Adaptive Code Offloading and Resource-intensive Task Delegation for Mobile Cloud Applications

Huber Flores
huber@ut.ee

Satish Nayarana Srirama
srirama@ut.ee

Institute of Computer Science, Mobile Cloud Lab
University of Tartu, Estonia
J. Liivi 2, Tartu

ABSTRACT

Mobile cloud computing is arising as a prominent domain that is seeking to bring the massive advantages of the cloud to the resource constrained smartphones, by following a delegation or offloading criteria. In a delegation model, a mobile device consumes services from multiple clouds by following their Web API. In the offloading model, a mobile application is partitioned and analyzed so that the most computational expensive operations at code level can be identified and offloaded for remote processing. This paper proposes a mobile cloud architecture that enhances the decision of offloading process with asynchronous cloud processing that targets cloud infrastructure and code offloading traces, also taking advantage of the middleware based delegation model.

1. INTRODUCTION AND MOTIVATION

Mobile cloud computing (MCC) is arising as a prominent domain that is seeking to bring the massive advantages of the cloud to the resource constrained smartphones. Mobile devices are looking towards the computational and storage infrastructure of clouds, driven by their growing interest to provide ubiquitous PC-like functionality to end users. Mobile cloud applications [3] are considered as the next generation of mobile applications, due to their promise of linked and elastic computational cloud functionality that enables to augment their processing capabilities on demand, power-aware decision mechanisms that allow to utilize efficiently the resources of the device and dynamic resource allocation approaches that allow to program and utilize cloud services at different levels (SaaS, IaaS, PaaS).

Mobile applications may be bonded to cloud resources by following a delegation or offloading criteria. In a delegation model, a mobile device may utilize the cloud to perform resource-intensive operations which are time-consuming, programmable and parallelizable among multiple servers, and computationally unfeasible for offline devices (e.g. MapReduce). This kind of operations require mostly middleware support [4]. In contrast, in an offloading model, a mobile application may be partitioned (e.g. methods, classes) and analyzed *a priori* (at development stages) or *a posteriori* (at runtime) so that the most computational expensive operations at code level can be identified and offloaded for remote processing. A mobile operation may be offloaded or not, depending on the impact of its execution over the mobile resources. Conceptually, offloading is preferable only if a mobile operation requires high amounts of computational processing and the same time, low amounts of data need to be sent in the communication. Unlike delegation, offloading does not require online connectivity as it can be executed by the mobile in standalone. However, offloaded mobile operations require a low-level scale processing compared with delegated mobile tasks.

In the case of mobile delegation, current solutions try to overcome the problems of multi-cloud service integration in mashups that can be accessible from the handset [9]. However, they just focus on primitive services at SaaS level, which are not data-intensive. In contrast, most of the research works have proposed solutions to bring the cloud to the vicinity of a mobile [3, 2, 1, 5, 7] from an offloading perspective, and thus addressing partially the issues of determining *what*, *when* and *how* to offload from mobile to cloud. Basically, these solutions foster an architecture, in which, the cloud provides the virtual computational resources and the mobile introduces the partition strategies (e.g. static analysis), the decision logic based on local parameters (e.g. network bandwidth) and the basic implementation primitives (e.g. annotations) that enables to synchronize a mobile application stack with a virtual machine running in the cloud. However, given this context, from both points of view, we can argue that much of the advantages of cloud computing are left unexploited and poorly considered.

2. CHALLENGES

From an offloading perspective, cloud computing may introduce many other dynamic variables (e.g. performance metrics) that could affect the overall offloading decision process. Consequently, a code offloading model should not just target mobility aspects, but also target oscillating changes in cloud infrastructure. Moreover, we think that a mobile cloud architecture must be one that not just increases the storage and computational functionalities of the smartphones when communication is suitable, but rather, it periodically enhances the mechanisms of the devices with offline cloud analysis (e.g. exploration of code offloading traces) delivered asynchronously using cloud-based mechanisms such as notification services, so that each time a device may have opportunity to interact with the cloud again, the handset does it better using refined profiling strategies that allow to increase performance and save energy.

On the other hand, from a delegation perspective, hybrid cloud and cloud interoperability are essential for mobile scenarios in order to foster the de-coupling of the handset to a specific cloud vendor, to enrich the mobile applications with the variety of cloud services provided on the Web and to

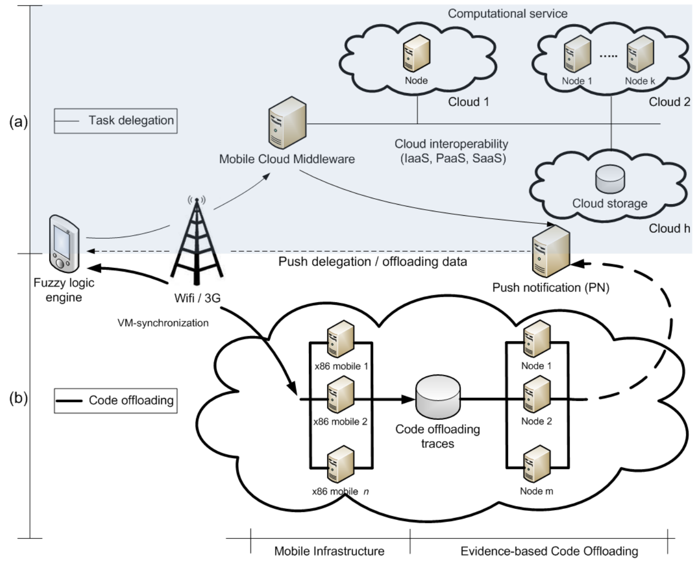

Figure 1: Resource-intensive task delegation and evidence-based code offloading: Architecture

