DOI: 10.1504/ijahuc.2021.113384

Monitoring in Fog Computing: State-of-the-Art and Research Challenges

Haftay Gebreslasie Abreha*
IMDEA Networks Institute
Madrid, Spain
email: haftay.abreha@imdea.org
*Corresponding author

Carlos J. Bernardos
Department of Telematics Engineering
University Carlos III of Madrid (UC3M)
Madrid, Spain
email: cjbc@it.uc3m.es

Antonio de la Oliva
Department of Telematics Engineering
University Carlos III of Madrid (UC3M)
Madrid, Spain
email: aoliva@it.uc3m.es

Luca Cominardi
Department of Telematics Engineering
University Carlos III of Madrid (UC3M)
Madrid, Spain
email: luca.cominardi@uc3m.es

Arturo Azcorra
IMDEA Networks Institute /
Department of Telematics Engineering
University Carlos III of Madrid (UC3M)
Madrid, Spain
email: arturo.azcorra@imdea.org

***Abstract*—Fog computing has rapidly become a widely accepted computing paradigm to mitigate cloud computing-based infrastructure limitations such as scarcity of bandwidth, large latency, security, and privacy issues. Fog computing resources and applications dynamically vary at run-time, and they are highly distributed, mobile, and appear-disappear rapidly at any time over the internet. Therefore, to ensure the quality of service and experience for end-users, it is necessary to comply with a comprehensive monitoring approach. However, the volatility and dynamism characteristics of fog resources make the monitoring design complex and cumbersome.The aim of this article is therefore threefold: (*i*) to analyze fog computing-based infrastructures and existing monitoring solutions; (*ii*) to highlight the main requirements and challenges based on a taxonomy; and (*iii*) to identify open issues and potential future research directions.**

***Index Terms*—Fog Computing, Edge Computing, Cloud Computing, Resource Monitoring and Management**

Biographical notes: Haftay received his BSc. degree in Electronics and Communication Engineering from Mekelle Institute of Technology, Ethiopia in 2009. After graduation, he was hired by the Aksum University as an assistant lecturer and researcher in the department of Electrical and Computer Engineering. He joined and successfully completed the Graduate Program in Information and Communication Technologies(GP ICT) given jointly by the University of Trento and Scoula Superiore Santa' Anna, Italy in 2012-2015. Moreover, he developed his Master thesis in the German Aerospace Centre (DLR), Munich, Germany. After graduation, he won the Italian Research Education Network, GARR consortium scholarship to pursue his research study at the University of Trento during 2016-2018. Starting March 2018, Haftay has become a member of the IMDEA Networks research team as part of the Networked Systems and Algorithms Group. His research interest is SDN/NFV, Fog/Edge computing and Machine Learning.

Carlos J. Bernardos received a telecommunication engineering degree in 2003 and a Ph.D. in telematics in 2006, both from UC3M, where he has been an associate professor since 2008. His current work focuses on virtualization in heterogeneous wireless networks. He has published over 70 scientific papers in international journals and conferences, and he is an active contributor to the IETF. He has served as a Guest Editor of IEEE Network.

Antonio de lA olivA received his telecommunications engineering degree in 2004 and his Ph.D. in 2008 from the Universidad Carlos III Madrid (UC3M), Spain, where he has been an associate professor since then. He is an active contributor to IEEE 802 where he has served as Vice-Chair of IEEE 802.21b and Technical Editor of IEEE 802.21d. He has also served as a Guest Editor of IEEE Communications Magazine. He has published more than 30 papers on different networking areas.

LUCA COMINARDI (luca.cominardi@imdea.org) received his Bachelors and Masters degrees in computer science at the University of Brescia, Italy. He did an internship and undertook a Masters in telematics engineering at UC3M. Currently he is working at IMDEA Networks Institute and pursuing his Ph.D. at UC3M. His main research interests are SDN, NFV, and integration of the wireless medium into the two former.

Arturo Azcorra received his M. Sc. degree in telecommunications engineering from UPM in 1986 and his Ph.D. in 1989. In 1993, he obtained an M.B.A. with honors from Instituto de Empresa. He has participated in more than 50 research projects. He has coordinated the CONTENT and E-NEXT European Networks of Excellence, and the CARMEN, 5G-Crosshaul, and 5G-TRANSFORMER EU projects. He is the founder and director of the international research center IMDEA Networks.

I. Introduction

Cloud computing has been a commonly used computing paradigm that provides on-demand access to a shared pool of configurable computing resources (i.e., computing, storage, and networks), enabling resources to be rapidly provisioned and released [1]. Furthermore, it provides a flexible pricing model (such as the pay-as-you-go model), and secure applications and services provisioning for both providers and operators, by sharing the computing infrastructure to reduce the maintenance and management complexity of both hardware and software resources. It is understood that Software Defined Networking (SDN) will play a significant role to achieve the simultaneous demands for different requirements such as security, virtualization, manageability, mobility, and agility in the computing environment. SDN is an emerging paradigm that introduces a programmable network by decoupling the control plane from the underlying physical routers and switches [2]. Network Function Virtualisation (NFV) and SDN collectively enhance the flexibility of service offerings and network management in the cloud environment. Moreover, NFV enables mobile operators a degree of freedom to dynamically deploy services in response to the needs of the traffic and customers by decoupling the network functions from the underlying hardware platform [3]. Nowadays, cloud computing is a widely used computing paradigm, and several companies (such as Amazon Web Services[1], Kamatera[2], Microsoft Azure[3], Google Cloud Platform[4], Vmware[5], IBM Cloud[6], etc...) provide computing resources such as storage, database, servers, and networking to customers and operators at reduced costs compared to self-deployed enterprise computing resources. Despite the many advantages of cloud computing, there are still limitations with the adaptation required in different companies and use-cases. Since cloud-based systems are Internet-dependent; they expose potential vulnerabilities for an attack, security, and privacy challenges. Furthermore, the emergence of the Internet of Things (IoT) causes a large number of connected things to generate massive and heterogeneous data that is cumbersome to process and store with the cloud computing paradigm. Furthermore, the explosion of demands of real-time and latency-sensitive applications from things such as sensors, actuators, and smart devices poses additional challenges to the network bandwidth. Thus, cloud computing cannot cope alone with these issues [4]. Multi-Access Edge Computing (MEC) provides an IT service environment with cloud-computing capabilities at the edge of the mobile network, within the Radio Access Network (RAN) and close to mobile subscribers. MEC has standardized in an ETSI Industry Specification Group (ISG), and it aims to reduce latency, ensure highly efficient network operation and service delivery, and offer an improved end-user service. MEC, SDN, and NFV together are key emerging technologies for 5G networks by leveraging programmable approaches, software networking, and the use of IT virtualization technologies extensively within the telecommunications infrastructure, functions, and applications to satisfy the demanding requirements of 5G in terms of expected throughput, latency, scalability, and automation [5].

In 2012, researchers at Cisco proposed a new computing paradigm called fog computing to ensure low latency while supporting high mobility as a complement to the cloud solution. Fog computing extends the cloud services to the edge of the network, placing the computing, storage, control, and networking functions closer to edge devices and end-users to achieve low-latency, enhance mobility, network bandwidth, security, and privacy [6]–[9]. Edge computing is another contemporary computing paradigm which refers to enabling technologies allowing computation at the edge of the network, e.g., on downlink data on behalf of cloud services and uplink data on behalf of IoT services [10]–[12], [16]. Fog computing is also often referred to as edge computing, though edge tends to be limited to a small number of resource layers, and fog encompasses multiple layers.

The fog/edge computing paradigm improves the capabilities and responsibilities of resources at the edge of the network compared to traditional centralized cloud architectures by not only placing services in the proximity of end-users or devices but also using new data transfer protocols to improve the interaction with data center-based services. Fog computing also provides a low latency response time for the application. To keep the required Service Level Agreement (SLA), Quality of Service (QoS), and Quality of Experience (QoE) for the end-users and customers, it is necessary to apply a comprehensive monitoring approach. This article aims to analyze the main characteristics of fog computing-based infrastructures, existing monitoring solutions proposed for such environments, enumerate their shortcomings, and, based on a proposed taxonomy, identify potential research directions in this field.

This article is organized into sections as follows: Section II covers the research objectives and methodology chosen in this work. Section III introduces the fog computing-based infrastructure advantages and challenges. Section IV illustrates the new thematic taxonomy and challenges in fog computing. Section V reviews the state-of-the-art of the existing monitoring solutions and their challenges in adapting them for fog environment based on the identified thematic taxonomy. Furthermore, section VI presents three case studies that would elaborate on the monitoring solution in the fog computing paradigm. We foresee different possible research directions for monitoring solutions in fog computing in section VII. Finally, we conclude our work by identifying potential future activities in section VIII.

