Um Sistema de Gerenciamento de Redes Baseado em Mashups

Rafael Santos Bezerra, Carlos Raniery Paula dos Santos, Leandro Márcio Bertholdo, Lisandro Zambenedetti Granville, Liane Rockenbach Tarouco

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre, RS – Brazil

{rafaelsbz, crpsantos, granville}@inf.ufrgs.br
{bertholdo, liane}@penta.ufrgs.br

Abstract. *Mashups are a new breed of Web applications, created from the integration of external resources available in the Web. Recently, they have been considered a hallmark of Web 2.0 technologies, because they place the end user on a developer role and incentivate the collaboration and reuse of existing resources. Following the path of increasing efforts in research about new approaches to network management, mashups present themselves as technology that potentially has many advantages to offer to the field. However, to this date, there was no investigation on the usage of mashups in network management. Therefore, this paper approaches this technology, proposing a Mashup Development Tool geared to network management. We discuss both the architecture of such system and a proof of concept prototype. Such prototype consists on a Mashup Development Tool geared to integrate information of Autonomous Systems routing across the internet.*

Resumo. *Mashups são aplicações criadas a partir da integração rápida e dinâmica de recursos e dados disponíveis na Web. Eles são uma das principais tecnologias da Web 2.0 e seus principais objetivos são cooperação, reusabilidade e orientação ao usuário final. Considerando a crescente demanda por novas ferramentas de gerenciamento de redes, a utilização das tecnologias da Web 2.0, em especial os mashups, aparece como uma proposta em potencial. Entretato, os autores não têm conhecimento de quaisquer outras investigações nesse sentido na área. Este artigo tem como objetivo explorar essa oportunidade de pesquisa, ou seja, avaliar a aplicação dos mashups no gerenciamento de redes. Para tal fim, é proposta a arquitetura de uma ferramenta de gerenciamento de redes baseada em mashups. Como prova de conceito para essa arquitetura, é demonstrado um sistema de criação de mashups baseado na mesma, voltado para a integração de informações de roteamento de Sistemas Autônomos.*

1. Introdução

O aumento na complexidade que as redes de computadores têm apresentado nos últimos anos vem obrigando os pesquisadores a olhar com mais atenção para a qualidade das ferramentas oferecidas aos administradores. Novas soluções de gerenciamento têm sido propostas no intuito de auxiliar corporações a administrar de forma mais adequada suas infra-estruturas de rede. O *Simple Network Management Protocol* (SNMP) [Harrington et al. 2002], definido na década de 80 e revisto nos anos 90, porém, ainda se apresenta como a solução *de facto* para o gerenciamento de redes TCP/IP. Soluções

convencionais como o SNMP não são mais suficientes para satisfazer as necessidades de gerenciamento das redes de computadores atuais [Soldatos and Alexopoulos 2007]. Por exemplo, situações onde os administradores desejam visualizar informações de forma muito específica ou agregar dados de diversas fontes heterogêneas não são facilmente suportadas pelas ferramentas atuais. Este cenário é especialmente problemático no gerenciamento de grandes *backbones* (*e.g.*, *backbone* da RNP[1]), onde as informações gerenciadas são provenientes de um grande número de sistemas e equipamentos diferentes.

Nesse contexto, existe um movimento recente de pesquisa que investiga a aplicação de tecnologias consolidadas em outros meios no gerenciamento de redes. Exemplos dessas novas tecnologias incluem *Peer-to-Peer* (P2P) [Granville et al. 2005], computação autonômica e Web Services [Vianna et al. 2007]. Em meio a elas, um conjunto de tecnologias ainda mais recente, agrupadas sob o título de Web 2.0, têm atraído atenção da indústria e da academia. O termo Web 2.0 serve para designar uma nova categoria de sistemas Web onde os usuários são incentivados a colaborar na criação e organização de recursos disponibilizados na rede, saindo do papel de consumidores para o papel de colaboradores [O'Reilly 2005]. Dentro desse contexto, uma tecnologia específica é de especial interesse para esse artigo, os *mashups*, aplicações criadas a partir da composição e reutilização de recursos da Web. As propostas atuais a respeito de *mashups* descrevem sistemas que alegam permitir que o usuário final construa suas próprias aplicações em um navegador Web através da integração dinâmica de recursos disponíveis na Internet, facilitada por semânticas de alto nível e interfaces de usuário dotadas de alta usabilidade. A principal abordagem dessas propostas é colocar o usuário no papel do desenvolvedor, incentivando a criação colaborativa e a reutilização de recursos da Web [Merril 2003]. Apesar de já existirem iniciativas tanto da indústria [Ennals and Gay 2007] quanto da academia [Liu et al. 2007] que propõem tais sistemas de criação de *mashups*, não há indícios de qualquer trabalho que verifique se as características e vantagens propostas são atingíveis de fato.

Em face das potenciais vantagens da proposta de *mashups*, bem como da ausência de trabalhos investigando a aplicação das tecnologias da Web 2.0 no gerenciamento de redes, tais tecnologias serão o foco deste trabalho. O objetivo principal deste trabalho é abordar especificamente os *mashups*, de forma a verificar se as características propostas são atingíveis de maneira viável e se elas são vantajosas no contexto do gerenciamento de redes. Para esse fim, é proposta a arquitetura de um Sistema de Gerenciamento Baseado em *Mashups*. Também é apresentado um estudo de caso que consiste no desenvolvimento, baseado na arquitetura proposta, de um sistema de criação de *mashups* com o objetivo específico de agregar informações de roteamento entre Sistemas Autônomos (*Autonomous Systems -AS*) na Internet que operam com o protocolo BGP (*Border Gateway Protocol*). As informações a serem integradas dizem respeito à troca de tráfego e a quantidades de rotas anunciadas entre diferentes AS's. Tal conjunto de relacionamentos inter-AS é conhecido como *peering BGP* e sua observação é especialmente importante, pois dela depende o bom funcionamento de um ramo da rede que compõe Internet. A rápida observação e comparação do número de rotas anunciadas por parceiros em relação à quantidade de tráfego trocada fornece indícios de diversos problemas. Por exemplo, políticas definidas em acordos de nível de serviço (*Service Level Agreement* - SLA) podem não estar sendo

[1]Rede Nacional de Ensino e Pesquisa

mantidas, ou conexões em um Ponto de Troca de Tráfego (PTT) podem estar sofrendo de malfuncionamento.

