2018 32nd International Conference on Advanced Information Networking and Applications Workshops

From the Cloud to Edge and IoT: a Smart Orchestration Architecture for Enabling Osmotic Computing

Lorenzo Carnevale *Student Member, IEEE* *, Antonio Celesti, *Member, IEEE* *§,
Antonino Galletta, *Student Member, IEEE* *, Schahram Dustdar, *Fellow, IEEE* ‡ and
Massimo Villari, *Member, IEEE* *§

* Department of Mathematics and Computer Science, Physics and Earth Sciences, University of Messina,
Viale F. Stagno D'Alcontres 31, I-98166 Messina, Italy - {lcarnevale, acelesti, angalletta, mvillari}@unime.it
‡ Distributed Systems Group TU Wien, Argentinierstrasse 8, A-1040, Vienna, Austria - dustdar@infosys.tuwien.ac.at
§ on behalf of GNCS — Gruppo Nazionale per il Calcolo Scientifico - INdAM,
P.le Aldo Moro 5, 00185, Roma, Italy

***Abstract*—The latest technological and conceptual developments have destroyed the centralized Cloud Computing model, moving Cloud services in emerging ICT infrastructures such as Edge, Fog and Internet of Things (IoT) that are closer to end users. Specifically, current Cloud computing programming models and resource orchestration techniques are challenged by the recent evolution of the IoT phenomenon because smart devices are becoming more and more pervasive, powerful and inexpensive. Therefore, services need to be place near such devices. In this regard, the Osmotic Computing aims to provide a new computing paradigm based on the deployment and migration strategies related to the infrastructures and applications requirements across Cloud, Edge, Fog an IoT layers. In this scientific paper, we investigate the Smart Orchestration of a new software abstraction called MicroELement (MEL), that encapsulates resources, services and data necessary to run IoT applications. Several use cases are presented for describing the Artificial Intelligence processes that enables the MELs deployment.**

***Index Terms*—Osmotic Computing, Edge Computing, Cloud Computing, IoT, Orchestration, Elasticity, Artificial Intelligence.**

I. Introduction

In the last ten years, we observed an unstoppable growth of complementary technologies, such as Cloud Computing, Big Data and Internet of Things (IoT). The latter has increased the connected devices number up to an estimate of 36 billion in 2021[1], thanks to the miniaturization and low cost of these devices. Among these, almost a billion will be the wearable[2], whereas about 3 billion will be the smartphone users[3]. Therefore, the amount of global IP data traffic in the same year is estimated at 280,000 petabytes per month[4] with the inevitable need to handle huge data amounts from both migratory and computational point of view. Thus, the IoT model has become both a Big Data and a computation problem for each Cloud, Fog and Edge layer.

On the other hand, 2017 saw these three technologies converge towards a single point of contact, called Osmotic Computing. It aims to identify, design and implement a paradigm for managing data, resources and processes' software across IoT, Edge and Cloud systems, satisfying the end users' Quality of Service (QoS). The Cloud-Edge Computing integration benefits have also been acknowledged by academic and industry initiatives, such as Cisco, Amazon AWS and the Open Fog Consortium.

Purpose of this scientific work is to present a first version of the Osmotic Computing architecture, focusing just on the microservices deployment aspect. Indeed, the design of the Osmotic Smart Orchestrator has been investigated, presenting different enabling technologies, use cases related to Osmotic component registration, orchestration training and prediction and microservices deplyment. Therefore, the core of the orchestration process will be an Artificial Intelligence (AI) module that will learn through the monitoring of the Osmotic resources deployed on CLoud, Edge and/or IoT.

The rest of the paper is organized as follows. The Section II summarizes the works related to the Osmotic Computing and similar methodologies. A MicroELement graph is described in the Section III, whereas a complete description of the Osmotic platform is provided in the Section IV. The smart orchestrator architecture overview is presented in the Section V, addressing the content toward in-depth descriptions. Finally, conclusion and lights for the future are presented in the Section VI.

