MpOS: Uma Proposta para Sistema de Offloading em Múltiplas Plataformas

Philipp B. Costa
Universidade Federal do Ceará
Campus do Pici, s/n, CEP 60451-970
Fortaleza – Ceará - Brasil
philippcosta@great.ufc.br

José Neuman Souza
Universidade Federal do Ceará
Campus do Pici, s/n, CEP 60451-970
Fortaleza – Ceará - Brasil
neuman@great.ufc.br

Fernando A. M. Trinta
Universidade Federal do Ceará
Campus do Pici, s/n, CEP 60451-970
Fortaleza – Ceará - Brasil
fernando.trinta@great.ufc.br

RESUMO

Os dispositivos móveis especificamente os smartphones e os tablets, evoluíram bastante nos últimos anos em termos computacionais, se tornando indispensáveis no uso diário das pessoas. Apesar dos seus avanços de hardware nos últimos anos, a principal limitação desses aparelhos é relacionada com a questão energética. Deste modo, com base nesse contexto, surgiu o paradigma do Mobile Cloud Computing (MCC), que estuda formas de estender os recursos computacionais e energéticos dos dispositivos móveis, através da utilização das técnicas de offloading. A partir do levantamento bibliográfico dos frameworks em MCC, verificou-se ausência de soluções de offloading voltadas para o problema da heterogeneidade em plataformas móveis. Diante deste problema, este trabalho propõe um framework, para apoiar o desenvolvimento de aplicações móveis para plataformas heterogêneas, que necessitem utilizar do procedimento de offloading de processamento e/ou armazenamento, utilizando um cloudlet como ambiente de execução remoto.

Categories and Subject Descriptors

D.3.3 [**Programming Languages**]: Language Constructs and Features – frameworks, modules, packages.

C.5.3 [**Computer System Implementation**]: Microcomputers – portable devices.

General Terms

Management, Performance, Languages.

Keywords

Mobile Computing, Mobile Cloud Computing, Offloading, Heterogeneity of mobile platform, Cloudlet.

1. CONTEXTO TEÓRICO

Os telefones celulares surgiram com uma atividade bem especifica trazer mobilidade para os sistemas telefônicos tradicionais, que utilizavam os fios como meio físico para transmitir as vozes dos seus assinantes. Com o surgimento da tecnologia do GPRS, as redes de celulares passaram a transmitir, além do canal de voz, um canal de dados para Internet [1]. Os telefones celulares, ao longo dos anos, deixaram de ser simples dispositivos telefônicos, para se tornarem avançados dispositivos de computação móvel, sendo denominados atualmente de *smartphones*. Há também uma outra categoria de dispositivos móveis, denominada de *tablet*, que geralmente tem um *hardware* mais robusto do que os *smartphones*. Apesar das evoluções dos dispositivos móveis, eles ainda são computacionalmente limitados, em relação a um *desktop* ou *notebook*. Entretanto, nos últimos anos, a questão energética tem se destacado como a maior limitação dos dispositivos móveis. Este fato está relacionado com a evolução mais rápida das tecnologias de *hardware* dos dispositivos móveis, se comparada com a evolução tecnológica empregada nas baterias.

Uma das técnicas, utilizadas para reduzir o consumo de energia dos dispositivos móveis, é a execução remota do processamento, também conhecida na literatura como técnica de *offloading* [2, 3]. O aplicativo móvel que utilizar esta técnica estará "terceirizando" seu processamento e armazenamento para ser executado numa outra infraestrutura, podendo proporcionar ganhos de ordem energética, como também aumento no desempenho do aplicativo. De acordo com Verbelen *et al.* [6], essa infraestrutura pode ser composta por máquinas virtuais que estão em execução numa nuvem pública na Internet, ou por máquinas pessoais que residem numa mesma rede local em que estão os dispositivos móveis. Para Satyanarayanan *et al.* [11], esse ambiente de execução, que interage com os dispositivos móveis dentro de uma mesma rede local é denominado *cloudlet*. Um *cloudlet* tem como objetivo melhorar o desempenho dos aplicativos móveis, porque aproxima o ambiente de execução dos usuários, além de diminuir a dependência da Internet para acessar recursos na nuvem.

