Energy efficiency in Mobile Cloud Computing Architectures

Thinh Le Vinh, Reddy Pallavali, Fatiha Houacine, Samia Bouzefrane

Energy efficiency in Mobile Cloud Computing Architectures

Thinh Le Vinh[1], Reddy Pallavali[2], Fatiha Houacine[1] & Samia Bouzefrane[1]
[1]CEDRIC Lab, CNAM, 292 rue Saint Martin, Paris, France
[2]Dept. of computer science, Andhra Pradesh, India
Email: { le_vi2@auditeur.cnam.fr, pallavali@gmail.com,
samia.bouzefrane@lecnam.net, houcin_f@auditeur.cnam.fr }

***Abstract*— Mobile Cloud Computing (MCC) is an emerging and popular mobile technology which uses fully available Cloud Computing services and functionalities. This technology provides rich computational services to the users, network operators and Cloud service providers as well. However due to users mobility and high computational operations, consumption of energy is a major issue. Energy efficiency over MCC is needed since 57% of generated energy is used by ICT related devices and other negative impacts over environment. This paper investigates different mobile Cloud computing architectures and their performance over energy efficiency by examining different approaches: OSGi, overlay, and container based solutions.**

***Index Terms*—mobile computing, Cloud computing, OSGi, container.**

I. Introduction

Heterogeneous mobile devices or networked objects (from smartphones, laptops and wearable devices to embedded objects) are assumed to have the capability of sharing data [1]: studies point to the existing of more than 50 billion objects by 2020 [2]. The heterogeneity comes from software's, hardware, and architectural point of view as explained in [3]. Nowadays, mobile computing is an essential part of human life which makes daily life more convenient and effective regardless of time and place [4]. However, the mobile devices are facing several challenges over communication networks such as mobility and resources, that is, battery life, storage and bandwidth [5].

Cloud computing is a third party with large-scale storage servers and data centers used to provide infrastructures, software development and distribution platforms with low costs in the computing technology. Cloud computing also enables elastically on-demand services to the users. Hence, mobile applications over Cloud computing can be rapidly provisioned and released with minimal efforts of service providers and management [4]. Therefore, the Mobile Cloud Computing (MCC) paradigm came into picture by combining Cloud computing with mobile environment. MCC provides full Cloud services to its users. This mobile Cloud computing paradigm provides the required support for the creation of cyber-physical systems, which may be used to improve the daily life experience of citizens as well as to bring social and economic benefits.

Mobile networks are suffering from "capacity crunch", meaning that network providers are struggling to meet the demand of mobile data services. To provide Quality-of-Service to the user, the network provider must provide a rich set of services such as, increasing the network capacity and energy efficiency based on the user's mobility patterns.

In the context of MCC, energy efficiency must be considered not only at the Cloud side but also at the mobile side. The former is used to solve the existing limitation of the latter by using remote resource provider rather than processing/storing data locally [6]. In terms of saving energy for mobile devices, the externalization of data and applications is viewed as a good solution. However, this depends on the MCC architecture that is considered. In fact, if the externalization occurs between the mobile device and the Cloud, the application or the data offloading may be costly comparatively to a local mobile computation.

By pointing out the mobile computing concerns, especially energy consumption, the contribution of this paper is to discuss in detail the existing mobile Cloud computing architectures to address the issue.

In this paper, after a related work in Section II, we will investigate the three main MCC architectures and focus on the promising one in terms of energy efficiency in Section III. Section IV presents different offloading models that are suitable with the MCC architecture that we consider here. Section V discusses the conducted experiments using the offloading models to study their energy efficiency, before concluding in Section VI with future works.

II. Related Work

In this section, we present the analysis of energy efficiency over MCC approaches. Since, mobile devices have limited computational resources; there is a need for offloading of computations to the Cloud. Offloading is a process of migrating computations to more resourceful systems like Cloud environments for processing and retrieving the results to mobile devices. Therefore, this section provides the analysis of offloading schemes with their limitations towards energy efficiency.

