Integração de Dispositivos Móveis em Ambientes Ubíquos usando o *Device Service Bus*

Gustavo Medeiros Araújo, Frank Siqueira

Departamento de Informática e Estatística – Universidade Federal de Santa Catarina
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brazil

{gustavo.medeiros,frank}@inf.ufsc.br

Abstract. *The Device Service Bus (DSB) consists in a middleware infrastructure employed for integration of heterogeneous embedded devices in ubiquitous computing environments. Based on the Devices Profile for Web Services (DPWS) as the underlying integration technology, the DSB allows the interaction among devices that adopt different networking standards. This paper presents a prototype implementation of the Device Service Bus, which provides software components that are deployed on devices responsible for building an integration bus among heterogeneous devices. A real-world application scenario with Sun SPOT sensors and RFID readers, and performance measurements obtained with these devices are also presented in this paper.*

Resumo. *O Device Service Bus (DSB) consiste em uma infra-estrutura de middleware empregada para integração de diferentes dispositivos embarcados em ambiente de computação ubíqua. Baseado no Devices Profile for Web Services (DPWS) como tecnologia de integração, o DSB permite interação de vários dispositivos que adotam diferentes padrões de comunicação. Esse artigo apresenta uma implementação de um protótipo do DSB, que provê componentes de software que são implantados em dispositivos responsáveis por construir um barramento de integração entre dispositivos heterogêneos. Um cenário de aplicação real reunindo sensores Sun SPOT e leitores RFID, e medições de desempenho obtidas com estes dispositivos também são apresentados nesse artigo.*

Palavras-Chave: Web Services, Integração de Sistemas, Computação Ubíqua.

1. Introdução

Com a constante evolução dos dispositivos computacionais, novos desafios surgem a cada dia em relação à integração de tais dispositivos, que se faz necessária para permitir a execução conjunta de tarefas computacionais. A crescente tendência de adoção destes equipamentos, que vão desde PDAs e *SmartPhones* até dispositivos com recursos computacionais mais limitados, como sensores e *tags* RFID, instiga pesquisadores à investigação, e pode vir a tornar obsoleta grande parte da infra-estrutura computacional utilizada nos dias atuais.

Muitos dispositivos não se beneficiam de uma infra-estrutura de rede estável, como ocorre com outras aplicações distribuídas. Por outro lado, esses dispositivos

dependem de protocolos de rede *ad-hoc* e não têm conhecimento prévio de outros equipamentos conectados à rede e nem dos serviços que os mesmos possam prover. Como dispositivos podem entrar e sair da rede de modo imprevisível, não há garantias da disponibilidade de serviços que estejam em execução. Tais restrições estão presentes em um cenário que foi definido por Mark Weiser como ambiente de computação ubíqua [Weiser 1993].

Com a evolução das redes sem fio e dos dispositivos móveis, outros padrões têm surgido, atendendo a diferentes aplicações e classes de dispositivos. A integração de diferentes classes de dispositivos que utilizam redes heterogêneas é ainda uma área aberta à pesquisa.

Nesse artigo é apresentada uma infra-estrutura de *middleware* que tem como objetivo prover meios para integração de diferentes dispositivos no ambiente de computação ubíqua. A solução proposta é baseada na tecnologia de Serviços Web, que tem tido sucesso em aplicações coorporativas distribuídas. Para que a execução de Serviço Web possa ser realizada em equipamentos com restrição de recursos e limitação na capacidade de comunicação, a arquitetura proposta adotou o *Device Profile for Web Services* [Microsoft 2006], um padrão projetado para construção de Serviços Web em dispositivos com tais características.

O artigo é organizado da seguinte maneira: A seção 2 apresenta conceitos e tecnologias adotadas no desenvolvimento da solução proposta e descreve projetos similares presentes na literatura. Na seção 3 é apresentada a infra-estrutura de *middleware* proposta para integração de dispositivos e são feitas comparações com algumas infra-estruturas de integração propostas na literatura. A seção 4 provê detalhes sobre a implementação do protótipo do *middleware*, e a seção 5 apresenta medições de desempenho obtidas com o protótipo desenvolvido. Por fim, a seção 6 sumariza as contribuições apresentadas no artigo e descreve algumas perspectivas para a continuação do desenvolvimento nessa área de pesquisa.

