2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing

Exploiting Local Clouds in the Internet of Everything Environment

Francisco Javier Nieto de Santos
ATOS Research and Innovation
ATOS
Bilbao, Spain
Francisco.nieto@atos.net

Sergio García Villalonga
ATOS Research and Innovation
ATOS
Barcelona, Spain
Sergio.garciavillalonga@atos.net

***Abstract*—The Internet of Everything is opening new opportunities and challenges which will be faced during the following years. Huge amounts of data will be generated and consumed, so Internet of Things frameworks will need to provide new capabilities related to Big Data analysis, scalability and performance. We believe the formation of local clouds of devices, close to the location where data is created and consumed, is a good solution to overcome these issues which may impact in security as well. The combination of local and remote resources together with the appropriate allocation algorithms for their management will provide the means to enable the new required features, going beyond the current state of the art and still leaving enough evolution capacity for future scenarios.**

Keywords-Mobile Computing; Local Clouds; Big Data; Internet of Everything; Allocation Algorithms

I. INTRODUCTION

Internet of Things (IoT) environments have evolved to a wider concept known as Internet of Everything (IoE), where everything (people, things, content…) is expected to be connected. This kind of environments has been gaining momentum in the market during the last years. It is not only the fact of the number of smart devices and sensors increasing year by year, but also the industrial activities around, such as the collection of up to 57 application scenarios for the IoT [1] done by the Internet of Things initiative (IoT-i).

A huge amount of physical objects will provide tons of data to be used intensively by many mobile applications expected to be in use. IoT and Cloud technologies should be combined somehow, in order to take the maximum advantage from data while, at the same time, scalability and performance reach optimal levels.

Although some IoT frameworks are starting to exploit the Cloud as a way to store and provide data, there is still a lot of potential to be exploited by means of added value services. IoT frameworks must take into account new advancements and trends in the field, such as virtualization and Big Data, which can enrich them and improve their way to operate. This is what we intend to do by proposing a solution based on the usage of local clouds made of local devices.

The BETaaS (Building the Environment for the Things as a Service) project proposes to build a content-centric platform distributed over a local cloud providing a new environment for applications accessing M2M services and devices, where virtualization is exploited for adapting the platform behavior (increasing or decreasing resources for Big Data analytics) and for enabling the usage of local resources for running applications consuming data generated in the environment. Such usage of virtualization is based in a set of principles and ideas (Section II) that aim at getting a balance in resources consumption between local and remote sources. Once a decision is taken, the platform carries out concrete processes for the local allocation of resources (Section III) or for the remote allocation and interaction with external Clouds (Section IV), always based on certain VM templates (Section V), as a way to fulfill the requested features. Related work (Section VI) is mentioned, explaining the position of our solution, as well as conclusions and future work (Section VII), describing the benefits and potential evolution of the work done.

II. RESOURCES VIRTUALIZATION IN THE MOBILE ENVIRONMENT

A. *Main Concepts, Principles and Objectives*

Applying virtualization in a mobile environment, in our case, is done in the context of a platform focused on the management of things connected to gateways locally called BETaaS. In this context, we will have some pieces of hardware running the platform (i.e. on boards like Cubietruck [1]) and several sensors and actuators (things) generating data and effects in the environment. Moreover, several applications may connect to this platform and request high amounts of data or even data analytics functionalities. This platform aims at exploiting local capabilities in what we call local clouds of resources, since keeping computation and storage close to the data sources has several benefits:

- Data is not moved away from the place in which it is consumed, improving performance and security;
- The local instance does not rely on external cloud platforms for providing data analytics related features;
- Depending on the system load and required resources, it is possible to scale up and down the platform, adapting its resources and energy consumption;
- It is possible to work only in the local environment, avoiding continuous access to the Internet

[1] http://cubieboard.org/

1066-6192/15 $31.00 © 2015 IEEE
DOI 10.1109/PDP.2015.117

In order to create these local clouds, several gateways may decide to join their efforts under what we call a BETaaS Instance, in such a way they can share data and resources whenever necessary, in the local environment. Moreover, it is still possible to access external Cloud platforms in case it is considered necessary. We differentiate between physical gateways and logical gateways (known as BETaaS Gateways), so a physical gateway could host one or more logical gateways, by using Virtual Machines (VMs).

