Fog Function Virtualization: a Flexible Solution for IoT Applications

Damian Roca, Josue V. Quiroga
and Mateo Valero
Barcelona Supercomputing Center (BSC-CNS) and
Universitat Politècnica de Catalunya (UPC)
Barcelona, Spain
Email: {name}.{surname}@bsc.es

Mario Nemirovsky
ICREA Senior Research Professor at BSC-CNS
Barcelona, Spain
Email: mario.nemirovsky@bsc.es

***Abstract*—The Internet of Things applications must carefully assess certain crucial factors such as the real-time and largely distributed nature of the "things". Fog Computing provides an architecture to satisfy those requirements through nodes located from near the "things" till the edge. The problem comes with the integration of the Fog nodes into current infrastructures. This process requires the development of complex software solutions and prevents Fog growth. In this paper we propose three innovations to enhance Fog: (i) a new orchestration policy, (ii) the creation of constellations of nodes, and (iii) Fog Function Virtualization (FFV). All together will complement Fog to reach its true potential as a generic scalable platform, running multiple IoT applications simultaneously. Deploying a new service is reduced to the development of the application code, fact that brings the democratization of the Fog Computing paradigm through ease of deployment and cost reduction.**

I. Introduction

A new paradigm that it is still to explode is the Internet of Things (IoT). Based on estimations, in 2020 there will be around 50 billion devices connected to the Internet [1]. These "things" are in their majority sensors and actuators interacting with the real world. Hand by hand with these devices, a huge amount of data is collected, requiring a process of analysis and actuation. These applications must carefully assess certain crucial factors such as the real-time and largely distributed nature of the "things", maintaining trustworthy communications, and adapting for mobility and the harsh environments where the "things" are deployed. To achieve these objectives with traditional Cloud Computing solutions is complicated since a centric design approach precludes critical IoT requirements, such as real-time and geographically-aware computing.

Fog Computing [2], an architecture that appeared in the past years, can be used as a base to provide a good solution to the IoT requirements [3]. It is a highly distributed platform, with nodes located from near the end-user devices till the edge of the network. These nodes offer resources such as computing, storage, and networking to the applications operating under this infrastructure. An access point with enhanced computing capabilities constitutes an example of a possible Fog-capable node. Fog processes the data close to where its generated, reducing the network utilization and improving the aggregation from the bottom of the infrastructure [4]. Low-latency, widespread geographic distribution, heterogeneity, and mobility are part of its main advantages. In consequence, Fog becomes an extension of the Cloud rather than a substitute since its nodes are connected to the Cloud.

The true potential of Fog Computing lies in the implementation of a generic multi-tenant platform supporting a wide rage of applications simultaneously [5]. This approach reduces deployment costs, eliminates hardware redundancy, and improves the scalability of the system. However, current IoT deployments are based on "things" covering an area with a set of proprietary nodes connected to the Internet. As a consequence, each application constitutes a subsystem or silo [6] inside IoT and there is no exchange of information among applications that users could benefit from. Even though a layer of Fog nodes is available, applications have to develop complex software solutions to integrate those nodes into their infrastructures or face vertical deployments from the "things" to the Cloud. Then, a paradigm shift as shown in Figure 1 is required to enable a generic IoT infrastructure.

For Fog Computing to grow as a platform and reach that vision, it should adopt the ease of deployment from on-demand platforms such as Cloud and the flexibility from software defined technologies such as SDN. To accomplish these objectives, we propose to enhance Fog with three key innovations. First, a new orchestration policy to provide more flexibility to the infrastructure breaking the execution in the

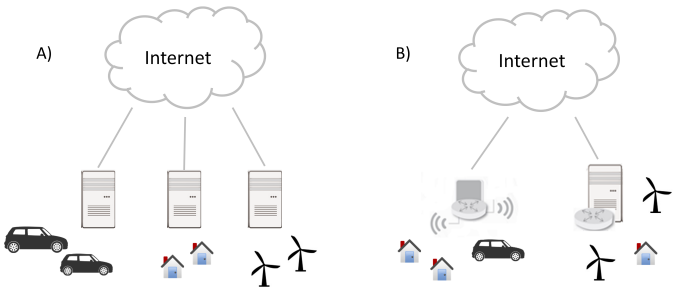

Fig. 1. Moving from a silo-based implementation where each application has its own infrastructure represented in (A) to a generic Fog deployment capable of executing several applications simultaneously represented in (B)

 DOI: 10.1109/FMEC.2017.7946411

Cloud by default. Second, the creation of constellations of heterogeneous Fog nodes to aggregate their capabilities, increasing the available resources at the bottom of the hierarchy. Last, the definition of the Fog Function Virtualization (FFV) concept. These functions cover aspects such as analytics, sensors' functionalities, and computational resources among others. Hence, the system exposes its capabilities through these functions without jeopardizing the complexity nor the scalability.