2. Contextualização e Definições

Primeiramente, a sessão atual analisa a adoção da Arquitetura Orientada a Serviços (SOA) no contexto de sistemas embarcados. Na seqüencia, é descrito o *Device Profile for Web Services* (DPWS), que é uma especificação baseada em SOA com intuito prover comunicação independente de plataforma entre diversos dispositivos computacionais móveis e embarcados. Por fim, são apresentadas infra-estruturas de integração propostas na literatura e são discutidas suas limitações , que nos levaram a propor uma nova infra-estrutura de *middleware* para integração de dispositivos em um ambiente de computação móvel e ubíqua.

2.1 Serviços Web e Dispositivos Embarcados

A Arquitetura Orientada a Serviço (SOA) pode ser definida como um paradigma que permite a construção de componentes de software com baixo acoplamento, que provê serviços a cliente que podem ser localizados dinamicamente e invocados usando um protocolo de comunicação conhecido pelas partes.

Serviços Web consistem na tecnologia mais popular para construção de software baseado na arquitetura SOA. A tecnologia de Serviços Web adota padrões largamente disponíveis para representação e comunicação de dados, basicamente XML, HTTP e SOAP. A adoção desses padrões permite a implantação de serviços web em

virtualmente qualquer ambiente computacional [W3C 2004], habilitando dessa forma a interação entre provedores de serviço (i.e servidores) e consumidores (clientes) em um ambiente heterogêneo de computação.

SOA e Serviços Web têm sido adotados com sucesso em ambiente de negócio, onde diferentes sistemas de informação aplicados para construção de regras de negócio têm se transformado em serviços, facilitando a comunicação e a troca de informações entre sistemas. Essa integração de sistemas permite que as corporações redirecionem o foco no desenvolvimento de processos de negócio, ao invés de gastar recursos na operação e manutenção de sistemas. Apesar de ser o cenário mais comum na qual SOA é usada atualmente, este não é o único cenário na qual SOA é capaz de prover integração entre *softwares* [Machado 2006].

Durante os últimos anos, houve um grande aumento na fabricação de dispositivos programáveis – como telefones móveis, PDAs, maquinário industrial, equipamentos de *health care* e vários outros – e a disponibilização destes no dia a dia das pessoas tem sido notada. Tais equipamentos têm evoluído seu poder de processamento e vêm ganhado novas funcionalidades, que incluem a capacidade de comunicação entre pares utilizando redes sem fio e também de executar protocolos de rede de alto nível que são requeridos pela arquitetura SOA.

Apesar do rápido crescimento na produção e venda de dispositivos embutidos, a falta de adoção de padrões comuns de comunicação e para representação de dados torna difícil e muitas vezes inviável a interação entre os dispositivos. Algumas tecnologias como UPnP [UPnP Forum 2006]e Jini [Sun 2005] foram propostas com o objetivo de padronizar a comunicação e mecanismos de localização de serviços, permitindo interação entre equipamentos. Jini provê uma solução completa para permitir interoperabilidade entre dispositivos, entretanto é limitada à plataforma Java. UPnP adota padrões de implementação de Internet bem conhecidos como HTTP, SOAP e XML para comunicação entre dispositivos, mas sua adoção é limitada a redes locais.

A próxima sessão descreve o DPWS, uma especificação de protocolo proposta por um grupo de empresas que objetiva permitir interoperabilidade entre dispositivos não sendo limitado pelo escopo da rede e pelas características de plataforma computacional em questão.

2.2 O *Device Profile for Web Services*

O *Device Profile for Web Services* é uma pilha de protocolos que provê mecanismos de comunicação de alto nível para interoperabilidade entre dispositivos. Seguindo os padrões utilizados pelos Serviços Web, DPWS une o cenário de sistemas embarcados com o mundo de aplicações baseadas em SOA, proporcionando às aplicações embarcadas o mesmo nível de interoperabilidade disponível para sistemas de informação. A pilha de protocolos, apresentada na Figura 1(a), demonstra como o DPWS emprega protocolos padrão da Internet, como TCP ,UDP unicast e *multicast*, e HTTP. Para troca de mensagens, o protocolo SOAP é empregado tanto sobre UDP quanto sobre HTTP.

