ANÁLISE DE QUALIDADE DE SERVIÇO PARA APLICAÇÕES DE TEMPO REAL UTILIZANDO RESERVA DE RECURSOS NA INTERNET

Ana Luísa Pereira Schmidt

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Profa. Rosa Maria Meri Leão, Dr.

Profa. Rosângela Fernandes Coelho, Dr.

Prof. Aloysio de Castro Pinto Pedroza, Dr.

RIO DE JANEIRO, RJ - BRASIL

SETEMBRO DE 1999

SCHMIDT, ANA LUÍSA PEREIRA

Análise de Qualidade de Serviço para Aplicações de Tempo Real utilizando Reserva de Recursos na Internet [Rio de Janeiro] 1999

XII, 115 p. 29,7 cm (COPPE/ UFRJ, M.Sc., Engenharia de Sistemas e Computação, 1999)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Qualidade de Serviço 2. Protocolo RSVP 3. Controle de Tráfego 4. Parâmetros de Desempenho

I. COPPE/UFRJ II.Título (série)

Dedico aos pais e sogros,

pelo carinho e compreensão,

a meu esposo, pelo apoio de

todas as horas e a nosso filho,

João Henrique.

Agradecimentos

Nesse espaço, eu gostaria de expressar a minha profunda gratidão aos meus familiares: meus avós, minha mãe, meu pai, eternos amigos, e meus irmãos que sempre me viram como um exemplo a ser seguido.

Agradeço a meu esposo, por manter comigo uma relação baseada na amizade, compreensão e respeito. Por todos os momentos de apoio, troca de idéias e dicas que me ajudaram na realização deste trabalho.

Meus sinceros agradecimentos a meus sogros, que souberam compreender nossas ausências.

Agradeço à Jane pela dedicação com que cuida de meu bebê. Certamente sem sua ajuda, meu esforço para a realização deste trabalho seria incomparavelmente maior. Também gostaria de agradecer às minhas meninas Mylla e Filó. Suas existências enchem a nossa casa de carinho e amizade.

Agradeço a meu filho João Henrique, que me ensinou a ver a Humanidade de outra forma. Aprendi com ele como nós, seres humanos, somos frágeis e precisamos uns dos outros para crescer. Esse é o verdadeiro sentido da vida.

Também gostaria de destacar a atuação de minha orientadora. Ao final desta jornada percebo que ela sempre esteve aberta ao diálogo, respeitando minhas idéias e compreendendo minhas decisões. Agradeço pela dedicação a mim empregada para que todo o trabalho chegasse ao final.

Agradeço aos professores da COPPE, pelo incentivo e torcida para que eu realizasse todos os meus sonhos. Agradeço também pela permissão no uso de seus laboratórios.

Agradeço aos funcionários da COPPE e do NCE pelo apoio para a realização dos testes.

A todos os colegas de curso que diretamente me incentivaram para a realização desta tese.

Agradeço ao Governo do Brasil, por patrocinar meus estudos desde o Segundo Grau. Tenho certeza que o privilégio que eu obtive, com a educação de primeira linha, deve ser levado a todos os cidadães brasileiros, sem distinção. No futuro, quando a possibilidade dos estudos não mais for realmente um privilégio, terei a certeza de estar vivendo num país melhor. Lutarei por isso.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.).

Análise de Qualidade de Serviço para aplicações de tempo real utilizando reserva de recursos na Internet

Ana Luísa Pereira Schmidt

setembro/1999

Orientador: Rosa Maria Meri Leão

Programa: Engenharia de Sistemas e Computação

A utilização de aplicações de tempo real, como voz e vídeo, sobre a Internet, mostrou a importância de um controle mais rigoroso de parâmetros de desempenho como retardo, jitter e perda de pacotes. Nesse sentido, foi criada uma extensão chamada de serviços integrados ao modelo ora implementado pela Internet. Nessa extensão, uma série de parâmetros, ditos de Qualidade de Serviço (QoS), foram definidos para representar condições locais, para aplicações, e globais para a rede. Além disso, para corretamente transportar tais parâmetros de QoS pelos elementos da rede que fazem parte da comunicação, foi criado o gerenciador de recursos. O gerenciador recomendado para uso na Internet é o RSVP. Para que seja possível fornecer a QoS requisitada pela aplicação é necessário um módulo de controle de tráfego atuando em conjunto com o gerenciador RSVP. Este módulo deve implementar funções básicas como classificação, escalonamento e controle de admissão de pacotes. A proposta deste trabalho é observar a QoS obtida pelas aplicações em duas situações: quando o ambiente onde as comunicações são realizadas possuir controle de QoS (RSPV e controle de tráfego) e sem a presença de controle sobre as comunicações. Diversos experimentos foram realizados de forma a podermos avaliar o impacto na qualidade de serviço obtida pelas aplicações nesses ambientes.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

Quality of Service analysis of real-time applications using resource reservation in the Internet

Ana Luísa Pereira Schmidt

September/1999

Advisor: Rosa Maria Meri Leão

Department: Computer and Systems Engineering

The recent growing of real-time applications over public networks such as Internet claimed to the development of more sofisticated mecanisms to control several type of network parameters like jitter , delay and loss probability. One approach called Integrated Services was created trying to accomplish this issue: a set of quality of service (QoS) parameters was defined to denote local and global conditions to applications and to the network, respectively. A Resource Manager was created to carry such QoS parameters across the network path formed among the involved applications. The RSVP protocol is the one recomended to be use in the Internet. To obtain the QoS required by the applications, it is necessary that a Traffic Control Module work in cooperation with RSVP. This module should implement functions such as packet classification, scheduling and admission control. The goal of this study is to perform a set of experiments in order to evaluate the QoS obtained by the real-time applications considering the following scenarios: when the network provides mechanisms to control the QoS (RSVP and Traffic Control) and when no such mechanisms are used.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O presente documento mostra a análise realizada em um ambiente de rede com garantias de Qualidade de Serviço (QoS), utilizando aplicações com características de tempo real. Foi observado o comportamento de parâmetros que expressam a QoS, como jitter e perda de pacotes para essas aplicações, diante de cenários com e sem o gerenciador de recursos RSVP e o módulo de controle de tráfego, traçando comparações de forma a determinar se a presença desse tipo de controle das comunicações traz ou não vantagens para essas aplicações.

1.1 Histórico

A Internet e maioria das redes de comutação de pacotes oferecem serviços nos quais há, por parte da sub-rede de comunicação, esforço máximo para transferir a informação fim-a-fim, fornecendo a melhor qualidade de serviço possível. Isso significa que as aplicações que utilizam essas redes como rede de suporte, não possuem mecanismos para requisitar um controle priorizado para suas comunicações. Cada participante da comunicação, seja emissor, receptor ou roteador(es) intermediário(s), entrega aos pacotes todos os recursos disponíveis para a realização do serviço, em termos de largura de banda e *throughput*. Esse tipo de comunicação, chamada de "*best-effort services*", possuem as seguintes aplicações típicas: *telnet*, *FTP*, *login* remoto ou mesmo o correio eletrônico.

No entanto, com o desenvolvimento de aplicações ditas de tempo real (onde o parâ-

metro tempo é um requisito importante) e multiponto (i.e. transmissão de uma fonte para vários destinos ou de várias fontes para vários destinos), foi necessário desenvolver mecanismos para que a sub-rede de comunicação pudesse prestar um serviço com uma QoS diferenciada, uma vez que essas aplicações como vídeo e voz exigem uma qualidade mínima em termos de parâmetros temporais (como atraso e jitter) e capacidade efetiva de transmissão (como largura de banda).

Para a realização desse serviço diferenciado, é necessário que as aplicações forneçam parâmetros que indiquem os requisitos ideais para seu fluxo de dados e que os roteadores implementem algoritmos para controlar tais fluxos. Assim sendo, a monitoração e controle de parâmetros de QoS ao longo do percurso de dados torna-se uma medida importante.

Pensando nesse cenário, foi proposta uma nova abordagem para os serviços das redes de pacotes chamada de Serviços Integrados para Redes de Pacotes: **ISPN**[1] (do termo em inglês *Integrated Services Packet Network*). Na metodologia proposta pelo **IETF** (*Internet Engineering Task Force*), cada aplicação que necessite de uma QoS para seus pacotes, deve informar as condições que julge necessárias ou imprescindíveis para que isso se concretize.

A fim de expressar a QoS, um conjunto geral de parâmetros foi criado. Com eles a aplicação tem a sua disposição, mecanismos para informar qual o melhor ambiente às suas comunicações. Também faz-se necessário um controle maior de todos os elementos do processo: roteadores intermediários, rotas, bem como transmissores e receptores visando entregar as informações com a QoS requisitada pelos participantes.

Mecanismos de controle de admissão e escalonamento de pacotes tornaram-se importantes para que os roteadores mantenham uma dinâmica dos recursos mais equilibrada, de forma que todas as aplicações tenham iguais oportunidades de realizar suas comunicações. O controle de admissão implementa um algoritmo que garante ou nega os recursos, mantendo a carga da rede em níveis aceitáveis para todos os usuários. O escalonamento de pacotes por sua vez, utiliza-se de algoritmos que decidem em que ordem devem ser enviados os pacotes que estejam aguardando transmissão, de acordo com a prioridade definida pela aplicação.

Fez-se necessário também a existência de um protocolo de gerência de recursos, de forma a requisitar e reservar em um determinado caminho, a QoS requerida pelas aplicações, estando estas em ambiente ponto-a-ponto ou multiponto. Como

recursos mais comumente utilizados para a gerência, podemos citar a largura de banda desejada pela aplicação, o controle de tamanho de buffers (para comportar possíveis rajadas), etc.

A utilização de um gerenciador de recursos no contexto da rede, é importante à medida que implementa mecanismos para o tratamento particularizado do tráfego (seja ele interativo ou não). Para tráfegos que possuam características comuns em termos de parâmetros de QoS, mecanismo desenvolvido justamente para classificar e controlar os fluxos, o compartilhamento de recursos torna-se factível. Caso não haja possibilidade de compartilhamento, torna-se possível adotar políticas diferenciadas para cada fluxo.

Uma das propostas mais bem aceitas para a gerência de recursos da rede, foi desenvolver um protocolo de reserva que fosse transparente o suficiente para ser utilizado por todos os mecanismos de roteamento ponto-a-ponto ou multiponto já implementados. Foi criado então o **ReSerVation Protocol** (RSVP)[8] que tem como seguintes características principais [9]:

1. Adaptação dinâmica a mudanças no cenário, seja por alteração de rotas ou de participantes da comunicação.

2. RSVP é simplex, realizando a reserva para somente um sentido do fluxo de dados.

3. É o receptor o responsável por indicar e manter as condições de reserva usada pelo fluxo de dados.

4. Os roteadores possuem mapeamentos das condições da rede a fim de adaptarem-se automaticamente a mudanças na rede.

5. O RSVP não é um protocolo de roteamento, fazendo-se necessário trabalhar em conjunto com o protocolo disponível.

6. Disponibiliza estilos de reserva que ajudarão a classificar e trabalhar os diversos fluxos de forma mais eficiente.

7. Seus pacotes são encapsulados de forma a não gerar conflitos com os roteadores que não realizem reserva de recursos.

8. Realiza reserva tanto para o protocolo IPv4 quanto para o IPv6.

Como descrito anteriormente, o RSVP é um protocolo que utiliza os parâmetros de QoS requisitados pelas aplicações para controlar os recursos ao longo do caminho desse fluxo de dados. A fim de realizar o controle em cada ponto é necessária a presença de mecanismos de controle de tráfego. No contexto deste trabalho usamos o software ALTQ[25], que foi desenvolvido para máquinas com sistema operacional FreeBSD e que possui interface com o RSVP. O ALTQ implementa mecanismos de escalonamento de pacotes, de classificação de fluxos de tráfego e de controle de admissão.

Características do protocolo RSVP serão detalhadas no capítulo 3. A descrição do módulo de controle de tráfego ALTQ será feita no capítulo 4. Esses mecanismos de controle da rede foram utilizados como base para o ambiente de teste que será abordado a seguir.

1.2 Trabalho realizado

O objetivo deste trabalho é o estudo e a configuração de um ambiente com QoS e a execução de diversos experimentos.

Visando investigar o comportamento de aplicações multimídia nesse ambiente, serão traçadas comparações entre o funcionamento dessas aplicações com e sem reserva de recursos. Com este trabalho, poderemos avaliar o impacto do uso do protocolo de reserva de recursos RSVP, juntamente com seu módulo de controle de tráfego (ALTQ) na QoS fornecida às aplicações. Assim como foi realizado para um ambiente próprio, imaginamos que o ambiente possa ser extendido à toda Internet levando um controle diferenciado aos demais pontos da rede.

O teste consistirá na utilização de aplicações de tempo real de voz e vídeo e a correspondente avaliação de dois parâmetros básicos: o jitter e a perda de pacotes.

A aplicação de voz será simulada usando uma fonte markoviana ON-OFF. A cada período ON, a fonte gera pacotes de tamanho fixo a intervalos determinísticos que são captados por outros pontos da rede simultaneamente. Os pontos receptores analisam os pacotes, medindo a distribuição do jitter, a sua média e variância, além de registrar a perda de pacotes, caso haja, ocorrida no processo.

O mesmo princípio será utilizado para registrar a chegada dos pacotes da fonte de vídeo, que transmite seqüências codificadas segundo o padrão MPEG. Serão

realizadas investigações com respeito aos parâmetros de tráfego em seqüências de características distintas de curta e longa duração, representando a transmissão de um filme.

Em termos de configuração de testes, o trabalho contará com a utilização de dois ambientes: o primeiro, dito controlado, será montado em uma rede local isolada. A transmissão e recepção ocorrerá dentro da mesma sub-rede, eliminando-se com isso, problemas causados por tráfegos concorrentes. O segundo, dito remoto, contará com a utilização de máquinas em dois laboratórios distintos situados em duas unidades do campus da UFRJ, um localizado no NCE e o outro no CT-Bloco I.

O presente documento está organizado em 5 capítulos. O segundo capítulo conta com uma abordagem generalizada sobre a literatura envolvendo a QoS, bem como um estudo sobre a comunicação multiponto. O terceiro aborda os aspectos mais importantes do protocolo de reserva de recursos, RSVP e seus módulos acessórios. O quarto capítulo descreve em mais detalhes os mecanismos de controle de tráfego utilizados pelo RSVP. O quinto capítulo conta com a organização, montagem e configuração dos ambientes de testes. Todos os resultados obtidos também estão incluídos no quinto capítulo. Encerrando esta documentação temos um resumo das principais conclusões organizado no sexto capítulo. A documentação conta ainda com um Apêndice, onde são colocadas as principais estruturas contidas nas mensagens de QoS trocadas entre os participantes da comunicação.

Capítulo 2

Revisão Bibliográfica

O presente capítulo pretende revisar os principais tópicos estudados, a fim de situar o trabalho realizado. Basicamente, para criar um ambiente de rede que dispusesse aos pacotes de aplicações de tempo real (vídeo e voz) estabilidade em termos de atrasos e perdas por congestionamentos, foi proposta a arquitetura de rede ISPN (*Integrated Services Packet Network*), pelo IETF (*Internet Engineering Task Force*), preocupada em adicionar mecanismos de controle para a rede de pacotes tradicional e oferecer uma real Qualidade de Serviço (QoS) às aplicações. Tal QoS define uma série de parâmetros gerais que cada aplicação necessita para sua comunicação, sendo então tarefa dos mecanismos de reserva de recursos e de controle de tráfego disponibilizar as condições ao longo de todos os pontos do percurso do fluxo de dados em conformidade com os parâmetros de QoS pedidos. Vale ressaltar que as aplicações em questão utilizam-se, em sua maioria, de comunicações multiponto, isto é, de um ponto para vários outros, onde o modelo da comunicação associado difere em relação ao utilizado para aplicações ponto a ponto. O estudo abrange três temas principais: a metodologia criada pelo ISPN; a necessidade de comunicar QoS aos pacotes das aplicações e definições necessárias para realização de comunicação multiponto.

2.1 Serviços Integrados para Redes de Pacotes [1]

Para dar suporte às aplicações típicas de comunicação como correio eletrônico ou *telnet*, o modelo até então utilizado para as camadas básicas, mais especificamente a camada de transporte, era o *"best-effort service"* ponto-a-ponto, onde cada elemento da rede disponibiliza todos os recursos disponíveis para cada fluxo de dados associado. Tal modelo, no entanto, não apresenta nenhuma previsão de mecanismos de controle de admissão, escalonamento de pacotes ou reserva de recursos que possa garantir a qualidade de serviço exigida por aplicações como vídeo e voz. A ocorrência de atrasos nos pacotes em geral não acarreta problemas para as aplicações de dados, pois essas não necessitam de respostas em tempo real.

À medida que foram surgindo aplicações multimídia (vídeo e voz interativos), onde grandes variações em parâmetros como atraso e banda são significativas a ponto de interferir diretamente na inteligibilidade da mensagem, uma adaptação do modelo de comunicação "*best-effort*" tornou-se fundamental para que este corretamente interprete e corrija as oscilações existentes.

O caminho escolhido para melhorar o modelo, foi extendê-lo de forma a agregar em seus pacotes informações específicas sobre a QoS desejada pela aplicação. Isso foi conseguido definindo-se uma série de parâmetros que sinalizam as diversas necessidades das aplicações[4]. Além disso, tornou-se necessária a habilidade de controlar o compartilhamento de banda de um nó particular entre diferentes tráfegos. Isto é realizado através da divisão do tráfego entre algumas classes visando atribuir-lhes uma percentagem mínima de banda para que sempre haja condições de servir ao maior número de aplicações possíveis. Essas classes podem representar diferentes grupos de usuários ou diferentes famílias de protocolos. O gerenciamento do nó é comumente chamado de controle de compartilhamento de enlace (do inglês *controlled link-sharing*).

Todas essas extensões do modelo tradicional fazem parte de uma arquitetura conhecida como serviços integrados, onde há a proposição de um modelo para a Internet que compreenda as seguintes funções: capacidade de escalonamento de pacotes de dados; suporte a aplicações de tempo real e multiponto e gerenciamento de todo o ambiente participante da comunicação (não apenas controle fim-a-fim) em termos de recursos diponíveis para as aplicações.

Para realizar esse ambiente, existem dois pré-requisitos[10]:

- Os roteadores e elementos intermediários das diversas redes, ao longo do percurso realizado pelos pacotes de dados de uma determinada aplicação, devem incorporar mecanismos para controlar a QoS entregue a esses pacotes;
- Deve-se disponibilizar uma forma de comunicar as necessidades de uma determinada aplicação aos elementos de rede ao longo do caminho e gerenciar a QoS ao longo do caminho.