O restante deste trabalho está organizado como segue. Na Seção 2 são apresentados os principais trabalhos realizados sobre *mashups*. Na Seção 3 é apresentada a arquitetura da solução proposta neste trabalho. Com base nesta arquitetura, foi desenvolvido um sistema de criação de *mashups*, através do qual foi possível criar uma solução para o problema da obtenção rápida e dinâmica de informações de *peers* BGP. Detalhes do protótipo desenvolvido são apresentados na Seção 4 com o objetivo de validar a proposta apresentada neste trabalho. Por fim, as considerações finais e os trabalhos futuros são apresentados na Seção 5.

2. *Mashups*

Os *mashups* podem ser vistos como uma tecnologia de composição, pois seu objetivo é possibilitar a criação de novos serviços e aplicações a partir da integração de recursos obtidos de diferentes fontes [Liu et al. 2007]. Comparados às tradicionais tecnologias de composição (*e.g.*, BPEL [Alves et al. 2007]), os *mashups* propõem um maior dinamismo, através da possibilidade de um processo de composição mais amigável ao usuário e mais ágil. Além disso, a proposta de *mashups* prevê que os mesmos possam ser criados mesmo por usuários sem conhecimentos em linguagens de programação, através da utilização de abstrações da composição para semânticas de alto nível e de paradigmas de interação mais amigáveis do que linguagens de programação tradicionais.

Os recursos que são integrados em *mashups* utilizando navegadores Web. Eles podem ter diferentes origens e ser acessados utilizando diferentes protocolos. Para esse fim, é necessário um sistema de criação de *mashups* que utilize tecnologias que facilitem a criação das composições (*e.g.*, AJAX, Flash). Além disso, deve haver uma infra-estrutura que forneça suporte à criação e execução dos k. De acordo com Merril [Merril 2003], tal infraestrutura deve possuir três elementos principais, como mostra a Figura 1.

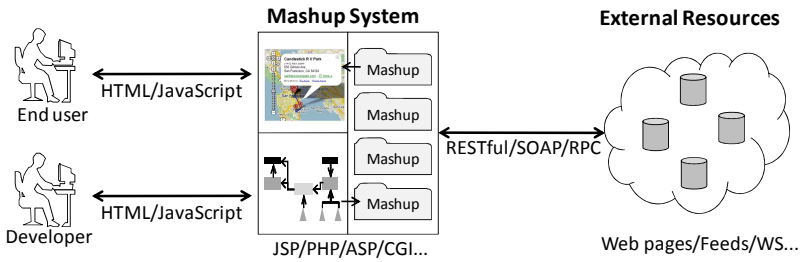

Figura 1. Principais Elementos Existentes na Criação de um *Mashup*

1. **Provedor de conteúdo**: são os recursos externos ao sistema e que são integrados pelos usuários para criar os *mashups*. O acesso a tais recursos pode ocorrer utilizando diversos métodos, por exemplo, através de SOAP, RPC ou Syndication (*e.g.*, RSS e ATOM)[Liu et al. 2005].
2. **Sistema de *Mashups***: onde é armazenada a lógica dos mashups criados, porém, não é necessariamente onde eles são executados. Este sistema pode ser criado utilizando tecnologias de desenvolvimento Web tradicionais, por exemplo, CGI, PHP. Alternativamente, o conteúdo do Mashup pode ser gerado dinamicamente

através do navegador Web do cliente, usando para isto scripts do lado do cliente como *JavaScript*.

3. **Navegador Web**: é onde o resultado das composições (*i.e.*, *mashups*) é representado graficamente. Também é através do navegador Web que estas composições são estabelecidas, através do acesso ao sistema de *mashups*.

O conceito de *mashups* é bem recente, e a distinção entre ele e os métodos tradicionais de composição de recursos pode ser difícil. Contudo, tais sistemas apresentam diferenças significativas, por exemplo, a composição de serviços tradicional exige o conhecimento das interfaces dos serviços que se deseja integrar e esta integração não necessariamente é realizada a partir de aplicações Web. Também há um aumento constante no formalismo das aplicações de composição de serviço, que são tipicamente baseadas em *Web Services* e desenvolvidas por programadores profissionais. O uso destas aplicações é especialmente difícil para usuários sem conhecimento técnico. *Mashups* por outro lado, propõem a abstração da complexidade da integração para o usuário final, permitindo que o mesmo utilize técnicas mais simples para conectar as informações e serviços. Autores como Liu *et. al.* [Liu et al. 2007], definem que as principais características dos *mashups* são: reusabilidade e foco no consumidor final.

2.1. Trabalhos Relacionados

Atualmente, existem algumas iniciativas de desenvolvimento de sistemas para criação de *mashups* desenvolvidas tanto por parte de empresas quanto por parte de universidades e centros acadêmicos. Alguns exemplos de sistemas de *mashups* comerciais são: GOOGLE MASHUP EDITOR [1] da Google, YAHOO! PIPES [2] da Yahoo! e POPFLY [3] da Microsoft. Entretanto, essas iniciativas não são voltadas para o gerenciamento de redes de computadores. Em geral, elas foram desenvolvidas para a composição de informações de páginas Web, ou então de recursos disponibilizados por ferramentas proprietárias.