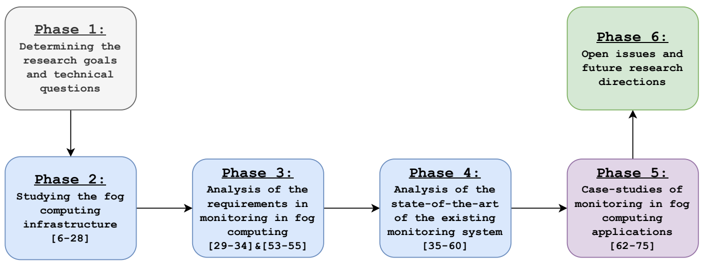
II. Research Methodology

The objective of this section is to clarify the Technical Questions (TQ) that probably found in the monitoring solutions for a fog computing environment and the research methodology adopted in this work. Furthermore, the goal of this paper is to

[1]https://aws.amazon.com/
[2]https://goo.gl/VW87Za
[3]https://azure.microsoft.com/en-in/
[4]https://cloud.google.com/
[5]http://www.vmware.com/in/cloud-services/infrastructure.html
[6]https://www.ibm.com/cloud-computing/

address the following technical questions.
TQ-1: What are the main characteristics of fog computing resources and applications considered?
The goal of this question is to explain the main characteristics and properties of fog computing-based infrastructure. The solution to this question supports the researchers to investigate a monitoring system that satisfies tracking measuring metrics in the fog computing environment.
TQ-2: What are the main requirements considered for resource monitoring in fog computing?
This question intends to identify the relevant monitoring requirements and features that are required by a monitoring solution or tool that enables them to monitor accurately, effectively, and efficiently resources in the fog computing paradigm.
TQ-3: Are there any existing monitoring tools or solutions technically possible solutions for the fog computing paradigm?
The objective of this question is to review the existing solutions (i.e., cloud computing monitoring solutions and tools) and to verify if they are technically fit enough in the fog computing framework.

TQ-4: What are the most reliable case-studies considered in the monitoring solutions in fog computing?
The main objective of this question is to discover the fog monitoring solution in different domains, which are fog computing enabled infrastructures.
TQ-5: What are the future research directions and open issues for the monitoring solutions and tools in the fog computing systems?
The goal of this question is to stipulate the open issues and research directions in the field of fog monitoring approach. Consequently, the answer to this question encourages the researchers to understand the current research results and trends in the area of the fog monitoring solution.
Our research methodology comprises six core phases, as presented in Fig. 1. Moreover, Fig. 1 shows the research methodology phases adopted in our work and the kind of literature used to accomplish the aim of each phase. Phase 1 has already outlined in this section that deals with determining the research goals and technical questions. Besides, fog computing enabled infrastructure has then been studied in phase 2 and described in the third section. Furthermore, the fourth section illustrates the requirements and features of monitoring in fog computing and analyzed in the third phase. Phase 4 describes the existing monitoring solution, as depicted in section V. Furthermore, in phase 5 deals with the case studies in fog computing and depicted in section VI. Finally, the sixth phase details the open issues and future research directions and explains in section VII.


Phase 1:
Determining the research goals and technical questions
Phase 2:
Studying the fog computing infrastructure [6-28]
Phase 3:
Analysis of the requirements in monitoring in fog computing [29-34]&[53-55]
Phase 4:
Analysis of the state-of-the-art of the existing monitoring system [35-60]
Phase 5:
Case-studies of monitoring in fog computing applications [62-75]
Phase 6:
Open issues and future research directions
Fig. 1. Research Methodology

III. FOG COMPUTING BASED INFRASTRUCTURE AND ITS CHALLENGES

Fog computing is a distributed computing paradigm that provides the missing link in the cloud-to-thing continuum. It gives computing, storage, control, and networking functions closer to the end devices/users with seamless integration of the existing cloud infrastructure [16] to solve the bandwidth and latency limitations of the cloud-only computing paradigm. Fog nodes reside on the local access network, closer to IoT and end-user devices. These devices can directly access and consume resources from the fog, enhancing the security [14], and providing localization and mobility support [15].

Furthermore, fog based infrastructures include a large number of geographically dispersed heterogeneous nodes, which could be switches, routers, computers, and even cell phones. And thus, fog computing enables users to upload or download data to/from both data centers in the cloud and nearest fog nodes. Similarly, it allows users to receive or send data from/to neighboring nodes, and other users using Device-to-Device communication (D2D) [13], where devices can communicate peer-to-peer within a fog domain and through cloud cross fog domain. Besides, node distribution and proximity characteristics allow controlling location, mobility, network condition, and behavior to efficiently implement customized services to meet the required SLA and QoS.

Instead of waiting for batch processing in a data center, fog nodes provide computing services closer to the end-user devices [18]. And thus, it is suitable for real-time interactive applications. A fog node can filter irrelevant data and significantly aggregate only essential data that should be streamed to the cloud to reduce the workload among local and remote servers. Likewise, the rapid growth of IoT devices and applications reduces the energy efficiency of the system. However, tasks can be offloaded from end devices to fog nodes, which is closer to the end device/user and thus reduces energy consumption. The aforementioned important features make the fog computing paradigm to be used in different infrastructures such as Smart Cities [18], [62]–[65], Telesurveillance [20], Health care [20], [69]–[75], [77] and Smart Transportation [21].

A. *Fog computing architecture*

Several studies [17], [23], [24], and [6] have proposed various architectures of Fog Computing. The OpenFog reference architecture (shown in Fig. 2) is the most comprehensive one in which most computing characteristics are considered. The OpenFog architecture description is a composite of perspectives and multiple stakeholder views used to satisfy a given fog computing deployment or scenario. The three views that

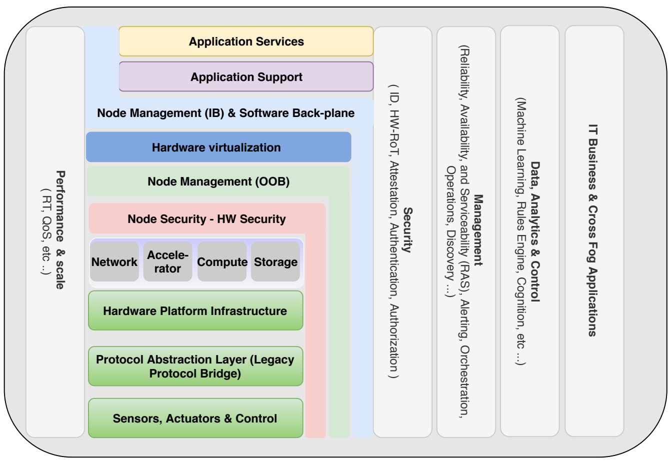

Application Services
Application Support
Node Management (IB) & Software Back-plane
Hardware virtualization
Node Management (OOB)
Node Security - HW Security
Network
Accele-rator
Compute
Storage
Hardware Platform Infrastructure
Protocol Abstraction Layer (Legacy Protocol Bridge)
Sensors, Actuators & Control
Performance & scale (RT, QoS, etc ..)
Security (ID, HW-RoT, Attestation, Authentication, Authorization)
Management (Reliability, Availability, and Serviceability (RAS), Alerting, Orchestration, Operations, Discovery ...)
Data, Analytics & Control (Machine Learning, Rules Engine, Cognition, etc ...)
IT Business & Cross Fog Applications

Fig. 2. Fog Computing Reference Architecture

the authors of [6] have identified include Software, System, and Node.

- *Software view:* It is represented in the top three layers shown in the architecture description and includes Application Services, Application Support, Node Management (IB), and Software Backplane.
- *System view:* It is represented in the middle layers, which include Hardware Virtualization down through the Hardware Platform Infrastructure.
- *Node view:* It is represented in the bottom two layers, which include the Protocol Abstraction Layer and Sensors, Actuators, and Control. A Fog computing environment comprises a hierarchical arrangement of fog nodes throughout the network between end-user devices and sensors, and the cloud at the core of the network.

B. Challenges in fog computing

Fog computing provides several advantages over the centralized cloud computing such as low latency, mobility support, and location awareness. However, it faces many challenges in its implementation and operation. In this subsection, we go through four main challenges in fog computing.

1) *Scalability and node placement:* As we have discussed in the prior sections, the number of IoT devices is growing drastically (it is in the order of billions), producing large and heterogeneous data. Therefore, a fog node with a sufficient amount of resources to treat the rise of IoT devices and applications is required. However, fog nodes are resource restrained to satisfy the demands [25]. A group of fog nodes operates collaboratively and coordinately to provide a better service to their customers. Therefore, optimal placement of fog nodes would enhance the chance to support the number of users with better quality. Thus, various user demands at different locations, and how to optimally distribute the nodes becomes a challenging problem [26].
2) *Heterogeneity and design complexity:* Fog computing aims to support diverse and massive IoT devices and sensors designed by different manufacturers with different software and hardware requirements. The devices have various specifications such as computing power, storage, communication protocol, and security level. Therefore, designing a fog node to support those issues is cumbersome [25].
3) *Dynamic resource management:* Fog computing resources appear and disappear rapidly. Therefore, the dynamic variation and volatility property of fog resources make them difficult to manage [25], [27].
4) *Security and privacy:* Fog nodes contain highly distributed, heterogeneous and limited resources such as memory, CPU, and RAM that are not suitable to deploy the existing security and privacy algorithms. Therefore, new techniques for data privacy, usage privacy, and location privacy adapted to these constraints need to be designed [25], [28].

