A Viabilidade do Rerroteamento Pró-ativo em Redes MPLS usando Tecnologia Ativa

R. de B. Correia, E. L. Cecilio, A. P. M. Dumont, L. F. Rust da C. e Luci Pirmez

Núcleo de Computação Eletrônica - Universidade Federal do Rio de Janeiro
Tel: 021 2598-3159 - Caixa Postal 2324, Rio de Janeiro, RJ, Brasil

reinaldo@posgrad.nce.ufrj.br, cecilio@ime.eb.br,
dumont@unisys.com.br, {rust,luci}@nce.com.br

Abstract. *Routing systems must offer a flexible interface to performance management systems in networks, which services must be guaranteed. Moreover, pro-active functionalities must also be incorporated in both systems in order to trigger actions before failure occurrences. This paper presents a pro-active rerouting architecture to forward streams through alternatives routes in which resources are available. These rerouting actions are taken when they are requested by performance management system. A prototype was implemented considering active and MPLS technologies. Preliminary testes allowed ascertaining the feasibility of the proposed architecture.*

Resumo. *O sistema de gerenciamento de desempenho em redes com níveis de serviços assegurados deve dispor de interfaces flexíveis com o sistema de roteamento. Além disso, funcionalidades pró-ativas devem ser incorporadas a ambos os sistemas para que ações sejam desencadeadas antes da ocorrência das falhas, tentando evitá-las. Este artigo propõe uma arquitetura de rerroteamento pró-ativo que tem como objetivo redirecionar fluxos por rotas alternativas, onde não exista falta de recursos, quando solicitado pelo gerenciamento de desempenho. Um protótipo foi implementado com o uso de tecnologia ativa e sobre uma infra-estrutura MPLS, onde foram executados testes preliminares com o objetivo de se averiguar a viabilidade da arquitetura proposta.*

1. Introdução

A infra-estrutura de comunicação do futuro, para fornecer níveis garantidos de serviços, deverá não somente dispor de tecnologias de hardware mais complexas, mas também de arquiteturas de software com mecanismos que deverão atuar desde o nível de aplicação até a camada de enlace. O sistema de roteamento, que é o componente da camada de rede, também deverá evoluir de maneira a atender a novos requisitos funcionais. Dentre esses novos requisitos funcionais, o rerroteamento de fluxos pode ser empregado para garantir níveis mínimos de QoS às aplicações. As degradações dos níveis de QoS são comumente temporárias e causadas pela escassez localizada de recursos. Ao redirecionar os fluxos de aplicações por caminhos com recursos ociosos, o rerroteamento pode minimizar ou até mesmo eliminar os efeitos dessas degradações nas aplicações. O ideal é que as ações de rerroteamento sejam feitas de forma pró-ativa de maneira que os fluxos sejam encaminhados por rotas alternativas antes da ocorrência da falha de QoS.

A tecnologia MPLS (*Multiprotocol Label Switching*) tem sido empregada como uma fer-

ramenta para a engenharia de tráfego objetivando a maximização da utilização dos recursos da rede por meio do balanceamento de tráfego nos enlaces é nós. O MPLS também vem sendo utilizado em esquemas para dotar as redes com maior grau de tolerância a falhas, rerroteando fluxos mediante falhas de nós e enlaces. O MPLS aliado ao roteamento baseado em Qos vem sendo considerado para dotar as redes IP com níveis mínimos assegurados de QoS.

A tecnologia ativa que faz uso dos paradigmas de redes ativas e agentes móveis tem recentemente sido considerada como um meio de prover flexibilidade e capacidade de processamento aos dispositivos de redes (roteadores e comutadores) e estações. O presente trabalho propõe uma arquitetura de rerrroteamento pró-ativo que pressupõe o uso das tecnologias ativa e MPLS além do roteamento baseado em QoS e da arquitetura de gerenciamento distribuída ativa AGAD, que foi desenvolvida no NCE/UFRJ. O cenário de aplicação é o ambiente ServiMídia [4][5] também desenvolvido no NCE/UFRJ.

Os esquemas de rerroteamento disponíveis na literatura são meramente reativos com relação às falhas em enlaces e em nós de comutação enquanto a abordagem adotada neste trabalho é pró-ativa e redireciona fluxos mediante tendências de falhas de QoS. Entretanto o rerroteamento em redes baseadas em circuitos virtuais, em especial redes MPLS, tem sido pesquisado com o objetivo de minimizar os tempos de recuperação diante de falhas de nós e de enlaces. Essa redução é vital para viabilizar a implantação de aplicações multimídia em tempo real e de aplicações de missão crítica.

O esquema proposto em [12] utiliza vários caminhos multiponto/ponto para prover balanceamento de carga e recuperação no caso de falhas de nós. As árvores são construídas de tal forma que o rerroteamento seja efetuado pelo nó de ingresso. Este esquema acelera o rerroteamento devido à ausência de sinalização para o redirecionamento do fluxo.

Scott [13] apresenta um algoritmo para fornecer serviços robustos com garantias de QoS sobre uma infra-estrutura de rede IP com nós e enlaces suscetíveis a falhas. O algoritmo proposto calcula múltiplos caminhos com o menor número possível de elementos (nós e enlaces) em comum, minimizando assim o impacto das falhas nos serviços.

Em [14], é salientado que o rerroteamento pode ser empregado tanto na camada de rede quanto na camada física e de enlace. O rerroteamento na camada de rede tem a vantagem de poder ser realizado com uma menor granularidade, permitindo que serviços específicos sejam atendidos. A desvantagem é um tempo de recuperação elevado. Nas camadas física e de enlace, em especial as de redes orientadas a circuito, os tempos de recuperação são reduzidos. A contrapartida é que o rerroteamento, quando efetuado, atinge uma grande quantidade de fluxos simultaneamente. [15] propõe que o rerroteamento para redes IP sobre WDM seja efetuado tanto na camada IP quanto na WDM. O esquema proposto tenta otimizar as ações de rerroteamento considerando restrições de topologia, consumo de recurso na rede e características do rerroteamento nas duas camadas.

Além desta introdução, o artigo está organizado em mais quatro seções. A próxima seção descreve os conceitos básicos que são relevantes para o entendimento da proposta. A terceira seção apresenta a arquitetura de rerroteamento, enquanto a quarta seção descreve o ambiente de implementação, os testes que foram realizados e apresenta análises dos resultados dos mesmos. Por fim, as conclusões, vantagens e limitações, bem como os trabalhos futuros, são apresentados.

2. Conceitos básicos

Esta seção apresenta os conceitos básicos relevantes que nortearam a concepção da arquitetura proposta neste trabalho.

