When the Cloud Goes Pervasive: Approaches for IoT PaaS on a Mobiquitous World

Luiz Angelo Steffenel[1(✉)] and Manuele Kirsch Pinheiro[2]

[1] Université de Reims Champagne-Ardenne Laboratoire CReSTIC, Reims, France
luiz-angelo.steffenel@univ-reims.fr
[2] Université Paris 1 Panthéon-Sorbonne, Centre de Recherche en Informatique, Paris, France
manuele.kirsch-pinheiro@univ-paris1.fr

Abstract. Today, IoT applications are heavily dependent on public cloud computing services to perform data storage and analysis. Unfortunately, the cloud computing paradigm is unable to meet the requirements of critical applications that require low latency or enhanced privacy levels. The deployment of private cloud services on top of pervasive grids represent an interesting alternative to traditional cloud infrastructures, allowing the use of near-environment resources for IoT data analysis tasks. In this work we discuss the challenges associated with the deployment of IoT services over pervasive environments, and present a study case deployed over CloudFIT, a computing middleware for pervasive systems. Hence, we evaluate the behavior of a data-intensive application under volatility and heterogeneity constraints, bringing to light to the use of low-end devices that are usually located at the proximity to IoT sensors/actuators.

Keywords: IoT · Pervasive environments · PaaS · Cloud computing

1 Introduction

According to [1], in the next 25 years, most of the things and devices we interact with will be linked to a global computing infrastructure. This massive integration of communicating capabilities on physical objects symbolize the advent of IoT. The Internet of Things (IoT) represents a new tendency on IT industry, in which physical environment is populated by interconnected and communicating objects, capable of interacting with each other and with the environment itself. The strength of this concept lies in the seamlessly integration of sensors, actuators and other devices in the environment in a large scale, allowing interacting and collecting information from this.

Several factors are contributing the increasing development of IoT, among them the cost of sensors, bandwidth and processing power that have decline in the last years [5]. Thanks to current technology and its reducing costs, IoT is already becoming a reality. Nowadays, it is possible to put a wireless interface on

B. Mandler et al. (Eds.) IoT 360° 2015, Part I, LNICST 169, pp. 347–356, 2016.
DOI: 10.1007/978-3-319-47063-4_36

almost all every day object, making possible interaction between them [9]. Such communicating capabilities open countless opportunities in different application domains, like health-care and smart cities, just to name a very few.

However, the full potential of IoT will only be reached if the data collected by IoT devices can be analyzed and explored. As suggested by Jones [5], big data analytics is making IoT possible. Indeed, only collecting data from the environment is not enough without an appropriate computing analysis allowing actions and decisions to be made based on these data.

Currently, computing IoT data is been performed mostly on cloud computing infrastructures. Different authors [2,3,6] have been pointing the integration with cloud infrastructures as a key aspect for IoT platforms, since storage and computing power of IoT devices is often limited. Indeed, cloud computing are offering powerful and flexible capabilities for running IoT data services and applications by using Internet infrastructure [12]. By using cloud platforms, it is possible to analyze increasingly volume of data, following an on-demand model, in which new resources can be easily allocated according application needs.

Despite its advantages, cloud platforms have also some important drawbacks. Among these, we may cite security and privacy concerns, as well as network latency [4]. Indeed, the transfer of large volume of data from IoT environments to cloud platforms may be significantly costly and time consuming, and it may also expose private data to a public infrastructure. Such drawbacks may prevent the use of public cloud infrastructures on IoT applications particularly concerned by privacy issues or by the transfer of large volume of data. In order to overcome these limitations, the use of private cloud have been considered. Further, most of private cloud platforms suppose the availability of dedicated resources, such as a cluster, which represent an important investment for concerned organizations.

In this paper, we explore a different approach for IoT applications concerned by these issues. We consider, in this paper, the deployment of IoT applications over pervasive grids. Pervasive grids represent the extreme generalization of grid platforms, in which heterogeneous resources may dynamically and opportunistically integrate (or leave) the platform [8,13]. Pervasive grids allow exploring under-utilized resources available on the near environment for IoT data analysis tasks, reducing the need for expensive data transfers and costly computing infrastructures. We propose here the deployment of the CloudFIT platform, a private PaaS (Platform as a Service) cloud, over a pervasive grid and discuss challenges and opportunities this deployment offers for IoT applications.