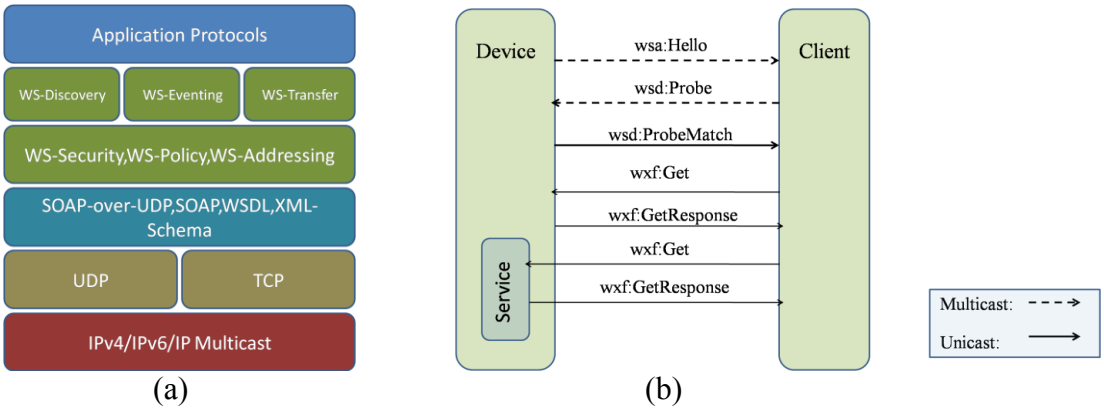

Figura 1. Protocolos (a) e Mensagens (b) do DPWS

Como mostra a Figura 1(a), os principais protocolos padrão utilizados pelos serviços web também são adotados pela especificação DPWS. Esses padrões são:

- *WS-Addressing*: encapsula todas as informações de endereçamento de rede em campos no cabeçalho da mensagem, como "To:" e "Reply to:". Isso permite ao protocolo SOAP ser independente do protocolo de transporte.
- *WS-MetadataExchange*: provê acesso dinâmico aos metadados que descrevem o *hosted* e *hosting service*.
- *WS-Discovery*: define um mecanismo de descobrimento de dispositivos, baseado em mensagens *multicast* para localização de serviços disponíveis na rede. Para estender o descobrimento para outras redes, o padrão define o *Discovery Proxy,* que reduz o tráfego de mensagens geradas pelo uso do protocolo *multicast*.
- *WS-Policy*: provê um modo de adicionar informações à descrição WSDL, que são especificadas na forma de políticas suportadas pelos serviços.
- *WS-Eventing*: define um conjunto de operações para notificação de eventos, permitindo que serviços publiquem e recebam mensagens assíncronas.
- *WS-Security*: provê mecanismos de segurança para interação entre equipamentos, garantindo propriedades de segurança como autenticação, confidencialidade e integridade.

Os metadados trocados entre os dispositivos descrevem suas características, como o nome e URL do fabricante, código e nome do modelo, número serial, *friendly name* e versão do *firmware*. São descritos ainda diversos detalhes sobre os serviços fornecidos pelo dispositivo, como o WSDL, no qual são especificadas as operações, faltas e estruturas de dados suportadas pelo serviço, e as relações entre serviços.

As mensagens definidas pela especificação DPWS, que são trocadas entre os dispositivos durante o processo de descobrimento, são mostradas na Figura 1(b). A mensagem *Hello* é um anúncio *multicast* realizado pelo dispositivo quando entra na rede. A mensagem *multicast Probe* é enviada por um cliente quando busca por um dispositivo com características particulares descritas no corpo da mensagem. Um ou mais dispositivos que corresponderem às características procuradas respondem com a mensagem *ProbeMatch*, contendo o metadado no qual os serviços são listados. A descrição dos serviços disponíveis pode ser obtida enviando a mensagem *Get*, que resulta na mensagem *GetResponse,* que é retornada ao cliente contendo a descrição do serviço requerido.

