Análise de Performance de Frameworks de Desenvolvimento Mobile Multiplataforma

Kamile A. Wahlbrinck, Bruno B. Boniati

Universidade Federal de Santa Maria – (UFSM)
Caixa Postal 54 – 98.400-000 – Frederico Westphalen – RS – Brasil

{kamilewahlbrinck,brunoboniati}@gmail.com

Resumo. *Desenvolver aplicações mobile não é tarefa fácil devido as várias plataformas móveis existentes. Para facilitar o desenvolvimento dessas aplicações existem frameworks de desenvolvimento multiplataforma, que permitem que se escreva um único código que funcione em mais de uma plataforma móvel, o que torna o desenvolvimento mais rápido e fácil. Este trabalho tem como objetivo analisar a performance desses frameworks comparada à de aplicativos nativos. Serão abordados diferentes frameworks (que utilizam diferentes linguagens de programação) e partindo-se de uma aplicação nativa com uso intenso de CPU pretende-se mapear as diferenças no tempo de execução de tal aplicativo utilizando-se o mesmo hardware e diferentes frameworks.*

Abstract. *Developing mobile applications is not a easy task because of the several existing mobile platforms. In order to facilitate the development of these applications there are cross-platform development frameworks which allow the user to enter a unique code that works on more than one mobile platform making it a faster and easy development. This paper aims to analyze the performance of these frameworks compared to native applications. It will be addressed different frameworks (ones that use different programming languages) and starting from a native application with intensive CPU usage it is intended to track differences in performance time as such application using the same hardware and different frameworks.*

1. Introdução

Com a crescente evolução das tecnologias, os dispositivos inteligentes estão cada vez mais presentes no dia a dia das pessoas. A utilização de dispositivos móveis é bastante variada, profissionalmente pode ser utilizado para ler e-mails, realizar transações bancárias e utilizar ferramentas de comunicação providas pela internet [PREZOTTO 2014]. O fato de a tecnologia estar cada vez mais acessível a diferentes perfis de consumidores torna esses dispositivos ferramentas de trabalho, lazer e aprendizagem e um grande mercado consumidor para quem tiver interessado em oferecer serviço e *softwares*.

O advento do conceito de sistemas operacionais ou plataformas para dispositivos móveis é um dos fatores que possibilitou o crescimento na oferta de aplicativos para tais plataformas [LECHETA 2013]. Observa-se, no entanto, que não há um padrão de mercado em relação à plataforma de desenvolvimento a ser adotada e dependendo da escolha do profissional de Tecnologia da Informação (TI) o *software* desenvolvido será incompatível para ser executado em uma plataforma diferente daquela para o qual foi projetado [CARVALHO 2014].

Diante de tal realidade programadores procuram uma forma fácil e rápida para atender as necessidades do maior número possível de plataformas, pois desenvolver um aplicativo para cada dispositivo é uma tarefa economicamente desgastante. A solução é recorrer a *frameworks* de desenvolvimento multiplataforma, que a partir de um único código tornam possível que sua aplicação funcione em mais de uma plataforma.

A intenção deste trabalho é estudar diferentes *frameworks* para desenvolvimento *mobile* e realizar um teste de performance com os mesmos. Entende-se que o fato de os *frameworks* utilizarem diferentes linguagens e tecnologias é natural que se comportem de forma diferenciada. Para tanto a próxima seção conceitualiza os *frameworks* multiplataforma e descreve alguns que estão sendo utilizados para os testes. A seção 3 apresenta os resultados parciais obtidos com os testes e na seção 4 são feitas as considerações finais.

2. Frameworks de Desenvolvimento Multiplataforma

Segundo Fayad e Schmidt (1997), um *Framework* é uma aplicação semi-completa, reusável que pode ser especializada para produzir aplicações customizadas. No contexto do desenvolvimento de aplicações multiplataforma, os *frameworks* podem ser entendidos como blocos de códigos pré-implementados que serão reutilizados no momento da compilação do código escrito pelo desenvolvedor para gerar a aplicação final [FAYAD e SCHMIDT 1997].

Resumidamente, *frameworks* de desenvolvimento multiplataforma nos permitem escrever um único código que poderá ser executado em mais de uma plataforma móvel. Durante o desenvolvimento desse trabalho são utilizadas as seguintes ferramentas: Phonegap, AppGyver e Corona SDK. Para fins de comparação, uma aplicação nativa, que executa o mesmo algoritmo também será implementada, neste caso será utilizada a linguagem de programação Java com SDK para Android.

2.1. PhoneGap

Open-source e gratuito, o PhoneGap provê um container *web* no qual você constrói suas aplicações multiplataforma usando apenas o básico da *web*: HTML5 + JavaScript e CSS. Ele provê APIs JavaScript para que o desenvolvedor tenha acesso aos recursos de *hardware* do dispositivo móvel e exige apenas que os dispositivos tenham *browsers* que suportem esses recursos básicos e padrões da *web* [PHONEGAP 2014].

2.2. AppGyver

AppGyver utiliza linguagens *web* (HTML5 e JavaScript) para desenvolvimento de aplicativos móveis. É conhecido principalmente por suas ferramentas de prototipagem e por usar Steroids.js, uma ferramenta de linha de comando que permite criar rapidamente aplicativos HTML5. Um recurso muito interessante do serviço é que pode-se gerar um código QRCode e digitalizá-lo no aplicativo móvel do AppGyver para acompanhar as alterações feitas no desenvolvimento em seu telefone em tempo real [APPGYVER 2014].