The combination of these three techniques contributes to the democratization of the IoT services by truly enabling a generic infrastructure to run multiple applications simultaneously. The deployment of a new service only requires the application code without worrying about the node integration, since virtualization techniques hide the entire infrastructure from "things" to Cloud.

In this paper we first describe the Fog architecture, its functionalities and the desired requirements in Section II. We then elaborate the three innovations proposed in Section III. Lastly, Section IV illustrates its usefulness in dealing with different IoT deployments through a two scenario case study.

II. Fog Computing Architecture

A representative architecture of a generic IoT infrastructure is depicted in Figure 2. At the lower levels there are "things", responsible of gathering information. The next layer is formed by heterogeneous Fog nodes, which constitute the aggregation points. The "things" and the nodes communicate mostly through wireless technologies, since both "things" and nodes can move. In addition to this vertical communication, Fog nodes can communicate horizontally (i.e. between two Fog nodes at the same hierarchical level). Due to Fog nodes' wide geographic deployment and their locality, they can offer real-time resources processing the data close to where it is generated. These characteristics enable most of the Fog Computing advantages (i.e. mobility support, low-latency, etc.). Till reaching the Cloud, Fog nodes form an interconnected hierarchy. Among these nodes there might be non-compatible Fog devices operating normally. The Cloud constitutes the last layer, offering a large pool of resources at low-cost but without any latency guarantees.

The decision on which layer executes the application depends on its requirements although the final decision corresponds to the orchestrator. The original Fog architecture defined that applications run on the Cloud by default and only those which strictly require Fog capabilities use the Fog layers. Once the Fog layers are chosen, other parameters such as the node's visibility serve to decide which Fog nodes execute it. Exploiting the visibility is important for applications covering a wide geographic area, since aggregation can take place at higher nodes. The resultant advantage is a traffic reduction since just the strictly necessary data is sent from one level to another.

Now, the objective turns to optimize the applications' execution. To this end, both Cloud and Fog use virtualization techniques. They rely on virtual machines to offer security, isolation, dynamism, and ease of management [7]. Recently, the "things" themselves are offered as a service too [8] [9]. For example, a company deploy a set of complex sensors measuring different events. They can offer virtual instances of these sensors to other companies fulfilling the requirements of a new application. In consequence, companies have at their disposal the entire infrastructure as a service.

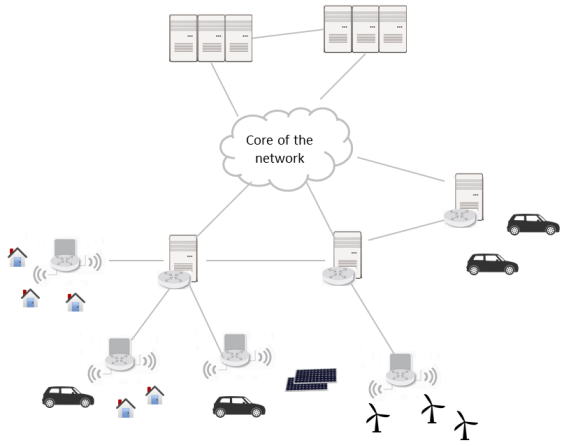

Fig. 2. Illustrative example of a generic Fog-based infrastructure serving multiple IoT applications. Fog nodes are interconnected forming a hierarchy.

Security and privacy pose many challenges that delay the IoT explosion [10]. While silos provide some natural protection mechanisms due to its isolation, a multi-layer infrastructure augments the attack surface. In addition, Fog operating as a generic infrastructure poses new threats such as side channel and resource exhaustion attacks. For instance, an attacker could access the "things" themselves and make an IoT device to malfunction. In certain cases such as a pacemaker this is critical.

