IEEE Commun Mag 57(3):63–69
12. da Silva Veith A, de Assunçao MD, Lefevre L (2018) Latency-aware placement of data stream analytics on edge computing. In: International conference on service-oriented computing. Springer, pp 215–229
13. Das RB, Di Bernardo G, Bal H (2018) Large scale stream analytics using a resource-constrained edge. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 135–139
14. Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113
15. Dolui K, Datta SK (2017) Comparison of edge computing implementations: fog computing, cloudlet and mobile edge computing. In: 2017 Global Internet of Things summit (GIoTS). IEEE, pp 1–6
16. Dräxler S, Karl H, Mann ZÁ (2017) Joint optimization of scaling and placement of virtual network services. In: 17th IEEE/ACM international symposium on cluster, cloud and grid computing (CCGrid). IEEE, pp 365–370
17. ETSI (2019) Multi-access edge computing (MEC): framework and reference architecture. ETSI GS MEC 003, V2.1.1
18. Fan K, Pan Q, Wang J, Liu T, Li H, Yang Y (2018) Cross-domain based data sharing scheme in cooperative edge computing. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 87–92
19. Farahani B, Firouzi F, Chang V, Badaroglu M, Constant N, Mankodiya K (2018) Towards fog-driven IoT eHealth: promises and challenges of IoT in medicine and healthcare. Future Gener Comput Syst 78:659–676
20. Farris I, Militano L, Nitti M, Atzori L, Iera A (2017) MIFaaS: a mobile-IoT-federation-as-a-service model for dynamic cooperation of IoT cloud providers. Future Gener Comput Syst 70:126–137
21. Gedawy H, Habak K, Harras K, Hamdi M (2018) An energy-aware IoT femtocloud system. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 58–65
22. Gia TN, Dhaou IB, Ali M, Rahmani AM, Westerlund T, Liljeberg P, Tenhunen H (2019) Energy efficient fog-assisted IoT system for monitoring diabetic patients with cardiovascular disease. Future Gener Comput Syst 93:198–211
23. Harth N, Anagnostopoulos C (2018) Edge-centric efficient regression analytics. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 93–100
24. Hasan R, Hossain M, Khan R (2018) Aura: an incentive-driven ad-hoc IoT cloud framework for proximal mobile computation offloading. Future Gener Comput Syst 86:821–835
25. Hassan N, Gillani S, Ahmed E, Yaqoob I, Imran M (2018) The role of edge computing in internet of things. IEEE Commun Mag 56(11):110–115

26. Hossain MS, Muhammad G, Amin SU (2018) Improving consumer satisfaction in smart cities using edge computing and caching: a case study of date fruits classification. Future Gener Comput Syst 88:333–341
27. Hu P, Dhelim S, Ning H, Qiu T (2017) Survey on fog computing: architecture, key technologies, applications and open issues. J Netw Comput Appl 98:27–42
28. Iorga M, Feldman L, Barton R, Martin MJ, Goren NS, Mahmoudi C (2018) Fog computing conceptual model. NIST Special Publication 500–325. https://doi.org/10.6028/NIST.SP.500-325
29. Kephart JO, Chess DM (2003) The vision of autonomic computing. Computer 36(1):41–50
30. Khan A, Muhammad A, Kim Y, Park S, Tak B (2018) Edgestore: a single namespace and resource-aware federation file system for edge servers. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 101–108
31. Kruchten PB (1995) The 4+1 view model of architecture. IEEE Softw 12(6):42–50
32. Kunde C, Mann ZÁ (2020) Comparison of simulators for fog computing. In: Proceedings of the 35th annual ACM symposium on applied computing, pp 1792–1795
