

Análise de Desempenho em Dispositivos Limitados e Emulados Estudo de Caso: Raspberry Pi e Web Services RESTful

Luiz H. Nunes¹, Luis H. V. Nakamura¹, Rafael M. Libardi¹,
Júlio C. Estrella¹, Marcos J. Santana¹, Regina H. C. Santana¹

¹ Universidade de São Paulo (USP)
Instituto de Ciências Matemática e de Computação (ICMC)
Departamento de Sistemas de Computação (SSC)
São Carlos – SP – Brasil

{lhnunes, nakamura, mira, jcezar, mjs, rcs}@icmc.usp.br

Abstract. *This paper analyzes the impact of RESTful web services consumption and provision on devices with limited resources in real and emulated environments. To perform the experiments, a service for ordering numbers was created using the CXF framework and deployed on a Raspberry Pi. A performance evaluation was conducted to analyze the Raspberry Pi architecture behaviors in both real and emulated devices.*

Resumo. *Este trabalho analisa o impacto do consumo e provisão de web services RESTful em dispositivos com recursos limitados em ambientes reais e emulados. Para realizar os experimentos, um serviço para ordenação de números foi criado utilizando o framework CXF e implantado em um Raspberry Pi. Uma avaliação de desempenho foi conduzida para analisar o comportamento da arquitetura do Raspberry Pi tanto em dispositivos reais como em emulados.*

1. Introdução

O Raspberry Pi [Raspberry 2014] é um computador de baixo custo composto por um conjunto de microchips semelhante aos encontrados nos telefones móveis e possui o tamanho de um cartão de crédito. Embora existam muitos projetos para o Raspberry Pi, a limitação de seus recursos ainda é um problema, principalmente em relação a capacidade de processamento e memória. Tal limitação torna esses dispositivos inviáveis para a execução de aplicações que consumam uma grande quantidade de dados ou exijam alta capacidade computacional [AlShahwan and Moessner 2010]. Uma abordagem utilizada para contornar estas limitações e tentar otimizar o desempenho em dispositivos embarcados, inclusive no Raspberry Pi, é o emprego de aplicações orientadas a serviço.

Alguns trabalhos disponíveis na literatura tais como [Mizouni et al. 2011], [Aihkisalo and Paaso 2012] e [Hameseder et al. 2011] analisam o desempenho da provisão e utilização de serviços REST e SOAP em dispositivos móveis com recursos limitados e concluem que serviços RESTful são mais adequados para esses cenários. Porém, nenhum desses trabalhos realizam testes emulados destes dispositivos e verificam o impacto causado por ela ao prover e consumir estes serviços.

A contribuição deste trabalho está centrada na análise comparativa de desempenho entre dispositivos reais e emulados cujos os recursos são limitados. Com um enfoque mais

específico no Raspberry Pi, os resultados deste trabalho permitem que desenvolvedores de *web services* RESTful verifiquem os impactos dos resultados, em termos de desempenho, que são inerentes ao uso ou não de emuladores. Antes da aquisição de novos equipamentos torna-se, por muitas vezes, essencial a realização de um dimensionamento e uma avaliação de desempenho de equipamentos com recursos limitados, a fim de averiguar se tais recursos são realmente suficientes para determinadas tarefas. Dessa forma, justifica-se a necessidade de uma análise que determine a diferença de desempenho entre dispositivos reais e emulados em um campo recentemente explorado como é o caso de aplicações distribuídas em dispositivos semelhantes ao Raspberry Pi.

Neste trabalho, é apresentada uma avaliação de desempenho voltada para análise do comportamento de *web services* RESTful no Raspberry Pi por meio do *framework* CXF [Apache 2013b]. Para esta avaliação, os experimentos analisam os tempos de empacotamento e desempacotamento das mensagens e o tempo total de processamento do serviço com diferentes tamanhos. Além disso, os testes foram realizados tanto em dispositivos reais quanto em dispositivos emulados por meio do emulador QEMU [Bellard 2014].

2. Frameworks e Tecnologias

O modelo REST fornece um conjunto de padrões, os quais são aplicados por meio de protocolos livres de estado para comunicação, tipicamente HTTP. Os clientes e os provedores de serviço trocam representações dos recursos por meio de uma interface padrão. Estes recursos são identificados por uma URI e manipulados por meio dos métodos HTTP.