2.3 Trabalhos Relacionados

A literatura científica é rica em esforços de pesquisa que têm como objetivo prover interoperabilidade entre dois protocolos padrão de comunicação. Destacamos aqui o trabalho de [Siegemund e Flörkemeier 2003], que provê interconectividade entre Bluetooth e dispositivos RFID, utilizando Bluetooth como ponto de acesso, e assim provendo mobilidade aos Leitores RFID. Como o objetivo deste projeto se limita a duas tecnologias específicas, ele não provê uma solução completa para integração de dispositivos em ambiente ubíquo.

Por outro lado, projetos como [Yim, Oh, Hwang e Lee 2007] e [Raverdy, Riva, de La Chapelle, Chibout e Issarny 2006] têm como objetivo permitir que dispositivos com diferentes tecnologias de rede possam ser interconectados. Entretanto, as soluções propostas em [Yim, Oh, Hwang e Lee 2007] e [Raverdy, Riva, de La Chapelle, Chibout e Issarny 2006] requerem infra-estruturas robustas para gerenciar dispositivos e serviços, sendo assim, não podem ser implantadas em dispositivos com recursos de rede e computacionais limitados.

Com relação à limitação apresentada pelos projetos de pesquisa relatados, a proposta de um *middleware* de integração para interconectar todos os tipos de dispositivos, que empregam diversos tipos de tecnologias de comunicação, permanece uma oportunidade em aberto para pesquisa.

3. O *Device Service Bus*

O *Device Service Bus* (DSB) é uma infra-estrutura de *middleware* que objetiva prover integração de dispositivos heterogêneos em ambiente de computação ubíqua. O DSB é uma infra-estrutura de software baseada em uma plataforma portável e leve que pode ser executada desde em máquinas como estações de trabalho até em dispositivos com recursos computacionais limitados, como celulares e PDAs. Essa infra-estrutura é baseada no paradigma SOA, permitindo interação entre provedores e consumidores de serviço agindo como um *Service Broker*.

O principal objetivo dessa infra-estrutura é criar um barramento de comunicação, mostrado na Figura 2, através do qual equipamentos com tecnologias específicas, como RFID [Want 2006], Bluetooth [Chatschik 2001] e Sun SPOT [Sun 2004], possam disponibilizar serviços na rede. A interoperabilidade entre dispositivos é possível de realizar utilizando DPWS e componentes de integração de tecnologias específicas. Isso permite que dispositivos com recursos computacionais limitados executem a pilha de protocolos do DPWS e se integrem a outros dispositivos com suporte nativo para DPWS.

Figura 2. Visão Geral do *Device Service Bus*

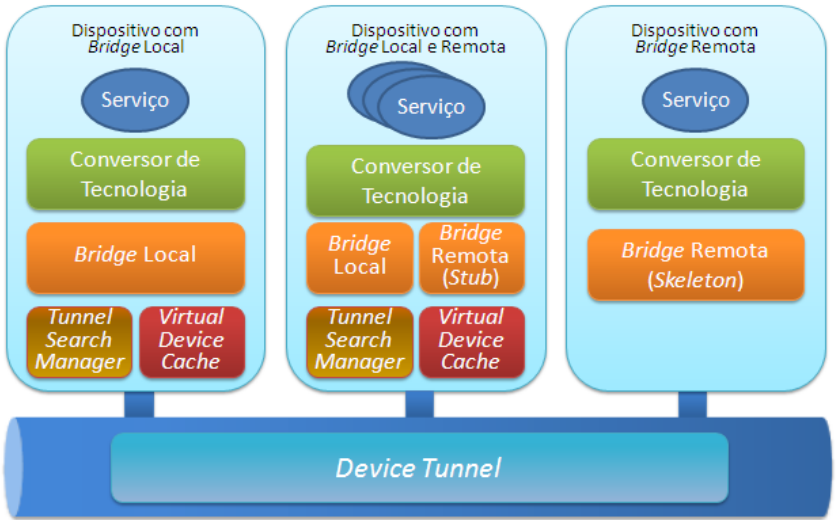

Figura 3. Componentes do *Device Service Bus*

3.1 Componentes do *Middleware*

O *Middleware* proposto é composto por cinco tipos de componentes, que são mostrados na Figura 3.