An adaptive offloading scheme was proposed, in [7], to search for a resourceful Cloud server with a critical value on-demand for a specified mobile device. In this approach, the computations take place in a Cloud server and the results are sent to the mobile device in order to save energy while improving the mobile device performances. However, the

978-1-5090-3946-3/16 $31.00 © 2016 IEEE
DOI 10.1109/W-FiCloud.2016.72

server selection phase required for offloading should be done before computations take place in the mobile device because it consumes bandwidth and other additional resources, especially when data type is audio or video.

In [8], the authors used several communication technologies for offloading the data between mobile devices and the Cloud. Several task execution mechanisms were considered such as local, offloading from wearable to smartphone or Cloud. The Wi-Fi based communication gives better results than LTE technology in terms of time consumption. However, there is no single optimal solution based on their conclusions.

Because the above approaches considered offloading mechanisms without addressing the overhead of components for migration at runtime, the authors of [9] proposed a distributed Energy Efficient Computational Offloading Framework (EECOF) for MCC. This framework migrates computationally intensive components to the Cloud centers at runtime. This has led to the reduction of data transmission rate and energy consumption while offloading computations to MCC. However, it suffers from bandwidth consuming when the data seamlessly need to be transferred between devices and the Cloud. In [10], a dynamic resource provisioning scheduler was proposed for MCC offloading that jointly minimizes the consumption of energy over computations and communications. The authors considered parameters like execution time, good put and bandwidth usage to evaluate the proposed algorithm with existing approaches. This scheduling (dynamic load balancing and online job decomposition) can be done by suing internet based virtual data centers for adaptive resource management. However, this model is based on TCP/IP. Phone2Cloud [11] is another energy saving method by offloading mobile application computations to the Cloud. It offloads all or part of running applications to the Cloud to reduce energy consumption and execution time while improving user's experience (i.e. users delay tolerant experience). However, smartphone energy was heavily consumed when it tries to predict face finder with the data increase. In addition, considering the delay tolerance as always constant is not possible in real world settings. In [12], the authors designed a mathematical model for representing optimization problem for energy efficiency. They proposed a free sequence protocol that allows dynamic execution of applications based on the combination of clients and servers. They used a compression technique to transmit data to the Cloud. If the computation is small then it takes place in local mobile only. They also found that Wi-Fi is the better option than the 3G since 3G consumes more battery and bandwidth than Wi-Fi. Moreover, there is an increase in energy consumption through this dynamic method as authors explained.

In order to analyses the energy efficiency of mobile Cloud computing depending on the interaction models that are used, in the rest of this paper: we will first investigate the different architecture models of MCC before focusing on the Cloudlet model that is a promising architecture. We will then identify different interaction models in the Cloudlet context and analyze the conducted experiments to show the most suitable models in terms of time execution and energy consumption from the mobile point of view. The interaction models include offloading mechanisms as well as client/server models.

III. MCC Architectures

The major benefit of Cloud Computing for mobile devices is the ability to run applications between resource-constrained devices and Internet-based Clouds [13]. Hence, resource-constrained devices can outsource computation/communication/resource intensive operations to the Cloud. The principal motivation of offloading is to achieve less execution time and less energy consumption within mobile devices. This section aims to highlight the principles behind three architecture models of MCC.

A. Cloud Server

Mobile Cloud computing aims to prolong the capabilities of storage/computation-limited devices and to provide seamless access to data/application on a remote resource rich server from anywhere. A remote Cloud server acts as a service provider to mobile devices. The network connectivity from the device to the Cloud server needs to be optimized to ensure the quality of service and seamless handover. With the term of "No Cloud without Virtualization", to reduce the processing time and improve the efficient energy, there are many existing solutions that support this architecture by using the virtualization technique such as Virtual Machines-based, Container-based virtualization.

B. Virtual Cloud

Another approach [14] is to build up a Cloud with peer-to-peer connected mobile devices for data storage and processing so that mobile devices are resource providers of a virtual Cloud. In this architecture, the mobile devices work as either service providers or consumers, and as such each mobile device can cooperate with its neighbors to collect/distribute its surrounding information for specific purposes such as traffic app, healthcare monitor, etc.

