SUSTENTATIBILIDADE DE SOFTWARE: UM ESTUDO DE CASO DA AVALIAÇÃO DE APLICATIVOS ANDROID

ALINE RODRIGUES TONINI[1]; LEONARDO MATTHIS FISCHER[2]; LISANE BRISOLARA DE BRISOLARA[2]

[1]Grupo e Arquiteturas e Circuitos Integrados-UFPel – artonini@inf.ufpel.edu.br
[2]Programa de Pós-Graduação em Computação -UFPel – lmfischer@inf.ufpel.edu.br
[2]Programa de Pós-Graduação em Computação -UFPel – lisane@inf.ufpel.edu.br

1. INTRODUÇÃO

O uso de dispositivos móveis, tais como *smartphones* e *tablets*, tornou-se popular nos últimos anos, consequentemente a procura por aplicativos cada vez mais sofisticados e para diferentes fins também cresceu. Com isso surgiu a necessidade da criação de plataformas mais completas e padronizadas, que permitissem aos desenvolvedores criar aplicativos independentes da arquitetura.

Dentre as plataformas móveis, a Android (GOOGLE, 2014) se destaca por ser uma solução *open source* e estar presente em uma grande variedade de dispositivos do mercado. Para os desenvolvedores de aplicativos, esta plataforma se mostra atrativa, pois usa Java como linguagem de programação, possui sua própria IDE de desenvolvimento o Android Studio, da mesma forma que recursos específicos para desenvolvimento Android já estão disponíveis na IDE Eclipse.

Contudo, o uso de dispositivos móveis, assim como outros tipos de tecnologia que necessitam de energia, traz a tona o problema do uso eficiente dos recursos energéticos disponíveis no planeta ou na bateria. Atualmente, a preocupação com o uso sustentável dos recursos do planeta tem sido foco de discussões. Com isso, temas como Engenharia de Software Sustentável e sustentabilidade de software tem se tornado emergentes (JOHANN et al., 2011).

Observando essas novas tendências relacionadas a software sustentável, otimizações tem sido propostas para melhorar a sustentabilidade dos produtos. Em Siebra et. al. (2013) o uso de algoritmos e estruturas de dados mais eficientes é recomendado para prover melhor desempenho em software e hardware. A Intel lançou diretrizes focadas em otimização para melhorar a eficiência energética de aplicações (INTEL, 2010).

No contexto da plataforma Android, o Google também propôs boas práticas de programação (GOOGLE, 2014), as quais quando aplicadas melhoram o desempenho geral da aplicação.

Entre as práticas sugeridas pelo Google, duas são consideradas nesse trabalho: a escolha da implementação de laços do tipo *for* de acordo com a estrutura de dados usada e a prática de não utilização de métodos de acesso. Através de experimentos, este trabalho avalia estas boas práticas, analisando seu impacto no desempenho e no consumo energético. Desta forma, o presente trabalho pode ser visto como um estudo de caso de avaliação de sustentabilidade de aplicativos móveis desenvolvidos para a plataforma Android.

O trabalho está organizado da seguinte maneira. A Seção 2 apresenta a metodologia adotada, enquanto a Seção 3 discute os principais resultados obtidos. Na Seção 4, as principais conclusões são apresentadas e trabalhos futuros são delineados.

2. METODOLOGIA

Para realizar o estudo de caso, foi desenvolvido um aplicativo de Agenda, onde são inseridos quatrocentos contatos e após, esses contatos são pesquisados na Agenda. Nos experimentos foram utilizados dois dispositivos móveis: um *tablet* Asus Transformer com Android 4.0.3 e um tablet Samsung Galaxy Note 8.0, modelo GT-N5110. Nestes experimentos três estruturas de dados (EDs) foram analisadas: o *ArrayList* (ORACLE, 2014a), *LinkedList* (ORACLE, 2014b) e o *Vecto*r (ORACLE, 2014c). O aplicativo foi executado trinta vezes em cada dispositivo com o código original, após foram aplicadas as boas práticas e realizadas mais trinta execuções. Este mesmo experimento foi repetido para cada tipo de estrutura de dados. Por fim, foi calculada a média das trinta execuções de cada experimento e estas são comparadas usando o teste estatístico "*t de student*".

Para realizar o rastreamento e análise das execuções são empregados métodos da biblioteca *android.os.debug* e o uso do DDMS (*Dalvik Debug Monitor System*) (GOOGLE, 2014). Os resultados do rastreamento são visualizados na ferramenta Traceview (GOOGLE, 2014). Dentre estes dados estão o *Incl CPU Time*, *Excl CPU Time*, número de chamadas de métodos, número de chamadas de métodos recursivos e para cada método a visualização de seus métodos filhos. Dentre essas informações, o *Incl CPU Time* será adotado em nossas análises. Esta métrica indica o tempo de CPU ou tempo de execução, e é representado em *ms*. O *Incl CPU* Time representa o tempo de execução do método principal da aplicação, incluindo o tempo de execução dos métodos filhos, diferente do *Excl CPU Time* que não considera a chamada a métodos filhos. A análise de consumo energético é realizada com o aplicativo POWER TUTOR (ZHANG et al, 2010), que faz uma análise de consumo de energia dos aplicativos que estão sendo executados no dispositivo e provê os resultados em Joule.