Furthermore, people are reluctant to expose personal data in the Internet arising from a wide range of IoT devices. And aggravating the problem, different legislations apply and there is no consensus on security or privacy standards. In any case the protocols implemented cannot jeopardize the real-time characteristic and should be able to run on the low-power simple devices in which IoT relies on.

A. Implementation requirements

We have seen the basics of Fog architecture and how it deals with the new requirements brought by IoT. The hardware platform required to deploy new applications is available, but not so with the software. To integrate these Fog devices into a current infrastructure requires a vast effort to develop the algorithms that manage the execution through the different layers. This fact has prevented the explosion of Fog-based deployments. Then, the pending question is: What are the desired characteristics from the application's perspective to make Fog an attractive and successful solution?

The system should present a great flexibility to offer its hardware resources. Regarding this aspect, the on-demand based Cloud solutions have proven themselves as an optimal technique [7]. New users can ask for instances to start running their applications within minutes. The same solution can be applied to Fog. Now, these instances cover the Fog nodes that bring extra complexity. The nodes can belong to different companies, they can have different capabilities, and they are distributed geographically among others. Hong et al. used the assumption of ideal instances to define their API [11]. However, they exposed the entire infrastructure to the application developers, aggravating the software problem. Since mobility is an IoT pillar, it is really difficult to anticipate which devices are in a certain area to program them in advance.

To make a successful IoT platform, applications would like those instances to be totally transparent to them. They do not want to know the nodes responsible of executing their code or who owns them as long as their requirements are satisfied. Among these requirements security is critical. If security is weak, there is a potential lack of control over the data. Additionally, users do not expose sensitive data without certain guarantees that today are not fulfilled. In consequence, IoT applications will not achieve the market estimations unless these issues are solved.

Another critical aspect in these instances is the connectivity between the different layers. First, applications use a wide range of protocols to communicate "things" with Fog nodes. However, this diversity may suppose a problem if different hardware is required. For instance, a Fog node could execute two applications, one using Bluetooth and the other using WiFi. In this specific case, that node has hardware support for both standards. Once the information is in the Fog layers, the network among these nodes is not uniform and this fact may affect the application requirements. Hence, applications need adaptivity and transparency regarding the interconnection system, having the generic platform for IoT as the final objective.

The aforementioned requirements deal with a key aspect, the ease of deployment. If new applications just require an instance to start running their code without the concern of managing the entire infrastructure, Fog will become the "de facto" IoT platform.

III. Innovations beyond Fog

To facilitate the achievement of the goal presented in Section II-A, we present three key innovations to Fog. Each technique applies to a different area and their combination enhances the flexibility of the platform to resemble that of Cloud. In addition, our solution extends prior work on Cloud and Grid Computing to adapt to real time constraints and latency requirements imposed by the "things" and the critical applications they enable [12]. The areas where we focus are the orchestrator, the resources available at the Fog layers, and how the infrastructure offers its capabilities to the applications.

A. Distributed orchestrator

The first step to implement that flexibility and overcome its limitations [13] is the modification of the orchestration policy that englobes the policy decision and the policy enforcement point. It is necessary to break the default Cloud option to exploit the advantages of the new Fog layers and their capabilities. At the end, Fog Computing implies the location of computational resources close to the "things". Then, our proposal is to execute the applications on the most convenient layer without a preset decision.

If the Fog layers are selected to execute workloads, the orchestrator takes into account different parameters to decide which Fog nodes are the responsible for each task. These parameters include geographic proximity, congestion, node's capabilities, and application requirements. Precisely, these two last parameters drive the decision. If a node does not have the required resources, it is automatically discarded. For example, if a node cannot guarantee a certain latency response, critical applications must discard that device.

Once this matching process is solved, an equitable distribution between the different nodes becomes fundamental. The system needs free resources to allocate the dynamic IoT applications while the execution of static services continues. In parallel, it can exploit the node visibility to its advantage. If an application executes under a unique Fog node, there is no need to migrate it to higher nodes. For example, imagine a wide range of sensors deployed within a smart building. In this case the closest Fog node can process all that data keeping the applications running. Only some statistical information can be sent up in the hierarchy if required.

