# FogIoT Orchestrator: an Orchestration System for IoT Applications in Fog Environment

Bruno Donassolo[1][2], Ilhem Fajjari[1], Arnaud Legrand[2], and Panayotis Mertikopoulos[2]

[1] Orange Labs - Châtillon, France
firstname.lastname@orange.com
[2] Univ. Grenoble Alpes, CNRS, INRIA, LIG - Grenoble, France
firstname.lastname@imag.fr

**Abstract.** Fog environment for IoT applications is extremely challenging due to the heterogeneity of its devices, their various capabilities and constraints. In this context, orchestration is still a hot topic and innovative approaches are required to deal with such a complexity. To lessen such a burden, we propose and demonstrate an orchestration system for IoT applications in fog environment. This solution relies on Calvin tool and makes use of both Grid5000 and FIT/IoT-LAB to build a realistic fog environment with efficient orchestration mechanisms.

## 1 Introduction

The Internet of Things (IoT) is a promising concept that revolutionizes our daily lives and the way we interact with our surrounding environment. The number of connected devices is expected to reach 75 billions in 2025[3]. Such an explosion will be inevitably the catalyst of Cloud infrastructure transformation. Being established too far away, traditional data centers struggle to meet stringent bandwidth and latency requirements of IoT systems. That is why the deployment of a new generation of infrastructure is crucial to deal with the huge amount of data transmitted by sensors and connected devices. In this context, fog computing [3] is shaping the future IoT solutions. Fog provides nearby physical resources, performing analytics tasks and thus taking the opportunity to capitalize on data. However, it raises new challenges in terms of applications orchestration. It is straightforward to see that the orchestration is a key stone of fog technology. It is crucial to deliver IoT services, based on the composition of micro-services. However, the heterogeneity, the dynamicity and the large-scale distribution of such an environment make existing orchestration solutions, even mature, ill-adapted. To deal with the aforementioned challenges, we propose an orchestration architecture for the fog environment. To put forward the efficiency of our design, we implemented a prototype while making use of both Grid5000 [2] and FIT/IoT-LAB [1] to build a realistic fog environment. We describe, in this paper, the implementation and the tools selected for this prototype.

## 2 Architecture and implementation

We design and implement `FogIoT Orchestrator`, an orchestration system for the automation of the deployment, the scalability management, and migration of micro-service based IoT applications. In this context, an IoT application, deployed in a fog environment, is composed of a set of micro-services, also called components, which are containerized and running on the nodes of the infrastructure. We adopt an actor model to describe our fog applications where each micro-service is modelled as a set of actors and communicates with other micro-services through flows. An actor is characterized by a private internal state and a set of communication ports through which tokens are transmitted.

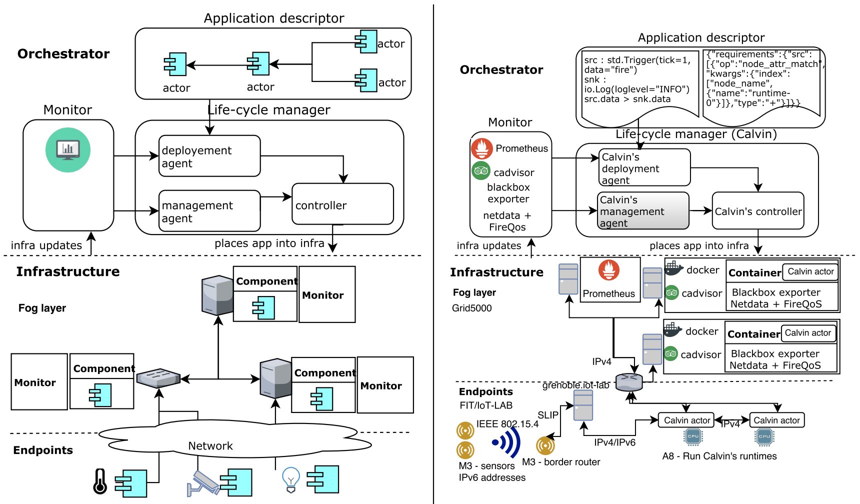
Unlike existing orchestration solutions, such as Kubernetes[4] and Mesos[5], our orchestrator is device-aware and hence handles the heterogeneity of the fog environment. Applications components can be easily deployed on IoT sensors and fog nodes. It is worth noting that a fog environment relies on distributed infrastructure encompassing three main layers: i) **Cloud layer**: centralized resources with almost unlimited processing and storage capacities; ii) **Fog layer**: resources distributed in the network, closer to the endpoints and characterized by limited capacities; iii) **Endpoints**: set of sensors which collect information about the environment.

Advanced orchestration mechanisms are put forward to deal with the stringent requirements of such applications. The global architecture for the proposed orchestration system is depicted on  1 showing its components and their roles. To deploy an application, the developer makes use of the **application descriptor** to describe the actors, their requirements in terms of both location and computational effort, and how data should circulate between them. It is worth noting that network related requirements could be specified during the description of links between the actors in order to ensure a guaranteed QoS. Once the description is submitted, the **application deployment agent** handles the mapping between the application (i.e., node, links) components and the nodes hosting the latter. Its main objective consists in optimizing the placement of actors and their links while considering both computational and network requirements. Once deployed, applications could be continually monitored by the **application management agent**, with the aim of being managed

[3] Statista. Available at: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[4] Kubernetes. Available at: https://kubernetes.io/.

