ERLab: A *Middleware* for Remote Access Electronic Laboratories

Admilson R. L. Ribeiro, Marco T. Chella, Luiz Marcus M. A. Santos, Joanna C. S. Santos, Wedla R. Melo

Universidade Federal de Sergipe
Departamento de Computação
Avenida Marechal Rondom, S/N, Jardim Rosa Elze
CEP 49.100-000, São Cristóvão, SE – Brazil
{admilson,chella}@ufs.br, {luizm1000,joannacss,wedlaa}@hotmail.com

ABSTRACT
The main difficulty in building online research laboratories is the use of computing resources and electronic instruments in multiple hardware and software platforms. Thus, this paper describes a service-oriented middleware using Web Services called ERLab (Electronic Remote Laboratory). The main goal of ERLab is to share data and scientific instruments, which operate and communicate over different protocols for different users on the Internet.

RESUMO
A principal dificuldade em projetar laboratórios de pesquisas *online* é a utilização de recursos computacionais e instrumentos eletrônicos em múltiplas plataformas de *hardware* e *software*. Dessa forma, nesse artigo é apresentado um *middleware* orientado a serviços utilizando Web Services chamado ERLab (*Electronic Remote Laboratory*). O principal objetivo do ERLab é o compartilhamento de dados e instrumentos científicos, que operam e se comunicam através de protocolos diferentes, para diversos usuários através da Internet.

Categories and Subject Descriptors
C.2.4 [**Computer Systems Organization**]: Distributed Systems - – Distributed Applications

General Terms
Experimentation, Design, Standardization, Languages

Keywords
middleware, Remote Access Laboratories, Web Services.

Palavras-chave
middleware, Laboratórios de Acesso Remoto, Web Services.

6th EATIS, May 23–25, 2012, Valencia, Spain.

1. INTRODUÇÃO

Um dos objetivos das instituições de ensino superior é a realização de pesquisas científicas que se destinam à formação de excelência de docentes, pesquisadores e profissionais para atuarem na elaboração e difusão do saber e no desenvolvimento da ciência e da tecnologia. Esse objetivo é normalmente atingido através de pesquisas em laboratórios que são montados, em sua maioria, através de verbas conseguidas com submissões de projeto aos órgãos fomentadores. Entretanto, além de atender a um número limitado de pesquisadores, a instalação e a manutenção de um laboratório de pesquisa bem estruturado e eficiente acarretam custos financeiros muito elevados.

Uma alternativa para contornar essas deficiências, são os chamados laboratórios de experimento *online*. Através desses laboratórios, obtêm-se procedimentos que garantem as atividades de experimentos de pesquisa, de forma autônoma, aumentando a disponibilidade efetiva dos laboratórios e propiciando o atendimento a um grande número de pesquisadores [4]. Dessa forma, há uma racionalização dos custos associados à pesquisa.

Os atuais laboratórios de experimento *online* são classificados em duas categorias: laboratórios virtuais que fornecem um ambiente de simulação, em que pesquisadores ou estudantes conduzem seus experimentos, e laboratórios de acesso remoto que permitem que pesquisadores e estudantes usem uma GUI (*Graphical User Interface*) para operar instrumentos reais [19].

Diferentemente dos laboratórios virtuais, os laboratórios remotos proporcionam uma vivência consistente com os equipamentos através da experimentação à distância controlada pelo pesquisador. Eles podem até mesmo aumentar a faixa de experimentos disponíveis em uma instituição e prover, para os pesquisadores à distância, experimentos em tempo real [13]. Através da utilização de laboratórios remotos é possível otimizar o uso dos recursos, pois eles ficam disponíveis por um grande período de tempo e podem ser acessados por várias instituições de pesquisas. Consequentemente um maior número de pesquisadores pode participar simultaneamente dos experimentos em diferentes locais e sem restrição de tempo.

Existem diversos trabalhos publicados na literatura sobre laboratórios de acesso remoto. Alguns como iLab (http://ilab.mit.edu/iLabServiceBroker/), Prolearn (www.prolearn-project.org) e eMerge (http://emerge.teikoz.gr/website/index.htm)

que foram projetados com fins de pesquisa científica. Essas soluções apresentam diversas limitações sendo a principal delas a dificuldade em projetar laboratórios de pesquisas *online* com a utilização de recursos computacionais e instrumentos eletrônicos em múltiplas plataformas de *hardware* e *software*. As soluções existentes comumente são limitadas por apresentarem as seguintes características [14]: (1) arquitetura cliente-servidor, (2) *software* proprietário do fabricante, (3) instrumentos de uma única companhia e, (4) sistema operacional Microsoft Windows.