All together allows to exploit the advantages of the Fog layers, optimizing executions over the infrastructure. Now, data is processed close to where it is generated and consumed. In consequence, aggregation takes place at lower layers of the hierarchy, eliminating unnecessary traffic and reducing the bandwidth because only necessary data is sent to the higher layers. Although bandwidth is not a problem yet, transmitting data from billions of "things" to the Cloud may pose structural problems to the underlying infrastructure.

B. Constellations of Fog nodes

The orchestrator modification led to another problem, the resources available at the Fog level are not the same as at the Cloud. The Fog nodes are a compendium of heterogeneous devices that are geographically distributed. Some of these nodes are complex devices with many capabilities (i.e. server with enhanced communication capabilities) while others are pretty simple (i.e. gateway) [14]. Hence, the infrastructure may not have the required capabilities in a desired location to execute a service [15].

To avoid sending these workloads to the Cloud or to unnecessarily deploy more devices, we propose to create constellations of Fog nodes. Aggregating the nodes' capabilities gives the perception of larger pools of resources close to the end users, although still far from the Cloud resources. These larger sets of resources come at a cost of latency since they

rely on distributed nodes. Here there is a trade-off between the constellation latency and the Cloud one. If Fog nodes exploit their locality to create these groupings, their latency should be below that of the Cloud. However, in certain situations where further nodes are needed the Cloud may appear as a feasible solution.

The idea of sharing resources was also presented in other environments such as Mobile Cloud [16]. They combine user devices capabilities without including system nodes in their local clouds, reducing its advantages. In opposition, constellations focus on Fog nodes from different layers and virtualization becomes the way to create and offer them.

Through virtualization, constellations also eliminate the view of multiple owners. Applications observe capabilities on a per constellation basis and not for each node individually. These virtual groupings also enhance the security and ensure the isolation between applications running on the same Fog nodes. The criteria to form groupings can be proximity or to add a certain capability such as hardware accelerators. Once groupings are formed, constellations can prioritize determined user demands based on the criticality of the application. Furthermore, the Fog nodes forming a constellation change dynamically due to the mobility of the devices. For example, a node on a bus can join/leave constellations based on the vehicle mobility.

Once the infrastructure uses this technique, applications have the required resources close to the "things". Workloads can be executed close to where the data is generated and thus exploiting the main Fog advantages without deploying extra hardware.

C. Fog Function Virtualization (FFV)

Currently, the deployment of a new service is a daunting task. Within the complex software required, there are the functionalities to exploit the devices and their capabilities. Solutions like Service Oriented Architectures (SOA) also face the same problem due to lack of abstraction in the devices' functionalities [17]. Then, the subsequent allocation between applications requirements and infrastructure capabilities becomes critical to enable IoT services.

Nowadays, solutions rely on proprietary code designed for specific devices that prevents reutilization. This fact supposes a barrier for new applications due to the huge effort required to deploy a new service and delays the explosion of Fog-based applications. Building on the concept of Network Function Virtualization (NFV) [18], we applied it to IoT and Fog computing through the definition the concept of Fog Function Virtualization (FFV). With these functions, a Fog-based architecture exposes their capabilities as high-level characteristics, regardless of each application aims. Computation and storage can be controlled as functions, but also "things" functionalities and analytics.

FFVs map the applications' requirements into the capabilities of Fog nodes and "things". As a result, the application development is reduced to choose the FFVs that produce the desired service. To achieve complex functionalities, developers may choose a chain of functions. In order to increase functions' re-usability, preexisting FFVs can be available through libraries. Each of these functions has a set of inputs (i.e. sensor measurements), it performs a processing, and produce an output (i.e. their average value) to be used for other application's layers.

Since Fog nodes are highly heterogeneous, FFVs are not capable of running in all the Fog nodes due to the nodes' capabilities. In case a function cannot run in the designated node there are different options. One of these options is to notify the developers that the desired node can only execute a subset of their FFV chain. This technique leads to the second option, that is to communicate that fact but also expose other nodes that can run it integrally at the cost of a higher latency. The third option arises from the definition of the functions themselves. If these functions are defined generically, nodes only execute the subset of the function that the node's capabilities can handle. Another option consists of developers adapting a preexisting function to perform the desired task in the available nodes. For instance, imagine an FFV computing the average of sensors measurements and comparing it with historical values within a database. If a Fog node does not have the resources to perform a database query, developers can take that FFV and perform only the average at the closest Fog node.

