

UNIVERSITÀ DI PISA  
DIPARTIMENTO DI INFORMATICA

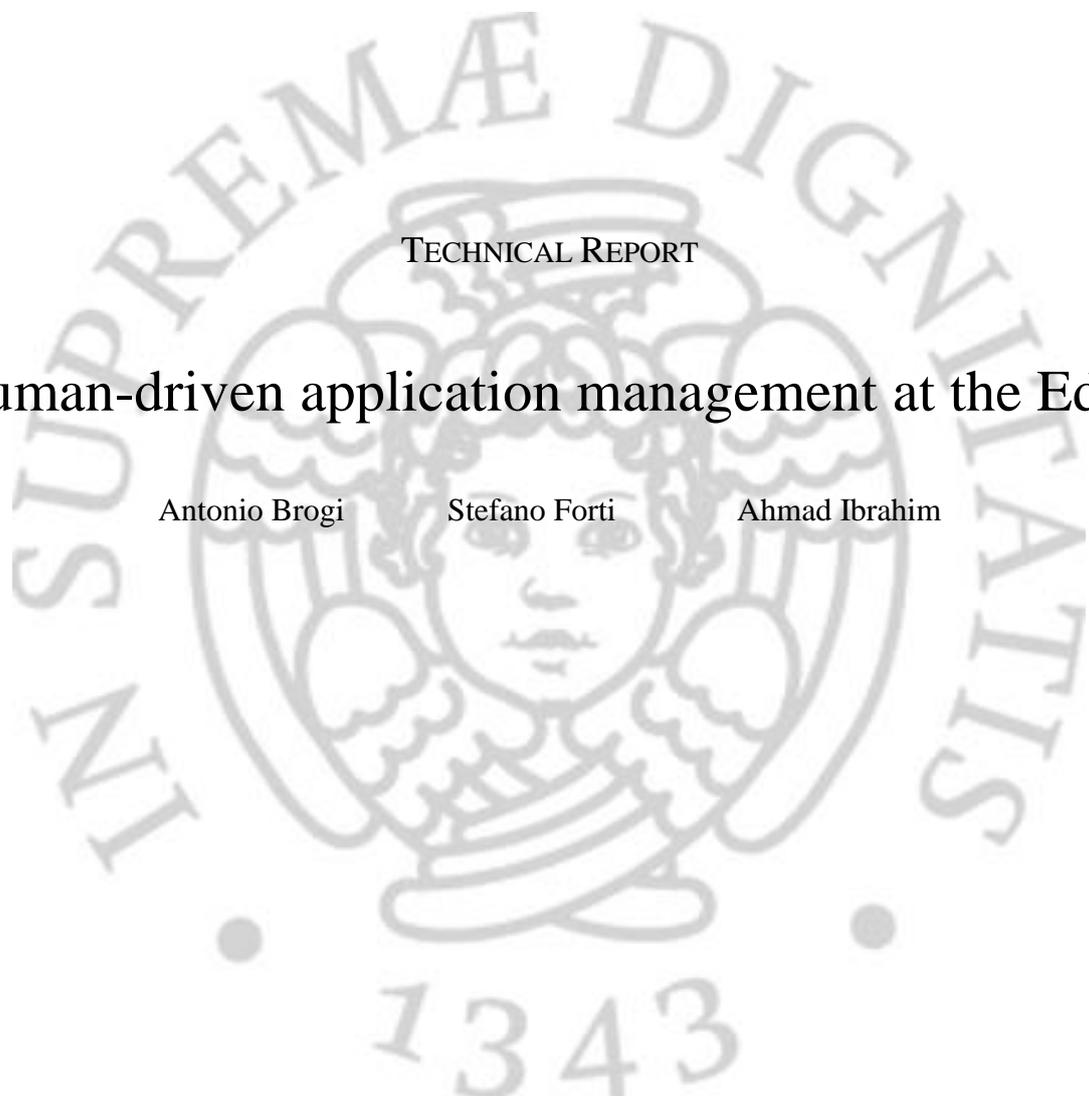
TECHNICAL REPORT

# Human-driven application management at the Edge

Antonio Brogi

Stefano Forti

Ahmad Ibrahim



8<sup>th</sup> May 2017

LICENSE: **Creative Commons: Attribution-Noncommercial - No Derivative Works**

ADDRESS: Largo B. Pontecorvo 3, 56127 Pisa, Italy - Tel. +39 050 221270

# ***Human-driven application management at the Edge***

*A. Brogi, S. Forti, A. Ibrahim*

*Department of Computer Science  
University of Pisa, Italy*

**Abstract** - The design and management of Edge systems will proactively involve human intelligence at the Edge, according to a human-driven approach that increases productivity and improves usability. Due to its ubiquity and heterogeneity, the Edge will give to application administrators a more decisional role in application deployment and resource management. Final decisions on where to distribute application components should be informedly taken by them during the entire application lifecycle, accounting for compliance to QoS requirements.