3. RESULTADOS E DISCUSSÃO

Esta seção apresenta e discute os resultados experimentais de avaliação de duas das boas práticas do Google. As Figuras 1 e 2 ilustram os resultados para desempenho e consumo energético, respectivamente, obtidos para os experimentos com a prática do *For* e dos métodos de acesso.

Para os resultados de desempenho, bem como para os de consumo energético, observa-se que as EDs ArrayList e LinkedList mostram-se mais eficientes do que a estrutura de dados Vector, quando as configurações com mesmo tipo de laço são comparadas. Considerando o tipo de laço utilizado, a implementação do *for* com tamanho ("escrito à mão") possui melhores resultados quando comparada ao *For-Each* e ao *For* sem Tamanho em todas estruturas de dados e dispositivos avaliados. Isso por ser uma implementação onde o tamanho do arranjo já é obtido antes da busca, evitando fazer isso a cada iteração. Quando o *For* sem tamanho é implementado, o método *size()* é chamado a cada iteração para buscar o tamanho do arranjo, o que é custoso. Apesar de possuir uma melhor visualização, a implementação do *For-Each* torna-se mais custosa do que os outros tipos de laços avaliados devido as chamadas internas aos métodos *hasNext()* e *next()*.

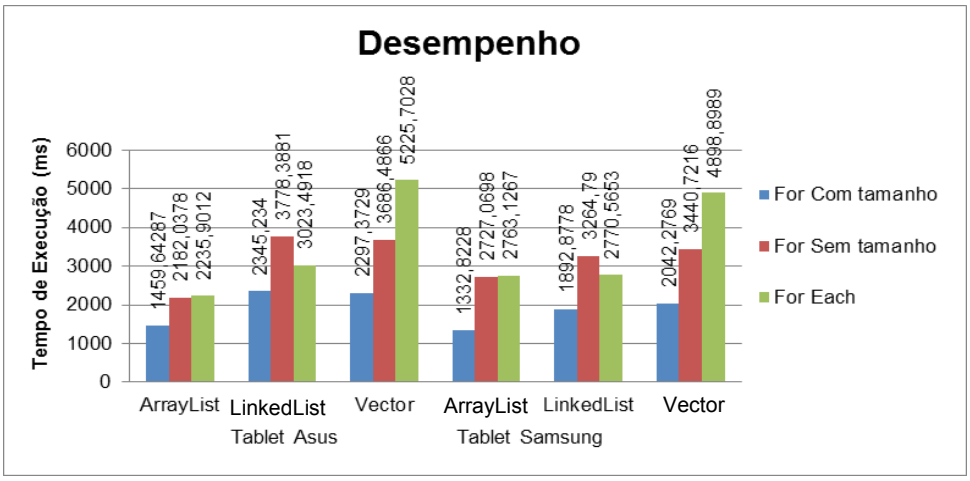

Figura 1. Resultados para desempenho da prática para a prática de loops.

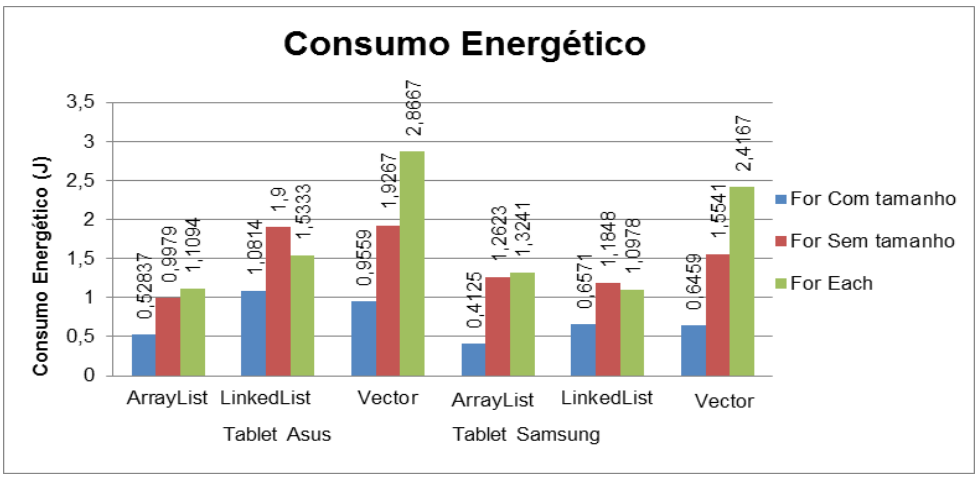

Figura 2. Resultados para consumo energético da prática para *loops*.