Na academia, as iniciativas de pesquisa a respeito dos *mashups* incluem, por exemplo, o trabalho realizado por Banerjee *et al.*, abordando a aplicação de *mashups* no desenvolvimento de aplicações de telecomunicação [Banerjee et al. 2007]. Esse artigo apresenta uma arquitetura em várias camadas e um protótipo como prova de conceito da arquitetura. Nela, são utilizados principalmente *Web Services* como componentes, tornando a iniciativa bastante similar a tecnologias de composição de serviço.

Outro trabalho acadêmico abordando *mashups*, este de especial interesse no presente artigo, é o realizado por Yu *et. al.*, onde são abordadas as principais idéias e tendências do desenvolvimento de *mashups* [Yu et al. 2008]. Após uma exposição a respeito das principais ferramentas de desenvolvimento de *mashups* disponíveis, o artigo trata dos princípios envolvidos nesses softwares. O modelo discutido pelos autores é um dos principais referenciais para a arquitetura descrita na próxima seção.

3. Proposta

A arquitetura proposta neste trabalho, segue uma divisão em três camadas (*i.e.*, apresentação, lógica e dados). Esta divisão, também chamada de *three-tier*

[1] http://code.google.com/gme
[2] http://pipes.yahoo.com
[3] http://www.popfly.com

[Eckerson 1995], possui como principal vantagem possibilitar uma melhor estruturação dos elementos funcionais de aplicações Web. Com base nesta arquitetura, foi desenvolvido um sistema de gerenciamento que possibilita a criação de *mashups* específicos para o problema do *peering* BGP, apresentado anteriormente. Detalhes desta arquitetura são apresentados a seguir.

3.1. A Arquitetura

Na arquitetura, são definidos dois tipos de operadores humanos que possuem papéis diferentes: os administradores e os desenvolvedores. Em ambos os casos, a interação com o sistema ocorre através do navegador Web. O administrador é o operador humano interessado em utilizar os *mashups* disponíveis. O desenvolvedor por sua vez, tem como função definir as composições que formam os *mashups*. De fato, um mesmo operador pode assumir ambos papéis, porém, funcionalmente, são dois papéis distintos.

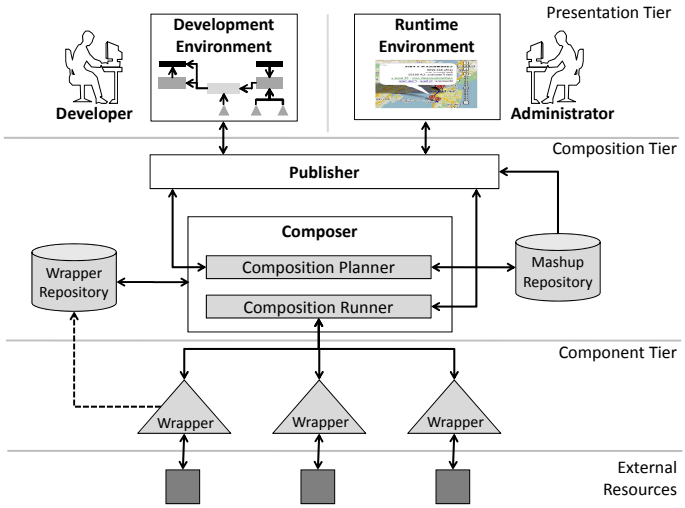

Figura 2. Arquitetura do Sistema Proposto

Os elementos apresentados na Figura 2 são divididos nas três camadas apresentadas sumarizadas a seguir:

- ***Component Tier***: onde estão localizados os *wrappers*. Os *wrappers* funcionam como *gateways* para acessar recursos externos ao sistema de gerenciamento. É através da interligação de componentes que representam estes *wrappers*, que os desenvolvedores conseguem realizar as composições;
- ***Composition Tier***: onde estão localizados os elementos responsáveis por controlar a composição dos recursos que irão resultar em um *mashup*. Nessa camada existe um módulo de composição e um de publicação. O módulo de composição, chamado *Composer*, possui como função principal possibilitar que os fluxos de integração definidos pelos desenvolvedores possam ser criados e executados. Já o módulo de publicação, *Publisher*, tem como função principal, recuperar o resultado das composições e disponibilizá-lo para os administradores;
- ***Presentation Tier***: onde ocorre a interação com o usuário final. Nessa camada, existem dois módulos principais. O *Developer Environment* (DE) é responsável

por implementar a interface gráfica que será utilizada pelo usuário interessado em desenvolver um *mashup* (*i.e.*, o desenvolvedor). A outra interface, o *Runtime Environment* (RE), é responsável por possibilitar a interação do usuário interessado apenas em acessar ou consumir determinado *mashup* (*i.e.*, administrador).

3.2. Elementos da Arquitetura

Os *wrappers* têm como objetivo possibilitar que diferentes recursos externos possam ser acessados e integrados pelo sistema. Isto ocorre mesmo que os formatos de dados e o meio de acesso a estes recursos sejam diferentes (*e.g.*, SOAP, RPC, *Syndication*). Na definição de um *wrapper*, é estabelecido um conjunto de meta-informações sobre cada um dos recursos que serão acessados. Exemplos de meta-informações incluem o tipo do recurso, parâmetros de entrada, endereço de rede, descrição e valores de saída. É com base nessas meta-informações que os desenvolvedores podem estabelecer as composições utilizando os componentes de adaptação. Durante a execução de um *mashup*, estas meta-informações são lidas a partir do repositório de *wrappers*, para então ser utilizada pelo *Composition Runner* na execução dos *wrappers*. O desenvolvimento destes *wrappers* ocorre por meio de uma API disponibilizada junto ao sistema e que possibilita que o próprio sistema controle o ciclo de vida de cada um dos *wrappers* registrados.

