Uma Proposta de Arquitetura para Experimentos DiffServ em Web Labs

Lucio Agostinho[1], Adriano F. Farias[1], Luis F. Faina[1],
Eliane G. Guimarães[2], Eleri Cardozo[3], Paulo R.S.L.Coelho[3]

[1] Faculdade de Computação - FACOM - Universidade Federal de Uberlândia - UFU
38400-902 - Uberlândia - MG - CP 593
[2] Centro de Pesquisas Renato Archer
13083-970 - Campinas - SP
[3] Faculdade de Engenharia Elétrica e de Computação - FEEC
Universidade Estadual de Campinas - UNICAMP

Abstract. The implementation of DiffServ management resources is necessary for creating point-to-point services with QoS between different domains. A DiffServ network needs to offer consistency of services among and inside domains and the Bandwidth Broker implementation has high value. This article introduces the architecture used to set up a Bandwidth Broker experiment in Web Labs using Web Services, but other experiments can be carried out using the same architecture. Consequently, it is possible to perform a composition of services and turn experiments scalable, with safe access, portable and with fewer restrictions over firewalls rules between domains.

Keywords: Bandwidth Broker, DiffServ, Web Services, Web Labs.

1 Introdução

O uso da Internet como meio de comunicação colaborativo permite a utilização de laboratórios de experimentação remota (Web Labs) como novas ferramentas de ensino e pesquisa. Um Web Lab é um laboratório remoto controlado através da Internet [1]. Através dele é possível realizar experimentos remotos que possuem maiores exigências quanto à configuração do ambiente de testes. Essa característica também exige um estudo mais cuidadoso da arquitetura de comunicação utilizada de forma que a experimentação remota não seja prejudicada em função da qualidade do enlace ou da tecnologia de interação entre o usuário e o laboratório. Para os laboratórios de instrumentação remota [2] o desempenho da comunicação entre a aplicação do usuário, o servidor de experimentos e o software de manipulação do recurso físico deve ser considerado importante para o sucesso da experimentação.

A convergência de tecnologias de comunicação e computação tem propiciado o surgimento de novos tipos de aplicações e, conseqüentemente, exigido o desenvolvimento de suportes e novas técnicas que dêem sustentação a essas aplicações [3]. Os Web Labs aparecem como um novo tipo de aplicação que depende da qualidade dos recursos de rede envolvidos. Além disso, a qualidade da

comunicação entre os elementos do sistema do Web Lab é fundamental para a realização eficiente dos experimentos que utilizam recursos físicos.

As aplicações de tempo-real que utilizam a Internet como meio de integração de serviços de telefonia fixa, difusão de fluxos de áudio e vídeo, laboratórios virtuais, teleconferência e outros mais, exigem um nível maior de qualidade de envio e recepção de informações se comparado com aplicações convencionais. Ainda que a interligação de redes com TCP/IP permita a entrega de pacotes sem duplicações, perdas e erros, não há garantias quanto à vazão, variação do atraso de envio e recepção (jitter), entre outros. As aplicações multimídia (fluxos contínuos de áudio e/ou vídeo) exigem essas garantias de recursos da rede para um adequado funcionamento e várias propostas de tecnologias foram sugeridas pelo IETF (*The Internet Engineering Task Force*) para oferecer Qualidade de Serviço (*Quality of Service - QoS*) na Internet. Dentre elas destacam-se: IntServ/RVSP (*Integrated Services* [7, 8]), MPLS (*Multi Protocol Label Switching* [10]) e *Diffserv* [12]), mas a tecnologia *DiffServ* se destacou por promover maior escalabilidade em relação às demais.

Diante disso, o artigo propõe uma arquitetura de software para o desenvolvimento de experimentos remotos em Web Labs que contemplam a experimentação na área de Serviços Diferenciados. Essa arquitetura utiliza *Web Services* entre a aplicação do usuário e o laboratório priorizando o desempenho da comunicação, escalabilidade, segurança no tráfego de informações entre *firewalls* e independência de plataforma. Como estudo de caso são apresentadas a implementação de um *Bandwidth Broker* junto a um experimento *DiffServ* e a implementação de um experimento que utiliza composição de *Web Services*. Ambos seguem a arquitetura proposta para o desenvolvimento de experimentos junto ao Web Lab.