There are different kinds of BETaaS Gateways, depending on the hardware hosting them. Only physical gateways with enough computing power and with compatible hardware can exploit virtualization. In our case, the main objective of exploiting the virtualization is to host web applications or particular components of the BETaaS Instance (such as resources for the Big Data capabilities), not available in other IoT platforms.

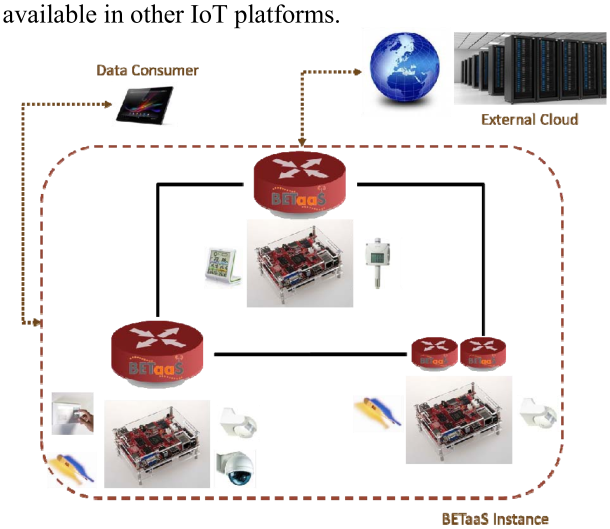

Figure 1. Elements of a BETaaS Instance.

Depending on the amount of resources used in the different BETaaS Gateways in an instance, it is possible to determine an optimal way to balance the load between available logical gateways, by starting simple VMs where new applications or BETaaS features can be executed (like VMs for Big Data storage and/or computation capabilities). From the applications deployment perspective, the usage of virtualization increases security by executing the applications in an isolated environment, protecting the instance from dangerous or harmful operations and protecting applications from other applications and services.

From the platform management perspective, virtualization facilitates the provision of high QoS levels and robustness to the platform itself and the applications executed on it, as well as the balance of energy consumption between gateways. Light VMs may be started and shut down depending on the context (devices capabilities, battery, etc…) and current load, having a positive impact in the whole system performance and scalability. Taking into account that some studies and benchmarks say that performance using virtualized Operating Systems (OSs) fells only around 1.5%-2.8% using KVM [2], this loss of performance is a trade-off we assume, as we consider the benefits we obtain are bigger.

Finally, as mentioned before, in case many resources are needed in a concrete moment and an Internet connection is available, it is possible to use external Cloud platforms in order to use more resources.

B. *Local and Remote Virtualization*

Current implementations of certain hypervisors support virtualization on ARM based devices, enabling the possibility to virtualize on top of devices which are becoming popular. Since the BETaaS platform aims at exploiting as much as possible the local capabilities, the preferred option is to use the local resources provided by gateways. Although some local elements, such as Cloudlets, may be very powerful, when thinking about virtualization requirements, we must face the fact that gateways could have high hardware restrictions.

Since the gateways can be modest and limited devices, high performance calculations using things data can be unaffordable in some cases. The way to solve such situations is to use external Cloud platforms in order to externalize computation and storage under certain circumstances. Our objective is to minimize their usage, since it may increase the cost of operating the platform, it may be a security risk (due to the need to move big amounts of data to hosts lacking full transparency) and it could have some impact in the performance. Therefore, the process defined in Figure 2 aims at finding the best choice depending on the circumstances.

Once a VM is to be deployed, there is a process of information retrieval, so all the gateways provide the available resources, current VMs running and potential connection to a remote Cloud. With all the information, BETaaS decides to allocate local or remote resources.