create new business opportunities and alliances. However, developing a mobile cloud application involves adapting different Web APIs from different cloud vendors within a native mobile platform. Deploying a Web API on a handset is demanding for the mobile operating system due to many reasons like compiler limitations (e.g. Dalvik limited java features), additional dependencies, code incompatibility etc.

3. INITIAL RESULTS AND HYPOTHESIS

We have studied the delegation of mobile tasks (figure 1a) to hybrid clouds in detail and we have developed a Mobile Cloud Middleware [4] framework (MCM) that addresses the issues of interoperability across multiple clouds, transparent delegation and asynchronous execution of mobile tasks that require resource-intensive processing, and dynamic allocation of cloud infrastructure. Moreover, we have developed several successful study cases of data-intensive mobile cloud applications that benefit by going cloud-aware [8, 3].

While our current solution follows a common client-server model, it can be extended to fit into a code offloading architecture as shown in figure 1. In this case, the REST operation may be offloaded at thread level [5] from the handset to the mobile x86 platform running in the cloud. However, the identification of this kind of candidate mobile operations cannot depend exclusively on local logic of the mobile profilers due to it dependence on many infrastructural parameters (e.g. Load of the cloud instance, cluster size, etc) and no deterministic sources.

3.1 Hypothesis

Basically, our hypothesis is that code offloading may fail in some cases, as the information currently being utilized to make the decision is not enough to measure a real benefit for the handset. On the basis of this assumption, we propose a mobile cloud architecture that enhances the decision offloading process of a device with asynchronous cloud processing that targets cloud infrastructure and code offloading traces. This information can be analyzed and transformed into knowledge that can be used by the device to counter issues such as device diversity, adaptive application execution and unpredictable code profiling. The complete architecture is shown in figure 1b and consists of 1) a mobile offloading decision mechanism based on Fuzzy Logic, 2) a virtualized mobile infrastructure (based on Android x86), 3) a repository of code offloading traces along with a cluster to analyze it and 4) a cloud-based messaging framework to push data to the handset. The contributions of this work are envisioned as an approach that enables to enhance the code offloading decision process of a mobile device with cloud power, "Evidence-based mobile code offloading" approach. Moreover, we are also interested in merging our delegation approach in order to provide resource-intensive functionality supported with code offloading.

4. RELATED WORK

Middleware approaches similar to MCM have also been addressed in the literature. MCCM (Mobile Cloud Computing Middleware) [9] is a framework that aims the integration of SaaS services for mobiles. Unlike our MCM, such middleware does not support data-intensive processing and lacks support for asynchronous communication. On the other hand, lot of research have been conducted concerning code offloading [2, 1, 5, 7]. However, most similar approach that can be comparable with our proposed solution is an adaptive offloading mechanism which targets pervasive devices [6], however, does not involve cloud.

5. REFERENCES

[1] B.-G. Chun, Ihm, et al. Clonecloud: elastic execution between mobile device and cloud. In *Proceedings of the 6th Conf. On Computer systems*, pages 301–314, 2011.

[2] E. Cuervo, Balasubramanian, et al. Maui: making smartphones last longer with code offload. In *Proceedings of the 8th Int. Conf. On Mobile systems, applications, and services*, pages 49–62. ACM, 2010.

[3] H. Flores, S. Srirama, and C. Paniagua. Towards mobile cloud applications: Offloading resource-intensive tasks to hybrid clouds. *Int. Journal of Pervasive Computing and Communications*, 8(4):344–367, 2012.

[4] H. Flores, S. N. Srirama, and C. Paniagua. A generic middleware framework for handling process intensive hybrid cloud services from mobiles. In *Proceedings of the 9th Int. Conf. On Advances in Mobile Computing and Multimedia (MoMM)*, pages 87–94. ACM, 2011.

[5] M. S. Gordon, Jamshidi, et al. Comet: code offload by migrating execution transparently. In *Proceedings of the 10th USENIX Conf. On Operating Systems Design and Implementation*, pages 93–106. USENIX, 2012.

[6] X. Gu, Nahrstedt, et al. Adaptive offloading for pervasive computing. *Pervasive Computing, IEEE*, 3(3):66–73, 2004.

[7] S. Kosta et al. Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading. In *INFOCOM*, pages 945–953. IEEE, 2012.

[8] S. N. Srirama, C. Paniagua, and H. Flores. Social group formation with mobile cloud services. *Service Oriented Computing and Applications*, 6(4):351–362, 2012.

[9] Q. Wang and R. Deters. SOA's last mile connecting smartphones to the service cloud. In *IEEE Int. Conf.On Cloud Computing*, pages 80–87, 2009.