O Apache CXF é um *framework* de código fonte aberto que provê o fácil desenvolvimento de *web services* em Java, tanto para o protocolo SOAP quanto para os conceitos arquiteturais RESTful [Balani and Hathi 2009]. Isso é feito por meio de um modelo de programação baseada em padrões. O CXF JAX-RS determina a semântica para a criação de *web services* RESTful e permite a troca de informações em diferentes formatos de dados como, por exemplo, JSON e XML [Apache 2013a]. O fluxo das mensagens no Apache CXF é dividido em fases. Cada fase possui um conjunto de interceptores são responsáveis por gerenciar os recursos e informações de outros componentes do CXF mas também preparar a mensagem e realizar a comunicação entre cliente e serviço [Balani and Hathi 2009].

3. Avaliação de Desempenho

No cenário definido para a avaliação de desempenho o dispositivo cliente inicia a aplicação gerando uma sequência de números aleatórios e, em seguida, realiza a serialização dessa sequência em uma mensagem XML. Por fim, essa mensagem é enviada para processamento no dispositivo servidor. Ao receber a mensagem, o servidor primeiramente faz a deserialização. Em seguida é feito o processamento que ordena os números recebidos do cliente. Assim, uma nova mensagem com os números ordenados é gerada, serializada no formato XML e enviada ao dispositivo cliente. O cliente deserializa a resposta do servidor, obtém a mensagem ordenada e finaliza a sua execução.

3.1. Configuração do Ambiente e Configuração dos Experimentos

Na configuração do ambiente para os experimentos nos dispositivos reais foram utilizados dois microcontroladores Raspberry Pi modelo B com um cartão SD de 16GB [?] junta-

mente com o sistema operacional Raspbian Linux [Raspbian 2013]. Para os experimentos executados em ambientes emulados, o Qemu foi usado para emular um: processador ARMv6 (697,95 BogoMIPS), 256 MB de RAM, 16 GB de disco e Raspbian Linux. Devido à divergência na quantidade de memória RAM entre os dispositivos reais e emulados definiu-se que a quantidade máxima de memória utilizada na JVM (*Java Virtual Machine*) fosse de 128MB.

A modelagem do experimento é composta por quatro experimentos que envolvem a variação dos seguintes fatores (e níveis): dispositivo (real, emulado) e tamanho da mensagem (100Kb e 500Kb). Os *web services* desenvolvidos com o *framework* CXF foram instrumentados para coletar os tempos de serialização, deserialização e total das aplicações clientes e provedoras de serviço. Os experimentos foram replicados 50 vezes, a fim de se estabelecer um intervalo de confiança (IC) com valor de 95% de confiança.

4. Análise dos Resultados

A Tabela 1 contém os valores dos tempos totais médios (T.T) das amostras capturadas juntamente com o intervalo de confiança (I.C) de 95%. A partir dos valores desta tabela é notório que o *framework* CXF é capaz de deserializar mensagens de forma mais rápida do que serializa-las, isso se dá pois o processo de leitura de arquivos é menos custoso que o processo de escrita em arquivos. Nota-se também que a sobrecarga nos tempos de serialização, deserialização e tempo total é diretamente proporcional ao tamanho das mensagens utilizadas.

Tabela 1. Resultados obtidos juntamente com intervalo de confiança de 95%

Cliente								
Dispositivo Tamanho da Mens.	Real				Emulado			
	100Kb		500Kb		100Kb		500Kb	
	T.T	C.I	T.T	C.I	T.T	C.I	T.T	C.I
Deserialização (ms)	1842	44	2506	15	1879	23	3360	67
Serialização (ms)	4514	12	6630	125	4655	134	7900	168
Total (s)	35,2	0,4	125,7	0,7	42,16	0,96	174,6	1,8
Servidor								
Dispositivo Tamanho da Mens.	Real				Emulado			
	100Kb		500Kb		100Kb		500Kb	
	T.T	C.I	T.T	C.I	T.T	C.I	T.T	C.I
Deserialização (ms)	205,7	1	1023,1	77	305,2	6,6	1362,2	36,4
Serialização (ms)	361,6	62,4	3319,8	28,8	426,8	20,8	4769	117,3
Total (s)	16,8	0,1	82,14	0,74	22,6	0,6	118,2	1,8