II. Related Work

Osmotic computing was introduced in 2016 as a new promising paradigm for the integration between a centralised Cloud layer and Edge/Internet of Things (IoT) layers [1], whereas its basic principles and enabling technologies were

[1]https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/
[2]https://www.statista.com/statistics/487291/global-connected-wearable-devices/
[3]https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/
[4]https://www.statista.com/statistics/499431/global-ip-data-traffic-forecast/

978-1-5386-5395-1/18/$31.00 ©2018 IEEE
DOI 10.1109/WAINA.2018.00122

presented in [2]. Such a new paradigm can be used in different application scenarios requiring an intensive interaction between centralised Cloud systems and edge devices. In [3] it was considered to design a Hospital Information System (HIS) interconnecting medical devices and patients' personal body networks with Hospital Cloud systems. Another recent application fields regarded the efficient trust management in pervasive online social networks [4] and the management of IoT workflows [5].

One of the major issue of Osmotic computing is the service orchestration management considering hybrid Cloud/Edge/IoT systems. Up to now, service orchestration has been a topic of discussion in Cloud computing [6], [7], [8], Edge/Fog computing [9], [10], [11] and IoT [12] considered independently. Regarding service orchestration in Cloud computing many initiatives regarded the control of the network layers. Architecture for the dynamic management of end-to-end connections in a Cloud environment considering Software Defined Networking (SDN) technologies were investigated in [13], [14], [10], [15], and [16]. An End-to-end SDN/NFV orchestration for video analytics using edge and Cloud computing over programmable optical networks is described in [17]. A piece of framework and novel computing models for grid and cloud service orchestration aimed at supporting scientists and business analysts at large scale was discussed in [18]. Dynamic resource orchestration for multi-task applications in heterogeneous mobile Cloud computing in [19]. Here, the resource orchestration was formulated as multi-objective optimal problem considering energy consumption, cost and availability metrics. A Distributed pieces of framework for cloud computing orchestration able to manage the migration of Virtual Machines (VMs) were discussed in [20]. Platforms able to control high performance scientific workflow by means of computing Cloud applications were discussed in [21], [22] and [23]. An orchestration engine based on a temporal reconfiguration approach that partitions the amount resources of cloud servers proportionally between BPEL processes applying a temporal partitioning algorithm was discussed [24]. A SLA (service level agreement) driven orchestration based methodology for cloud computing services was presented in [25]. The problem of security and privacy governance in cloud computing via SLAs and a policy orchestration service was investigated in [26]. The state of the art of container-based orchestration in Cloud even considering future challenges was discussed in [27].

Regarding service orchestration in IoT, the distributed orchestration in large-scale systems was discussed in [28], whereas, the requirements of a semantic based service orchestration were investigated in [29]. Intent-based management and orchestration of heterogeneous openflow/IoT SDN domains was discussed in [30], whereas testbed set-up for SDN orchestration across network cloud and IoT domains was presented in [31]. The opportunities of applying an orchestration model in cognitive IoT solutions interconnecting instrumented worlds were discussed in [32]. An object-oriented model for object orchestration in smart environments able to extend and customise smartphones was described in [33]. A model for trustworthy orchestration in the IoT using a public/subscribe approach and MQTT was presented in [34]. A scalable piece of framework for the provisioning large-scale IoT deployments was discussed in [35]. The orchestration in distributed web-of-objects for creation of user-centered IoT services was discussed in [36], [37] and [38].

Service orchestration in Edge/Fog computing is a quite new topic. The deployment orchestration of microservices with geographical constraints using OpenStack Heat component was discussed in [39]. Whereas, and end-to-end SDN/NFV orchestration for video analytics using edge and cloud computing over programmable optical networks was discussed in [17]. A service orchestration architecture for Fog-enabled infrastructures was presented in [40]. An adaptive orchestration platform called ECHO for hybrid dataflows across cloud and edge was described in [41]. In particular, the ECHO s hybrid dataflow composition is able to operate on diverse data models such as streams, micro-batches and files, and interface with native runtime engines like TensorFlow and Storm to execute them. An architecture able to move IoT application on Edge computing layers was described in [42]. A SDN/NFV orchestration of 5G services in hybrid Cloud/Fog multi-domain networks was discussed in [43]. A preliminary discussion about the use of container virtualization to orchestrate Edge and Fog computing enviroments was discussed in [44].

Differently from the aforementioned scientific works, in this paper we focus on Osmotic computing service orchestration considering the management of microservices from the Cloud to Edge/Fog and IoT enviroments.