O paradigma do *Mobile Cloud Computing* (MCC) surgiu nesse contexto e envolve uma combinação de tecnologias em diversas áreas, como computação em nuvem, computação móvel, infraestrutura de comunicação sem fio, dentre outras áreas [3, 4, 5]. Segundo Shiraz *et al.* [5], o principal esforço de pesquisa em MCC está na camada de aplicação, onde esse paradigma estuda as diversas estratégias de *offloading*, para ser adotado durante o desenvolvimento dos aplicativos móveis.

Bahl *et al.* [13] propuseram que a utilização do conceito *cloudlet* são mais evidentes nessa categoria de aplicações, sensíveis a percepção de latência, como aplicativos de reconhecimento de voz, processamento de linguagem natural, visão computacional e gráfica, realidade aumentada e aprendizagem de máquina para tomada de decisão e planejamento.

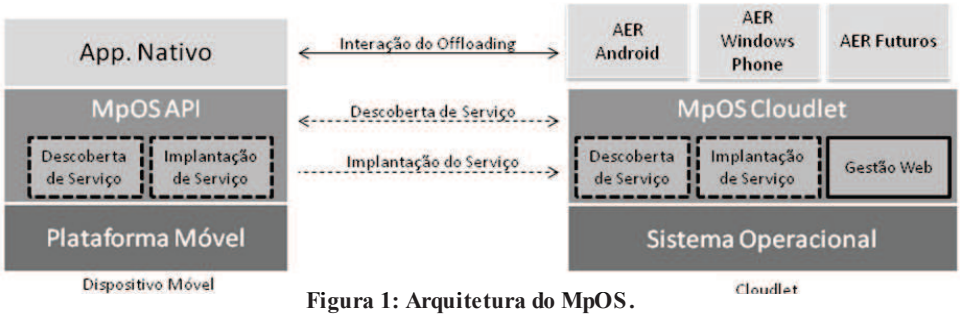

Figura 1: Arquitetura do MpOS.

Na literatura existem diversas soluções [2, 6, 7, 8, 9, 10] que tratam da técnica de *offloading* em diversos níveis de granularidade, que podem variar deste da migração da *thread* até a migração de um processo para o ambiente de execução remota. No entanto, Shiraz *et al.* [5] desenvolveram uma taxonomia para classificar as decisões envolvidas no desenvolvimento de *frameworks* em MCC. Dentre as decisões estão:

1) Qual o parâmetro usado para decidir quando executar ou não uma técnica de *offloading*? O parâmetro pode ser baseado na estimativa do gasto energético desse processamento executado localmente no dispositivo e quanto de energia será consumido para transferir o processamento para um ambiente de execução remoto. Um outro parâmetro, que pode ser utilizado está relacionada com as condições de rede (e.g. latência e largura da banda).

2) Qual o padrão de migração (granularidade) utilizado pelo *framework*? Os *frameworks* podem adotar diversos padrões de migração, baseado na migração de máquinas virtuais, no uso de agentes móveis, na migração de código binário ou na utilização de *Proxy* de aplicação.

3) Qual a abordagem utilizada para o particionamento do aplicativo móvel? Os *frameworks* podem suportar até dois tipos de particionamento, sendo o primeiro tipo conhecido por estático, pois um determinado segmento do aplicativo (que é marcado pelo desenvolvedor) é executado remotamente. Enquanto o segundo tipo é conhecido por particionamento dinâmico, no qual o *framework* decide automaticamente, de acordo com as condições no momento da execução do aplicativo, se vale a pena ou não migrar certo processamento para execução remota.

2. IDENTIFICAÇÃO DO PROBLEMA

Durante a revisão da literatura [2, 6, 7, 8, 9, 10], foram encontrados somente soluções que realizavam a operação de *offloading* em *Mobile Cloud Computing* usando tecnologias especificas de plataforma móvel e de ambiente de execução remoto. Por isso, que a questão sobre o desenvolvimento de *frameworks* que realizam a operação de *offloading* em plataformas móveis heterogêneas é considerada ainda um desafio em aberto para o paradigma de MCC [5, 12].

Portanto, considerando o cenário em que, uma empresa precisa desenvolver um aplicativo ou um serviço móvel, multiplataforma e que tenha suporte à técnica de offloading, essa empresa terá que utilizar atualmente pelo menos duas soluções distintas de *framework* em MCC. Sendo assim, é desejável o desenvolvimento de um *framework* que trate da questão da heterogeneidade das plataformas móveis, a fim de simplificar o desenvolvimento, o gerenciamento e a implantação dos componentes do aplicativo móvel num ambiente externo.