IV. A TAXONOMY OF MONITORING SOLUTIONS IN FOG COMPUTING AND CHALLENGES

Monitoring is a vital component in a management system, which is responsible for tracking changes in an operational infrastructure to identify and take remedy against any deterioration in the system performance. Besides, it is essential to ensure the Quality of Service (QoS) of applications, Quality of Experience (QoE) for the end-users, and provide information and Key Performance Indicators (KPIs) for both platforms and applications. Furthermore, monitoring is an important task to guarantee a Service Level Agreement (SLA) between providers and consumers by controlling and managing hardware and software infrastructures, tracing and taking remedy against violations of agreed terms (for both the provider and the consumers) [29], [30]. Monitoring in an environment as complex, heterogeneous, and volatile as fog computing is an extreme challenge. Analyzing this challenge and providing directions for future research is the outcome of this article.

This section introduces a taxonomy for the analysis of monitoring solutions in fog computing. Fig. 3 summarizes the possible classification of desired monitoring solutions in fog computing, based on an analysis of different parameters. The parameters chosen for this taxonomy include: (*i*) Monitoring architecture, (*ii*) Monitoring Design parameters, and (*iii*) Monitoring requirements. After presenting this taxonomy, we analyze the main challenges of monitoring in fog computing.

A. Monitoring Architecture

A monitoring system with very particular features is required to monitor the fast-changing and appear-disappear behavior of the heterogeneous fog resources. Furthermore, fog nodes are resource-constrained, and hence, a lightweight monitoring method is required to deploy and operate with small resource consumption. Therefore, as depicted in Fig. 4, monitoring systems are decomposed into three service functions and, they can be deployed in different devices to enhance the monitoring system availability and non-intrusiveness [7],

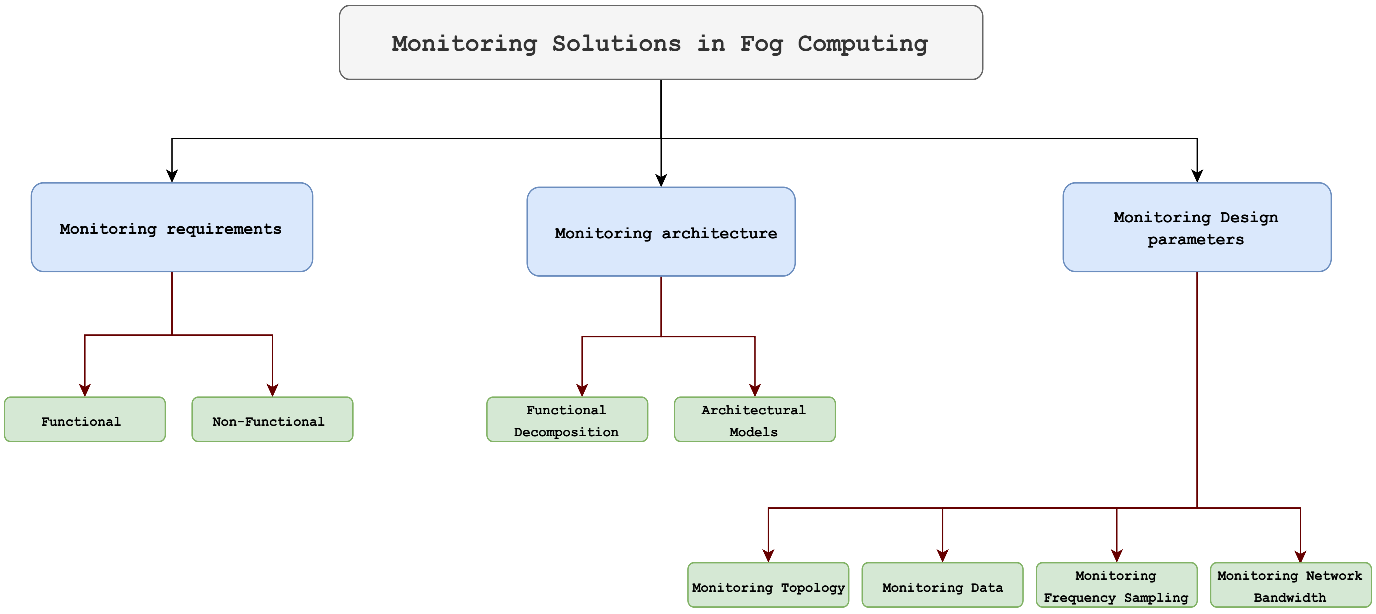


Fig. 3. Taxonomy of monitoring solutions in fog computing.

[23], [29]–[32], [78]. The three fundamental service functions of a monitoring system are [30]:

1) *Observation function:* It collects monitoring data from distributed probes through specified APIs and sends them to the processing function.
2) *Processing function:* It processes the collected measured-data, which includes measurement data aggregation and filtering, measurement transformation into events, event processing, and notification management. Finally, it sends the processed measured data to the exposition session.
3) *Exposition function:* It sends measured metrics values, alerts/notifications to the corresponding management system(s).

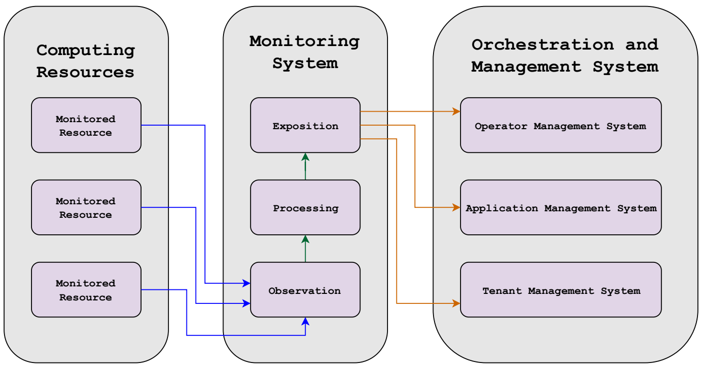


Fig. 4. Fundamental Monitoring Services and Functional Decomposition

The authors of [30] distinguished nine categories of monitoring architectures that differ from each other according to their functional decomposition (none, basic or fine-grained) and their architectural model (centralized, hierarchical, or peer-to-peer). The monitoring architecture is essential to deploy a monitoring system with high availability. Besides, decomposing and geographically distributing the monitoring services play a prominent role to overcome the fog resource constraints, locality, and mobility of users.

The monitoring function is performed solely via a monolithic component in the No Decomposition category. Furthermore, the observation function is deployed on the monitored resource, separately from the other monitoring functions (processing and exposition) in the Basic Decomposition monitoring decomposition. In the Fine-grained Decomposition category, each of the monitoring functions deployed in separate servers or devices.

Furthermore, in the Centralized Model, only one instance of each monitoring function is deployed on a centralized fog node except the observation function. And hence, several instances locally deployed on the fog resources. The hierarchical model deploys an adequate number of instances of each monitoring function according to the size of the infrastructure and the management systems' requirements, unlike the centralized model. The Peer-to-peer Model allows the replication of instances without forcing any distinct structure to organize the relationship between them. It allows multiple instances of a given function and communication between instances. The nine categories of monitoring architectures are summarized in Fig. 5.

B. Monitoring Parameters

The careful design of a monitoring system capable of collecting and aggregating metrics on time is required to achieve the main monitoring features (such as timeliness, scalability, non-intrusiveness, accuracy, and adaptability). These monitoring requirements are, however, affected by parameters like

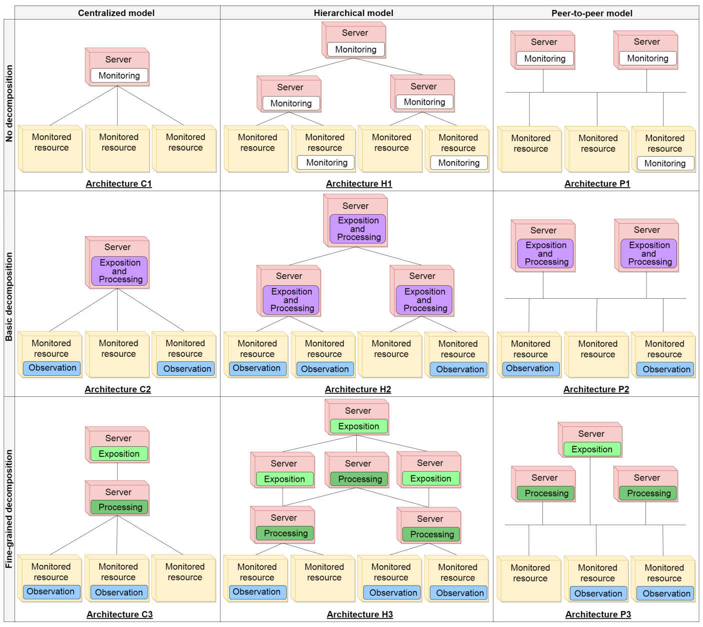


Fig. 5. Monitoring Architectures Classification Rows correspond to the decomposition granularity from no to fine-grained decomposition. Columns correspond to the software architecture model: centralized, hierarchical and peer-to-peer. [30]

monitoring topology, monitoring data, and frequency sampling [53]–[55].