O artigo é estruturado da seguinte forma: A Seção 2 apresenta um modelo de referência para Web Labs segundo a Arquitetura Orientada a Serviços. A arquitetura do laboratório remoto de redes NetLab Web Lab é descrita na Seção 3. A arquitetura utilizada para implementar o experimento *Bandwidth Broker* é explicada na Seção 4. Finalmente, a Seção 5 faz as considerações finais do artigo.

2 Modelo de Referência para Web Labs

A Arquitetura Orientada a Serviços para construção de Web Labs define um modelo de referência para Web Labs e uma família de serviços para suportar os elementos do modelo. O modelo de referência para Web Labs é mostrado na Fig. 1. O diagrama UML mostra os principais elementos do modelo, assim como os relacionamentos entre eles. Os dois elementos centrais são o Participante e o Web Lab. Um participante pode ser um usuário individual ou um grupo. Um grupo é uma coleção de participantes formado por usuários ou grupos. A linha conectando os participantes ao Web Lab representa o relacionamento de uso.

Para utilizar um Web Lab um participante deve possuir credenciais e estabelecer uma ou mais sessões com o Web Lab. As credenciais e sessões referem-se a participantes específicos acessando um Web Lab específico. Exemplos de credenciais incluem a identificação do usuário, regras (estudante, instrutor, por exemplo),

permissões (uso, gerenciamento, por exemplo) e os privilégios (líder do grupo, por exemplo) [5].

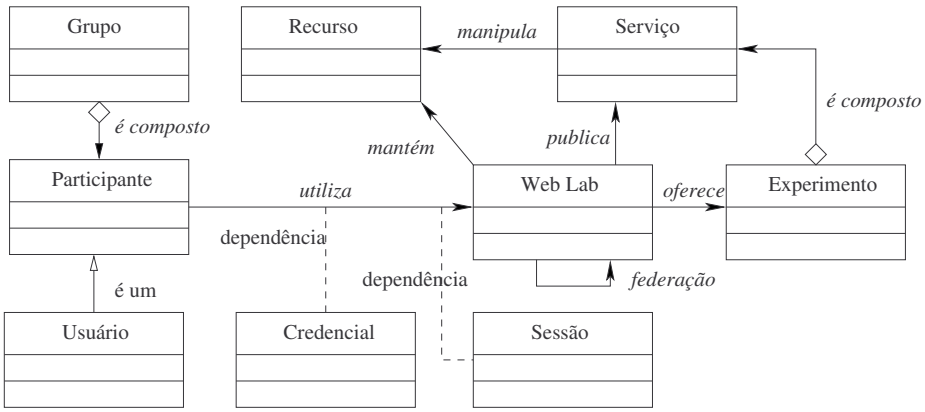

Fig. 1. Um modelo de referência para Web Labs.

Sessões são responsáveis por gerenciar a interação entre um participante e um Web Lab. Sessões mantêm um estado de informação relacionado à interação tal como a identificação do participante, o tempo de acesso restante e as ações que o participante realizou. Web Labs publicam os serviços que eles oferecem. Serviços são unidades da composição na interação com o Web Lab empregado para propósitos tais como acesso, interação, comunicação e gerência. Experimentos oferecidos pelos Web Labs mantêm um conjunto de recursos físicos tais como interfaces de rede e câmeras, e lógicos, tais como configuração de IP, rotas e assim por diante. Recursos são manipulados remotamente através de serviços. Um Web Lab tem um auto-relacionamento que modela uma federação de Web Labs [5].

3 Arquitetura do NetLab Web Lab

O NetLab Web Lab tem o objetivo de promover o ensino de redes de computadores através de um laboratório operado remotamente. Para a avaliação desta proposta foi implementado um laboratório para a configuração de uma rede de serviços diferenciados. Atualmente, a garantia de recursos para aplicações que exigem um fluxo contínuo de informações é crucial e a oferta de serviços diferenciados permite priorizar diferentes tipos de tráfego de dados. Mas essa configuração exige esforço para dispôr a rede física que suporte as complexas configurações a serem realizadas.