O principal elemento da arquitetura proposta é o *Composer*, responsável por validar e executar as composições criadas pelos desenvolvedores. Dentro do *Composer*, há o *Composition Planner* (CP), que possui a definição de todos os componentes existentes no sistema, incluindo externos (*i.e.*, adaptação) como internos (*i.e.*, visuais, operação e controle). O resultado desta composição, é uma metadescrição de um *mashup*, a qual é armazenada no *Mashup Repository* e que será acessada pelo *Publisher* quando um administrador quiser utilizar um *mashup* existente. Outro elemento interno ao *Composer* é o *Composition Runner* (CR), que possui como função orquestrar as chamadas aos *wrappers*. Ele interage diretamente com os *wrappers*, que acessam os recursos externos e retornam as informações necessárias. Essas informações serão processadas e integradas, e o resultado dessa integração será recebido pelo *Publisher*, responsável por publicá-lo.

O *Publisher* é responsável por disponibilizar os componentes e os *mashups* aos operadores humanos. Ele serve como intermediário para toda a comunicação com a camada de apresentação, a qual realiza as interações com os usuários. Através dele, um usuário pode acessar o *Composition Planner*, compor seus *mashups* e publicá-los, ou simplesmente buscar *mashups* no repositório e executá-los. Além disso, o *Publisher* permite que desenvolvedores busquem outros *mashups* publicados para serem reutilizados em seus próprios. Finalmente, o *Publisher* é responsável pela administração do sistema, controlando todas as operações que cada operador pode realizar. Assim, ele permite, por exemplo, que usuários criem seus *mashups* e definam grupos que podem executá-lo, reutilizá-lo como base para novas composições ou modificar seu próprio funcionamento.

A camada de apresentação, executada no navegador do usuário (*e.g.*, Firefox, Opera), possui dois módulos principais: o *Developer Environment* (DE) e o *Runtime Environment* (RE). O DE é o ambiente utilizado pelo desenvolvedor para selecionar os componentes que serão integrados para criar os *mashups*. Dentre os principais requisitos do DE, pode-se citar: facilidade de uso através de uma interface gráfica intuitiva, interatividade e por fim, validação, em uma primeira etapa, das composições. O módulo RE,

por sua vez, é responsável por exibir os resultados finais das composições na forma de um *mashup*. Quando um administrador acessa o RE, é encaminhada uma chamada ao *Publisher*, que por usa vez, interage com o CR para executar a composição selecionada e gerar o resultado a ser retornado, que pode incluir, por exemplo, código HTML e JavaScript. Este resultado é então apresentado como uma página Web.

3.3. Componentes de Integração

Para possibilitar que os desenvolvedores integrem os recursos externos, foram definidos componentes de interação que são representados visualmente no *Developer Environment*. Estes componentes podem ser interligados, estabelecendo assim um fluxo de informações, o qual define um *mashup*. Foram definidos 4 tipos principais de componentes, que são apresentados a seguir:

- **Visuais**: responsáveis por representar visualmente o resultado das composições, por exemplo, através de mapas, tabelas ou gráficos;
- **Controle**: definidos para possibilitar a execução de comandos básicos de uma linguagem de proramação, por exemplo, laços e operadores condicionais;
- **Operação**: são funcionalidades integradas no sistema proposto e que servem para executar operações sobre as informações recuperadas dos recursos externos. Exemplos deste tipo de componente são: concatenação, soma, multiplicação e extração de *substrings*;
- **Adaptação**: são os componentes criados com base nas meta-informações definidas nos *wrappers* e que representam, de fato, os recursos externos acessados pelo sistema e de maior interesse dos desenvolvedores.

Há ainda um quinto tipo de componente que serve para representar *mashups* previamente criados. Desta forma, torna-se possível criar composições mais sofisticadas, reutilizando para isto, *mashups* já existentes. O contexto de execução dos componentes depende da implementação do sistema: eles podem tanto executar no lado do cliente, através de linguagens de *client-side scripting* como JavaScript, ou do lado do servidor, utilizando tecnologias *server-side* como Java e PHP.

Na seção seguinte, será vista uma instanciação nessa arquitetura, abordando o estudo de caso do *peering* BGP. Nela, será possível observar uma instância da arquitetura, contendo tanto os tipos de componentes apresentados quanto a integração dos mesmos.

4. Prova de Conceito

Baseando-se na arquitetura proposta na sessão 3, foi desenvolvido um sistema de criação de *mashups* voltado para o problema de *peering* BGP mencionado na sessão 1. O objetivo desse sistema é utilizar tecnologias da Web 2.0 para um abordar um problema que não é facilmente solucionável utilizando as ferramentas de gerenciamento tradicionais. Além disso, ele deve servir tanto como prova de conceito para a arquitetura proposta quanto como objeto de estudo e análise a respeito da eficiência e aplicabilidade das tecnologias utilizadas. Com base nessa análise, espera-se extrair indícios de quais das características alegadas pelos proponentes da Web 2.0 existem na prática, bem como se as tecnologias envolvidas são vantajosas ou não para a utilização no contexto do gerenciamento de redes.

4.1. *Peering* BGP

A Internet é divida em dezenas de milhares de domínios administrativos independentes entre si, a maioria deles pertencente a *Internet Service Providers* (ISPs), onde nenhum dos quais é proprietário da Internet. Ela, na verdade, emerge da combinação e interconexão entre esses domínios. Dentro desses domínios, políticas de roteamento, priorização e qualidade de serviço são definidas pelos seus administradores independentemente, razão pela qual eles são chamados de Sistemas Autônomos (*Autonomous System* - AS). Para definir e controlar as conexões entre esses AS's, o protocolo de roteamento utilizado é denominado BGP. Quando dois AS's criam uma conexão, um AS anuncia rotas para o outro atavés desse protocolo. Tais rotas consistem em conjuntos de endereços IP nos quais o anunciante vai receber e rotear tráfego. Dois AS's conectados são denominados de *peers*, e o processo de interconexão entre os diversos ASs é denominado *peering* BGP [Battista et al. 2006].