A solução apresentada neste artigo objetiva superar essas limitações através do desenvolvimento de um *middleware* chamado ERLab (*Electronic Remote Laboratory*), que provê duas características fundamentais: interoperabilidade e flexibilidade. Este artigo está organizado como segue: na seção 2, são apresentados os trabalhos relacionados; na seção 3 são descritas as metas de projeto consideradas no desenvolvimento do modelo do ERLab; na seção 4, a arquitetura do ERLab é exposta; na seção 5, são explicitadas algumas questões de desenvolvimento do ERLab; na seção 6, se faz uma avaliação qualitativa do *middleware* e apresenta-se um exemplo de uso do *middleware* e na seção 7 são feitas as considerações finais.

1. TRABALHOS RELACIONADOS

Além dos trabalhos mencionados na introdução deste artigo, outras plataformas de laboratórios de pesquisa online apresentam as mesmas dificuldades sobre a interoperabilidade entre equipamentos e detecção automática de instrumentos.

Em [4] é descrita a arquitetura de um laboratório de acesso remoto com aplicação no ensino de engenharia eletrônica. A proposta tem como objetivo desenvolver uma ferramenta para criar experimentos remotos e disponibilizar recursos para a elaboração de experimentos que envolvam o controle de instrumentos, acionamento de relés e chaves eletrônicas, por exemplo, sem haver preocupação com a comunicação do *hardware* dos instrumentos e a integração com a rede Internet.

O LADIRE [1] é um laboratório de acesso remoto que permite aos usuários realizarem experimentos práticos através da Web. O sistema é composto por uma unidade servidora e um determinado número de unidades clientes. A unidade servidora está localizada no laboratório remoto, hospedando o servidor Web e permite que a bancada de medições seja controlada. Cada unidade cliente pode realizar, por meio de dispositivos apropriados ligados ao computador, a emulação de medições de uma bancada de medição real.

Em [3] é apresentado um laboratório de microcontroladores de acesso remoto baseado na Web, desenvolvido pela Afyon Kocatepe Universitesi, que se baseia em simulações para que os usuários possam realizar seus experimentos em microcontroladores. O experimento ocorre da seguinte forma: o cliente envia o código do seu programa, previamente compilado, do computador cliente para um servidor Web. Após o processo de programação, entradas digitais e analógicas passam a ser controladas pelo circuito conectado à porta serial do servidor. O experimento pode ser visto através de uma câmera Web.

Outro trabalho relacionado com a disponibilização remota de recursos laboratoriais é o AIM-Lab [9]. A arquitetura desse laboratório remoto baseia-se no modelo cliente-servidor, no qual os clientes (estudantes) se comunicam através da Internet com um servidor e seu experimento usando navegadores Web. O AIM-Lab é atualmente dedicado a experimentos com um grupo de dispositivos de teste que incluem um conjunto de dispositivos semicondutores e um conjunto de diodos emissores de luz.

2. METAS DE PROJETO

O ERLab está baseado em dois pilares: interoperabilidade e flexibilidade. A interoperabilidade é fornecida através de um *middleware* orientado a serviços utilizando Web Services, que são adequados para prover interação entre máquinas em uma rede de computadores e integrar recursos computacionais heterogêneos. Por flexibilidade, entende-se a facilidade em realizar modificações, tanto em *hardware* quanto em *software*. A detecção *plug-and-play* (conecte-e-opere) permite que sejam incorporados novos equipamentos sem a necessidade de paralisação da rede de comunicação.

2.1 Interoperabilidade

Desde cedo se compreendeu que a interoperabilidade entre instrumentos era muito difícil devido ao fato de cada instrumento possuir a sua própria interface de programação. Assim, programadores usavam comandos que eram dependentes do instrumento. Essa falta de padronização significava que mesmo dois multímetros digitais do mesmo fabricante poderiam não usar o mesmo conjunto de comandos. Para resolver este problema, um conjunto de empresas formou em 1990 o consórcio SCPI (*Standard Commands for Programmable Instrumentation*) [8]. A plataforma SCPI propõe um conjunto de regras para uniformizar as mensagens ASCII (*American Standard Code for Information Interchange*) que servem de base à programação dos instrumentos. Este conjunto definido de comandos para controlar instrumentos usa caracteres ASCII, fornecendo pouca padronização e consistência de comandos usados para controlar instrumentos.