III. Osmotic MELS

The convergence between Cloud Computing, Edge, and IoT requires an Osmotic management of resources, services, and data, whose elements can move across different heterogeneous infrastructures. More specifically, referring to Figure 1, IoT applications deployed in distributed environments may be viewed as a graph of MicroELementS (MELS), composed of:

- MicroServices (MS) for implementing specific functionalities, which can be deployed and migrated across the virtualized infrastructures;
- MicroData (MD) for representing information flows from/to IoT devices, which can be in different data formats.

the MELS graph needs to be orchestrated across Cloud, Edge, Fog, and IoT according to specific QoS requirements. Let us remark that MELS are not physical resources, but represent software and data abstractions. According to the Figure 1, the leaf node is represented by MicroUserService (MUS, i.e., an IoT application) and MicroOperationalService (MOS, i.e., an Operating System) along with MicroUserData (MUD, i.e., User Data) and MicroOperationalData (MOD, i.e., MS configuration). The MELS are smartly deployed on Cloud and Edge in virtual components, such as lightweight containers (e.g., Docker, Google Container, Amazon Compute Cloud Container, etc.); whereas uPython-VM, uLUA-VM, Javascript

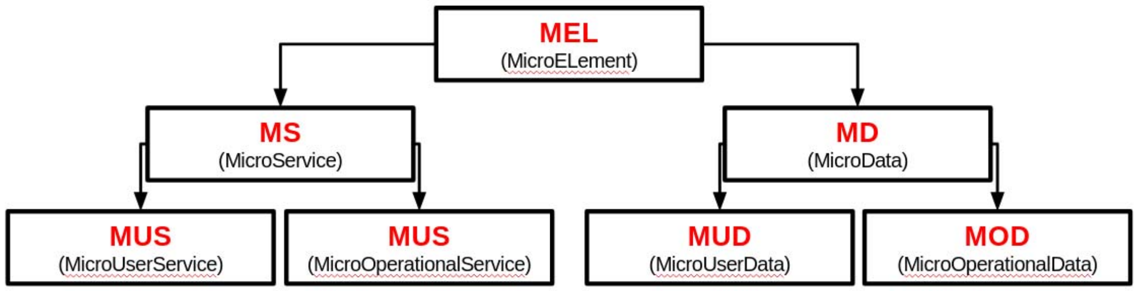

Figure 1. MicroElementS hierarchy.

on IoT have emerged as a lightweight alternative to the hypervisor-based approach used in the Cloud Data Center. A container encapsulates only well-defined software components (e.g., database servers), reducing the deployment overhead and increasing instance density on a single device. For instance, an example of MS could be a Docker Container and an MD contains JSON-based metadata. Moreover, MD can be both passive data (can be read or updated on devices) and active (can be queried), e.g., MD stored in NoSQL DBs as MongoDB, Cassandra, etc.

IV. Osmotic Computing Platform

Borrowing the term from chemistry, "osmosis" represents the MELS spread across the Cloud Data Center (CDC) and the Edge micro Data Center (EmDC). The Osmotic Computing overcomes the MELS elastic management concept, since the deployment and migration strategies are related to the requirements of both infrastructures (i.e., load balancing, reliability, availability) and applications (i.e., detection, implementation, awareness of the context, proximity, QoS). Moreover, in order to overcome the heterogeneity of IoT resources, the MEL abstraction allows us to support a virtual environment that can be adapted according to the available hardware equipment.

Looking at Figure 2, the Layer 3 (L3), the one closest to end users and/or physical entities, shows how MELS are deployed in embedded devices. L3 IoT devices communicate according to standardized protocols, such as CoAP (Constrained Application Protocol), supported by the RESTful interface.

Furthermore, Figure 2 shows MELS (at L2) deployed in different embedded devices (i.e., IoT Gateway such as Raspberry Pi 3). Gateway nodes perform operations (mean, min, max, filtering, aggregation, etc.) on the data flow acquired from L3. More often, these devices acquire data with a predefined frequency, depending on specific system requirements and the device ability to gather data.

In Figure 2, a more complex computational and storage capability is available to MELS at L1, allowing to perform simulations and/or analyses on data.