VMs will be deployed in the remote Cloud only in case the instance cannot provide the resources required and in case resources and cost limits configured for remote Clouds do not constraint its usage. Eventually, in the case that what the instance needs is a computation node for the Big Data analysis, even if there are enough resources locally, if the gateway is not powerful, the instance will use the remote Cloud for deploying the VM with the computation node, leaving local resources for storage nodes, as a powerful remote node could execute data analytics more efficiently. This can be applied also to those VMs created for applications deployment, although we take into account that applications consuming a lot of local data will benefit from its execution in the local environment. While VMs requiring more computation can benefit from powerful remote infrastructures, storage nodes may obtain advantages when managing data, since data is created in the local environment and sending and receiving large amounts of data frequently through the Internet will decrease the performance.

According to these principles, we have determined that VMs for storage nodes and light applications deployment will have priority over those VMs for computation nodes and heavy applications deployment, in case the instance is running out of resources or in case computation requirements are very high.

Moreover, in order to predict the usage of resources before taking a decision, we apply a Double Exponential Smoothing (also known as Holt's Linear method [3]) for determining the amount of storage, CPU and memory usage in the following hours. The Triple Exponential Smoothing (Holt-Winters) would require to store and process too much historical data about past resources usage while not providing much better predictions exploiting seasonality, so it is not being used.

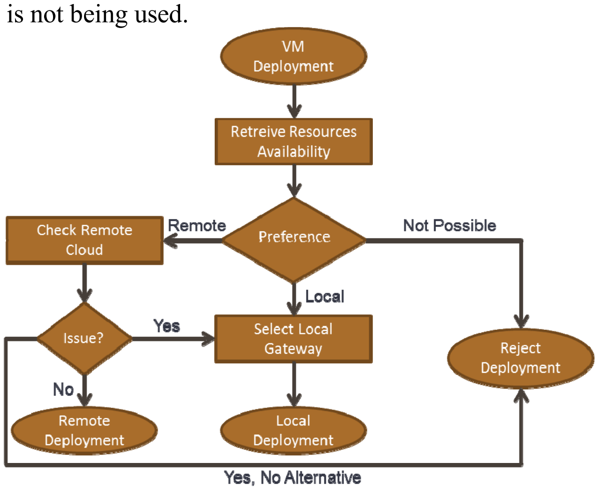

Figure 2. VMs Allocation Algorithm.

If the local deployment is selected as the best option, there will be a process for determining the most adequate gateway for hosting the new VM. In case the remote deployment is selected, there will be a check of the remote Cloud conditions and availability (the Internet connection might be down). In case there is an issue and the local deployment is an option, it will be performed.

It may happen that there are not enough resources available at all or that the remote deployment cannot be done and the local one is not an option. In these cases, the VM deployment will be cancelled. It is possible to minimize these cases by stopping and deleting VMs as soon as they are not necessary, releasing resources for future VMs.

C. *Platform Architecture*

The virtualization feature in local clouds is managed by BETaaS Platform, by means of several components which are part of its core architecture. This core architecture (deployed in each BETaaS Gateway) [4] is divided in three layers: *Service Layer* (the top layer offering APIs to applications for their installation and management), *Things as a Service (TaaS) Layer* (layer in charge of the management of resources, which provides things as services and controls their allocation to applications) and *Adaptation Layer* (the closest layer to the physical world, acting as a bridge between the things and the BETaaS Platform).

While the TaaS Resources Manager is in charge of the resources allocation, the component responsible of managing the lifecycle of the VM resources is the VM Manager. Both of them are located in the TaaS layer and interact with other components such as the Big Data Manager (which provides functionalities to the Service and TaaS layers), the Service Manager (located in the Service layer only) and the Dependability Manager (also, in the Service layer).

Since the architecture has been designed to work in a distributed way, gateways located in the same instance share information about their resources availability and can request resources from others. Therefore, the TaaS Resources Manager is able to determine the best deployment strategy.