2.1. Tecnologia ativa e o gerenciamento de desempenho pró-ativo

Os paradigmas de redes ativas [8] e agentes móveis [9] utilizam recursos computacionais no interior e/ou nas bordas da rede para a execução de programas sob demanda. Desta forma, novos tipos de aplicações e de serviços podem ser implementados ou atualizados com rapidez e flexibilidade. Embora esses dois paradigmas tenham sido originados em diferentes comunidades de pesquisas visando resolver diferentes problemas, eles se superpõem, fato que populariza o termo tecnologia ativa para referência a um ou a ambos os paradigmas [10]. A diferença entre eles é que redes ativas usam o conceito de processamento na camada de rede, ou seja, voltado para o encaminhamento de pacotes, enquanto agentes móveis executam como aplicações. Sistemas baseados em agentes móveis são projetados para a construção de um ambiente de computação distribuído e interligado por um sistema de comunicação, enquanto o propósito das redes ativas é disponibilizar e tornar mais eficientes e flexíveis as facilidades no sistema de comunicação. Ainda, agentes móveis podem migrar baseados em decisões autônomas, além de poderem também gerar processos filhos ou *threads* para tratarem de problemas específicos e trocar mensagens entre si, funcionalidades não previstas nas redes ativas.

Os agentes, quando instalados nos dispositivos gerenciados, podem realizar processamento local das informações. Esta abordagem, além de reduzir substancialmente os tempos de detecção de falhas e do restabelecimento do funcionamento normal do elemento gerenciado, minimiza a utilização da capacidade dos enlaces da rede devido à redução do tráfego de gerenciamento. A filtragem e fusão de informações antes de as mesmas serem enviadas aos servidores de gerenciamento são exemplos do tipo de processamento que podem ser efetuados nos elementos gerenciados. O simples *polling* via rede pode ser praticamente abolido. Granularidades mais finas para monitoração e controle dos parâmetros de QoS podem também ser utilizadas.

Certos aspectos da monitoração realizada pelo gerenciamento de desempenho, especialmente quanto a medidas de tempo, são difíceis de serem considerados nas abordagens centralizadas de gerenciamento. O retardo gerado pela transmissão dos dados pela rede torna a precisão de medidas questionável. O processamento local ao elemento gerenciado reduz esse problema.

Adicionalmente, programas locais podem, por intermédio de cálculos sobre variações, detectar a tendência de comportamento de parâmetros de QoS. Ações podem então ser desencadeadas localmente e automaticamente e notificações podem ser enviadas ao servidor de gerenciamento ou a outras partes interessadas, quando limites dessa tendência forem ultrapassados. Dessa forma, é possível, mediante análise das tendências, se prever a ocorrência de falhas de QoS, ou seja, realizar-se o gerenciamento pró-ativo em vez de reativo. O processamento realizado localmente por esses agentes ou aplicações, no entanto, não pode ser excessivo em relação à capacidade de processamento disponível no elemento gerenciado em questão.

2.2. Roteamento com QoS

O roteamento IP tradicional ou *best-effort* considera nos cálculos das rotas somente o menor custo enquanto o roteamento baseado em QoS determina as rotas que atendam uma ou várias restrições de QoS. A Figura 1 ilustra a distinção entre os tipos de roteamento. Se estivesse sendo executado um protocolo de roteamento do tipo *best-effort* nos roteadores da Figura 1, onde o custo fosse número de saltos, o caminho escolhido pelo roteador A para alcançar o roteador H seria AH. Entretanto, se o algoritmo de roteamento fosse baseado em QoS e estabelecesse restrição de banda de 10 Mbit/s, os caminhos AEH, ABDH e ACFGH poderiam ser escolhidos. Entre estes, AEH é o de menor custo, considerando-se saltos, embora essa escolha dependa do algoritmo utilizado, uma vez que nem todos realizam a otimização das rotas. Para uma aplicação com uma restrição de retardo fim a fim menor que 45 ms, o melhor caminho seria ACFGH. Já AEH poderia ser selecionado por questões de segurança (roteamento por política).

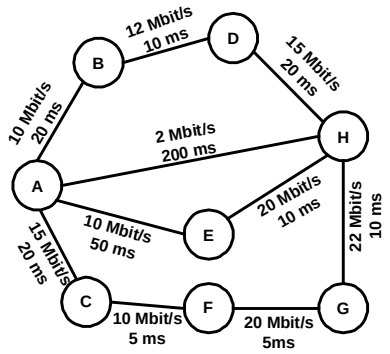


Figura 1 - Cenário de comparação entre os roteamentos *best-effort* e baseado em QoS.

As informações sobre a topologia já são comumente disponibilizadas pelos protocolos de roteamento. Já as informações de restrições de QoS e de estado dos nós e enlaces devem ser obtidas de outras formas. Para isso, algum protocolo de sinalização como, por exemplo, o RSVP, deve ser utilizado para fornecer as informações de restrições dos fluxos. Quanto as informações de estado dos nós e enlaces, extensões aos protocolos tradicionais, como por exemplo aquelas do OSPF para QoS, podem ser utilizadas

Apostolopoulus [6] descreve em detalhes os prós e contras do cálculo de rotas sob demanda ou pré-computadas. A vantagem da primeira abordagem está no fato de que as informações topológicas e de estado são mais recentes e precisas, maximizando a eficiência do protocolo de roteamento. Todavia, caso a taxa de chegada de mensagens de sinalização seja elevada, o nó estará submetido a uma carga computacional elevada, mesmo com o uso de algoritmos de baixa complexidade. A pré-computação de rotas em intervalos de tempo regulares tem o inconveniente de impor ao nó uma maior carga computacional porque toda a tabela tem que ser recalculada. Na abordagem anterior, somente um único caminho sob alguma restrição de QoS é calculado.

O controle das mensagens de atualização do protocolo de roteamento é mais crítico quando restrições de QoS são incorporadas no cálculo de rotas. Isso se deve a algumas métricas de QoS serem dinâmicas e ao envio de mensagens por inundação. O ideal seria o envio de mensagens de atualização para qualquer alteração de valor das métricas, uma vez que as informações de estado ficam mais precisas, tornado o protocolo de roteamento mais eficiente. Todavia, os benefícios seriam anulados pelos custos computacional e de comunicação impostos à rede.

2.3. Multiprotocol Label Switching

A tecnologia MPLS cria um circuito virtual, o caminho MPLS ou *Label Switched Path* (LSP) sob a infra-estrutura IP. Uma vez criado o LSP, os pacotes seguem esse circuito virtual, chegando em ordem ao destino. O *Label Switching Router* (LSR) de borda ou (LER – *Label Edge Router*), em particular, o de entrada, classifica os pacotes de acordo com alguma regra estabelecida utilizando as informações de nível 3 e/ou 4 do cabeçalho do pacote. O pacote é, então, associado a classe (FEC – *Fowarding Equivalent Class*) e um rótulo é anexado à frente do cabeçalho IP. A FEC corresponde a um grupo de pacotes que terão o mesmo tratamento nos LSR's pertencentes ao LSP.

Os LSR's intermediários, ao receberem os pacotes, realizam o seu encaminhamento com base apenas no rótulo. É realizada uma consulta à tabela de rótulos (LIB – *Label Information Base*) para determinar a interface de saída para o próximo salto e o rótulo que substitui àquele recebido. O LER de Saída quando recebe o pacote, retira o rótulo e o encaminha da forma usual.