33. Lai P, He Q, Abdelrazek M, Chen F, Hosking J, Grundy J, Yang Y (2018) Optimal edge user allocation in edge computing with variable sized vector bin packing. In: international conference on service-oriented computing. Springer, pp 230–245
34. Lai Y, Lin H, Yang F, Wang T (2019) Efficient data request answering in vehicular ad-hoc networks based on fog nodes and filters. Future Gener Comput Syst 93:130–142
35. Li W, Santos I, Delicato FC, Pires PF, Pirmez L, Wei W, Song H, Zomaya A, Khan S (2017) System modelling and performance evaluation of a three-tier Cloud of Things. Future Gener Comput Syst 70:104–125
36. Li Y, Wang S (2018) An energy-aware edge server placement algorithm in mobile edge computing. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 66–73
37. Mach P, Becvar Z (2017) Mobile edge computing: a survey on architecture and computation offloading. IEEE Commun Surv Tutor 19(3):1628–1656
38. Mahmud R, Kotagiri R, Buyya R (2018) Fog computing: a taxonomy, survey and future directions. In: Internet of everything. Springer, pp 103–130
39. Mahmud R, Srirama SN, Ramamohanarao K, Buyya R (2019) Quality of Experience (QoE)-aware placement of applications in fog computing environments. J Parallel Distrib Comput 132:190–203
40. Mann ZÁ (2019) Optimization problems in fog and edge computing. In: Fog and edge computing: principles and paradigms. Wiley, pp 103–121
41. Mann ZÁ, Metzger A, Prade J, Seidl R (2019) Optimized application deployment in the fog. In: International conference on service-oriented computing. Springer, pp 283–298
42. Manogaran G, Varatharajan R, Lopez D, Kumar PM, Sundarasekar R, Thota C (2018) A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system. Future Gener Comput Syst 82:375–387
43. Masip-Bruin X, Marin-Tordera E, Jukan A, Ren GJ (2018) Managing resources continuity from the edge to the cloud: architecture and performance. Future Gener Comput Syst 79:777–785
44. Moosavi SR, Gia TN, Nigussie E, Rahmani AM, Virtanen S, Tenhunen H, Isoaho J (2016) End-to-end security scheme for mobility enabled healthcare Internet of Things. Future Gener Comput Syst 64:108–124
45. Mouradian C, Naboulsi D, Yangui S, Glitho RH, Morrow MJ, Polakos PA (2017) A comprehensive survey on fog computing: state-of-the-art and research challenges. IEEE Commun Surv Tutor 20(1):416–464
46. Mukherjee B, Wang S, Lu W, Neupane RL, Dunn D, Ren Y, Su Q, Calyam P (2018) Flexible IoT security middleware for end-to-end cloud-fog communication. Future Gener Comput Syst 87:688–703
47. Offutt J (2002) Quality attributes of web software applications. IEEE Softw 19(2):25–32
48. OpenFog Consortium Architecture Working Group (2017) OpenFog reference architecture for fog computing
49. Pallasch C, Wein S, Hoffmann N, Obdenbusch M, Buchner T, Waltl J, Brecher C (2018) Edge powered industrial control: concept for combining cloud and automation technologies. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 130–134
50. Pérez JL, Gutierrez-Torre A, Berral JL, Carrera D (2018) A resilient and distributed near real-time traffic forecasting application for fog computing environments. Future Gener Comput Syst 87:198–212
51. Perry DE, Wolf AL (1992) Foundations for the study of software architecture. ACM SIGSOFT Softw Eng Notes 17(4):40–52

52. Rahmani AM, Gia TN, Negash B, Anzanpour A, Azimi I, Jiang M, Liljeberg P (2018) Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: a fog computing approach. Future Gener Comput Syst 78:641–658
53. Ravindra P, Khochare A, Reddy SP, Sharma S, Varshney P, Simmhan Y (2017) ECHO: an adaptive orchestration platform for hybrid dataflows across cloud and edge. In: International conference on service-oriented computing. Springer, pp 395–410
