Sumário

Capítulo

3

Serviços Diferenciados

3.1 Introdução

[1]

O modelo atual de serviços da Internet tem suas raízes em pesquisas que datam de três décadas atrás. Embora tenha funcionado e ainda funcione muito bem para vários tipos de aplicações, já são notados sinais de que o mesmo possa estar atingindo o seu limite. Isso gera a necessidade de se pensar em um novo paradigma de serviços para a Internet e, conseqüentemente, para a Web, conduzindo-a a novos níveis de funcionalidade.

A Internet é uma rede global que depende do protocolo IP para o transporte dos dados, o qual foi projetado para ser bastante flexível, podendo funcionar sobre uma gama de tecnologias de rede diferentes, fato que contribuiu para a disseminação da Internet e das redes IP por todo o mundo. Com isso, serviços como o correio eletrônico e a Web tornaram-se parte da vida das pessoas, seja para fins de entretenimento, educativos ou comerciais. Além disso, o que se nota atualmente é a convergência de outros tipos de redes, já há mais tempo estabelecidas em nossa rotina, como as redes de telefonia, rádio e TV, para a Internet (Stardust, 1999b), a qual pretende ser o principal meio de transmissão de informações, condução de negócios e entretenimento no futuro. A questão que se coloca é se a mesma está preparada para responder a essas novas demandas.

Este capítulo aborda os problemas atuais da Internet e como a introdução da noção

[1]Este capítulo é parte integrante de: TEIXEIRA, Mário A. M. Suporte a Serviços Diferenciados em Servidores Web: Modelos e Algoritmos (Orientador: Prof. Marcos José Santana). *Tese de Doutorado*, USP–ICMC, 2004. 140p.

de qualidade de serviço pode torná-la mais eficiente. Em particular, destacam-se a abordagem de serviços diferenciados sobre redes IP e a viabilidade de seu emprego em nível de aplicação, particularmente em servidores web, tema que vem a ser a razão desta tese.

3.2 Limitações da Internet Atual

O tráfego da Internet tem crescido enormemente nos últimos anos, principalmente depois do surgimento da Web e não se vê sinais de que essa tendência vá se reverter em um futuro próximo. Este crescimento não se deve apenas ao aumento do número de usuários e de aplicações, mas também se dá em decorrência do aparecimento de novos tipos de aplicações, principalmente multimídia e de tempo real, que exigem da Internet uma resposta para a qual ela não foi projetada.

O fato é que uma rede como a Internet, baseada no protocolo IP, somente é capaz de fornecer um serviço de "melhor esforço" (*best-effort*), ou seja, a rede tentará, de todas as formas, entregar os dados que lhe foram confiados, no menor tempo possível e sem erros, mas não será dada nenhuma garantia às aplicações de que isso realmente ocorrerá. Este comportamento é decorrente da própria concepção do protocolo IP, o qual foi projetado para ser simples, eficiente e flexível, fornecendo um serviço de entrega de datagramas do tipo não-confiável. Na verdade, em situações nas quais o tráfego na rede é inferior a sua capacidade, o protocolo IP e seus congêneres conseguem realizar suas funções com bastante rapidez e eficiência. O problema está nos momentos de congestionamento, quando o serviço na rede pode apresentar níveis inconsistentes ou até mesmo imprevisíveis, devido à variação do atraso na entrega dos pacotes e à perda de dados (Vasiliou, 2000).

Tradicionalmente, a abordagem empregada pelas redes IP, desde o seu início, foi tornar os elementos extremos da rede (*hosts*), complexos e os elementos internos (roteadores), mais simples, destinando-se estes meramente a verificar o endereço IP dos datagramas em uma tabela de rotas a fim de determinar qual o próximo "salto" (*hop*) a ser dado (Stardust, 1999b). Esta solução tem sido suficiente para a transmissão de dados convencionais (dados provenientes de *e-mail*, transferência de arquivos, navegação na Web), comumente encontrada na Internet, mas se revela inadequada para os novos tipos de aplicações que se espera colocar sobre a rede.

O tráfego atual na Internet não é apenas intenso, mas se origina de aplicações que possuem requisitos operacionais variados, que a infra-estrutura existente na rede não está preparada para suportar. Aplicações multimídia, por exemplo, requerem uma alta largura de banda e, em geral, podem suportar pequenas perdas de pacotes em favor de uma melhor sincronização na entrega dos mesmos. Por outro lado, aplicações de tempo real, como a telefonia sobre IP, colocam pouca demanda sobre a largura de banda, mas, em compensação, possuem requisitos de temporização bastante rígidos (Stardust,

1999b) que, se não forem atendidos, podem inviabilizar a comunicação entre as partes. Aplicações de transferência de arquivos, por sua vez, demandam uma certa largura de banda e não podem tolerar a perda de nenhum bit de informação. Há também aplicações que necessitam de um serviço *multicast* como, por exemplo, videoconferência e transmissão de rádio e TV via Web. Finalmente, a Web vem sendo cada vez mais utilizada para a condução de negócios, uma área que exige alta confiabilidade e disponibilidade do meio de transmissão e cujos usuários estão dispostos a pagar mais por um serviço mais previsível e de melhor qualidade (Xiao & Ni, 1999).

Possíveis Soluções

Do exposto acima, conclui-se que não existe um tipo de rede que possa satisfazer a toda a gama de requisitos colocados. À primeira vista, o problema da Internet poderia parecer de largura de banda, ou seja, supondo-se que fosse possível acrescentar capacidade de transmissão indiscriminadamente à rede, teoricamente seria possível acolher todo o tráfego existente, satisfazendo a todos.

Infelizmente, a experiência mostra que, não importa a quantidade de largura de banda que seja disponibilizada, a tendência é, em pouco tempo, tal quantidade ser completamente exaurida. Além disso, como foi comentado, para algumas aplicações o ponto crítico não está na quantidade de largura de banda disponível, mas sim na latência de transmissão. Há que se considerar também que a Internet é uma rede composta por redes, compreendendo as mais diferentes tecnologias, portanto, não é tão simples assim incrementar a capacidade dos seus canais de comunicação uniformemente. Outro problema é a existência de tráfego em rajadas, que pode levar a demanda por largura de banda a níveis muito superiores aos experimentados usualmente.

Sendo assim, vê-se que dotar a rede da maior capacidade possível, além de economicamente inviável, não é a solução adequada para alguns casos. O que se precisa é de um gerenciamento ativo da largura de banda disponível, de modo que se possa fornecer aos usuários e aplicações diferentes classes ou níveis de serviço (Xiao & Ni, 1999). Para tanto, os serviços IP precisam ser incrementados e isto pode ser alcançado agregando-se alguma "inteligência" aos elementos internos da rede a fim de diferenciar o tráfego que passa pelos mesmos. É sobre isso precisamente que trata a qualidade de serviço, assunto da próxima seção.

3.3 Qualidade de Serviço

3.3.1 Conceitos Básicos

Qualidade de Serviço (QoS) pode ser definida como a capacidade de fornecer a um elemento da rede (aplicação, *host* ou roteador) algum nível de segurança de que seus requisitos de tráfego e serviço serão satisfeitos (Stardust, 1999b). Está relacionada com a garantia de um atraso de entrega (*delay*) e uma perda de pacotes suficientemente baixos para certos tipos de aplicações ou tráfego (Zhao *et al.*, 2000). Os requisitos de qualidade podem ser determinados por fatores humanos, por exemplo, limites máximos para o atraso a fim de permitir um diálogo entre duas pessoas; por razões comerciais, por exemplo, a necessidade de concluir uma certa tarefa em um tempo mínimo; ou por aplicações críticas cujo funcionamento fique comprometido caso o atraso ou *jitter* (variação do atraso) superem certos limites.