A infraestrutura de software do laboratório é composta por um Web *container* Apache Tomcat com serviços Web Axis2/Java, um banco de dados relacional MySQL e utiliza a tecnologia Java RMI para realizar a comunicação entre a rede interna do Web Lab e o container de *Web Services*. A Fig. 2 ilustra o ambiente onde os experimentos do domínio *DiffServ* são realizados. O host HODES mantém o Web Server Tomcat no laboratório e o *container* para Web Services Axis2. Os hosts

HELIOS, GAIA, POSEIDON, URANO e ZEUS e suas respectivas interfaces de rede são os recursos físicos do laboratório. Cada um desses hosts possui interfaces de rede Ethernet 10/100/1000Mbps e estão conectados a um Switch Gigabit Ethernet. Eles possuem instalado o sistema operacional Slackware 10.2 e kernel 2.6.15.6 com suporte a *DiffServ*. Os hosts do laboratório também estão conectados a uma rede de retaguarda através de um hub 10/100Mbps. Essa rede garante a conectividade entre os hosts e é através dela que é realizado o início da configuração e a finalização consistente dos experimentos da rede. A interface de rede para a rede de retaguarda não está disponível para manipulação pelos experimentos.

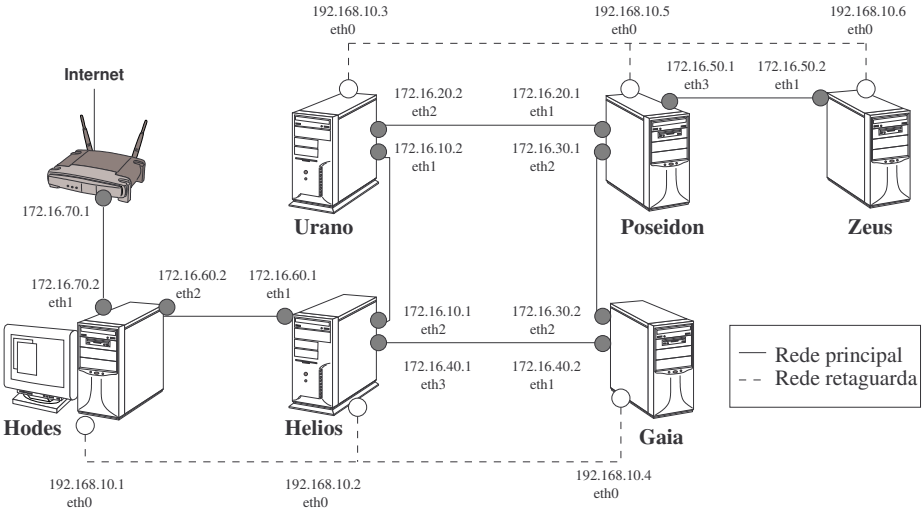

Fig. 2. Arquitetura do NetLab Web Lab.

3.1 Serviços Diferenciados

O NetLab Web Lab foi implementado para respeitar a arquitetura de Serviços Diferenciados (*DiffServ*) e permitir o estudo dos diferentes comportamentos de uma rede física submetida a essas configurações. O interesse pelo estudo *DiffServ* surgiu em muitos dos esforços do IETF para especificar mecanismos para fornecer QoS em redes IP. Inicialmente esses estudos se voltaram para o modelo de RSVP IntServ que é fortemente ligado ao IP *multicast* e requer que cada elemento no caminho entre a origem e o destino forneçam QoS para cada fluxo da rede. Embora grande esforço tenha sido feito sobre esse modelo ele foi considerado inadequado quanto à escalabilidade de seu uso no ambiente Internet [6]. A proposta *DiffServ* busca realizar o tratamento de microfluxos que possuem necessidades semelhantes de QoS, agrupando o tratamento deles em classes de comportamento. Essa medida torna a proposta escalável o suficiente para ser utilizada em grandes redes e ainda permitir que uma variedade de aplicações tenha a garantia de recebimento de recursos de rede fim-a-fim entre domínios administrados separadamente.

A arquitetura *DiffServ* baseia-se no tratamento diferenciado do encaminhamento de pacotes IP que recebem uma marcação ao entrar no domínio. A marcação é a atribuição do código DSCP (*DiffServ Code Point*) ao campo *DS Field* (campo ToS para IPv4 ou campo *Traffic Class* para IPv6) do cabeçalho do pacote IP [9, 11]. Já o tratamento desses pacotes é conhecido como *Per Hop Behavior* (PHB). Os hosts do domínio implementam esses PHB através de uma variedade de mecanismos e disciplinas de fila.