Na metodologia criada pelo ISPN, o primeiro pré-requisito é desempenhado pelos serviços de controle de Qualidade de Serviço tais como **Serviço de Carga-Controlada** (*Controlled-Load*)[2] ou **Serviço Garantido** (*Guaranteed*)[3]. O segundo pode ser realizado de várias formas, porém é freqüentemente implementado por um protocolo de reserva de recursos como o **RerSerVation Protocol[8, 9, 10]** ou **Stream Protocol[11, 12]**.

Sendo o RSVP projetado para utilização com diversos serviços de controle de QoS e sendo a QoS desenvolvida para atender a uma série de mecanismos de reserva, a interface entre os dois foi definida de forma bastante genérica. A especificação do RSVP não define o formato interno dos campos do protocolo ou objetos que são relacionados ao serviço de controle de QoS. Para o RSVP esses campos ou objetos são completamente transparentes, sendo estes os responsáveis por transportar diferentes informações para diferentes aplicações e necessidades do serviços de controle de QoS. O protocolo RSVP será devidamente detalhado no próximo capítulo. De maneira semelhante, as interfaces para os serviços de controle de QoS são definidas em um formato genérico, de forma que os serviços possam ser integrados com uma variedade de mecanismos de controle e gerenciamento de rede. Veremos a seguir mais informações sobre os parâmetros que integram a QoS.

2.2 Qualidade de Serviço (QoS)[4]

Para compor um mecanismo tal que realize a comunicação das necessidades das aplicações em relação à QoS, foi definido um conjunto *geral* de parâmetros de controle e caracterização, a serem utilizados por todos os elementos de rede que venham a reunir-se ao modelo dos serviços integrados. O termo *geral* indica que tais parâmetros devem possuir uma definição comum, compartilhada por todos os serviços que

fazem parte da estrutura implementada. Para representar tal estrutura, foi criado um identificador de parâmetro, seguido do valor do parâmetro, conforme abaixo:

<**número-do-serviço, número-do-parâmetro**>valor_do_parâmetro,

onde, se o número-do-parâmetro estiver entre 2 e 127, indica que seu significado é o mesmo em todos os serviços; caso esteja entre 128 e 254, o significado dependerá do número do serviço. O serviço com carga controlada (detalhado mais adiante) possui número de serviço 5. Já o serviço garantido possui número 2.

Os parâmetros estão classificados como a seguir:

Parâmetros de controle são usados pelas aplicações informando à rede quais os requisitos de controle de QoS desejados, como por exemplo a especificação_do_tráfego (*traffic specification* ou TSPEC).

Parâmetros de caracterização são utilizados para caracterizar o ambiente de gerenciamento de um caminho específico a ser percorrido pelo fluxo de pacotes sobre os quais o controle de qualidade de serviço se aplica. Tais parâmetros de caracterização podem, eventualmente, serem utilizados tanto pela própria aplicação que requisita o controle ou pelos outros elementos da rede, compondo um mecanismo de acompanhamento dinâmico do cenário sobre o qual uma determinada aplicação está envolvida. Como exemplo da utilidade deste parâmetro dinâmico, pode-se com ele estimar, ao longo de um determinado caminho, a quantidade de banda disponível e o atraso, facilitando a decisão de uma determinada aplicação no momento de variar algum requisito de qualidade.

Além disso, cada parâmetro de caracterização possui dois tipos de valores: um **local** (dito "por interface"), plenamente acessível pela aplicação e que reflete as condições de um determinado elemento da rede e outro **composto** (dito "por próximo nó"), associado à composição dos diversos valores locais ao longo do caminho, dependendo da regra de composição definida para o parâmetro.

Cada parâmetro define a regra de composição para si próprio, além de propriedades como faixa de valores e precisão. A regra de composição descreve como combinar um valor recebido da rede e o próprio valor local para criar um novo valor composto que será propagado ao próximo elemento do caminho, registrando as condições de QoS pelos diversos caminhos ou rotas possíveis entre a aplicação fonte e a destino.

O transporte e gerenciamento de tais informações pela rede, não é contudo tarefa dos mecanismos de controle de QoS, devendo existir um agente externo com tal atribuição. Uma vez que os valores dos parâmetros de caracterização locais e compostos têm identificadores distintos é possível requisitá-los em qualquer ponto da rede, facilitando o gerenciamento.

Todos os parâmetros descritos na especificação de serviços integrados são obrigatórios, no sentido de que qualquer elemento de rede que deseje receber qualidade de serviço deve reconhecer a chegada desses parâmetros em mensagens de configuração e gerenciamento do protocolo, processá-las corretamente e propagar valores como resposta. Para determinados parâmetros de caracterização, ditos gerais, é obrigatório que o elemento de rede propague um determinado valor local. Para outros parâmetros, é aceitável que o elemento de rede indique que não pode enviar um valor local exato ou aproximado, informando valores do tipo "indeterminado" ou mesmo "inválido". Neste caso, a composição de um resultado fim-a-fim será questionável.

2.2.1 Parâmetros gerais

Conforme a definição, parâmetros gerais são parâmetros de caracterização que têm um significado comum ao longo de um caminho constituído de elementos que implementem a QoS. Muito freqüentemente, o mesmo valor de um desses parâmetros será aceitável para todos os serviços de controle de QoS existentes em um elemento de rede. Conseqüentemente, não existe a necessidade de se enviar uma cópia separada de um valor para cada mecanismo de controle de QoS; basta propagar um valor único para todos os outros elementos.

Um valor para um parâmetro geral que se aplique para todos os serviços suportados em um elemento de rede é chamado um valor *default* ou global. Por exemplo, se todos os serviços presentes em um determinado nó têm o mesmo tamanho máximo de pacote, o nó pode enviar uma mensagem para todos os outros elementos com único valor padrão para o parâmetro *PATH_MTU*, ao invés de uma mensagem contendo diversos valores do parâmetro PATH_MTU (um para cada serviço). Como ganho associado a essa simplificação, temos a redução do tamanho da mensagem bem como do custo de processamento das informações em cada nó.

A representação do identificador de parâmetro para um valor *default* é a seguinte:

<1, número-do-parâmetro>

Ocasionalmente, um serviço pode informar um valor diferente do valor *default* para um determinado parâmetro. Restrições no escalonamento de pacotes associados aquele elemento de rede específico ou mesmo considerações de *hardware*, podem indicar que, por exemplo o parâmetro *PATH_MTU* deva ter um valor abaixo do *default*. Nesse caso, o protocolo de reserva deve entregar aos outros elementos, bem como à aplicação, ambos os valores (o *default* e o específico).

Essa distinção entre parâmetros *default* e específicos não faz sentido para parâmetros que não sejam gerais, mesmo por que a definição e o valor deste parâmetros serão sempre específicos a um serviço particular.

Descrição de parâmetros gerais

A seguir será apresentada uma descrição suscinta dos parâmetros gerais (aqueles que mantém o mesmo significado, bem como as regras de composição) passados a todos os serviços de controle de QoS em um determinado caminho. O formato das mensagens trocadas usa o padrão *XDR* (*Extendend Data Representation*) para a representação dos parâmetros.

- *NON_IS_HOP* - proporciona a informação sobre a presença de elementos de rede que não implementam o controle de QoS ao longo do caminho ou rota associada. Seu identificador *default* associado é **<1,2>** e sua regra de composição é uma função **OU**.

- *NUMBER_IS_HOP* - expressa o número dos pontos do percurso onde os serviços integrados estão ativos. A regra de composição é realizada por um contador que é incrementado caso a QoS esteja sendo implementada. Seu identificador *default* é **<1,4>**.

- *AVAILABLE_PATH_BANDWIDTH* - informa sobre a quantidade de banda disponível ao longo de um determinado caminho seguido por um fluxo de dados. É composto por informações locais levando em consideração todas as informações disponíveis no elemento de rede sobre o caminho, bem como dados sobre políticas de controle sobre a banda além de recursos físicos. A regra de composição deste parâmetro é fornecer o mínimo dos valores recebidos dos outros participantes do caminho seguido pelo fluxo. Seu identificador *default* é **<1,6>**.

- *MINIMUM_PATH_LATENCY* - é o valor do tempo de processamento incorporado ao pacote na passagem por um determinado elemento de rede. A

regra de composição para este valor é a soma do valor local com os valores informados até então. Sua quantidade final informa o atraso fim-a-fim que um fluxo de dados será submetido. O número-do-parâmetro para o atraso local é **7** e para o atraso recebido até então é **8**.

- *PATH_MTU* - computa a máxima unidade de transmissão para os pacotes seguindo um fluxo de dados. A regra de composição é levar em consideração o menor valor informado. O número-do-parâmetro local é **9** e o global é **10**.
- *TOKEN_BUCKET_TSPEC* - é usado para descrever os parâmetros de tráfego que uma determinada aplicação tenciona gerar. Ele é considerado um parâmetro geral pois pode ser utilizado por vários serviços de controle ao mesmo tempo. As regras de composição para este parâmetro estarão detalhadas na próxima seção. Internamente sua estrutura contém os seguintes campos: especificação do *token bucket*, incluindo taxa média (r) e altura (b) do *bucket*, além da taxa de pico (p), o tamanho mínimo (m) e o tamanho máximo da mensagem (M). Essa estrutura e sua regra de composição serão estudadas com mais detalhes a seguir. Seu número-do-parâmetro é **127**.

2.2.2 Regras de composição de QoS

Esse mecanismo é interessante pois permite, variando-se a regra de composição, estabelecer diferentes critérios de caracterização das informações da rede facilitando para os demais elementos escolher este ou aquele caminho. É responsabilidade do adminstrador da rede documentar uma faixa de valores para parâmetros de caracterização tal que permita que todos os nós ao longo do caminho estejam em uma faixa aceitável de valores. É sugerido que apenas um valor seja enviado ou propagado, facilitando os mecanismos de decisão dos outros componentes da rede.

Um resumo completo das regras de composição para os parâmetros de caracterização encontra-se abaixo descrito. Tais regras deverão ser aplicadas a cada um dos parâmetros recebidos, aplicando-se a regra de composição associada. Seja um par hipotético recebido $<S,P>x$, $<S,P>y$ o par local e $<S,P>z$ o par resultado enviado, onde S, P, x , y e z referem-se ao número-do-serviço, número-do-parâmetro e valores do parâmetro.

1. Examinar o par $<S,P>$ associado ao valor x que está sendo enviado pelo nó anterior do caminho.

2. Se $P > 128$, i.e, um parâmetro que depende do número do serviço S, compor x com o valor local y disponibilizado pelo serviço local correspondente passando o resultado ao próximo nó do percurso na forma $<S,P>z$.

3. Se $2 <= P <= 127$ e $y \neq x$, aplicar-se-á a regra de composição do parâmetro P correspondente, enviando o resultado ao próximo elemento na forma $<S,P>z$.

4. Se o ocorrer o caso anterior, porém o serviço S local não possuir um valor y para o parâmetro P, o nó utilizará seu valor *default* local para o parâmetro $<1,P>y$, criando um novo par $<S,P>z$ que será transmitido.

5. Se o par recebido contiver um valor *default* para o parâmetro P, i.e. par $<1,P>x$ e quaisquer serviços locais S exportarem um valor local y diferente do global x recebido, aplicar-se-á a regra de composição do parâmetro P, enviando o resultado ao próximo nó, na forma $<S,P>z$. Vale ressaltar que tal ocorrência aumentará o tamanho da mensagem.

6. Se o caso anterior ocorrer e o valor local y correspondente ao parâmetro testado for idêntico ao valor *default*, i.e. $x=y$ e $S=1$, a regra de composição será aplicada, enviando-se o valor *default* $<1,P>z$ para o próximo nó. Esse passo é realizado mesmo tendo ocorrido o passo anterior.

2.2.3 Serviços de controle para a QoS

Basicamente existem duas implementações para os serviços de controle de qualidade de serviço.

- **Serviço Garantido**

  No serviço garantido (em termos de atraso e banda), os datagramas de um determinado fluxo chegarão dentro de um tempo de entrega e não serão perdidos (a menos de um congestionamento), respeitando os parâmetros de QoS requisitados pela aplicação. Este serviço é direcionado para aplicações que requeiram uma garantia rígida de que seus pacotes não cheguem depois de um tempo máximo de atraso. Essa modalidade

de QoS é importante para aquelas aplicações com padrões rígidos de tempo real. Este serviço não tenciona minimizar o *jitter,* somente controla o máximo atraso possível de um determinado fluxo.

- **Serviço com Carga-Controlada**

  O serviço com carga controlada foi desenvolvido para uma ampla gama de aplicações existentes, que sejam sensíveis à condições de carga na rede, isto é, situações que degradam seu desempenho.

  Para isso, cada elemento de rede que aceite uma requisição para serviço com carga-controlada, deve assegurar que os recursos adequados de banda e processamento de pacotes estejam disponíveis na forma exigida pela aplicação. Sua função é proporcionar um ambiente tal como se a aplicação estivesse recebendo serviços *best-effort* sob baixas condições de carga na rede. Desde modo, ele não garante nada em relação ao atraso como o serviço anterior, porém realiza a comunicação atribuindo prioridades para os fluxos.

  De uma maneira geral, o sucesso para que essa qualidade de serviço seja conseguida é alocar mais banda do que aquela que fora requisitada pela aplicação.

2.2.4 O parâmetro TOKEN_BUCKET_TSPEC

A rede implementa o serviço de carga controlada através do gerenciamento dos fluxos. Tal gerenciamento é baseado na especificação dos parâmetros de tráfego desejados pela aplicação e nas condições da rede. As requisições pedidas para um fluxo novo só serão aceitas se o elemento de rede possuir a capacidade de entregar a QoS requisitada. Da mesma forma, modificações na QoS de fluxos já aceitos, também deverão ser analisadas como novas requisições, fazendo com que o controle de admissão seja novamente aplicado àquele fluxo modificado. Requisições para serviços com QoS inferiores, ao contrário, seriam sempre permitidas.

O parâmetro TOKEN_BUCKET_TSPEC é utilizado para caracterizar o fluxo de dados. Essa especificação contém os seguintes campos: a especificação do *token bucket*, que inclui a taxa média (r) e o tamanho do *bucket* (b), a taxa de pico (p), o tamanho mínimo (m) e o maior tamanho da mensagem (M).

O significado de cada campo pode ser interpretado da seguinte forma:

r taxa média de dados produzida pela aplicação (*bits*/tempo)

b número máximo de mensagens, quando a aplicação encontra-se em um pico (*bits*)

p taxa de pico da aplicação (*bits*/tempo)

m menor tamanho de mensagem que a aplicação espera gerar (*bits*)

M maior tamanho de mensagem que a aplicação espera gerar (*bits*).

A ordenação de diversos parâmetros TSpecs é dada pela seguinte formulação. Sendo TSpecA descritor de um tráfego A e TSPecB descritor do tráfego B, o descritor de A poderá ser substituído pelo de B se e somente se:

1. o campo r e o campo b e o campo p e o campo M de TSpecA forem iguais ou maiores do que o correspondente de TSpecB e
2. o campo m de TSpecA for igual ou menor do que o correspondente de TSpecB.

Com isso, todos os TSPecs podem ser ordenados com relação uns aos outros. Como exemplo da utilização da ordenação desses descritores, caso um descritor seja "igual ou maior do que" outro, o descritor ora implantado poderá ser substituído pelo novo descritor, sem que isso represente alteração das condições de QoS para aquele tráfego já existente.

Além disso, esse parâmetro possui algumas regras de composição que vão permitir a união de fluxos. A seguir daremos exemplo de uma delas.

A união de um conjunto de fluxos, cada um com um determinado TSpec, terá como resultado um fluxo com TSpec resultante segundo a forma abaixo.

1. a maior taxa de *token bucket*;
2. o maior dos valores para o campo b;
3. a maior taxa de pico;
4. a menor unidade de m;
5. o menor dos valores para o campo M;

Outras regras de composição com o parâmetro TOKEN_BUCKET_TSPEC podem ser obtidas em [2].

2.3 Comunicação multiponto

O desenvolvimento das aplicações multimídia nos últimos anos, apressou soluções para os problemas de comunicações de ponto-multiponto ou multiponto-multiponto, ou seja, comunicações onde existem grupos de usuários que desejam trocar informações. Para resolver a questão, outros modelos de sistema precisaram ser criados, uma vez que o modelo OSI (*Open System Interconnection*) ou a estrutura que até então suporta o TCP/IP só contempla comunicações ponto-a-ponto.

O modelo de comunicação ponto-a-ponto é realizado entre dois participantes, onde um é tipicamente o cliente (ou o responsável pelo início da comunicação ou principal) e o servidor (ou passivo ou participante escravo). Numa sessão multiponto, diferentemente, qualquer participante pode decidir se e quando ele deve juntar-se ou deixar uma dada sessão. Essas operações devem ser simples, sem efeitos colaterais para os outros participantes, podendo o grupo ter proporções pequenas (duas subredes por exemplo) ou proporções continentais (dezenas de subredes espalhadas por diversos países).

A proliferação de aplicações multimídia associadas a redes de alta velocidade (baseadas em tecnologia ATM) tem conduzido à necessidade de um modelo onde os mecanismos de comunicação em grupo sejam confiáveis. No modelo empregado para a Internet atual, o TCP é um protocolo ponto-a-ponto. Entretanto recentemente tem-se adaptado o protocolo UDP para dar suporte a algumas modificações, construindo aplicações multiponto (e não confiáveis pela natureza do UDP) sobre o *MBone* (*Multicast Backbone*). Questões como roteamento multiponto, controle de tráfego, multiponto com qualidade de serviço, confiabilidade e segurança estão sendo analisadas e devem ser resolvidas para se criar um ambiente ideal para aplicações multiponto.

No decorrer dos próximos ítens, serão abordados temas centrais sobre o ambiente necessário para que as aplicações multiponto possam atuar. Foram incluídos assuntos relevantes sobre endereçamento, gerenciamento de entrada e saída em grupos e a descrição de alguns protocolos de roteamento.

2.3.1 Endereçamento de grupo

O grupo é uma abstração que representa o conjunto de entidades que desejam se comunicar. Para controlar uma comunicação de grupo, foram introduzidos os conceitos

de endereço de grupo, identificador de grupo, propriedades do grupo e gerenciamento de grupo sobre o modelo IP.