In consequence, FFVs enable the interoperability between the different players of an IoT system while hiding all their complexity to the users. FFVs create a dynamic architecture that can reuse components and thus enabling new business models for Fog-based systems. The time to market is then reduced and obstacles for new deployments are eliminated since just the non-existent functions need to be implemented.

Summary: Resulting from the three innovations, new services do not need to develop the entire software stack. Now, the requirements are the development of the FFVs in case they are not already implemented, request an instance with the desired resources, and start executing the application. The result is the democratization of the IoT applications by reducing the entry barriers. Small companies can offer their services to end users without being constrained by operational expenses, similarly as what happens with mobile devices' applications.

IV. Case Study

To demonstrate the utility of the innovations in dealing with IoT applications, we present a two scenario case study. It reflects different situations encountered when deploying new services over a generic Fog-based infrastructure.

A. Fundamentals

Before explaining the details of each scenario, it is necessary to explain the illustrative underlying infrastructure we use for this study. There are two types of Fog nodes as depicted in Figure 3. FN_A consists of wireless access points with some computational power. Its main advantage is an excellent connectivity with "things" while its drawback is its reduced

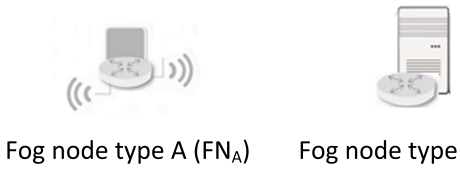

Fig. 3. Two illustrative categories of Fog nodes named FN_A and FN_B.

set of computational capabilities. In contrast, FN_B consists of a server with some network cards. This node does not have wireless communications but has available an excellent computational capacity. Although Fog nodes such as FN_B can have a rich set of functionalities (i.e. CPU power, connectivity), they do not operate as a distributed datacenter. Nodes can form constellations to increment their available resources but there is no awareness of belonging to a global datacenter.

These nodes are deployed hierarchically on a smart city environment, supporting different working applications. Among these services there are connected vehicles and automated homes. Each application has deployed sensors to ensure their proper operation, integrating them into the infrastructure. For example, the cars have a set of sensors that monitors the pollution, sense their near environment (i.e. other cars or pedestrians), and monitors the engine behavior.

Additionally, there is a generic FFV implemented. Operating through a standard interface, this function reads as inputs the sensors' measurements and provides their average in real time as output. We assume all sensors provide 64-bit floating point values with the same granularity.

B. Scenario 1: Using available resources

The idea is to deploy a smart grid use-case to enable a more efficient use of renewable energy sources [19]. After analyzing the current infrastructure, this new service can be provided using pre-existing basic blocks. More concretely, automated homes can provide the power consumption on all appliances in real-time, plus control over certain devices including air conditioner, refrigerator, and electric car among others.

Using the predefined FFV, the application obtains the average power consumption per home. Later, this value can be computed by regions such as districts in the city. Based on these requirements, a vertical constellation – with Fog nodes on different hierarchical layers – is created as shown in Figure 4. Different FN_As are used to compute per-home consumptions, while the FN_B calculate the region's average due to their larger visibility (FN_B scope englobes a set of FN_As). Thanks to the modified orchestrator, only the necessary data is sent to upper layers although all the computation could take place at the node with more visibility. In consequence, the constellation hides the infrastructure's complexity and the application only sees its requirements fulfilled through its instance. In this case, Fog is chosen over the Cloud because of its operational capabilities more than to Cloud latency problems [6].

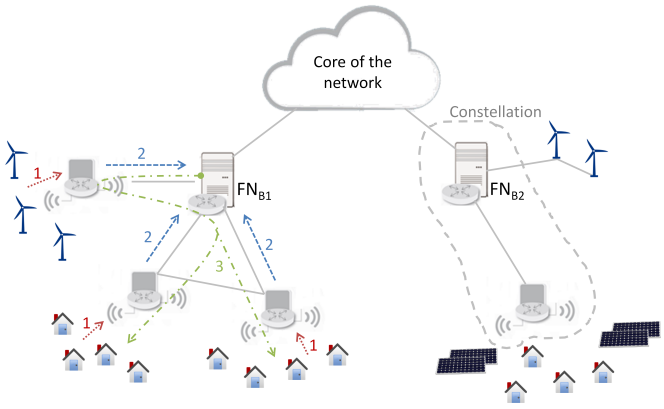

Fig. 4. Example of a Fog architecture using the three innovations for a smart grid application. The system forms vertical constellations and the orchestrator involves two Fog layers.