- *Monitoring Topology:* It describes how the monitoring services (agents and servers) are placed. Monitoring topologies are structures utilized as a support to the collecting and publishing of monitoring data. Based on the functional decomposition and architectural models, elements for monitoring data collection, processing, and publishing are installed with different topologies, e.g., with different distances, delays, number of communication links, and *network bandwidth*. Moreover, the topology between the monitoring agents and managers affects system performance.
- *Monitoring data:* It is the measurement metrics from distributed probes, such as %CPU usage, %Memory usage, throughput, delay, and jitter. Therefore, a monitoring system is required to collect, process, and publish a certain amount and types of monitoring data generated from monitored resources at a given monitoring time interval. Large amounts of monitoring data typically cause delays between the event occurrence and warning notification (i.e., response time). Therefore, an appropriate monitoring data choice helps us to avoid violations in SLAs. Furthermore, since processing the monitoring data requires resources, they impair the monitoring performance in the resource-constrained fog computing environment.
- *Frequency sampling:* It identifies how often monitoring data is collected. Using short monitoring intervals (high-frequency sampling) facilitates monitoring events on time. However, that increases the overhead to the system. Furthermore, along the monitoring interval overcomes the overhead issue at the sacrifice of monitoring accuracy. Therefore, an appropriate and optimum monitoring interval that changes dynamically is required to monitor dynamic, volatile, and resource-constrained fog resources.

C. Monitoring requirements

Monitoring systems are in charge of collecting measurement metrics, process them, and send the results to the management system. In this section, we review desirable features that should be achieved by a monitoring service. In general, there are two types of monitoring requirements: Functional and Non-Functional requirements, which we describe below.

1) Non-Functional Requirements:

- *Scalability and heterogeneity:* A monitoring system is required to monitor a diverse and enormous number of monitored resources and services regardless of the technology used for its development [31], [32]. Moreover, monitoring systems need to be able to control resources at multiple levels, such as VM-level, Container-level, end-to-end link quality, and application levels [34].
- *Adaptability and dynamism:* A monitoring system has to adapt to the dynamic variations in fog computing resources, for example, computational load and network load. Besides, it has to be able to provide mechanisms to dynamically adjust the configuration of its elements, such as the measurement of the collection/sending rate according to the status of the infrastructure, the number of running entities, and the characteristics of the network service instances [31], [33].
- *Robustness:* A monitoring system has to be robust against failures and detect changes in the environment, adapting to new situations, and remaining operative [31].
- *Non-intrusiveness:* Lightweight tools are required to monitor applications and infrastructures [31], [32], [77], which can be managed by the resource-constrained fog devices.
- *Interoperability and Federation:* Monitoring systems should not be specific to a given infrastructure or application that resides on other fog domain infrastructure [31], [32]. And it needs to work cooperatively with other monitoring systems.
- *Elasticity:* A monitoring system has to monitor resources regardless of the amount of data so that virtual resources are created and destroyed by expanding and contracting networks are monitored correctly [32].
- *Autonomic:* A monitoring framework has to keep running and reconfiguration without human intervention [32].

2) Functional Requirements:

- *Monitoring of time-varying virtual and physical resources:* Fog computing resources are heterogeneous and include both virtualized and physical devices. Besides, mobility is high at this new computing paradigm, where measured values vary dynamically. Therefore, a monitoring system that able to monitor time-varying virtual and physical underlay resources is required.

- *Support of adaptive monitoring sampling frequency:* Adapting the monitoring interval is required to monitor the dynamic and resource-constrained fog resources without incurring high signaling overheads and latency.
- *Support of adaptive filtering monitoring data:* A monitoring system is required to collect, process, and publish a certain amount and types of monitoring data generated from monitored resources at a given monitoring time interval. A large amount of monitoring data causes a delay between the event occurrence and warning notification (i.e., response time). Therefore, an appropriate monitoring data choice helps us to avoid violations in SLAs and consequent financial penalties of an infrastructure. Consequently, an optimum and dynamic filtering approach is needed.
- *Automatic detection of removed and added resources:* In fog computing environments, applications can migrate from a physical/virtual host to another one at any time. Furthermore, resources are volatile. Thus, the monitoring system has to be able to monitor correctly any resources [31] and able to detect the arrival and removal of resources.

D. Monitoring challenges in Fog computing based infrastructures

Fog/edge infrastructures have some intrinsic characteristics that make monitoring design complex and challenging [30], [31], [34]:

1) Large and massively distributed: Edge/Fog resources are placed across a large number of distributed sites. In addition, the distance fluctuates based on the users' mobility and the nature of the link connecting them.
2) Heterogeneous: Fog based infrastructure contains various types of resources, such as storage servers, compute servers, specific routers, general-purpose switches, network links, home gateways, user devices, and application platforms. All those resources have different characteristics in terms of capacity (e.g., high-performance servers vs. modest edge devices), reliability (dedicated to the infrastructure like routers in PoPs or provisional like an offered user terminal), and usage. Virtualization introduces another level of heterogeneity since the operating resources can be physical or virtual.
3) Highly dynamic: Various applications in Fog/Edge environment are frequently instantiated and migrated at variable rates. Similarly, network topologies need to be modified on-the-fly to cope up with the dynamic changes both in the resources and end-user behavior. Finally, it also occurs at the lowest level of the infrastructure where end devices or users may join and leave the network permanently or temporally according to service usage, failures, policies, and maintenance operations.

V. Review and Analysis of State-of-Art of Existing Monitoring Solutions

This section discusses the state of the art of current monitoring solutions. For that purpose, we analyze existing monitoring services and tools of cloud computing resources as they have similarities with fog resources. However, we conclude using our requirements' taxonomy that cloud monitoring approaches are not suitable for fog environments.

General-purpose infrastructure monitoring tools typically utilize a client-server model by placing an agent in every computing resources to be monitored. As shown in Fig. 6, monitoring agents are in charge of measuring relevant metrics from the monitored components and send them to the monitoring server. The monitoring server stores the collected metrics into a database, analyzes them, and sends alert or notification to the management system. Furthermore, it may generate graphs, trending reports, SLA reports based on the monitored metric values retrieved from the database [35].

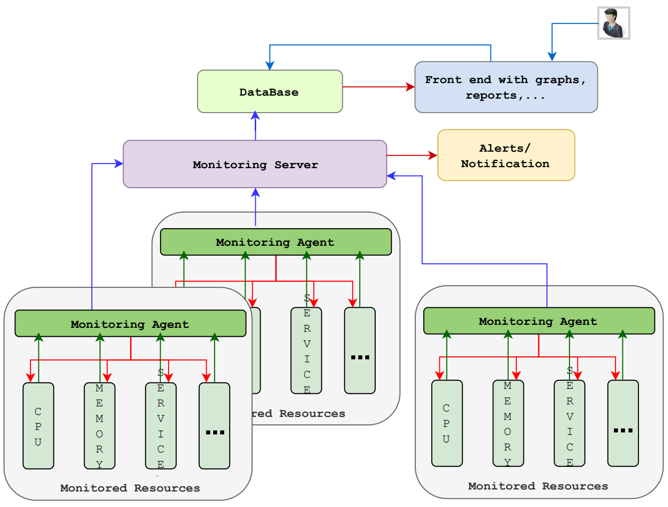


Fig. 6. General purpose infrastructure monitoring tool architecture

Ganglia [36] is a robust, hierarchically distributed, and scalable monitoring system designed to monitor high-performance computing systems such as clusters and Grids. Moreover, it leverages widely used technologies such as XML for data representation, XDR for compact, portable data transport, and RRDtool for data storage and visualization. Thus, all nodes can have the information of the whole clusters state. For this reason, Ganglia was used as a fundamental monitoring function to develop different management systems. For example, YCSB++ (Yahoo Cloud Serving Benchmark extension) uses Ganglia as a monitoring system [37]. However, Ganglia was designed for fixed, and relatively slowly changing physical infrastructure. Moreover, Ganglia employs the basic decomposition and hierarchical architectures, H2, model, where the observation function is dispatched from the rest of the monitoring service. Therefore, the monitoring system is not non-intrusive and adaptable to deploy. Ganglia lacks the capability of the auto-discovery of a dynamic appearance and disappearance of resources, because, Ganglia monitoring parameters are static, which are not able to detect a dynamic and fast change of values of measurement metrics. Ganglia could not be applied in a rapidly changing and dynamic infrastructure [32], as seen in the fog computing environments. Accordingly, Ganglia is not a suitable choice as a fog monitoring solution.

Nagios[7] is an open source monitoring system to monitor networks, servers, and applications of computing infrastructure. It continuously observes the network, application, service, or process problems, and take action to reduce downtime for the application users. Furthermore, Nagios sends an alarm when a failure occurs, and notifies to the management system when a problem has been detected and solved [38]. Nagios requires an additional resource for exporting collected measured data due to the lack of an internal database that limits important monitoring features such as adaptability, scalability, and non-intrusiveness. Moreover, Nagios has a basic decomposition with centralized architecture, C2, where only the observation function is isolated from the exposition and processing monitoring functions. Thus, it lacks non-intrusiveness and resilience to network failure. Furthermore, the monitoring parameters are static, which is not able to detect a dynamic and fast change of values of resources. Nagios is difficult to configure and lacks the automatic discovery of devices [39] and only suitable for monitoring resources that do not require a high rate to update [40]. For that reason, Nagios is not capable to efficiently monitor volatile and dynamic resources as seen in the fog environment.