3. PROPOSTA DO FRAMEWORK MpOS

Esse trabalho de mestrado pretende desenvolver um *framework* para apoiar o desenvolvimento de aplicações móveis, que desejem suportar a técnica de *offloading* em relação ao processamento e/ou armazenamento dos dados, denominado de MpOS (Multiplatform Offloading System). O trabalho planeja desenvolver esse *framework* para pelo menos duas plataformas móveis distintas, tendo apenas um único *endpoint*[1] para acessar esses dois ambientes de execução remotos (AER), que estarão instalados num mesmo *cloudlet*.

O *framework* MpOS pretende simplificar o gerenciamento dos ambientes remotos através de uma interface Web. Esse gerenciamento também envolve a atividade de controlar os recursos que podem ser alocados para os serviços no ambiente de execução, como quantidade de CPU, de memória ou espaço em disco. A Figura 1 mostra a interação dos componentes do *framework* MpOS, tanto na parte do dispositivo móvel, como na parte do *cloudlet*.

Nas próximas seções os componentes do MpOS (API Cliente e Cloudlet) serão detalhados, bem como os serviços relacionados a cada componente.

3.1 MpOS API Cliente

O *framework* MpOS interage com aplicação móvel na parte do dispositivo móvel através de uma API, tendo algumas tarefas automatizadas por essa API, como Descoberta de Serviço, Implantação de Serviço e a própria decisão sobre realizar ou não uma operação de *offloading* sobre um determinado recurso.

3.1.1 DESCOBERTA DE SERVIÇO

O funcionamento dessa tarefa de Descoberta de Serviço ocorre em dois passos. O primeiro passo envolve tanto os dispositivos móveis, como o *cloudlet,* no qual ambos os dispositivos devem estar sobre uma mesma rede Wifi, pois o *cloudlet* utiliza da técnica de IP Multicast para anunciar a sua presença na rede local. O segundo passo envolve o procedimento em que o aplicativo "perguntará" sobre a existência de um determinado serviço de

[1] *Endpoint* em arquitetura orientada a serviço (SOA) é um ponto de entrada para um serviço, um processo, ou mesmo, uma fila.

offloading no *cloudlet*. Se for positiva essa resposta, será enviado para o aplicativo móvel endereço IP e porta, onde esse serviço está sendo executa no *cloudlet*. Caso contrário, a API procederá com o processo da Implantação do Serviço no *cloudlet*, que envolve enviar as dependências do aplicativo móvel, que são necessárias para o *cloudlet* executar o segmento de código no ambiente de execução remoto.

Entretanto, se a tarefa de Descoberta de Serviço na API não detectar nenhum *cloudlet* na rede local, então o aplicativo decidirá executar localmente o processamento marcado para *offloading*.

3.1.2 OPERAÇÃO DE OFFLOADING

Essa operação precisa da intervenção do desenvolvedor apenas para marcar os métodos no código fonte, que precisam suportar a operação de *offloading* no aplicativo móvel. No entanto, o procedimento de *offloading* será utilizando métodos de JSON-RPC, por ser considerado um método leve de chamada remota de procedimento, se comparando com XML-RPC.

O *framework* pode adotar os dois conceitos de particionamento do aplicativo. O primeiro conceito, denominado de particionamento dinâmico do aplicativo, que envolve decidir automaticamente, sobre a viabilidade de migrar, ou não, um método "anotado" para um ambiente de execução remoto. Essa decisão é realizada de acordo com as condições da rede Wifi (valor da latência entre o dispositivo móvel e o cloudlet). Já no segundo conceito referente ao particionamento estático, o desenvolvedor considera que é sempre necessário realizar uma operação de *offloading* sobre um método anotado.

No entanto, especificamente para o particionamento estático, se a API Cliente não encontrar um *cloudlet* na área, vai ser disparado um erro para caracterizar ausência da presença de um *cloudlet* na rede local. No caso do particionamento dinâmico, se esse mesmo evento acontecer, o *framework* decidirá executar localmente o processamento marcado para *offloading*.

3.2 MpOS Cloudlet

O *framework* MpOS, na parte do *cloudlet*, possui também diversas tarefas automatizadas, dentre estas tarefas estão a Implantação do Serviço, a Inicialização do Serviço e a Descoberta do Serviço.

A intervenção do usuário é apenas necessária em relação à gestão dos ambientes remotos de execução, através de uma interface Web. O processo de gestão não permite que o usuário implante manualmente uma dependência do aplicativo móvel no *cloudlet*, porque esse procedimento automatizado pelo *framework* MpOS.