This paper is organized as follows: Sect. 2 presents related works on cloud platforms for IoT processing. Section 3 discusses opportunities and challenges of using pervasive grids for IoT processing. Section 4 introduces the platform CloudFIT as a private PaaS platform for IoT, while Sect. 5 analyses experimental results of CloudFIT on pervasive grids. Finally, Sect. 6 presents our conclusions and future works.

2 Related Works

One of the most important outcomes of IoT is the possibility of creating an unprecedented amount of data, which has to be stored and used intelligently for smart monitoring and actuation [3]. This ability of sensing physical phenomena or triggering actions on the physical reality is what differentiates IoT from traditional networked systems. IoT focus is on data and information, since, from the conceptual standpoint, IoT is about entities acting as providers and/or consumers of data related to the physical world [6].

In this context, cloud platforms may act as a receiver of data from the IoT environment, offering computer power to analyze and interpret the data [3]. Different cloud-based platforms have been proposed for distribute and manage IoT applications and data. Villalba et al. [15] proposes a scalable platform for IoT data storage and processing in the cloud, named ServIoTicy, which focus on data stream processing, offering IoT applications data store and access facilities through a REST based API. Similarly, Fazio et al. [2] focus on data storage services, proposing a monitoring-oriented cloud architecture for storage of big data. These authors propose a platform offering services for managing and querying data of different kinds, from simple measures performed by sensing devices up to complex multimedia objects.

Serrano et al. [12] share a similar focus, by considering IoT Cloud service data management based on annotated data of monitored Cloud performance and user profiles. They consider enabling management systems to use shared public infrastructures and resources in order to provide an efficient deployment mechanism for IoT services and applications. By this mechanism, these authors focus indeed on enabling elasticity of IoT Cloud services. Similarly, the Aneka platform [3], a .Net based PaaS (Platform as a Service) platform, offers cloud management services and support resources coming from other private and public cloud platforms, such as Microsoft Azure.

Finally, Mulfari et al. [7] propose a message-oriented middleware for cloud, named MOM4C, which allows composing cloud facilities according to client requirements. This platform offers services to dynamically deploy applications running on smart objects by means of container-based virtualization techniques. Virtualization isolates applications from heterogeneity of IoT environment, but, in the case of MOM4C, this also limits targets objects to those based on Linux, preventing other smart kinds of objects to contribute with the platform.

Establishing on-demand cloud services on top of existing resources is also alternative to the complete externalization of services in a cloud. For example, [10] explore the limitations of mobile devices through the use of Cloudlets, i.e., virtual machines deployed on-demand in the vicinity of the demanding devices. Using cloudlets deployed as Wi-Fi hotspots in coffee shops, libraries, etc., the authors of [10] suggest a simple way to offer enough computing power to perform complex computations (services) all while limiting the service latency. This idea of consuming proximity resources is also explored by pervasive grids, which promote the use of heterogeneous devices in an opportunistic way. Next section discusses the use of such grids for IoT applications.

3 Pervasive Grids for IoT

Previous section demonstrated how cloud platform can significantly contribute to IoT applications. Such platforms are commonly used for distributing processing and storage capabilities necessary to IoT applications. Although advantageous, cloud platforms have also important drawbacks that may limit their adoption by some IoT applications. First of all, public cloud platforms are prone to privacy and security concerns. Public cloud providers do not offer sufficient protection for organizations that depend upon classified or proprietary information [4]. Schadt et al. [11] also underline important privacy issues related to medical or biometric data. Besides, network latency may have an important impact on the transfer of large volume of data to cloud platforms. Indeed, even high-speed connections have a limited bandwidth that can be overloaded by the transfer of important volume of data. As applications make even-more intense use of large volume of data, data transfer poses an increasing bottleneck [4].

In these cases, an alternative for IoT application can be the use of pervasive grids. Pervasive grids seamlessly integrate pervasive sensing/actuating instruments and devices together with classical high performance systems [8]. These grids lie on the use of idle and under-explored resources as a dynamic computing platform. In the context of IoT, pervasive grids represent an opportunity to deploy computing tasks, and notably data analysis ones, in computing resources available around IoT devices, minimizing data transfer over distant network. Pervasive grids offer the possibility of consuming computing power and storage from any available resources, independently of its nature, from small Raspberry Pi devices up to virtual machines deployed on cluster infrastructures.