A plataforma SCPI teve pouco sucesso, pois os comandos SCPI ainda eram destinados a um específico instrumento em vez de endereçar um conjunto de dispositivos de um mesmo tipo. Assim o problema de interoperabilidade continuou, porém, em 1998, um conjunto de empresas liderado pela National Instruments formou um consórcio chamado IVI (*Interchangeable Virtual Instruments*). A plataforma IVI propõe um conjunto de *drivers* de alto nível, através dos quais é possível se comunicar com qualquer instrumento independentemente da sua marca ou modelo. Usando um *driver*, pode-se acessar o instrumento chamando uma sub-rotina em determinada linguagem de programação em vez de formatar e enviar uma string ASCII como se faria com SCPI.

Além de um *driver* de instrumento, é necessário também manipular uma biblioteca de I/O (*Input/Output*) para formatar e enviar os comandos, e depois construir a resposta em uma variável. Todos os *drivers* de instrumento IVI precisam se comunicar com o dispositivo através de uma biblioteca de I/O. Normalmente, é utilizada a plataforma VISA (*Virtual Instrument Software Architecture*), uma biblioteca padrão largamente utilizada para comunicação com instrumentos. A conjugação das plataformas IVI e VISA permite a criação de *software* totalmente independente do *hardware*. A independência se faz ao nível das interfaces de comunicação (plataforma VISA) e ao nível dos instrumentos (plataforma IVI).

Entretanto, a combinação das plataformas IVI e VISA levou a uma interoperabilidade limitada dos instrumentos, pois ocorreu uma proliferação de plataformas IVI-VISA. Cada fabricante possui sua própria plataforma IVI-VISA. Com isso, se torna complicada a troca de um mesmo tipo de instrumento de determinado fabricante por outro instrumento de fabricante diferente em um mesmo sistema.

2.2 Flexibilidade

Geralmente um laboratório de *hardware* possui osciloscópios, multímetros, fontes de tensão e outros instrumentos de diferentes

fornecedores. A instalação, configuração e integração desses equipamentos heterogêneos é um enorme desafio, consome muito tempo e é sujeito a erros mesmo para especialistas.

Como solução, o ERLab proporciona a flexibilidade de que um componente ao ser conectado ao servidor seja incorporado discretamente sem que o sistema seja paralisado. Para alcançar esse objetivo, o sistema apresenta a facilidade *plug-and-play*. Para suportar a implantação e detecção de dispositivos *plug-and-play*, o ERLab é baseado no padrão IEEE 1451 [18] e usa o padrão TEDS (*Transducer Electronic Data Sheet*) para geração dinâmica das descrições dos dispositivos virtuais.

3. ARQUITETURA DO ERLAB

Para suportar as metas de projeto, o ERLab é constituído das seguintes camadas: camada física, camada de mapeamento de dispositivos, camada de descrição de serviços, camada de comunicação de serviços e camada de aplicação como pode ser visto na figura 1.

Figura 1. Arquitetura do *middleware* ERLab

3.1 Camada física

A execução remota de experimentos, utilizando como recurso de comunicação a rede *Internet*, demanda dispositivos e programas que propiciem ao usuário uma experiência mais próxima possível a que ele teria caso estivesse realizando as atividades presencialmente no laboratório. Essas atividades incluem controlar e obter dados de instrumentos comuns a laboratórios de eletrônica como osciloscópios, geradores de função, placas de Entrada/Saída, entre outros. Esses instrumentos estão situados na camada física.

Assim, a camada física é constituída de vários dispositivos e instrumentos eletrônicos que estão conectados através de uma rede de comunicação. Esses dispositivos são conectados ao servidor através da interface USB (*Universal Serial Bus*). Nesta camada podem existir ainda dispositivos que, embora conectados na interface USB, simulam uma interface de comunicação serial (RS232).

3.2 Camada de Mapeamento de Dispositivos

Essa camada se comunica com uma variedade de dispositivos e equipamentos eletrônicos representando-os para as outras camadas do *middleware* ERLab de uma maneira uniforme. Ela faz aquisição de dados para um tipo específico de dispositivo e processa esses dados para posterior armazenamento. Assim, nessa camada se converte qualquer instrumento eletrônico da camada física em um instrumento virtual (*Virtual Instrument*), que é constituído de vários serviços que podem ser consumidos ou fazerem parte de uma composição de outros serviços. Dessa forma, programadores podem definir novos serviços sem precisar entender os detalhes da camada física. O desacoplamento da definição de serviços de instrumentos de sua plataforma garante facilidade de integração e permite introduzir novas tecnologias quando disponível.

Um *Virtual Instrument* (VI) é o principal componente desta camada e é definido em um simples arquivo XML (*Extensible Markup Language*) [16]. Ele abstrai os detalhes de implementação dos diversos instrumentos existentes e pode descrever qualquer tipo de instrumento eletrônico a ser controlado remotamente. A especificação de um VI fornece todas as informações requeridas para a implantação e utilização do instrumento.