A proposta de se utilizar Serviços Diferenciados em uma rede de computadores possibilita fornecer QoS para as aplicações. Em redes de computadores isso diz respeito à garantia de que certos recursos estarão disponíveis quando o usuário de uma determinada aplicação precisar deles. Em adição ao encaminhamento de pacotes, a arquitetura *DiffServ* requer dispositivos de borda que incluem componentes de condicionamento de tráfego que sejam capazes de classificar, marcar, modelar e descartar pacotes assim que eles entram e saem de um domínio *DiffServ* [6].

3.2 Proposta de Implementação do *Bandwidth Broker*

Para automatizar o processo de negociação de acordos entre domínios, o controle de admissão de fluxos e a gerência das configurações de QoS nos dispositivos do domínio, foi sugerido o uso do componente *Bandwidth Broker* (BB) na arquitetura *DiffServ* sugerida pelo IETF [13]. O componente BB é o agente centralizador que realiza o controle da alocação de largura de banda, mantém marcadores da alocação atual do tráfego sinalizado e recebe novas requisições para marcar o tráfego de acordo com as políticas estabelecidas, respeitando as alocações de banda já realizadas. O BB tem duas responsabilidades: a) parcelar as alocações e configurar os roteadores de borda do domínio; b) gerenciar as mensagens que são enviadas através dos limites das regiões adjascentes [13].

O experimento BB permite criar a configuracao *DiffServ* conforme mostra a Fig. 3. Podem ser criadas disciplinas de fila, classes e filtros através de um ambiente gráfico em que o usuário informa apenas os parâmetros de cada comando TC do pacote IPROUTE2. O aplicativo permite acompanhar a estrutura da configuração *DiffServ* através de uma árvore porque os comandos *DiffServ* são dependentes uns dos outros e devem ser inseridos respeitando a hierarquia de formação de cada PHB. Os parâmetros *DiffServ* são enviados pelo aplicativo BB do usuário para o host da rede interna do laboratório. Cada host possui um objeto servidor capaz de interpretar esses parâmetros, montar o comando TC e executá-lo. O trecho que ser mostrado a seguir exibe os comandos para controle do tráfego de pacotes IP gerados pelo objeto servidor Java no host do Web Lab, de acordo com a configuração da Fig. 3.

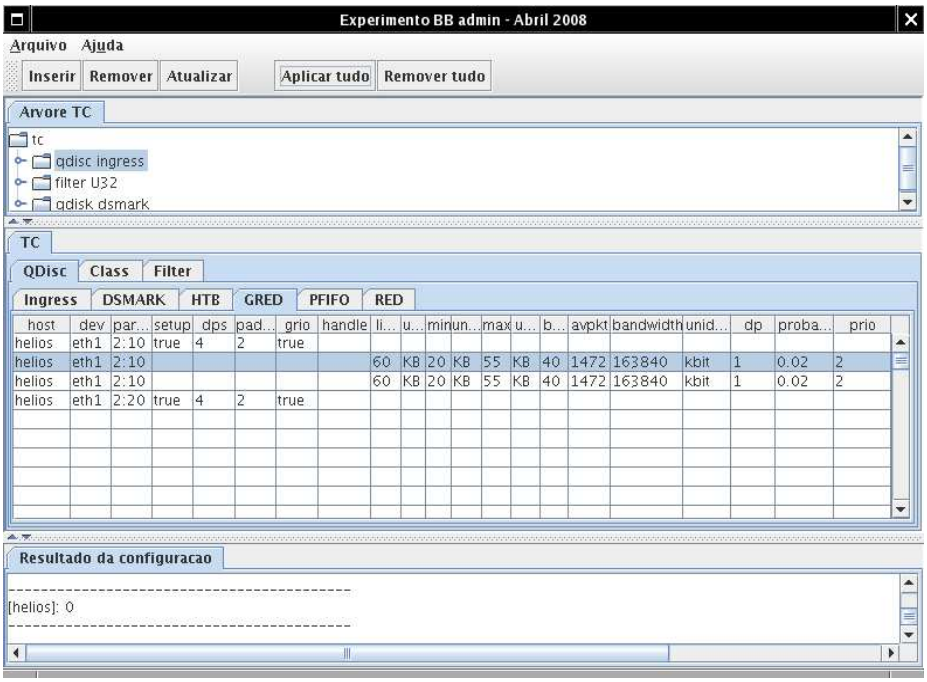

Fig. 3. Experimento BB no NetLab Web Lab.