As a first step, this requires devising new tools that suitably abstract heterogeneity of edge systems, permit simulating different runtime scenarios and ease human-driven management of such systems by providing meaningful evaluation metrics. In this article, we discuss how human decision-making can be supported to solve QoS-aware management related challenges for Edge computing.

## 1 Introduction

Edge computing, also referred to as Fog computing [1], promises to extend the Cloud towards the Internet of Things (IoT) to better support time-sensitive and bandwidth hungry IoT applications. Edge computing will trigger development of new applications, capable of exploiting a hierarchy of cooperating capabilities between Cloud datacentres and end-user devices. Indeed, Edge capabilities promise both to reduce processing latencies and mandatory need for Internet connectivity, and to filter the amount of data travelling from the IoT to Cloud datacentres and back.

A substantial amount of computation, storage and networking is expected to happen over a collaborative crowd of near-user edge devices working contiguously and interdependently with the Cloud. Self-driving cars, autonomous domotics systems, energy production plants, agricultural lands, supermarkets, healthcare, schools will more and more exploit devices and Things that are integral part of the Internet and of our existence without us being aware of them. On the other hand, this move of computation from the Cloud to the Edge of the network permit to human users to take back control of their applications and data, being part of the decision loop at various levels.

As highlighted in [2], there is a possibility to design a new human-centred paradigm, fruitfully considering the involved human stakeholders, their requirements and their knowledge [3]. All of this to avoid complete delegation of applications and systems control from users to Cloud providers, to better control privacy and data disclosure, and to avoid the considerable waste of (idle) resources on powerful end devices at the edge (e.g., tablet, smartphones, set-top-boxes). Overall, proximity, intelligence, trust and control reside at the edge of the Internet, where data is produced, often involving human actors (e.g., wearable devices, GPS, e-health). Thus, it is essential to investigate the roles that human actors can play when designing and managing Edge systems, also supported by new tools that proactively assist their decision-making in a usable and useful manner.

Managing location-awareness, exploitation of heterogeneous and pervasive capabilities, compliance to non-functional constraints (latency, bandwidth and security) are some of the challenges posed by the new scenario to reduce the deficiencies of the Cloud-only paradigm. If architecture is about functionality allocation [4], a human-centred design of Edge computing cannot help envisioning the cornerstone decisions that concern human stakeholders when mapping functionalities (e.g., storage, analytics, cyber-physical processing) over computing capabilities along the continuum from Cloud to Things. Our motivation in this article is precisely to contribute understanding how to enable human-guided management of Edge computing applications, with a closer focus on deciding how to distribute application functionalities (i.e., deploy them) all through the IoT+Edge+Cloud infrastructure in a QoS-aware manner.

In what follows, after framing the scenario of Edge applications management, we describe the involved stakeholders with focus on the role of application administrators (Sect. 2). We then describe an e-health application example to show complexity of analysis and decision-making for such a role (Sect. 3). Afterwards, we comment on how this type of analysis can be performed including human experts in the loop (Sect. 4). Finally, we mention two existing tools to help decision making when mapping application functionalities to IoT+Edge+Cloud infrastructures (Sect. 5), and we briefly conclude (Sect. 6).

## 2 Edge Application Management

### 2.1 Multicomponent Applications

With service-oriented architectures becoming established, applications are made from (loosely coupled) components that programmatically interact to deliver comprehensive software solutions. Such software solutions are known as multicomponent (or composite) applications, being the standard in modern enterprise IT. An application deployment can be defined as a one-to-one mapping from application components to computational resources, like Cloud or Edge nodes. Deployment of multicomponent application is very much functionality allocation.

Components represent a set of program logic, they require supporting hardware (e.g., CPU, RAM, storage) and software (e.g., libraries, OS), and they expose inter-component interaction capabilities to cooperate with other components. Moreover, composite Edge applications will intensely interact with IoT devices, which become a new requirement to be appropriately fulfilled at deployment time. Each component can be deployed (i.e., installed) independently from another, provided its software, hardware and Things requirements are suitably satisfied.