Os rótulos podem ser criados com base nas tabelas de roteamento (OSPF, BGP), nas requisições de conexão (RSVP) ou nas informações dos pacotes durante a classificação. Após serem criados, as associações de rótulos e FEC's devem ser anunciadas aos demais LSR's para que o LSP seja criado. Essas associações podem ser feitas sob demanda ou de maneira pré-configurada. A distribuição de rótulos pode ser efetuada por intermédio de protocolos existentes modificados como, por exemplo, BGP e OSPF, ou através de protocolos específicos, como é o caso do LDP (*Label Distribution Protocol*). Existem extensões do LDP para suportar roteamento explícito baseado em QoS.

A principal vantagem do MPLS é o melhor desempenho no encaminhamento dos pacotes, uma vez que os LSR's não levam em consideração as informações do cabeçalho IP, tornando o processo mais rápido. Ainda, os pacotes podem ser encaminhados baseados em informações que não estão disponíveis no roteamento IP. Assim, garantias de QoS podem ser estabelecidas atribuindo-se prioridades às FEC's, além do suporte à engenharia de tráfego ser facilitado.

3. Arquitetura de rerroteamento pró-ativo

A arquitetura de rerroteamento proposta neste artigo tem como objetivo dotar o Sistema de Roteamento de um mecanismo de rerroteamento de fluxos pró-ativo. Uma infra-estrutura MPLS será utilizada para aplicação desse mecanismo. A Arquitetura foi concebida tomando-se como base a Arquitetura de Gerenciamento Distribuído (AGAD) [3] e a sua extensão para Gerenciamento de Desempenho Pró-Ativo [2], ambas desenvolvidas no NCE/UFRJ.

Os requisitos funcionais adotados no projeto do sistema de roteamento em uso na Internet visavam robustez, de maneira a torná-los imunes às falhas de nós e enlaces. Entretanto, os mecanismos que compõem esses sistemas de roteamento e o relacionamento entre eles não apresentam flexibilidades e funcionalidades adequadas para permitir que o estado da rede seja configurado de forma rápida e eficiente. O controle do estado da rede que pode ser efetuado interna ou externamente ao sistema de roteamento, além de ser vital para o reencaminhamento de fluxos em torno de áreas críticas, se faz necessário para se alcançar uma melhor distribuição tanto do tráfego nos enlaces quanto dos recursos computacionais dos nós. As operações atualmente oferecidas aos gerentes para exercer tal controle res-

tringem-se a configurações de alguns poucos parâmetros. Essa limitação torna a tarefa de gerenciamento penosa e traz, na maioria dos casos, dependendo do porte da rede, resultados práticos inexpressivos. Esse esquema não oferece suporte ao roteamento baseado em QoS e às tarefas de rerroteamento diante de ocorrências de congestionamento ou de tendências de falhas de QoS.

Cecilio [2] propõe o emprego das funcionalidades pró-ativas do gerenciamento de desempenho em benefício de um fluxo de uma aplicação multimídia adaptativa, como o ServiMídia [4][5]. Esta aplicação desencadeia um processo de adaptação, reestruturando dinamicamente a apresentação, quando da ocorrência de falhas de QoS. Essa reestruturação consiste em substituir as mídias que estavam sendo apresentadas por outras com requisitos de QoS mais brandos de forma a manter a QoP e, ao mesmo tempo, aliviar o consumo de recursos da rede. Entretanto, a latência de adaptação [4] causa a interrupção da apresentação. A capacidade de detectar a tendência de falha de QoS viabilizada pela abordagem de gerenciamento pró-ativo antecipa aquele processo de adaptação, amenizando ou, até mesmo, eliminando o impacto na apresentação. No entanto, o processo de adaptação pode ser evitado através do desencadeamento de ações de rerroteamento dos fluxos por caminhos alternativos que disponibilizem a QoS necessária, conforme sugerido em [2].

3.1. Roteamento pró-ativo

O sistema de roteamento está inserido em um conjunto de mecanismos que Pacifici [1] denomina Sistema de Controle de Tráfego em Tempo Real. Esses mecanismos coordenam a alocação dos recursos das estações e dos nós de comutação como, por exemplo, controle de admissão, classificação de pacotes, escalonamento de filas, política de descarte de pacotes, gerenciamento de filas e de distribuição de rótulos, entre outros.

Neste contexto, a interação entre o operador da rede e o sistema de roteamento ocorre via sistema de gerenciamento de desempenho. Este, por sua vez, interage com o sistema de roteamento monitorando-o através da aquisição de valores de parâmetros e controlando-o a fim de alcançar algum objetivo. Ações de controle podem ser desencadeadas em decorrência de simples violações tanto de valores dos parâmetros em relação à faixa de valores permitida para uma determinada métrica de QoS, quanto de tendências de falhas de QoS. As primeiras são classificadas como ações reativas e as últimas como ações pró-ativas.

O Sistema de Gerenciamento de Desempenho Pró-ativo deve atender, no que tange ao mecanismo de detecção de tendências de falha de QoS e seu impacto no Sistema de Roteamento Pró-ativo descrito a seguir, a dois requisitos conflitantes: (i) capacidade antecipativa e (ii) taxa de acerto das previsões. Quanto mais prematura for a previsão da tendência, melhor será, porque o sistema de roteamento terá mais tempo para reagir. Da mesma forma, caso a taxa de acerto seja baixa, o Sistema de Roteamento Pró-ativo pode sobrecarregar a rede com um excesso de mensagens de atualização dos protocolos de roteamento e de distribuição de rótulos, além do aumento das computações nos nós para recalcular as tabelas de roteamento e atualizar tabelas de rótulos.

O Sistema de Roteamento Pró-ativo é assim denominado por ser capaz de, a partir de uma notificação do Sistema de Gerenciamento de Desempenho Pró-ativo, executar uma ação em tempo hábil, como, por exemplo, a redefinição das políticas de escalonamento de pacotes nos nós de comutação, de controle de admissão, e, também, o disparo, antecipado, do processo de atualização das tabelas de roteamento ou de rótulos, para efeito de

rerroteamento de fluxos. Caso não exista tempo ou recursos disponíveis, o sistema pode simplesmente notificar a impossibilidade de executar qualquer ação. O Sistema de Roteamento Pró-ativo, pode, então, ser conceituado como uma coleção de mecanismos, regulados por uma arquitetura, que permitem o controle do estado da rede, mediante sinalizações reativas e, principalmente, pró-ativas, disparadas com o objetivo de garantir ou manter em níveis aceitáveis a QoS dos serviços que estão sendo prestados, além de otimizar a utilização dos recursos com o uso de engenharia de tráfego.

Uma utilização de um Sistema de Roteamento Pró-ativo pode ser, por exemplo, em uma rede baseada em MPLS, o rerroteamento, ou seja, a substituição de um LSP por outro com recursos suficientes para atender os requisitos de QoS do fluxo em questão. O caráter pró-ativo da abordagem reside também no fato de que todas as ações possíveis são tomadas antes da solicitação de rerroteamento, ou seja, em paralelo com o monitoramento, e não somente após o recebimento de um alarme externo de solicitação de rerroteamento. Assim, a redução da latência do rerroteamento é considerável, pois, ao chegar o momento da solicitação resta, somente, efetivar a troca de rótulos para desviar o fluxo para a nova rota.