3.3 Aspectos de Implementação do *Bandwidth Broker*

O experimento BB promove a gerência de recursos *DiffServ* no domínio do NetLab Web Lab. Foram utilizados *Web Services* (ou Serviços Web) como mediadores da comunicação entre o aplicativo BB do usuário e a rede interna do laboratório. *Web Services* são aplicações XML mapeadas para programas, objetos, base de dados ou funções de negócio compreensíveis [4].

Os *Web Services* não estão sujeitos às políticas convencionais de segurança de sistemas distribuídos tais como CORBA, RMI e DCOM, por exemplo. Isso simplifica o processo de encaminhamento de mensagens entre *firewalls* de domínios distintos uma vez que são trocadas mensagens XML utilizando o protocolo SOAP (*Simple Object Access Protocol*) entre as aplicações remotamente distribuídas.

Quando o participante deseja acessar o experimento ele deve fornecer as suas credenciais (nome de usuário e senha) para que o site do laboratório realize a autenticação do usuário. Antes de iniciar o *download* da aplicação, o Web Lab instancia os objetos servidores nos hosts alocados para o experimento. Esses objetos são instanciados através da fábrica de objetos que foi inicializada no processo de boot de cada um dos hosts do Web Lab.

O aplicativo é então *downloaded* automaticamente na máquina do usuário através da tecnologia Java Web Start (JWS). Esse aplicativo do experimento conhece apenas as interfaces dos métodos disponibilizados pelo *Web Service* BB no *Web Server* do

laboratório. Quando uma requisição de configuração de disciplina de fila é solicitada, o aplicativo cliente informa os parâmetros da configuração e o host do laboratório onde a alteração deve ser realizada através de uma mensagem SOAP. O *Web Service* no servidor do laboratório recebe a requisição e a encaminha para o host correspondente. O processo de automatização promovido pelo BB garante que as alterações em um host possam ser replicadas em todos os demais, e esse esforço de tratamento diminui com a visualização gráfica das configurações das disciplinas de fila. Cabe ao usuário definir os hosts de borda e núcleo e realizar a configuração *DiffServ* em cada um deles com o auxílio da ferramenta gráfica do aplicativo BB.

4 Arquitetura do Experimento *Bandwidth Broker*

A arquitetura para implementar o experimento *Bandwidth Broker* no NetLab Web Lab é ilustrada na Fig.4. Antes de iniciar um experimento a aplicação Web do site do NetLabWeb Lab solicita a instanciação do objeto servidor TC (*Traffic Control*) em cada um dos hosts do experimento. Esse objeto servidor irá realizar as configurações *DiffServ* que forem solicitadas pelo usuário do experimento. O site comporta-se como um cliente SOAP e solicita a instância do objeto servidor TC. O *Web Service* FabricaRMI encaminha a solicitação para o objeto servidor FabricaRMI de cada um dos hosts do experimento.

Os *Web Services* comportam-se como clientes RMI quando acessam diretamente os métodos dos objetos servidores RMI nos hosts do laboratório e como servidores SOAP quando recebem requisições das aplicações remotas (site do NetLab e aplicação JWS do experimento). Os objetos servidores são instanciados a partir de uma fábrica de objetos. Essa fábrica é um objeto servidor que é registrado no *daemon* rmiregistry de cada host. A fábrica pode criar dinamicamente um ou mais objetos servidores de acordo com a especificidade do experimento. O mesmo objeto servidor encontra-se no host servidor do laboratório para armazenar em uma base de dados as configurações *DiffServ*. Isso é necessário para respeitar a especificação *DiffServ* que confere ao BB uma base de dados para armazenar as configurações do domínio. O acesso às configurações do domínio é realizado por meio de consultas à base de dados do BB. O trecho de código a seguir ilustra parte da implementação em Java do *Web S* BB que atua como servidor SOAP e cliente RMI.