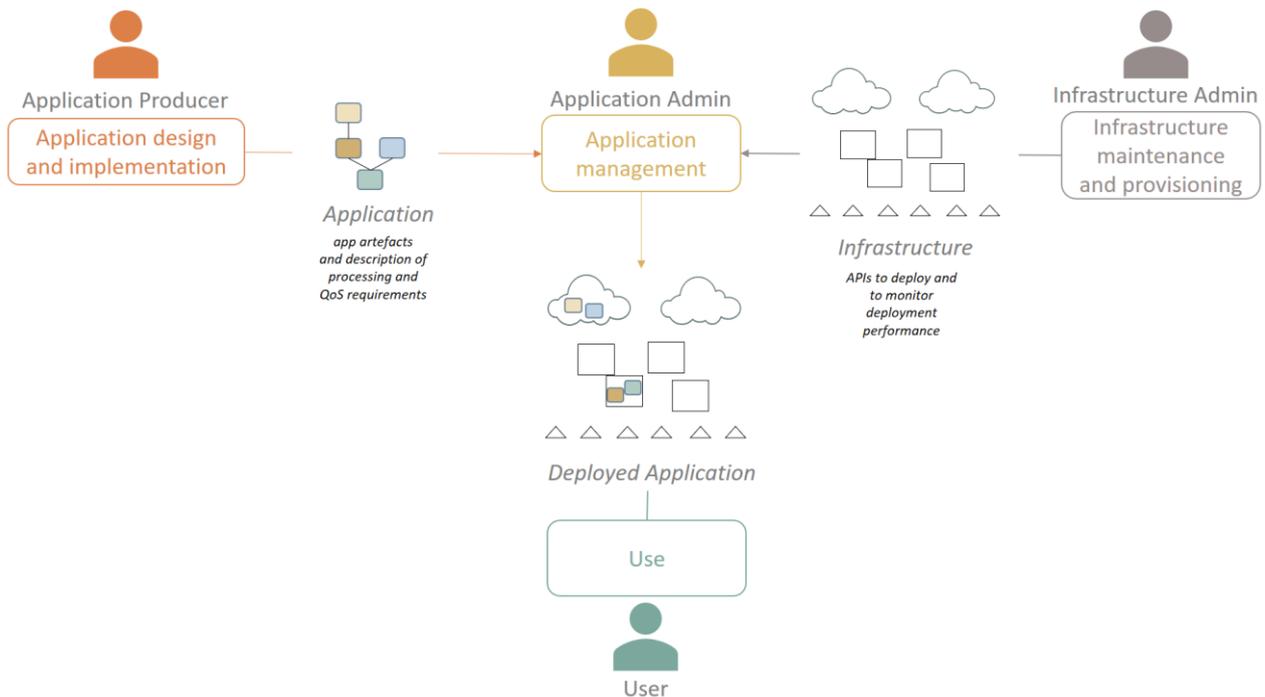
Cooperating components must do so in such a way well-defined levels of service are met, i.e. not only should the application work correctly, utterly it should perform well. For instance, consider the braking system of a driverless car: reaction to sensed obstacles is not sufficient per se, but must also happen within a very brief time interval (less than 500 ms) to prevent potentially severe damages to other cars or to persons. Hence, in addition to hardware and software capabilities, the availability of adequate connectivity QoS (e.g., latency, bandwidth, jitter) turns out essential to achieve QoS-aware application deployments.

### 2.2 Stakeholders and roles

When considering composite IoT applications at the Edge, the stakeholders involved in their deployment can play different roles. In the simplest scenario, we can undertake four main roles, as sketched in Figure 1:

- *Application producer*, in charge of designing and implementing software solutions, providing all necessary artefacts (i.e., executable files) and information (i.e., software/hardware and QoS requirements) to deploy them,
- *Application administrator*, in charge of designing and managing an application deployment by specifying its constraints (e.g., privacy or user needs), by monitoring its performance, and by redistributing or reconfiguring application components when needed,
- *Infrastructure administrator*, managing targeted Edge, Cloud and IoT resources, and providing monitoring APIs for the provisioned infrastructure,
- *User*, that is the end user of the deployed application and may add further preferences or requirements to consider.