The other important part is to monitor the energy production. Each renewable source can provide the power generated in real time, processed with Fog nodes in a similar way the consumption is. These sources can be distributed geographically, solar panels can be in the city but windmills are outside. In consequence, the match between consumption and generation takes place in a distributed fashion at nodes where both values are visible. For example, in Figure 4 FN_{B2} monitors more energy produced than consumed in its coverage area. This remainder can be distributed to another region for consumption without polling each generator individually.

In addition, they want to activate the home appliances avoiding peak hours to benefit from lower energy prices. To reach this goal requires to define an FFV to provide analytics, mainly the energy consumption and generation per hour. Once this function is implemented, the nodes with control over energy sources can trigger the nodes controlling houses to activate appliances, as shown in the left branch of Figure 4.

C. Scenario 2: Adding resources to the infrastructure

Another application aims to provide a real-time contamination map including pollution and noise levels. In this case, connected vehicles proportion the pollution measurements with a key differentiation, mobility. While the cars move around the city, FN_A and FN_B nodes remain fixed. This fact provokes that a node cannot establish static associations with a certain set of sensors. Instead, each node covers an area and sensors migrate from one node to another, as reflected in Figure 5 when the car changes its Fog node from position P_1 to position P_2.

To gather the missing information, the application needs to deploy noise sensors across the city using streetlights. Afterwards, this new equipment can be integrated into the generic Fog-based infrastructure. In this way, the infrastructure capabilities improve with each deployment, becoming more attractive to new services. Now, the available system can provide all the contamination information to populate the map. Since this information can be displayed upon each

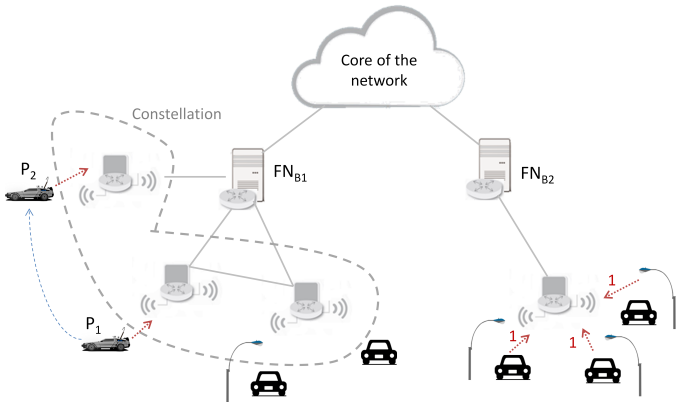

Fig. 5. Fog infrastructure hosting a different application, a contamination map that includes pollution and noise levels. In this case, the system uses horizontal constellations and the orchestrator only exploits the first Fog layer.

sensor location, there is no need for higher aggregation levels. In consequence, the orchestrator function is simpler than in Scenario 1.

Based on the aforementioned requirements, horizontal constellations can handle this application as shown in Figure 5. This service could stop here but the predefined FFV can serve another purpose, to improve the sensor's accuracy. Collecting the values from nearby sensors allows to use their average value as the real measure. The precision increases as a function of the number of sensors and their type (i.e. not all the sensors will be equal or have the same error). The dynamism affecting nearby sensors seems challenging, but FFVs provide a flexible framework to deal with it. With a function implementing a discovery process, each node can determine the sensors under its influence. Then, and based on that information, the node applies the average function.

V. Future Work

We have outlined a program to enhance Fog's architecture complementing the Cloud [20], opening three main areas for future research. Each area addresses different pillars of the generic Fog Architecture that entities such as the Open Fog Consortium are consolidating [5]. The first line of research focuses on orchestration policies to decide which application run on the Fog layers (based on their requirements, current conditions, etc.), and how these policies are enforced on Fog's heterogeneous nodes.