Na notação decimal comumente utilizada para endereçamento na Internet, a faixa de endereço de grupos multiponto é de 224.0.0.0 até 239.255.255.255, permitindo a existência de 250 milhões de grupos ao mesmo tempo. O endereço **224.0.0.0** tem garantia de não ser atribuído a nenhum grupo

Os membros de um grupo são dinâmicos, significando que um membro poderá entrar ou deixar o grupo a qualquer tempo. Cada membro também poderá pertencer a um ou mais grupos ao mesmo tempo. Os endereços de um grupo podem ser dinâmicos ou permanentes. Para os dinâmicos, enquanto existir um único membro conectado, haverá sentido para o grupo, passando este a não existir quando não mais existirem membros associados. Grupos permanentes são aqueles em que mesmo a ausência de membros não é suficiente para a extinção do grupo.

Alguns exemplos de grupos permanentes:

224.0.0.1 - Todos os sistemas interligados a uma determinada rede local

224.0.0.2 - Todos os roteadores de uma determinada rede local

224.0.0.5 - Todos os roteadores de uma rede local que entendam o protocolo MOSPF

Os grupos dinâmicos devem ser criados antes de serem utilizados. Um processo deve pedir para que sua interface de rede se junte (ou se libere) de um grupo específico, utilizando para isso algumas funções especialmente destinadas a esse propósito.

O processo de gerência de grupo utiliza-se do endereço 224.0.0.1 para, uma vez a cada minuto aproximadamente, pedir que todos os participantes da subrede o informem quais grupos estão sendo utilizados naquele momento. Isso faz com que cada elemento, mande uma mensagem para todos aqueles pontos em que esteja conectado ou que deseje manter conexão (por exemplo grupos permanentes).

O procedimento descrito acima, utilizando mensagens, faz parte do protocolo IGMP (*Internet Group Management Protocol*). Ele possui apenas dois tipos de pacotes: *Consultas* e *Relatórios*, cada qual com um formato fixo contendo algumas informações de controle além do endereço de classe D. O protocolo será descrito no item seguinte com mais detalhes.

2.3.2 Gerenciamento de grupo

Para algumas aplicações, seus processos são executados em um ambiente distribuído; elas trabalham em grupos, formando uma seqüência de operações que necessitam de suporte de rede para trocar mensagens e tomar as devidas ações. Como exemplo, podemos citar:

- atualização de todas as cópias de um banco de dados distribuído;
- envio de vídeo e voz a todos os membros de uma teleconferência ou
- o envio de resultados para todos os participantes de uma computação distribuída.

Se o grupo for pequeno, trocas de mensagens ponto-a-ponto podem ser consideradas como procedimento válido. No entanto, à medida que o grupo cresce, o custo de mandar várias mensagens com o mesmo conteúdo para vários membros do grupo aumenta, tornando o procedimento inviável. Para resolver esse problema, alguns algoritmos de roteamento multiponto foram desenvolvidos. Tecnicamente, o roteamento multiponto é definido como a transmissão de datagramas IP para um grupo de máquinas identificadas por um único endereço de grupo. Os membros de um determinado grupo são dinâmicos, significando que a qualquer momento um membro poderá entrar ou sair do grupo. Também não há restrições quanto ao número de participantes nem em relação à localização geográfica de cada um deles. Um membro pode pertencer a mais de um grupo simultaneamente; além disso, uma máquina não precisa se conectar a um determinado grupo para enviar mensagens para ele.

A principal informação requerida para se criar um efetivo roteamento para um grupo é que os roteadores sejam avisados sempre que uma determinada máquina de sua rede deseje trocar informações com algum(ns) grupo(s). Qualquer alteração de cenário, isto é, entrada ou saída do grupo também precisa ser devidamente informada aos roteadores responsáveis. Tais informações são propagadas aos roteadores vizinhos, compondo um ambiente integrado.

Podemos concluir que para que o roteamento multiponto possa ser realizado ele precisa conhecer a localização dos membros do grupo. A função de gerenciar e fornecer essas informações para o roteamento é realizada por um protocolo de gerenciamento de grupo.

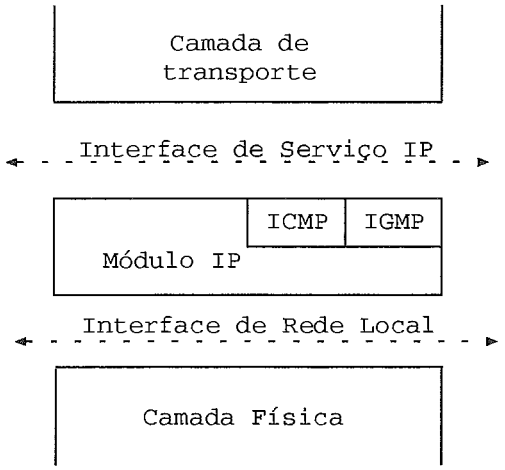


Figura 2.1: O IGMP atua como um módulo da camada IP.

O protocolo **IGMP** (*Internet Group Management Protocol* [6]) foi desenvolvido para esta finalidade. Em termos de representação o protocolo atua na camada IP como mostra o diagrama da figura 2.1. Um roteador multiponto gerencia os membros do grupo que estão localizados em sua rede local. A gerência é feita através da troca de mensagens entre o roteador e as máquinas da rede local. As mensagens IGMP têm o formato, conforme apresentado na figura 2.2:

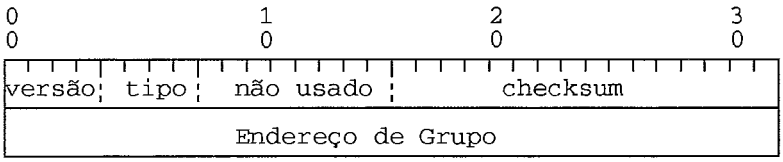


Figura 2.2: Formato das mensagens IGMP.

Os campos têm as seguintes definições:

Versão - atualmente a versão 1 do protocolo IGMP está implementada, segundo [6].

Tipo - tipo de mensagens trocadas entre as máquinas de uma rede local e seus respectivos roteadores: 1 = *Consultas*; 2 = *Relatórios.*

Checksum - para realizar conferências do conteúdo.

Endereço de grupo - em um *Relatório*, é usado para informar o endereço do grupo desejado. Não é utilizado pelas mensagens de *Consultas.*

Operação

Roteadores multiponto enviam mensagens de *Consulta* (definida no protocolo como *Membership Query messages*), a fim de determinar quais grupos possuem membros

em sua rede local. Tais *Consultas* são enviadas para o endereço de grupo 224.0.0.1, conhecido pelo nome de "*all-hosts group*", com o campo TTL (*time-to-live*) do IP igual a 1.

Os membros respondem utilizando-se de mensagens *Relatórios* (definidas como *Membership Reports*), informando sobre cada grupo a qual eles pertençam através da interface de rede por onde a *Consulta* tenha sido recebida. Um *Relatório* é enviado com o endereço de destino IP igual ao grupo desejado e com o parâmetro TTL igual a 1, de forma que os outros membros do mesmo grupo, e que também façam parte da mesma rede local, possam receber o *Relatório*.

A fim de diminuir o número de *Relatórios* enviados, o seguinte mecanismo é implementado pelos membros da rede local:

1. Quando um membro recebe uma *Consulta*, ao invés de respondê-la imediatamente, inicia a contagem de um tempo aleatório entre 0 e D segundos. Quando o tempo expirar, o *Relatório* é enviado. Isso evita a presença de diversos *Relatórios* presentes ao mesmo tempo.

2. Se um membro detecta o envio de um *Relatório* para o grupo ao qual ele pertença, ele interrompe seu próprio contador, não gerando o *Relatório*.

3. Assim apenas um *Relatório* será enviado, evidentemente pelo membro que possuir o menor contador interno.

Existem duas exceções ao ambiente descrito acima. Primeiro: se uma nova *Consulta* chegar até um determinado membro e este ainda estiver contando o tempo para responder a uma *Consulta* anterior, este não será interrompido. Segundo: uma máquina nunca envia um relatório para o endereço 224.0.0.1.

Quando um membro se junta a um determinado grupo, ele deve imediatamente transmitir um *Relatório*, ao invés de aguardar pela próxima *Consulta*. Para evitar que o primeiro *Relatório* seja extraviado, uma repetição deste é realizada em um intervalo menor do que o normal.

Roteadores multiponto enviam *Consultas* periodicamente a fim de manter sua tabela com informações tão atualizadas quanto possível. Se nenhum *Relatório* for recebido de um grupo particular após um determinado número de *Consultas*, o roteador assume que aquele grupo não contém membros instalados em sua rede local.

Consultas são enviadas à razão de uma por minuto, não causando um custo muito grande para esse protocolo. No entanto, no ínicio da operação de um roteador multiponto, essa taxa é muito acelerada, a fim de que este possa preencher suas tabelas de roteamento.

2.3.3 Roteamento multiponto

O desenvolvimento de protocolos de encaminhamento de uma fonte para diversos destinos é um problema complexo: tanto os membros do grupo podem mudar a qualquer instante quanto pode-se ter uma falha em algum nó ou enlace de comunicação. Deve-se os seguintes objetivos:

- Minimizar a carga da rede, evitando "*loops*" e concentração de tarefas em um determinado ponto da rede;
- Proporcionar meios básicos para confiabilidade;
- O algoritmo de roteamento deve achar uma rota ótima a cada alteração de participantes no grupo e a cada mudança de topologia;
- Minimizar o armazenamento de informações nos roteadores para que grupos com muitos participantes possam ser bem realizados.

Existem alguns algoritmos básicos para a construção de árvores multiponto. Em essência exigi-se o conhecimento de vários parâmetros que não são fáceis de quantificar, como topologia da rede, dinâmica do grupo e localização dos membros. A fim de não perturbar a dinâmica do grupo, o algoritmo deve possuir as seguintes características:

- Alterações do grupo devem ser transparentes para os outros membros;
- A nova árvore estabelecida deve manter as mesmas características (em termos de retardo, por exemplo) da rota original e
- O fluxo de dados dos membros do grupo não pode ser interrompido.

Os algoritmos descritos abaixo trabalham com "*best-effort services*", não fornecendo QoS para os membros.

A seguir descreveremos brevemente os algoritmos para roteamento multiponto que estão recomendados para uso na Internet.

Os dois primeiros (DVMRP e MOSPF) são baseados na construção da árvore de menor caminho para cada par fonte-grupo e os receptores do grupo. O terceiro (PIM) é baseado na construção da árvore compartilhada entre as diversas fontes e os receptores de um mesmo grupo.

1) DVMRP - Distance Vector Multicast Routing Protocol

O primeiro e ainda predominante, é baseado em [13]: os primeiros pacotes de uma fonte são enviados em *broadcast* usando o algoritmo RPF (*Reverse Path Forwarding*). Ou seja, um pacote só é passado para os vizinhos se chegar pela interface que diretamente une o roteador àquela determina fonte.

Os roteadores após a passagem destes primeiros pacotes, criam uma entrada na sua tabela de roteamento multiponto para o par fonte-grupo. Esta tabela contém informações do "pai", dos "filhos" e a distância do roteador até a fonte. Ela é construída através de mensagens trocadas entre um roteador e seus vizinhos (a construção é baseada no algoritmo de Bellman-Ford distribuído[26]).

Quando todos os roteadores tiverem finalizado a atualização de suas tabelas, a árvore multiponto estará construída. Tão logo um roteador observe que não existem membros por um determinado caminho, este ramo da árvore é descartado (*pruned*). Após todos os ramos desnecessários terem sido descartados, tem-se então a árvore *multicast* para aquela fonte do grupo.

Cabe ressaltar que as mensagens de controle trocadas entre os roteadores precisam ser encapsuladas da mesma forma como os pacotes de dados de forma a permitir tráfego livre através dos *firewalls* e outros roteadores que só interpretem comunicações ponto-a-ponto.

2) MOSPF - Multicast Open Shortest Path First

É baseado no OSPF versão 2[14]. Cada roteador conhece a topologia da rede. Quando o primeiro pacote da fonte chega em um roteador, ele constrói a árvore de caminho mínimo da fonte para todos os destinos na rede. Ele armazena, usando o algoritmo de Djikstra[26], na sua tabela de roteamento multiponto para o par

fonte-grupo o enlace de entrada e os enlaces de saída e descarta a árvore de caminho mínimo.

Os ramos da árvore que não possuem membros do grupo são eliminados através de mensagens de *prunne.* Todas as mudanças na topologia da rede são comunicadas através de mensagens enviadas em *broadcast.*

3) PIM - Protocol Independent Multicast[15]

É um protocolo desenvolvido para utilização tanto em modo denso quanto esparso em relação ao número de participantes. Caso o cenário tenha muitos participantes reunidos em uma região geográfica limitada, o protocolo é essencialmente o protocolo DVMRP.

Se, por outro lado, o cenário do grupo contiver membros espalhados, o protocolo PIM constrói uma árvore compartilhada entre as diversas fontes do grupo e os destinos. Para tal, utiliza-se de mensagens que serão enviadas a pontos estratégicos chamados **RP**'s (*Rendez vous Points*). Uma fonte quando deseja entrar em um grupo envia uma mensagem de registro para todos os RP's do grupo. Cada RP envia mensagens de *join* para a fonte confirmando a sua entrada. Os pacotes de dados são enviados então da fonte para os RP's usando o roteamento ponto-a-ponto.

Um receptor quando deseja se juntar a um grupo envia uma mensagem de *join* para o RP mais próximo (ela é encaminhada usando o roteamento ponto-a-ponto). Cada roteador que receber a mensagem de *join* criará uma entrada para o grupo na sua tabela de roteamento multiponto contendo enlaces de saída (por onde as mensagens de *join* chegaram) e o enlace de entrada (enlace para alcançar o RP). Desta forma, a árvore multiponto é construída dos RP's para os destinos.

Salienta-se que a árvore compartilhada pode não oferecer o caminho com menor retardo da fonte para os destinos pois todos os caminhos devem passar por um RP.

Se o transmissor desejar trocar de modo esparso para denso, além de transformar-se em ponto de partida para a árvore de caminho mínimo, também continuará a enviar seus dados aos RP's do grupo a fim de atingir os nós mais distantes que continuarem a usar a árvore compartilhada.

Muitos problemas ainda precisam ser resolvidos como interoperacionalização dos esquemas denso e esparso, a escolha do RP, os critérios para a mudança entre os modos, interação com modos de policiamento e roteamentos com QoS e com gerenciadores de recursos como o RSVP.

2.3.4 MBone

O Mbone é a abreviação do termo em inglês *Multicast Backbone*, representando toda a rede baseada sobre a Internet, capaz de processar transmissões de aplicações multiponto como áudio e vídeo digitais. Este *backbone* está acessível desde 1992.

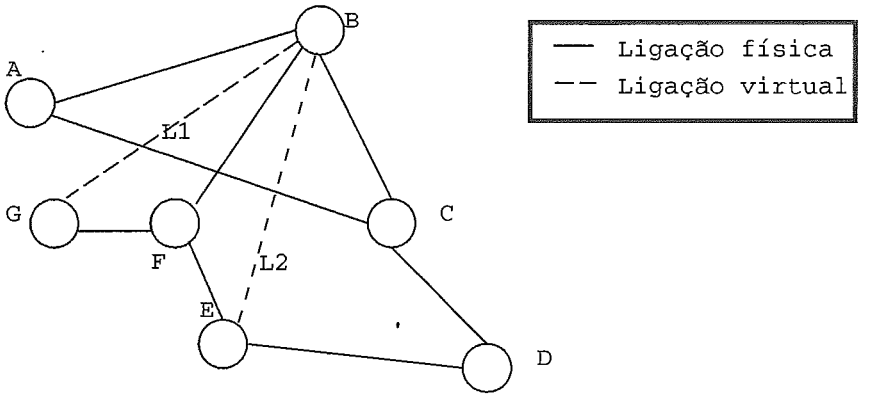


Figura 2.3: Representação genérica de uma rede

Tecnicamente, MBONE é uma rede virtual sobre a rede Internet. Para permitir encaminhamento de pacotes multiponto, mesmo com roteadores que não implementem um protocolo multiponto, foi desenvolvida a técnica conhecida como *tunneling*. Através dela, os datagramas de IP multiponto são encapsulados em datagramas IP *unicast* e endereçados aos roteadores que suportem roteamento multiponto, criando um túnel lógico que permite uma conexão direta entre dois elementos quaisquer. Assim, temos a rede de multiponto formada sobre o IP.

Podemos observar na figura 2.3 uma topologia de rede, cujos nós estão nomeados de **A** até **G**. Neste caso, o nó **B** possui uma configuração virtual através de um túnel com os nós **E** e **G**, permitindo-o "enxergar" diretamente essas duas sub-redes. Os pacotes que fluem entre os roteadores pertencentes ao Mbone são codificados em uma ponta e decodificados na outra extremidade do túnel. Cada ilha ou nó (tipicamente uma rede local ou um grupo de LANs interconectadas) deve ser capaz de realizar o processamento dos pacotes, através da interpretação dos pacotes IP encapsulados. Em algum futuro, quando todos os roteadores forem capazes de lidar com tráfego multiponto diretamente, essa estrutura criada não mais será necessária.

A maioria dos roteadores do MBone (chamados de *mrouters*) são simplesmente máquinas UNIX executando softwares especiais. A maior parte da pesquisa envolvendo o Mbone tem sido de como utilizá-lo eficientemente, uma vez que as transmissões

são sempre baseadas em IP com aplicações usando o UDP, gerenciamento de grupo realizado por IGMP e o roteamento baseado no protocolo DVMRP. Como cada subrede possui um administrador, teoricamente podem ser criados tantos túneis quanto sejam de interesse, levando muitas vezes a rotas confusas e desnecessárias, aumentando o tráfego de pacotes de inicialização.

Capítulo 3

Reserva Recursos

O desenvolvimento de uma arquitetura integrada para atuar junto às aplicações distribuídas de tempo real como voz e vídeo, tem levado ao estudo de dois dos principais requisitos dessas aplicações que são: garantias da qualidade de serviço e comunicações multiponto. Com o propósito de estabelecer e manter os recursos de rede de acordo com as necessidades das aplicações, motivou-se a criação de um elemento gerenciador do processo. Basicamente duas propostas se destacaram, resultando na criação de dois protocolos de reserva de recursos: o ST-II e o RSVP. Nesse capítulo, ambos os protocolos serão descritos, dando-se ênfase ao protocolo RSVP.

3.1 Gerenciamento de recursos

Para a composição de um ambiente integrado, no qual exista uma qualidade de serviço associada, foi definido, como unidade básica do processo de comunicação, um conjunto de pacotes que pertença a uma mesma aplicação e para o qual será requisitado uma mesma QoS foi denominado de *fluxo*.

A fim de lidar com cada fluxo, a arquitetura proposta em [1] para o elemento roteador é formada de quatro módulos: um escalonador, um classificador de pacotes, um algoritmo de controle de admissão atuando a nível de *kernel* e um gerenciador de recursos a nível de usuário. Esta arquitetura pode ser visualizada na figura 3.1.