O *Device Tunnel* é o componente capaz de descobrir seus pares, bem como outros dispositivos com suporte nativo para DPWS. O *Device Tunnel* foi construído como um *hosting service*, no qual é implementada a pilha de protocolos do DPWS.

O *Virtual Device Cache* mantém uma lista de *Virtual Devices* conhecidos, que provêem interfaces DPWS para dispositivos que não implementam ou não suportam DPWS. Cada *Virtual Device* pode hospedar um ou mais *Virtual Services*, que representam serviços implementados por dispositivos não-DPWS. Além disso, um *Virtual Service* pode ter um ou mais *Virtual Action* e *Virtual Events*, que representam, respectivamente, as ações e eventos implementados por esses serviços.

A *Bridge* permite a interação entre dispositivos DPWS e *Virtual Devices*. Quando um *Virtual Device* é adicionado ou removido pela *Bridge*, mensagens de *Hello* e *Bye* são enviadas via *multicast*, conforme definido na especificação DPWS [Microsoft 2006]. Invocações de serviços enviadas para os *Virtual Devices* são encaminhadas para a *Bridge*, que as repassa para o Converter correspondente ao dispositivo. A *Bridge* pode ser tanto local (ou seja, pode executar no próprio dispositivo) como remota. Uma *Local Bridge* pode interagir com o *Device Tunnel* local, enquanto a *Remote Bridge* precisa localizar um através da rede.

Depois que um *Device Tunnel* é encontrado, a *Remote Bridge* precisa adicionar ao *Device Tunnel* todos os *Virtual Devices* gerenciados por esta *Bridge*, que são mantidos em sua *cache*. Em geral, opta-se por utilizar a *Remote Bridge* quando o dispositivo não tem capacidade suficiente para comportar todos os componentes do DSB. Conceitualmente, a *Remote Bridge* é similar ao *Discovery Proxy* definido pela especificação *WS-Discovery* [Microsoft 2005].

Um *Converter* gerencia um ou mais dispositivos que empregam a mesma tecnologia de rede. O *Converter* conhece detalhes de implementação dos dispositivos, sua interface e seus serviços. O *Converter* é responsável por extrair metadados dos dispositivos reais,

fornecer a descrição dos seus serviços e por encaminhar invocações para tais serviços utilizando os mecanismos de comunicação suportados por cada dispositivo hospedeiro.

O *Tunnel Search Manager* é responsável por realizar buscas por dispositivos e serviços requeridos pelos *Converters*. Essa busca pode ser realizada local e remotamente. No caso da busca local, o componente executa a requisição de busca na *Virtual Device Cache* do próprio DSB. A busca remota é realizada por meio do envio de mensagens *multicast* na rede (*Probe*). Caso o dispositivo e serviços sejam encontrados, seus metadados e a descrição dos serviços por ele fornecidos são repassados ao *Converter*, que nesse caso faz papel de cliente em busca de um provedor de serviços. O *Converter* processa o retorno da requisição utilizando o mecanismo de comunicação suportado pelo dispositivo e a repassa para o dispositivo real. Para realizar a invocação, o *Converter* utiliza a *Remote Bridge* ou a *Local Bridge*.

O componente *Device Tunnel* expõe dispositivos e serviços encontrados pela *Bridge* e *Converters*. A *Bridge* manipula todos os *Converters* disponíveis e é responsável por conectar os *Virtual Devices* ao núcleo da arquitetura. O *Converter* lida com assuntos específicos da tecnologia para todos os dispositivos que empregam uma tecnologia em particular. Por essas razões, o DSB torna equipamentos que não implementam DPWS capazes de interagir com outros equipamentos através da sua infra-estrutura.

Os dispositivos conectados ao DSB têm seu próprio ciclo de vida, e podem ficar indisponíveis sem aviso prévio. Por isso, os *Converters* têm um mecanismo de *keep-alive* que mantém a rastreabilidade dos dispositivos ativos. Se um aviso de *time-out* ocorre para algum dispositivo, o *Converter* notifica a *Bridge*, que remove o dispositivo da *Virtual Device Cache*. O número de dispositivos que podem ser conectados ao DSB e o tempo de *time-out* do mecanismo de *keep-alive* podem ser configurados. Além disso, o mecasnimo de *keep-alive* pode ser desabilitado em ambientes com restrições de consumo de recursos computacionais e de rede.