Fornecer QoS não é uma tarefa trivial, mesmo em redes pequenas e proprietárias, muito menos em uma rede global como a Internet. Para que se tenha QoS, faz-se necessária a cooperação de todos as camadas da rede, de cima a baixo, assim como de todo e qualquer elemento da rede, de fim-a-fim. Qualquer garantia de QoS será tão forte quanto o mais frágil elemento nessa cadeia entre o emissor e o receptor (Stardust, 1999b).

É importante ressaltar que o emprego de QoS não é capaz de criar largura de banda, ou seja, a rede nunca poderá fornecer aquilo que ela não tem. O que a QoS faz é administrar a largura de banda existente segundo a demanda das aplicações e dentro de certos parâmetros de gerenciamento e desempenho da rede. Uma rede habilitada para fornecer QoS continuará dando suporte ao tráfego de melhor esforço, contudo parte da largura de banda será reservada para as aplicações de mais alta prioridade. Também faz parte das atividades de gerenciamento de QoS garantir que essas aplicações menos "exigentes" não serão anuladas pelas aplicações mais "nobres".

A QoS pode ser descrita de forma absoluta ou relativa. Uma especificação de QoS em termos absolutos fornecerá métricas a serem cumpridas, relacionadas, em geral, ao atraso ou perda de pacotes, por exemplo, uma meta como "nenhum pacote poderá ter um atraso maior que x milissegundos". Caso a rede não seja capaz de garantir um serviço com essa característica, então a aplicação não terá acesso ao meio de comunicação.

A QoS relativa, por sua vez, trabalha com a idéia de diferenciação de serviços, comparando o tratamento recebido por uma classe de pacotes com aquele dado a outra classe (Zhao *et al.*, 2000). Neste caso, a rede garante que uma aplicação em uma classe de mais alta prioridade nunca receberá um serviço pior que o de qualquer classe inferior. Neste modelo, não é possível dar garantias rígidas às aplicações, em termos de métricas, como no caso anterior, pois a QoS resultante dependerá da carga atual na rede, em cada classe (Vasiliou, 2000).

Segundo Zhao *et al.* (2000), é possível identificar dois aspectos importantes dentro do tema de QoS: *garantia de desempenho* e *diferenciação de serviços*. A primeira se relaciona com o gerenciamento de largura de banda, perda de pacotes, atraso e *jitter*. A largura de banda é o recurso mais importante e a sua disponibilidade e forma de alocação afetam as outras características. A diferenciação de serviços consiste em fornecer diferentes níveis de QoS para diferentes aplicações segundo seus requisitos. Esta diferenciação pode ser feita em um grupo de pacotes individual ou em um agregado de grupos de pacotes relacionados (Vasiliou, 2000). O primeiro caso se remete à definição de fluxo, conceito explorado a seguir.

Um fluxo é definido como um fluxo contínuo de dados (*stream*), individual, unidirecional entre duas aplicações (emissor e receptor) (Vasiliou, 2000). Tradicionalmente, um fluxo é caracterizado por uma quíntupla composta por: endereço IP de origem, número de porta de origem, endereço IP de destino, número de porta de destino e protocolo de transporte. Garantir QoS em uma granulosidade tão fina quanto a de um fluxo permite isolá-lo de outras aplicações possivelmente mal-comportadas, mas, por outro lado, compromete a escalabilidade desta solução em redes globais como a Internet, que podem, em um certo instante, apresentar centenas de milhares de fluxos (Zhao *et al.*, 2000). Por essa razão, às vezes é mais vantajoso realizar o gerenciamento de QoS em um nível mais alto, agrupando-se os pacotes (fluxos) em classes com requisitos de QoS similares, cada uma tratada de modo diferente. Assim, resolve-se o problema de escalabilidade, embora as garantias de desempenho dadas não possam ser tão rígidas quanto aquelas da abordagem por fluxos individuais.

3.3.2 Caracterização do Tráfego

Como já foi visto, diferentes aplicações possuem diferentes requisitos operacionais e demandam da rede diferentes serviços para que possam funcionar satisfatoriamente.

O tráfego gerado pelas aplicações tem uma influência direta nas demandas de QoS impostas à rede e pode ser caracterizado segundo duas dimensões: previsibilidade da taxa de dados e sensibilidade ao atraso de entrega e *jitter* (Stardust, 1999b; Magalhães & Cardozo, 1999). Considerando-se a previsibilidade (Tabela 3.1), o tráfego pode ser do tipo fluxo contínuo (*stream*) ou em rajadas. Tráfego de fluxo contínuo é característico de aplicações que geram mídia contínua, como áudio e vídeo. Tráfego em rajadas é típico de aplicações como transferência de arquivos e interações cliente-servidor.

O tráfego também pode ser caracterizado, segundo sua tolerância ao atraso e *jitter*, em cinco categorias (Tabela 3.2). O tráfego assíncrono, também denominado "elástico", não possui restrições temporais e admite bem o serviço de melhor esforço. Aplicações tradicionais como *ftp* e *e-mail* estão incluídas nesta categoria. O tráfego síncrono possui restrições temporais, mas é possível compensar as variações observadas através de esquemas de sin-

Tipo da Taxa	Descrição
Fluxo Contínuo	Entrega de dados a uma taxa relativamente constante (CBR — *Constant Bit Rate*)
Em Rajadas	Blocos de dados são entregues de forma imprevisível, a uma taxa de dados variável (VBR — *Variable Bit Rate*), que pode chegar a utilizar toda a largura de banda se não for empregado algum tipo de controle

Tabela 3.1: Caracterização do tráfego segundo a taxa de dados (Stardust, 1999b)

cronização no receptor. Aqui se enquadra o tráfego gerado por aplicações de áudio e vídeo sob demanda. O tráfego interativo impõe valores máximos ao atraso e *jitter*, sob pena de comprometer a interatividade da comunicação. É característico de situações em que duas ou mais pessoas interagem em tempo real, por exemplo, aplicações de vídeo-conferência e telefonia sobre IP. No tráfego isócrono, atraso e *jitter* altos comprometem a qualidade da informação transmitida e podem afetar a usabilidade da aplicação. É gerado por aplicações que necessitam de um baixo atraso, como difusão de rádio e TV em redes comutadas por pacotes. Por fim, o tráfego de missão-crítica é gerado por aplicações cuja funcionalidade fica comprometida caso o atraso e *jitter* ultrapassem certos limites, por exemplo, em aplicações de tele-comando, tele-supervisão e de automação industrial (Magalhães & Cardozo, 1999).

Tolerância a Atrasos	Tipo de Tráfego	Descrição
alta	Assíncrono	Tráfego não possui restrições temporais
	Síncrono	Dados são sensíveis ao tempo, mas flexíveis
	Interativo	Atrasos podem ser notados pelos usuários, mas não afetam a usabilidade e funcionalidade da aplicação
	Isócrono	Sensível ao tempo de uma forma que pode comprometer a usabilidade da aplicação
baixa	Missão-crítica	Atrasos podem comprometer a funcionalidade da aplicação

Tabela 3.2: Caracterização do tráfego segundo a sensibilidade ao atraso (Stardust, 1999b)

3.3.3 Arquiteturas para QoS

Em resumo, pode-se afirmar que o objetivo da introdução de QoS na Internet atual é fornecer algum nível de previsibilidade e controle, além do serviço de melhor esforço das redes IP (Stardust, 1999a), pois somente assim será possível atender às crescentes demandas das aplicações atuais e futuras. O tema de QoS na Internet tem sido objeto de intensas pesquisas nos últimos anos e várias abordagens foram propostas nesse sentido, sempre tendo em mente que o sucesso da Internet se explica, em grande parte, pela simplicidade

dos protocolos que funcionam sobre a rede, portanto assim ela deverá permanecer.