Resumidamente, os módulos possuem as seguintes funções:

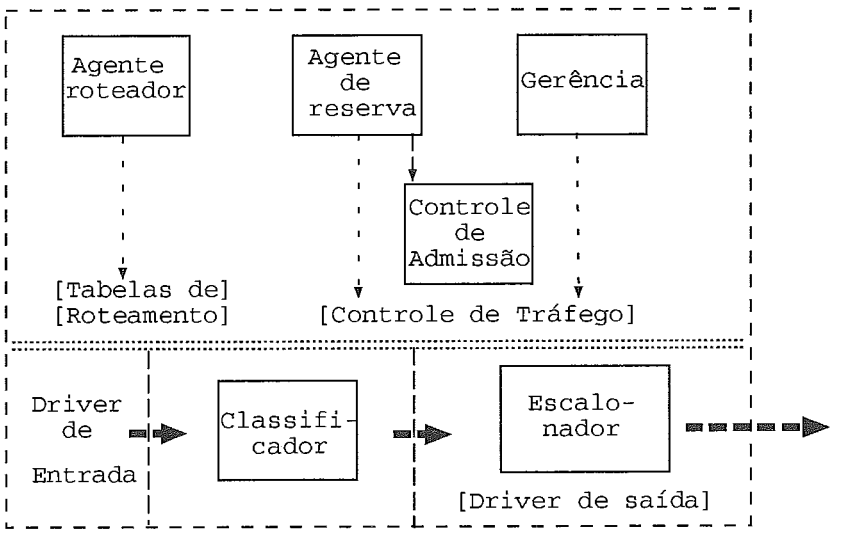


Figura 3.1: Modelo de referência de implementação para roteadores.

- Classificador de pacotes - realiza uma filtragem dos pacotes; o classificador enquadra os pacotes em uma determinada classe que corresponde a uma certa QoS. Se nenhuma classe é encontrada, a classe *default* é utilizada.

- Escalonador de pacotes - a tarefa do escalonador é gerenciar o envio de diferentes fluxos de mesma classe, usando um conjunto de filas para auxiliar o envio de pacotes pela interface de saída.

- Controle de Admissão - determina se a máquina possui os recursos disponíveis para fornecer a QoS para cada pacote.

- Gerenciador de recursos - sua tarefa é alocar e liberar os recursos da rede ao longo do caminho de dados para garantir que os requisitos de Qualidade de Serviço sejam alcançados. É através dele que as condições de QoS serão transmitidas por todos os elementos de rede que façam parte da comunicação.

Além dos quatro principais módulos, devem existir: uma interface entre o agente de reserva e o de roteamento, que poderá consultar as tabelas de roteamento e um mecanismo responsável pelo gerenciamento da rede, sendo este capaz de modificar as políticas de classificação e escalonamento dos pacotes na saída, além de alterar as políticas do módulo de controle de admissão. Caberá aos *drivers* de entrada e saída, implementar as rotinas que permitem adequar o sistema ao *hardware* específico de cada elemento.

A primeira versão de um protocolo para realizar o gerenciamento dos recursos da rede foi batizada de **ST** *Stream Protocol* e o posterior **ST-II**[11], que foi classificado como um protocolo experimental. Uma segunda proposta, desenvolvida para atuar segundo os critérios da ISPN, é o **RSVP[8]**. No decorrer das seções seguintes, uma detalhada abordagem sobre gerenciador RSVP e assuntos relacionados serão mostrados, além de descrições sobre os módulos de controle de tráfego.

3.2 ReSerVationProtocol (RSVP)

3.2.1 Arquitetura

É um protocolo desenvolvido para permitir que as aplicações requisitem diferentes QoS para seus fluxos de dados. Para isso, dois pré-requisitos devem ser observados:

1. Elementos de redes, tais como roteadores, devem adequar-se aos mecanismos de controle de qualidade de serviço para garantir a entrega dos pacotes de dados;

2. Capacitar a aplicação em fornecer os parâmetros ideais de QoS.

O RSVP não é um protocolo de roteamento, trabalhando em conjunto com este. É usado por uma aplicação para requisitar uma qualidade de serviço específica da rede. O protocolo atua tanto em máquinas do usuário quanto em roteadores, responsabilizando-se nesse caso a estabelecer e manter as condições para o serviço requisitado. O diagrama esquemático pode ser visto na figura 3.2.

O RSVP negocia a reserva de recursos em um único sentido de cada vez ou seja, de forma *simplex*. Com isso, ele trata distintamente receptores e transmissores, operando juntamente com a camada de transporte.

O RSVP não realiza transporte de dados sendo apenas um protocolo de controle, atuando no mesmo nível de outros protocolos como o ICMP (*Internet Control Message Protocol*), o IGMP (*Internet Group Management Protocol*) ou protocolos de roteamento, conforme mostrado no desenho esquemático da figura 3.3. O gerenciamento ocorre no início da comunicação, sendo reiniciado de tempos em tempos. Caberá ao receptor a responsabilidade em requisitar uma QoS específica. O protocolo RSVP foi feito de forma a garantir que as arquiteturas mais antigas sejam

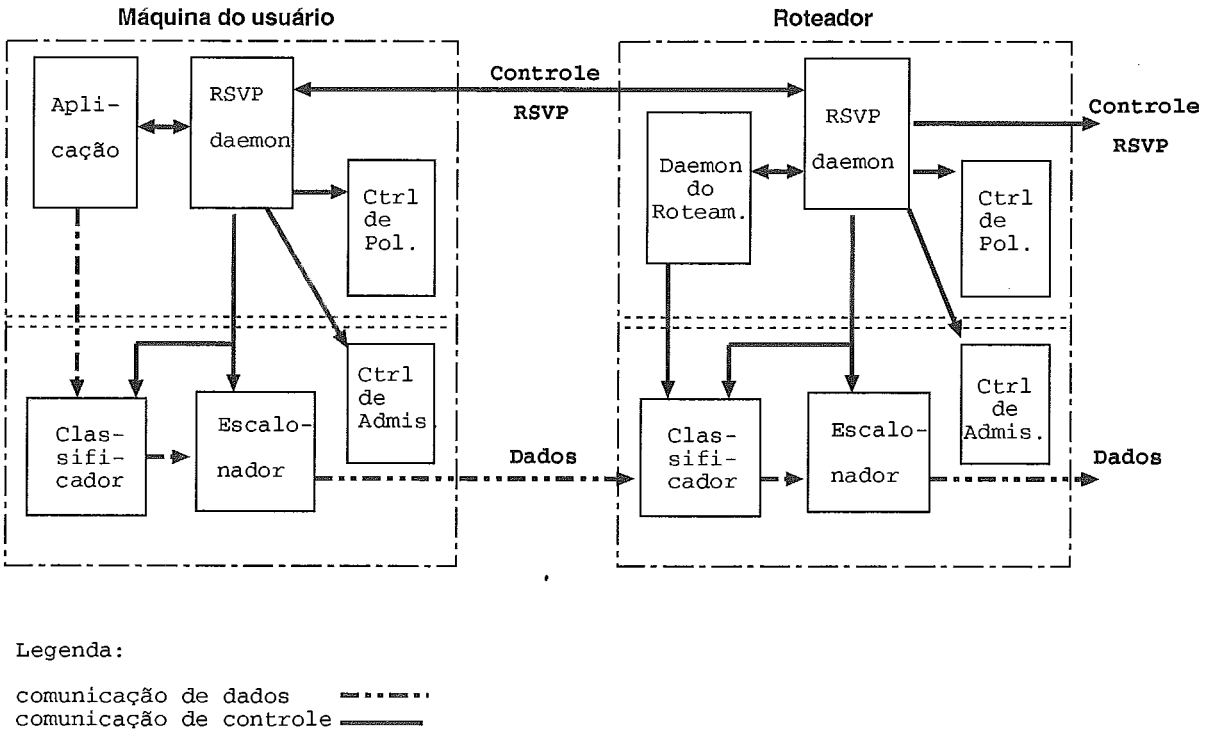


Figura 3.2: RSVP em máquinas do usuário e roteadores.

compatíveis com o novo sistema, através do encapsulamento de seus pacotes de controle.

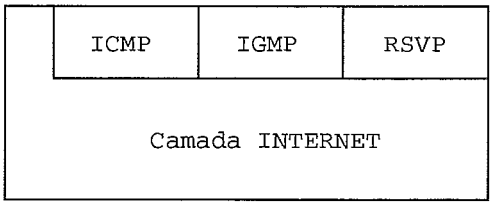


Figura 3.3: Camada de atuação do protocolo RSVP.

A QoS é implementada para um fluxo particular de dados através de mecanismos chamados de controle de tráfego. Estes mecanismos incluem (1) um classificador de pacotes, (2) um controle de admissão e (3) um escalonador. O classificador determina a classe para cada pacote. O escalonador possui a função de alcançar a prometida QoS, realizando tarefas como criação de filas independentes para cada classe.

Durante a reserva, uma requisição de QoS RSVP é passada aos dois módulos de decisão locais: o controle de admissão e o controle de policiamento. O primeiro determina se o nó possui recursos disponíveis para fornecer a QoS desejada. O segundo, se o usuário possui permissão para realizar a reserva. Se ambos os módulos recusarem, a reserva é recusada e o protocolo retorna uma mensagem de erro para o receptor apropriado. Se ambos têm sucesso, o nó usa as informações fornecidas pela aplicação para configurar os módulos do controle de tráfego; o escalonador recebe as informações de QoS apropriadas e o classificador organiza os pacotes das diversas sessões (caso multiponto) em classes a fim de auxiliar a transmissão de dados.

A fim de melhor ilustrar o funcionamento do protocolo, a sessão seguinte foi organizada, contendo os principais conceitos que serão utilizadas ao longo do texto.

3.2.2 Definição de conceitos básicos

Sessão O protocolo RSVP define como *sessão* todo enlace de comunicação pelo qual se relacionam as camadas de transporte de todos os participantes da comunicação podendo ser ponto-a-ponto ou multiponto. Cada sessão é tratada independentemente. O conceito de sessão é propositalmente genérico, pois uma sessão pode ser estabelecida baseando-se em valores de QoS diferentes daqueles requisitados pelo receptor inicialmente. Tal diferença ocorre uma vez que a QoS é montada através de uma série de regras de composição, tendo cada parâmetro a sua própria regra, baseada em informações locais e compostas, estas recebidas dos outros elementos da rede. Como exemplo, podemos imaginar o efeito nos valores de QoS quando em um determinado elemento se encontram diversos fluxos, cada um com valores próprios para seus parâmetros.

Soft-state O protocolo RSVP é baseado na noção de *soft-state*. Este termo foi inicialmente proposto por [18], definindo o "estado" que um determinado elemento pertencente ao percurso de dados de um determinado par fonte-destino se encontra quando uma reserva está estabelecida. O início do *soft-state* ocorre quando uma mensagem de reserva é recebida e realizada no elemento; este estado é periodicamente realimentado pelos receptores. Ao invés de entregar à rede a responsabilidade em detetar e responder a falhas, o RSVP delega aos receptores o trabalho de reenviar periodicamente suas requisições de serviços; caso uma falha ocorra, somente uma nova requisição do serviço restabelecerá o *soft-state* nos roteadores.

OPWA[17] A fim de tornar o protocolo mais ágil em termos de monitoramento dos recursos, o RSVP utiliza-se do conceito de *um passe com advertência.* Inicialmente o RSVP propôs a utilização do modelo de *um passe*, no qual um receptor envia uma requisição de reserva no sentido do transmissor e cada nó no caminho tanto pode aceitar quanto rejeitar a requisição. Neste esquema, o receptor não conhece as condições da rede, portanto não existe flexibilidade para um possível ajuste dinâmico de parâmetros. No novo modelo implementado, chamado de *um passe com advertência* (do inglês *One Pass With Advertising* ou **OPWA**), o protocolo pode controlar os pacotes que são enviados, seguindo o caminho de dados, colecionando informações que podem ser usadas para predizer a QoS. Os resultados, ou advertências, são entregues pelo protocolo ao receptor ou à aplicação em mensagens especiais. Com isso, pode-se construir ou ajustar dinamicamente uma requisição de reserva apropriada.

3.2.3 Mensagens

Uma série de mensagens[1] (ou objetos do RSVP) devem ser trocadas entre as aplicações e os elementos de rede para corretamente requisitar a qualidade de serviço para uma determinada sessão.

Uma sessão é identificada por três informações: (end_destino, protocolo, porta_destino). End_destino é o endereço IP (ponto-a-ponto ou multiponto) do destino do fluxo de dados. O protocolo poderá ser UDP ou TCP e o campo porta_destino é opcional, no caso ponto-a-ponto. Se houver necessidade de estabelecer sessões entre diferentes aplicações de um mesmo grupo multiponto, o campo porta_destino deverá ser explicitado.

Após a definição da sessão, serão trocadas mensagens de controle RSVP. As mensagens **RSVP** fundamentais são **RESV** e **PATH**. Cada uma deverá explicitamente requisitar a QoS através de objetos descritores do tráfego e filtros. Dependendo do serviço de QoS, se através de carga-controlada ou serviço garantido, esses objetos possuirão alterações a fim de comportar os diversos parâmetros de interesse. Os principais são[10]:

[1] Além dessas mensagens, utilizadas para requisitar qualidade de serviço diretamente, o protocolo RSVP transporta informações de autenticação, segurança e auditoria, a fim de gerenciar o uso desses outros serviços. Essas mensagens não serão especificadas neste documento.

SENDER_ TSPEC - da expressão em inglês *sender traffic specification* - especificação do tráfego gerado pelo transmissor. Este objeto RSVP é gerado no transmissor, seguindo por todos os elementos intermediários até chegar ao receptor. Seu conteúdo nunca é modificado por nenhum dos membros envolvidos no processo, chegando incólume ao seu destino.

FLOWSPEC - do termo em inglês *flow specification* - especificação do fluxo. As informações geradas por cada receptor descrevendo o serviço QoS desejado, (como carga controlada ou serviço garantido), a descrição do fluxo de tráfego ao qual deve-se ativar a reserva de recursos e quaisquer outros parâmetros que sejam utilizados nesse fim são transportados pelo objeto *flowspec*. Esse objeto é transportado pela mensagem RESV sendo seu sentido sempre para o receptor, passando pelos elementos intermediários, até chegar ao transmissor da comunicação. As informações transportadas por esse objeto podem sofrem mudanças em seu conteúdo durante o trajeto, mediante união de fluxos ou outros fatores como regras de composição dos parâmetros de QoS. Seu conteúdo será tratado pelo escalonador de pacotes.

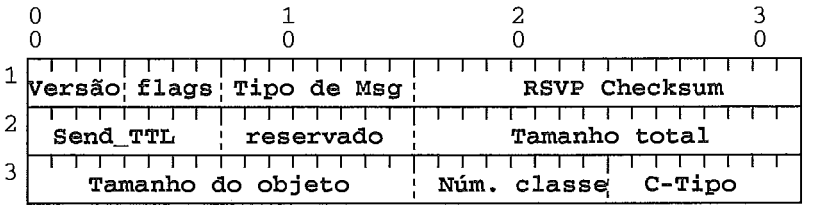
FILTER SPEC - especificação de filtro. Contém o endereço IP de um (ou conjunto de) transmissor(es). Seu conteúdo depende do estilo de reserva. Existe uma relação unívoca entre a especificação do conjunto de pacotes de dados - o fluxo - que receberá a reserva definida pelo objeto *flowspec* e este objeto. Suas informações serão processadas pelo classificador de pacotes e utilizadas na execução das regras de composição dos parâmetros de QoS. Mais detalhes sobre este objeto serão vistos na seção de estilos de reserva.

ADSPEC - da expressão *advertising specification* - especificação de advertência. Esse objeto é montado com informações da rede e é utilizado pelos receptores para fazer decisões de reserva. Estas informações podem incluir serviços disponíveis, estimativas de atraso e banda e parâmetros de operação (por exemplo número de roteadores RSVP no percurso) usados por serviços específicos de QoS. São coletadas na rede e transportadas até o receptor pelo objeto *adspec*. Elas representam um resumo, considerando as condições em cada elemento do percurso.

O formato e o conteúdo de cada objeto, não são de responsabilidade do protocolo RSVP. Devido à sua importância para o entendimento do protocolo, os objetos

acima foram descritos com mais detalhes no **Apêndice 1.**

Toda mensagem RSVP possui um cabeçalho comum com o formato representado na figura 3.4, com uma descrição sucinta dos diversos campos:




```
Campos:
Versão do protocolo (valor = 1)
Flags (nenhum foi definido até essa versão)
Tipo de Msg (RESV, PATH, RESVCONF, etc.)
Controle
Valor do TTL com o qual a msg foi enviada
Tamanho total em bytes, incluindo o cabeçalho comum
Tamanho do objeto - tamanho que virá a seguir.
Núm. Classe identifica o objeto a seguir
      (estilo, sessão, flowspec, etc.)
C-Tipo - tipo da classe. Permite a utilização de
       várias formas do mesmo objeto (ex. IPv4/IPv6)
```

Figura 3.4: Cabeçalho comum a todas as mensagens RSVP.

A mensagem RESV, enviada pelo receptor, é constituída do par (flowspec,filter spec). Ao par dá-se o nome de descritor de fluxo. Esta mensagem é ponto-a-ponto seguindo do receptor até o transmissor. O objeto *flowspec* define a QoS desejada, enquanto que o *filter spec*, aliado com a identificação da sessão, define o conjunto de pacotes de dados, i.e. o fluxo, que receberá a QoS requisitada no *flowspec*. As informações contidas no *flowspec* serão usadas pelos mecanismos de escalonamento, enquanto que as informações contidas no objeto *filter spec* serão usadas pelo classificador de pacotes. Àqueles pacotes de dados de uma sessão que não se enquadrem na especificação do fluxo obtida, não será empregado nenhum controle de QoS.

A mensagem PATH, enviada pelo transmissor, é propagada pelo caminho ponto-a-ponto ou multiponto, seguindo a rota informada pelos mecanismos de encaminhamento até os receptores. Um elemento no caminho dos dados, ao receber um PATH, criará um estado chamado ***PATH state***. As mensagens de PATH armazenam o estado de cada nó (contido no objeto *adspec*) por onde ela transitou. Ela contém os seguintes objetos: o *sender_tspec*, o *sender_template* e o *adspec*. O *sender_template*

contém o endereço IP da fonte; o *sender_tspec* contém a especificação do tráfego gerado pela aplicação e o *adspec* informa sobre o estado da rede.

Mediante à troca dessas mensagens, o protocolo toma uma série de decisões, como por exemplo, aceitar ou não um novo fluxo, criando um ambiente para que os recursos sejam reservados. Cada parâmetro utilizado para a requisitar a QoS está representado nas mensagens do RSVP.