3.1. A infra-estrutura AGAD e o gerenciamento de desempenho pró-ativo

A AGAD [3], cuja visão geral é mostrada na Figura 2, provê a infra-estrutura que permite o gerenciamento distribuído das redes, serviços, estações e aplicações.

O Gerente Mor (GM) é o responsável pelo gerenciamento de um conjunto de domínios, via Gerentes de Domínio. O GM cria os Gerentes de Domínio (GD) e os envia para seus respectivos domínios. O funcionamento desses GD passa então a ser monitorado via mensagens do tipo *keep alive* e a integridade, não só dos GD, como dos demais elementos da AGAD, passa a ser monitorada pelos Guardiões. Um domínio equivale a um conjunto de elementos a serem gerenciados sob uma mesma autoridade administrativa.

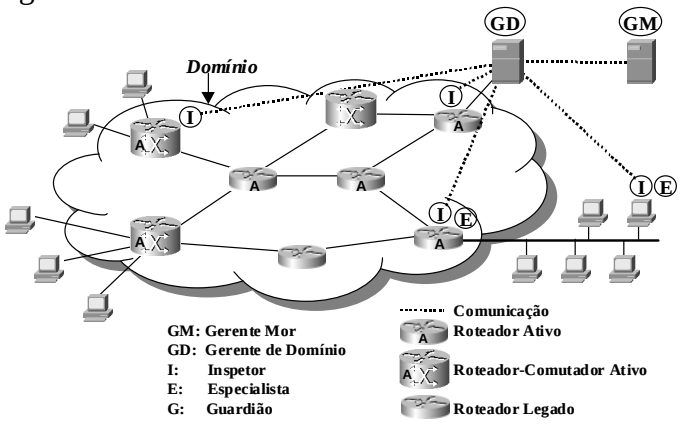


Figura 2 - Visão geral da AGAD

O GD cria e envia Inspetores e Especialistas aos elementos do domínio que devem ser gerenciados. Os Inspetores são os responsáveis pela obtenção local dos dados referentes às funcionalidades que são gerenciadas, ou seja, a monitoração dos parâmetros de QoS. A obtenção desses dados é feita com o uso das facilidades disponíveis localmente, como, por exemplo, acesso às MIB. Os dados podem então sofrer um processamento simplificado o suficiente para não sobrecarregar o elemento gerenciado em questão, para que apenas um mínimo de informações sejam enviadas ao GD. A implementação do Inspetor é configurável dinamicamente, de forma a permitir que as diferentes áreas funcionais de gerenciamen-

to sejam tratadas independentemente, de acordo tanto com a natureza do elemento a ser gerenciado quanto com as intenções do administrador do domínio. É possível também a atualização dos códigos em tempo de execução.

Durante a monitoração, são coletados dados das fontes disponíveis (MIB's, RTCP, sistema operacional, etc) na estação ou no nó de comutação, ou das próprias aplicações. Esses dados correspondem aos parâmetros de controle que devem ser monitorados. As variações desses dados são calculadas para que sejam detectadas as tendências de ocorrência de falha de QoS, quando então Alarmes de Tendência são enviados ao Gerente de Desempenho de Domínio ou a outros sistemas interessados. Mediante extrapolações, realizada pelo Gerente, são calculados os instantes de tempo quando, caso as tendências se mantenham, limites mínimos ou máximos venham a ser desrespeitados. Em função do tempo disponível até que, de fato, as falhas de QoS ocorram, diferentes ações podem ser desencadeadas.

As informações de gerenciamento (consolidadas) recebidas dos Inspetores pelo GD são armazenadas e, após a análise e classificação das mesmas, Especialistas podem ser criados pelo GD para que sejam enviados aos dispositivos ou às estações nas quais são necessárias a realização automática de tarefas especializadas relativas ao gerenciamento. Exemplos de ações que podem ser realizadas por Especialistas são a alteração de políticas de escalonamento, a reconfiguração da utilização de memória, a interação com uma aplicação adaptativa, o envio de uma mensagem para sinalizar a necessidade de que seja desencadeado um rerroteamento, etc.

3.2. Arquitetura proposta

A arquitetura de rerroteamento pró-ativo proposta neste artigo pode ser melhor visualizada através da sua aplicação em um cenário de uma aplicação como o ServiMídia. Além disso, ela utiliza e é baseada na infra-estrutura AGAD, no Sistema de Gerenciamento de Desempenho Pró-ativo [2], na Tecnologia Ativa e no roteamento baseado em QoS. A infra-estrutura de rede que é utilizada nesse cenário é baseada em MPLS. A Figura 3 apresenta o esquema do cenário de aplicação da arquitetura proposta, bem como o ambiente no qual ela atua, os seus elementos e os relacionamentos entre os mesmos.

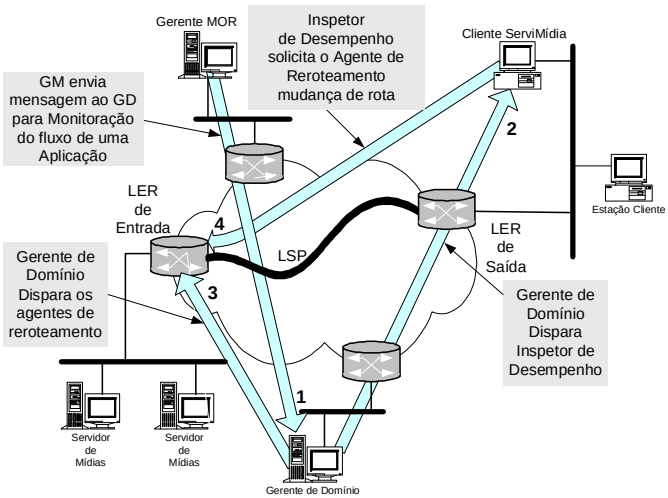


Figura 3 - Infra-estrutura básica necessária ao roteamento pró-ativo

A solicitação de monitoramento de um fluxo é disparada pelo operador da rede ou, automaticamente, quando o serviço é solicitado pelo usuário, para o Gerente de Domínio con-

forme indica a seta 1 da Figura 3. Este envia um Inspetor de Desempenho à estação Cliente ServiMídia (seta 2, figura 3) e um grupo de agentes de rerroteamento ao LER de entrada do LSP ao qual o fluxo pertence (seta 3, figura 3). A seta 4 da figura 3 representa o envio, pelo Inspetor de Desempenho, de um alarme, quando foi detectada uma tendência de falha, para que seja desencadeado o processo de rerroteamento daquele fluxo.