Sendo assim, cada dispositivo conectado ao servidor deve possuir um arquivo XML compatível com o esquema de um VI. Esse esquema é definido em um arquivo XSD (*XML Schema Definition*) [17]. Nesse arquivo é especificada a estrutura do documento XML de um *Virtual Instrument*. Na figura 2 são apresentados os principais aspectos definidos nesse esquema XML representado em um diagrama de classes UML (*Unified Modeling Language*).

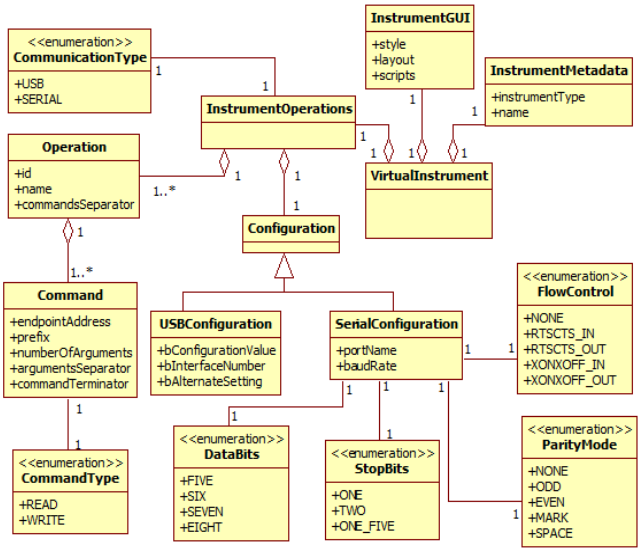

Figura 2. Esquema de um *Virtual Instrument*

Conforme observado na figura 2, um *Virtual Instrument* é composto de três partes principais: metadados, operações e interface gráfica do usuário.

Os metadados de um instrumento, *InstrumentMetadata*, abrangem informações sobre o tipo de instrumento e o seu nome.

Na seção de descrição das operações do instrumento, *InstrumentOperations*, são descritas as operações do instrumento, especificando os comandos necessários para realização da operação. Nessa seção do documento XML também é descrita a configuração necessária para efetuar a comunicação com o instrumento (por exemplo, nome da porta serial no caso de dispositivos que simulem esse tipo de interface de comunicação).

A parte da interface gráfica do usuário (*InstrumentGUI*) abrange os componentes gráficos utilizados para o controle do dispositivo pelo usuário. Nela o *layout* da interface gráfica do usuário para o controle do dispositivo é descrito usando HTML (*HyperText Markup Language*).

Para realizar o mapeamento entre um instrumento conectado à camada física, cada arquivo XML é nomeado pelos

identificadores do vendedor (idVendor) e do produto (idProduct) do tipo específico de instrumento ao qual o arquivo descreve. Esses identificadores são obtidos através de um descritor USB [2].

Dessa forma, para cada dispositivo conectado ao servidor, observam-se os identificadores do vendedor e do produto e realiza-se o mapeamento entre um instrumento real e um instrumento virtual descrito em um arquivo XML. Com isso, as informações contidas no arquivo XML do instrumento virtual são recuperadas, validadas a partir do esquema de um *Virtual Instrument* (VirtualInstrument.xsd) e utilizadas para disponibilização do instrumento para o controle remoto. Na figura 3 é possível observar essas etapas do mapeamento para um osciloscópio.

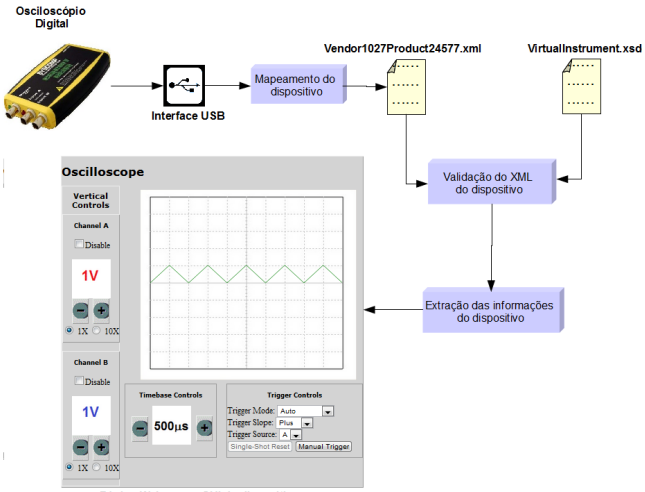

Figura 3. Exemplo de mapeamento de um osciloscópio

Com essa forma de mapeamento é possível que dispositivos heterogêneos possam ser integrados ao servidor através da descrição unificada de cada instrumento.