Naturally, one stakeholder can play more than one role at a time and there can be more stakeholders playing the same role [5], e.g., a telecom provider can provide the IoT+Edge+Cloud infrastructure to its customers, whilst managing applications and services over the very same infrastructure, or an end-user can act as infrastructure provider when sharing her home router as Edge capability for application deployment.



**Figure 1 - Stakeholder roles in Edge application management.**

In the DevOps culture, where software is built within cross-functional teams of IT experts and software developers, the distinction we have depicted gets more and more blurred, i.e., the roles of application administrator and producer merge together. Edge computing also assumes users to be able to deploy their own applications over a provider's infrastructure integrated with user-managed devices, i.e. the roles of application administrator, infrastructure administrator and user collapse to a single human.

Thus, later, even focusing on human-guided application management and concentrating on the role of the application administrator, we will assume the other roles to be part of the application life cycle. As we will show, new methods, models and tools are needed to support human-guided application management, whilst taming scale and heterogeneity of Edge systems.

### 2.3 Application Management

Managing multicomponent software systems in IoT+Edge+Cloud environments will conceivably involve human decision-makers from deployment to retirement of any application. Indeed, to keep composite Edge applications up and running, their administrators will have to follow a loop such as the one in Figure 2. This reveals a MAPE-like loop [6] for Edge application management, where application administrators should:

- **analyse** – process data about the application, the infrastructure and monitored performance as to take informed decisions on how to (re-)distribute application components,
- **plan** – identify the actions sequence to (re-)distribute components to different Edge or Cloud nodes based on QoS data and specified policies,
- **execute** – orderly perform the deployment of all components or redistribute part of the application, and
- **monitor** – monitor QoS performance of the running application to check if it complies to the SLA provided to final users.

If **plan** can be largely automated (i.e., solutions exist to determine the actions needed to achieve a set deployment) and **execute, monitor** are usually fully automated, **analyse** gives a chance to human actors for closing the loop of predictive distributed systems, and take control on application management.



**Figure 2 - Application Management.**

The depicted process is especially challenging when the application to be deployed is privacy critical, mission critical or even life critical. On one hand, unpredictable performance or inconsistencies are not tolerated by such types of vertical. On the other hand, quantity, heterogeneity and (often) limited availability of resources of Edge devices make the duty of managing application deployments intrinsically complex to be tuned just manually. This is the reason why, whilst also **analyse** could be substantially automated for some verticals, we envision others in which the ultimate choice shall be taken informedly by human experts and, only after, programmatically performed.

Next, we describe in more detail the four steps involved in the proposed loop for human-driven Edge application management.

### 2.3.1 Analyse & Plan

When deciding how to distribute application components, decisions will need to trade-off different QoS parameters that a certain deployment is expected to achieve over a range of non-functional parameters such as privacy, QoS-assurance or resource consumption of resource-constrained devices at the Edge. Indeed, the analysis of application specific requirements along with data on infrastructure behaviour may lead decision towards different segmentations of application components from the IoT to the Cloud. Besides, the fluid and interoperable system of connectivity that Edge platforms will seamlessly offer makes it difficult for application administrator to determine SLAs for their customers and enforce compliance to promised service level objectives.

New tools and methods should enable the application admin to integrate application requirements specified by the application producer – hardware, software and QoS – with deployment policies that are business dependent, or directly come from final users’ needs. For instance, an industrial plant may be interested in keeping on local (or partner) Edge nodes those components that include undisclosed business procedures or data. Here, reliable predictive simulation tools can help to evaluate beforehand the impact of tactical moves, and to plan optimal strategies to achieve the desired QoS for the deployed application.

As we will better detail later, components distribution along the continuum from Cloud to Things should consider resiliency and robustness to churn and failure, capacity planning and technical feasibility even before any actual implementation. Modern enterprise IT is eager for tools that permit to virtually compare business scenarios by abstracting unnecessary details and design SLAs for their offerings.

### 2.3.2 Execute

Actual deployment of application components should then be made as easy as possible, also accounting for the fact that final users of Edge applications can also play the role of application administrators. After an

eligible (and suitably good) deployment plan has been identified, it should be easy to perform it automatically, without worrying too much about technical details, interoperability issues or low-level protocols to access IoT sensors and actuators. Continuous delivery of software components (i.e., the ability to continually deploy reliable software in production environments) also requires to be able to quickly roll back any software change that does not perform as expected, as well as to be able to roll out software updates.