3.2 Análise

O DSB emprega uma abordagem para integração de dispositivos que é similar à estratégia adotada por outros *middlewares* de integração encontrados na literatura, como [Yim, Oh, Hwang e Lee 2007] e [Raverdy, Riva, de La Chapelle, Chibout e Issarny 2006], os quais permitem que dispositivos de diferentes tecnologias de rede interoperem. Entretanto, diferindo das outras propostas, a base do protocolo DSB é leve e portável, podendo ser implantada em dispositivos com poucos recursos computacionais, como PDAs, *Smartphones* e *Set-top boxes*. Além disso, o DSB pode ser disponibilizado em vários dispositivos, assim criando vários pontos de acesso com a finalidade de trocar informações através do DSB.

A base do projeto DSB é uma implementação DPWS, que permite ao projeto ganhar com os benefícios do baixo acoplamento da arquitetura orientada a serviços, construída sobre protocolos como SOAP, HTTP e UDP *multicast* e *unicast*. A capacidade *plug and play*, que não está presente nos serviços web comuns, permite a descoberta dinâmica de serviços e simplifica o esforço necessário para expor serviços providos por dispositivos. Essa característica é empregada pelo DSB para expor dispositivos e serviços que não suportam DPWS nativamente.

4. Implementação e Testes

A implementação do protótipo do DSB foi desenvolvida sobre o *framework* WS4D, no qual o *Device Tunnel* foi construído e estendido para poder lidar com *Virtual Devices* e *Virtual Services*. Os componentes do DSB foram construídos em Java ME CDC 1.1.2.

Cada *Virtual Device* tem seu próprio metadado, o qual corresponde à descrição de um dispositivo real, com número serial, número e nome do modelo, nome do fabricante e *friendly name*, com o objetivo de atender à especificação DPWS. Um *Virtual Device* encapsula a descrição dos *Virtual Services* estabelecendo uma associação de um-para-muitos. O *Device Tunnel* expõe como serviço web todos os V*irtual Devices* e V*irtual Services* associando um endereço único para cada um deles. Este endereço é usado pelo *Device Tunnel* para demultiplexar requisições e informar à *Bridge* qual *Virtual Device* e *Virtual Service* foram invocados. A *Bridge* mantém em sua *cache* uma lista de *Converters*, e cada *Converter* mantém um ponto de conexão para os dispositivos que empregam uma tecnologia de comunicação específica.

No protótipo, três *Converters* foram implementados, dois para tecnologia RFID (Mercury M5 e M5e *Readers*) e outro para sensores Sun SPOT.Um quarto *Converter*, para Bluetooth, está em fase de testes. O protótipo foi testado com Sun Java Toolkit 1.0 para CDC. Dispositivos com suporte a Java ME CDC, como PDAs, *Smartphone* e *Set-top boxes* podem executar DPWS nativamente, não sendo necessário ter um *converter* para interagir com outros dispositivos.

5. Medições de Desempenho

Testes foram realizados com a intenção de avaliar a capacidade do *Device Service Bus* de lidar com vários *Virtual Devices* e requisições simultâneas para os *Virtual Services*. Os testes foram realizados em um ambiente com a seguinte configuração:

- Cliente implementado utilizando *Sun Java Toolkit 1.0 for CDC*, sendo executado em uma máquina com processador Pentium 4 com 2.4 GHz, 512 MB de Memória e sistema operacional Windows XP.
- DSB (*Local Bridge*) implementado utilizando Java SE 1.5, executado em um processador Intel Core 2 Duo 2.0 GHz com 2GB de memória e sistema operacional Windows Vista.
- DSB (*Remote Bridge*) implementado em Java SE 1.5, executado em processador Intel Core 2 Duo 2.0 GHz com 2GB de memória e sistema operacional Windows XP.
- Pilha DPWS com WS4D estendido.
- Roteador Wireless 802.11g com velocidade de 54Mbps.
- Três tipos de *tags* RFID passivas, modelos UMP RAFLATAC *shortdipole*, ALN-9540 e RR;
- Leitor RFID modelo Mercury M5e;
- Dois sensores Sun SPOT e uma estação-base com SPOT SDK versão 4.0;