Os agentes de rerroteamento instalam-se no LER de entrada de forma a começar seus trabalhos concomitantemente ao início do monitoramento pelo Inspetor de Desempenho. Eles identificam o maior número possível de trechos de LSP alternativos e ficam preparados para efetuar a troca de rótulos quando do recebimento de um Alarme de Tendência no LER de entrada. O Gerente de Desempenho de Domínio monitora as atividades dos agentes para que, quando necessário, comande a migração dos mesmos para um nó menos sobrecarregado, encerrando seu processamento ou atualizando seus códigos.

O rerroteamento pode ser pleno (de todo o LSP) ou parcial (apenas de um trecho). A segunda abordagem exige o conhecimento de qual é o trecho, que será chamado de crítico, que contribui com o maior percentual do valor fim a fim da métrica de QoS adotada. Se o retardo é a métrica escolhida, o trecho crítico deve ser aquele cujo somatório do retardo dos seus enlaces tenham um valor tal que, percentualmente em relação ao retardo fim a fim do LSP, seja elevado. O critério inicialmente adotado considera como crítico aquele que possuir o maior atraso entre todos os trechos existentes no caminho MPLS.

O rerroteamento parcial se justifica pelos resultados publicados por Apostololous [6] sobre o tempo de processamento do cálculo de rotas em função da distância em saltos entre origem e destino. O tempo aumenta exponencialmente em relação ao número de saltos. A estratégia do rerroteamento parcial é, então, diminuir a distância (em saltos) entre origem e destino (do trecho a ser rerroteado). O número de saltos entre origem e destino tem, também, impacto quando são usados agentes móveis na descoberta dos trechos alternativos em torno do trecho crítico. A área de atuação dos agentes móveis pode ser imaginada como sendo um círculo com raio mínimo igual ao comprimento do trecho crítico cujo centro é o primeiro nó desse trecho. Uma maior área aumenta as chances de se localizar trechos alternativos, porém, aumenta o consumo de recursos computacionais dos nós e enlaces da rede. O uso de agentes móveis tem o mérito de tornar a tarefa de rerroteamento de fluxos independente do protocolo de roteamento.

A Arquitetura de Rerroteamento Pró-ativo proposta é constituída de três agentes: *AgenteLerEntrada*, *AgenteLsr* e *AgenteRotaAlternativa*. O primeiro é o único que não migra e não se clona. Após ser instalado no LER de Entrada, monitora todo o processo de rerroteamento, criando os demais agentes e recebendo mensagens de estado do LSP. O *AgenteLsr* replica o seu código instalando-se em todos os LSR's do LSP do fluxo monitorado para obter os valores de atrasos dos enlaces e envia esses valores ao *AgenteLerEntrada*. O *AgenteRotaAlternativa*, após ser criado, migra para o primeiro nó do trecho crítico (quando este for descoberto), iniciando o processo de descoberta dos trechos alternativos. Este agente também realiza o chaveamento do fluxo, através da troca de rótulos, mediante solicitação do Inspetor de Desempenho.

As tarefas necessárias ao processo de rerroteamento podem então ser divididas em três atividades: (i) instalação de agentes, (ii) difusão de agentes em torno do trecho crítico e (iii) mudança de rota.

3.2.1. Instalação dos agentes ao longo do LSP

Esta atividade consiste na instalação do *AgenteLerEntrada* no LER de Entrada, dos *AgenteLsr* em todos os LSR's e do *AgenteRotaAlternativa* no primeiro nó do trecho crítico. As informações necessárias (endereço IP destino e id do fluxo) estão disponíveis no Gerente de Domínio de Desempenho e são passadas ao *AgenteLerEntrada*. Chegando ao LER de entrada, este agente cria o *AgenteLsr*, que migra para os demais LSR's, deixando uma cópia em cada um deles, até o LER de saída (Figura 4). O *AgenteLsr* hospedado no nó anterior ao LER de Saída envia periodicamente mensagens de estado contendo o valor do atraso do enlace *downstream* ao *AgenteLsr* instalado no seu vizinho *upstream*. A operação se repete até que a mensagem chegue ao *AgenteLerEntrada*, com todas as informações de retardo dos enlaces do LSP.

O *AgenteLerEntrada*, ao receber as mensagens, extrai os valores do retardo dos enlaces e atualiza uma tabela de estado do LSP. Essa tabela é utilizada no cálculo que determina o primeiro e o último nó do trecho crítico. O *AgenteRotaAlternativa* é então criado e enviado para o primeiro nó do trecho crítico.

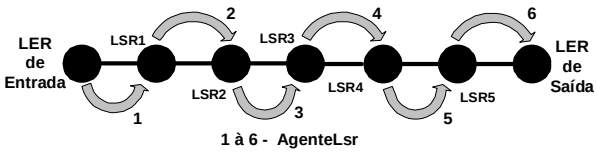


Figura 4 - Instalação do *AgenteLsr*

Para evitar ou minimizar os problemas causados pelas possíveis mudanças de posição do trecho crítico, foi implementado no *AgenteLerEntrada* uma função de disparo que define quais os critérios para que o *AgenteRotaAlternativa* seja iniciado. Este só é criado quando forem obtidos três resultados consecutivos iguais no cálculo do trecho crítico. Este número pode ser configurado. Um valor elevado deste parâmetro diminui as chances de criar desnecessariamente os *AgenteRotaAlternativa's*, mas impõe um maior retardo na definição de trechos alternativos em torno do trecho crítico

3.2.2. Difusão dos agentes em torno do trecho crítico

A difusão do *AgenteRotaAlternativa* em torno do trecho crítico baseia-se na técnica de inundação limitada, restrita à área de busca. O algoritmo é baseado em um esquema híbrido, no qual a migração de um agente só ocorre caso haja necessidade de processamento no nó destino. Caso contrário, utilizou-se de mensagens para troca de informações. Este esquema minimiza o custo de comunicação nos enlaces e computacional nos nós porque a sobrecarga é, em geral, maior para transportar e executar um agente em outro nó do que o transporte e processamento de uma mensagem, conforme documentado em [6].

A topologia da Figura 5 exemplifica o conceito do círculo de busca e facilita a descrição do algoritmo executado pelo *AgenteRotaAlternativa*.

O destino (para a busca de rotas alternativas) do *AgenteRotaAlternativa* pode ser o último nó do trecho crítico (nó 20, figura 5) ou qualquer nó do LSP após o trecho crítico, no sentido do fluxo (nó 24, figura 5). O parâmetro que define o tamanho da área de busca é o Raio da Área de Busca (*RAB*) que é uma variável do *AgenteRotaAlternativa*, cujo funcionamento é análogo ao TTL do pacote IP. O agente só se clona e migra se o RAB não for igual a zero. A segunda condição que inibe a difusão é a localização corrente do agente.

Clones do agente só podem ser enviados pelas interfaces que apresentem retardo acumulado menor do que o trecho crítico e banda disponível maior ou igual à banda exigida pelo fluxo. Uma última condição foi estabelecida para evitar que o nó destino receba um agente através do trecho crítico e, posteriormente, envie mensagens de estado. Sempre que o *AgenteRotaAlternativa* envia uma réplica sua pela interface do enlace pertencente ao LSP, atribui anteriormente o valor *false* à variável *caminhoCrítico*. Assim, quando sua réplica alcançar o nó vizinho *upstream* (nó 13 da figura 5) do LER de Entrada, não será replicada.