Destacam-se dois tipos básicos de arquiteturas para QoS na Internet (Stardust, 1999a; Zhao *et al.*, 2000):

- *Reserva de Recursos.* Os recursos da rede são atribuídos às aplicações segundo suas demandas de QoS e submetidos à política de gerenciamento de largura de banda. Esta é a abordagem empregada pela arquitetura de Serviços Integrados (*IntServ*).
- *Priorização.* O tráfego na rede é classificado de acordo com suas características de demanda e recebe os recursos segundo a política de gerenciamento vigente. As classes de tráfego mais exigentes recebem tratamento preferencial, abordagem adotada pela arquitetura de Serviços Diferenciados (*DiffServ*).

No primeiro caso, a *aplicação* receberá os recursos de que necessita antes da transmissão dos dados, sendo necessária a reserva de caminhos (*paths*) e recursos ao longo da rede. No segundo caso, o *tráfego* é dividido em classes e os pacotes pertencentes a uma certa classe são marcados diferentemente a fim de receber o serviço adequado (Xiao & Ni, 1999).

As arquiteturas de Serviços Integrados e Serviços Diferenciados serão discutidas com mais detalhes a seguir.

3.4 Serviços Integrados

A abordagem de Serviços Integrados, definida na RFC 1633 (Braden *et al.*, 1994), foi a primeira das soluções propostas para dar suporte a QoS na Internet, no âmbito da IETF. Tinha como objetivo inicial atender certas garantias exigidas por determinados tipos de tráfego, como transmissão de áudio e vídeo. A arquitetura *IntServ* visa fornecer, em uma rede comutada por pacotes, como a Internet, o serviço mais próximo possível da abstração de circuitos virtuais.

A idéia principal subjacente ao *IntServ* é a de reserva de recursos. Antes de iniciar a transmissão dos dados, as aplicações precisam encontrar um caminho até o receptor que satisfaça suas demandas de QoS, reservando, ao longo do mesmo, os recursos necessários (Xiao & Ni, 1999). Para tanto, as aplicações fazem uso do protocolo RSVP (*Resource Reservation Protocol*), um protocolo de controle e sinalização que atua na camada de rede, sendo responsável por reservar caminhos e recursos na sub-rede de comunicação.

O protocolo RSVP é considerado a mais elaborada e complexa das soluções para suporte a QoS e representa uma grande ruptura com a abordagem tradicional de melhor esforço das redes IP (Stardust, 1999a), na qual, antes de se enviar um pacote pela rede, não é feita nenhuma reserva de recursos de qualquer tipo, nem se procura encontrar um

caminho disponível. A abordagem de serviços integrados trabalha sobre fluxos individuais usando o protocolo RSVP para reservar recursos nos roteadores da rede, de fim-a-fim, com o objetivo de atender às demandas de QoS dos fluxos (Vasiliou, 2000). Melhoramentos recentes no RSVP têm procurado estendê-lo para trabalhar também com a noção de agregação de fluxos.

3.4.1 Classes de Serviço

A arquitetura de serviços integrados define duas classes de serviço, além do modelo de melhor esforço, tradicionalmente encontrado nas redes IP. São elas:

- *Serviço Garantido*[2]. Especificado na RFC 2212 (Shenker *et al.*, 1997), fornece um limite superior rígido para o atraso fim-a-fim, além de garantir a disponibilidade de largura de banda. Este serviço se destina a aplicações que possuem requisitos estritos de tempo real para funcionar, atingindo um alto nível de QoS na Internet.

- *Serviço de Carga Controlada*[3]. Especificado na RFC 2211 (Wroclawski, 1997), fornece um serviço equivalente ao modelo de melhor esforço em uma rede pouco utilizada, com quase nenhuma perda ou atraso. Em situações de sobrecarga, esta abordagem será capaz de compartilhar a largura de banda entre múltiplos fluxos, de uma maneira controlada, garantindo um serviço melhor que o usual. Entretanto, este modelo não oferece garantias de atraso máximo, apenas um limiar probabilístico, assim como também não pode assegurar que pacotes não serão perdidos.

3.4.2 Protocolo RSVP

O protocolo RSVP, especificado na RFC 2205 (Braden *et al.*, 1997), constitui-se no protocolo de controle e sinalização usado por aplicações para reserva de recursos ao longo da rede. Apresenta algumas características interessantes, analisadas a seguir (Stallings, 2002):

- *Protocolo de controle.* Embora pertença à camada de rede, o RSVP não é um protocolo de roteamento. Ele não transporta dados, apenas informações de controle e sinalização referentes à reserva de recursos ao longo de um caminho ou árvore de distribuição (*spanning tree*).

- *Unicast e Multicast.* As reservas feitas pelo RSVP podem ser para transmissão *unicast* ou *multicast*, tendo a capacidade de se adaptar dinamicamente a mudanças no número de membros do grupo e a modificações nas rotas.

[2] *Guaranteed Service*
[3] *Controlled-Load Service*

- *Unidirecional.* O RSVP faz reservas para fluxos unidirecionais, apenas. Assim, para trocas de dados entre dois sistemas, é preciso fazer reservas separadas nos dois sentidos.

- *Reserva iniciada pelo receptor.* O receptor é quem toma a iniciativa de fazer a reserva de recursos, especificando seus requisitos de largura de banda, atraso e *jitter*.

- *Soft State.* O protocolo RSVP usa a abordagem de *soft state*, ou seja, a informação sobre a reserva de recursos para os fluxos fica armazenada temporariamente nos roteadores. Dessa forma, é necessário renová-la periodicamente, tarefa que é responsabilidade dos *hosts* finais (*end hosts*).

- *Diferentes estilos de reserva.* Os usuários de um mesmo grupo *multicast* podem especificar como as reservas devem ser agregadas nos roteadores intermediários, a fim de economizar recursos da rede.

- *Operação transparente através de roteadores não-RSVP.* Como as reservas feitas são independentes do protocolo de roteamento, o tráfego pode atravessar roteadores não habilitados para RSVP. Neste caso, é usado o modelo de melhor esforço, não sendo possível assegurar a QoS nesse trecho do caminho.

- *Suporte para IPv4 e IPv6.* O RSVP pode explorar o campo *Type-of-Service* no cabeçalho do protocolo IPv4 e o campo *Flow Label* do IPv6.

A Figura 3.1 ilustra o funcionamento do protocolo RSVP: o emissor envia uma mensagem do tipo PATH a um ou mais receptores informando as características do tráfego(1). À medida que a mensagem PATH se propaga pela rede, cada roteador grava informações sobre o caminho na mensagem(2) (por exemplo, o roteador de origem da mesma). O receptor, ao receber a mensagem PATH(3), responde com uma mensagem do tipo RESV(4), requisitando recursos ao longo do caminho gravado na mensagem PATH, em ordem inversa, do receptor para o emissor. Quando um roteador intermediário recebe a mensagem RESV(5), ele verifica se pode atender a solicitação de recursos. Caso seja possível, ele faz a reserva de largura de banda e espaço em buffers e passa a mensagem RESV adiante; caso contrário, envia uma mensagem de erro ao receptor. Se o emissor recebe a mensagem RESV(6), isto significa que a reserva de recursos foi bem sucedida e é possível iniciar a transmissão dos pacotes(7).

Com o uso do protocolo RSVP, consegue-se atingir a maior QoS possível na Internet (Stardust, 1999a), pois ele permite fazer o gerenciamento em uma granulosidade bem fina, em nível de fluxo, conseguindo dar excelentes garantias de qualidade às aplicações. Entretanto, há um preço a se pagar por isso: o RSVP possui sérios problemas de gerenciamento e escalabilidade (Vasiliou, 2000). Cada roteador ao longo do caminho precisa dar suporte a RSVP para que se possa assegurar a QoS, sendo necessário manter as informações de estado e fazer o escalonamento e enfileiramento dos pacotes para cada fluxo.