The Cloud, Edge and IoT infrastructure also has its own target function which influences the performed operations. For example, Edge (L2 in the Figure 2) generally includes devices with limited resources (i.e., limited battery power, network range, etc.) which must perform operations by these constraints. Therefore, the storage and compute capacity in the Edge must be shared among multiple concurrent data streams (probably from L3 in Figure 2), limiting the analysis to the streams number and time constraints. Cloud operations (L1 in the Figure 2) are based on pre-agreed goals between a client and a data center provider, such as throughput, response times, costs, etc. It is a key research challenge for real-time streaming applications understanding how an application hosted on a Cloud in L1 can interact and coordinate with the IoT (in L3) and the Edge (in L2). Driven by QoS requirements, the MELS can be distributed among Cloud, Edge and IoT. The distribution of data analysis through these different infrastructures can improve the overall performance of the IoT application and reduce the load on the main network.

Orchestration in the Osmotic environment smartly configures the movement and deployment of MELS in response to QoS, security/privacy requirements and runtime requests. Indeed, static and fixed approaches are not able to provide IoT solutions. The Osmotic Computing aims to abstract the services (MELS) and the infrastructure (IoT, Cloud, Edge) to decouple the applications from the hardware and enable the possibility to flow the services from Cloud to Edge and/or IoT and vice versa.

For example, consider a situation in which there are many IoT devices, which in particular situations are collecting large volumes of data on L3. Furthermore, consider that given the stability and capacity of the network, the amount of data produced and their subsequent transmission to a Cloud (L1) are unsustainable from the network point of view. If these data were to be analyzed in the Cloud, this would be unrealistic and the current system may not be able to continue. Using the Osmotic approach, when a bottleneck of this type is detected, a Smart Orchestrator moves the processing of some data to the Edge (L2).

V. Architecture Overview

In this paper, we designed a multi-tenant Osmotic-based architecture able to carry out a workflow that drives MELS registration, migration and computation requests toward a learning process, according to the MELS monitor. The whole process requires the interaction of different actors, as shown in Figure 3. More specifically, after the device requirements and constraints inserting in the platform, the Osmotic architecture

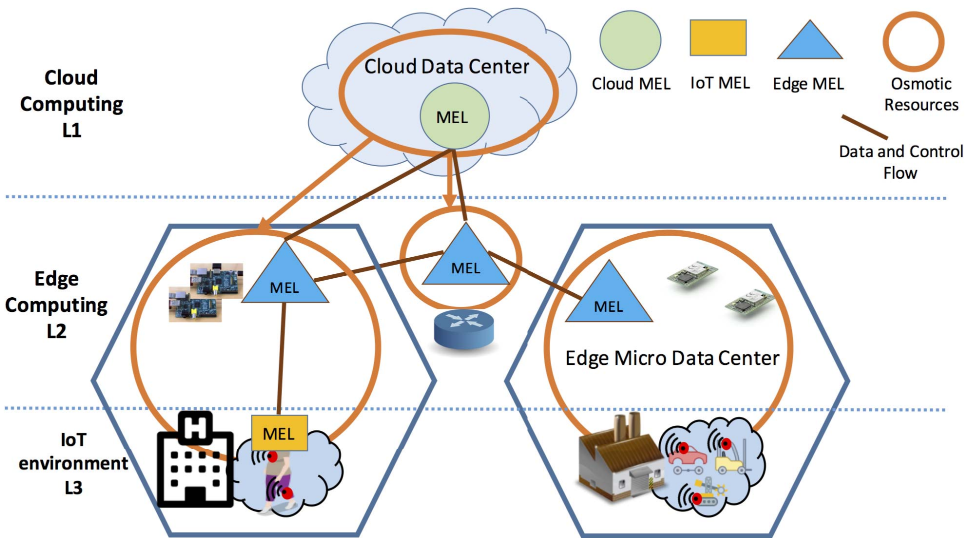

Figure 2. Osmotic computing scenario.