3.3 Camada de Descrição de Serviços

Nesta camada estão presentes as descrições dos serviços oferecidos pelo ERLab, os quais compreendem os serviços de identificação e detecção de dispositivos USB e serviços de descrição das operações e atributos específicos dos instrumentos suportados pelo sistema.

A descrição dos serviços de identificação e detecção de dispositivos é facilitada devido à disponibilização de descritores em dispositivos USB [2]. Esses descritores são padronizados e fornecem informações sobre o dispositivo como um todo ou um elemento no mesmo de acordo com a especificação USB. Na figura 4 é apresentada a hierarquia dos principais descritores USB.

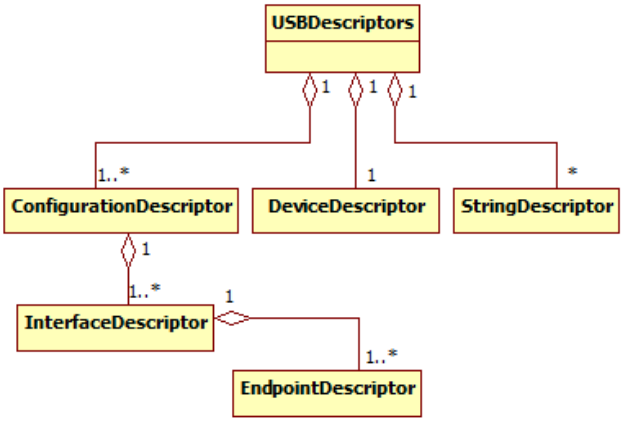

Figura 4. Diagrama de descritores USB

Conforme observado nessa figura, os descritores são agrupados em um conceito mais geral chamado *USBDescriptors*. A seguir são apresentadas, de maneira resumida, as informações que cada tipo de descritor fornece.

- *DeviceDescriptor*: Descreve o tipo de classe, subclasse, protocolo e o número de configurações do dispositivo.
- *ConfigurationDescriptor*, *InterfaceDescriptor* e *EndpointDescriptor*: Este grupo de serviços expõe os atributos relacionados à operação do dispositivo e suas funcionalidades.
- *StringDescriptor*: É utilizado para acesso às strings adicionais suportadas pelo dispositivo.

Os descritores USB fornecem as informações como identificadores, ou seja, as informações são codificadas em *bytes* conforme a atribuição feita pela usb.org. Sendo assim, com o intuito de decodificar essas informações e apresentá-las de forma legível, foi criado um arquivo XML que mapeia esses identificadores aos seus respectivos nomes. Esse arquivo XML armazena os nomes correspondentes aos identificadores dos vendedores, dos tipos de produto, das classes, das subclasses e dos tipos de protocolo de dispositivo. As informações contidas nesse arquivo são obtidas e atualizadas a partir do repositório de identificadores USB (www.linux-usb.org/usb-ids.html).

O serviço de detecção de dispositivos utiliza a tecnologia *plug-and-play*. Através dela, é possível detectar os eventos de inserção e remoção de instrumentos ao servidor. Sendo assim, é possível saber quais são os dispositivos atualmente conectados ao servidor ou aqueles que geraram eventos de conexão/desconexão.

A segunda classe de serviço descreve de forma genérica os dispositivos que são suportados pelo ERLab. Inicialmente, são utilizados os três principais instrumentos que devem estar disponíveis em um laboratório de eletrônica: osciloscópio, gerador de função e placa de Entrada/Saída. Para cada instrumento, são identificados os serviços candidatos e definidos seus requisitos.

3.4 Camada de Comunicação de Serviços

Essa camada efetua a comunicação entre os aplicativos e a camada de descrição de serviços. O ERLab considera dois tipos de fontes de dados: baseado em evento e baseado em *polling*. Para fontes de dados baseadas em evento, esta camada suporta comunicação assíncrona usando o paradigma *publish/subscribe*. Nesta abordagem, um componente que gera eventos (produtor) publica os tipos de eventos que estarão disponíveis para outros componentes (consumidores) [5]. O consumidor interessado em um evento determinado “assina” para este evento, recebendo a partir deste momento notificações sobre o evento assinado. Essas

notificações são enviadas assincronicamente dos produtores para todos os assinantes interessados.

Ela executa as funções de coletar mensagens dos produtores, filtrar e transformar tais mensagens, quando necessário, além de roteá-las para os consumidores apropriados. Para fontes de dados baseado em *polling*, esta camada periodicamente consulta a fonte para novos dados.