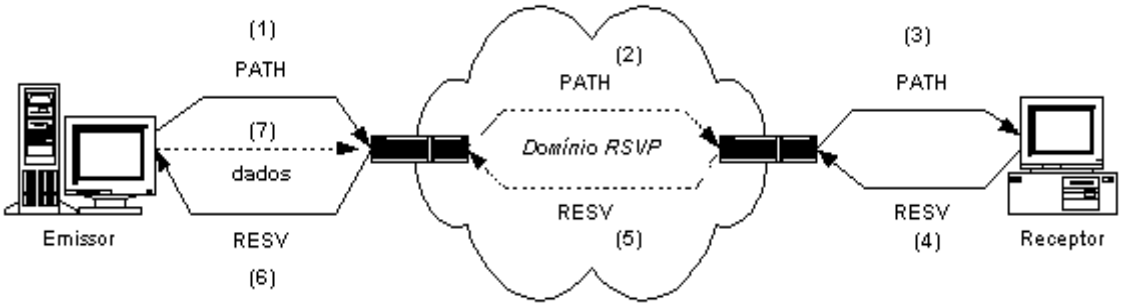

Figura 3.1: Funcionamento do protocolo RSVP

Na Internet, em que dezenas de milhares de fluxos concorrentes podem passar por um mesmo roteador, o gerenciamento dessas informações parece quase impossível, até por limitações de memória e capacidade de processamento nos roteadores. Com isso, vê-se que o RSVP introduz uma complexidade significativa no núcleo da Internet, o que representa uma ruptura com o seu modelo tradicional de serviços, que sempre procurou manter a rede simples e levar a complexidade para os *hosts* finais (Stardust, 1999a).

A abordagem de serviços integrados, portanto, aplica-se melhor a um ambiente de rede local, fornecendo uma QoS intra-domínio para aplicações que dela necessitem. No âmbito da Internet, a abordagem de serviços diferenciados, tratada a seguir, revela-se mais adequada.

3.5 Serviços Diferenciados

Os problemas de escalabilidade da arquitetura *IntServ* e a dificuldade de sua implantação em uma rede com as proporções da Internet motivaram os esforços para o desenvolvimento da arquitetura de Serviços Diferenciados, no âmbito da IETF, conforme proposto na RFC 2475 (Blake *et al.*, 1998).

Enquanto o *IntServ* realiza as reservas de recursos por fluxo, exigindo dos roteadores a manutenção de informações de estado para cada fluxo, fim-a-fim, a abordagem *DiffServ* baseia-se na idéia de agregação de fluxos em umas poucas classes de serviço (Magalhães & Cardozo, 1999). Dessa forma, o *DiffServ* fornece diferenciação de serviços local para grandes agregados de tráfego, enquanto o modelo *IntServ* dá garantias de performance fim-a-fim para fluxos individuais (Vasiliou, 2000).

A arquitetura de serviços diferenciados usa a classificação de pacotes como mecanismo para atingir a QoS. Para tanto, é redefinido o layout do octeto *Type-of-Service* do cabeçalho do protocolo IPv4 (ou o campo *Traffic Class* do IPv6), que passa a ser chamado de campo DS (*Differentiated Services*), conforme a RFC 2474 (Nichols *et al.*, 1998). Até o momento, somente os seis primeiros bits do campo DS são usados, recebendo a denominação de campo DSCP (*Differentiated Services Codepoint*), como mostrado na

Figura 3.2. Seu objetivo é especificar o tratamento dado ao encaminhamento de pacotes em cada roteador, o chamado PHB (*Per-hop Behavior*). Os PHBs são mecanismos de priorização que permitem a agregação de fluxos gerados por diferentes aplicações, definindo uma classe de serviço (Magalhães & Cardozo, 1999).

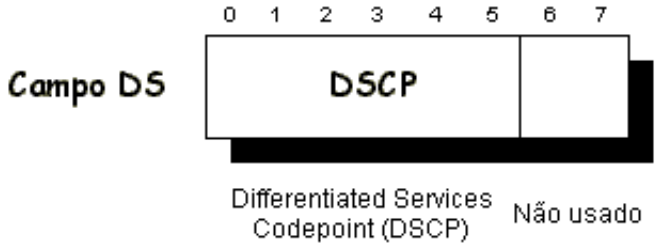

Figura 3.2: Layout do campo DS

O modelo *DiffServ*, portanto, opera através da marcação de pacotes de forma distinta, o que permite que os mesmos sejam tratados internamente à rede de maneira diferenciada, segundo a classe de serviço à qual pertencem. A abordagem empregada pelo *DiffServ* baseia-se em um esquema de prioridades relativas, ou seja, ele garante que o tráfego gerado por uma aplicação, com um certo nível de prioridade, receberá um tratamento melhor que o gerado por qualquer outra que possua uma prioridade inferior (Xiao & Ni, 1999).

A marcação dos pacotes se dá nos pontos de ingresso na rede, como os *hosts* finais e os roteadores de borda. Dessa forma, retém-se um dos princípios básicos do projeto da Internet, que é colocar a complexidade na fronteira da rede, ao contrário da abordagem *IntServ*, que exige complexidade fim-a-fim. O modelo *DiffServ* não necessita de um protocolo próprio (como no caso do RSVP, usado no *IntServ*), pois aqui se utiliza um campo do próprio datagrama IP. Tudo que um roteador precisa fazer é examinar o campo DSCP de cada pacote para determinar qual o tratamento a ser dado ao mesmo. Assim, pacotes marcados da mesma forma recebem tratamento igual. Não é necessário manter nenhum tipo de informação relacionada a fluxos, pois os roteadores só precisam ser capazes de distinguir entre um certo número de classes de serviço pré-definidas.

3.5.1 Classes de Serviço

Embora outras alternativas sejam possíveis, existem atualmente duas classes de serviço principais definidas na arquitetura de serviços diferenciados, que são abordadas a seguir: Encaminhamento Expresso e Encaminhamento Garantido (Kilkki, 1999).

Encaminhamento Expresso

O serviço denominado de Encaminhamento Expresso[4] (*Expedited Forwarding* — EF), definido na RFC 2598 (Jacobson *et al.*, 1999), permite a adaptação do modelo de serviço garantido da arquitetura *IntServ* à arquitetura de serviços diferenciados. Ele oferece garantias de QoS absoluta, com baixos valores de perda, atraso e *jitter*, fornecendo o equivalente a uma linha privada virtual com largura de banda fixa entre dois *hosts*. É indicado para aplicações de telefonia sobre IP, videoconferência e para a criação de linhas dedicadas em redes privadas virtuais (VPNs).

Sua vantagem sobre o serviço equivalente na arquitetura *IntServ* está na simplicidade de implementação, pois não é necessário manter nos roteadores nenhuma informação relativa a fluxos. Os equipamentos que implementam este comportamento (PHB) simplesmente devem escalonar o tráfego de maneira a manter descongestionadas as filas de saída, a fim de que o tráfego passe o menor tempo possível no equipamento (Magalhães & Cardozo, 1999). Em geral, os pacotes pertencentes a esta classe são colocados em uma fila de maior prioridade que a do tráfego de melhor esforço e são os primeiros a serem encaminhados em qualquer situação. Em relação à política de descarte, este serviço evita a todo custo descartar os pacotes para o tráfego em conformidade com o perfil contratado. Já para o tráfego sem conformidade, ele é implacável: os pacotes simplesmente são descartados. Uma conseqüência disso é que os usuários não podem exceder a taxa de pico solicitada; caso contrário, o tráfego em excesso será descartado. Em contrapartida, a arquitetura *DiffServ* garante que a largura de banda contratada estará disponível quando o tráfego for enviado (Xiao & Ni, 1999).