O experimento foi realizado com um dispositivo Sun SPOT conectado via estação-base a um computador e o leitor RFID M5e ligado a outro computador no qual foram simulados outros 98 *Virtual Devices*. O metadado e serviços do Sun SPOT foram expostos por uma *bridge* local DSB e do leitor RFID M5e foram expostos utilizando

uma *bridge* remota. O cliente e o DSB (*RemoteBridge*) executavam em máquinas separadas, com o Mercury M5e RFID conectado ao computador via cabo serial. O intuito do experimento era realizar medições de tempo de resposta da requisição de um cliente DPWS remoto aos dois dispositivos, utilizando o DSB com e sem a *RemoteBridge*. Todas as requisições foram feitas para os *Virtual Devices* que representavam os dispositivos. Para cada rodada de testes, foram realizadas 100 execuções e obtida a média dos tempos de resposta.

No experimento realizado com o dispositivo Mercury M5e RFID Reader, cada *Virtual Device* tinha um *Virtual Service* e este tinha uma *Virtual Action* com um parâmetro de entrada e um de saída. O leitor RFID tem um serviço que efetua a leitura de *tags* e retorna o conteúdo das *tags* lidas. Foi avaliado o tempo de invocação do serviço, variando a quantidade de dados retornados pelo leitor RFID. Na medição do tempo de invocação foi considerado somente o tempo de invocação do *Virtual Action* e o recebimento da resposta, que variou de 10 *KBytes* a 100 *KBytes*. A Figura 4 mostra os resultados dos testes. O resultado mostra que o tempo de resposta cresce a uma taxa menor que o tamanho da resposta. Isso mostra que o mecanismo de invocação tem funcionamento adequado mesmo no caso de serviços que apresentem altas taxas de transmissão de dados.

Também foi testada no experimento a possibilidade de o sensor Sun SPOT atuar como cliente em busca de outros dispositivos e serviços, nesse caso o leitor RFID M5e**.** Medições também foram realizadas para esse caso, e o resultado é demonstrado na Figura 5.