MonALISA (Monitoring Agents in A Large Integrated Services Architecture) [41] is another robust and scalable monitoring system that provides a distributed monitoring service. Monalisa provides interfaces to integrate multiple monitoring services to collect monitoring metrics value of both infrastructure and applications by leveraging an SNMP protocol. However, like Ganglia and Nagios, Monalisa is designed for relatively fixed and slowly changing resources which makes it unfit to monitor resources that appear-disappear or migrate from one node to another rapidly as in the fog nodes. Moreover, Monalisa has no decomposition with peer-to-peer architecture, P1, where all the three monitoring functions could not be separated. Thus, it lacks non-intrusiveness and resilience to network failure. Furthermore, the monitoring parameters are static that is not able to detect a dynamic and fast change of values of resources.

Zabbix [42] is an opensource monitoring system to monitor massive and distributed computing resources. The design is based on the hierarchical architecture of Zabbix servers to improve the scalability features of the monitoring system. Moreover, an alert is sent to the system administrator from the servers when a problem detected in the infrastructure. Zabbix is developed in C programming and relatively a lightweight monitoring system. However, the server components are centralized and hierarchically arranged to monitor federated cloud resources. As a result, the system is not resilient for its deployment and operation. Moreover, Zabbix has no decomposition with a hierarchical architecture, H2, where the three monitoring functions could not be isolated. Therefore, Zabbix lacks non-intrusiveness and resilience to network failure. Furthermore, the monitoring parameters are static that is not able to detect a dynamic and fast variance of values of resources. Furthermore, auto-discovery in Zabbix takes a long time, and thus, suitable only for fixed and slowly changing resources. For those reasons, Zabbix is an unfit monitoring system for fog resources.

The ***private cloud monitoring system (PCMONS)*** [43] is another opensource monitoring system which was primarily developed to overcome the lack of opensource solutions for private cloud systems. PCMONS retrieves, gathers, and prepares relevant information that integrated with the Eucalyptus IaaS platform and Nagios for monitoring data for visualization. Thus, it inherits the behavior of Nagios that is only suitable to monitor relatively fixed and gradually changing resources.

OpenNebula [44], [45] is an opensource industry-standard cloud management toolkit. The OpenNebula monitoring subsystem monitors the resources of infrastructure such as host status, VM status, and capacity consumption by executing a set of static probes. The output of these probes are sent to OpenNebula either in the push or pull model. In pull mode, OpenNebula periodically actively queries each host and executes the probes via SSH. While in the UDP-based push mode, each host periodically sends monitoring data via UDP to the frontend for further process in a dedicated module. The PANACEA[8] project and OnLife project [46] use OpenNebula based cloud management system due to its lightweight, easy installation and configuration features. However, the monitoring data updating frequency is static and periodic, which is not able to monitor in a dynamic and volatile resource environment. Moreover, the OpenNebula monitoring system has not addressed interoperability requirements.

The ***Distributed Architecture for Resource manaGement and mOnitoring in cloudS (DARGOS)*** [47] is an opensource distributed monitoring system using a hybrid push/pull model to disseminate cloud resource information and provides measurements of both physical and virtual resources in the cloud infrastructure. Moreover, DARGOS provides adaptability and extensibility with new metrics. And, it integrates into a cloud deployment based on the OpenStack platform. Although DARGOS gives several features such as extensibility, flexibility, and non-intrusiveness, the monitoring method is not dynamic. However, DARGOS does not support the live-migration of servers, which hinders monitoring capabilities in an environment where rapid resource migration occurs.

The ***Lattice*** [33] monitoring system is an open-source, lightweight, and interoperable with several monitoring services introduced by the RESERVOIR project [57]. It was first developed for monitoring resources and services in virtualized environments, which comprise a large number of resources.

[7]https://www.nagios.org/

[8]http://projects.laas.fr/panacea-cloud/

TABLE I
ANALYSIS OF EXISTING MONITORING SYSTEMS

Tools	Architecture	Dynamic Monitoring Parameter Selection Capability	Auto Appear/Disappear Resource Detection
Ganglia	H2	No	Limited
Zabbix	H1	No	No
MonALISA	P1	No	Limited
Nagios	C2	No	No
PCMONS	H1	No	No
OpenNebula	C1	No	Limited
DARGOS	H1	No	No
Lattice	H2	No	Limited
JCatascopia	H2	No	Yes

TABLE II
SUMMARY OF RESEARCH PAPERS ON EXISTING MONITORING SOLUTIONS AND ANALYSIS OF THE APPLICABILITY TO EDGE/FOG

Paper	Summary of the paper	Analysis of the applicability to Edge/Fog
[30]	Holistic monitoring for a Fog/Edge infrastructure (qualitatively analysis of the existing solutions and proposed monitoring framework by leveraging the architectural model and functional decomposition model)	The most appropriate design (i.e., fine-grained functional decomposition with peer-to-peer architectural model) has not been implemented yet
[33]	Monitoring resources and services in virtualized federated clouds	Sampling-frequency and filtering monitoring data are static and periodic, and it does not provide full automation to detect when resources are added/removed
[36]	Monitoring federated cloud resources	Designed for fixed, and relatively slowly changing physical infrastructure. Moreover, it lacks the capability of automatic resource detection, and monitoring parameters are static
[38]	Monitoring the network, servers, and applications of a cloud computing infrastructure	It lacks automatic discovery of devices and it is only suitable for monitoring static resources
[41]	Monitoring distributed cloud resources	It lacks auto-recovery and auto-configuration features, designed for relatively fixed and slowly changing resources
[42]	Monitoring federated cloud resources	Lack of robustness, auto-discovery takes a long time and thus, only suitable for fixed and gradually changing resources
[43]	Monitoring private clouds	It lacks auto-recovery and auto-configuration, designed for relatively fixed and slowly changing resources
[44]	Monitoring distributed computing resources	The mechanism is interoperable. Updating frequency and filtering metrics are periodic and static
[47]	Monitoring distributed computing resources	It does not support live migration, and lacks dynamic monitoring scheme
[49]	Fully automated and dynamic resource monitoring	It requires an optimum algorithm for dynamic metric filtering and sampling frequency, JVM requirement limits its non-intrusiveness
[51]	Monitoring the QoS of metrics like throughput, delay, and packet loss	Periodic and static sampling frequency and constant monitoring data
[58]	Intelligent resource provisioning	The monitoring parameters are static
[59]	Network level monitoring applications over federated cloud infrastructures	The monitoring parameters are static
[60]	Latency monitoring for applications locally executed on the end user and over cloud infrastructures	Monitoring parameters are static and do not consider the dynamically varying environment
[52]	Multi-layered monitoring framework for measuring QoS at both application and infrastructure levels	It lacks an algorithm for optimum and dynamic selection of sampling frequency and amount of monitoring data
[53]	Demonstrating the mutual influence between the timeliness and monitoring requirements	No clear algorithm to select the monitoring time interval
[54]	Multi-level monitoring framework with a self-adaptive algorithm	The monitoring time-interval is constant. In addition, adding/removing one instance at a time is not flexible enough
[55]	Mathematical model to predict the interplay between timeliness and scalability for FAT-Tree topology.	Only for FAT-Tree topology based infrastructure and it does not explore the mathematical model for dynamic variation of resources

Lattice framework uses data sources and probes to collect and distribute various monitoring data using either multicast or UDP protocols. However, Lattice does not provide functions for visualization, evaluation, and automated alerts and notification [34]. Hence, Lattice is not a suitable monitoring solution in a fog environment where full automation, self-configuration, and self-healing are the most critical monitoring requirements. Moreover, Lattice does not have a library for monitoring probes, so developers need to implement their library. That is one of the significant limitations of Lattice to deploy in the resource-constrained fog computing frameworks. However, monitoring parameters such as sampling-frequency and filtering monitoring data are static and periodic.

JCatascopia [49] is an opensource, interoperable, and scalable monitoring system which supports a fully automated and dynamic resource provisioning. Moreover, it supports measurement metrics filtering with some communication and storage overheads. Catascopia monitoring system comprises three main components: *Monitoring Agents* are lightweight and deployable on any physical or virtual instances. And, they are responsible for the processing and aggregating collected metrics from different instances and transfer them to corresponding monitoring servers. *Probes* are metric collectors managed by monitoring agents. Probes are in charge of collecting low-level monitoring metrics from the monitored resources they reside on, and performance metrics from deployed user applications. *Monitoring Server* processes the collected metrics, and deployable on either physical or virtual instances without restriction to place in the same cloud platform with their monitoring agents. Furthermore, JCatascopia identifies dynamically when monitoring instances are added/removed. For those reasons, Jcatascopia was chosen as a baseline technology for developing a monitoring system for the SWITCH[9] and CELAR [50] projects. However, JCatascopia is written in Java, each container, which constitutes a Monitoring Agent requires some packages and a certain amount of memory for a Java virtual machine (JVM) even if the monitored application running in the container is not programmed in Java. Furthermore, although the different levels of filtering and sampling frequencies are important monitoring features in fog environment, optimized values are required to monitor the dynamic and volatile resources. Those are some of the main limitations of deploying JCatascopia in resource-constrained devices.