C. Cloudlets

The concept of Cloudlets as proposed in [15], is an alternative architecture of MCC in which, the Cloudlet is represented as intermediate offload elements in three-tier structure: mobile device- Cloudlet- Cloud. The Cloudlet is considered as resource-rich, well-connected and powerful computer installed in the public infrastructure with the connectivity to Cloud server. It can be implemented thanks to Wi-Fi hotspot servers that support hypervisors to manage VMs, or can correspond to powerful base stations in a mobile edge computing. Communication between the Cloudlet and Cloud is established only during setting-up and provisioning. This is useful for the proximate mobile device to offload its workload while ensuring low delay and high bandwidth. Simanta et al.,

in [16], present a reference architecture based on Cloudlets as part of *Elijah* project. The principle of Cloudlet approach is based on the overlay notion that will be discussed in the next section.

IV. Implementation Models for Cloudlet Based Architecture

Currently, the trend is to bring the information closer to the mobile user as in mobile edge computing or fog computing. Hence, the Cloudlet concept may be appropriate to provide proximate services with a high bandwidth rather than the remote Cloud. In this context, the challenge is how to interact efficiently with the Cloudlet to minimize the battery life of the mobile device.

In this paper, we focus on the Cloudlet based architecture because through it, it is easy to investigate the energy efficiency according to many scenarios.

Regarding the interaction, the mobile device in MCC can use one of the following methods to get a required service: 1) perform a remote call to the Cloudlet server or 2) offload a piece of code that will be executed on the Cloudlet. To allow the execution of offloaded mobile applications on the Cloudlet server despite their mobile platform dependency, some offloading approaches propose solutions that are independent from the mobile platform. In the following, we will explore three interaction models. The first discussed architecture relies on OSGi framework proposed by [17] for MCC architecture. The second approach is based on the overlay notion proposed by [18] to avoid heavy offloading. The third studied solution is to offload a layer of a Docker Container from the mobile device to the Cloudlet.

A. OSGi based approach

Through Open Service Gateway initiative (OSGi) platform, Houacine et al. in [17] demonstrated the possibility of building a Cloud with mobile devices such as Android smartphones to provide basic functions and services. In the OSGi approach, the OSGi framework provides an environment for the modularization of applications into smaller components called bundles. These bundles can be either executed on the mobile device or on the Cloud/Cloudlet. In addition, the OSGi framework needs to be installed in both mobile device and Cloud side. The former has a service consumer which is used to handle interaction with mobile apps that import and consume remote service offered by the Cloud servers. A service provider has been installed on the latter to implement and export services. Figure 1 illustrates the interaction between a mobile device and the Cloud/Cloudlet using OSGi framework. To address the inter-OSGi framework communication issues, they adopt an XMPP (Extensible Messaging and Presence Protocol) based solution. To integrate XMPP service within Android based framework, a signaling and communication agent bundle has been developed within Felix a lightweight implementation of OSGi. The mobile device implements a discovery bundle while the Cloudlet runs a service advertisement bundle. Each bundle interacts with an internal XMPP bundle.

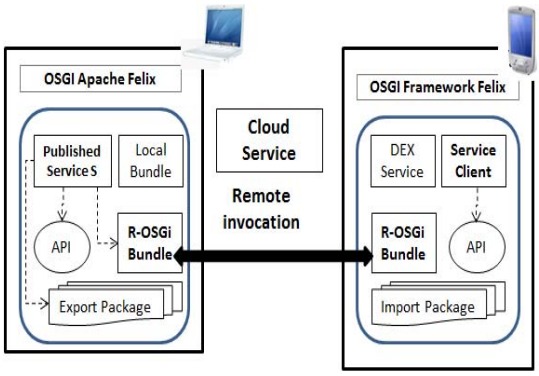

Fig.1. *OSGi based Cloudlet*

B. Overlay based approach

While the OSGi-based solution requires the installation of OSGi framework, Ha et al., in [18], introduced the implementation of the concept of Cloudlet by using an application overlay without the need of a pre-requisite environment. In this approach, the VM overlay refers to the compressed binary difference between a base VM image and a complete VM image. The complete VM image is a base VM in which the overlay application is installed. The mobile user carries only the overlays that can be either calculated offline or obtained from the Cloud via Cloudlet. As presented in Figure 2, when a mobile device is connected to a Cloudlet, an application overlay, such as augmented reality, face or object recognition, is offloaded to the Cloudlet instead of offloading a VM which is too heavy for a transfer. In the Cloudlet side, a VM instance is created from the received overlay and a base VM possessed by the Cloudlet. This process is called VM synthesis. The mobile device, consequently, can use this VM instance for its offload operations.