Os problemas relacionados a *peering* BGP residem no fato do comportamento dos relacionamentos entre os ASs nem sempre corresponder ao esperado. Por exemplo, existe a situação em que o administrador espera que certa quantidade de tráfego seja encaminhado para um determinado AS vizinho e repentinamente este comportamento muda. Essa situação pode ocorrer devido à ausência de rotas anunciadas pelo *peer* para o sistema local, o que pode infringir as políticas e Acordos de Nível de Serviço (*Service Level Agreement*-SLA) definidos, ou por problemas de configuração no próprio AS, o que requer manutenção. Em ambos os casos, como o volume de tráfego entre os ASs é enorme e o tráfego em questão constitui o próprio tráfego da Internet, a detecção rápida de erros é necessária.

Para um administrador observar adequadamente a situação do *peering* BGP do AS pelo qual ele é responsável, existe uma série de informações que são necessárias:

- Conjunto de *Peers* do AS: ASs que estão conectados ao AS administrado, seja recebendo ou anunciando rotas. Essa informação inclui tanto o número de *peers* como informações sobre os mesmos, tais quais identificador único (ASN), ISP responsável pelo AS e onde o roteador BGP daquele AS está localizado;
- Número de rotas anunciado: O número de rotas anunciado para um determinado *peer* ou o número de rotas que um *peer* anuncia para o AS administrado. Usualmente, esse parâmetro é definido em um SLA entre os ASs;
- Tráfego trocado: A quantidade de tráfego trocada entre o AS administrado e seus *Peers*. Há um custo envolvido nessa troca de tráfego, normalmente definido em SLA, e a administração da rede define políticas de divisão de tráfego roteado por cada AS no sentido de otimizar os custos envolvidos.

Através do SNMP, a MIB BGP-4 fornece o conjunto de *peers* e o número de rotas, entretanto, a extração de um significado dessa informação não é trivial. No caso dos *peers*, é necessário fazer a tradução entre o endereço IP interno do roteador BGP para o endereço externo, utilizando a MIB-2. Quanto ao número de rotas, a MIB BGP-4 fornece toda a tabela de roteamento BGP, contando com dezenas de milhares de entradas. É necessário o tratamento das informações no sentido de definir quais rotas são anunciadas por quais *peers* nesse caso [Battista et al. 2006]. Quanto ao tráfego, existem dificuldades adicionais. Utilizando o SNMP, é necessário recuperar a informação de quais interfaces de rede do roteador correspondem às conexões com os *peers*. A seguir, utilizando a MIB-2, é possível

obter números absolutos de octetos transferidos através dos objetos `If.InOctects` e `If.OutOctects`. Entretanto, essa informação não é uma representação fiel do tráfego trocado, visto que ela é um número absoluto desde a última conexão estabelecida, não fornecendo médias temporais, por exemplo. Assim, é necessária a utilização de uma ferramenta que automatize essa estatítica de tráfego, como por exemplo o MRTG. A integração desse tipo de ferramenta com a informação obtida através de SNMP não é trivial. Além disso, é necessário exibir essa integração de uma forma que o administrador possa extrair significado sem demandar muito tempo ou esforço de análise, visto a importância que é a resolução ágil de problemas envolvendo roteamento inter-AS.

4.2. A Solução Desenvolvida

Baseado na arquitetura proposta na seção 2, foi desenvolvido um sistema capaz de gerar *mashups* voltados ao problema de *peering BGP*, integrando informações sobre um ou mais roteadores BGP e seus *peers*. A função desse sistema é permitir a um administrador criar *mashups* específicos para visualizar informações de roteadores BGP em um mapa. *Mashups* de mapa são, atualmente, a categoria de *mashup* mais encontrada na Web [Wong and Hong 2008], graças, principalmente, à disponibilização e difusão de *APIs* como *Google Maps*, Yahoo Maps e Microsoft Virtual Earth. Mapas de rede são uma ferramenta clássica na visualização de informações de gerenciamento, e a utilização de informações geógraficas vem sendo considerada uma boa prática, adicionando uma nova informação a esses mapas, o posicionamento adequado dos dispositivos gerenciados em termos de latitude e longitude [Kamoun 2005]. Além disso, graças ao fato dos *mashups* de mapa consistirem na categoria mais popular de *mashups* atualmente, a utilização dos mesmos neste trabalho objetiva manter a coerência com o que está sendo realizado na prática em termos de *mashups*.

O protótipo desenvolvido permite que um administrador crie um mapa interativo com informações sobre um ou mais roteadores BGP. As informações agregadas dizem respeito ao tráfego trocado entre os roteadores BGP e seus *peers*, tanto absoluto quanto estatístico, além da quantidade de rotas anunciadas pelos *peers*. Elas são obtidas dinâmicamente, portanto, caso um *peer* ou uma conexão venha a falhar, a detecção é possível. Para criar o mapa, o administrador informa ao sistema o endereço IP do roteador BGP, a *string* de comunidade SNMP do roteador e as URLs das páginas Web de onde serão extraidos as estatísticas de tráfego e rotas anunciadas, tipicamente *front-ends* Web para ferramentas como o MRTG. De posse dessas informações, o sistema se encarregará de extrair as informações do roteador via SNMP, informações geográficas e gráficos de estatísticas. Após isso, o *mashup* é construído e disponibilizado para o usuário. Como a definição de *mashup* é muito ampla, pode-se argumentar que o sistema desenvolvido é, em si, um *mashup*. Entretanto, para fins de clareza, este artigo tratará o sistema como sistema de criação de *mashups* e como *mashups* as páginas Web com informações integradas que são geradas pelo mesmo.