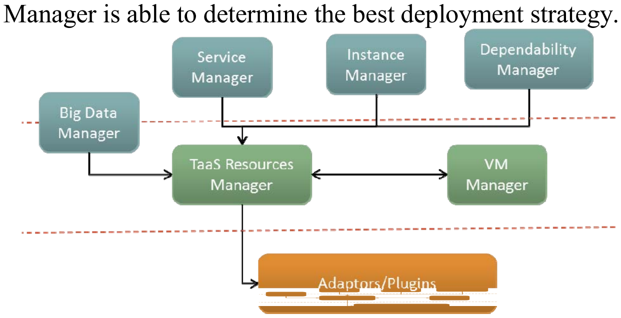

Figure 3. BEtaaS Gateway Architecture.

The VM Manager interacts with a hypervisor and controls the creation, deletion and other managing tasks related to the VMs (i.e. monitoring), both locally and remotely. Its is based on the usage of pre-defined images (called VM templates or flavors), available in the instance gateways and in a BETaaS public server.

While the Service Manager and the Big Data Manager are components merely asking for resources to the TaaS Resources Manager (for deploying a web application or providing resources for data analytics), the Dependability Manager supports the operation of the platform, detecting and solving availability issues in components and VMs.

Finally, the Instance Manager component takes care of the BETaaS instances formation. This component is the one managing these 'local clouds', since it controls the join procedure for new gateways and the dismiss procedure of old ones because of problematic or malicious behavior (using information about detected failures and trust calculations).

All the architecture components are implemented as OSGi bundles, so they can be deployed in any OSGi container easily.

III. LOCAL ALLOCATION OF RESOURCES

Every time a new VM is to be created, the preferred option will be to use one of the local gateways, allocating the local resources needed so the VM can run without problems or without decreasing the performance given to the end user.

Firstly, the VM Manager will retrieve available resources in the local gateway, sending them to the TaaS Resources Manager, which will request the same information to other gateways. As we know the minimum resources needed for running each type of VMs, available resources will be compared with required resources by the VM to be deployed.

We try to avoid overloading a concrete gateway so, if possible, VMs will be deployed in different gateways. The problem is to determine whether other VMs could be needed in the future. In order to do so, we will compare the prediction of resources available (with the mentioned Holt's Linear method) with resources required by the registered

VM template consuming the higher amount of resources, trying to guarantee that it would be possible to deploy one of those VMs if required in the future.

This means that we may have two gateways (A and B) with available resources, and a VM running in gateway B. In principle, the preferred option will be to deploy the new VM in gateway A (if there are enough resources). In case resources available in gateway B are not enough for deploying the most expensive VM template (in terms of computation, memory and storage), but are enough for running the required VM, we will prefer to deploy the new VM in gateway B, leaving gateway A available, in case we need to deploy new VMs of the most expensive type. In case there is no other VMs running, resources available are similar or there are no issues for deploying bigger VMs, the most powerful gateway will be selected. This solution reduces resources fragmentation, which will be translated in better allocation of resources and more performance, avoiding subsequent migration of VMs.

Gateway trust calculations are used as input as well, in order to discard VMs deployment in certain risky BETaaS Gateways. In case trust is low because of potential gateway failures, low battery levels (which may compromise gateway availability), networking issues or low reputation in general, it is better to avoid that gateway. For instance, a gateway consuming its battery and with low charge, is expected to increase its energy consumption with a new VM, so the battery level will decrease faster and the gateway could be running out of battery soon, not allowing the VM to fulfill its assigned tasks.

Since we want to provide a quite generic solution, we have decided to use libvirt for managing VMs, as an abstraction layer between our VM Manager and the hypervisor. This allows changing from one hypervisor to another one by maintaining the same APIs.

We were planning to virtualize on ARM processors using Xen, but its use requires a more complex host OS handling, since it needs a specific Linux kernel to run. The progress of the ARM virtualization capabilities of KVM [5] has led us to offer a virtualization environment based on KVM, since it is integrated in the Linux kernel and the complexity of installing and maintain a specific kernel is removed.

IV. REMOTE ALLOCATION OF RESOURCES

When deploying VMs in a BETaaS Instance, we need to be sure that resources available are enough for running the VM to be deployed. It may happen that the instance needs a new VM but there are not enough resources or not virtualization capabilities at all. In that case, it is possible to use remote traditional clouds for deploying the needed VMs.