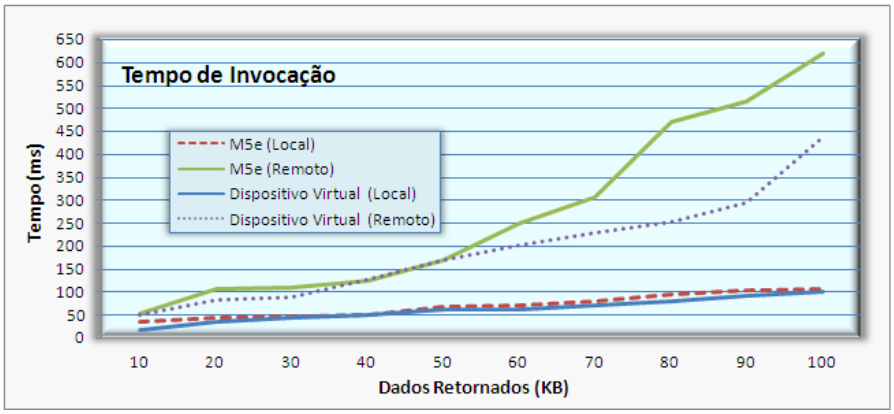

Figura 4. Tempo de Invocação para M5e e Virtual Devices

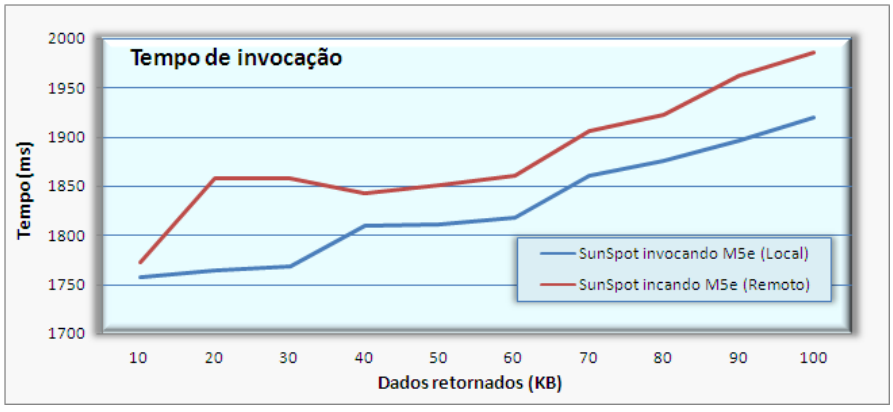

Figura 5. Tempo de Invocação do Sun SPOT como cliente do Leitor RFID M5e.

6. Conclusão

O presente artigo apresentou o *Device Service Bus*, uma infra-estrutura de *middleware* para integração de dispositivos heterogêneos que é baseado na especificação DPWS (*Device Profile for Web Service*). O DSB integra dispositivos que empregam diferentes tecnologias de comunicação, como RFID, Bluetooth e Sun SPOT, permitindo que dispositivos e serviços sejam expostos como serviços web.

A solução proposta para integração de dispositivos é mais leve que outras soluções propostas encontradas na literatura, sendo capaz de executar tanto em equipamentos móveis como PDA e *Smartphones*, como uma estação de trabalho. Além disso, dispositivos com recursos limitados como sensores e tags RFID podem ser integrados através de dispositivos de interconexão, que hospedem *Bridges* e *Converters* de tecnologias específicas.

Atualmente, estão sendo finalizados os testes dos *Converters* para Bluetooth e a inicialização de *Converter* para dispositivos GPS. No futuro, planejamos trabalhar no desenvolvimento de outros *Converters* para incorporar mais dispositivos e outras tecnologias de comunicação ao DSB, com a intenção de ter diferentes classes de dispositivos em um ambiente ubíquo e heterogêneo.

Referências

Weiser, M. (1993) Hot Topics: Ubiquitous Computing. IEEE Computer, 26(10):71–72.

Microsoft Corporation. Devices Profile for Web Services (DPWS). (2006). Available at http://schemas.xmlsoap.org/ws/2006/02/devprof/.

W3C. (2004)Arquitetura de Serviço Web. Available at http://www.w3.org/TR/ws-arch/.

Machado, Guilherme Bertoni et al.(2006) Integration of Embedded Devices Through Web Services: Requirements, Challenges and Early Results. Proceedings of the 11th IEEE Symposium on Computers and Communications (ISCC'06). Calgary, Italy.

UPnP Forum.(2006) Arquitetura de Dispositivo UPnP v1.0. Available at http://www.upnp.org/resources/documents.asp.

Sun Microsystems.(2005) Especificações JINI Archive v2.1. Available at http://java.sun.com/products/jini/2_1index.html.

Want. R. (2006) An Introduction to RFID Technology. IEEE Pervasive Computing, Vol 5(1).

Chatschik, B.(2001) An Overview of the Bluetooth Wireless Technology. IEEE Communications Magazine, Vol. 39(12).

Microsoft Corporation. (2005) The Web Services Dynamic Discovery (WS-Discovery). Available at http://specs.xmlsoap.org/ws/2005/04/discovery/.

Sun Microsystems.(2008) Getting Started with Sun SPOT. Available at http://www.Sun SPOTworld.com/docs/.

Siegemund, F., Flörkemeier, C. (2003) Interaction in Pervasive Computing Settings using Bluetooth-Enabled Active Tags and Passive RFID Technology together with Mobile Phones. Proc. IEEE PerCom.

Yim, Hyung-Jun., Oh, Il-Jin., Hwang, Yun-Young., Lee ,Kyu-Chul.(2007) Design of DPWS Adaptor for Interoperability between Web Services and DPWS in Web Services on Universal Networks .Proceedings of the IEEE International Conference.

Raverdy, Pierre-Guillaume; Riva, Oriana; de La Chapelle, Agnès; Chibout, Rafik; Issarny, Valérie: (2006) Efficient Context-aware Service Discovery in Multi-Protocol Pervasive Environments. Proceedings of the IEEE 7th International Conference on Mobile Data Management (MDM'06).