A Descoberta de Serviço no *cloudlet* atua no processo de anuncio da sua presença na rede local, além de monitorar os serviços que estão em execução nos ambientes remotos, respondendo para os dispositivos móveis se um determinado serviço existe ou não no *cloudlet*. Caso exista, será devolvido como resposta, endereço IP e porta, relativo a esse serviço requisitado. Senão o aplicativo móvel inicializará o procedimento de Implantação do Serviço no *cloudlet*.

Após o cliente do MpOS API enviar as dependências do aplicativo móvel para o *cloudlet*, será realizado nesse *cloudlet* um procedimento de implantação e inicialização do serviços relativo com essas dependências no ambiente de execução remoto. O *framework* também delegará para esse novo serviço um *endpoint*, para ser acessado posteriormente pelos dispositivos móveis. Assim a tarefa relativa com a Descoberta de Serviço detectará o novo serviço em operação, tornando-o disponível na rede local para consulta e acesso, pelos outros dispositivos móveis.

4. CONTRIBUIÇÕES ESPERADAS

A implementação do *framework* MpOS pretende mostrar como tratar do problema da técnica do *offloading* nas diferentes plataformas móveis do mercado (Android e Windows Phone). O *framework* dará a possibilidade do desenvolvedor escolher qual melhor forma de fazer particionamento do seu aplicativo, podendo ser estático ou dinâmico.

A abordagem de marcação dos métodos para operação de *offloading* pode ser considerada uma abordagem intrusiva no ponto de vista do desenvolvimento do aplicativo móvel, mas não é intrusivo no momento da instalação do aplicativo no dispositivo móvel. Assim, com a utilização dessa abordagem objetiva-se facilitar a adoção do *framework* MpOS por parte dos desenvolvedores de aplicativo móveis.

O sistema de decisão do *framework* MpOS funciona baseado nas condições da rede, sendo bem mais simples do que aplicando um *profiling* sobre esse componente que se pretende realizar a operação de *offloading*.

5. METODOLOGIA ADOTADA

As pesquisas realizadas nesse trabalho de mestrado procuram alcançar os seguintes objetivos: Revisar a bibliografia relativa ao tema de *Mobile Cloud Computing*. Essa revisão deve ser voltada principalmente nas questões de como desenvolver um *framework* que utiliza a técnica de *offloading*, além desse *framework* está alinhado com o problema da heterogeneidade em diferentes plataformas móveis.

A metodologia de trabalho planeja realizar leitura de artigos; estudar, programar e testar a proposta do *framework* MpOS com todos os seus serviços, para ambas as plataformas móveis (Windows Phone e Android) e ambiente de execução no *cloudlet*.

Para realizar a prova de conceito do funcionamento do *framework* MpOS, será desenvolvido um aplicativo de multimídia, que interagirá com recursos remotos de um *notebook* que servirá de *cloudlet*. Será realizada uma avaliação para mensurar o consumo de energia sobre a execução local ou remota de um aplicativo que utiliza a solução do *framework* MpOS.

6. ESTÁGIO ATUAL DO TRABALHO

A revisão da literatura e a especificação da arquitetura do *framework* MpOS já foram concluídas. Alguns componentes relativos com a descoberta de serviço e a detecção da latência entre o dispositivo móvel e o *cloudlet* estão funcional na plataforma Android.

A plataforma móvel do Windows Phone foi escolhida como segundo sistema para desenvolver o *framework* MpOS, porque o Windows Phone utiliza uma plataforma de execução (WinRT[2]), que permite um mesmo código (no formato de *bytecode*) execute tanto em processadores da arquitetura x86 como ARM. A outra motivação para escolha do Windows Phone é a semelhança entre a linguagem C# e o Java (Android), diferente do que ocorre caso fosse escolhido como plataforma secundaria o iOS da Apple que utiliza a linguagem Objective-C.

7. TRABALHOS RELACIONADOS

Conforme citado, várias abordagens [2, 6, 7, 8, 9, 10] utilizam diferentes técnicas de *offloading*, sendo listados nessa seção apenas os trabalhos que utilizam as mesmas técnicas de *offloading* relacionadas com essa proposta de mestrado.

MAUI [2] é o *framework* que mais se aproxima dessa proposta de mestrado, porque também visa utilizar a técnica de *offloading* para economizar energia do dispositivo móvel.