54. Sajja SSK, Prakash APKS, Tripathi R, Dwivedi S, Singhee A, Vermeulen M (2018) Enterprise scale privacy aware occupancy sensing. In: 2018 IEEE international conference on edge computing (EDGE). IEEE, pp 109–116
55. Sharma PK, Rathore S, Jeong YS, Park JH (2018) SoftEdgeNet: SDN based energy-efficient distributed network architecture for edge computing. IEEE Commun Mag 56(12):104–111
56. Shaw M, Garlan D (1996) Software architecture: perspectives on an emerging discipline. Prentice-Hall, Upper Saddle River
57. Skarlat O, Nardelli M, Schulte S, Dustdar S (2017) Towards QoS-aware fog service placement. In: 2017 IEEE 1st international conference on fog and edge computing (ICFEC). IEEE, pp 89–96
58. Sood SK, Mahajan I (2018) Fog-cloud based cyber-physical system for distinguishing, detecting and preventing mosquito borne diseases. Future Gener Comput Syst 88:764–775
59. Souza V, Masip-Bruin X, Marín-Tordera E, Sànchez-López S, Garcia J, Ren GJ, Jukan A, Ferrer AJ (2018) Towards a proper service placement in combined Fog-to-Cloud (F2C) architectures. Future Gener Comput Syst 87:1–15
60. Taylor RN, Medvidovic N, Dashofy E (2009) Software architecture: foundations, theory, and practice. Wiley, New York
61. Tiburski RT, Moratelli CR, Johann SF, Neves MV, de Matos E, Amaral LA, Hessel F (2019) Lightweight security architecture based on embedded virtualization and trust mechanisms for IoT edge devices. IEEE Commun Mag 57(2):67–73
62. Vaquero LM, Rodero-Merino L (2014) Finding your way in the fog: towards a comprehensive definition of fog computing. ACM SIGCOMM Comput Commun Rev 44(5):27–32
63. Wen J, Ren C, Sangaiah AK (2018) Energy-efficient device-to-device edge computing network: an approach offloading both traffic and computation. IEEE Commun Mag 56(9):96–102
64. Wood T, Ramakrishnan K, Hwang J, Liu G, Zhang W (2015) Toward a software-based network: integrating software defined networking and network function virtualization. IEEE Netw 29(3):36–41
65. Wu S, Mei C, Jin H, Wang D (2018) Android unikernel: gearing mobile code offloading towards edge computing. Future Gener Comput Syst 86:694–703
66. Xia Y, Etchevers X, Letondeur L, Coupaye T, Desprez F (2018) Combining hardware nodes and software components ordering-based heuristics for optimizing the placement of distributed IoT applications in the fog. In: Proceedings of the 33rd annual ACM symposium on applied computing. ACM, pp 751–760
67. Yassine A, Singh S, Hossain MS, Muhammad G (2019) IoT big data analytics for smart homes with fog and cloud computing. Future Gener Comput Syst 91:563–573
68. Yu R, Xue G, Kilari VT, Zhang X (2018) The fog of things paradigm: road toward on-demand Internet of Things. IEEE Commun Mag 56(9):48–54
69. Zhang C, Zhu L, Xu C, Sharif K, Du X, Guizani M (2019) LPTD: achieving lightweight and privacy-preserving truth discovery in CIoT. Future Gener Comput Syst 90:175–184
70. Zhang T, Jin J, Yang Y (2018) RA-FSD: A rate-adaptive fog service delivery platform. In: International conference on service-oriented computing. Springer, pp 246–254
71. Zhang Y, Zhang H, Long K, Zheng Q, Xie X (2018) Software-defined and fog-computing-based next generation vehicular networks. IEEE Commun Mag 56(9):34–41
72. Zhou Z, Yu H, Xu C, Chang Z, Mumtaz S, Rodriguez J (2018) BEGIN: big data enabled energy-efficient vehicular edge computing. IEEE Commun Mag 56(12):82–89
73. Zimmermann H (1980) OSI reference model—the ISO model of architecture for open systems interconnection. IEEE Trans Commun 28(4):425–432