The first step to be done is to enable this possibility in the BETaaS Instance configuration (through its GUI). It is necessary to provide certain information such as the Cloud endpoint (the URL to be used for interacting with the remote Cloud), the provider type (since each provider requires to use a concrete protocol), the VMs limitation (the maximum number of VMs allowed for avoiding issues when scaling up too much) and cloud credentials (for Cloud authentication). Currently, we support Open Stack and Open Nebula Clouds.

Since we use concrete VM templates, it is possible for the user configuring this capability to upload the BETaaS images to the target Cloud provider. These images will be publicly available in the BETaaS website, so they can be imported. The format is specific for hypervisors such as KVM and Xen, so those Clouds using other hypervisors (i.e. VMWare or Amazon EC2) may need to do some transformation of the images before they can be imported.

We interact with remote Clouds through the APIs they provide to start, stop, resume or destroy VMs. Once VMs are created, instead of destroying them, the preferred option for the VM Manager is to stop them, as resuming them is faster, assuming that, once we have requested a VM remotely, the probability of requiring the VM again will be high. If a stopped VM is not used for some time, it will be destroyed.

At this moment, we do not envisage a multi-cloud scenario, in which we could select between several Cloud providers. If more than one gateway provides credentials for a Cloud, we select the first one with available resources.

V. VIRTUAL MACHINES TEMPLATES

We foresee to provide three main capabilities thanks to the usage of virtualization in a BETaaS Instance:

- *VMs for services deployment:* We may use VMs in which a company deploys a service created for the instance, so end users may exploit their devices.
- *VMs for gateways virtualization:* It is possible to have more than one logical gateway for a physical gateway. This way, a physical gateway could show one logical gateway to different BETaaS Instances, so the instances operate independently.
- *Resources for Big Data operations:* Virtualization allows scaling up and down computation and storage nodes, optimizing resources usage and allocation.

Each situation requires VMs with certain capabilities and with certain software. Therefore, the way to use virtualization is based on templates of light VMs. We studied initially the use of Tiny Core Linux [2], a GNU/Linux distribution with ports to several platforms (including ARM) which provides a minimal operating system. Unfortunately, the software packages catalog is very limited. This factor made unfeasible to get a full featured image and, after some research, we decided to use an ARM-adapted version of the Arch Linux Distribution [3], because of its low resource-consumption and the high quantity of packages available.

In the first case, we use an ArchLinux distribution, with a Java VM and Tomcat installed, so it will be possible to deploy web applications in war format, as a way to customize the gateways, according to vendors' interests.

In the second case, the distribution is a basic ArchLinux with Java support and an OSGi container, so the BETaaS platform can be deployed. Specifically, we chose Apache Karaf[4] to handle OSGi bundles, since it is a lightweight

[2] http://distro.ibiblio.org/tinycorelinux/

[3] https://www.archlinux.org/

[4] http://karaf.apache.org/

OSGi container and it provides OSGi blueprint, native operating-system integration, support for Zookeper, etc.

In the third case, we use the ArchLinux distribution with those elements needed for creating storage and computation nodes for Big Data analysis. Some Hadoop components required to be ported to ArchLinux and ARM. The VMs are configured according to the role to be played (computation or storage), using the packages in Cloudera CDH5 platform.

VI. RELATED WORK

The resource constraints of smartphones and tablets, and its increasing popularity have led to research initiatives on topics about computation support using Cloud computing in these environments, also known as mobile cloud computing [6]. Main strategies for exploiting mobile computation are based on code partitioning and offloading of application parts. This allows using resources more efficiently, performing operations faster and reducing the energy footprint. Solutions such as MAUI [7] follow this strategy, although they are not yet fully exploiting virtualization (they execute application chunks in Android devices or they use traditional Clouds). Huerta-Canepa developed in [8] the concept of virtual mobile cloud computer providers. This work aims to off-load to other mobile devices accessible in the vicinity, so all mobile devices participating form a virtual mobile cloud provider. It uses JVM as virtualization means, and therefore does not achieve similar levels of isolation than our work. STRATUS [9] also aims to create device clouds out of resource pools that include from mobiles, netbooks and laptops to TVs and other consumer electronics. Unlike all later presented, its approach is based on virtualization in certain devices for video encoding/decoding.