3.5 Camada de Aplicação

Quando a comunicação com o dispositivo é obtida, se torna necessário manipulá-lo para que o experimento desejado seja realizado. Através de interfaces gráficas amigáveis ao usuário (GUI), são disponibilizados componentes que modelam o manuseio do equipamento, com botões e caixas de texto onde determinados valores podem ser modificados e visualizados, bem como gráficos trazendo formas de onda, no caso particular de um osciloscópio digital.

4. IMPLEMENTAÇÃO

A implementação do *middleware* ERLab segue o paradigma orientado a objetos. Este é constituído de um núcleo desenvolvido em Java em conjunto com códigos em C para a realização de tarefas específicas. Este núcleo utiliza a API Java libusb/libusb-win32 [10] para efetuar a comunicação USB com o dispositivo. A linguagem de programação Java foi escolhida para a codificação, a fim de extrair o máximo possível dos seus recursos e das diversas bibliotecas existentes para a linguagem [6], a exemplo da API RXTX [12] necessária para a comunicação do sistema com a porta serial (RS232) e da API Java libusb/libusbwin32 [11] para viabilizar a comunicação com a interface USB.

Além do desenvolvimento de classes voltadas para a comunicação com o *hardware*, o ERLab faz uso de arquivos XML para o reconhecimento de dispositivos. Os arquivos XML de reconhecimento armazenam os dados sobre um conjunto de dispositivos.

5. RESULTADOS

5.1 Avaliação Qualitativa

Um dos objetivos da avaliação qualitativa é suportar a viabilidade do modelo do ERLab de modo que este forneça abstrações significativas aos usuários e programadores de aplicação, o que facilita a configuração de experimentos eletrônicos. Validar essa tese é uma tarefa complexa, pois não há parâmetros quantitativos para que se possa calcular quando da execução de um conjunto de experimentos. Em vez disso, foi desenvolvido um cliente que consome os serviços atualmente implementados no núcleo do ERLab.

Esse cliente consome os serviços de detecção e identificação de dispositivos, exibindo as informações via navegador para o usuário. Quando a página de relação de dispositivos é acessada, o cliente exibe a lista de dispositivos conectados ao servidor, onde também é possível acessar os detalhes desses por meio de botões da GUI.

Na figura 5 visualiza-se uma listagem de todos os dispositivos conectados ao servidor. Para cada dispositivo pode-se solicitar a visualização dos seus respectivos detalhes. Nessa figura é possível observar os detalhes de um osciloscópio digital que simula uma comunicação serial com o computador usando USB. Com o intuito de obter uma melhor visualização, as informações do dispositivo foram suprimidas, entretanto, há a possibilidade de apresentação de mais informações contidas nos descritores USB.

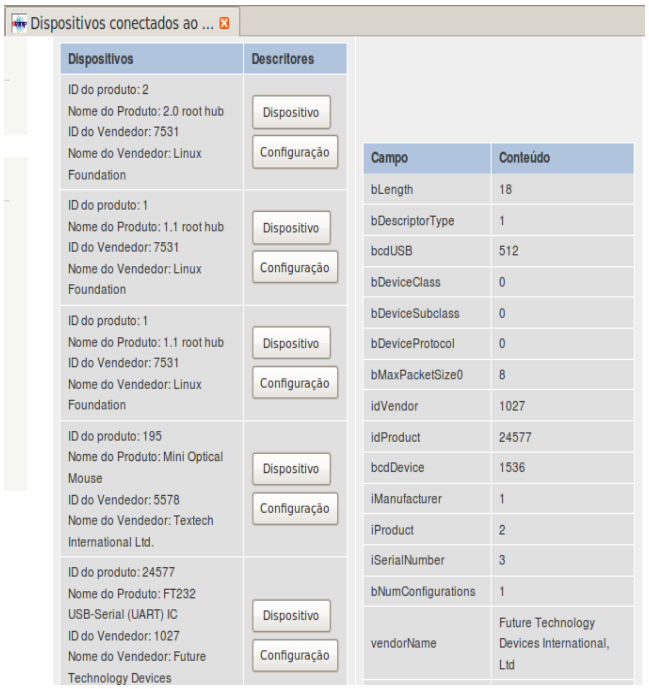

Figura 5. Aplicação cliente que faz uso *middleware* ERLab

Por fim, o cliente disponibiliza um histórico de conexão e desconexão dos dispositivos. Na figura 6 é possível observar um histórico de dispositivos sendo inseridos e removidos do servidor.

Com esse cliente, observa-se que todos os dispositivos USB podem ser reconhecidos e suas informações são exibidas.