[5] Apache Mesos. Available at: http://mesos.apache.org/.



```
src : std.Trigger(tick=1,
data="fire")
snk :
io.Log(loglevel="INFO")
src.data > snk.data
```

```
{"requirements":{"src":
[{"op":"node_attr_match",
"kwargs":{"index":
["node_name",
{"name":"runtime-
0"}]},"type":"+"}]}}
```

(a) Architecture Overview

(b) Implementation

Fig. 1: FogIoT Orchestrator

while considering the environment conditions. To do so, scale out/in and migration could be triggered to deal with load variation and to optimize resource usage. The **infrastructure monitor** is responsible for sketching out the telemetry information by extracting various resources metrics from the fog nodes and links.

To implement and experiment our orchestration solution, we make use of both FIT/IoT-LAB and Grid5000 infrastructures. Our main objective is to set up a realistic fog environment, enabling the deployment of IoT applications and hence their orchestration. To do so, we use M3 and A8 boards to set up the endpoint layer encompassing the sensors equipment. Then, we make use of the servers hosted in Grenoble to emulate the fog layer. It is worth noting that, for sake of simplicity, we don't emulate the cloud layer for these experiments. However, such a layer could be easily instantiated using other data centers of Grid5000, such as Rennes or Nantes. To connect both infrastructures, we create L3 VPNs between the A8 nodes and Grid5000. Also, we use public IPv6 addresses to connect the M3 sensors in the network, as detailed in the tutorials available at FIT-IoT platform. Each node of our fog infrastructure hosts: i) **Docker** engine to cope with hardware and software heterogeneity. Docker containers are used to create an image which encapsulates the software necessary in our setup, namely Calvin, Blackbox exporter and Netdata; ii) **Cadvisor** to ensure the monitoring of Docker containers performance. Some of the collected metrics are CPU and RAM utilization; iii) **Blackbox exporter** to measure the latency between the node and all other nodes of the infrastructure; iv) **Netdata** and **FireQoS**: to monitor the bandwidth utilization without the support of network equipment. In our setup, we use these tools to measure the bandwidth utilization of each actor's flow. Note that the flow is characterized by a TCP connection between two nodes. Using FireQoS, we configure the Traffic Control module in the Linux kernel to identify each flow and to keep statistics about them. Then, Netdata collects the information and sends it to our infrastructure monitor.

Our orchestration solution relies on both **Calvin** [4] and **Prometheus**[6] to instantiate the aforementioned building blocks of our functional architecture. Calvin is a project lead by Ericsson that proposes a framework for the development of IoT applications. First, we have extended the Calvin's application descriptor to support dynamic metrics such as available CPU and RAM, network latency and bandwidth. By doing that, we are able to consider the stringent requirements of IoT applications. Then, we make use of Calvin deployer to instantiate application's components while considering the specified requirements. To ensure a holistic monitoring, we make use of Prometheus to collect metrics about our fog platform behavior during the tests. Prometheus relies on the aforementioned monitoring tools, such as netdata, cadvisor and blackbox exporter to generate the metrics.

## 3 Conclusion

In this prototype paper, we propose a new orchestration solution for IoT applications in fog environment capable of handling the complexity and heterogeneity of such an environment. In order to experiment our orchestrator, we make use of both platforms Grid5000 and FIT/IoT-LAB to set up the system. By using physical devices of both platforms, we were able to mimic most of the fog characteristics, such as geographical distribution and heterogeneity. Hence, the importance of having both infrastructures as an enabler for complex studies in the fog computing area.

[6] Prometheus. Available at: https://prometheus.io/.

## References

1. Adjih, C., Baccelli, E., Fleury, E., Harter, G., Mitton, N., Noel, T., Pissard-Gibollet, R., Saint-Marcel, F., Schreiner, G., Vandaele, J., Watteyne, T.: Fit iot-lab: A large scale open experimental iot testbed. In: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT). pp. 459–464 (Dec 2015)
2. Balouek, D., Carpen Amarie, A., Charrier, G., Desprez, F., Jeannot, E., Jeanvoine, E., Lèbre, A., Margery, D., Niclausse, N., Nussbaum, L., Richard, O., Pérez, C., Quesnel, F., Rohr, C., Sarzyniec, L.: Adding virtualization capabilities to the Grid'5000 testbed. In: Ivanov, I.I., van Sinderen, M., Leymann, F., Shan, T. (eds.) Cloud Computing and Services Science, Communications in Computer and Information Science, vol. 367, pp. 3–20. Springer International Publishing (2013)
3. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: 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. 13–16. MCC '12, ACM, New York, NY, USA (2012)
4. Persson, P., Angelsmark, O.: Calvin – merging cloud and iot. Procedia Computer Science 52, 210 – 217 (2015), the 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015)