Fig.2. *VM Overlay in Cloudlet architecture*

C. Container based approach

The overlay-based solution, despite its flexibility, assumes the existence to a third party, like a Cloud or a Cloudlet, to run overlay applications. In fact, there is no way to run the overlays locally on the mobile device in case of bad connectivity. To overcome this drawback, we think that a containerization approach like Docker solution can overcome this drawback. Technically, we can define a container as a confined environment under the global environment. The Linux operating system provides some mechanisms based on

namespaces to create these confined environments or containers. The containers are isolated like VMs but are lightweight environments managed by an engine which is a part of the native OS. In BYOD[1] concept where a mobile device can host two distinct environments one for corporate purpose and another for personal needs, the container based approach is suitable in this context. It isolates the environments within two distinct containers making their data, sensitive or not, inaccessible and invisible from the other container. A mobile container includes the necessary environment (basic libraries) to execute the application that needs offloading. The advantage of offloading containers is that there is no need to operate a container synthesis unlike VM synthesis that is required with overlay based solution. However, offloading a container may be energy and bandwidth consuming. In Docker container technology for example, images are the result of a recursive mounting of different image layers. Each image layer has a parent image layer except for the root image layer. Hence, instead of transferring the whole heavy container (which is around 600 MB in our experiments), a specific small size layer is transferred from the mobile device to the Cloudlet.

In the next section, we will discuss the experiments we conducted based on the three approaches presented in this current section, while comparing the energy efficiency among these architectures to pinpoint their merit and demerit.

V. Experimentation and Results

In order to study the energy efficiency of the mobile device when using Cloudlet architecture, we conducted some experiments by considering different approaches: OSGi, overlay, and container based solutions.

To experiment on this work, we have used two devices to implement three discussed Cloudlet based architectures. Performance comparisons is conducted on a LAN (Local Area Network) to overcome Internet WAN latencies that are common to all solutions.

- Cloudlet side: Linux Ubuntu 14.04 LTS <64 bits> (CORE i7)
- Mobile device side: Windows 7/Linux-based OS <32 bits> (DUAL CORE)

(1) *Elijah project based on overlay notion.* The application overlay is calculated as the binary diff (VCDIFF RFC3284) using xdelta3 tool [19], between the complete VM disk image and the base VM disk image. The base VM is then deployed to any platform that will serve as a Cloudlet. We assume that we have a VM overlay in the mobile device side. In the Cloudlet side, a VM instance is rebuilt by this overlay. The mobile device connects to the Cloudlet upon detecting its IP address, using the program "synthesis_client" and supplies the VM overlay. When the mobile device sends the VM overlay to the Cloudlet, the Cloudlet server starts performing the VM synthesis operation.

[1] Bring Your Own Device

(2) *OSGi.* For the mobile device, the OSGi framework is installed on an emulator with the following characteristics:
Device Nexus S (4.0", 480 * 800: hdpi));
Android 2.2 – API level 8; Ram 343; VM Heap: 32;
Internal Storage: 60MB
Bundle size: 32 KB.
In the Cloudlet side, the OSGi framework is hosted in a specific VM.

(3) *Docker container.* Even though Docker container engine can be installed on Windows operating system, both the mobile device and the Cloudlet operated under Linux-based OS to run Docker containers. In Docker technology, the application is isolated within a container that is managed by a container engine within the mobile device. Instead of offloading the entire container to the Cloudlet, that can be heavy (600 MB in our experiments) in size and energy consuming, we offload only the layer that hosts the application.

To compare the difference among these discussed Cloudlet implementations, we consider calculating the execution time locally by examining four different sizes of programs as presented in table 1.