Encaminhamento Garantido

A classe de serviço Encaminhamento Garantido[5] (*Assured Forwarding* — AF), definida na RFC 2597 (Heinanen *et al.*, 1999), destina-se a aplicações que demandem da rede um serviço mais confiável que aquele de melhor esforço, mas sem todas as garantias de QoS dadas pelo encaminhamento expresso. Este serviço não oferece limites superiores para o atraso e *jitter*, mas garante um tratamento preferencial ao tráfego que dele se utilize. Ele é indicado quando se deseja obter da rede um serviço de entrega de pacotes mais consistente, por exemplo, a fim de oferecer uma melhor QoS a agregados de tráfego consistindo de rajadas de curta duração com destinos diferentes (o tráfego na Web) (Stoika & Zhang, 1998).

O princípio aqui é a divisão do tráfego em N classes, cada uma com alguns níveis de precedência de descarte (M). A especificação atual define $N = 4$ e $M = 3$ (Figura 3.3), embora, em uma situação real, nem todas elas possam vir a ser necessárias. O serviço

[4]Usa-se também a denominação *Premium Service*.
[5]Usa-se também a denominação *Assured Service*.

fornecido por uma certa classe independe do serviço das demais classes, sendo função apenas dos recursos alocados para cada classe pelo sistema (Magalhães & Cardozo, 1999).

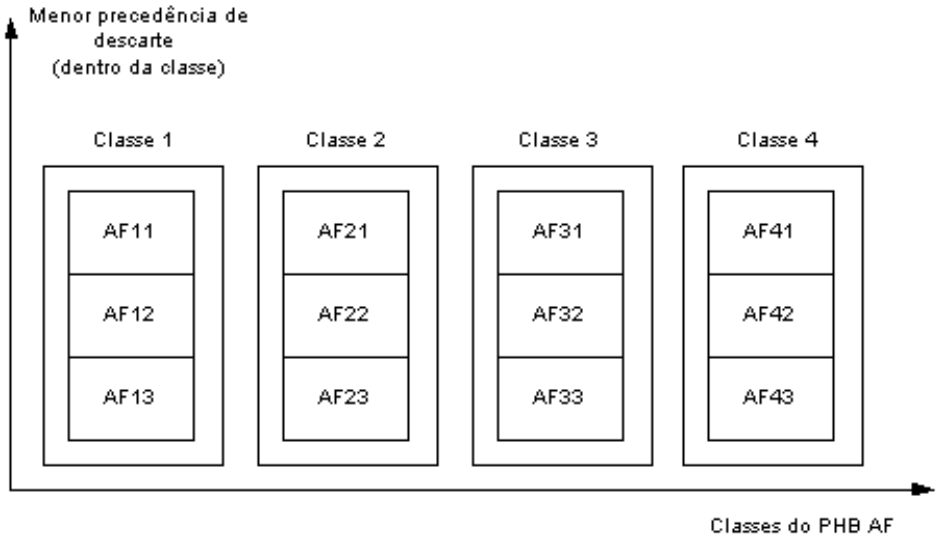

Figura 3.3: Classes do serviço de Encaminhamento Garantido (Kilkki, 1999)

Um usuário pode contratar de um provedor um dos quatro serviços de encaminhamento diferenciado, cada um com três níveis de prioridade de descarte. Em situações de sobrecarga, o tráfego de uma classe superior tem menor probabilidade de sofrer congestionamento que o tráfego de uma classe inferior. O mecanismo de diferenciação aqui utilizado se baseia na prioridade de descarte: quando este for inevitável, primeiro se descartam os pacotes pertencentes ao serviço de melhor esforço e, só então, passa-se para os pacotes associados ao serviço de encaminhamento garantido, segundo sua classe e nível de precedência na classe. Portanto, os pacotes pertencentes ao serviço AF são os últimos a serem descartados em situações de congestionamento.

É possível estabelecer uma relação entre os serviços das arquiteturas de serviços integrados e diferenciados. Em geral, pode-se mapear o Serviço Garantido (*IntServ*) para o Serviço de Encaminhamento Expresso (*DiffServ*) e o Serviço de Carga Controlada (*IntServ*) para o Serviço de Encaminhamento Garantido (*DiffServ*). Esta possibilidade de mapeamento é especialmente interessante em situações em que se utiliza as duas arquiteturas em conjunto para a provisão de uma QoS fim-a-fim (Seção 3.5.4).

3.5.2 *Service Level Agreements*

A fim de que um usuário possa receber serviços diferenciados de um provedor, é necessário que se estabeleça entre as partes um "acordo de serviço", conhecido como *Service Level Agreement* (SLA). Um SLA estabelece os critérios das políticas de QoS e define o

perfil esperado do tráfego gerado pelas aplicações. Em outras palavras, define as classes de serviço contratadas e a quantidade de tráfego permitida em cada classe.

Ao ser enviado um tráfego, o domínio de origem tem a responsabilidade de policiá-lo (*policing*) e suavizá-lo nos pontos de saída (*egress points*), pois um tráfego fora do perfil contratado não receberá nenhum tipo de garantia de QoS ao chegar ao próximo ponto de ingresso (*ingress point*), em outro domínio. Quando um pacote deixa um domínio e segue para outro, às vezes pode ser necessário remarcar o seu campo DS, como resultado de um SLA estabelecido entre os dois domínios, embora o ideal seja que o tráfego experimente o mesmo nível de QoS ao longo de todo o percurso, da origem até o destino, o que nem sempre é possível.

Os critérios empregados para a aplicação das políticas de QoS podem ser: data e hora, endereços de origem e destino, números de portas ou qualquer outra informação que possa ser extraída do conteúdo do tráfego, inclusive a contida nos cabeçalhos (Stardust, 1999b). Além desses aspectos, um SLA também pode especificar: procedimentos de tarifação e cobrança, serviços de criptografia e autenticação, procedimentos de renegociação dos parâmetros do SLA, ações a serem tomadas para o tráfego fora de perfil, entre outros (Vasiliou, 2000).

Um SLA pode ser estático, no sentido de que ele é negociado em bases mensais ou anuais, por exemplo, ou pode ser dinâmico, caso em que se faz necessário o uso de um protocolo de sinalização (como o RSVP) para a requisição de serviços de QoS sob demanda.

3.5.3 Protocolo MPLS

Uma abordagem alternativa aos serviços diferenciados, mas que tem recebido bastante atenção ultimamente, graças a sua simplicidade e eficiência, é o protocolo MPLS (*Multi-Protocol Label Switching*), especificado na RFC 3031 (Rosen *et al.*, 2001). Assim como o *DiffServ*, o MPLS marca os pacotes nos pontos de ingresso na rede e os desmarca nos pontos de saída, contudo, ao contrário do *DiffServ*, em que a marcação é apenas uma forma de atribuir prioridade aos pacotes dentro dos roteadores, o MPLS a utiliza efetivamente para fazer o roteamento, determinando qual o próximo *hop* a ser dado conforme o rótulo (*label*) colocado no pacote. O protocolo MPLS funciona integralmente nos roteadores, sem nenhum componente nos sistemas finais.

O rótulo de um pacote MPLS determina completamente qual caminho o mesmo deverá seguir na rede, permitindo se estabelecer canais (*pipes*) com largura de banda fixa, nos moldes dos circuitos virtuais de uma rede ATM ou *Frame Relay*. Como o MPLS opera entre as camadas de rede e enlace, ele pode funcionar sobre diferentes tipos de protocolos além do IP (Vasiliou, 2000).

A Figura 3.4 mostra o formato de um cabeçalho MPLS, o qual possui um rótulo de

20 bits. É este rótulo que será examinado pelos roteadores habilitados para MPLS (*Label Switching Routers* — LSRs), funcionando como um índice para uma tabela que especifica o próximo *hop* a ser dado e um novo rótulo para o pacote. Há também um campo de três bits reservado para fins experimentais, um bit usado para permitir o aninhamento de rotas MPLS (*Stack Flag*) e um campo de 8 bits para estabelecer um tempo de vida para o pacote (*Time-to-Live* — TTL).