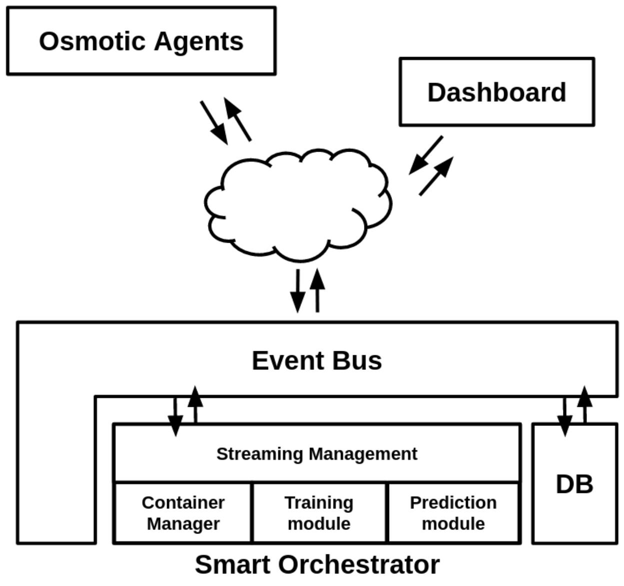

Figure 3. Logical architecture of the Osmotic Computing Smart Orchestration.

translates them into MEL. Then, the platform will create the necessary MELS that will be generated on the Cloud, Edge and IoT devices. Continuous monitoring of resources at all levels of implementation will result in a feedback-driven orchestration in which different MELS may be distributed over time.

A. How to Register the Osmotic Resources?

The idea behind the utilization of the Osmotic environment is the freedom to add own IoT, Edge and/or Cloud resources to the platform. Each user must be able to manually select an own resource through a user friendly web dashboard. The latter follows the registration process, asking unique identifiers, such as IP address, MAC address, geolocation and/or TAGs. Clearly, the resource will be registered according to the specific tenant, ensuring the authentication and authorization process.

Therefore, the registration is processed by the infrastructure backend in order to store the request into a NoSQL database and mark the resource. Indeed, the request is migrated to an Event Bus, as shown in Figure 4, before to be managed by the Container Manager, a software able to deploy the MELS.

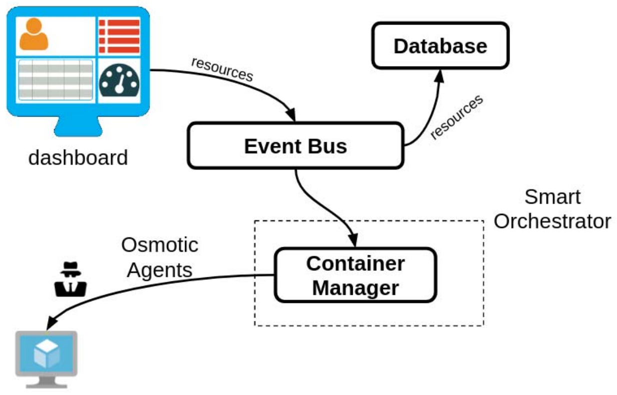

Figure 4. Use case of the Osmotic registration process.

Thus, a lightweight and functional Osmotic Agent (MEL) will be injected into the resource according to the device type, such as microprocessor (MPU) or microcontroller (MCU) IoT devices, virtual resource with limited or not limited capabilities or physical machines.

However, each Osmotic Agent will have common functional requirements. Currently, an Osmotic Agent is thought to be a virtualenv- or a container-like application, whose main task is to enable communication from/to the Osmotic Architecture.

Furthermore, the Osmotic Agent will be able to receive the injected MEL and send a resource activities monitoring.

B. How to Train the Osmotic Orchestrator?

The heart of the Smart Orchestrator is the training module shown in Figure 3. Target of the Deep Learning process is to train a predictive model able to create a MELS deployment manifest based on the previous experience gathered through the Osmotic Agents monitoring.

Indeed, referring to the figure 5, the Osmotic Agents send the monitoring data to the event bus in order to forward them to the Streaming Management. This is a component used for creating a small batch of the real time data received and facilitating the training process in terms of time and performance.

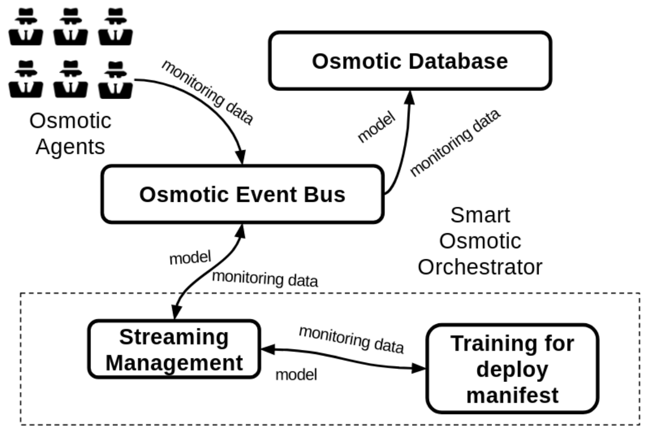

Figure 5. Use case of the smart orchestration training phase.