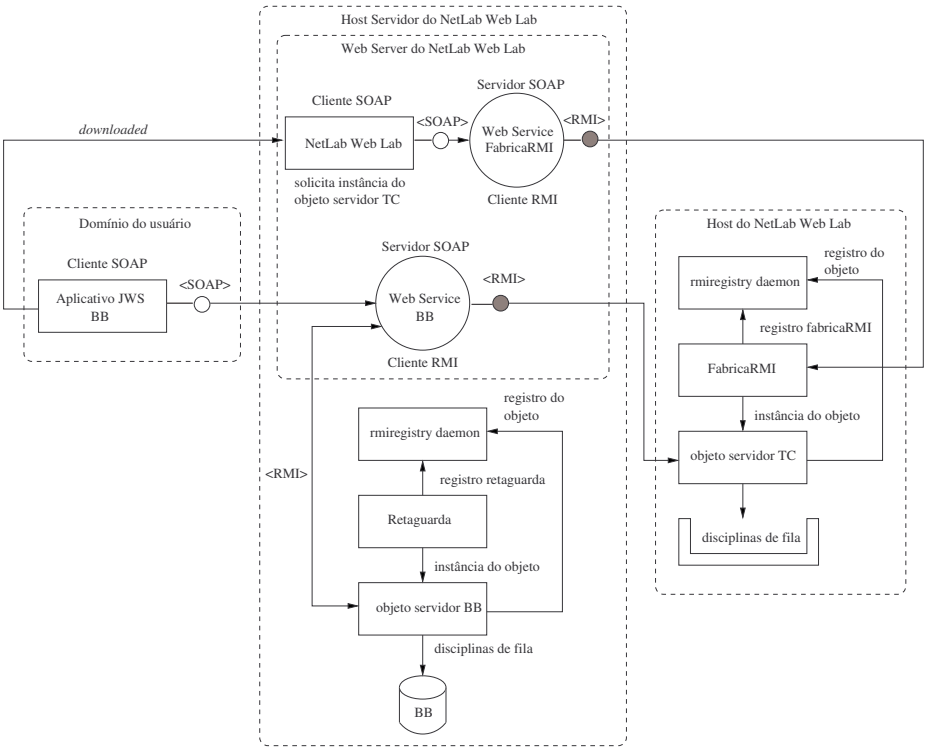

Fig. 4. Arquitetura do Experimento BB no NetLab Web Lab.

```
public void adquirirReferenciaRemota(String host){
  String caminhoServico = "rmi://" + host + "/" +
                          NOME_SERVICO;
  servicoTC = (IServicoTC) Naming.lookup(caminhoServico);
}//fim adquirirReferenciaRemota

public String addQDiscGRED(String hostLab,
                           String dev, String parent,
                           String setup, String dps,
                           String padrao, String grio){
   adquirirReferenciaRemota(hostLab);
   resultado = servicoTC.addQDiscGRED(dev, parent, setup,
                                      dps, padrao, grio);
   return resultado;
}//fim addQDiscGRED
```

4.1 Aspectos de Implementação do *Bandwidth Broker*

A arquitetura da Fig. 4 foi utilizada em outros experimentos do NetLab Web Lab: configuração de NIC, tratamento de rotas, teste de conexão entre hosts e submissão de

fluxos reais e virtuais. Isso demonstra a praticidade de se seguir a arquitetura para disponibilizar quaisquer tipos de experimentos. Cada *Web Service* está associado a um ou mais objetos servidores responsáveis pela implementação das requisições do aplicativo do experimento.

Por isso, novos experimentos podem ser criados através da composição de serviços, como mostra a Fig. 5. O aplicativo de um experimento precisa apenas possuir interfaces para acesso aos outros *Web Services* do laboratório. No entanto, o aplicativo e o *Web Service* do experimento não possuem a lógica para a realização das configurações: essa lógica é implementada apenas pelo objeto servidor instanciado no host do laboratório.

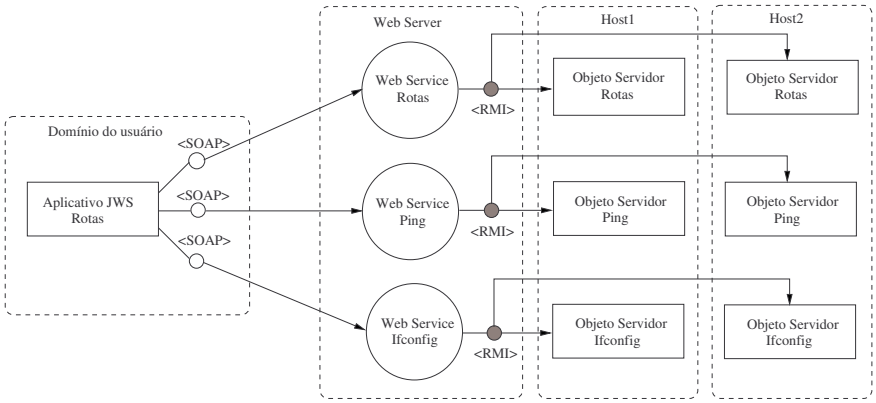

Fig. 5. Experimento Rotas com composição de serviços no NetLab Web Lab.