Os resultados da análise do não uso de métodos de acesso (*getters* e *setters*) são sumarizados na Tabela I, onde observa-se reduções, tanto no tempo de execução quanto no consumo, obtidas pelo acesso de forma direta aos dados. Neste conjunto de experimentos foi adotado o laço *For-each*.

Tabela I. Resultados para a prática dos métodos de acesso.

Resultados									
Dispositivo	**Estrutura de Dados**	**Com Getters/Setters**				**SemGetters/Setters**			
		Desempenho		**Consumo**		**Desempenho**		**Consumo**	
		Média(ms)	**σ(ms)**	**Média(J)**	**σ(J)**	**Média(ms)**	**σ(ms)**	**Média(J)**	**σ(J)**
Tablet Asus	**ArrayList**	2235.9012	14.7060	1.1094	0.1811	1521.4716	9.8519	0.52696	0.1533
	LinkedList	3023.4918	22.0223	1.5333	0.2324	2294.8181	21.2629	0.9962	0.0799
	Vector	5225.7028	33.2343	2.8667	0.1988	4488.9026	37.69909	2.44	0.1940
Tablet Samsung	**ArrayList**	2763.1267	21.3485	1.3241	0.1995	2067.0387	17.9553	0.7448	0.1718
	LinkedList	2770.5653	44.2433	1.0977	0.1286	2082.7273	16.0353	0.6124	0.1457
	Vector	4898.8989	59.0668	2.41666	0.3534	4172.5485	79.9402	2.0733	0.2703

4. CONCLUSÕES

Este trabalho apresentou uma avaliação de duas boas práticas para Android apresentadas pela Google. As práticas foram avaliadas comparando a eficiência

do código original com a do código obtido com a boa prática. Os resultados foram obtidos através de execuções das diferentes versões do aplicativo criado, realizando um rastreamento para obter o desempenho e o consumo energético e considerando dois dispositivos físicos diferentes.

De acordo com os resultados, o uso da ED e do tipo de laço adequado, assim como evitar métodos de acesso, podem prover uma melhora no desempenho e na eficiência energética. Este estudo demonstra que otimizações podem auxiliar na criação de aplicativos mais sustentáveis e que utilizam de forma eficiente os recursos do meio-ambiente.

5. REFERÊNCIAS BIBLIOGRÁFICAS

GOOGLE. **Plataforma Android**. Acessado em 18 jul. 2014. Online. Disponível em: http://www.android.com/.

JOHANN, T.; DICK, M.; KERN, E.; NAUMANN, S. Sustainable Development, Sustainable Software, and Sustainable Software Engineering An Integrated Approach. **International Symposium on Humanities, Science and Engineering Research**, p. 34 - 39, 2011.

SIEBRA, C., COSTA, P., SILVA, F. Q. B., SANTOS, A. M. L. e MASCARO, A. The Hardware and Software Aspects of Energy Consumption in the Mobile Development Platform. **Internation Journal of Pervasive Computing and Communications**. Vol 9 155:2, p. 139 – 152, 2013.

INTEL. **Energy-Efficient Software Guidelines**. Abr. 2010. Acessado em 18 jul. 2014. Online. Disponível em: https://software.intel.com/en-us/articles/energy-efficient-software-guidelines.

GOOGLE. **Best Practices for Performance**. Acessado em 18 jul. 2014. Online. Disponível em: http://developer.android.com/training/articles/perf-tips.html.

GOOGLE. **DDMS**. Acessado em 18 jul. 2014. Online. Disponível em: http://developer.android.com/tools/debugging/ddms.html.

GOOGLE. **Traceview**. Acessado em 18 jul. 2014. Online. Disponível em: http://developer.android.com/tools/debugging/debugging-tracing.html.

ZHANG, L.; TIWANA, B.; QIAN, Z.; WANG, Z.; DICK, R. P.; MAO, M.; YANG, L. Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones. **International Conference on Hardware/Software Codesign and System Synthesis.** p. 105 – 114, 2010**.**

ORACLE. **ArrayList.** Acessado em 18 jul. 2014. Online. Disponível em: http://docs.oracle.com/javase/7/docs/api/java/util/ArrayList.html.

ORACLE. **LinkedList.** Acessado em 18 jul. 2014. Online. Disponível em: http://docs.oracle.com/javase/7/docs/api/java/util/LinkedList.html.

ORACLE. **Vector.** Acessado em 18 jul. 2014. Online. Disponível em: http://docs.oracle.com/javase/7/docs/api/java/util/Vector.html.