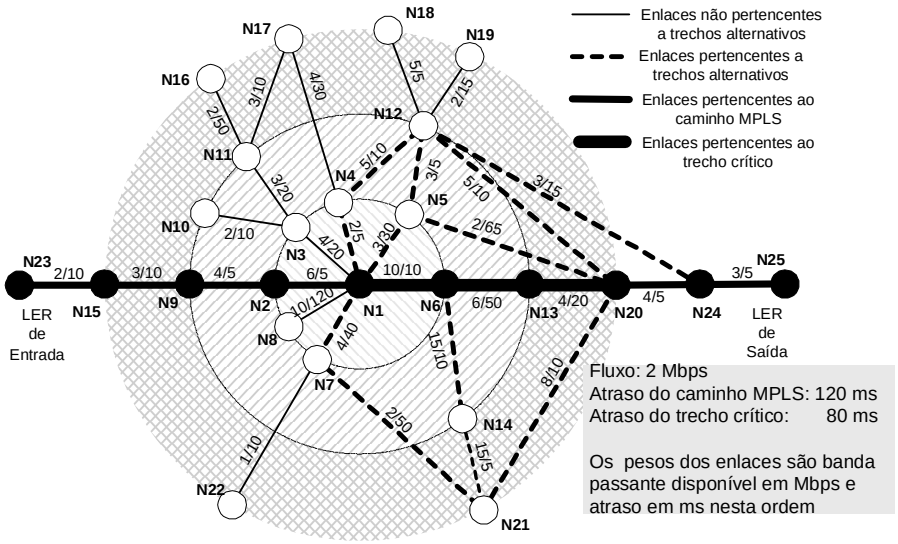


Figura 5 - Trechos alternativos descobertos pelo *AgenteRotaAlternativa*

O *AgenteRotaAlternativa* instala-se em todos os nós pertencentes aos trechos alternativos descobertos e nós destinos (20 e 24 da figura 5). Assim que esses agentes instalam-se nos nós intermediários (4, 5, 12 da figura 5) dos trechos alternativos, preparam-se para receber as mensagens de estado periódicas que são geradas por aqueles agentes hospedados nos nós destinos, para tratar essas mensagens, para obter os valores de banda disponível e atraso dos enlaces *downstream*, para computar o atraso acumulado do caminho alternativo, para agregar estes valores às novas mensagens e para, finalmente, enviá-las aos seus respectivos vizinhos *upstream*. As mensagens de estado são destinadas ao *AgenteRotaAlternativa* do primeiro nó do trecho crítico (nó 1 da figura 5).

Um mecanismo para identificar as rotas dos trechos alternativos, a partir dos nós destinos (nós 20 e 24 da figura 5) em direção ao nó origem (nó 1 da figura 5), para o retorno das mensagens de estado é necessário. Além disso, este mecanismo deve prover os meios para que o nó origem do trecho crítico conheça os trechos alternativos descobertos. Adotou-se um esquema no qual tabelas de retorno são criadas em todos os nós intermediários dos trechos alternativos durante a fase de difusão dos agentes. Os *AgenteRotaAlternativa* ao passarem pelos nós pertencentes aos trechos alternativos deixam registrados na tabela de retorno os números das interfaces pelas quais entraram, juntamente com o número de saltos para alcançá-los. A mensagem de estado criada inicialmente pelo agente no nó destino contém as seguintes informações: *id* do fluxo, *id* do nó origem, *id* do nó destino, número da interface local por onde chegou o agente, número de saltos que no nó destino é atribuído o valor 1, atraso acumulado do trecho crítico e banda disponível do trecho alternativo. Esta mensagem, alcançando o *AgenteRotaAlternativa* do primeiro nó *upstream* (nós 5 e 12 da figura 5), só é reenviada pelas interfaces cujo número de saltos associados (constan-

tes na tabela de retorno) mais o número de saltos acumulados da mensagem for menor do que o comprimento da área de busca (RAB). O número da interface, por onde a nova mensagem é reenviada, é previamente incorporada à esta. O número da interface de saída do nó anterior não é retirado da mensagem recebida. Desta forma, o *AgenteRotaAlternativa* do nó origem, após extrair essa informação da mensagem de estado, consegue identificar o trecho alternativo por meio da seqüência dos números das interfaces de saída que foram utilizadas pelos nós do trecho para enviar a mensagem de estado. O número de mensagens de estados recebidas no nó origem com seqüências diferentes é igual ao número de caminhos alternativos descobertos.

Dois procedimentos foram incluídos no algoritmo para evitar o surgimento de rotas circulares. O primeiro consiste em não enviar a réplica do agente pelas interfaces em que os agentes chegam ao nó. O segundo estabelece a criação de um vetor no qual são registrados os identificadores dos nós que o agente atravessa. O agente iniciando o seu processamento e constatando que o identificador do nó corrente está presente neste vetor, pára o seu processamento antes de desencadear qualquer ação no nó.

3.2.3. Mudança de Rota

As mensagens de estado, ao atingirem o nó destino, suprem o *AgenteRotaAlternativa* com informações para construir e manter a tabela de estado dos caminhos alternativos recém descobertos que permitirá a escolha do melhor trecho alternativo quando da solicitação de rerroteamento. Cada linha desta tabela é constituída dos retardos acumulados, distância em saltos até o nó destino, banda passante disponível do trecho, identificador do nó destino e a seqüência dos números das interfaces. O número de linhas corresponde ao número de caminhos alternativos descobertos. A escolha do trecho alternativo, dentre aqueles disponíveis quando da solicitação de rerroteamento, depende de critérios adotados e configurados. Os critérios adotados têm a seguinte prioridade: (i) menor atraso do que o trecho crítico, (ii) menor distância do novo caminho MPLS, (iii) menor distância do trecho alternativo, (iv) maior banda disponível, (v) menor atraso. Considerando estes critérios, a escolha do trecho alternativo no caso ilustrado na Figura 5, a partir de uma requisição de rerroteamento, recai sobre o trecho {N1, N5, N12, N24}, porque apresenta a menor distância do novo LSP, a menor distância do trecho alternativo e a maior banda disponível. Os critérios de distância, tanto do LSP quanto do trecho alternativo, não foram suficientes para definir a opção vencedora.

Após a escolha do trecho alternativo, o *AgenteRotaAlternativa* envia uma mensagem de criação de LSP alternativo contendo a seqüência do número de interfaces ao mesmo agente instalado no nó destino (nó 24, figura 5). Este agente, então, envia uma mensagem de geração e associação de rótulos ao seu vizinho *upstream* do trecho alternativo que está definido na seqüência do número de interfaces. Conforme as mensagens vão sendo processadas pelos *AgenteRotaAlternativa's* dos LSR's do trecho alternativo escolhido, os rótulos são criados e as associações estabelecidas. Quando a mensagem chega ao nó origem (nó 1, figura 5) com o rótulo de entrada do seu nó vizinho *downstream*, o chaveamento do fluxo é efetuado.