The second area focuses on constellations and their virtualization foundations. These groupings are influenced by networks conditions the infrastructure capabilities. This fact suggests simulation as a way to evaluate a rich set of scenarios where constellation design is enhanced. The last suggested area involves FFVs and the environment required to enable them. First, we need to develop the framework supporting the functions and acting as an interface between the different entities (ISPs, application developers, etc.). Later the actors need to deploy and offer "things" functionalities, creating a pool of resources from where the FFVs can emerge. Finally, the application developers can use those functions and enable new services and applications.

VI. Conclusion

In this paper we remarked the true potential of Fog by becoming a generic platform to support multiple applications simultaneously. To reach this objective, it is necessary to break the barriers that prevent its growth. The main obstacle is the amount of software required to integrate the Fog nodes into a current infrastructure.

To overcome this problem, we propose to enhance Fog with three innovations. First, to modify the orchestration policy allowing to execute more workloads on the Fog layers. Second, to create constellations of nodes to aggregate their capabilities, increasing the computational resources at the lower levels of the hierarchy. Third, the definition of the Fog Function Virtualization concept that provides great flexibility and adaptability. Now, the infrastructure's capabilities such as the "things" can be offered as functions and thus re-used for other applications. This solution brings Fog's democratization, enabling new applications to be deployed through the implementation of FFVs. Lastly, two scenarios were presented in a case study to show different implementations over a Fog-based platform.

Acknowledgment

The authors thanks Rodolfo Milito for his insightful comments and revisions. Damian Roca work was supported by a Doctoral Scholarship provided by Fundación La Caixa. Josue V. Quiroga work was supported by a Doctoral Scholarship provided by the Mexican National Council of Science and Technology (CONACyT). This work has been supported by the Spanish Government (Severo Ochoa grants SEV2015-0493) and by the Spanish Ministry of Science and Innovation (contracts TIN2015-65316-P).

References

[1] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, 2011.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the 1st edition of the workshop on Mobile cloud computing*. ACM, 2012.

[3] M. Yannuzzi, R. Milito, R. Serral-Gracià, D. Montero, and M. Nemirovsky, "Key ingredients in an iot recipe: Fog computing, cloud computing, and more fog computing," in *IEEE 19th International Workshop on CAMAD*, 2014.

[4] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*. IEEE, 2014, pp. 1–8.

[5] Openfog consortium reference architecture. [Online]. Available: https://www.openfogconsortium.org/ra/

[6] M. Yannuzzi, F. van Lingen, A. Jain, O. L. Parellada, M. M. Flores, D. Carrera, J. L. Pérez, D. Montero, P. Chacin, A. Corsaro *et al.*, "A new era for cities with fog computing," *IEEE Internet Computing*, vol. 21, no. 2, pp. 54–67, 2017.

[7] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," *Communications Magazine, IEEE*, 2013.

[8] S. Alam, M. M. Chowdhury, and J. Noll, "Senaas: An event-driven sensor virtualization approach for internet of things cloud," in *IEEE International Conference on NESEA*, 2010.

[9] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported by internet of things," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 1, pp. 81–93, 2014.

[10] A. Alrawais, A. Alhothaily, C. Hu, and X. Cheng, "Fog computing for the internet of things: Security and privacy issues," *IEEE Internet Computing*, vol. 21, no. 2, pp. 34–42, 2017.

[11] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: A programming model for large-scale applications on the internet of things," in *Proceedings of the 2nd ACM workshop on Mobile Cloud Computing*, 2013.

[12] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *2008 Grid Computing Environments Workshop*. IEEE, 2008.

[13] 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.

[14] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 Workshop on Mobile Big Data*. ACM, 2015, pp. 37–42.

[15] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 27–32, 2014.

[16] T. Nishio, R. Shinkuma, T. Takahashi, and N. B. Mandayam, "Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud," in *Proceedings of the first international workshop on Mobile cloud computing & networking*. ACM, 2013.

[17] S. d. Deugd, R. Carroll, K. Kelly, B. Millett, and J. Ricker, "Soda: Service oriented device architecture," *IEEE Pervasive Computing*, 2006.

[18] M. Chiosi, D. Clarke, P. Willis, A. Reid, J. Feger, M. Bugenhagen, W. Khan, M. Fargano, C. Cui, H. Deng *et al.*, "Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action," in *SDN and OpenFlow World Congress*, 2012.

[19] H. Farhangi, "The path of the smart grid," *Power and energy magazine*, 2010.

[20] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.