2.3. Corona SDK

Baseia-se na linguagem Lua e em *frameworks* consolidados de desenvolvimento como OpenGL ES, OpenAL, Box2D, entre outros. Muito bom para desenvolvimento de jogos

2D, deixa a desejar um pouco quando a assunto são apps com *look-and-feel* nativo dos *frameworks* tradicionais. Possui uma versão *trial ilimitad* mas um preço de licenciamento alto quando se deseja publicar a aplicação nas lojas de aplicativos [CORONA 2014].

2.4. Algoritmo Implementado

Para realizar os testes com os diferentes *frameworks* multiplataforma foi utilizado um algoritmo que calcula a soma dos números primos no intervalo de 1 a 50000. O mesmo algoritmo foi escrito em JavaScript para as ferramentas PhoneGap e AppGyver, em Linguagem Lua, para o Corona SDK e em JAVA para o aplicativo nativo. A figura 1 mostra o algoritmo utilizado, escrito em pseudo-código.

```
Variaveis
    soma <- 0, cont, intervalo, i, j : inteiro
Inicio
    Para i de 1 ate 50000 faca
    cont <- 0;
        Para i de 1 ate i faca
            Se (i%j == 0) entao cont <- cont + 1;
        FimPara
        Se (cont <= 2) entao soma <- soma + 1;
    FimPara
Fim
```

Figura 1. Algoritmo Implementado

3. Resultados Parciais

Com base na pesquisa feita sobre cada *framework* construiu-se uma tabela comparativa onde são listados os *frameworks* estudados (PhoneGap, AppGyver, Corona SDK) apresentando algumas de suas características. O resultado pode ser visualizado na tab. 1.

Tabela 1. Tabela Comparativa – Frameworks de Desenvolvimento Mobile

	PhoneGap	AppGyver	Corona
Plataformas Suportadas	Android, iOS, Windows Phone, Black Barry, Symbiam, Bada, Kindle Fire e Nook Color	Android, iOS	Android, iOS
Linguagens	HTML, JavaScript, CSS	JavaScript, HTML5	Lua
Acesso a recursos de Hardware	Sim	Sim	Sim

Conforme apresentado na tabela 1, percebe-se que o PhoneGap é o *framework* que abrange maior número de plataformas e assim como o AppGyver utiliza umas das linguagens mais conhecidas entre os programadores. É importante ressaltar que o PhoneGap e o AppGyver, executam os aplicativos usando *WebViews*, que se utilizam do *browser* para rodar o aplicativo, tanto *online* quanto *off-line*. Já o *framework* Corona utiliza interpretadores ou compiladores incorporados nas aplicações ou nos sistemas operacionais para executar o mesmo código fonte em mais de uma plataforma.

Os testes para análise de performance das ferramentas foram realizados em um mesmo *hardware*, um celular da Samsung, modelo Galaxy SIII mini aparelho GT-18190L com processador Dual Core 1GHz e 1GB de memória RAM, com sistema

operacional Android 4.1 Jelly Bean. A partir dos testes realizados, chegou-se a um tempo médio de execução para cada ferramenta, divisão da soma do tempo de cada execução dos dez testes realizados com o aplicativo desenvolvido em cada *framework*, apresentados na tabela 2.

Tabela 2. Tabela Comparativa – Tempo de execução dos algortimos

	PhoneGap	AppGyver	Corona	Android
Máximo	104,96s	112s	989,7s	72s
Mínimo	89,21s	90,15s	600s	62s
Médio	92,39s	95,77s	900,5s	63,7s
Percentual	47%	50,3%	1313%	-

Com base nas informações da tabela 2 pode-se observar que o aplicativo com melhor performance foi o nativo, seguido dos aplicativos desenvolvidos com linguagens *web* e por último, com pior desempenho, o aplicativo desenvolvido em Lua. Comparados ao tempo de execução do aplicativo nativo, os aplicativos desenvolvidos usando *frameworks* em HTML foram, em média, 48,65% mais lentos que o nativo, enquanto o *app* desenvolvido com linguagem Lua foi 1313% mais lento que o nativo.

4. Conclusão

O trabalho de pesquisa realizado, permitiu-nos identificar a existência de algumas iniciativas no sentido de oferecer ao profissional de TI um conjunto de ferramentas e bibliotecas para desenvolver uma única versão do *software* e portá-la para diferentes plataformas. Observa-se, no entanto, que tais ferramentas apresentam significativas diferenças que vão desde a linguagem utilizada para codificar até a forma de disponibilizar o software desenvolvido para instalação. Os recursos oferecidos pelos frameworks estudados também são distintos e não há uma homogeneidade em relação às plataformas suportadas.

As próximas etapas do projeto visam desenvolver um novo algoritmo para testar a performance dos *frameworks* quanto a memória e testar como funciona o acesso aos recursos nativos do *hardware*.

Referências

AppGyver. (2014) “Create beautiful mobile apps with”, http://www.appgyver.com/, acesso em setembro/2014.

Carvalho, M. S. (2014) “Titanium Mobile: Aplicações multiplataforma”, http://www.devmedia.com.br/titanium-mobile-aplicacoes-multiplataforma-revista-mobile-magazine-41/24055, acesso em setembro/2014.

Corona, S. (2014) “Develop cross platform mobile apps em games”, http://coronalabs.com/products/corona-sdk/, acesso em setembro/2014.

Fayad, E. M. e Schmidt, D. C. (1997) “Object-oriented application frameworks”, In: Communications of the ACM, Editora Guest, p. 32-38.

Lecheta, R. R. (2013), Aprenda a criar aplicações para dispositivos móveis com Android SDK, 3ª edição.

PhoneGap, (2014) “Easily create apps using the web technologies you know and love: Html, css and javascript”, phonegap.com/, acesso em setembro/2014.

Prezotto, E. D.. (2014), Estudo de frameworks multiplataforma para desenvolvimento de aplicações mobile híbridas. Universidade Federal de Santa Maria, Trabalho de Conclusão de Curso.