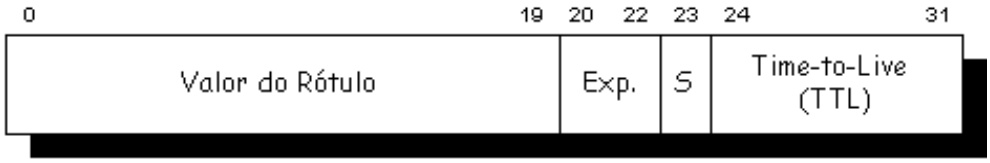

Figura 3.4: Layout do cabeçalho do protocolo MPLS

Um aspecto importante na implantação do MPLS está na forma de gerenciamento e distribuição dos rótulos entre os roteadores MPLS, a fim de garantir um significado comum para todos eles.

3.5.4 Comparação *IntServ* vs. *DiffServ*

As arquiteturas de serviços integrados e diferenciados não devem ser consideradas como competidoras, ao contrário, elas são complementares e podem ser usadas em conjunto, explorando-se as melhores características de uma e de outra. Entretanto, existem diferenças significativas entre as duas abordagens, as quais são comentadas a seguir.

Em primeiro lugar, destaca-se o nível de granulosidade em que cada mecanismo trabalha: o modelo *IntServ* tem seu foco em fluxos individuais fim-a-fim, entre uma origem e um destino; já o modelo *DiffServ* lida com o conceito de agregação de fluxos, os quais constituem as classes de serviço. Daí, tem-se que o gerenciamento das informações de estado associadas aos fluxos é muito mais complexo no *IntServ*, pois, em uma rede como a Internet, podem-se ter centenas de milhares de fluxos concorrentes, o que ultrapassa a capacidade de memória e processamento dos roteadores atuais. No futuro, com links da ordem de gigabits ou terabits, esse gerenciamento se tornará ainda mais difícil (Dovrolis & Ramanathan, 1999). No modelo *DiffServ*, a quantidade de informações de estado é proporcional ao número de classes de serviço definidas, que é muito inferior ao número de fluxos (Xiao & Ni, 1999). Por essa razão, considera-se que a arquitetura *IntServ* é mais adequada para a provisão de uma QoS intra-domínio, em redes locais e a *DiffServ*, com uma melhor escalabilidade, presta-se mais a redes do porte da Internet (contudo, é possível usar as duas abordagens em conjunto, como será comentado).

A arquitetura *IntServ* representa uma ruptura com a abordagem tradicionalmente empregada nas redes IP, pois ela leva complexidade para o núcleo da rede; já a arquitetura

DiffServ é mais natural, pois coloca a complexidade nos pontos externos da rede e deixa os roteadores simples, preservando as diretrizes do projeto original da Internet. O *IntServ* necessita de um protocolo de sinalização como o RSVP para funcionar, o qual precisa incluir informações de estado nos roteadores ao longo do caminho; o *DiffServ* apenas faz uma redefinição de um campo do cabeçalho IP, o que agiliza sua implantação. Portanto, a coordenação da QoS, no *IntServ*, é feita fim-a-fim e, no *DiffServ*, as decisões são tomadas localmente, a cada *hop*.

A classificação dos pacotes no *IntServ* é feita a partir de vários campos, como os endereços de origem e destino, números de portas e o protocolo utilizado, individualmente para cada fluxo; já o *DiffServ* usa apenas o campo DS do protocolo IP para este fim, sobre um conjunto de fluxos relacionados (agregação).

O gerenciamento da rede, nos serviços integrados, é feito de forma similar a redes comutadas por circuito, como a ATM; nos serviços diferenciados, o mesmo é semelhante ao já encontrado em redes IP.

O modelo *IntServ* está mais voltado para as demandas de serviço de cada aplicação, procurando satisfazê-las fim-a-fim e, por causa disso, exige uma reserva de recursos antes do início de qualquer transmissão de dados. Em contrapartida, o modelo *DiffServ* visualiza mais as propriedades do tráfego gerado, usando um mecanismo de priorização para classificá-lo segundo suas características de demanda de QoS. É responsabilidade das aplicações (ou de algum roteador de borda) marcar seus pacotes adequadamente a fim de que recebam o serviço de que precisam.

Por outro lado, a QoS fornecida pela arquitetura *IntServ* é a maior possível em uma rede baseada nos protocolos TCP/IP, sendo oferecidas garantias deterministas de perda, atraso e *jitter*. Já a arquitetura *DiffServ*, embora possa fornecer QoS absoluta, baseia-se mais na idéia de diferenciação de serviços relativa entre as classes.

Cooperação entre as Arquiteturas

Para concluir esta seção, cabem alguns comentários acerca da utilização dessas arquiteturas de provisão de QoS em conjunto, o que parece bastante promissor. Na prática, é improvável que qualquer uma das soluções apresentadas (RSVP, *DiffServ*, MPLS) seja utilizada independentemente para fornecer QoS fim-a-fim. O que geralmente ocorre é a combinação dessas arquiteturas no caminho entre o emissor e o receptor a fim de que se possa extrair o melhor de cada uma delas (Stardust, 1999a).

A recomendação geral é que o modelo *IntServ* (RSVP) pode ser melhor empregado nos pontos de ingresso na rede, com uma alta granulosidade, a fim de especificar as demandas de QoS das aplicações em termos de largura de banda, perda de pacotes, atraso e *jitter*. Os roteadores de borda nesses pontos poderão, então, mapear essas demandas para as classes de serviço do *DiffServ*, através da marcação adequada dos pacotes. Nesta solução híbrida,

o RSVP se beneficiaria dos aspectos de agregação da arquitetura *DiffServ* e esta, por sua vez, se utilizaria do mecanismo de sinalização de QoS do RSVP, a fim de proporcionar uma QoS absoluta fim-a-fim (Magalhães & Cardozo, 1999). Assim, o *DiffServ* teria seu melhor emprego no núcleo (*backbone*) da Internet, por ser uma solução mais leve e de melhor escalabilidade, enquanto o RSVP ficaria restrito às bordas da mesma. É para esta direção que se dirigem os estudos atuais dentro do grupo de trabalho *DiffServ*, na IETF (Stardust, 1999a).

Para finalizar, outra possibilidade é o uso de MPLS com *DiffServ*, o que não parece muito complicado, pela similaridade das duas abordagens. A solução seria mapear um valor particular do campo DS (uma classe de serviço) para um rótulo MPLS, tendo-se o cuidado de reservar recursos em cada roteador MPLS a fim de emular as prioridades do *DiffServ*. O RSVP também teria seu lugar neste cenário, reservando recursos para uso dedicado ao longo dos caminhos estabelecidos pelo MPLS (Vasiliou, 2000).

3.6 Serviços Diferenciados em Nível de Aplicação

Até este ponto, foram estudadas as formas de garantir QoS em nível de rede, particularmente as abordagens de serviços diferenciados e integrados. Contudo, como já foi comentado, qualquer esforço no sentido de fornecer QoS terá que ter a cooperação de todos os elementos do sistema, de cima a baixo, de fim a fim. Sendo assim, os *hosts* finais constituem-se em elementos essenciais neste cenário, uma vez que são eles, em última análise, que vão atender os usuários.

A discussão de QoS em nível de rede, nas seções anteriores, não foi sem motivo, uma vez que se pode aproveitar parte do conhecimento já adquirido e testado na área, desta feita na construção de servidores web. Pode-se perceber que um servidor web não preparado para oferecer QoS anulará quaisquer esforços que tenham sido empreendidos pela rede nesse sentido, pois ele tratará todas as solicitações que receber por igual, ignorando a prioridade relativa das mesmas. Infelizmente, a maior parte dos servidores atuais ainda trata todas as solicitações uniformemente, sem nenhum tipo de diferenciação, segundo uma disciplina FCFS (*First-Come First-Served*), o que muitas vezes pode transformá-los no ponto de estrangulamento do sistema. Por essa razão, a provisão de QoS na Web, de uma maneira ampla, não será possível se não se considerar o servidor web como um elemento essencial nesta cadeia, dotando-o dos mecanismos necessários para tal.