In these regards, current proposals for Edge architectures include the design of a middleware responsible for protocol bridging, resource discovery and exposing APIs to make application deployment operations (deploy, reconfigure, update, undeploy) trouble-free, and to make IoT data available to all deployed services. Once the application administrator specifies the IoT or Edge devices the application should work with (e.g., all cars entering a high-way), it should not worry about the operations for continuously discovering newly entered devices. This will be made possible by proper virtualisation of the infrastructure hardware and by convergence towards standard interfaces offered by different infrastructure and assets providers. Tools will be available to launch virtual instances on demand, running components' artefacts onto compatible nodes.

Consider as an example a scenario in which customers exploit their smartphones to scan the articles they buy when grocery shopping at the supermarket. End-devices continuously enter and leave the shopping facility, yet we cannot ask the application administrator to worry about how they interface with the shopping service deployed on the central Edge node at the supermarket.

### 2.3.3 Monitor

Analogously, monitoring of the infrastructure should be made available via proper interfaces and should empower users to access that information even when connectivity is scarce. Human experts should be immediately informed when deployment faces performance degradation or cannot work properly. In a context where edge devices, including near-user and user-managed assets, could be used to deploy mission-critical applications guesstimate is not sufficient to understand how deployed services are behaving. On the other hand, complete “centralised” monitoring of the system would be unfeasible or impractical due to the scale of Edge infrastructures.

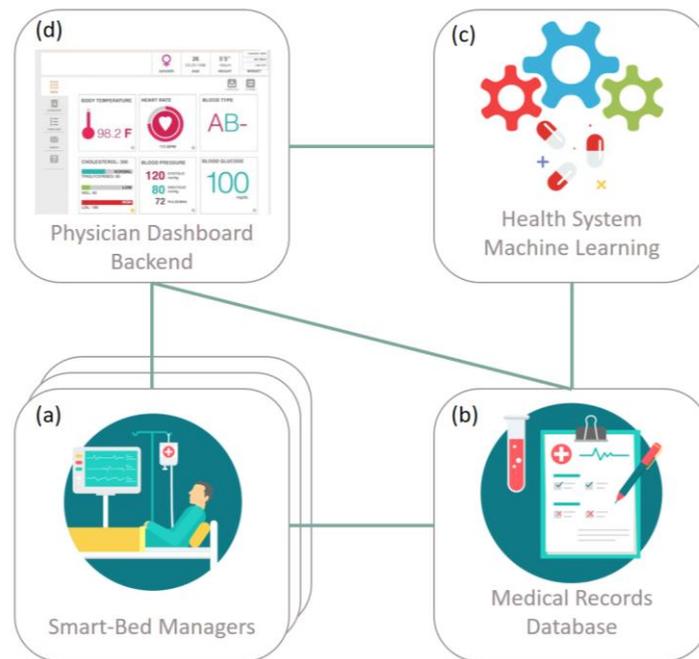
Edge computing calls for techniques and methods to permit crowdsourced and distributed monitoring of both the infrastructure and the applications running on top of it. Monitoring tools could then exploit this information to notify any anomaly in the behaviour of deployed applications, asking for human intervention when needed. Also, final users should be included in the monitoring loop, by asking feedback on their experience and or on their preferences on application deployment.

Retaking the supermarket example, in case the Edge node managing customers at the shopping centre gets overloaded, the application administrator should immediately be informed about the problem and promptly intervene. Tools may suggest replicating components on different available nodes as soon as possible, which includes purchasing external resources, also asking end-users to deploy parts of the application on their smartphones offering them in turn shopping discounts.

## 3 An e-Health Example

As aforementioned, Edge computing envisions a multitude of computing capabilities (e.g., mini-datacentres, routers, personal devices) spanning the continuum from IoT to Cloud. Those devices support virtualisation technologies – e.g., virtual machines or containers – and can be exploited for deploying applications that sense from and act upon IoT sensors and actuators respectively. The activity of properly mapping components to computational nodes is proven to require worst-case exponential time, even for relatively small-sized application/infrastructure couples and it is inherently challenging (i.e., NP-hard [7]).