Através da análise dos parâmetros trazidos pelos objetos RSVP, algumas simplificações podem ser feitas, de forma a agrupar fluxos distintos de uma mesma sessão que possuam características comuns. Tal tarefa, apesar da complexidade, proporciona, quando é possível, uma economia dos recursos utilizados, principalmente em termos de largura de banda. Para que isso possa ocorrer, é essencial a utilização dos estilos de reserva, assunto que será abordado em seguir.

3.2.4 Estilos de Reserva

Uma requisição de reserva inclui, além da QoS desejada, a opção sobre o estilo de reserva. Através dos estilos, o protocolo RSVP apresenta uma forma mais elaborada de tratar os recursos disponíveis sobre seu controle, tornando possível uma economia dos mesmos se assim permitirem os elementos envolvidos no processo. A necessidade dos estilos de reserva é mais evidente quando em presença de uma comunicação multiponto. Como exemplo, ao invés de reservar recursos para cada palestrante em uma teleconferência, uma única reserva compartilhada pode ser estabelecida de forma que seja suficiente para que algumas pessoas conversem simultaneamente. Todo o grupo multiponto formado para atender aos participantes, compartilharia uma única reserva.

A presença de múltiplos receptores em uma aplicação favorece a união dos fluxos. Quando dois receptores requisitam uma reserva para o mesmo fluxo, suas requisições podem ser combinadas numa única reserva em todos os enlaces do caminho que forem comuns a ambos os receptores. A união nesse caso, deverá fornecer uma QoS para cada um dos participantes, no mínimo igual ou tão boa quanto seria para cada um deles individualmente.

Pelo RSVP, há duas formas de tratamento de reserva para diferentes transmissores dentro da mesma sessão: estabelecimento de reservas distintas para cada transmissor

ou fazer uma reserva que seja compartilhada entre todos os pacotes dos transmissores selecionados.

Outra opção de reserva controla a seleção de transmissores. Esta opção pode ser com uma lista explícita de todos os transmissores de uma sessão ou um asterisco (em inglês *Wildcard*) que implicitamente seleciona todos os transmissores de uma determinada sessão. Em uma opção seleção explícita do transmissor, cada *filter spec* deve concordar exatamente com um transmissor enquanto que no esquema *wildcard* nenhum *filter spec* é necessário. Os estilos de reserva estão representados na tabela 3.1 abaixo.

Tabela 3.1: Estilos de Reserva.

Seleção do trasmissor	Reserva Distinta	Reserva Compartilhada
Explícita	Estilo FF	Estilo SE
Qualquer	Não definido	Estilo WF

Os seguintes estilos estão definidos:

1) Estilo Fixo (FF)

O estilo **FF** é composto das opções: reserva distinta e seleção explícita do transmissor, conforme representado na tabela anterior. Esse estilo indica que uma reserva feita para os pacotes de dados de um transmissor particular, não poderá ser compartilhada com outras fontes da mesma sessão.

A simbologia:

FF (SQ),

onde **S** representa o transmissor selecionado e **Q** é a *flowspec* correspondente. O par forma um descritor de fluxo. No exemplo da figura 3.5-(1) ilustrado, temos dois receptores **R1** e **R2** que trocam informações com um transmissor **T** em comum. Apesar de os fluxos fazerem parte da mesma sessão, o roteador não está autorizado a empregar nenhum tipo de mecanismo que permita o compartilhamento dos recursos, utilizando portanto dois canais independentes, um para cada fluxo de dados.

2) Estilo Compartilhado Explícito (SE)

Este estilo engloba as opções: reserva compartilhada e seleção explícita do transmissor. Essa configuração cria uma reserva única sendo compartilhada pelos transmissores selecionados. Diferente do estilo **WF**, este permite ao receptor especificar o conjunto de transmissores a serem incluídos.

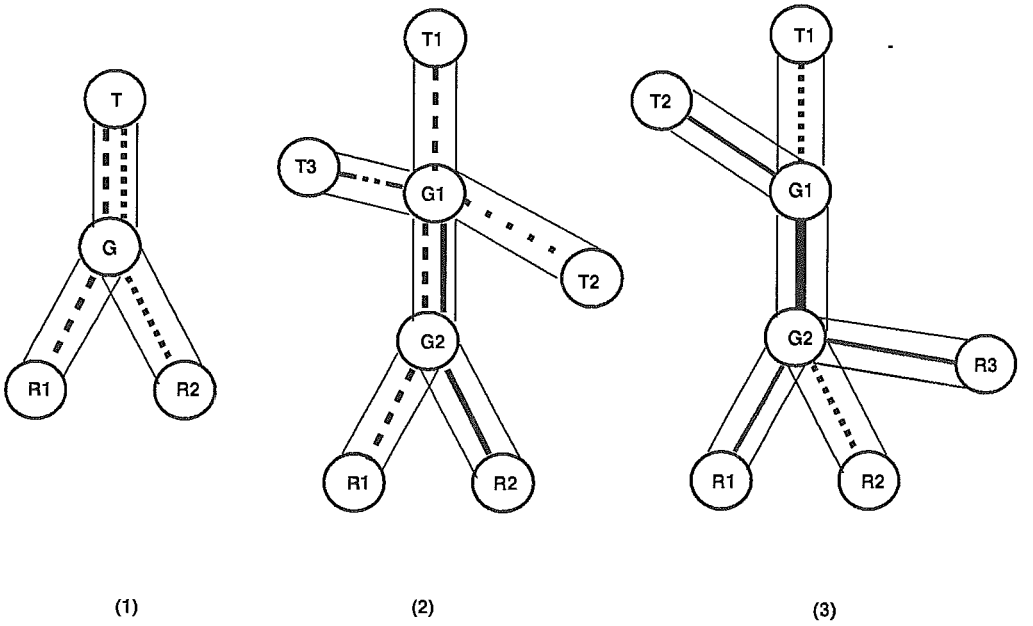


Figura 3.5: Exemplos de estilos de reserva: (1) FF, (2) SE e (3) WF.

A simbologia é esta:

SE ((S1, S2, ...)Q),

onde S1, S2, ... representam os transmissores e Q o *flowspec*.

Vejamos o esquema da figura 3.5-(2):

Temos um cenário onde R2 deseja que o fluxo que venha tanto de **T2** quanto de **T3** tenham as mesmas condições de reserva para uma dada sessão. Assim sendo os fluxo svindos de **T2** e **T3** são compartilhados ao longo do caminho comum a ambos os fluxos. Note-se que tal esquema não foi imposto ao fluxo do receptor **R1**, cujo transmisssor é **T1**, uma vez que este se mantém independente ao longo do caminho percorrido.

3) Estilo *Wildcard* (WF)

O estilo *wildcard* é composto pelas opções: reserva compartilhada e seleção do transmissor "qualquer". Este estilo cria uma reserva única a ser compartilhada por todos os fluxos de todos os transmissores da sessão, isto é, não importa qual seja o transmissor. Nesse caso o protocolo deve tentar realizar o compartilhamento máximo dos fluxos. Pode ser pensada como um túnel compartilhado cujo tamanho é o maior de todos os pedidos por todos os receptores, conforme a regra de composição do parâmetro largura de banda, independente do número de transmissores que o estejam

usando.

Simbolicamente pode ser representada como:

WF(*Q),

onde o asterisco representa a seleção *wildcard* e Q representa o *flowspec*, ou a especificação de fluxo. Um exemplo pode ser visualizado na figura 3.5-(3):

Aqui, para esse cenário apresentado, qualquer composição de parâmetros de QoS que for possível realizar estará permitida (de acordo com as regras de composição), pela seleção WF do transmissor. No caso, os três receptores **R1**, **R2** e **R3** aceitaram compartilhar os recursos com os fluxos vindos dos transmissores **T1** e **T2**, implicando que entre os roteadores **G1** e **G2** exista um melhor aproveitamento dos recursos (como por exemplo em termos de largura de banda). No item a seguir foi criado um exemplo completo do funcionamento do protocolo.

Reservas compartilhadas, no caso os estilos **SE** e **WF**, são próprias para aquelas aplicações multiponto em que múltiplas fontes não podem, ou não precisam, transmitir simultaneamente. Áudio é um exemplo, uma vez que um número limitado de pessoas falam ao mesmo tempo. Por outro lado, o estilo **FF**, que cria reservas distintas para o tráfego entre diferentes transmissores, é apropriado para sinais de vídeo.

As regras do protocolo não permitem união entre tráfegos com reservas compartilhadas e reservas distintas, devido à característica antagônica dos mesmos. Também há a proibição de reunião de tráfegos com seleção explícita ou qualquer de transmissores. Assim sendo, os estilos **WF**, **SE** e **FF** são mutuamente exclusivos.

3.2.5 Modelo de reserva

Vejamos um cenário simples, ponto-a-ponto, em que todos os elementos compreendam o protocolo RSVP. Em cada nó roteador, além do gerenciador de recursos, responsável pela reserva, existem módulos de controle de tráfego e policiamento que auxiliam o RSVP na tarefa de reservar os recursos requisitados, baseado no esquema da figura 3.2. Como pré-requisito, uma sessão RSVP deverá ser instalada.

Seja **T1** uma máquina que contém uma aplicação que tenha requisitos de QoS, **R1** outra máquina que receberá dados dessa aplicação e **G** roteadores intermediários. A aplicação em **T1**, para iniciar sua transmissão, envia uma mensagem de controle

chamada de mensagem de caminho (ou *path message* ou **PATH**), que seguirá seu caminho, pelos diversos roteadores até chegar em **R1**, conforme figura 3.6. A mensagem **PATH** contém um cabeçalho RSVP e todas as informações sobre o tráfego que a aplicação em **T1** espera gerar, i.e, valores para os parâmetros de QoS. A cada roteador **G** existente entre **T1** e **R1**, será criado um estado chamado de **PATH** *state.*

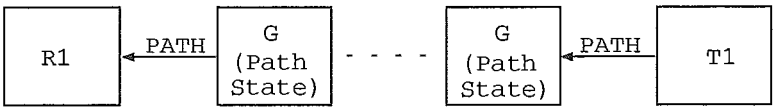


Figura 3.6: Mensagem PATH seguindo de T1 a R1 .

Ao chegar em **R1**, este analisa as informações contidas em **PATH** e seleciona os parâmetros de reserva desejados, montando a mensagem de reserva (ou *reservation message* ou **RESV**). Por onde passar, a **RESV** provocará nos roteadores um estado chamado ***SOFT state***, até chegar de volta em **T1**, informando-o sobre as condições dos parâmetros de QoS da reserva realizada por **R1** e propriedades do caminho entre ambos, como é mostrado na figura 3.7.

Figura 3.7: Reserva entre **R1** e **T1**, criando o *Soft State* nos elementos intermediários.

Caso multiponto:

No caso de comunicação multiponto, um membro, digamos **R1**, envia uma mensagem IGMP para juntar-se ao grupo multiponto. Sendo aceito no grupo, receberá mensagens de **PATH** contendo as advertências quanto ao cenário atual no qual se encontra a rede em termos dos parâmetros de QoS. Concordando em fazer parte daquela sessão RSVP, o membro passará a registrar o **PATH-state**. Ao ser requisitada comunicação por um transmissor ou conjunto de transmissores, este membro então enviará uma mensagem de **RESV** para reservar os recursos ao longo do caminho, traçado pelo encaminhamento, de volta até os transmissores, conforme o exemplo acima, ilustrado nas figuras 3.6 e 3.7. A reserva seria realizada segundo o estilo de reserva escolhido pelo receptor e conforme as possibilidades de composição dos parâmetros de QoS em cada nó do caminho entre R1 e **T1**, sendo o resultado

entregue a **T1**. Dependendo do resultado, **T1** poderá decidir manter a comunicação ou não.

Pelo modelo de reserva criado, é o receptor o responsável por requisitar a QoS desejada, conforme sua conveniência, entre os dois tipos de serviços de controle de qualidade de serviço: com carga-controlada[2] ou serviço garantido[3]. O protocolo RSVP se encarrega de passar essa informação para todos os nós (ou roteadores) no caminho reverso, até chegar ao transmissor, mantendo o "*Soft state*". Em cada nó intermediário ou roteador, o processo RSVP comunica-se com dois módulos de decisão: o de controle de admissão e o de policiamento.

A requisição de reserva que um nó receptor envia para o nó transmissor pode variar da requisição inicial recebida por duas razões:

1. Os nós intermediários não puderam fornecer a QoS desejada;
2. As reservas para um mesmo transmissor, ou conjunto de transmissores, de diferentes ramos da árvore multiponto, conforme o estilo de reserva requisitado, podem ser agrupadas. Logo, quando um nó receptor envia para o seu transmissor os valores de recursos desejados ao longo do caminho, estes são combinados em cada nó de acordo com a regra estabelecida para cada parâmetro (por exemplo, no caso da largura de banda, apenas a maior banda ao longo do caminho é fornecida a todos os participantes).

Quando o receptor inicia uma requisição de reserva, ele também pode pedir uma mensagem de confirmação indicando que sua requisição está instalada na rede. Vejamos o exemplo ilustrado na figura 3.8.

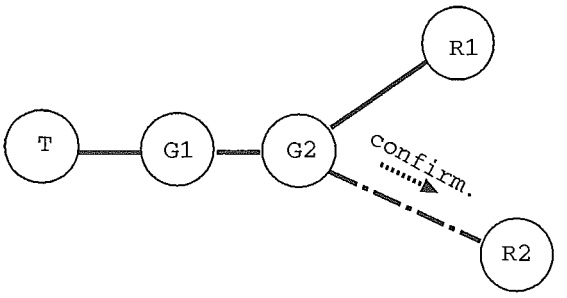


Figura 3.8: Exemplo da mensagem de confirmação.

Digamos que já exista um ambiente de reserva estabelecido entre **T** e **R1**, passando pelos roteadores **G1** e **G2**. Supondo que uma nova requisição, com pedido de

confirmação vinda de **R2**, é propagada pela árvore multiponto até alcançar **G2** onde a reserva já existente é igual ou maior (em termos de banda por exemplo) do que a reserva requisitada. A partir daí, o fluxo é unido (*merged*) com as reservas já presentes, não havendo necessidade de configuração nos próximos pontos. Nesse momento, uma mensagem de confirmação, vinda de **G2** é enviada de volta ao receptor. Essa mensagem não caracteriza uma realização efetiva da reserva, porém uma alta probabilidade desse fato.

Nessa seção, os detalhes mais importantes do protocolo RSVP foram revistos. Maiores informações podem ser obtidas na especificação do protocolo em [9].

3.3 Interface RSVP x Encaminhamento

Para que a tarefa de reservar os recursos ao longo de um caminho seja bem sucedida, o RSVP precisa de alguns pré-requisitos. A aplicação no transmissor é responsável por definir os parâmetros que representam as características de seu tráfego, como atraso ou banda, realizando seu serviço da forma pedida pelo usuário. Ao conjunto de parâmetros requisitados pela aplicação dá-se o nome de Qualidade de Serviço.

Nesse momento, o protocolo de encaminhamento, sensível à Qualidade de Serviço de cada um dos possíveis caminhos entre destino e fonte, elege aquele que lhe atenda na totalidade, ou mais próximo disso, dos parâmetros requisitados pela aplicação. De posse do caminho eleito, o RSVP poderia então negociar a reserva ao longo deste trajeto.

Um ponto importante, é a escolha de parâmetros que comporão a Qualidade de Serviço. Segundo [19], considerar muitas características de tráfego para uma única rota conduz a um problema NP-completo. Em seu trabalho é demonstrado que, na prática, ao se considerar todos os principais parâmetros de tráfego, a saber: retardo, jitter, probabilidade de perda de pacotes, custo do caminho (métrica) e banda passante, devido a regras de composição dos parâmetros, o custo para se achar um caminho adequado torna-se NP-completo.

Ele mostra ainda que somente com dois desses parâmetros em conjunto, que seriam banda passante e algum dos outros quatro parâmetros, ter-se-ia uma solução de ordem polinomial para o problema. Como fator mais importante é eleito o atraso (pelo menos para as aplicações atuais de tempo real e multimídia) sendo que pelo

algoritmo apresentado, é indistinta a escolha tanto de jitter como custo para o segundo parâmetro, devido às regras de composição desses parâmetros.

No entanto, o cálculo da QoS nos diversos caminhos requer uma estabilidade de rotas, visto que a mudança de um caminho antes satisfatório por outro mais adequado do ponto de vista do roteamento, pode conduzir o protocolo RSVP à impossibilidade de manter uma reserva já estabelecida. Também destaca-se que o ambiente onde essas aplicações atua é basicamente multiponto, onde a comunicação é tratada dentro de um ou mais grupos de máquinas.

As principais informações que este algoritmo de roteamento deveria dispor para a realização da reserva são as seguintes:

- fornecer em tempo real as entradas na tabela de roteamento para um par fonte-destino e
- providenciar notificação caso haja alteração das rotas já estabelecidas.

Sendo um protocolo de reserva de recursos, o RSVP necessita de um protocolo de encaminhamento adequado, que permita informar às aplicações sobre as condições de QoS ora instaladas ao longo do caminho entre ela (aplicação fonte) e o(s) possível(is) destino(s).

No entanto, nada impede que a aplicação tenha mecanismos particularizados de escolha de QoS para cada um dos pares fonte-destino, possibilitando com isso, uma ação mais rápida e efetiva em um momento de mudança da topologia da árvore de caminho mínimo. A aplicação então comporia uma configuração para seus parâmetros locais de QoS, com maiores chances de sucesso de implantação pelo gerenciador ao longo tanto do percurso sobre a árvore de caminho mínimo, quanto para as outras possibilidades de rotas, permitindo uma rápida ação diante de uma mudança futura. Esta configuração de parâmetros mais próxima das condições reais da rede seria propagada pelos elementos intermediários, criando condições para que uma possível reserva seja efetivada ou, na presença de uma alteração de topologia, apenas aqueles elementos modificados negociariam as alterações de QoS entre si, propagando seus resultados aos demais elementos do caminho.

Baseada nos princípios acima descritos, foi criada uma interface entre o protocolo de gerenciamento de recursos e o protocolo de roteamento. A interface permite ao RSVP requisitar informações e serviços do roteamento na forma de um protocolo

assíncrono tipo pergunta-e-resposta, além de receber informações atualizadas sobre a QoS ao longo dos possíveis trajetos fonte-destino sempre que alguma condição ou rota seja alterada. A ela foi dado o nome de RSRR, da expressão em inglês *Routing Support for Reservation Resources.* Com essa interface, o RSVP pode atuar juntamente com um algoritmo de roteamento que não seja sensível a QoS requisitada pela aplicação.