Nesta seção, serão examinados alguns trabalhos que se preocupam com o fornecimento de QoS em nível de aplicação, particularmente no contexto de servidores web, assunto que é o objetivo do presente projeto de pesquisa. Esse é um tema dos mais relevantes, na medida em que a presença das empresas na Web cada vez mais se confunde com sua imagem no mundo real. Não se está tratando aqui apenas de aplicações multimídia, como

o tema de QoS pode levar a pensar, mas da utilização da Web como um meio para a condução de negócios, colocando-se o servidor web como o elemento central deste cenário, do qual serão exigidos requisitos de qualidade de serviço, disponibilidade, confiabilidade e segurança.

Serviços Diferenciados em Nível de Aplicação para Servidores Web

Os esforços atuais no sentido de fornecer múltiplos níveis de serviço na Internet concentram-se principalmente na rede e nos sistemas operacionais. Embora essas abordagens tendam a fornecer os resultados mais expressivos, na prática sua implantação em larga escala é difícil, pela impossibilidade de se fazer a atualização dos roteadores e sistemas operacionais por toda a Internet.

Eggert & Heidemann (1999) propõem mecanismos do lado servidor, tão somente em nível de aplicação, que, segundo os autores, mostram benefícios significativos, apesar de sua aparente simplicidade. A idéia principal é dividir as solicitações que chegam ao servidor em *foreground* e *background*, onde as solicitações de *background* recebem um serviço de "menor esforço" (*less effort*), ou seja, elas só serão processadas e transmitidas aos clientes caso haja recursos ociosos disponíveis no servidor. Se não houver, poderão ser postergadas indefinidamente ou até mesmo descartadas.

Dessa forma, criam-se, na prática, duas classes de serviço: uma classe de *foreground* e outra de *background*. Os autores apontam algumas situações em que esta solução pode ser aplicada: a primeira é no atendimento ao tráfego especulativo na Web, resultante de operações de cache antecipado (*prefetching*) ou de *pushes* feitos pelo servidor. Este tipo de tráfego, argumentam, não é primordial e pode ser descartado se necessário. Outra possibilidade é atribuir diferentes prioridades às requisições segundo o tipo de objeto requisitado, por exemplo, o código HTML poderia ter prioridade sobre imagens e *applets*. Finalmente, é sugerido que a QoS diferenciada poderia ser resultado de políticas externas ao servidor, no caso favorecendo-se usuários pagantes em detrimento dos não-pagantes.

Esses mecanismos foram implementados sobre o código do servidor Apache, usando-se estratégias como a limitação do número de processos em *background*, a diminuição da prioridade de processos em nível de sistema operacional e da sua taxa de transferência na rede. São mostrados resultados experimentais que validam os mecanismos implementados, sendo feita também uma comparação com o desempenho do servidor Apache não modificado.

Provisão de Diferentes Níveis de Serviço em *Web Hosting*

O trabalho de Almeida *et al.* (1998) está voltado à área de *web hosting*, ou seja, o caso de um provedor de Internet que possui vastos recursos e abriga sites de diferentes

empresas, instituições e indivíduos. A diferenciação de QoS se dá entre os diferentes sites hospedados no provedor. Evidentemente, nesta situação, a existência de diferentes níveis de serviço é mais do que necessária, pois as empresas esperam que as requisições dos seus clientes sejam atendidas com uma qualidade compatível com a quantia que foi paga pelos serviços de hospedagem do site.

O mecanismo usado para a provisão de QoS é o escalonamento baseado em prioridades, tanto no modo de usuário quanto de kernel, diferentemente do trabalho de Eggert & Heidemann (1999), que só emprega mecanismos em nível de aplicação. São oferecidos dois níveis de QoS: alta prioridade e baixa prioridade. A primeira classe de serviço é para clientes que pagam uma taxa de hospedagem e a segunda, para aqueles que a tem de graça.

O primeiro passo consiste em classificar as requisições dos clientes em categorias, algo que normalmente não é feito pelos servidores web e que demandou a modificação do código do servidor Apache. No nível de usuário, um processo escalonador decide a ordem em que as requisições serão atendidas, limitando também o número de processos em cada categoria. Num segundo momento, as prioridades das requisições são mapeadas nas prioridades dos processos HTTP que as estão atendendo, o que requereu alterações no kernel do Linux. Foram implementadas duas políticas de escalonamento: uma conservadora (*work-conserving*), que permite que processos de baixa prioridade executem como processos de alta prioridade, caso haja recursos disponíveis na categoria superior e uma não-conservadora (*non-work-conserving*), que não permite a migração de processos de uma categoria para outra.

Os resultados obtidos mostram que restringir o número de processos concorrentes revela-se um mecanismo eficiente para a obtenção de QoS diferenciada, com a política não-conservadora apresentando os melhores resultados.

Suporte a Qualidade de Serviço em Servidores HTTP

Os trabalhos de Eggert e de Almeida realizam a diferenciação de serviços em alto nível, considerando apenas processos de alta e baixa prioridade. Já Pandey *et al.* (1998) propõem uma arquitetura de serviços web que permite uma granulosidade mais fina no escalonamento das requisições. Nela, é possível personalizar como um servidor HTTP deve responder às requisições dos clientes, através da atribuição de prioridades e da alocação dos recursos do servidor às requisições de páginas.

As páginas web são modeladas como objetos e as requisições às mesmas são consideradas invocações de métodos. O servidor, portanto, torna-se um sistema que gerencia a execução de várias invocações de métodos, dentro das restrições de QoS vigentes. É especificada uma linguagem, denominada WebQoSL, que permite ao administrador do sistema informar detalhes de alocação dos recursos do sistema às páginas, determinar a

disponibilidade de conjuntos de páginas e especificar garantias quanto à taxa de transferência das mesmas.

A arquitetura implementada consiste de cinco servidores web distribuídos e um *daemon* controlador de QoS. Quando um servidor recebe uma requisição, ele envia uma mensagem ao *daemon* perguntando qual a ação a ser tomada. O *daemon* pode responder de três formas: processar a requisição, negar o processamento ou redirecionar a requisição. Essas decisões são tomadas a partir das informações mantidas pelo *daemon* a respeito da utilização dos recursos do sistema, com dois objetivos em mente: satisfazer as restrições de QoS e otimizar o uso dos recursos do sistema. Vários experimentos foram realizados a fim de validar a arquitetura proposta.

Provisão de Serviços Diferenciados a partir de um Servidor Web

Chen & Mohapatra (1999) apresentam uma solução para um servidor web com diferenciação de serviços que consiste de um servidor web distribuído com roteamento de tarefas centralizado. O servidor é formado por quatro componentes principais: um iniciador de tarefas, um escalonador, um conjunto de servidores de tarefas e o canal de comunicação. As requisições são recebidas pelo iniciador, que pode aceitá-las ou rejeitá-las, caso a capacidade do sistema tenha sido excedida. Uma vez aceita, a requisição recebe um nível de prioridade, dado pelo escalonador e é atendida por um dos servidores de tarefas. O canal de comunicação modela a carga na rede, que é um dos parâmetros levados em conta para o escalonamento.

A abordagem aqui empregada não é a experimentação, como nos trabalhos anteriores. Neste caso, foi construído um modelo de rede de filas para o sistema, o qual é resolvido por simulação, utilizando-se, como entrada, *traces* de acessos a servidores web comerciais. Seus resultados demonstraram, mais uma vez, a eficácia do escalonamento baseado em prioridades na diferenciação de serviços, pois a degradação do desempenho das tarefas de mais alta prioridade aconteceu a um nível de utilização do servidor bem maior do que no caso das tarefas de mais baixa prioridade. Além disso, as requisições de alta prioridade experimentaram um baixo atraso mesmo quando o sistema se aproximou da utilização completa.