O acesso aos aplicativos fora do domínio doWeb Lab é realizado independente do sistema operacional graças à tecnologia JWS e não são necessários tratamentos alternativos das informações transmitidas porque o formato das mensagens segue o padrão SOAP. Dentro do domínio, a *performance* dos experimentos é otimizada porque os *Web Services* possuem apenas as referências para os hosts do Web Lab e para os respectivos métodos nesses hosts que executam a ação solicitada pelo usuário. Os *Web Services* fazem então o acesso direto aos métodos dos objetos servidores Java com o uso de RMI. Esses objetos realizam as alterações necessárias nos hosts e devolvem o resultado para o *Web Service* que, por sua vez, apenas encaminha a resposta para o aplicativo JWS do usuário.

5 Considerações Finais

Este artigo apresenta uma proposta de implementação de um experimento *Bandwidth Broker* segundo a arquitetura *DiffServ*. Essa implementação respeita uma arquitetura que utiliza *Web Services* e objetos servidores instanciados a partir de fábricas de objetos RMI. Nesta arquitetura, os blocos elementares são serviços (*Web Services*) que, recursivamente, podem ser combinados na construção de experimentos mais complexos, mas a orquestração desses serviços está além do escopo desse artigo.

Cada recurso do laboratório (físico ou lógico) é modelado e implementado para interagir com *Web Services* e a arquitetura apresentada permite que os experimentos oferecidos pelo NetLab Web Lab sejam construídos por meio da composição destes serviços.

A arquitetura para a implementação de experimentos apresenta as vantagens de ser independente de plataforma, possuir um formato de mensagens padronizado (SOAP) para a transferência de informações entre o domínio do usuário e o Web Lab, escalável o suficiente para representar e configurar os hosts do laboratório e permitir a inserção de novos recursos físicos, com a segurança do acesso garantida com o uso de credenciais mantidas pelo Web Lab.

Referências

1. García-Zubia, J., López-de-Ipiña, D., Orduña, P.: Remote Control of Web 2.0 enabled laboratories from Mobile Devices. In: Conference on E-Science and Grid Computing. Proceedings of the Second IEEE International. IEEE(2006)
2. García-Zubia, J., López-de-Ipiña, D., Orduña, P.: An Approach for Web Labs Analysis. In: International Journal of Online Engineering, Vol.3, No.2. iJOE (2007)
3. Montez, C.: Implementando Diferenciação de Serviços em Servidores Web Através de Escalonamento Adaptativo. In: 20o. Simpósio Brasileiro de Redes de Computadores. SBRC (2002)
4. Pulier, E., Taylor, H.: Compreendendo SOA Corporativa. Editora Ciência Moderna, pp. 57. (2008)
5. Coelho, P.R.S.L., Sassi, R.F., Cardozo, E., Guimarães, E.G., Faina, L.F., Lima, A.Z., Pinto, R.P.: A Web Lab for Mobile Robotics Education. 2007 IEEE International Conference on Robotics and Automation, pp. 1387-1386. ICRA (2007)
6. Teitelbaum, B., Hares, S., Dunn L., Neilson, R., Narayan, V., Reichmeyer, F.: Internet2 QBone: building a testbed for differentiated services. IEEE Network, Vol.13, No.5, pp.8-16. IEEE Network (1999)
7. Braden, R., Clack, D., Shenker S.: Integrated Services in the Internet Architecture: an Overview. RFC1633, Internet Engineering Task Force (1994)
8. Wroclawski, J.: The Use of RSVP with IETF Integrated Services. RFC2210, Internet Engineering Task Force (1997)
9. Nichols, K., Blanke, S., Baker, F., Black, D.: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC2474, Internet Engineering Task Force (1998)
10. Rosen, E., Viswanathan, A., Callon, R.: Multiprotocol Label Switching Architecture. RFC3031, Internet Engineering Task Force (2001)
11. Grossman, D.: New Terminology and Clarification for DiffServ. RFC3260, Internet Engineering Task Force (2002)
12. Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.: An architecture for Differentiated Services. RFC2475, Internet Engineering Task Force (1998)
13. Nichols, K., Jacobson, V., Zhang, L.: A Two-bit Differentiated Services Architecture for the Internet. RFC2638, Internet Engineering Task Force (1999)