O RSRR atualmente inclui dois pares de mensagens. O processo RSVP envia uma mensagem quando inicia suas operações, antes de receber qualquer pedido de reserva. O protocolo de roteamento envia de volta uma resposta que inclui o conjunto de interfaces virtuais que está utilizando naquele momento, indicando aquelas desabilitadas. Quando o RSVP necessita então uma entrada da tabela de roteamento que contenha determinada dupla fonte-grupo, ele envia uma outra mensagem específica. A resposta do protocolo de roteamento vem incluindo qual a interface virtual de entrada e as possíveis interfaces virtuais de saída para atender àquele par fonte-grupo solicitado. A interface virtual é simplesmente um número de identificação para referenciar a interface de entrada/saída. A implementação da interface é transparente para o RSVP. Além disso, uma mensagem de notificação, caso haja alguma mudança de topologia ou na tabela de roteamento, deverá ser fornecida pelo encaminhamento, permitindo ao RSVP reenviar mensagens de PATH e RESV nos ramos da árvore necessários, permitindo recompor as informações de seus parâmetros de QoS.

Mais informações podem ser obtidas em [20].

3.4 ST-II[11]

A ISPN tem dois objetivos: proporcionar uma forma eficiente para que as aplicações requisitem qualidade de serviço e prover comunicações multiponto. Assim sendo, os protocolos de gerenciamento de recursos precisaram incorporar mecanismos para a realização de comunicação multiponto. Basicamente, o protocolo ST-II controla uma reserva de recursos como uma via simplex de dados partindo da fonte para todos os receptores através de uma árvore de distribuição multiponto. Todas as operações de gerência do grupo são realizadas através de mensagens trocadas entre a fonte e os participantes do grupo multiponto. A dinâmica do grupo é atualizada pelos receptores ao enviar mensagens para a fonte. Cabe aos receptores enviar pedidos para inclusão no grupo e esperar por mensagens de permissão ou rejeição que será

conferida pela fonte. Sempre que a fonte receber uma mensagem de aceitação vinda do receptor, com uma especificação de fluxo menor do que ela pretenda trabalhar, ela tanto pode adaptar-se para funcionar com uma QoS mais baixa ou enviar uma mensagem de rejeição de volta ao receptor.

Confiabilidade e robustez do protocolo são incorporadas através de dois mecanismos separados. Primeiro, todas as mensagens de controle são enviadas usando confirmação ponto-a-ponto com retransmissão. Segundo, o protocolo *Hello* é usado para questionar o estado de agentes ST vizinhos que compartilham o mesmo fluxo.

A principal diferença entre a proposta do ST-II e o RSVP reside no nível de comunicação multiponto proporcionado pela rede. O ST-II constrói uma árvore de distribuição multiponto baseada nas tabelas de roteamento, realizando então o envio de pacotes. Já o protocolo RSVP, desvincula-se do roteamento multiponto, assumindo que a rede será responsável pelo envio dos pacotes de dados. Esse fato deve-se basicamente ao fato de que nenhum protocolo de roteamento multiponto estava disponível quando o ST-II foi proposto. Mais detalhes sobre a comparação entre os dois protocolos está descrita em [12].

3.5 Controle de Tráfego

Além das propriedades específicas que os protocolos de encaminhamento devem fornecer para a realização da reserva em rede, outras funções devem ser utilizadas para se chegar ao objetivo de conseguir e manter a qualidade de serviço requisitada pelas aplicações. Conforme mostrado na figura 3.2, o RSVP atua em conjunto com outros módulos, ditos de controle de tráfego, a saber: escalonador de pacotes, classificador de pacotes e controle de admissão. Estes atuam tanto nos elementos finais (transmissores e receptores) quanto nos roteadores ao longo do caminho. A seguir foram destacadas as principais características dos módulos de controle de tráfego.

3.5.1 Escalonamento de pacotes

A função básica do escalonamento é implementar uma política para servir os pacotes na fila de saída. O esquema mais comumente utilizado é o **FIFO** (do inglês *First In First Out*). Nele, os pacotes são servidos estritamente na ordem de chegada.

Entre os muitos esquemas propostos, talvez o mais simples seja um esquema de prioridades, onde o pacote que tenha maior prioridade seja servido em primeiro lugar. Um problema inerente a essa solução é que na presença de pacotes de alta prioridade, aqueles que não possuem nenhum tipo de privilégio podem ser atrasados em muito no seu trajeto. Pode-se citar o esquema de *Weight Fair Queueing* ou **WFQ** que conseguiria uma alocação mais igualitária, distribuindo pesos aos diversos fluxos de saída. Outro mecanismo de escalonamento que podemos citar é o esquema que trabalha com classes: **CBQ** (do termo *Class Based Queue*). Nele, os pacotes dos diversos fluxos são enquadrados em diversas classes, organizadas de forma a criar um esquema de prioridades entre os fluxos de dados da saída. Maiores informações podem ser conseguidas em [22].

3.5.2 Controle de Admissão

O controle de admissão implementa um algoritmo de decisão que um roteador ou uma máquina utiliza para determinar se um novo fluxo de dados poderá ser aceito ou não. A decisão deverá ser tomada de forma a não comprometer os fluxos previamente aceitos pelo roteador. O controle de admissão é evocado em cada ponto para fazer uma decisão simples: aceitar ou rejeitar, no momento que a requisição do fluxo chega a esse ponto.

Assim, o roteador do fluxo que está fazendo a requisição. Ele deve ser capaz de calcular a capacidade efetiva do conjunto de fontes admitidas de forma a que a QoS seja respeitada.

É notável que apesar de o controle de admissão ser parte de um modelo global, os detalhes dos algoritmos utilizados por cada roteador são locais.

3.5.3 Classificador de Pacotes

A discussão acima sobre escalonamento e controle de admissão presumem que os pacotes de uma sessão foram classificados de acordo com algum critério de reserva. No caso de existir um outro fluxo com a mesma classificação, estes serão unidos caso o estilo da reserva permita ou esta nova seqüência de pacotes deverá ser tratada de forma específica. Um pré-requisito faz-se necessário então: uma forma de classificação dos pacotes. A seleção é feita geralmente olhando-se exclusivamen-

te o endereço de destino dos pacotes. Com a introdução de qualidade de serviço, requer-se mais informação para a realização dessa tarefa.

Em uma rede que implementa circuitos virtuais, a QoS do fluxo é conhecida no momento do estabelecimento da conexão. Todos os pacotes subseqüentes deste fluxo serão identificados pelo número do circuito virtual, o que torna mais fácil a classificação dos pacotes. Outro método, é permitir ao módulo classificador identificar mais campos da área de cabeçalho dos pacotes, tais como endereço da fonte, porta e número do protocolo. Uma determinada seqüência de vídeo, por exemplo, poderia ser reconhecida por uma porta particular. Aprofundando tal busca, poderíamos obter um método onde seria possível permitir a investigação mais rigorosa dos pacotes, talvez até a área de dados, a fim de identificar alguma característica da aplicação.

As implementações de um classificador são complexas e incrementam o custo do processo. Uma proposta para reduzir o custo seria proporcionar um novo campo de identificação de fluxo no cabeçalho do pacote, por exemplo na camada IP. O valor desse campo poderia ser guardado e usado como um classificador do pacote como um todo. Alguns algoritmos tem-se mostrado eficientes, permitindo inclusive a possibilidade de adicionar suporte de qualidade de serviço à aplicações que não tenham natureza de vídeo ou voz como telnet, que são baseadas em tecnologia já existente.

Os módulos de controle de tráfego serão melhor abordados no capítulo 4 a seguir.

3.6 Serviços Diferenciados

Nos últimos anos, o desenvolvimento das aplicações com requisitos de qualidade de serviço levou à criação de uma arquitetura de serviços integrados, além do protocolo RSVP para eficientemente construir e manter o estado da rede para as aplicações que necessitassem seguir tal arquitetura. Entretanto, com o passar do tempo, algumas limitações puderam ser constatadas, impedindo o crescimento dessa tecnologia no âmbito geral:

- O modo de funcionamento do RSVP, que realiza processamento por fluxo, implica o gerenciamento de muitas mensagens à medida que se trabalha com topologias maiores.

- Ainda hoje, um pequeno número de elementos da rede geram sinalização RSVP. Enquanto espera-se que este número cresça amplamente, muitas aplicações já existentes podem nunca ter uma interface de trabalho desenvolvida para gerar RSVP.

- Muitas aplicações requerem algum tipo de qualidade de serviço porém são incapazes de expressar essas necessidades segundo o conjunto de parâmetros fornecido pelo modelo de serviços integrados.

Atualmente, a pressão pelo desenvolvimento de uma tecnologia associada que requeira qualidade de serviço levou ao surgimento de uma nova abordagem aos serviços integrados: os serviços diferenciados. Em contraste com a orientação do RSVP, que é baseado na gerência de fluxos, as redes *diff-serv*, como são chamadas, classificam em fluxos agregados, todos pacotes que possuam o mesmo valor para o campo **ToS** (*Type of Service*) do cabeçalho IP. Assim sendo, além de eliminar a dependência da criação de um estado por fluxo, a abordagem *diff-serv* para montar a qualidade de serviço pode ser conseguida usando o próprio IP, sem nenhuma necessidade de sinalização fim-a-fim.

Em termos de topologia, foram consideradas duas regiões distintas: uma onde todos os elementos fazem uso dos serviços integrados, isto é, onde se usa a identificação de fluxos aos quais se dispõe de sinalização (através do RSVP), controle de admissão e escalonamento para entregar as garantias de QoS e outra, onde se proporciona apenas serviços de QoS agregados (Figura 3.9).

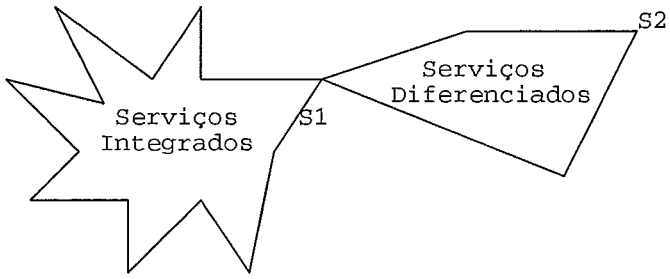


Figura 3.9: Esquema de duas regiões da Internet com diferentes tipos de tratamento de QoS.

Na figura acima, o roteador **S1** está localizado na fronteira entre as duas regiões. Ele possuirá a função de intermediar as duas regiões possuindo características especiais para tal. O roteador **S2**, por estar contido na região exclusiva de serviços

diferenciados, possui funções de controle de admissão, escalonamento e classificação dos pacotes. Mais detalhes podem ser encontrados em [20].

Capítulo 4

Ambiente para realização dos experimentos

Com o objetivo de verificar o ganho efetivo de aplicações de tempo real, em termos dos parâmetros de desempenho como jitter e perda de pacotes, na presença do gerenciador de recursos, um experimento completo foi realizado. A análise dos dados foi feita comparativamente aos resultados obtidos quando o ambiente não possuía qualquer controle sobre as comunicações. Para compor o ambiente de testes, foi necessário o estudo e aprofundamento dos módulos acessórios ao RSVP de controle de tráfego que permitem criar condições diferenciadas para o fluxo de dados ao qual se deseja Qualidade de Serviço. O presente capítulo contém um item dedicado a base teórica que serviu para a criação do pacote ALTQ (do inglês *Alternate Queueing*) responsável pela classificação e escalonamento da saída de dados, permitindo criar prioridades para os diversos fluxos. Além disso, apresentaremos uma análise mais detalhada sobre os parâmetros de Qualidade de Serviços explicitamente requeridos pelo gerenciador de recursos e sua interface com o escalonador. Os programas aplicativos desenvolvidos para gerar tráfego de características de tempo real e medir parâmetros de desempenho também estão relacionados.

4.1 Controle de tráfego

Neste capítulo será apresentada a descrição do módulo de controle de tráfego que foi utilizado juntamente com o protocolo de reserva (RSVP) para prover a QoS requisitada pelas aplicações. O módulo de controle de tráfego está reunido em um pacote de software chamado ALTQ[25]. Ele foi desenvolvido baseado nos mecanismos propostos em [22] para classificação dos fluxos de tráfego e distribuição da capacidade do canal entre os diversos fluxos de tráfego.

4.1.1 Estrutura geral do ALTQ

Efetivar o controle preciso e gerenciamento do tráfego gerado por uma aplicação de tempo real, tem se mostrado como forma eficaz de garantir uma qualidade mínima de inteligibilidade à mensagem, à medida que se prioriza esse tráfego entre outros. Em essência, tal esquema necessita de algum mecanismo para controlar o tráfego gerado pela aplicação e de um mecanismo de escalonamento para evitar congestionamento e proporcionar garantias para os diversos fluxos.

Muitas disciplinas para escalonamento são descritas na literatura[31], e suas vantagens conhecidas através de simulação de ambientes. No entanto, a implementação de esquemas capazes de escalonar tráfegos reais, esbarrava na utilização de equipamentos caros, ou adaptação de roteadores já existentes. A alternativa utilizada pelo módulo ALTQ foi acoplar um controle de saída de pacotes no *kernel* do sistema operacional BSD UNIX (que tem ampla utilização nos dias atuais). Tradicionalmente, o *kernel* do sistema BSD UNIX implementa a disciplina FIFO para o controle de saída dos pacotes e a incorporação de outras disciplinas de escalonamento à saída da interface de rede tornou-se relativamente simples. O pacote ALTQ veio nesse sentido, habilitando a criação de diversas estruturas de escalonamento, com diferentes componentes: estratégias de escalonamento, esquemas para o descarte de pacotes, alocação de buffers, múltiplos níveis de prioridade e filas de trabalho não conservativo. Além disso, foram desenvolvidos mecanismos de filtragem e classificação de fluxo através da observação do cabeçalho IP dos pacotes. Portanto, o ALTQ realiza como principais funções a classificação e o escalonamento de pacotes. Uma função simples de controle de admissão também foi implementada.

O ALTQ implementa duas políticas de escalonamento relevantes para o trabalho apresentado: WFQ (*Weighted Fair Queueing*) e CBQ (*Class-Based Queueing*).

A comunicação entre o ALTQ e o módulo gerenciador RSVP é feita de forma transparente se a política CBQ é utilizada. Ou seja, baseado em um parâmetro de QoS transportado pela mensagem RESV, o ALTQ irá criar uma classe para o tráfego da sessão especificada na mensagem de RESV. Para tal, é necessário que o ALTQ seja configurado previamente de forma que a classe criada tenha os parâmetros de QoS adequados. Daremos mais detalhes no quinto capítulo.

Para a política WFQ, o ALTQ ainda não possui uma interface com o RSVP, de forma que o usuário deverá atribuir pesos aos fluxos de acordo com a QoS solicitada pela aplicação. As sessões seguintes contêm mais detalhes sobre as funções implementadas no ALTQ.

4.1.2 Escalonamento CBQ

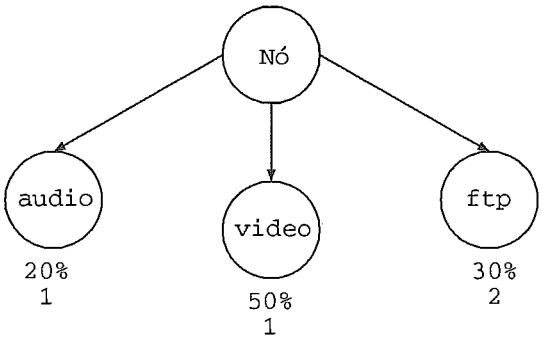


Figura 4.1: Compartilhamento de um nó entre classes de serviços.

Do termo inglês *Class-Based Queueing*, o escalonamento CBQ, também conhecido como escalonamento hierárquico, está baseado na criação de classes ou hierarquias para os diversos tipos de fluxo que um determinado elemento roteador deverá lidar. Para cada uma dessas classes, é alocada uma parte da banda disponível e também é atribuída uma prioridade. Podemos definir as classes baseadas nos tipos de fluxo, por exemplo classes de tempo real, para aplicações de voz ou vídeo interativo e classes de dados para aplicações tradicionais como *ftp* e *telnet*.

A seguir daremos dois exemplos, visando o melhor entendimento do funcionamento do CBQ.

No primeiro exemplo, representado esquematicamente pela figura 4.1, temos três classes, cada uma com um percentual de banda alocado e com uma certa prioridade. Os percentuais de banda e prioridade nesse caso são ilustrativos.

No caso de todas as classes estarem recebendo tráfego, cada uma terá seu percentual de banda de acordo com a alocação solicitada inicialmente. O tráfego que exceder o percentual fixado será descartado.

Se a classe *ftp* (prioridade = 2) parar de receber pacotes, haverá uma parte da capacidade do canal (30%) que ficará ociosa. Esta capacidade será dividida igualmente entre as classes áudio e vídeo.

Caso a classe de áudio ou vídeo (prioridade = 1) parem de receber tráfego, toda a capacidade ociosa será passada para a outra classe (vídeo ou áudio) de mesma prioridade.

Podemos expandir a estrutura anterior, criando dois níveis de hierarquia, onde o primeiro nível distribuiria a banda entre duas organizações. Vejamos o segundo exemplo usando a árvore da Figura 4.2

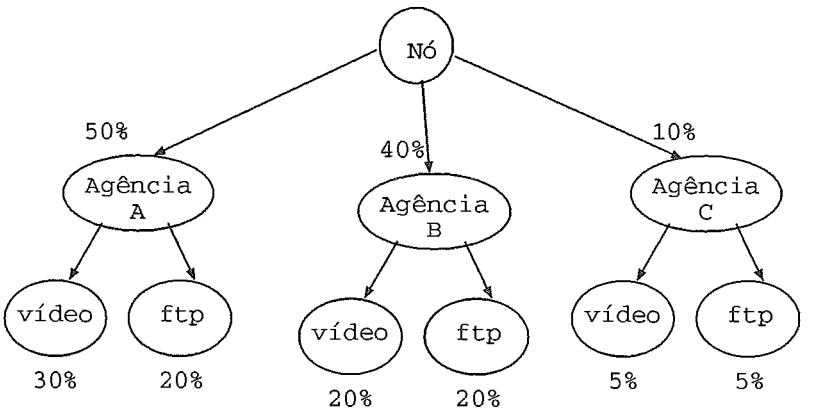


Figura 4.2: Compartilhamento de um nó entre organizações e classes de serviço, criando uma estrutura hierárquica.

Além da banda e da prioridade que devem ser atribuídas a uma classe, um terceiro parâmetro pode ser selecionado. Este parâmetro indica a permissão de uma classe usar a banda disponível de outras classes superiores na hierarquia ("pedir emprestado"). No exemplo da figura 4.2, suponha que as classes *ftp* de todas as agências não possam pedir emprestado e as classes vídeo possam. Logo se a classe vídeo de A parar de receber tráfego, sua banda será distribuída proporcionalme às classes vídeo B e C.