4. Ambiente de implementação e testes

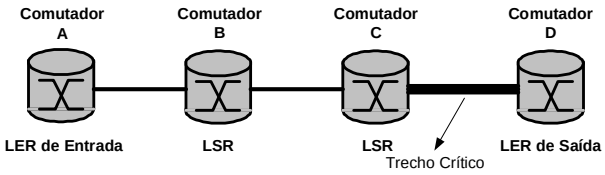
Foi montada a topologia de LSR's que está apresentada na Figura 6. Os trechos alternativos foram omitidos por não serem relevantes no teste que foi realizado. As estações utili-

zadas foram PC's cuja configuração está descrita na mesma figura, com sistema operacional Linux Red Hat 7.2 (kernel 2.4.19). As funcionalidades necessárias à comutação de rótulos foram incorporadas pela atualização do Kernel com o *patch* desenvolvido por James R. Leu [16], que transforma a estação em um LSR MPLS. Os agentes, através de scripts desenvolvidos em Perl, consultam e alteram as tabelas de rótulos e de roteamento, criando, alterando e encerrando LSP's. O LSP implementado foi configurado manualmente.

A infra-estrutura de mobilidade oferecida pelo ambiente µCode [11] cria um ambiente computacional (o µServer) sobre a JVM para a execução de *threads*, que são os agentes móveis. Os µServer's são responsáveis pela serialização e remontagem dos agentes. O µCode implementa a mobilidade fraca, ou seja, os dados são mantidos enquanto o estado de execução se perde após a migração. O fator determinante na escolha do µCode para atuar na camada de rede foi a mobilidade gradativa, permitindo a realocação de código na medida exata para atender à necessidades específicas.

Os agentes foram implementados na forma de *threads* móveis devido à menor sobrecarga computacional ao processador em comparação ao uso de processos. Quando necessário, foram usadas técnicas de sincronização.

O teste que foi realizado referia-se às tarefas de instalação dos agentes e consistia em determinar o tempo decorrido entre o instante em que o *AgenteLerEntrada* é instanciado e o instante em que o *AgenteRotaAlternativa*, já instalado no primeiro nó do trecho crítico, executa a sua primeira instrução. Esse tempo permite visualizar a ordem de grandeza do tempo mínimo de duração de um fluxo para o qual seja possível o seu monitoramento para fins de rerroteamento.



Comutador	CPU	Clock (MHz)	Memória Cache (Kbytes)	Memória RAM
A	'K6-II	350	-	256
B	K6-II	500	-	256
C	K6-II	500	-	256
D	K6-II	500	512	256

Figura 6 – Topologia de teste e configuração das plataformas

As rotas para a troca de mensagens de controle e estado entre os agentes foram configuradas estaticamente nas tabelas de roteamento via *ip route*. Vale ressaltar que, em redes MPLS reais, os pacotes IP (não rotulados) referentes a essas mensagens seguiriam o caminho determinado pelos protocolos de roteamento tradicionais (RIP, OSPF, IGRP), podendo ser, inclusive, diferentes daquele do LSP.

No caso das mensagens de estado geradas pelo *AgenteLsr* do comutador anterior ao LER de Saída em direção ao *AgenteLerEntrada*, as rotas estáticas não são necessárias. Isto ocorre porque essas mensagens devem ser processadas em todos os nós intermediários antes de chegar ao LER de Entrada, para que as informações locais de estado desses nós intermediários sejam incorporadas a elas.

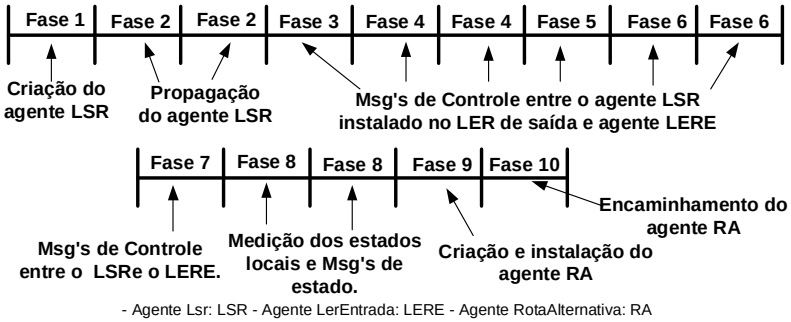


Figura 7 - Fases de Instalação dos agentes de rerroteamento no caminho MPLS

Para determinar os limites do esquema de rerroteamento adotado, é necessário não somente medir o tempo total gasto, mas também os tempos das diversas fases que compõem a instalação dos agentes ao longo do LSP. Os tempos de execução de certas operações dependem do comprimento do LSP e da distância percorrida, em saltos, pelas mensagens de controle e de estado. Essas fases, ilustradas na **Figura 7**, não ocorrem sempre de forma seqüencial. Dependendo do comprimento do LSP, algumas fases podem ser executadas tantas vezes quantos forem o número de saltos entre o primeiro comutador após o LER de Entrada e o LER de Saída, enquanto outras podem até mesmo não serem executadas. A fase 2, por exemplo, não existiria caso o caminho MPLS possuísse somente um LER de Entrada e um LER de Saída.

Vários são os fatores que causam impacto negativo no tempo total gasto para a instalação dos agentes de rerroteamento ao longo do LSP. Consultando a **Figura 7**, verifica-se que as fases 2, 4, 6 e 10 são executadas em função do número de nós do LSP. Em todas estas fases, a operação de encaminhamento IP tem obrigatoriamente que ser realizada para o envio de mensagens de estado e controle. Neste teste, as tabelas de roteamento foram povoadas artificialmente com a adição de 50 rotas estáticas.

O tamanho das mensagens é um fator que influencia o tempo de instalação. As mensagens de controle devem ser o mais curtas possíveis para minimizar o tempo de transmissão dos pacotes correspondentes e, principalmente, eliminar a fragmentação. Como o tamanho das mensagens não é maior do que 18 octetos, não ocorreu fragmentação durantes o teste. Já as mensagens de estado possuem tamanhos variáveis. As informações de retardo dos enlaces são agregadas a essas mensagens à medida que vão passando pelos LSR's, fazendo com que o seu tamanho aumente. Para simplificar a análise dos resultados dos testes, adotou-se um tamanho fixo para todas as mensagens de estado que corresponde ao tamanho da mensagem que o *AgenteLerEntrada* receberia caso o comprimento do LSP fosse de dez saltos (84 octetos). Esta simplificação acarreta em tempos maiores pois, se o LSP tivesse 10 saltos, o tamanho das mensagem variaria de 21 até 84 octetos. É esperado que o impacto desta simplificação seja maior no processamento do que na transmissão, porque a diferença teórica entre os tempos totais de transmissão na situação real e na situação simplificada de teste, a 10 Mbit/s, é de 50.4 µs. A operação de cálculo do trecho crítico da fase 9 depende, também, do comprimento do LSP, devido ao número de iterações do algoritmo variar em função do número de enlaces e quantidade de métricas envolvidas. Estabeleceu-se que o algoritmo foi sempre executado para um caso com dez saltos, independentemente do comprimento real do caminho MPLS.