A arquitetura do sistema de criação de *mashups* desenvolvido está exposta na Figura 3, e seus elementos serão discutidos abaixo. Como supracitado, ela segue os princípios da arquitetura proposta na seção 3. Os recursos externos são integrados através de *wrappers* implementados na linguagem Java. A integração é realizada por uma *engine* Java/JSP no lado do servidor. A apresentação é realizada através de duas páginas, um formulário HTML onde o desenvolvedor pode definir os parâmetros para a criação dos

mashups (DE), e os *mashups* em si (RE), páginas Web baseadas em AJAX que exibem um mapa criado com a API do Google Maps, contendo as informações integradas de acordo com a definição do desenvolvedor.

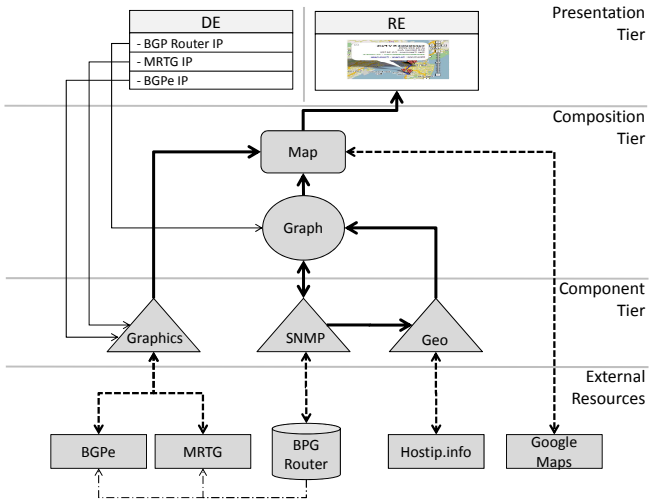

Figura 3. Esquema representativo do protótipo: geração de mashups de visualização de informação de roteadores BGP

Para a construção do sistema, foram utilizados os seguintes componentes:

- SNMP: *wrapper* para acesso a dispositivos gerenciados com suporte a SNMP, que suporta as principais operações do protocolo (*e.g.*, GET, GET-NEXT, WALK), funcionando como um *gateway*. Recebe como entrada o endereço do dispositivo alvo, a *string* de comunidade SNMP do dispositivo, a operação a ser realizada e o OID do objeto que irá sofrer a operação. Retorna o resultado da operação recebido também via SNMP;
- Geo: *wrapper* para o serviço de geolocalização *hostip.info*. Recebe como entrada um endereço IP e retorna as coordenadas geográficas (*i.e.*, latitude e longitude) desse endereço;
- Gráficos: *wrapper* de extração de gráficos para *front-ends* Web de ferramentas de gerenciamento. Para o protótipo apresentado, esse *wrapper* extrai gráficos das ferramentas MRTG, responsável pelas estatísticas de tráfego, e BGPe, responsável pelas estatísticas de rotas anunciadas pelos *peers*. Tais ferramentas podem encontrar-se tanto na Intranet como na Internet, desde a URL submetida para o *wrapper* o aponte para um endereço válido;
- Grafo: *operador* de agregação das informações em um grafo, representando as conexões entre um roteador BGP e seus *peers*. Esse operador recebe como entrada informações geográficas e informações SNMP, principalmente das MIBs BGP-4 e MIB2. Com elas, ele compõe um grafo onde o nó central é o roteador BGP e os demais nós são *peers*. Ele retorna uma *string* em formato JSON representando o grafo.
- Mapa: esse componente consiste em um *wrapper* de visualização. Ele é o responsável por criar o mapa, desenhar a representação da rede e anotar as informações necessárias. Para isso, ele recebe tanto uma descrição da rede no

formato *JSON* e as figuras obtidas no *wrapper* de gráficos, posicionando-as nas conexões adequadas. O resultado desse *wrapper* é um código do *mashup* resultado, em HTML/Javascript, o qual será exibido para o usuário.

Exceto pelo operador de grafo, todos os componentes foram concebidos e implementados de forma independente do protótipo. Eles podem ser reutilizados como pequenas aplicações *standalone* ou em outras composições que utilizem um modelo de *wrappers* baseado em classes Java. Eles também foram projetados com interfaces de acesso bem definidas, de maneira que a implementação dos mesmo pode variar sem prejuízo para as composições dependes dele. Isso é interessante, em especial, no componente de geolocalização. Como a precisão de serviços de geolocalização é limitada, toda a exibição dos *mashups* acaba herdando as mesmas limitações do serviço *hostip.info*, tendo sido necessário no componente de mapas realizar tratamento para evitar sobreposições de nós. O encapsulamento do *wrapper* permite que sua implementação seja alterada para utilizar outros serviços possivelmente mais precisos.