Podemos visualizar na figura 4.3, uma classe *default*. Esta classe serve para escoar todo o tráfego que não esteja em conformidade com as classes 1 e 2.

A incorporação de mecanismos de compartilhamento de enlace, leva a simplificação e adiciona robustez aos serviços de tempo real na Internet [22]. A reunião de classes de mesma prioridade e a proteção daquelas cujo tráfego tenha menor prioridade, proporciona flexibilidade à Internet, promovendo bons resultados em termos de controle de congestionamento. Vemos a seguir, na Figura 4.3, a arquitetura do CBQ.

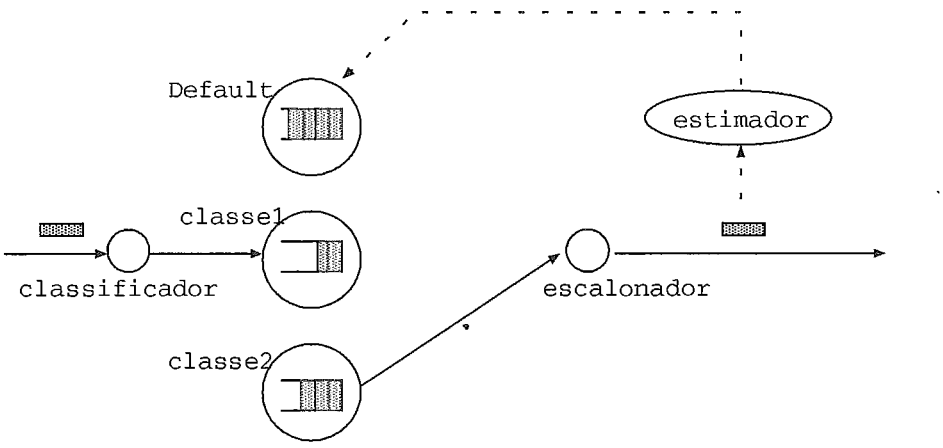


Figura 4.3: Componentes do CBQ.

Cada classe "folha" possuirá sua própria fila, a qual será atribuída sua porção de banda e uma prioridade. Uma classe poderá usufruir de recursos de uma classe "irmã", ou pedir emprestado (de seu "pai") caso estes tenham banda disponível.

O CBQ trabalha da seguinte forma: o classificador atribui a classe apropriada ao pacote. O estimador computa a banda recentemente utilizada por essa classe. Se a classe tem excedido seu limite pré-definido, o estimador marca a classe como sobrecarregada. O escalonador determina então o próximo pacote que deverá ser servido examinando as diversas filas, baseado em prioridades e estado das classes. Um esquema de pesos é utilizado entre classes de mesma prioridade. Vejamos a seguir um estudo mais detalhado dos módulos escalonador e estimador do CBQ.

4.1.3 Módulos estimador e escalonador

Com o objetivo de melhor entendimento, as classes podem encontrar-se nos seguintes estados:

1. **sobrecarregadas** - aquelas classes que recentemente utilizaram mais banda do que a alocada;

2. **sub-utilizadas** - aquelas que tem utilizado menos banda do que a alocada e

3. **no-limite** - caso contrário.

Cabe ao estimador definir o estado de uma classe. A partir do conhecimento do estado da classe e de informações sobre o estado das classes "pai" e "irmãs", um tipo de controle será definido e empregado à classe. Uma classe pode permanecer sem nenhum controle se não está sobrecarregada, ou no caso onde esteja sobrecarregada e possua uma classe irmã ou pai com banda disponível. Não ocorrendo estas duas situações a classe será então controlada.

O escalonador por sua vez implementa uma política que depende do tipo de controle que deve ser exercido sobre a classe.

4.1.3.1 Estimador

Este módulo possui dois parâmetros fundamentais: uma constante w e a frequência com a qual atualiza os estados de cada classe.

O estado de uma classe é calculado após a transmissão de cada pacote de dados da classe.

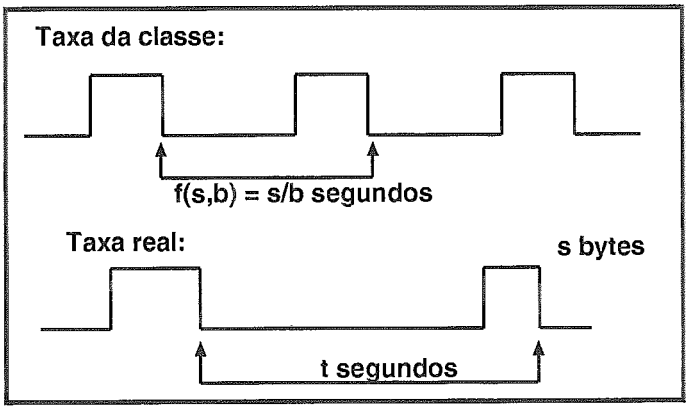


Figura 4.4: Variáveis usadas para computar o estado de cada classe.

Seja s o tamanho do recém transmitido pacote em *bytes*, b a banda alocada para a classe em *bytes*/s e t em segundos, o tempo medido entre dois pacotes consecutivos da mesma classe, conforme mostrado na figura 4.4. Se o roteador enviar pacotes de tamanho s para uma classe cuja banda alocada seja b, então o tempo entre pacotes sucessivos será $f(s,b) = s/b$ s.

Seja $diff$ a diferença de tempo entre o esperado e o real:

$$diff = t - f(s, b).$$

Nota-se que $diff$ é negativo se a classe exceder sua banda alocada e não-negativo caso contrário. A média móvel da variável $diff$ é computada conforme a seguinte equação:

$$avg \leftarrow (1 - w)avg + w * diff$$

O constante w determina o peso que será dado aos valores calculados anteriormente e ao valor atual.

O valor da variável avg, computado pelo estimador, após a transmissão de cada pacote, indica o estado da classe. A partir do valor de avg e considerando o estado de classes irmãs e pai, o estimador irá definir o valor da variável tempo para o envio do próximo pacote ($T\_envio\_prox\_pacote$). É esta variável que será usada pelo escalonador para decidir se deve transmitir ou não o próximo pacote.

A variável $T\_envio\_prox\_pacote$ é calculada da seguinte forma:

- Se $avg > 0$, a classe está sub-utilizada ou no-limite. Então $T\_envio\_prox\_pacote = 0$ (não será exercido nenhum controle).
- Se $avg < 0$, a classe está sobrecarregada. Se não existe nenhuma classe irmã ou pai com banda disponível, então $T\_envio\_prox\_pacote = f(s, b)$. A classe só poderá usar o máximo permitido para ela.
- Se $avg < 0$ e existe uma classe irmã ou pai com banda disponível, então $T\_envio\_prox\_pacote = avg\frac{1-w}{w} + f(s, b)$. Nesse caso é exercido um controle parcial sobre a classe.

Duas questões adicionais são colocadas: 1) a banda alocada de um nó deverá ser a soma das alocações dos filhos (conforme representado na figura 4.2) e 2) o escalonador realiza serviço conservativo, ou seja, ele nunca "descansará" enquanto houver uma classe com pacotes.

4.1.3.2 Escalonador

O escalonador sempre envia os pacotes das classes de maior prioridade primeiro. Dentro de classes de mesma prioridade, o escalonador atribui pesos proporcionais às bandas alocadas para cada classe, usando uma política semelhante à *Weighted Round Robin*[22]. O peso determina o número de *bytes* que a classe poderá enviar a cada rodada. Quando a classe envia mais do que o número de *bytes* permitido, a alocação de *bytes* para a próxima rodada é reduzida correspondentemente.

Antes do escalonador transmitir um pacote de uma determinada classe, ele compara o valor da variável *T_envio_prox_pacote* com o tempo corrente. Três casos podem ocorrer:

1. *T_envio_prox_pacote* é igual a zero (classe sem controle). O pacote é transmitido.
2. *T_envio_prox_pacote* é menor que o tempo corrente (classe com controle). O pacote é transmitido.
3. *T_envio_prox_pacote* é maior que o tempo corrente (classe com controle). O pacote não é transmitido.

Com essa política, o escalonador exercerá controle sobre a classe até que a variável *T_envio_prox_pacote* volte ao valor zero.

4.1.3.3 Exemplo

O objetivo deste exemplo é mostrar o comportamento do CBQ no caso em que as classes de maior prioridade recebem um tráfego que tem como característica períodos de silêncio e períodos de rajada.

Sejam 4 classes **C1, C2, C3** e **A**, com prioridades 1,2,3 e 4 respectivamente, cada qual recebendo 25% da banda total. Suponha que a classe **A** esteja enviando pacotes e as três outras classes estejam em silêncio. No exato tempo t as três outras classes iniciam o envio de uma rajada. Nota-se que não será exercido nenhum controle sobre as classes C1, C2 e C3 (por elas estarem sub-utilizadas em t) até que elas tenham usado o percentual da banda alocado para elas. A partir deste instante (por exemplo, t + t/4 para C1) a classe ficará sobrecarregada e então será controlada.

Podemos ver na figura 4.5, a banda alocada a cada classe ao longo do tempo. Por ter prioridade maior do que todas, **C1** inicia sua transmissão até que tenha atingido sua banda alocada no instante **T/4** (25T/100) segundos. Nesse momento, o escalonador continua a transmissão de **C1**, ocupando no máximo 25% da banda disponível e o restante tornar-se-á disponível para a classe **C2** que transmitirá durante **T/3** (25T/75) segundos, ocupando 75% da banda. Ao atingir a quota de **C2**, o escalonador inicia a transmissão de **C3** da mesma forma, mantendo a porção de banda para **C1** e **C2** até o momento de a classe **A** voltar a transmitir, após a espera de um tempo total de **T+T/12** segundos.

Em [22] é dado um limite superior para o tempo que a classe A espera para transmitir.

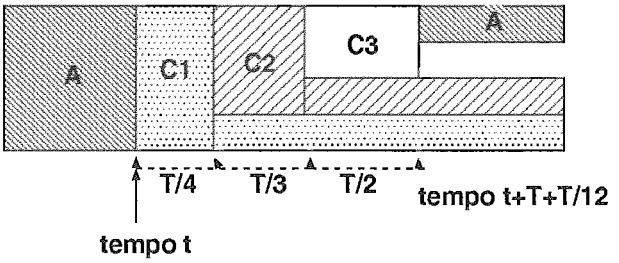


Figura 4.5: Espera que a classe A deverá sofrer.

4.1.4 Escalonador WFQ

O pacote ALTQ implementa também o escalonamento baseado na política **WFQ** (*Weighted Fair Queueing*). O princípio de funcionamento do WFQ é baseado na criação de filas de serviço para cada conexão, servindo-as em *round-robin*. As filas vazias são puladas. Na proposta inicial, chamada de **FQ** (*Fair Queueing*) dada a existência de n filas, o escalonador as serviria de forma cíclica enviando um pacote de cada fila em cada ciclo de serviço.

A proposta revisada do FQ foi realizar o envio dos pacotes bit a bit. Isso, no entanto, acarreta que a banda seja igualmente distribuída entre as filas. Esse efeito foi limitado, dividindo a banda em ciclos de tamanho m-*bits*, sendo $(m > n)$ e alocando *bits* extras para aquelas filas que necessitem, segundo os pesos atribuídos para cada fila. Essa extensão ao FQ é chamada de WFQ.

Com isso, aloca-se banda por fluxo e não mais por máquina, criando uma fila para cada fluxo, tornando o WFQ um esquema para dividir banda entre os diversos fluxos

que passam através de um roteador. Através da prova de Parekh[33],verificou-se que o WFQ dá extremas garantias de alocação de banda aos fluxos. O peso dado resulta na distribuição da capacidade da rede em proporções diferentes para cada fila de saída. Assim sendo, pode-se proteger um fluxo de bruscas variações ocorridas em outros fluxos, dando garantias de porções do recurso compartilhado (nesse caso específico, garantias em relação à banda previamente alocada).

A implementação contida no ALTQ[25] é a do WFQ, onde uma função *hash* é utilizada para mapear um fluxo em um conjunto de filas, possibilitando dois fluxos serem mapeados dentro da mesma fila. Nenhuma garantia individual aos fluxos poderá ser proporcionada por essa implementação do WFQ, dado que mais de um fluxo pode compartilhar a mesma fila.

4.1.5 Classificador de pacotes

A tarefa do classificador de pacotes, é examinar os campos do cabeçalho de cada pacote, determinando qual a fila o pacote deve ser encaminhado. Esse problema é mais complexo, à medida que aumentam os critérios para classificar os pacotes, criando a necessidade de utilização de diversos campos do cabeçalho IP para a identificação final. Por exemplo, tomando o esquema observado na Figura 4.2, os pacotes poderão ser encaminhados para uma de muitas agências, requerendo exames de endereços de fonte e destino; poderão ser classificados em relação à aplicação, como ftp, telnet, através da observação dos números de portas. Para fluxos de vídeo, pode-se incrementar a classificação com critérios como o nível de descarte de pacotes, contido em campos especiais.

Na implementação do ALTQ, os pacotes são classificados segundo os campos do cabeçalho IP. Caso um pacote não faça parte de nenhum critério pré-estabelecido, este será encaminhado à classe *default.*

4.1.6 Controle de Admissão

Embora não seja responsabilidade específica dos módulos de controle de tráfego, o controle de admissão é um módulo acessório que verifica a condição local de congestionamento, validando ou não a entrada de novas conexões. Ele portanto, ao contrário do escalonamento que atua na saída da rede, age como controlador na entrada.

Uma implementação restrita foi executada pelo pacote ALTQ, simplesmente comparando a banda requisitada por novas conexões com a banda já alocada, a fim de impedir o acesso de fluxos que necessitem de mais banda do que aquela disponível.

4.2 Interface RSVPxALTQ

O objetivo da implementação do pacote ALTQ serviu basicamente a três propósitos[25]:

- proporcionar uma plataforma de desenvolvimento de métodos de escalonamento da saída;
- proporcionar um *kernel* com controle de tráfego para o RSVP e
- fazer possível o gerenciamento das filas de saída.

Todas as máquinas onde o pacote ALTQ está implantado (transmissor e roteadores), na política de escalonamento da saída CBQ, ao receberem uma mensagem RSVP RESV (indicando que uma reserva deva ser feita para um determinado fluxo) vão criar uma nova classe folha com o parâmetro de QoS especificado. Todos os pacotes do fluxo serão encaminhados para essa classe. O único parâmetro de QoS analisado pelo ALTQ é a taxa média (r) do TOKEN_BUCKET_TSPEC do serviço de carga-controlada. Ele serve para definir a taxa média que a aplicação espera gerar. Os outros parâmetros passados não são relevantes para a atual implementação.

4.3 Aplicativos desenvolvidos

A fim de determinar o comportamento de alguns parâmetros de desempenho de aplicações submetidas ao ambiente controlado de comunicações, foram construídos dois programas capazes de gerar tráfego com requisitos de tempo real. O primeiro programa, simula uma fonte de voz típica, através de um modelo markoviano ON-OFF[30]. O tráfego gerado, multiponto por construção, pode então ser captado por diversos pontos receptores. Caberá ao receptor medir o jitter percebido, indicando sua distribuição, média, variância e a perda de pacotes. O segundo programa, transmite uma seqüência real de vídeo codificada em MPEG, também em uma sessão multiponto. O receptor de vídeo observa e registra a distribuição do jitter, sua média

e variância ao longo da recepção e a perda de pacotes. Descreveremos a seguir, a estrutura do transmissor e a base para o cálculo dos parâmetros de desempenho observados na chegada para ambas as fontes implementadas.

4.3.1 Fonte de voz

O modelo implementado para a fonte de voz, pode ser representado por uma cadeia de Markov simples, conforme a figura 4.6 abaixo:

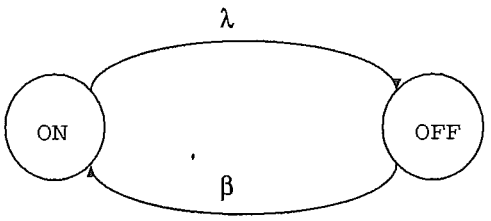


Figura 4.6: Modelo para a fonte ON-OFF.

Se a fonte está no estado ON, são gerados pacotes de tamanho fixo de 64 *bytes* a intervalos de 20 ms. Se está no estado OFF, ela permanecerá em silêncio não enviando pacotes. A fonte passa do estado ON para o estado OFF com uma taxa λ, onde $\frac{1}{\lambda}$ é igual a 352 ms, indicando a média de permanência no estado ativo. Ela fica no estado OFF um tempo médio igual a $\frac{1}{\beta}$, ou seja, 650 ms. Os valores acima são empíricos e estão recomendados em [30].

Após a transmissão de todos os períodos ativos de interesse, o receptor terá armazenado uma matriz D contendo todas as medidas de jitter, em todos os períodos observados. Seja $D(i,j)$ um elemento da matriz D, onde $1 \leq i \leq m$ e $1 \leq j \leq n$, m representando o número de períodos ativos e n representando o número de valores de jitter coletados em um período. A média do jitter em um período i, $1 \leq i \leq m$, será dada por:

$$M(i) = \sum_{j=1}^{n} \frac{D(i,j)}{n}$$

A média do jitter para os m períodos é obtida através de:

$$M = \sum_{i=1}^{m} \frac{M(i)}{m}$$

Para a variância de um período, temos a expressão:

$$V(i) = \sum_{j=1}^{n} \frac{(D(i,j) - M(i))^2}{n-1}$$

e a variância média é dada por:

$$V = \sum_{i=1}^{m} \frac{V(i)}{m}$$

A fração de pacotes perdidos P é calculada tendo os valores do número total de pacotes enviados ($pac_{enviados}$) e o número total de pacotes recebidos ($pac_{recebidos}$), fazendo-se a seguinte razão:

$$P = \frac{pac_{enviados} - pac_{recebidos}}{pac_{enviados}}$$

4.3.2 Fonte de vídeo

Para a fonte de vídeo, utilizou-se seqüências reais de filmes, encontradas na Internet. As seqüências, codificadas em MPEG, são analisadas por um programa chamado MPEG_STAT, disponível para a platarforma UNIX, que extrai uma série de informações do arquivo MPEG, dentre elas todos os tamanhos dos quadros MPEG contidos na seqüência. Os tamanhos dos quadros são colocados em um arquivo com a seguinte formatação:

0 I 33424

1 B 2172

2 P 4524

3 I 33632

4 B 4492

5 P 2548

etc.

onde estão representados os números de seqüência na primeira coluna, o tipo de quadro MPEG (se B, P ou I) na segunda e o tamanho em *bits* do quadro na terceira.