Finally, this module will have the task of running the training algorithm and returning a model to be saved on the database. From the Deep Learning algorithm point of view, we have not yet investigated it due to the dependence on the deployed MELS deploying technology.

C. How to Predict the Osmotic Deploy Manifest?

As done for model training, the prediction phase uses similar components. In this use case, it is interesting to note that users can submit an execution request directly from the dashboard, by selecting the MELS they want to use in the form of MS and MD. This request is taken over by the Event Bus and forwarded to the streaming management. In the end, the prediction module uses the previously saved model to predict a manifest useful for deploying MELS on different IoT, Edge, Fog and/or Cloud levels.

D. How to Osmotically Deploy?

The last step in the MELS orchestration Osmotic process is the execution of the manifest previously created. In this regard, the container manager uploads the MELS to the aforementioned resources through its MS and MD components. Then, the Osmotic Agent will install and start the received component.

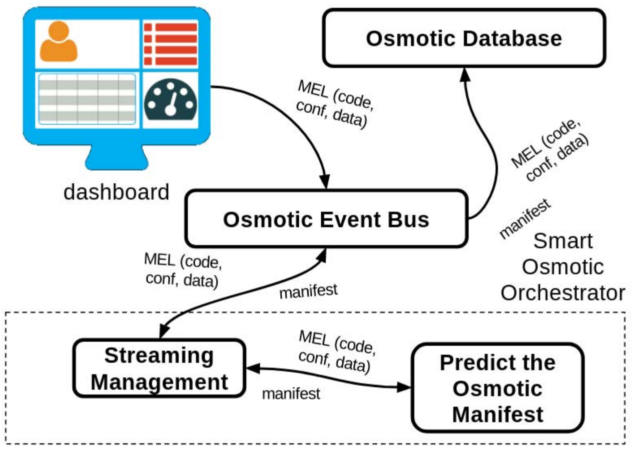

Figure 6. Use case of the smart orchestration prediction phase.

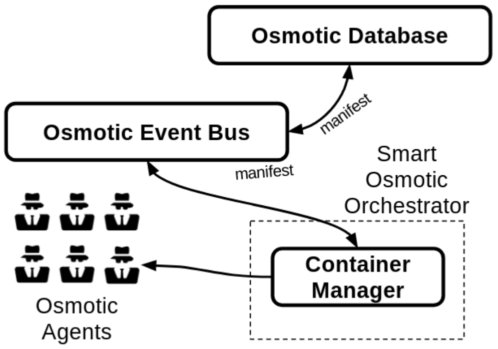

Figure 7. Use case of the MELS deploying process

VI. Conclusion

The convergence between Cloud Computing, Edge, and IoT requires the management of resources, services, and data, whose elements can move across different heterogeneous infrastructures. In this paper, in order to solve this problem, we presented an orchestration architecture based on the new emerging Osmotic computing paradigm. In particular, starting from the analysis of IoT applications deployed in distributed environments as a graph of MicroELementS (MELS), we discussed how their back-end systems can be deployed across Cloud and Edge layers considering the Osmotic computing paradigm. In the end, a multi-tenant Osmotic-based architecture able to carry out a workflow that drives MELS registration, migration and computation requests toward a learning process and according to the MELS monitor was described. In future works, we plan to deeply analyse the Artificial Intelligence algorithms that enable the MELs deployment.

Acknowledgment

This work has been supported by Cloud for Europe (C4E) Tender: *REALIZATION OF A RESEARCH AND DEVELOPMENT PROJECT (PRE-COMMERCIAL PROCUREMENT) ON "CLOUD FOR EUROPE"*, Italy-Rome: Research and development services and related consultancy services Contract notice: 2014/S 241-424518. Directive: 2004/18/EC. (http://www.cloudforeurope.eu/).

References

[1] M. Villari, M. Fazio, S. Dustdar, O. Rana, and R. Ranjan, "Osmotic computing: A new paradigm for edge/cloud integration," *IEEE Cloud Computing*, vol. 3, no. 6, pp. 76–83, 2016.