Nevertheless, the use of pervasive grids raises important challenges, related to the dynamic nature of these environments. Among these challenges, two of them, heterogeneity and volatility, are quite related to IoT applications, which also have to cope with mobility and network volatility, and by consequence, with temporary unavailability of objects and resources [9]. Handling heterogeneity means to be able to seamlessly integrate resources of different natures in the same computing environment. Whatever it nature is, a resource should be able to contribute with computing tasks assigned to the grid, according its own capabilities. Considering volatility, it comes from the dynamic nature of the resources composing these grids. A pervasive grids rely on volatile resources that may appear and disappear from the grid, according their availability [13]. It can be a laptop that come and go, according its owner's moving, or a Raspberry Pi that switches off due to a low battery condition. Pervasive grids platforms have to deal with this volatility, allowing resources to seamlessly leave the grid or new ones to join it, without a significant impact on tasks execution. Application executing on pervasive grids might keep executing despite this volatility, taking advantage from the resources while they are available.

4 CloudFIT as a PaaS

As stated in the previous section, one of the major concerns on the design of a pervasive platform is to be able to ensure the execution of an application in spite of failures, a constraint that requires the use of decentralized coordination, fault tolerance and replication techniques.

CloudFIT [13] is a P2P distributed computing middleware structured around collaborative nodes connected over an overlay network (Fig. 1) and based on the FIIT (Finite Independent Irregular Tasks) paradigm. CloudFIT was designed to be independent of the underlying overlay, and the current version supports both FreePastry and TomP2P overlay networks, as well as their respective DHT services. While initially designed for computing intensive applications (combinatorial problems, etc.), the association with the storage capabilities from DHTs offer interesting possibilities for the big data and data analysis for IoT.

As previously presented in [14], we believe that CloudFIT can be used to provide a pervasive PaaS for IoT applications. Indeed, CloudFIT can be deployed on heterogeneous devices, from dedicate servers to Rapsberry PI-like devices, and is supported on both Android, Linux, Windows or MacOS. While this flexibility allows CloudFIT to be run directly on some recent IoT devices, the limited resources from these nodes make this approach very unreliable. A better approach, instead, is to use CloudFIT as a computing backend for IoT devices and applications. This mixed architecture, as illustrated in the left side of Fig. 1, allows an IoT application connected to CloudFIT network to act as an interface to gather data and launch computing tasks according to the application needs.

The development of an interface for IoT devices can be provided through *REST* calls or even a direct a connection to the devices via Bluetooth or Wi-Fi, but such development is outside the scope of this paper. Instead, the next section

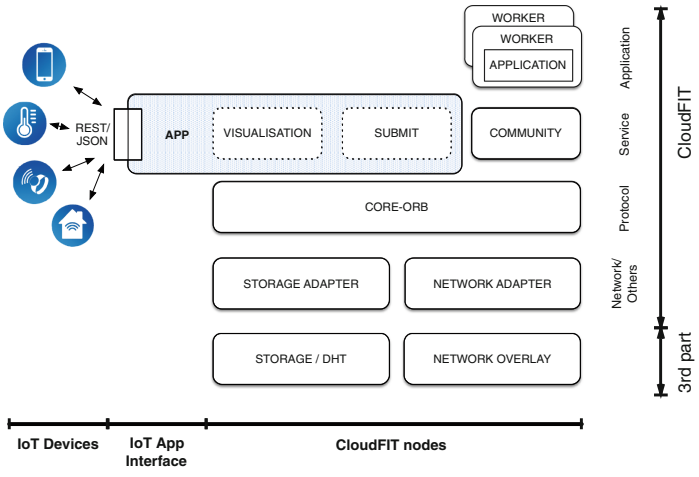

Fig. 1. CloudFIT architecture stack

analyzes the deployment of a data intensive application over a pervasive cluster using CloudFIT, with a special attention to both volatility and heterogeneity aspects of the execution.

5 Experiments

The experiments in this paper presents the deployment of a Map-Reduce application over a cluster of nodes running CloudFIT. As in a previous work [14] we compared the performance of CloudFIT against the well-known Hadoop framework, this paper focus on the impact volatility and heterogeneity on the behavior of CloudFIT.

5.1 Impact of Volatility

This first experiment presents the deployment of a WordCount application with a total of 1 GB of data, split in blocks of 64 MB. Figure 2 shows the Gantt diagram for an execution with no failures (for clarity, we limit each node to one single execution core). Indeed, we observe the deployment of several map tasks (with variable execution lengths), plus a reduce task at the end.