Ainda analisando os resultados é possível observar que o comportamento entre os dispositivos reais e emulados é o mesmo tanto para clientes quanto para o servidor, apresentando apenas tempos maiores na emulação, que é justificado pela sobrecarga existente ao utilizar emulação de dispositivos. Nos dispositivos clientes esta sobrecarga é de até 25% para a deserialização, 16% para a serialização e 28% do tempo total enquanto que nos dispositivos servidores esta sobrecarga é de até 32,6% na serialização, 30,3% na

deserialização e 30,5% do tempo total. Por fim, é válido destacar que tanto o tempo de serialização quanto o de deserialização são responsáveis apenas por uma pequena parte do tempo de serviço, o qual é quase totalmente influenciado pelo tempo de processamento do microcontrolador.

5. Conclusão

Atualmente, a utilização de dispositivos de recursos limitados, como é o caso do Raspberry Pi, para as mais diversas finalidades é crescente. O advento do IPv6 também permite que novas aplicações distribuídas para tais dispositivos se notassem mais comuns. Dentro deste escopo, os Web Services RESTful são exemplos práticos de aplicações distribuídas que podem ser utilizadas, porém surge a questão se tais dispositivos são capazes de suportar todos os tipos de aplicações e quais seriam os seus limites de desempenho. Além disso, grande parte dos equipamentos incluídos neste nicho são importados e no caso de uma compra em larga escala se faz preciso analisar o desempenho de tais equipamentos antes de sua aquisição. Uma solução adotada proposta para esse problema é a utilização da técnica de emulação. Emulando tais dispositivos o desenvolvedor pode validar a sua utilização, verificando suas capacidades e limitações.

Contudo, uma sobrecarga gerada pela emulação não permite ao desenvolvedor identificar a real capacidade dos dispositivos reais. Neste ponto, este artigo buscou identificar qual seria essa sobrecarga em *web services* RESTful utilizando o Raspberry Pi. Os resultados indicam uma sobrecarga considerável que impacta no desempenho do tempo de processamento entre 16% à 32,6% dependendo da atividade sendo realizada. Notou-se também, como já era esperado, que a grande maioria dos intervalos de confiança são menores para os dispositivos reais, o que indica resultados com menor variabilidade.

Em trabalhos futuros pretende-se realizar novos experimentos variando a carga de trabalho, utilizando outros tipos de aplicações distribuídas e bem como outros *frameworks* de aplicações. O monitoramento e a análise do consumo de energia do dispositivo real também estão planejados para trabalhos futuros.

Referências

- Aihkisalo, T. and Paaso, T. (2012). Latencies of service invocation and processing of the rest and soap web service interfaces. In *Services (SERVICES), 2012 IEEE Eighth World Congress on*, pages 100–107.
- AlShahwan, F. and Moessner, K. (2010). Providing soap web services and restful web services from mobile hosts. In *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*, pages 174–179.
- Apache, S. F. (2013a). Apache cxf - jax-rs (jsr-311). Disponível em: <http://cxf.apache.org/docs/jax-rs.html>. Último acesso: 06/06/2013.
- Apache, S. F. (2013b). Apache cxf software architecture guide. Disponível em: <http://cxf.apache.org/docs/cxf-architecture.html>. Último acesso: 06/06/2013.
- Balani, N. and Hathi, R. (2009). *Apache CXF Web Service Development*. Packt Publishing.

- Bellard, F. (2014). Qemu - open source processor emulator. Disponível em: http://wiki.qemu.org/Main_Page. Último acesso: 31/05/2014.
- Hameseder, K., Fowler, S., and Peterson, A. (2011). Performance analysis of ubiquitous web systems for smartphones. In *Performance Evaluation of Computer Telecommunication Systems (SPECTS), 2011 International Symposium on*, pages 84–89.
- Mizouni, R., Serhani, M., Dssouli, R., Benharref, A., and Taleb, I. (2011). Performance evaluation of mobile web services. In *Web Services (ECOWS), 2011 Ninth IEEE European Conference on*, pages 184–191.
- Raspberry, P. F. (2014). What is a raspberry pi? Disponível em: <http://www.raspberrypi.org/help/what-is-a-raspberry-pi/>. Último acesso: 31/05/2014.
- Raspbian (2013). Welcome to raspbian. Disponível em: <http://www.raspbian.org/>. Último acesso: 06/06/2013.