Authors of [58] introduced a lightweight and scalable monitoring system that can monitor with small collection interval to trace the dynamic changes in the resources. However, the fine-grained monitoring interval increases overhead in the system. Furthermore, the monitoring parameters are static, which cannot be applied in highly dynamic conditions. Likewise, authors of [59] developed a lightweight monitoring framework for applications over federated cloud infrastructures to monitor network-level parameters, such as round trip time, packet loss, and latency. Furthermore, another semi-automatic monitoring system for video gaming [60] named GALAMETO.KOM developed to monitor latency both on the cloud and end-user devices. However, both articles do not count the dynamically changing resource characteristics, as observed in the fog computing-based infrastructure.

Taherizadeh [51] demonstrated run-time variations in the network quality of links between application servers and end-users. The system is implemented in JCatascopia monitoring framework which, monitors QoS of metrics including throughput, delay, jitter, and packet loss of a connection link measured by monitoring probes on top of each edge node at the distinct time interval. However, periodic sampling is not an appropriate parameter to optimize the tradeoff between accuracy and resource utilization (such as bandwidth, memory, and CPU exploited by the monitoring data). Furthermore, [52] presented a monitoring system that facilitates on-the-fly self-configuration for measuring QoS at both application and infrastructure levels. They demonstrated the effect of monitoring time interval on the average CPU load, and average bandwidth used for publishing an application metrics to the monitoring platform. Furthermore, they evaluated monitoring accuracy at different sampling periods. Moreover, [53] demonstrated the mutual influence between the timeliness and scalability, average response time with the network topologies, amount of monitoring data, and frequency sampling. However, the authors of both papers did not describe monitoring interval selection algorithms.

[54] presented a JCatascopia based multi-level monitoring framework with a self-adaptive algorithm to add certain application services. When the value of the metrics is still below a given threshold, it terminates one of the running container instances or applications whenever the measurement metrics are below a certain threshold so that the system performance will not degrade. However, the monitoring time interval is static, and adding/removing one instance at a time is not efficient enough in a dynamic and volatile environment as in fog computing. As an extension of this work, they presented in [55] a mathematical model to predict the interplay between timeliness and scalability for FAT-Tree topology. The response time is modeled mathematically in terms of frequency sampling, network bandwidth, network topology, and amount of monitoring data. However, the model is only for FAT-Tree topology-based infrastructure.

To the best of our knowledge, none of the existing monitoring services are suitable for the heterogeneous, dynamic, and volatile fog computing resources. Existing monitoring systems usually require constant connectivity, huge bandwidth, and resources for continuous monitoring. However, the fog monitoring system needs to be light-weighted and able to monitor rapidly changing resources. Table I shows the analysis of the existing monitoring solutions characterized by the monitoring requirements. Furthermore, Table II summarizes existing monitoring solutions research papers and how they apply to Edge/Fog computing environments.

VI. Fog Monitoring Solution Case Studies

This section describes the case studies that are important to understand the requirements, features, and roles of fog monitoring solutions in different domains. Some of the use-cases and solutions using fog computing to investigate the monitoring solutions are:

A. *Smart City*

The emerging problems due to the rising urban migration, and rapid population growth, the citizens in large cities encounter several difficulties. Furthermore, the development and prosperity of society remain a new challenging issue. A smart city is a framework that is used by several academia and industries for large cities to improve the utilization of city resources, increase operational efficiency, share information with the public, and improve both the quality of government services and citizen welfare [62], [63].

The main objective of the concept of fog computing enabled smart city [64] is to optimize resource consumption. The services provided in a wide range of areas, such as healthcare, transportation, buildings, education, energy, and

[9]http://www.switchproject.eu/

so on, are integrated into the concept of a smart city. These services solve several problems face in big cities such as air pollution, shortage of resources, traffic congestion, and so on, in the urban area. In fog architectures, orchestration should be distributed across the multiple fog nodes, which are then responsible for the local resource provisioning and deployment of applications and services, thereby ensuring the necessary Quality of Service (QoS). Furthermore, fog computing framework enabling autonomous management and orchestration functionalities in 5G-enabled smart cities [65].

B. Smart Industry

Industry 4.0 describes the growing trend towards automation and data exchange in manufacturing technologies and processes. It includes diverse emerging technologies such as the industrial internet of things (IIoT), cyber-physical systems (CPS), Smart manufacture, and Smart factories. Industrial Internet-of-Things (IIoT) applications require real-time processing, near-by storage, ultra-low latency, reliability, and high data rate, all of which can be accomplished by fog computing architecture [66]–[68].

Fog nodes in factories[10] can take advantage of streaming data in a layered model. A fog node connected to a set of local sensors and actuators analyzes the data, interprets an anomaly, and then if authorized, could autonomously react and compensate for the problem or fix the issue. Alternatively, the fog node can send the appropriate requests for service higher up the fog hierarchy to more skilled technical resources, machine learning capabilities, or a maintenance service provider. Furthermore, topologies and heterogeneity of distributed fog computing enabled infrastructures in a factory can vary considerably; in many cases, intelligent management and orchestration systems are required, capable of dynamically deploying, managing, and elastically scaling the computing infrastructure.

Monitoring in the fog enabled industry is a vital tool to track applications and resource for real-time decision-making. Therefore, it avoids potential latency issues, queue delays, or network/server downtime that could result in industrial accidents, reduced production efficiency, or poor product quality. Furthermore, it can add more functionality such as visualization of production line operation, monitoring the status of malfunctioning machines, tuning of production parameters, modification of production planning, ordering supplies, and sending alerts to the appropriate people.

C. Smart Health-care

The communication network in health-care infrastructure plays a great role in determining the quality of medical service delivery. For example, network failure or late service delivery affects the patient quality of life even leads to death. The IoT-related health-care systems are low-powered network devices such as wearable or implantable sensors that collect and share biophysical data over time and provide intelligent decisions for preventing serious illness. The fog computing node is an intermediate layer between the sensors and remote cloud data center that alleviates a series of issues in the area of bandwidth consumption reduction, latency decrease, and seamless operation which collectively provides reliable services [69]–[71]. Furthermore, fog nodes are geographically distributed which enables to monitor of the condition, location, and mobility of patients. Moreover, they collect a number of heterogeneous data generated IoT-enabled health-care applications that need to be managed and monitored effectively.

The integration of IoT and fog computing paradigm provides reliable medical service provisioning with high security and privacy of patients with fast and accurate treatment delivery, reduction of medical cost, improvement of doctor-patient contacts, and the delivery of personalized treatment-oriented to users need or preferences [69]–[73]. Data management has an important role in fog computing enabled health-care systems. The medical sensory data is locally processed to extract meaningful information for user feedback and notifications. According to health-care system architecture [74], [75], [77], the fog node continuously receives a large amount of sensory data in a short period from the sensor network. Thus, it should manage the incoming data to provide a fast response regarding various user and system conditions to avoid latency and uncertainty in decision making that might cause irreversible damages for the patients. The fog computing system provides the basic functionality of protocol conversion, local storage, data analysis, filtering, confining, and fusion. To provide these functionalities accurately, an efficient and effective monitoring system is required. As aforementioned, problems in monitoring and managing huge medical sensory data might lead to delay in treatment and bad decision making. Therefore, designing a reliable and efficient monitoring solution is required to avoid problems.

VII. Open Issues and future research directions

In addition to the research results reported above, in the next future, we foresee different possible research directions for monitoring solutions in fog computing as detailed in the following.

1) New Monitoring Approach
   The authors of [77] suggested new directions for the research to develop methods for the monitoring of edge/fog computing. For example, the new monitoring solution needs to be able to manage movement monitoring, monitoring data, and monitor decentralized resources. Furthermore, it needs to be able to manage intelligent resource scheduling using intelligent workload offloading with scheduling algorithms such as moth-flame optimization algorithm [76] and manage to monitor service replications. In addition, effective monitoring techniques should be able to provide very fine-grained measures and able to trace the randomly fast-changing fog resource values. At the same time, the techniques should optimize the performance over-

[10]https://www.controleng.com/articles/fog-computing-for-industrial-automation/

head to the system. Therefore, appropriate optimization techniques or algorithms are required to be analyzed and implemented with the monitoring techniques. And thus, new monitoring techniques and tools specifically designed for fog computing paradigm are needed.

2) Monitoring in Federated Fog Computing
The standardized collaboration across multiple fog infrastructures is referred to as resource federation, which is required for the efficient and automated provision of resources. Monitoring in such an environment plays a great role to manage resources and taking a decision so that the system performance would improve. However, such a standardization process and monitoring solutions are still at their early stage. The high heterogeneity among different fog monitoring infrastructures challenges the possibility to obtain a comprehensive solution for federated fog nodes, and this has not been properly addressed in ligature yet.