Here, we can observe the basic scheduling mechanism implemented on CloudFIT, developed to be totally decentralized and fault tolerant. For instance, when an application is deployed, a list of tasks is distributed among the nodes. Each node rearranges the list of tasks in a random order. When node picks a task tagged as *available*, it changes its status to *in execution* and advertises this to the others nodes. When a task is *completed*, its status is broadcasted to the other nodes and its status is updated. If all tasks marked as *available* were picked, a node may start computing other tasks marked as *in execution*.

This scheduling algorithm ensures that all tasks will be computed with little coordination between the nodes. We can easily recover tasks from failed nodes or perform speculative executions on tasks that take too long due to a slow processor, for example. Also, when a node joins the CloudFIT community, it receives an update about the tasks current status and the working data, allowing

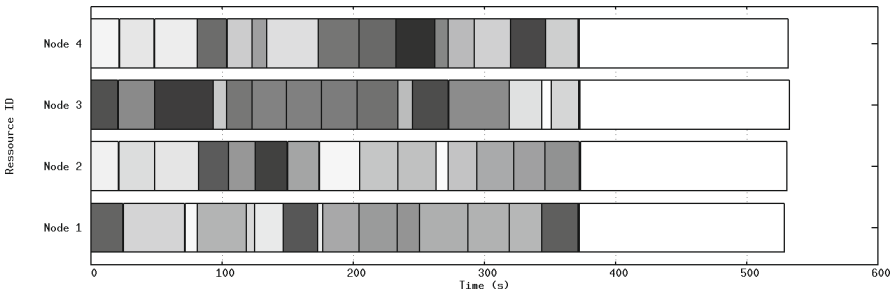

Fig. 2. Regular execution of WordCount (1 GB, 64 MB data blocks)

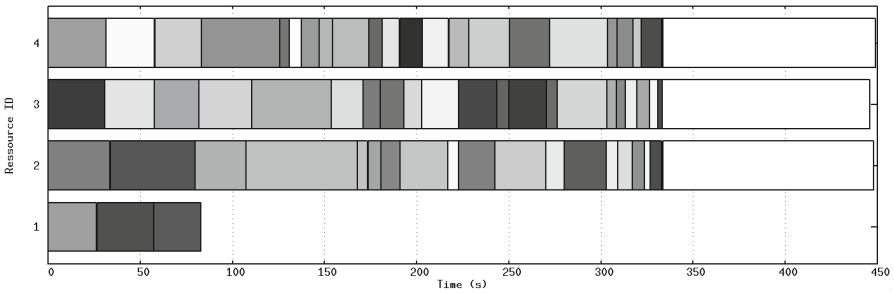

Fig. 3. WordCount execution when one node fails (1 GB, 64 MB data blocks)

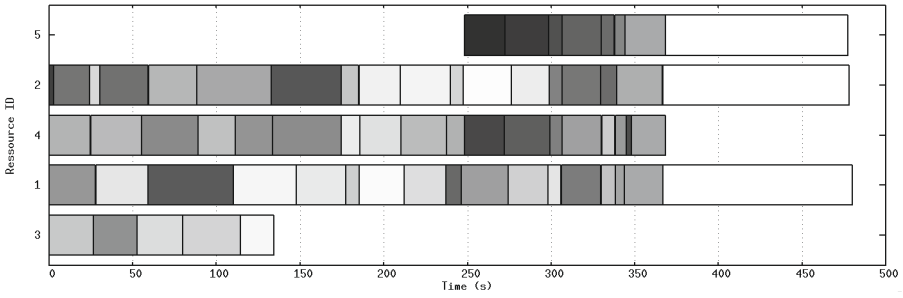

Fig. 4. Execution when a node joins after another failed (1 GB, 64 MB data blocks)

it to start working on available (incomplete) tasks. Figures 3 and 4 respectively illustrate a situation where one node fails and another where a failed node is replaced by a new node.

One eventual drawback of the totally decentralized scheduler is the fact that a task may be launched by multiple nodes simultaneously. This is indeed the reason why our experiments show several nodes executing the reduce task. Please note that CloudFIT allows users to develop additional scheduling algorithms that respond to specific need. Currently we are studying how to integrate context-awareness to incorporate additional parameters such as CPU speed, available memory and network speed in order to optimize the execution of the applications.