We then compare these programs under OSGi, Elijah and Docker platform to measure their execution time. In this test, communication is established via Wi-Fi to access to the Cloudlet server. Figure 3 presents time in seconds for downloading and launching the overlay for Elijah, OSGi bundle, and Docker layer.

	Program size			**Execution time on computer**
	OSGi	**Elijah**	**Docker**	
Program 1	3.8 kb	4.96 kb	4.96 kb	~30 s
Program 2	1.22 Mb	1.22 Mb	1.22 Mb	~30 s
Program 3	7.922 Mb	7.923 Mb	7.923 Mb	~30 s
Program 4	15.843 Mb	15.844 Mb	15.844 Mb	~30 s

Table 1.*Size and execution time of the tested programs*

Generally, increasing size of program leads to a rise of execution time for all models. Although transferring an overlay via high speed LAN Wi-Fi, however, Elijah takes more time in comparison with the others especially soar to double time in Program 4 which is the heaviest program. The concept of a VM overlay is similar to copy-on-write virtual disk files or VM image hierarchies [18]. It means that the overlay solution is time-consuming task because of the heavy overlay offloading.

On the other hand, with installed Java virtual machine and OSGi framework on each node [17], applications supporting OSGi can interact with a proxy bundle which is generated dynamically with the exported methods without transferring any heavy code. As a result, running bundle in OSGi framework is the fastest solution in term of time-consuming. The remaining solution, Docker, has insignificant higher execution time than OSGi due to the layer transferring.

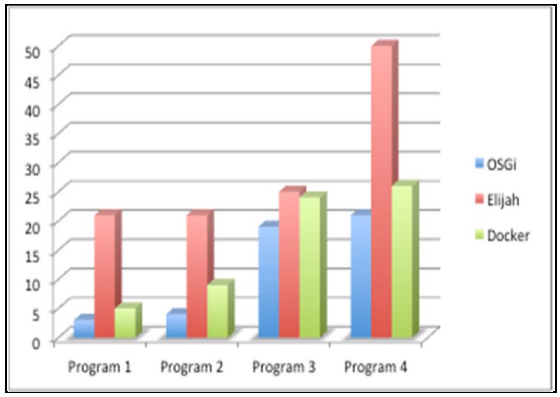

Fig. 3 *Time in second for downloading and launching the overlay, OSGi bundle, and Docker layer*

VI. DISCUSSIONS

From the above analysis, we can clearly say that the mobile Cloud computing aims to allow the mobile user to overcome its heavy works by providing a seamless and rich functionality regardless of the resource limitations of mobile devices. Although mobile Cloud computing becomes the future dominant model for mobile applications, the energy consumption is necessary to consider so that to find a suitable offloading model that is energy efficient. As described in [6], the researchers focused on offloading mechanisms to reduce the computation time and to improve the life time of mobile energy.

In this paper, we discussed the three different MCC architectures and their merits and demerits towards energy efficiency while considering the offloading mechanism. For the efficiency of offloading to enhance saving energy, we consider the following formulas as defined in [20] to apply with our experiments described in Section V.

$\frac{M}{Sm}$ (1): Formula 1 denotes the execution time to run the program locally, where M is the amount of computation and S_m is the speed of the mobile device.

$\frac{D}{B}+\frac{M}{Sc}$ (2): Formula 2 refers to the execution time and the offloading time, where *D, B, and* S_c are respectively the size of input data, the bandwidth, and the speed of the remote server.

We consider two cases:

Case #1: If $\frac{M}{Sm} > \frac{D}{B}+\frac{M}{Sc}$,i.e., if running a computation takes locally more time than offloading it and running it remotely, then the performance of the whole system is improved by using offloading technique.

Case #2: However, if $\frac{M}{Sm} < \frac{D}{B}+\frac{M}{Sc}$, it means that the offloading does not meet the requirement for energy efficiency.

Adapting these cases to the results discussed in Section V, it can be clearly seen that the offloading result for three architectures is almost positive. Take *OSGi based approach* for a specific example, the value of (1) is always greater than the (2)'s value regardless of the program's size. However, this does not mean that the offload method is always a suitable solution in the context of saving energy. The program 4 in *Elijah* has proved this drawback of the offloading.