A aplicação cliente foi desenvolvida usando o gerenciador de conteúdo Drupal [7]. Além disso, também é utilizado PHP (Hypertext Preprocessor) [15], JavaScript e AJAX (*Asynchronous JavaScript and XML*) [11] para tornar as páginas Web mais interativas com o usuário e realizar solicitações assíncronas de informações.

Figura 6. Cliente consumindo o serviço de descoberta de novos dispositivos na rede

5.2 Exemplo de utilização do ERLab

O ERLab será utilizado por laboratórios de acesso remoto visando disponibilizar experimentos que façam uso de dispositivos eletrônicos, tais como osciloscópios, geradores de função, multímetros, entre outros. Em algumas disciplinas de introdução à instrumentação eletrônica, por exemplo, cabe ao aluno analisar o comportamento de elementos presentes em circuitos simples através de atividades práticas.

O docente pode disponibilizar alguns circuitos, requisitando ao aluno que realize alguns cálculos para que seja possível conhecer o comportamento de um determinado elemento, que pode vir a sofrer ou realizar modificações em outros elementos do sistema. Fazendo uso dos instrumentos adequados, os alunos podem conferir as respostas encontradas teoricamente com aquelas obtidas de maneira prática, através de um experimento realizado remotamente.

Para que seja possível realizar esta atividade, é necessário que os usuários cadastrem-se no *site* que oferece os serviços disponibilizados pelo ERLab. É necessário ainda que haja reserva dos recursos que serão utilizados, isto é, reservam-se os dispositivos que irão compor o experimento.

Na figura 7 é mostrado o protótipo de tela de reserva de dispositivos. Nela o usuário informa o tempo em que pretende ficar utilizando o instrumento eletrônico, a data e o horário de realização do experimento, e, por fim, lhe é disponibilizada uma listagem dos dispositivos de modo que ele selecione aquele que será utilizado. Caso o horário escolhido para posse do dispositivo já tenha sido reservado, uma mensagem será exibida e será requisitado que o usuário informe um novo horário para reserva do instrumento.

Figura 7. Protótipo de formulário para reserva de horário de utilização do dispositivo

Na figura 8 é possível ver um exemplo de atividade presente em laboratórios de engenharia eletrônica. Essa atividade é realizada através de um circuito eletrônico em que se pede a tensão no capacitor C2, após serem efetuados, respectivamente, a abertura e o fechamento das chaves à esquerda e à direita do capacitor C1. Essas chaves são digitais, e podem ser controladas pelos discentes remotamente através do acionamento das saídas digitais de uma placa de I/O (*Input/Output*).

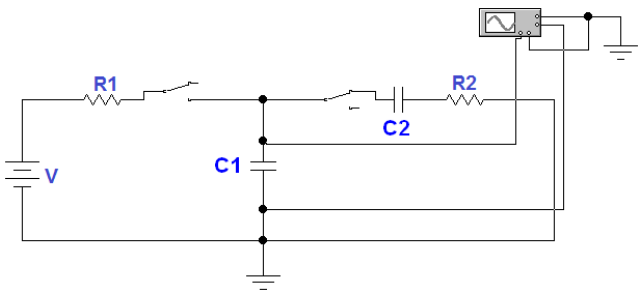

Figura 8. Circuito Resistivo-Capacitivo chaveado utilizado para se conhecer o comportamento de um capacitor

Neste experimento, em especial, utiliza-se apenas um canal do osciloscópio, com o outro desabilitado. Entretanto, é possível examinar o comportamento de um segundo elemento, tendo em vista que o osciloscópio possui dois canais. Outras funcionalidades como escala de tensão e de tempo podem ser modificadas durante a obtenção dos resultados.

6. CONSIDERAÇÕES FINAIS

As atividades práticas em laboratórios físicos envolvem o uso de instrumentos eletrônicos para a visualização de eventos a serem estudados e obtidos seus resultados. Entretanto, a realização dessas atividades é limitada ao período de funcionamento das instituições de ensino aos quais esses laboratórios pertencem. Sendo assim, os laboratórios de acesso remoto podem otimizar o uso dos recursos, pois eles ficam disponíveis por um grande período de tempo e podem ser acessados por várias instituições de pesquisas. Entretanto, o desenvolvimento dessa modalidade de laboratórios possui alguns empecilhos relacionados à dificuldade em projetá-los com a capacidade de utilizar recursos computacionais e instrumentos eletrônicos em múltiplas plataformas de *hardware* e *software*. Dessa forma, o *middleware* ERLab, apresentado neste artigo, objetiva superar essas limitações.

O ERLab ainda está em fase de desenvolvimento e implantação. Do modelo apresentado já foram concluídas a configuração da camada física, a camada de descrição de serviços e parcialmente as camadas de aplicação e de mapeamento de dispositivos. Na camada de comunicação de serviço, é utilizado um protocolo simples de consulta/resposta.