No entanto, esse *framework* foi desenvolvido especificamente para a plataforma do Windows Phone, que suporta linguagem C# e plataforma .NET, não podendo ser utilizado por outras plataformas móveis, como Android e iOS. O *framework* MAUI também não menciona nenhuma medida para evitar que um componente malicioso de *offloading* monopolize todos os recursos da máquina servidora do *framework* MAUI.

Scavenger [9] é um *framework* que utiliza também chamadas RPC como método de *offloading*, sendo um método similar ao empregado pelo *framework* MAUI.

Esse trabalho tentou também tratar da questão da heterogeneidade de plataformas móveis, utilizando a linguagem Python, que "teoricamente" executaria em qualquer plataforma móvel do mercado. Como o *framework* foi desenvolvido baseado nessa linguagem, os aplicativos móveis precisam também ser desenvolvidos utilizando Python. Contudo, o problema dessa abordagem é que nenhuma das três maiores plataformas móveis do mercado (Android, iOS e Windows Phone, na ordem de procedência[3]), utiliza nativamente a linguagem Python para o desenvolvimento de aplicativos móveis. Outro problema dessa abordagem é que os próprios autores não conseguiram validar o *framework* utilizando duas plataformas móveis diferentes.

Os trabalhos listados carecem de um melhor suporte em relação algumas questões do gerenciamento dos recursos, como o controle da quantidade de recursos (memória, CPU ou disco) alocados para um determinado serviço visando evitar que esse serviço monopolize todo o recurso físico do *host*. Além do problema central dessa dissertação, relacionado com a aplicação da técnica de *offloading* nas diferentes plataformas móveis do mercado.

[2] http://www.infoq.com/news/2011/09/Design-Details-Windows-Runtime

[3] http://venturebeat.com/2013/05/16/windows-phone-jumps-to-third-in-global-smartphone-market-share-and-could-be-second-faster-than-you-think/

8. REFERÊNCIAS

[1] De Vriendt, R. et al. (2002). Mobile network evolution: a revolution on the move. IEEE Communications Magazine, Portland, vol. 40, no. 4, 104-111.

[2] Cuervo, E. et al. (2010). Maui: Making Smartphones Last Longer with Code Offload. Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys 2010), San Francisco, California, EUA.

[3] Dinh, H. T. et al. (2011). A survey of mobile cloud computing: architecture, applications and approaches. Wireless Communication and Mobile Computing. DOI: 10.1002/wcm.1203.

[4] Kovachev, D., Klamma, R. (2012). Beyond the Client-Server Architectures: A Survey of Mobile Cloud Techniques. IEEE Workshop on Mobile Cloud Computing (MobiCC'12), Beijing , China.

[5] Shiraz, M. et al. (2012). A Review on Distributed Application Processing Frameworks in Smart Mobile Devices for Mobile Cloud Computing, *Communications Surveys & Tutorials, IEEE* , vol.PP, no.99, 1-20.

[6] Verbelen, T. et al. (2012). AIOLOS: Middleware for improving mobile application performance through cyber foraging, Journal of Systems and Software, vol. 85, no. 11, 2629-2639.

[7] Chun B-G. et al. (2011). CloneCloud: elastic execution between mobile device and cloud. In *Proceedings of the sixth conference on Computer systems* (EuroSys '11), 301-314.

[8] Marinelli, E. E. (2009). Hyrax: Cloud Computing on Mobile Devices using MapReduce. MS thesis, Carnegie Mellon University, Pittsburgh, USA.

[9] Kristensen, M. D., Bouvin, N. O. (2010). Scheduling and development support in the Scavenger cyber foraging system, Pervasive and Mobile Computing, vol. 6, no. 6, 677-692.

[10] Kemp, R. et al. (2012) Cuckoo: A Computation Offloading Framework for Smartphones, Second International ICST Conference, MobiCASE 2010. DOI: 10.1007/978-3-642-29336-8_4.

[11] Satyanarayanan, M. et al. (2009). The Case for VM-Based Cloudlets in Mobile Computing. *IEEE Pervasive Computing*, vol. 8, no. 4, 14-23.

[12] Sanaei, Z. et al. (2013). Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges, *Communications Surveys & Tutorials, IEEE* , vol. PP, no. 99, 1-24.

[13] Bahl, P. et al. (2012). Advancing the state of mobile cloud computing. In *Proceedings of the third ACM workshop on Mobile cloud computing and services* (MCS '12), 21-28.