The energy consumed by the mobile device in each MCC model is mainly composed of:

- *The computing energy:* consumed by the mobile device for execution of local services such as initiation and service request, discovery service, locally executed processes. This energy depends on both the mobile characteristics (computing and battery features) and the performed computing; and
- *The communication energy:* consumed during the I/O operations using mobile network. It depends on the mobile characteristics, the network characteristics and the size of data to be transferred.

The local computing energy is calculated as in formula (4).

Computing Energy **Service** = *Mobile Capacity x Time* **Service** (4)

Where:

- *Computing Energy* Service: is the local energy consumed by a service S (in Joule or kWH).
- *Mobile Capacity:* is the battery capacity of the mobile device (in milliampere or Joule).
- *Time* Service: is the time of the service execution.

Nexus S (4.0", 480 * 800: hdpi)) is used for our experimentation with android 2.2 system. The energy capacity of this device is 19152 Joules (1440 mAH).

Table 2 shows the consumed energy measured for each tested solution.

Solution	Execution Time ms	Mobile Capacity mAH	Consumed Energy joules
OSGi	30	1440	0,1596
Elijah	79	1440	0,42028
Docker	30	1440	0,1596

Table 2.Consumed energy for program execution

As shown in Figure 4, increasing the time of program execution leads to a rise of the consumed energy on the same device. The launching part (bundle or VM start) is less consuming than the program execution part. For the same program, Elijah consumes more energy in comparison with the other solutions.

For the ubiquitous environment for the MCC, crossing architectures also need to be considered. It may lead to increase network latency and transmission rate. As can be seen in Section V, the execution time is based on the application size and the result does not always meet the expectation. For a function partitioning [4], application functions could be determined which part is to be offloaded to the remote Cloudlet and which part is processed locally on the device. For example, the appearance of application is displayed locally, while offloading heavy computations to the Cloudlet. This takes more energy consumption in case of heavy code offloading. Consequently, the offloading technique is not always considered as a good solution for MCC. In fact, the OSGi based solution that relies on Client/Server interaction without

offloading mechanism can confirm that offloading is not always the suitable solution in terms of energy efficiency.

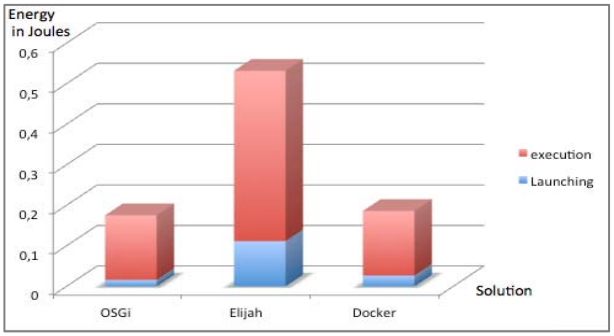

Fig.4. Service energy consumption

The next section concludes our analysis and includes future works towards MCC architectures.

VII. Conclusion

The shortage of energy is still one of the crucial challenges in mobile world. It makes the researchers spend time to mull it over before delivering new approaches. For this reason, this paper analyses the energy efficiency over three mobile Cloud computing architectures by examining into different approaches: *OSGi, overlay, and container based solutions*. To look through these approaches, we also discuss how the offloading gains an advantage in MCC. From the performance analysis of our experiment, we conclude that the constraint energy of the mobile device is overcome by adapting the benefits of Cloud computing, regardless of remaining some exception cases. Mobile Cloud computing will continue to be the trend of technology in computing environment. Hence, we plan to build a new approach which enables offloading while addressing security issues so that to improve the work initiated in [21].