A Tabela 1 resume os tempos de execução das diversas fases. Estão indicados os tempos máximos, mínimos e médios de cada rodada. Os tempos das fases correspondem a um ú-

nico salto.

Tabela 1 – Tempos de execução das fases do teste de instalação.

	Fase 1	Fase 2	Fase 3	Fase 4	Fase 5	Fase 6	Fase 7	Fase 8	Fase 9	Fase 10
Médias	1108	502	6	3,5	6	3,5	6	5	663	22

Obs: tempos em milissegundos.

A diferença encontrada nos tempos das fases 1 e 2 reside no fato de que na primeira ocorrem três operações de alto consumo de tempo de CPU, que são as instanciações dos *AgenteLerEntrada*, dos *AgenteLsr* e clonagem deste último. Nas fases em que ocorre migração de agentes (fases 1, 2 e 11) vários scripts em Perl são executados para consultar as tabelas de roteamento e de rótulos com a finalidade de determinar os destinos dos agentes. Isto eleva o tempo de execução dessas fases, razão pela qual é necessário investigar outras formas de interação com o sistema operacional do nó.

Os tempos obtidos apesar de serem, até certo ponto, elevados, não inviabilizam o rerroteamento, porque essas operações são executadas durante a monitoração do fluxo. Apesar de não terem sido apresentados os tempos das atividades de mudança de rota, que ocorrem somente após a solicitação de rerroteamento, é possível antever que essas atividades ocorrerão em tempo hábil, porque as operações envolvidas são trocas de mensagens entre os *AgenteRotaAlternativa* e poucas chamadas aos sistemas operacionais dos nós para a geração e associação de rótulos.

6. Conclusões e trabalhos futuros

Neste artigo, inicialmente, foi salientada a necessidade dos sistemas de roteamento evoluírem de maneira a viabilizar a automação das ações dos sistemas de gerenciamento de desempenho, preferencialmente, de forma pró-ativa. Essa evolução implica na disponibilização de interfaces flexíveis e no atendimento a novos requisitos funcionais, como, por exemplo, o rerroteamento de fluxos a partir de solicitações externas e não somente a eventos internos ao sistema de roteamento para a recuperação de rotas devido a falhas de enlaces ou nós. Neste contexto, foi proposta uma arquitetura de rerroteamento pró-ativo que, mediante solicitação externa, permite efetuar o chaveamento de fluxos de aplicações por caminhos alternativos que atendam aos requisitos de QoS desses fluxos quando da tendência de ocorrência de falhas de QoS. O caráter pró-ativo da arquitetura está também no fato de que todas as ações possíveis, exceto a troca de rótulos, são executadas antes da solicitação de rerroteamento, ou seja, são feitas durante o monitoramento do fluxo.

Os resultados preliminares apontam que não somente a estratégia adotada na arquitetura é apropriada, como também é viável a utilização de uma linguagem interpretada e independente de plataforma como Java e a infra-estrutura de mobilidade µCode em operações da camada 3, em particular, no rerroteamento de fluxos.

Como trabalhos futuros, os testes referentes das outras duas atividades que são a difusão dos *AgenteRotaAlternativa* e a mudança de rota permitirão determinar os limites da arquitetura em relação ao tempo de duração dos fluxos que poderão ser atendidos em função do comprimento do caminho MPLS e do trecho alternativo. Outra linha de estudo é verificar o impacto do tamanho do código do agente nos tempos obtidos e, se necessário, fazer ajustes na arquitetura para otimizar seu desempenho. Finalmente, a escalabilidade é uma questão que deve ser cuidadosamente investigada. Como o esquema de rerroteamento pró-ativo atua em prol de fluxos individuais, a escalabilidade em princípio é baixa. En-

tretanto, dependendo de certas condições, por exemplo, fluxos pertencentes a um mesmo LSP, não haveria a necessidade de difundir novos agentes. Nestas condições, a estratégia adotada, em relação à migração de agentes, independe do número de fluxos monitorados.

7. Referências

[1] Pacifici, G. e Stadler, R., "An Architecture for Performance Management of Multimedia Networks", Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management, maio de 1995.

[2] Cecilio, E. L. "Uma Arquitetura de Gerenciamento de Desempenho Pró-Ativo Distribuído Usando Tecnologia Ativa", Dissertação de Mestrado, NCE/IM/UFRJ, Dezembro de 2002.

[3] Dumont, A. P. M, "AGAD: Uma Arquitetura de Gerenciamento Ativo Distribuído", Dissertação de Mestrado, NCE/IM/UFRJ, Outubro de 2002.

[4] Gomes, R.L., "Autoria e Apresentação de Documentos Multimídia Adaptativos em Redes", Dissertação de Mestrado, NCE/IM/UFRJ, 2001.

[5] Cunha, E.C., "Uma Estratégia de Criação e Apresentação de Documentos Multimídia Adaptativos em Rede", Dissertação de Mestrado, NCE/IM/UFRJ, 2000.

[6] Apostolopoulos, G. et al, "QoS Routing Mechanisms and OSPF Extensions", RFC-2676, IETF, agosto 1999.

[7] Rubinstein, M.G., Duarte, O.C.M. e Pujolle, G., Evaluating the Performance of Mobile Agents in Network Management, Proceedings of IEEE Global Telecommunications Conference, 1999.

[8] Psounis, K., "Active Networks: Applications, Security, Safety and Architectures", IEEE Communications Surveys, Abril de 1999.

[9] Bieszczad, A., Pagurek, B. e White, T., "Mobile Agents for Network Management", IEEE Communication Surveys, Fourth Quarter, 1999.

[10] Hu, C., Chen, W., "A Mobile Agent-based Active network Architecture", International Conference on Parallel and Distributed Systemas, ICPADS´00, IEEE, 2000.

[11] Picco, G.P., "µCode: A Lightweight and Flexible Mobile Code Toolkit", Proceedings of the 2nd International Workshop on Mobile Agents 98, September de 1998.

[12] H. Saito, Y. Miyao e M. Yoshida, *Traffic Engineering using Multiple Multipoint-to-Point LSPs*, Info- com'2000.

[13] Scott Seongwook Lee and Mario Gerla, *Fault tolerance and load balancing in QoS provisioning with multiple MPLS paths*, Lecture Notes in Computer Science, vol. 2092, 2001.

[14] Autenrieth, A. e Kirstdter, A. *Fault-Tolerance and Resilience Issues in IP-Based Networks*, Second International Workshop on the Design of Reliable Communication Networks (DRCN), abril de 2000.

[15] Andrea Fumagalli et al, *IP Restauration vs. WDM Protection: Is there an Optimal Choice?*, revista IEEE Network, novembro/dezembro de 2002.

[16] James R. Leu, http://sf.net/projects/mpls-linux/.