To better convey the complexity of this duty, we depict a more complete application example about IoT+Edge+Cloud systems for healthcare. Due to its time-sensitive and life critical nature, healthcare applications are good candidates to be supported by Edge systems [8]. Consider an application which monitors data about hospitalised patients in a hospital and stores their medical records in a central database over time. The application monitors the vitals of hospitalised patients with the help of several medical sensors and can react to stimuli through actuators, as specified by physicians. Doctors access and act upon data through a dashboard, whilst anonymised data about all patients and prescribed cures is processed for statistics purposes.



**Figure 3.** *e-Health application components.*

Assume the application consists of four components, interacting with each other as indicated by the lines in Figure 3. Each component works as follows:

- (a) *Smart bed manager*, that is deployed for each patient and stores their data to the Medical Records Database, acting when possible to maintain stable their physiological conditions or immediately alerting the nursing staff in case of anomalies;
- (b) *Medical Records Database*, a complete information system keeping records of all patients' data for a specific department of the hospital;
- (c) *Health System Machine Learning*, a regional system to store and analyse (anonymised) data, providing statistics about hospitalisations and trying to find correlation between symptoms, cures and recovery of patients.
- (d) *Physician Dashboard Backend*, accessible by the medical staff during the patients' hospitalisation or visits to retrieve data about their clinic history, to enforce prescriptions to the Smart-bed manager, or to receive alerts from patients monitoring;

Each of the components have hardware, software and QoS requirements to fulfil, and so do interactions among components. Particularly, (a) must reach out both IoT devices (e.g., heartrate, drips, HR) and component (b) within a bounded timespan, to ensure timeliness in monitoring the patient and reporting emergencies to the nursing staff. Component (c), in contrast, has more relaxed time constraints on interactions with (a) and (b), whilst it must be deployed to nodes owned and managed by the hospital for

privacy reasons. Finally, (d) is already deployed to a specific Cloud provider, requiring full support for the Hadoop MapReduce framework.

Overall, due to its life (and mission) critical characteristics, the application should comply to specified constraints as close as possible to 100% of the time. Moreover, patients continuously reach and leave the hospital, or simply are moved from a department to another. Finally, the hospital allocated a limited monthly amount of budget for keeping the whole deployment up and running, and aims at not exceeding the cost estimate of more than 5%.

The application administrators responsible for managing the e-health system will need to consider various factors to determine a best possible deployment. As we have seen, those factors include application requirements (e.g., hardware and software), enforced policies (e.g., privacy constraints), target metrics (e.g. cost, QoS, resilience), and monitoring of the infrastructure state (e.g., resources, latency, bandwidth, hospitalised patients). In the next section, we discuss how this type of predictive analysis can be carried out by application administrators.

## 4 Predictive Analysis at the Edge

With respect to unbounded virtual instances offered at the Cloud layer, Edge nodes will feature limited hardware capabilities, often falling in the category of resource constrained devices. Hence, application components should be smartly deployed over infrastructure to make efficient use of the Edge-Cloud interplay and avoiding load-unbalanced situations. Edge resources should not be overloaded nor remain unutilised. This is already challenging, even without considering dynamicity of the infrastructure, e.g., mobility, disconnections, and variations in the provided resources (Edge nodes and IoT devices) or QoS of communication links.

In typical Edge scenarios, devices connect and disconnect very often, joining or leaving the infrastructure seamlessly. If the node hosting a given application component disconnects, functionality of the application may be seriously affected. Also, when identifying the enabling infrastructure for a given application, care should be taken in making sure that resources will be enough to support an everchanging number of connected nodes. Thus, the analyse phase should rely on historical data about availability, uptime and reliability of involved nodes to enable well-versed decision making.

In between the extreme connected/disconnected cases, due to congestion, network QoS (bandwidth, latency, jitter) vary over time. Indeed, available bandwidth may decrease during peak and increase during non-peak hours, as the number of users connected through the same access point or Internet Service Provider changes. Conversely, it may happen that new Edge nodes join the network and provide faster connection to neighbouring devices. Eventually, Denial of Service attacks to a certain network or scheduled maintenance may impede access to the Internet or make latencies and bandwidth fluctuate.

Depending upon the type of application to be deployed (or re-configured), the analysis to plan for an eligible deployment can be entirely manual, semi-automated or fully-automated.

- *Manual analysis* - Manually deciding application deployment entrusts completely application administrators with the activity of finding the best deployment by hand, without exploiting any auxiliary tool but their previous experience. Although the clear advantage of full control over each decisional step, this approach shows major shortcomings when dealing with the size of the infrastructure, the churn (devices entering and leaving the infrastructure) and the successive continuous (re-)deployment of components. Furthermore, determining deployments (or redistribution) by hand configures as a full-time 24/7 responsibility and it is, by the human nature, error-prone.