References

[1] *R. H. Weber and R. Weber, Internet of Things. Springer, 2010.*

[2] *C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," IEEE Communications Surveys and Tutorials, vol. 16, no. 1, pp. 414–454, 2014.*

[3] *Schmohl, Robert, and Uwe Baumgarten. "Heterogeneity in Mobile Computing Environments." In ICWN, pp. 461-467. 2008.*

[4] *Dinh, Hoang T., Chonho Lee, Dusit Niyato, and Ping Wang. "A survey of mobile Cloud computing: architecture, applications, and approaches." Wireless communications and mobile computing 13, no. 18, 1587-1611, 2013.*

[5] *Satyanarayanan, Mahadev. "Fundamental challenges in mobile computing." In Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing, pp. 1-7. ACM, 1996.*

[6] *Fernando, N., Loke, S.W. and Rahayu, W., 2013. Mobile Cloud computing: A survey. Future Generation Computer Systems,* Volume 29, Issue 1*, January 2013, Pages 84–106.*

[7] *Wu, Huaming, Qiushi Wang, and Katinka Wolter. "Tradeoff between performance improvement and energy saving in mobile Cloud offloading systems." In Communications Workshops (ICC), 2013 IEEE International Conference on, pp. 728-732. IEEE, 2013.*

[8] *Ragona, Claudio, Claudio Fiandrino, Dzmitry Kliazovich, Fabrizio Granelli, and Pascal Bouvry. "Energy-Efficient Computation Offloading for Wearable Devices and Smartphones in Mobile Cloud Computing." In IEEE Global Communications Conference, San Diego, CA, USA, 2015. 2015.*

[9] *Shiraz, Muhammad, Abdullah Gani, Azra Shamim, Suleman Khan, and Raja Wasim Ahmad. "Energy efficient computational offloading framework for mobile Cloud computing." Journal of Grid Computing 13, no. 1 (2015): 1-18.*

[10] *Shojafar, M., Cordeschi, N., Abawajy, J.H. and Baccarelli, E., 2015, December. Adaptive Energy-Efficient QoS-Aware Scheduling Algorithm for TCP/IP Mobile Cloud. In 2015 IEEE Globecom Workshops (GC Wkshps) (pp. 1-6). IEEE.*

[11] *Xia, Feng, Fangwei Ding, Jie Li, Xiangjie Kong, Laurence T. Yang, and Jianhua Ma. "Phone2Cloud: Exploiting computation offloading for energy saving on smartphones in mobile Cloud computing." Information Systems Frontiers 16, no. 1 (2014): 95-111.*

[12] *Saab, S.A., Saab, F., Kayssi, A., Chehab, A. and Elhajj, I.H., 2015. Partial mobile application offloading to the Cloud for energy-efficiency with security measures. Sustainable Computing: Informatics and Systems, 8, pp.38-46.*

[13] *T. Le Vinh, S. Bouzefrane, J. Farinone, A. Attar, B. Kennedy. "Middleware to Integrate Mobile Devices, Sensors and Cloud Computing", The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015)), June 2015,* Vol. *25,* pp.*234–243, London*

[14] *G., Huerta-Canepa, & D., Lee, "A virtual Cloud computing provider for mobile devices", in: Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond, MCS'10, ACM, New York, NY, USA, 2010, pp. 6:1–6:5.*

[15] *M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The case for VM-based Cloudlets in mobile computing, IEEE Pervasive Computing 8 (2009) 14–23*

[16] *S. Simanta, K. Ha, G. Lewis, E. Morris, and M. Satyanarayanan, "A reference architecture for mobile code offload in hostile environments," in International Conference on Mobile Computing, Applications, and Services, 2012, pp. 274–293..*

[17] *F. Houacine, S. Bouzefrane, A. Adjaz. "Service Architectures for multi-environment Mobile Cloud Services", International Journal of High Performance Computing and Networking, pp. to appear, 2016*

[18] *Ha, K., Pilai,P., Richer, W., Abe, Y., & Satyanarayanan, M(2013). Just-in-time provisioning for cyber foraging (pp. 153-166). Mobisys.*

[19] *Xdelta.org, xdelta.org*

[20] *K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A Survey of Computation Offloading for Mobile Systems," Mob. Netw. Appl., vol. 18, no. 1, pp. 129–140, Feb. 2013.*

[21] *T. Le Vinh, S. Bouzefrane. "Trusted Platforms to secure Mobile Cloud Computing", The 16th IEEE International Conference on High Performance Computing and Communications, August 2014,* pp.*1096-1103.*