[2] M. Villari, A. Celesti, and M. Fazio, "Towards osmotic computing: Looking at basic principles and technologies," *Advances in Intelligent Systems and Computing*, vol. 611, pp. 906–915, 2018.

[3] L. Carnevale, A. Galletta, A. Celesti, M. Fazio, M. Paone, P. Bramanti, and M. Villari, "Big data his of the irccs-me future: The osmotic computing infrastructure," 2018, vol. 189, pp. 199–207.

[4] V. Sharma, I. You, R. Kumar, and P. Kim, "Computational offloading for efficient trust management in pervasive online social networks using osmotic computing," *IEEE Access*, vol. 5, pp. 5084–5103, 2017.

[5] M. Nardelli, S. Nastic, S. Dustdar, M. Villari, and R. Ranjan, "Osmotic flow: Osmotic computing + iot workflow," 2017, vol. 4, no. 2, pp. 68–75.

[6] F. Mrmol and M. Kuhnen, "Reputation-based web service orchestration in cloud computing: A survey," *Concurrency Computation*, vol. 27, no. 9, pp. 2390–2412, 2015.

[7] R. Ranjan, R. Buyya, S. Nepal, and D. Georgakopulos, "A note on resource orchestration for cloud computing," *Concurrency Computation*, vol. 27, no. 9, pp. 2370–2372, 2015.

[8] M. Katyal and A. Mishra, "Orchestration of cloud computing virtual resources," 2014, pp. 833–838.

[9] Y. Jiang, Z. Huang, and D. Tsang, "Challenges and solutions in fog computing orchestration," *IEEE Network*, 2017.

[10] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[11] Z. Wen, R. Yang, P. Garraghan, T. Lin, J. Xu, and M. Rovatsos, "Fog orchestration for internet of things services," *IEEE Internet Computing*, vol. 21, no. 2, pp. 16–24, 2017.

[12] C. Consel and M. Kabac, "Internet of things: From small-to large-scale orchestration," 2017, pp. 1748–1755.

[13] A. Mayoral, R. Vilalta, R. Muoz, R. Casellas, and R. Martnez, "Sdn orchestration architectures and their integration with cloud computing applications," *Optical Switching and Networking*, vol. 26, pp. 2–13, 2017.

[14] R. Bonafiglia, G. Castellano, I. Cerrato, and F. Risso, "End-to-end service orchestration across sdn and cloud computing domains," 2017.

[15] Y. Kim, S. Kang, C. Cho, and S. Pahk, "Sdn-based orchestration for interworking cloud and transport networks," 2016, pp. 303–307.

[16] A. Mayoral, R. Vilalta, R. Muoz, R. Casellas, and R. Martnez, "Performance analysis of sdn orchestration in the cloud computing platform and transport network of the adrenaline testbed," vol. 2015-August, 2015.

[17] R. Vilalta, I. Popescu, A. Mayoral, X. Cao, R. Casellas, N. Yoshikane, R. Martnez, T. Tsuritani, I. Morita, and R. Muoz, "End-to-end sdn/nfv orchestration of video analytics using edge and cloud computing over programmable optical networks," 2017.

[18] F. Messina, R. Mikkilineni, and G. Morana, "Middleware, framework and novel computing models for grid and cloud service orchestration," *International Journal of Grid and Utility Computing*, vol. 8, no. 2, pp. 71–73, 2017.

[19] Q. Qi, J. Liao, J. Wang, Q. Li, and Y. Cao, "Dynamic resource orchestration for multi-task application in heterogeneous mobile cloud computing," vol. 2016-September, 2016, pp. 221–226.

[20] R. Weingartner, G. Brascher, and C. Westphall, "A distributed autonomic management framework for cloud computing orchestration," 2016, pp. 9–17.

[21] J. de Carvalho Silva and F. de Carvalho Junior, "A platform of scientific workflows for orchestration of parallel components in a cloud of high performance computing applications," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9889 LNCS, pp. 156–170, 2016.

[22] L. Miroslaw, M. Pantic, and H. Nordborg, "Unified cloud orchestration framework for elastic high performance computing in the cloud," 2016, pp. 291–298.

[23] K. Shams, M. Powell., T. Crockett, J. Norris, R. Rossi, and T. Soderstrom, "Polyphony: A workflow orchestration framework for cloud computing," 2010, pp. 606–611.