It is not reasonable to expect human experts to be able to face such a complexity alone. For instance, it is highly probable that re-configuration is required at run-time when monitored performance does not reach up to set SLA thresholds. Human experts alone may risk to hastily choose the first alternative deployment for some components, which may be not the best to meet QoS or costs requirements.

- *Fully automated analysis* - Symmetrically opposite, fully-automated application management would lift application administrators from controlling deployment. All deployment-related decisions would be automatically taken based upon some rules that the application producer or administrator have set up. Although this looks like a best-case scenario, automatic deployment cannot be considered a cure-all.

The efficiency of any automatic system is strongly dependent upon the decision policies stored in the system. In case no appropriate policy is found in the knowledge base for a certain component, an automated system may either fail or take considerably sub-optimal decisions or totally wrong decisions. Such scenarios are very likely due to the human impossibility to predict future circumstances and to program policies accordingly, yet they have minor impact on application that are not, for instance, life critical (e.g., home entertainment).

Finally, enterprise IT includes many scenarios in which the application administrators do want to have the final word or must take full responsibility on application deployment due to laws and regulations. Simply consider the video-surveillance application deployed at an airport: who oversees deploying the system will probably have to sign documents certifying compliance with international security standards.

- *Semi-automated analysis* - Semi-automated approaches possibly take the best of the previous alternatives. Humans take control of the decisional process of how to map application components to Cloud/Edge nodes, providing necessary input (QoS and security preferences) and informedly deciding among suitable output alternatives. Segmenting functionalities over the continuum from IoT devices to Cloud datacentre, they define the architecture of their Edge system, assisted by simulation or predictive tools.

Indeed, based on the output results they can evaluate the quality of eligible deployment plans trading off among various estimated metrics (compliance to expected QoS, resource consumption, security policies and costs), managing performance against SLAs, whilst avoiding guesstimate or complete delegation.

Due to the costs for setting up actual deployments (or even testbeds) to evaluate performance of Edge systems, the new paradigm could exploit this type of tools also to efficiently simulate deployment scenarios and to evaluate beforehand performance of deployed applications. An application administrator can make use of supporting tools at design time so to be able to take decisions for the infrastructure and the application, making sure that the deployed application will work well.

## 5 Predictive Analysis Tools

Without the presence of analysis tools, manually determining the deployments and their runtime behaviour is laborious and time consuming. Tools considering user-centred objectives are key to best determine where to map functionalities to better satisfy customer needs. Due to the costs for setting up actual deployments (or even testbeds) to evaluate performance of Edge systems, the new Edge paradigm calls for methods and tools to efficiently simulate deployment scenarios and evaluate beforehand their performance. An application administrator can make use of supporting tools at design and runtime so to be able to take decisions for the infrastructure and the application, making sure that the deployed application will work well.

Definitely, the devised models and methodologies should capture essential elements of the Edge platforms (pervasiveness, latency-bandwidth-resource constraints, security) and permit to personalise SLO for different

customers, depending on their preferences or behaviour. Such tools, if enriched with lifelike cost and pricing models, may be also extensively used to design and test new business models for the Edge paradigm, impersonating different stakeholders.

To the best of our knowledge, only two research efforts have already gone towards this direction of understanding how to distribute application components over Edge infrastructure: iFogSim [9] and FogTorchII [10] namely.

### 5.1 iFogSim

iFogSim enables simulation of Edge computing environments to evaluate resource management and scheduling policies based on their impact on QoS parameters. The focus of iFogSim model is mainly on stream-processing applications and hierarchical tree-like infrastructures, to be mapped either Cloud-only or Edge-ward before comparing the results. Figure 4 shows a screenshot of iFogSim GUI.

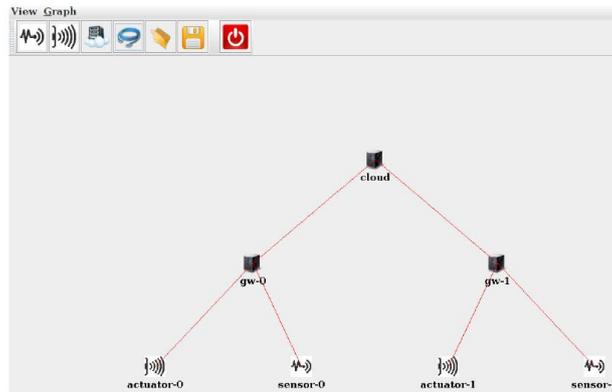


Figure 4. iFogSim GUI, as in [8].