5.2 Impact of Heterogeneity

This second experiment aims at observing the impact of node heterogeneity when running CloudFIT, as our proposal relies on the association of nodes with different characteristics to offer PaaS on a pervasive system. For instance, we interconnected four nodes with different specifications (cf. Table 1). As in the precedent experiment, we limit to one core per node to simplify the visualization.

We also modified the experiment parameters to perform over 512 MB of data split in chunks of 2 MB each, as we believe that this configuration is closer to the patterns from current IoT devices (each sensor/node generating a limited data

Table 1. Specification of the nodes on the pervasive cluster

Node	Processor	GHz	Memory	OS
MacBook Air	Intel Core i7-4650U	1.7	8 GB	MacOS 10.10.5
Lenovo U110	Intel Core2 Duo L7500	1.6	4 GB	Ubuntu Linux 15.4
Raspberry Pi 2	ARM Cortex-A7	0.9	1 GB	Raspbian Linux Wheezy
Virtualbox VM	Intel Core i7*	2.2*	1 GB	Debian Linux 8.2

*values provided by the virtual machine guest

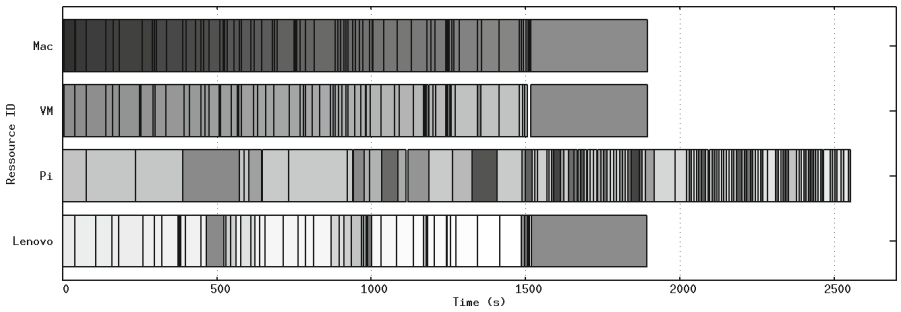

Fig. 5. WordCount execution on an heterogeneous network (512 MB, 2 MB data blocks)

amount). Also, this allows less powerful nodes to contribute with some tasks. Figure 5 shows the Gantt diagram for an execution on such scenario.

While the tasks distribution among the laptops and the virtual machine presents no distinctive difference, we observe without surprise that the Raspberry Pi does not perform as fast as the other nodes (as illustrated by the average task length on the first half of the execution). In addition, we observe that this node misses several status update messages and does not detects the end of the map phase and keeps trying to launch completed tasks. Unfortunately, this proves unsuccessful as the results are already in the DHT (the reason why the latter tasks take so little time).

This result does not refrain us from targeting small, less-powerful devices but, on the contrary, challenges us to understand and attack the causes of these problems. Up to now we identified that the DHT replication algorithm is one of the major factors affecting low-end devices. For instance, small devices have slow and limited memory/storage capacity (only a few hundred MBs of RAM, SD cards, etc.), and they expend a lot of resources trying to keep up with the replication process. As a consequence, this overhead interferes both with the computing performance and the message delivery between nodes, as observed in our experiment. We are currently investigating alternative techniques to integrate such nodes to the computing network without the burden of managing the DHT. Further developments, configuration and experiments with other DHT and overlays shall allow us to address these issues.

A complementary approach consists on developing context-aware schedulers so that small devices could contribute to tasks/jobs corresponding to their capabilities. This way, applications with specific needs such as response time, complex data transfer patterns or huge storage needs could be preferentially directed to nodes corresponding to these attributes, without overloading the small devices. As stated in the previous section, we are currently developing context-aware schedulers that eventually will perform such distinction among the nodes.

6 Conclusions and Future Work

IoT environments are the next important step towards the establishment of mobiquitous systems. Currently, computing IoT data is been performed mostly on cloud computing infrastructures, which are not always adapted to the needs from IoT devices. Indeed, moving data to distant platforms for filtering, analysis and decision-making is both expensive, time consuming and prone to security flaws, not always corresponding to the requirements from IoT applications.

In this paper, we explore a different approach for IoT applications concerned by these issues. Such approach relies on the deployment of IoT applications over pervasive grids, allowing the use of near-environment resources for IoT data analysis tasks. This way, we reduce the need for expensive data transfers and costly computing infrastructures, and are able to delimit the diffusion of the data. We present how CloudFIT can be used to create private PaaS clouds at the proximity of the demanding IoT devices. Using a P2P overlay, CloudFIT offers both storage and computing capabilities on top of pervasive networks.