Análise das Arquiteturas

Os trabalhos apresentados representam abordagens diferentes à questão de como fornecer QoS diferenciada em servidores web. Embora apresentem algumas similaridades, cada um tem mecanismos e detalhes de implementação próprios.

O trabalho de Eggert & Heidemann (1999) mostra claramente que é possível fornecer serviços diferenciados em servidores web empregando-se apenas técnicas em nível de

aplicação e foi a motivação inicial para este projeto de pesquisa. Almeida *et al.* (1998) apresentam uma solução mais sofisticada, com o uso de mecanismos para diferenciação de serviços em nível de kernel do sistema operacional, obtendo bons resultados. O trabalho de Pandey *et al.* (1998) propõe uma abordagem com o uso de uma sintonia fina para a provisão de QoS diferenciada, definindo até mesmo uma linguagem de especificação para este fim. Finalmente, Chen & Mohapatra (1999) desenvolvem um modelo em rede de filas para um servidor web diferenciado e o validam através de simulação, ao contrário dos outros trabalhos que se baseiam em experimentação.

Além dos artigos comentados nesta seção, vários outros abordando a diferenciação de serviços em servidores web foram objeto de estudo nesta revisão bibliográfica, como os trabalhos de Abdelzaher & Bhatti (1999), Banga *et al.* (1999), Bhatti & Friedrich (1999), Cardellini *et al.* (2001), Casalicchio & Colajanni (2000), Dovrolis & Stiliadis (1999), Kanodia & Knightly (2000), Rao & Ramamurthy (2001) e Vasiliou & Lutfiyya (2000). De uma maneira geral, pôde-se notar que os trabalhos estudados utilizam apenas mecanismos do lado servidor com o intuito de realizar a diferenciação de serviços. Nenhum deles contempla o mapeamento entre a QoS no servidor e os mecanismos em nível de rede, abordados neste capítulo.

3.7 Considerações Finais

Este capítulo apresentou uma visão geral da área de qualidade de serviço, abordando seus conceitos básicos. Deu-se especial atenção à possibilidade de utilização de QoS na Internet, a fim de que a mesma possa dar suporte a novos tipos de aplicações, oferecendo níveis de serviço superiores aos do modelo atual de melhor esforço. Foram destacadas as arquiteturas de serviços integrados e diferenciados, com ênfase nesta última, concluindo-se com uma revisão de alguns trabalhos na área de diferenciação de serviços em servidores web.

O próximo capítulo detalha uma das principais propostas desta tese: uma arquitetura para um servidor web com suporte à diferenciação de serviços.

Referências Bibliográficas

Abdelzaher, T.; Bhatti, N. (1999). Web content adaptation to improve server overload behavior. *Proceedings of the International World Wide Web Conference*, p. 92–103.

Almeida, J.; Dabu, M.; Manikutty, A.; Cao, P. (1998). Providing differentiated levels of service in web content hosting. *Proceedings of the 1998 SIGMETRICS Workshop on Internet Server Performance.*

Banga, G.; Druschel, P.; Mogul, J. C. (1999). Resource containers: A new facility for resource management in server systems. *Operating Systems Design and Implementation*, p. 45–58.

Bhatti, N.; Friedrich, R. (1999). Web server support for tiered services. *IEEE Network*, v.13, n.5, p.64–71.

Blake, S.; Black, D.; Carlson, M.; Davies, E.; Wang, Z.; Weiss, W. (1998). *An Architecture for Differentiated Services.* RFC 2475, IETF.

Braden, R.; Clark, D.; Shenker, S. (1994). *Integrated Services in the Internet Architecture.* RFC 1633, IETF.

Braden, R.; Zhang, L.; Berson, S.; Herzog, S.; Jamin, S. (1997). *Resource ReSerVation Protocol (RSVP) — Version 1 Functional Specification.* RFC 2205, IETF.

Cardellini, V.; Casalicchio, E.; Colajanni, M.; Mambelli, M. (2001). Web switch support for differentiated services. *ACM Performance Evaluation Review*, v.29, n.2, p.14–19.

Casalicchio, E.; Colajanni, M. (2000). Scalable web cluster with static and dynamic contents. *Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER '00).*

Chen, X.; Mohapatra, P. (1999). Providing differentiated services from an Internet server. *Proceedings of the IEEE International Conference on Computer Communications and Networks*, p. 214–217.

Dovrolis, C.; Ramanathan, P. (1999). A case for relative differentiated services and the proportional differentiation model. *IEEE Network*, v.13, n.5.

Dovrolis, C.; Stiliadis, D. (1999). Relative differentiated services in the Internet: Issues and mechanisms. *Proceedings of the International Conference on Measurement and Modeling of Computer Systems*, p. 204–5.

Eggert, L.; Heidemann, J. (1999). Application-level differentiated services for web servers. *World Wide Web Journal*, v.3, n.2, p.133–42.

Heinanen, J.; Baker, F.; Weiss, W.; Wroclawski, J. (1999). *Assured Forwarding PHB Group*. RFC 2597, IETF.

Jacobson, V.; Nichols, K.; Poduri, K. (1999). *An Expedited Forwarding PHB*. RFC 2598, IETF.

Kanodia, V.; Knightly, E. W. (2000). Multi-class latency-bounded web services. *Proceedings of the 8th IEEE International Workshop on Quality of Service (IWQoS '00)*, p. 231–239.

Kilkki, K. (1999). *Differentiated Services for the Internet*. Macmillan Publishing.

Magalhães, M. F.; Cardozo, E. (1999). Qualidade de serviço na Internet. Relatório técnico, UNICAMP/FEEC/DCA, Campinas, SP.

Nichols, K.; Blake, S.; Baker, F.; Black, D. (1998). *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*. RFC 2474, IETF.

Pandey, R.; Barnes, J. F.; Olsson, R. (1998). Supporting quality of service in HTTP servers. *17th SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, p. 247–56.

Rao, G.; Ramamurthy, B. (2001). DiffServer: Application level differentiated services for web servers. *Proceedings of the IEEE International Conference on Communications*, p. 1633–7.

Rosen, E.; Viswanathan, A.; Callon, R. (2001). *Multiprotocol Label Switching Architecture*. RFC 3031, IETF.

Shenker, S.; Partridge, C.; Guerin, R. (1997). *Specification of Guaranteed Quality of Service*. RFC 2212, IETF.

Stallings, W. (2002). *High-Speed Networks and Internets: Performance and Quality of Service*. Prentice Hall, 2. edição.

Stardust (1999a). White Paper — QoS protocols & architectures. Disponível em `http://www.qosforum.com`.

Stardust (1999b). White Paper — The need for QoS. Disponível em `http://www.qosforum.com`.

Stoika, I.; Zhang, H. (1998). LIRA: An approach for service differentiation in the Internet. *Proceedings of NOSSDAV*.

Vasiliou, N. (2000). Overview of Internet QoS and web server QoS. Relatório técnico, Univ. of Western Ontario, Canadá.

Vasiliou, N.; Lutfiyya, H. (2000). Providing a differentiated quality of service in a World Wide Web server. *ACM SIGMETRICS Performance Evaluation Review*, v.28, n.2, p.22–28.

Wroclawski, J. (1997). *Specification of the Controlled-Load Network Element Service*. RFC 2211, IETF.

Xiao, X.; Ni, L. M. (1999). Internet QoS: a big picture. *IEEE Network*, v.13, n.2, p.8–18.

Zhao, W.; Olshefski, D.; Schulzrinne, H. (2000). Internet quality of service: an overview. Relatório Técnico CUCS-003-00, Columbia University.