### 5.2 FogTorchII

FogTorchII modelling stays more general and describes any infrastructure or application type. FogTorchII inputs a multicomponent application with its requirements (hardware, software, QoS), an IoT+Edge+Cloud infrastructure annotated with probabilistic QoS performances for the communication links, along with application administrator deployment policies. Exploiting Monte Carlo simulations, FogTorchII can simulate and aggregate data over a myriad of possible scenarios and determine feasible deployments (i.e., meeting functional and non-functional requirements), ranked by their compliance to specified requirements (QoS-assurance) and resource consumption on specified Edge nodes, as shown in Figure 5.

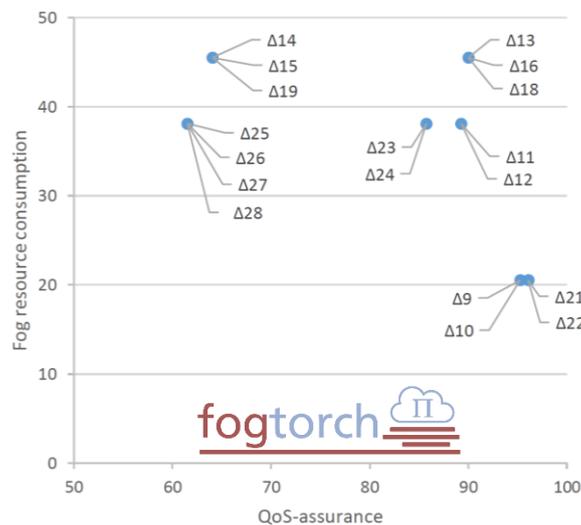


Figure 5. Eligible deployments, indicated by  $\Delta_i$ , as ranked by FogTorchII.

## 6 Concluding Remarks

In this paper, we discussed some ways of enabling humans to take back control on Edge application management. Principally, we focused on the role that human experts can play in analysing the complex scenarios of deploying multicomponent applications to Edge computing, assisted by predictive tools designed for faster and better decision making. At the same time, such tools can analyse monitored data, determine eligible deployments and help predicting future performance of the deployed application based on simulation. Needless to say, the future of Edge computing just started, and much remains to suitably balance between human and system interactions so to make the involved stakeholders aware of the choices to be made throughout Edge application management.

## 7 References

- [1] Bonomi, Flavio, et al. "Fog computing and its role in the internet of things." *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012.
- [2] Garcia Lopez, Pedro, et al. "Edge-centric computing: Vision and challenges." *ACM SIGCOMM Computer Communication Review* 45.5 (2015): 37-42.
- [3] Saariluoma, Pertti, José J. Cañas, and Jaana Leikas. *Designing for Life: A Human Perspective on Technology Development*. Springer, 2016.
- [4] Chiang, Mung, and Tao Zhang. "Fog and IoT: An overview of research opportunities." *IEEE Internet of Things Journal* 3.6 (2016): 854-864.
- [5] J.-P. R. B. a. S. L. Arcangeli, «Automatic deployment of distributed software systems: Definitions and state of the art.,» *Journal of Systems and Software*, n. 103, pp. 198-218, 2015.
- [6] Jacob, Bart, et al. "A practical guide to the IBM autonomic computing toolkit." *IBM Redbooks* 4 (2004): 10.
- [7] Brogi, Antonio and Stefano Forti, "QoS-aware Deployment of IoT Applications Through the Fog.", To appear in *IEEE Internet of Things Journal*, 2017.
- [8] Chakraborty, Suryadip, et al. "Fog Networks in Healthcare Application." *Mobile Ad Hoc and Sensor Systems (MASS), 2016 IEEE 13th International Conference on*. IEEE, 2016.
- [9] Gupta, Harshit, et al. "iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in Internet of Things, Edge and Fog Computing Environments." *arXiv preprint arXiv:1606.02007* (2016).
- [10] Brogi, Antonio, Stefano Forti, and Ahmad Ibrahim. "How to best deploy your Fog applications, probably.", *International Conference on Edge and Fog Computing*, 2017, Madrid.