[24] Z. Brahmi and C. Gharbi, "Temporal reconfiguration-based orchestration engine in the cloud computing," *Lecture Notes in Business Information Processing*, vol. 176 LNBIP, pp. 73–85, 2014.

[25] M. Irfan, Z. Hong, N. Aimaier, and L. Zhuguo, "Sla (service level agreement) driven orchestration based new methodology for cloud computing services," *Advanced Materials Research*, vol. 660, pp. 196–200, 2013.

[26] M. Mont, K. McCorry, N. Papanikolaou, and S. Pearson, "Security and privacy governance in cloud computing via slas and a policy orchestration service," 2012, pp. 670–674.

[27] A. Tosatto, P. Ruiu, and A. Attanasio, "Container-based orchestration in cloud: State of the art and challenges," 2015, pp. 70–75.

[28] E. Yigitoglu, L. Liu, M. Looper, and C. Pu, "Distributed orchestration in large-scale iot systems," 2017, pp. 58–65.

[29] E. Chindenga, M. Scott, and C. Gurajena, "Semantics based service orchestration in iot," vol. Part F130806, 2017.

[30] W. Cerroni, C. Buratti, S. Cerboni, G. Davoli, C. Contoli, F. Foresta, F. Callegati, and R. Verdone, "Intent-based management and orchestration of heterogeneous openflow/iot sdn domains," 2017.

[31] S. Fichera, M. Gharbaoui, P. Castoldi, B. Martini, and A. Manzalini, "On experimenting 5g: Testbed set-up for sdn orchestration across network cloud and iot domains," 2017.

[32] C. Li, F. Darema, and V. Chang, "Distributed behavior model orchestration in cognitive internet of things solution," *Enterprise Information Systems*, pp. 1–21, 2017.

[33] L. Bergesio, A. Bernardos, and J. Casar, "An object-oriented model for object orchestration in smart environments," vol. 109, 2017, pp. 440–447.

[34] M. Bottone, G. Primiero, F. Raimondi, and V. De Florio, "A model for trustworthy orchestration in the internet of things," 2016, pp. 171–174.

[35] M. Vögler, J. M. Schleicher, C. Inzinger, and S. Dustdar, "A scalable framework for provisioning large-scale iot deployments," *ACM Trans. Internet Technol.*, vol. 16, no. 2, pp. 11:1–11:20, Mar. 2016.

[36] N. Lee, H. Lee, W. Ryu, and K. Heo, "Web of object service architecture for device orchestration and composition," 2014.

[37] Y. Kim, S. Lee, Y. Jeon, I. Chong, and S. Lee, "Orchestration in distributed web-of-objects for creation of user-centered iot service capability," 2013, pp. 750–755.

[38] Y. Kim, S. Lee, and I. Chong, "Orchestration in distributed web-of-objects for creation of user-centered iot service capability," *Wireless Personal Communications*, vol. 78, no. 4, pp. 1965–1980, 2014.

[39] M. Villari, A. Celesti, G. Tricomi, A. Galletta, and M. Fazio, "Deployment orchestration of microservices with geographical constraints for edge computing," 2017, pp. 633–638.

[40] M. De Brito, S. Hoque, T. Magedanz, R. Steinke, A. Willner, D. Nehls, O. Keils, and F. Schreiner, "A service orchestration architecture for fog-enabled infrastructures," 2017, pp. 127–132.

[41] P. Ravindra, A. Khochare, S. Reddy, S. Sharma, P. Varshney, and Y. Simmhan, "Echo: An adaptive orchestration platform for hybrid dataflows across cloud and edge," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10601 LNCS, pp. 395–410, 2017.

[42] A. Hegyi, H. Flinck, I. Ketyko, P. Kuure, C. Nemes, and L. Pinter, "Application orchestration in mobile edge cloud : Placing of iot applications to the edge," 2016, pp. 230–235.

[43] R. Vilalta, A. Mayoral, R. Casellas, R. Martinez, and R. Munoz, "Sdn/nfv orchestration of multi-technology and multi-domain networks in cloud/fog architectures for 5g services," 2016.

[44] S. Hoque, M. Brito, A. Willner, O. Keil, and T. Magedanz, "Towards container orchestration in fog computing infrastructures," vol. 2, 2017, pp. 294–299.