A composição realizada assume a acessibilidade dos seguintes recursos externos:

- Um roteador BGP com agente SNMP que contenha a MIB2 e a MIB-BGP4;
- uma ferramenta de análise de tráfego com *front-end* baseado em página Web, como o MRTG, que foi utilizado durante os testes;
- uma ferramenta de análise de anúncios de rotas BGP, utilizou-se a BGPe;
- um serviço de geolocalização de IP, o serviço utilizado foi o *hostip.info*;
- uma API de construção de mapas, utilizou-se a API do *Google Maps*;

A composição recebe como parâmetros de entrada os dados do roteador BGP e das ferramentas de análise. O *wrapper* SNMP, então, coleta as informações dos *peers* BGP do roteador, endereços IP e números absolutos de tráfego enviado e recebido dos *peers*, verificando quais conexões estão ativas e agregando possíveis *peers* conectados através de mais de um endereço. Além disso, o *wrapper* SNMP alimenta o *wrapper* de geolocalização com os endereços IP dos *peers*. O *wrapper* de geolocalização acessa o serviço Web e retorna todas as latitudes e longitudes dos nós da rede, incluindo as do roteador BGP. De posse dessas informações, o operador de grafo agrega-as, montando uma representação canônica da rede, codifica essa representação no formato JSON e envia para o componente de mapa. O componente de mapa utiliza a API do Google Maps, disponível online, para criar um mapa a partir das informações providas pelo componente de grafo. São inseridos marcadores representando os pares BGP envolvidos nos seus respectivos pontos geográficos e, em seguida, criadas linhas representando as conexões BGP ativas. Durante o desenho, esses elementos visuais são anotados com as informações obtidas via SNMP e são posicionados também os gráficos de tráfego trocado e quantidade de rotas obtidos pelo *wrapper* de gráficos. Finalmente, a página web do *mashup* de mapa é montada e disponibilizada ao usuário. É possível inserir mais de um roteador BGP em um mesmo *mashup*, onde todo o processo será repetido. Também é possível tanto salvar quanto carregar esses *mashups*. Não é salva a página em si, mas sim uma metadescrição do *mashup*. No carregamento, a composição será re-executada e os dados obtidos serão atualizados. É possivel, também, programar o *mashup* para realizar auto-atualização em tempo de execução.

Na camada de apresentação, a página de DE, exibida na Figura 4 é onde o administrador vai construir o seu *mashup*, definindo quais roteadores BGP serão acessados

pelo sistema e inseridos no mapa. Além disso, é necessário que seja apontada as URLs com os endereços das ferramentas de análise de tráfego e anúncio de rotas, durante os testes, foram utilizadas BGPe para rotas e MRTG para tráfego. O administrador pode inserir mais de um roteador, salvar seus *mashups* e carregá-los. A RE, também exposta na Figura 4 exibe os mapas construídos para o usuário, inserindo-os em uma página HTML/JavaScript. Os *peers* BGP são representados por marcadores e as conexões entre eles por linhas entre esses marcadores. Caso haja nós sobrepostos, graças a roteadores na mesma localidade ou a imprecisão do serviço de geolocalização, um deles é deslocado alguns milímetros em uma direção aleatória. A decisão de usar esse mecanismo parte das premissas que a posição exata não é uma informação crítica e também não pode ser obtida, pois os serviços de geolocalização de IP trazem consigo certo grau de imprecisão. Ao selecionar um nó ou uma rota, o usuário pode acessar as informações pertinentes àquela conexão BGP, tais quais os ASs envolvidos, tráfego total, gráfico de média de tráfego e gráfico de quantia de rotas anunciadas. O usuário também pode controlar níveis de *zoom* no mapa de acordo com suas necessidades.

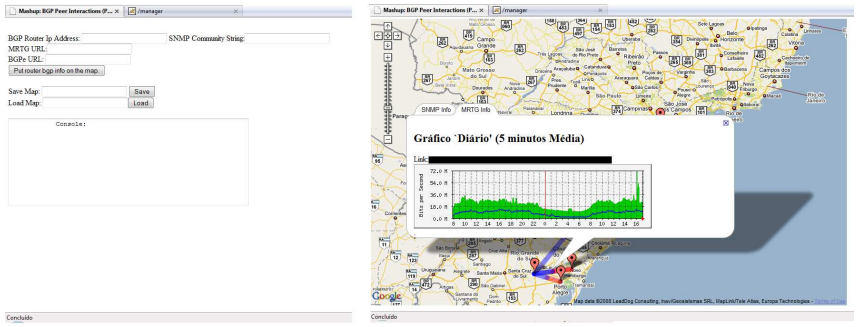

Figura 4. À esquerda, *Development Environment* (DE). À direita, *Runtime Environment* (RE)

As informações agregadas pelos *mashups* criados através do protótipo desenvolvido permitem ao administrador de rede enxergar de forma direta quais as conexões BGP estão ativas. Alguns poucos passos permitem-no verificar tanto informações de tráfego quanto de rotas nas conexões realizadas. Para verificar todas essas informações manualmente, um administrador teria que criar um *script* de obtenção de dados SNMP, ou utilizar um cliente SNMP para consultar diretamente seu dispositivo, além de ter de consultar diversas ferramentas para verificar tráfego, rotas e posição geográfica. Testes preliminares em um ambiente de produção demonstraram que o protótipo desenvolvido de fato reduz o número de passos necessários para a obtenção das informações de *peering bgp*.

5. Conclusões

Neste trabalho, foram apresentados uma arquitetura em três camadas para um sistema de gerenciamento de redes baseado em *mashups* e um protótipo de sistema de criação de *mashups* baseado na mesma. Ela é uma arquitetura em três camadas, cujo propósito é acessar recursos externos disponíveis na *web*, permitindo que administradores de redes integrem-os e exibir os resultados dessa integração, que são *mashups*. O protótipo criado permite que administradores agreguem informações de roteamento de borda dos seus roteadores BGP, tais como tráfego entre *peers* e números de rotas anunciadas, apresentando os resultados dessa agregação em *mashups* de mapa. Uma limitação do protótipo é sua

dependência da precisão de um serviço de geolocalização de IP, a qual não é garantida, e a solução desse problema de precisão foge ao escopo deste trabalho.