3) Energy and Cost Efficient Monitoring
Monitoring activities can be highly demanding in the dynamic, heterogeneous, distributed, and volatile fog environment. Furthermore, accurate and fine-grained monitoring activity is required to detect the monitoring variables which costs terms of computing and communication resources, and therefore in terms of energy and cost. Designing a cost-efficient and green monitoring solution in such an environment while assuring the monitoring requirements (i.e., functional and non-functional) with minimizing the related energy consumption and cost.

VIII. Conclusion and future works

Fog computing provides several important features compared to cloud computing such as scalability, mobility support, low-latency, and energy efficiency. However, due to its volatile (appear-disappear), heterogeneous and dynamic resource characteristics, designing a monitoring system for fog is way harder than for cloud. In this article, we describe the fog computing architecture and discuss how it provided additional benefits compared to the centralized traditional cloud-only computing paradigm. We analyze existing monitoring solutions for fog resources based on identified taxonomy. This analysis shows that, although there are several existing monitoring solutions, none of them are suitable for the fog computing paradigm. A major challenge of resource monitoring in fog computing is the need to properly adapt what to monitor and how often so it adapts to the volatility of the fog environment.

In addition, we consider the influence of the monitoring parameters on the monitoring features. Monitoring data and monitoring frequency sampling impair most of the monitoring requirement features such as scalability, and timeliness due to the resource constraints in fog devices. Therefore, an optimum system model to analyze the tradeoff between the monitoring parameters and the monitoring features is required. As a result, we foresee different possible research directions for monitoring solutions in fog computing.

Machine Learning is a popular method that has been used to model network parameters. The MF2C Project[11] is a machine learning project aimed to provide a modeling scheme to operate autonomously between the cloud and fog computing paradigms and create an interoperability fog-to-cloud framework. In addition, [48], [61] leverage Machine Learning algorithms for predicting offloading scheduling in edge computing and Fog Radio Access Network (F-RAN) respectively. Furthermore, [56] describe machine learning algorithms for modeling network quality parameters. Besides, we insist that machine learning algorithms enable to design an intelligent mechanism to provide a monitoring solution for the fog/edge operator infrastructure. Therefore, analyzing dynamic monitoring interval and filtering monitoring data by leveraging machine learning is our future research work. The analysis and implementation of an appropriate and lightweight monitoring algorithm, suitable for constrained fog resources falls also within our future research.

Acknowledgement

This work has been (partially) funded by H2020 EU/TW 5G-DIVE (Grant 859881) and H2020 5Growth (Grant 856709). It has been also funded by the Spanish State Research Agency (TRUE5G project, PID2019-108713RB-C52PID2019-108713RB-C52 / AEI / 10.13039/501100011033).

References

[1] Zhang, Q., Cheng, L., Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications, 1(1), 7-18.

[2] D. Kreutz et al., Software-defined networking: A comprehensive survey, Proc. IEEE, vol. 103, no. 1, pp. 1476, Jan. 2015.

[3] ETSI, Network Functions Virtualisation (NFV); Management and Orchestration; Architectural Options, European Telecommunications Standards Institute, GS NFV-IFA 009, July 2016.

[4] ETSI, Mobile Edge Computing (MEC); Framework and Reference Architecture, European Telecommunications Standards Institute, GS MEC 003, Mar 2016.

[5] Hu YC, Patel M, Sabella D, Sprecher N, Young V. Mobile edge computingA key technology towards 5G. ETSI white paper. 2015 Sep 5;11(11):1-6.

[6] OpenFog Consortium 2017. OpenFog Reference Architecture for Fog Computing.

[7] 5G-CORAL. 2018.Initial design of 5G-CORAL orchestration and control system Deliverable 3.1.

[8] Cisco White Paper, Fog computing and the internet of things: Extend the cloud to where the things are, 2015.

[9] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, Fog computing and its role in the internet of things, in Proceedings of the first edition of the MCC workshop on Mobile cloud computing, pp. 1316, ACM, 2012

[10] Salman, Ola, et al. "Edge computing enabling the Internet of Things." 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT). IEEE, 2015.

[11] Satyanarayanan, M. (2017). The emergence of edge computing. Computer, 50(1), 30-39.

[12] Varghese, Blesson, et al. "Challenges and opportunities in edge computing." 2016 IEEE International Conference on Smart Cloud (SmartCloud). IEEE, 2016.

[13] A. Asadi, Q. Wang, and V. Mancuso, A survey on device-to-device communication in cellular networks, IEEE Communications Surveys Tutorials, vol. 16, pp. 18011819, Fourthquarter 2014.

[11] http://www.mf2c-project.eu/

[14] M. Aazam, S. Zeadally, and K. A. Harras. Fog computing architecture, evaluation, and future research directions. IEEE Communications Magazine, 56(5):4652, May 2018. ISSN 0163-6804. doi:10.1109/MCOM.2018.1700707.

[15] L. M. Vaquero and L. Rodero-Merino, Finding Your Way in the Fog: Towards a Comprehensive Definition of Fog Computing, ACM SIGCOMM Computer Commun. Review, vol.44, no. 5, 2014, pp. 2732.

[16] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, Edge computing: Vision and challenges,IEEE Internet Things J., vol. 3, no.5, pp.637646,Oct. 2016

[17] Haouari, Fatima, Ranim Faraj, and Jihad M. AlJa'am. "Fog Computing Potentials, Applications, and Challenges." International Conference on Computer and Applications (ICCA), pp. 399-406. IEEE, 2018

[18] J. He, J. Wei, K. Chen, Z. Tang, Y. Zhou, and Y. Zhang, Multitier fog computing with large-scale iot data analytics for smart cities, IEEE Internet of Things Journal, vol. 5, no. 2, pp. 677686, 2018

[19] G. Kioumourtzis, M. Skitsas, N. Zotos, and A. Sideris, Wide area video surveillane based on edge and fog computing concept, 2017

[20] P. Verma and S. K. Sood, Fog assisted-iot enabled patient health monitoring in smart homes,IEEE Internet of Things Journal, pp.11, 2018

[21] A. J. Neto, Z. Zhao, J. J. Rodrigues, H. B. Camboim, and T. Braun,Fog-based crime-assistance in smart iot transportation system, IEEE Access, vol. 6, pp. 1110111111, 2018

[22] P. Raj, and A. Raman, Handbook of Research on Cloud and Fog Computing Infrastructures for Data Science: IGI Global, 2018.

[23] Gupta, Harshit, et al. "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments." Software: Practice and Experience 47.9 (2017): 1275-1296.

[24] Dastjerdi, A. V., Buyya, R. (2016). Fog computing: Helping the Internet of Things realize its potential. Computer, 49(8), 112-116.

[25] Atlam, H., Walters, R., Wills, G. (2018). Fog Computing and the Internet of Things: A Review.Big Data and Cognitive Computing,2(2),10

[26] Luan, Tom H., et al. "Fog computing: Focusing on mobile users at the edge." arXiv preprint arXiv:1502.01815 (2015).

[27] Prakash, P., et al. "Fog Computing: Issues, Challenges and Future Directions." International Journal of Electrical and Computer Engineering (IJECE) 7.6 (2017): 3669-3673.

[28] Yi, Shanhe, Zhengrui Qin, and Qun Li. "Security and privacy issues of fog computing: A survey." International conference on wireless algorithms, systems, and applications. Springer, Cham, 2015.

[29] Hong, CH, Varghese, B. (2018). Resource Management in Fog / Edge Computing: A Survey. arXiv preprint arXiv: 1810.00305

[30] Abderrahim, Mohamed, et al. "A Holistic Monitoring Service for Fog/Edge Infrastructures: a Foresight Study." Future Internet of Things and Cloud (FiCloud), 5th International Conference on. IEEE, 2017

[31] Tusa, Francesco, Stuart Clayman, and Alex Galis. "Real-Time Management and Control of Monitoring Elements In Dynamic Cloud Network Systems." 2018 IEEE 7th International Conference on Cloud Networking (CloudNet). IEEE, 2018.

[32] Clayman, Stuart, et al. "Monitoring service clouds in the future internet." Future Internet Assembly. 2010.

[33] Clayman, Stuart, Alex Galis, and Lefteris Mamatas. "Monitoring virtual networks with lattice." Network operations and management symposium workshops (NOMS Wksps), 2010 IEEE/IFIP. IEEE, 2010.

[34] Taherizadeh, Salman, et al. "Monitoring self-adaptive applications within edge computing frameworks: A state-of-the-art review." Journal of Systems and Software 136 (2018): 19-38.

[35] Fatema, Kaniz, et al. "A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives." Journal of Parallel and Distributed Computing 74.10 (2014): 2918-2933.

[36] Massie, Matthew L., Brent N. Chun, and David E. Culler. "The ganglia distributed monitoring system: design, implementation, and experience." Parallel Computing 30, no. 7 (2004): 817-840.