Other IoT frameworks provide things connectivity and management, so applications can exploit the APIs they expose. This is the case of frameworks such as Xively [10] and Libelium [11]. Although both offer some services in the Cloud, these are related to data storage and access through traditional Clouds, not exploiting the potential of the local clouds of devices in the environment.

The solution proposed in this paper aims at pushing the mentioned IoT platforms beyond, using not only traditional Clouds, but also local Clouds for increasing capabilities and improving performance and scalability, since the platform can use local and remote resources for adapting its behavior.

VII. CONCLUSIONS AND FUTURE WORK

We have proposed a solution for IoE environments in which it is possible to exploit locality of data created and its analysis by means of virtualization technologies and allocation algorithms for the mobile environment. Thanks to this approach, we can manage VMs locally and remotely, maximizing the efficiency and performance in constrained environments, for utilizing Big Data technologies.

Besides these aspects, there are other benefits associated to this solution, such as the isolated deployment of web applications locally and the virtualization of logical gateways, which can be useful in certain scenarios.

The proposed solution can be improved by exploring the usage of Linux containers and VMs live migration in order to optimize as much as possible the VMs allocation. Moreover, we intend to enable the multi-cloud scenario, so BETaaS could select between several Cloud providers. Finally, although we have not considered network virtualization in our solution, we believe it has interesting applications in this field, so certain sensors would connect only to certain virtual networks, increasing security.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7) under grant agreement n° 317674.

REFERENCES

[1] M. Presser, S. Krco: D2.1: Initial report on IoT applications of strategic interest. The Internet of Things Initiative Project, Report, July 2011.

[2] Major Hayden, “Performance benchmarks: KVM vs. Xen”. June 2014. Online: https://major.io/2014/06/22/performance-benchmarks-kvm-vs-xen/

[3] “NIST/SEMATECH e-Handbook of Statistical Methods”, http://www.itl.nist.gov/div898/handbook/, 2013.

[4] F. J. Nieto, “An Architecture for a Platform Providing Things As A Service” in Proceedings of the 16th International Symposium on Wireless Personal Multimedia Communications (WPMC 2013), Atlantic City, NJ, USA, June 24-27, 2013.

[5] C. Dall and J. Nieh, Supporting KVM on the ARM architecture. LWM.net [online]. July 2013. Available from internet: https://lwn.net/Articles/557132/

[6] Dinh, Hoang T., et al. "A survey of mobile cloud computing: architecture, applications, and approaches." Wireless communications and mobile computing, 2011.

[7] E. Cuervo, A. Balasubramanian, D.-K. Cho, A. Wolman, S. Saroiu, R.Chandra, P. Bahl, Maui: making smartphones last longer with code offload, in: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys’10, ACM, New York, NY, USA, 2010, pp. 49–62.

[8] Gonzalo Huerta-Canepa and Dongman Lee. A virtual cloud computing provider for mobile devices. In Proceedings of the 1st ACM Workshop on Mobile Cloud Computing and Services: Social Networks and Beyond (MCS'10), New York, NY, USA, 2010. ACM.

[9] Minsung Jang and Karsten Schwan. Stratus: Assembling virtual platforms from device clouds. In Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing (CLOUD '11), pages 476-483, Washington, DC, USA, 2011. IEEE Computer Society.

[10] Xively. “Xively REST API”. 2014. Online: https://xively.com/dev/docs/api/

[11] Libelium Comunicaciones Distribuidas S.L., “Wireless Sensor Networks with Waspmote & Meshlium”. November 2013. Online: http://www.libelium.com/development/waspmote/documentation/wireless-sensor-networks-with-waspmote-meshli