Dada a quantidade de ferramentas de gerenciamento de redes disponível atualmente, os *mashups* oferecem o benefício de integrar tais ferramentas, permitindo também a utilização de recursos na Web. O protótipo implementado demonstrou claramente tal vantagem, integrando em uma única aplicação cinco ferramentas diferentes. É possível observar também uma redução significativa no número de passos necessários para a obtenção das informações de *peering* BGP utilizando o protótipo. Originalmente, a obtenção dessas informações consiste na consulta e utilização de diversas ferramentas com interfaces distintas. A utilização do protótipo reduz os passos necessários através da integração das ferramentas necessárias, reduzindo a interação necessária a apenas duas interfaces. Na interface de desenvolvimento, são inseridos parâmetros conhecidos pelos administradores, como o endereço IP de roteadores gerenciados e o endereço do *front-end* Web de aplicações de gerenciamento. A interface de apresentação dessa integração exibe as informações em uma única interface baseada em mapas. A popularidade de sistemas Web com interfaces baseadas nesse tipo de mapa entre usuários finais é um forte indício de sua boa usabilidade, a qual o protótipo criado herda.

Os ganhos em facilidade de uso e rapidez exibidos pelo protótipo levam a concluir que a utilização de *mashups* é, de fato, potencialmente vantajosa no gerenciamento de redes. O crescimento das rede traz consigo uma maior complexidade no gerenciamento, criando a necessidade do uso de tecnologias que melhorem o processo de gerenciamento [Kamoun 2005]. A abordagem de construir *mashups* voltados especificamente para o gerenciamento pode trazer esses benefícios, através da integração de diversas ferramentas em uma única aplicação voltada a atender as necessidades do administrador. A verificação dessas propriedades e sua aplicação no gerenciamento é uma das principais contribuições desse trabalho.

Outra importante contribuição consiste na arquitetura proposta. Ela demonstrou-se útil na elaboração do protótipo apresentado, pois seus princípios de modularidade e componentização permitiram a utilização de componentes modulares e reutilizáveis. Além disso, ela provê uma visão eficiente da divisão entre lado do cliente e lado do servidor na criação de um sistema *Web*. A arquitetura representa com fidelidade as principais práticas adotadas na construção de sistemas de gerenciamento de *mashups*, sendo versátil suficiente para ser útil em contextos diferentes do gerenciamento de redes.

Como trabalho futuro, será desenvolvido um Sistema de Gerenciamento de Redes Baseado em *Mashups*, cujo escopo ultrapassará problemas específicos como o problema de *peering BGP* abordado. Esse sistema seguirá a arquitetura proposta em sua totalidade. Com a implementação desse sistema, obter-se-á uma instanciação direta da arquitetura, permitindo a realização de avaliações de desempenho, impacto na rede e usabilidade. Essas avaliações serão realizadas com o objetivo de aprofundar as conclusões a respeito da viabilidade da aplicação de tecnologias da Web 2.0 no gerenciamento de redes. Espera-se, também, avaliar melhor as vantagens providas por essas tecnologias, através da busca de novos recursos disponíveis na Web que possam ser aplicados no contexto para enriquecer os *mashups* de gerenciamento.

Referências

Alves, A., Arkin, A., Askary, S., et al. (2007). Web services business process execution language version 2.0 (OASIS standard). WS-BPEL TC OASIS, http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html.

Banerjee, N., Dasgupta, K., and Mukherjea, S. (2007). Providing middleware support for the control and co-ordination of telecom mashups. In *MNCNA*, page 11. ACM.

Battista, G. D., Refice, T., and Rimondini, M. (2006). How to extract bgp peering information from the internet routing registry. In *Proceedings of the 2006 SIGCOMM workshop on Mining network data*, pages 317–322, New York, NY, USA. ACM.

Eckerson, W. W. (1995). Three tier client/server architecture: Achieving scalability, performance, and efficiency in client server applications. Open Information Systems.

Ennals, R. and Gay, D. (2007). User-friendly functional programming for web mashups. In *Proceedings of the 2007 ACM SIGPLAN international conference on Functional programming*, pages 223–234, New York, NY, USA. ACM.

Granville, L. Z., da Rosa, D. M., Panisson, A., et al. (2005). Managing computer networks using peer-to-peer technologies. *Communications Magazine, IEEE*, 43(10):62–68.

Harrington, D., Presuhn, R., and Wijnen, B. (2002). RFC 3411 - An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. Available at: http://www.ietf.org/rfc/rfc3411.txt?number=3411.

Kamoun, F. (2005). Toward best maintenance practices in communications network management. *Int. J. Netw. Manag.*, 15(5):321–334.

Liu, H., Ramasubramanian, V., and Sirer, E. G. (2005). Client behavior and feed characteristics of rss, a publish-subscribe system for web micronews. In *Proceedings of the 5th ACM SIGCOMM conference on Internet measurement*, pages 1–6, New York, NY, USA. ACM.

Liu, X., Hui, Y., Sun, W., and Liang, H. (2007). Towards service composition based on mashup. In *Services, 2007 IEEE Congress on*, pages 332–339.

Merril, D. (2003). Mashups: The new breed of web app - an introduction to mashups. http://www.ibm.com/developerworks/web/library/x-mashups.html.

O'Reilly, T. (2005). What is web 2.0. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html.

Soldatos, J. and Alexopoulos, D. (2007). Web services-based network management: approaches and the wsnet system. *Int. J. Netw. Manag.*, 17(1):33–50.

Vianna, R. L., Polina, E. R., Marquezan, C. C., et al. (2007). An evaluation of service composition technologies applied to network management. In *Integrated Network Management*, pages 420–428. IEEE.

Wong, J. and Hong, J. (2008). What do we "mashup"when we make mashups? In *Proceedings of the 4th international workshop on End-user software engineering*, pages 35–39, New York, NY, USA. ACM.

Yu, J., Benatallah, B., Casati, F., and Daniel, F. (2008). Understanding mashup development. *IEEE Internet Computing*, 12(5):44–52.