As pervasive systems are characterized by strong volatility and heterogeneity of the resources, this paper analysis the CloudFIT behavior through the deployment of a data-intensive application under such constraints. With these experiments, we bring to light to the use of low-end devices like Raspberry Pi. These devices are usually located at the closest-area to IoT sensors/actuators, offering both interconnection and elemental processing capabilities.

Of course, the possibilities that CloudFIT offers to IoT are not limited to MapReduce applications. The CloudFIT API and its distributed computing model allow many other usages, as devices can use the platform as a storage support, data analysis support, intensive computing support, etc. By coordinating activities over CloudFIT, IoT devices and applications can elaborate a supply chain from data gathering to reasoning and actuation.

References

1. Broy, M., Schmidt, A.: Challenges in engineering cyber-physical systems. Computer **47**(2), 70–72 (2014)
2. Fazio, M., Celesti, A., Puliafito, A., Villari, M.: Big data storage in the cloud for smart environment monitoring. Procedia Comput. Sci. **52**, 500–506 (2015). The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015)

3. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (IoT): a vision, architectural elements, and future directions. Future Gener. Comput. Syst. **29**(7), 1645–1660 (2013)
4. Hofmann, P., Woods, D.: Cloud computing: the limits of public clouds for business applications. IEEE Internet Comput. **14**(6), 90–93 (2010)
5. Jones, M.: Internet of things: shifting from proprietary to standard. ValueWalk, July 2014. http://www.valuewalk.com/2014/07/internet-of-things-iot/
6. Miorandi, D., Sicari, S., Pellegrini, F.D., Chlamtac, I.: Internet of things: vision, applications and research challenges. Ad Hoc Netw. **10**(7), 1497–1516 (2012)
7. Mulfari, D., Fazio, M., Celesti, A., Villari, M., Puliafito, A.: Design of an IoT cloud system for container virtualization on smart objects. In: Celesti, A., Leitner, P. (eds.) ESOCC 2015 Workshops. CCIS, vol. 567, pp. 33–47. Springer, Heidelberg (2016). doi:10.1007/978-3-319-33313-7_3
8. Parashar, M., Pierson, J.M.: Pervasive grids: challenges and opportunities. In: Li, K.C., Hsu, C.H., Yang, L.T., Dongarra, J., Zima, H. (eds.) Handbook of Research on Scalable Computing Technologies, pp. 14–30. IGI Global (2010)
9. Paridel, K., Bainomugisha, E., Vanrompay, Y., Berbers, Y., Meuter, W.D.: Middleware for the internet of things, design goals and challenges. ECEASST **28** (2010). http://journal.ub.tu-berlin.de/index.php/eceasst/article/view/392
10. Satyanarayanan, M., Bahl, P., Caceres, R., Davies, N.: The case for VM-based cloudlets in mobile computing. IEEE Pervasive Comput. **8**, 14–23 (2009)
11. Schadt, E.E., Linderman, M.D., Sorenson, J., Lee, L., Nolan, G.P.: Computational solutions to large-scale data management and analysis. Nat. Rev. Genet. **11**(9), 647–657 (2010)
12. Serrano, M., Le-Phuoc, D., Zaremba, M., Galis, A., Bhiri, S., Hauswirth, M.: Resource optimisation in IoT cloud systems by using matchmaking and self-management principles. In: Galis, A., Gavras, A. (eds.) FIA 2013. LNCS, vol. 7858, pp. 127–140. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38082-2_11
13. Steffenel, L., Flauzac, O., Charao, A., Barcelos, P., Stein, B., Cassales, G., Nesmachnow, S., Rey, J., Cogorno, M., Kirsch-Pinheiro, M., Souveyet, C.: MapReduce challenges on pervasive grids. J. Comput. Sci. **10**(11), 2194–2210 (2014)
14. Steffenel, L.A., Pinheiro, M.K.: CloudFIT, a PaaS platform for IoT applications over pervasive networks. In: 3rd International Workshop on Cloud for IoT (CLIoT 2015), Taormina, Italy September 2015
15. Villalba, Á., Prez, J.L., Carrera, D., Pedrinaci, C., Panziera, L.: servioTicy and iServe: a scalable platform for mining the IoT. Procedia Comput. Sci. **52**, 1022–1027 (2015). The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015)