[37] Patil, Swapnil, et al. "YCSB++: benchmarking and performance debugging advanced features in scalable table stores." Proceedings of the 2nd ACM Symposium on Cloud Computing. ACM, 2011.

[38] Barth, W., 2008. Nagios: System and network monitoring. No Starch Press.

[39] Mongkolluksamee, S., Pongpaibool, P., Issariyapat, C. (2010). Strengths and limitations of Nagios as a network monitoring solution. In Proceedings of the 7th International Joint Conference on Computer Science and Software Engineering (JCSSE 2010). Bangkok, Thailand (pp. 96-101).

[40] Paterson, Michael. "Evaluation of nagios for real-time cloud virtual machine monitoring." University of Victoria, Canada (2010).

[41] Newman, Harvey B., Iosif C. Legrand, Philippe Galvez, Ramiro Voicu, and Catalin Cirstoiu. "Monalisa: A distributed monitoring service architecture." arXiv preprint cs/0306096 (2003).

[42] Petruti, Catalin-Marian, et al. "Automatic Management Solution in Cloud Using NtopNG and Zabbix." 2018 17th RoEduNet Conference: Networking in Education and Research (RoEduNet). IEEE, 2018.

[43] De Chaves, S. A., Uriarte, R. B., Westphall, C. B. (2011). Toward an architecture for monitoring private clouds. IEEE Communications Magazine, 49(12), 130-137.

[44] Moniruzzaman, ABM, Nafi, KW, Hossain, SA (2014, May). An experimental study of load balancing of OpenNebula open-source cloud computing platform. In Informatics, Electronics Vision (ICIEV), 2014 International Conference on (pp. 1-6). IEEE.

[45] Avresky, Dimiter, Eliezer Dekel, David Garcia Perez, Gokce Gorbil, Eduardo Huedo, and Olivier Brun. "D4. 1: Description of Feasible Use Cases." (2014).

[46] Montero, Rubn S., Elisa Rojas, Alfonso A. Carrillo, and Ignacio Martn Llorente. "Extending the Cloud to the Network Edge." IEEE Computer 50, no. 4 (2017): 91-95.

[47] Povedano-Molina, Javier, et al. "DARGOS: A highly adaptable and scalable monitoring architecture for multi-tenant Clouds." Future Generation Computer Systems 29.8 (2013): 2041-2056.

[48] Patman, Jon,et al."Predictive analytics for fog computing using machine learning and GENI"IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops-INFOCOM WKSHPS.IEEE, 2018.

[49] Trihinas, Demetris, George Pallis, and Marios D. Dikaiakos. "Jcatascopia: Monitoring elastically adaptive applications in the cloud." Cluster, Cloud and Grid Computing (CCGrid), 2014

[50] Giannakopoulos, Ioannis, et al. "Celar: automated application elasticity platform." Big Data (Big Data), 2014 IEEE International Conference on.

[51] Taherizadeh, Salman,et al."A network edge monitoring approach for real-time data streaming applications"International Conference on Economics of Grids, Clouds, Systems, and Services. Springer, Cham, 2016.

[52] Katsaros, Gregory, et al. "A self-adaptive hierarchical monitoring mechanism for clouds"Journal of Systems and Software 85.5(2012):1029-1041

[53] da Cunha Rodrigues, Guilherme, et al. "The interplay between timeliness and scalability in cloud monitoring systems." Computers and Communication (ISCC), 2015 IEEE Symposium on. IEEE, 2015.

[54] Taherizadeh, Salman, and Vlado Stankovski. "Quality of Service Assurance for Internet of Things Time-Critical Cloud Applications: Experience with the Switch and Entice Projects." Advanced Applied Informatics (IIAI-AAI), 6th IIAI International Congress on. IEEE, 2017.

[55] da Cunha Rodrigues, Guilherme, et al. "Unfolding the Mutual Relation Between Timeliness and Scalability in Cloud Monitoring." 2018 IEEE Symposium on Computers and Communications (ISCC). IEEE, 2018.

[56] Patman, Jon, et al. "Data-Driven Edge Computing Resource Scheduling for Protest Crowds Incident Management." 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA).

[57] Clayman, Stuart, et al. "Monitoring services in a federated cloud: the reservoir experience." Achieving Federated and Self-Manageable Cloud Infrastructures: Theory and Practice. IGI Global, 2012. 242-265.

[58] Agelastos, Anthony, et al. "The Lightweight Distributed Metric Service: A Scalable Infrastructure for Continuous Monitoring of Large Scale Computing Systems and Applications."

[59] Taherizadeh, Salman, et al. "Runtime network-level monitoring framework in the adaptation of distributed time-critical cloud applications." Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2016.

[60] Lampe, Ulrich, Qiong Wu, Ronny Hans, Andr Miede, and Ralf Steinmetz. "To Frag or to Be Fragged-An Empirical Assessment of Latency in Cloud Gaming." In CLOSER, pp. 5-12. 2013.

[61] NASSAR, Almuthanna T .; YILMAZ, Yasin. Reinforcement-Learning-Based Resource Allocation in Fog Radio Access Networks for Various IoT Environments. arXiv preprint arXiv: 1806.04582 , 2018.

[62] Jawhar, Imad, Nader Mohamed, and Jameela Al-Jaroodi. "Networking architectures and protocols for smart city systems." Journal of Internet Services and Applications 9, no. 1 (2018): 26.

[63] Javadzadeh, G. and Rahmani, A.M., 2020. Fog Computing Applications in Smart Cities: A Systematic Survey. Wireless Networks, 26(2), pp.1433-1457.

[64] Naranjo, Paola G. Vinueza, Zahra Pooranian, Mohammad Shojafar, Mauro Conti, and Rajkumar Buyya. "FOCAN: A Fog-supported smart

city network architecture for management of applications in the Internet of Everything environments." Journal of Parallel and Distributed Computing 132 (2019): 274-283.

[65] Santos, J., Wauters, T., Volckaert, B. and De Turck, F., 2018. Fog computing: Enabling the management and orchestration of smart city applications in 5g networks. Entropy, 20(1), p.4.

[66] Basir, R., Qaisar, S., Ali, M., Aldwairi, M., Ashraf, M.I., Mahmood, A. and Gidlund, M., 2019. Fog Computing Enabling Industrial Internet of Things: State-of-the-Art and Research Challenges. Sensors, 19(21), p.4807.

[67] Aazam, M., Zeadally, S. and Harras, K.A., 2018. Deploying fog computing in industrial internet of things and industry 4.0. IEEE Transactions on Industrial Informatics, 14(10), pp.4674-4682.

[68] Chalapathi, G.S.S., Chamola, V., Vaish, A. and Buyya, R., 2019. Industrial Internet of Things (IIoT) Applications of Edge and Fog Computing: A Review and Future Directions. arXiv preprint arXiv:1912.00595.

[69] Andriopoulou, F., Dagiuklas, T. and Orphanoudakis, T., 2017. Integrating IoT and fog computing for healthcare service delivery. In Components and services for IoT platforms (pp. 213-232). Springer, Cham.

[70] Ulusar, U.D., Turk, E., Oztas, A.S., Savli, A.E., Ogunc, G. and Canpolat, M., 2019. IoT and Edge Computing as a Tool for Bowel Activity Monitoring. In Edge Computing (pp. 133-144). Springer, Cham.

[71] Akrivopoulos, O., Amaxilatis, D., Mavrommati, I. and Chatzigiannakis, I., 2019. Utilising fog computing for developing a person-centric heart monitoring system. Journal of Ambient Intelligence and Smart Environments, 11(3), pp.237-259.

[72] Gia, T.N. and Jiang, M., 2019. Exploiting fog computing in health monitoring. Fog and Edge Computing: Principles and Paradigms, pp.291-318.

[73] Negash, B., Gia, T.N., Anzanpour, A., Azimi, I., Jiang, M., Westerlund, T., Rahmani, A.M., Liljeberg, P. and Tenhunen, H., 2018. Leveraging fog computing for healthcare iot. In Fog computing in the internet of things (pp. 145-169). Springer, Cham.

[74] Paul, A., Pinjari, H., Hong, W.H., Seo, H.C. and Rho, S., 2018. Fog computing-based IoT for health monitoring system. Journal of Sensors, 2018.

[75] Dash, S., Biswas, S., Banerjee, D. and Rahman, A.U., 2019. Edge and Fog Computing in HealthcareA Review. Scalable Computing: Practice and Experience, 20(2), pp.191-206.

[76] GhobaeiArani, M., Souri, A., Safara, F. and Norouzi, M., 2020. An efficient task scheduling approach using mothflame optimization algorithm for cyberphysical system applications in fog computing. Transactions on Emerging Telecommunications Technologies, 31(2), p.e3770.

[77] Prasad, V.K., Bhavsar, M.D. and Tanwar, S., 2019. Influence of montoring: Fog and edge computing. Scalable Computing: Practice and Experience, 20(2), pp.365-376.

[78] Cao, H., Liu, S., Wu, L., Guan, Z. and Du, X., 2019. Achieving differential privacy against nonintrusive load monitoring in smart grid: A fog computing approach. Concurrency and Computation: Practice and Experience, 31(22), p.e4528.