Nos próximos passos, o ERLab será utilizado para a realização de experimentos remotos *online* a exemplo do experimento apresentado na seção 6.2 deste artigo. Esses experimentos serão adicionados à página do ERLab uma vez que os instrumentos utilizados por eles estejam totalmente disponíveis para operação de maneira remota. Além disso, pretende-se disponibilizar uma API do ERLab em Java para que os programadores possam definir novos serviços.

É importante frisar que a interoperabilidade é fornecida através de serviços utilizando a tecnologia *Web Services* e a facilidade de *plug-and-play* é contemplada através da API Java libusb/libusb-win32.

Por meio da avaliação qualitativa foi possível observar a viabilidade do modelo proposto pelo ERLab através da implementação de um ambiente que faz uso do mesmo como mediador da comunicação entre a aplicação e os instrumentos.

7. REFERÊNCIAS

[1] Andria, G. et al. (2007) *Remote Didactic Laboratory 'G. Savastano', The Italian Experience for E-Learning at the Technical Universities in the Field of Electrical and Electronic Measurements: Overview on Didactic Experiments*. IEEE transactions on instrumentation and measurement, VOL 56. No. 4, August.

[2] AXELSON, Jan. *Usb Complete: The Developer's Guide*. 4. ed. Lakeview Research, 2009. 504 p.

[3] Çimen, H.; Yabanova, D.; Çina, S. M. (2008) *Web Based Remote Access Microcontroller Laboratory*. World Academy of Science, Engineering and Technology 44.

[4] Chella, M. T. e Ferreira , E. C. (2007). Architecture for a Remote Access Laboratory with Application to Electronics. In: *Webist 2007 - Web Information Systems and Technologie*s, Barcelona

[5] Couloris, G. *et al.* (2007). *Distributed systems: concepts and design.* Harlow:Addison-Wesley.

[6] Deitel, H. M., Deitel P. J. 2006. *Java: Como Programar*. Ed. 6. Upper Saddle River, New Jersey, Prentice Hall.

[7] DRUPAL.ORG (2011). *Drupal – Open Source CMS*. http://drupal.org/. 12 de fevereiro de 2011.

[8] IVI Foundation (2011). *SCPI*. http://www.ivifoundation.org/scpi/default.aspx. 25 jan 2011.

[9] Fjeldly, T. A.; Shur, M. S.;Shen, H. and Ytterdal, T. (2000). *AIM-Lab: A System for Remote Characterization of Electronic Devices and Circuits over the Internet.* Proc. 3rd IEEE Int. Caracas Conf. on Devices, Circuits and Systems (ICCDCS-2000), Cancun, Mexico, pp. I43.1–6.

[10] Oracle. (2011), *Java libusb/libusb-win32 wrapper*. http://sourceforge.net/projects/libusbjava. 25 jan 2011.

[11] Powell, Thomas. (2008). *Ajax: The Complete Reference*. McGraw-Hill Osborne Media.

[12] RXTX.ORG (2010). *RXTX : serial and parallel I/O libraries supporting Sun's Comm API*. http://www.rxtx.org. 22 nov 2010.

[13] Schafer, T. , Seigneur, J. M. e Donelly, A. (2002) . *PEARL: A Generic Architecture for Live Experiments in a Remote Lab*. http://iet.open.ac.uk/pearl/publications/icsee03.pdf.

[14] Shen, H., Xu, Z., Dalager, B., Kristiansen, V., Strom, Ø., Shur, M. S., Fjeldly, T. A ., Lü, J. e Ytterdal, T. (2006) *Conducting Laboratory Experiments over the Internet.* IEEE transactions on education, vol. 42, no. 3, p.180-185.

[15] The PHP Group (2011). *PHP: Hypertext Preprocessor*. http://www.php.net/. 27 jul 2011.

[16] W3C. (2000)**.** *Extensible Markup Language (XML) 1.0 (Second Edition)*. W3C Recommendation. http://www.w3.org/TR/2000/REC-xml-20001006. 24 jan 2011.

[17] W3C. (2000)**.** *XML Schema*. W3C Recommendation. http://www.w3.org/XML/Schema. 24 jan 2011.

[18] Wobschall, D. (2007). *IEEE 1451 - A Universal Transducer Protocol Standard.* Autotestcon, IEEE , vol., no., pp.359-363, 17-20.

[19] Yan, Y., Liang, Y., Du, X., Saliah-Hassane, H. e Ghorbani, A. (2006). *Putting Labs Online with Web Services*. IT Pro IEEE March/April.