Usando esse arquivo como entrada, foi criado um programa, que envia, através de comunicação multiponto, pacotes cujos tamanhos correspondem ao indicado no arquivo, seguindo a seqüência. Em termos práticos, o efeito é igual à emissão do filme pela rede.

Como não há um tamanho fixo para cada quadro, foram escolhidas as chamadas de função do sistema operacional **sendmsg/recvmsg** para realizar a comunicação. Essas funções trocam informações através de uma estrutura única, de tamanho fixo: a **msghdr**, conforme abaixo (retirado do manual *on-line* do UNIX):

```
ssize_t sendmsg(int s, const struct msghdr *msg, int flags)

ssize_t recvmsg(int s, struct msghdr *msg, int flags)

    struct msghdr {
            caddr_t msg_name;       /* optional address */
            u_int   msg_namelen;    /* size of address */
            struct  iovec *msg_iov; /* scatter/gather array */
            u_int   msg_iovlen;     /* # elements in msg_iov */
            caddr_t msg_control;    /* ancillary data */
            u_int   msg_controllen; /* anc. data buffer len */
            int     msg_flags;      /* flags on received msg */
    };
```

Através da estrutura msghdr, é passado o endereço destino e seu tamanho, um ponteiro para a área de dados a serem enviados (contido na estrutura *iovec*), o número de fragmentos em que a área de dados será dividida, dentre outros.

O receptor de vídeo recebe tais informações, sendo responsável por monitorar a ausência de algum número da seqüência, registrar os tempos de chegada de cada pacote e calcular do valor do jitter conforme a definição:

$$jitter = t_2 - t_1 - T_g,$$

onde t_1 e t_2 indicam o tempo de chegada do pacote anterior e o tempo do pacote atual respectivamente e T_g, indica o intervalo de tempo de geração de pacotes no transmissor, possuindo um valor de acordo com a taxa de transmissão da seqüência, geralmente de 24 ou 30 quadros por segundo. Cada valor de jitter é armazenado em

uma posição de um vetor, doravante chamado $Dif(n)$, onde n é o número de valores observados. A média do jitter M é calculada conforme abaixo:

$$M = \sum_{i=1}^{n} \frac{Dif(i)}{n}$$

A variância do jitter V pode ser obtida através de:

$$V = \sum_{i=1}^{n} \frac{(Dif(i) - M)^2}{n-1}$$

O cálculo da perda P é realizado exatamente como no item anterior.

Capítulo 5

Resultados dos experimentos

Para a realização de medidas que pudessem comprovar a importância da utilização de reserva de recursos em um ambiente de rede, duas aplicações de tempo real multiponto foram utilizadas, conforme descrito no capítulo anterior. A idéia central foi observar variações no comportamento dos parâmetros de desempenho jitter e perda de pacotes das aplicações em dois cenários distintos: um onde houvesse o gerenciador de recursos RSVP e o módulo de controle de tráfego ALTQ, e outro cenário onde foram replicadas as mesmas condições porém sem a presença de nenhum controle de QoS específico. A implantação do ambiente de testes foi a etapa de maior complexidade, destacando-se dois pontos: a implantação dos programas gerenciadores (e pré-requisitos) e a utilização de máquinas em rede externa. Dois ambientes foram organizados: um controlado, onde montou-se uma pequena rede local e o outro remoto, entre dois laboratórios da própria Universidade. No presente capítulo estão descritos os cenários dos testes realizados e os diversos valores dos parâmetros obtidos.

5.1 Ambiente controlado

O objetivo dos testes realizados neste ambiente foi a observação da QoS obtida pelas aplicações, dado o controle total sobre o tráfego gerado e sobre os elementos que fizeram parte da comunicação (transmissor, receptor e roteadores). Além do

tráfego gerado pelas aplicações de vídeo e voz, utilizamos fontes de tráfego adicionais, descritas no item 5.1.2, de forma a criar situações de congestionamento no roteador e podermos observar como o módulo de controle de tráfego atuaria nesses casos.

5.1.1 Descrição do ambiente

A seguinte configuração de máquinas em uma rede local foi organizada, conforme a figura 5.1 abaixo:

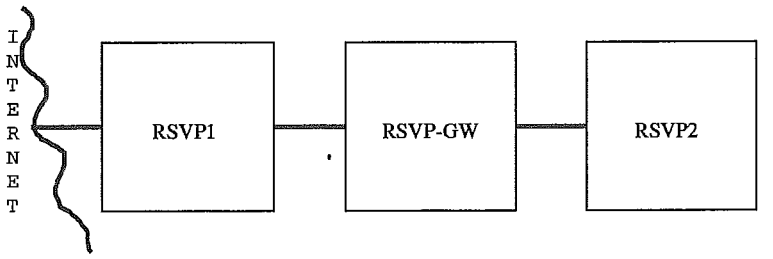


Figura 5.1: Representação esquemática do ambiente controlado.

Configurações de hardware:

- RSVP-GW- 486DX2- 32M*Bytes* de RAM
- RSVP1 - 486SLC - 8M*Bytes* de RAM
- RSVP2 - 486DX2 - 24M*Bytes* de RAM

Em todas as três máquinas utilizadas, foi instalado o sistema operacional FreeBSD, v3.2.

Configuração de rede:

Do ponto de vista de configuração da rede, a máquina RSVP1 serviu para conectar a rede controlada à rede Internet, permitindo o acesso externo à rede de testes, conforme indicado na figura 5.1. Como roteador da rede de testes, na máquina RSVP-GW instalou-se o software MROUTED (versão 3.9 beta2) com a opção de compilação RSRR, permitindo corretamente interpretar os pacotes de controle do

gerenciador de recursos. Além disso, o *mrouted* é usado para rotear os pacotes necessários entre as duas interfaces de rede existentes, e, através dele, ser possível a instalação de túneis virtuais. Dessa forma, é possível conectar a rede interna a outras redes de forma transparente. A máquina RSVP2 por sua vez, ficou conectada à máquina RSVP-GW, formando com esta uma rede local.

Arquivos de configuração:

Por ser o gateway entre RSVP1 e RSVP2, a máquina RSVP-GW necessitou da instalação e configuração do *mrouted*, cujo arquivo de configuração é colocado resumidamente a seguir:

```
# mrouted.conf,v 3.8 1995/11/29 22:40:47 fenner Rel
#
#   This is the configuration file for "mrouted", an IP multicast router.
#   mrouted looks for it in "/etc/mrouted.conf".
. . .
phyint ed0
phyint ed1
#   tunel entre a maquina a maquina RSVP-GW e araruama.land (provisorio)
146.164.240.65 146.164.47.193 metric 1 threshold 1 rate_limit 10000
. . .
```

Como podemos observar, o *mrouted* através do comando ***phyint*** habilita as duas interfaces de rede ed0 e ed1 presentes na máquina RSVP-GW e depois disponibiliza um túnel entre ele e outra máquina com o comando ***tunnel***. A criação do túnel requer, basicamente, que ambos os lados envolvidos apontem um para o outro. A implantação do *mrouted* tem como único pré-requisito a necessidade da opção MROUTING compilada no *kernel* da máquina. Mais detalhes sobre o programa, bem como sobre o arquivo de configuração podem ser obtidos no manual *on-line* do UNIX.

Configuração do RSVP e ALTQ:

A fim de observar o gerenciamento de recursos atuando, foi instalado o pacote RSVP (*release* 4.2a4). Como módulo acessório, o pacote ALTQ (versão 1.1.3) também foi instalado, a fim de permitir a aplicação de políticas de escalonamento na saída de pacotes. O ALTQ, cria um *daemon* do rsvp (*rsvpd*) modificado que executará além das funções de gerenciamento, as funções de escalonamento implementadas. A

política de escalonamento do ALTQ que conta com interface para uso com o RSVP é o escalonamento *Class-based Queueing* (CBQ).

A configuração do CBQ dá-se através de um arquivo no sistema. Vejamos o arquivo **cbq.conf** em uma das configurações realizadas no roteador RSVP-GW:

```
rsvp-gw:/etc > cat cbq.conf
interface ed1 bandwidth 10M cbq
 class cbq ed1 root_class NULL priority 0 admission none pbandwidth 100
 class cbq ed1 def_class root_class borrow priority 1 \
                    pbandwidth 80 default
   filter ed1 def_class 224.9.9.9 0 0 0 17
 class cbq ed1 tra_class root_class borrow priority 0 admission cntlload \
                    pbandwidth 20

interface ed0 bandwidth 10M cbq
 class cbq ed0 root_class NULL priority 0 admission none pbandwidth 100
 class cbq ed0 def_class root_class priority 1 pbandwidth 80 default
    filter ed0 def_class 224.9.9.9 0 0 0 17
 class cbq ed0 tra_class root_class priority 0 admission cntlload \
                    pbandwidth 20
```

Podemos observar nesse arquivo, a existência de 3 comandos básicos de configuração:

comando interface - habilita o escalonamento CBQ na interface de rede, informando qual a banda total disponível.

comando class - define a estrutura hierárquica do escalonamento, informando os nós "pais" e "filhos", bem como suas prioridades e porção da banda a que tem direito. A classe *default* é necessária para informar ao escalonador que ela receberá todos os pacotes que não possuírem características especiais. No exemplo acima, ambas as interfaces da máquina RSVP-GW estão configuradas com escalonamento CBQ; as classes sempre informam seu nome e o nome da classe "pai", da qual herdam todas as características, com exceção daquelas especificadas no próprio comando. No caso da classe *default*, a única variável herdada é a com relação à não existência de controle de admissão, uma vez classe *default* atribui valores próprios para as variáveis de prioridade e banda.

comando filter - define as características do tráfego permitindo o enquadramento deste em alguma classe. Terá sempre uma referência sobre a classe ao qual pertença, bem como campos para endereço destino, endereço fonte, portas de comunicação, identificador de protocolos e outros identificadores de campos do cabeçalho dos pacotes.

Mais detalhes podem ser obtidos na documentação do pacote ALTQ em [25, 28].

Inicialização do RSVP:

Tão logo os pré-requisitos anteriormente descritos estejam instalados, o estabelecimento de uma reserva pelo RSVP é realizado através da interface interativa que o programa disponibiliza ao usuário. Abaixo, temos a tela inicial que é mostrada ao usuário.

```
rsvp-gw:rsvpd# ./rsvpd -D
19:17:43.465  RSVP Version 4.2a4, Compiled on: Sep 14 1999 at 15:09
19:17:43.474  Log level:7  Debug mask:7  Start time: Wed Sep 15 19:17:43 1999
19:17:43.584  CBQ config files is /etc/cbq.conf
19:17:43.605  CBQ enabled on interface ed0 (mtu:1500)
19:17:43.608  CBQ enabled on interface ed1 (mtu:1500)
19:17:43.612  Physical, Virtual, and API interfaces are:
19:17:43.615   0 ed0                                    146.164.240.2/26  TCup M
19:17:43.619   1 ed1                                   146.164.240.65/26  TCup M
19:17:43.623   2 API                                         0.0.0.0/0   NoIS
19:17:43.626   3 ed0                                 146.164.240.2 vif 0/24  TCup M
19:17:43.630   4 ed1                                146.164.240.65 vif 1/24  TCup M
19:17:43.634   5 ed0                                 146.164.240.2 vif 2/24  TCup M
rtap [v3.1] RSVP Test Application
RSVP API version 5.02

Enter ? or command:
T1>
```

Resumidamente a seguir serão mostrados os principais comandos para configurar corretamente dois pontos que desejam trocar informações. Vale lembrar que todos os roteadores intermediários devem executar o módulo RSVP para corretamente interpretar as ações enviadas pelos participantes.

```
          fonte:    146.164.240.65 porta 9001
          destino:  224.9.9.10     porta 9000
          taxa 4000B/s (~32Kbps)
fonte  ------------   roteador   -------   receptor
(146.164.240.65)
-------------------------------------------------------------
# dest udp 224.9.9.10/9000              # dest udp 224.9.9.10/9000
# sender 9001 [t 4000 1000 5000 100]
    (porta-fonte  r     b   p    m)
**** início do envio da msg de path ****
                                        # receive
                                        (join no grupo multicast)
                                        **** chegada da msg de path ***
                                        # reserve ff 146.164.240.65/9001 \
                                             [cl 4000 1000 100 5000]
                                        (usando estilo fixed-filter)
                                        (ou) reserve wf \
                                             [cl 4000 1000 100 5000]
                                        (usando estilo wildcard-filter)
                                  **** início do envio da msg de resv ****
**** chegada da msg de resv ****
     (Início da reserva)

                                        # close
 # close
```

5.1.2 Fontes geradoras de tráfego extra

Nesta seção estão descritas as fontes geradoras de tráfego adicional, utilizadas durante todos os experimentos. O uso dessas fontes teve como objetivo provocar situações de congestionamento e com isso averiguar se os mecanismos de controle de tráfego realmente conseguem fornecer QoS aos pacotes de dados de um determinado fluxo.

1) Fonte geradora de tráfego determinístico espalhado

A fonte de tráfego determinístico espalhado, cujo *throughput* pode ser observado na figura 5.2, produz um tráfego constante. Como parâmetros de entrada, a fonte recebe o tamanho da mensagem em *bytes*, o tamanho de cada pacote em *bits*, o tempo total no qual a mensagem deverá ser enviada em segundos e o número de mensagens a serem enviadas.

De posse desses dados, o programa calcula o tempo necessário entre o envio de pacotes. Após o envio de cada pacote é verificado se o tempo para o envio do pacote seguinte esgotou-se. Caso positivo, ela está apta a enviar o próximo pacote; caso contrário, a fonte aguarda até o tempo de envio do próximo pacote. O procedimento é repetido até que toda a mensagem tenha sido enviada e não mais existam mensagens para enviar.

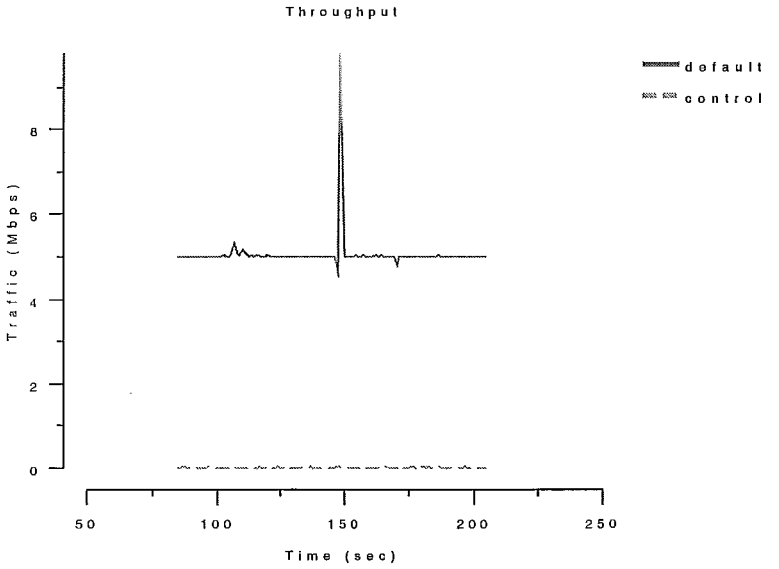


Figura 5.2: Tráfego espalhado gerado pela fonte espalhado.

Dessa forma, a fonte de tráfego consegue enviar a mensagem durante todo o intervalo

de tempo requisitado, "espalhando" o envio ao longo da transmissão. A figura 5.2 mostra o *throughput* de saída de uma máquina, por onde a fonte de tráfego está sendo transmitida. Podemos observar como a fonte geradora produz uma saída constante ao longo do tempo.

2) Fonte de tráfego determinístico em rajadas

Para essa fonte, a implementação realizada foi a seguinte: como dados de entrada, a fonte recebe o tamanho da mensagem em *bytes*, o tamanho do pacote em *bits*, o tempo total no qual toda a mensagem deverá ser enviada e o número de mensagens.

Com esses dados, a fonte calcula quantos pacotes deverá enviar. Nesse momento o programa entra num *loop*, enviando os pacotes tão rápido quanto possível, até que não reste mais nenhum pacote para ser enviado. Em seguida, é realizada uma comparação para determinar se o tempo total para envio da mensagem foi ou não alcançado. Caso ainda haja tempo sobrando, a fonte descansa até alcançar o tempo requisitado, senão, um novo ciclo é iniciado.

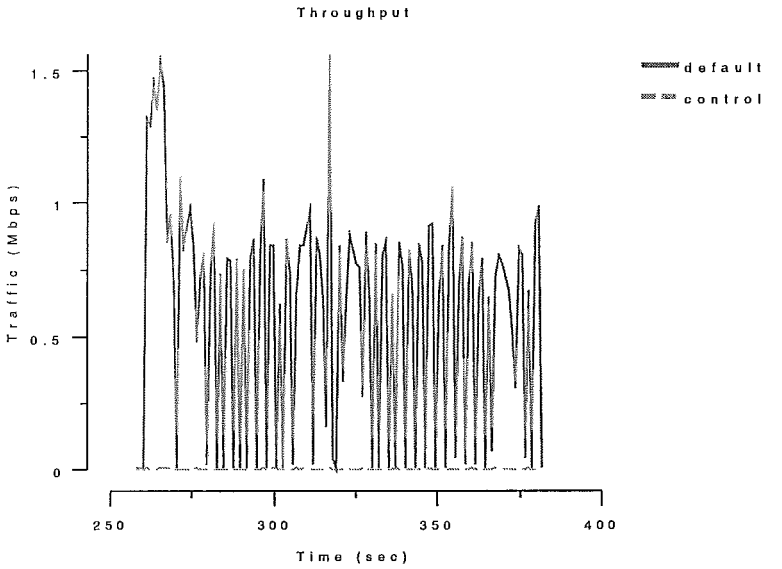


Figura 5.3: Tráfego em rajadas.

O tráfego em rajada, pode ser visualizado na figura 5.3. Cabe ressaltar que, pela própria construção da fonte, a rajada será mais acentuada quanto maior for o processamento da máquina que a estiver gerando.

5.1.3 Experimento com CBQ

Como citado anteriormente, o pacote ALTQ implementa duas políticas de escalonamento: CBQ e WFQ. A política CBQ sendo mais complexa, no sentido de possuir muitos parâmetros de configuração a serem considerados para cada classe, (ex.: porção da banda, prioridade, possibilidade de pegar emprestado banda disponível de outras classes, etc.), foi estudada a parte.

Com o objetivo de determinar o comportamento exato da política CBQ, dadas as características das fontes de tráfego e a configuração das classes, foi realizado o experimento descrito a seguir.

Este consistiu do envio de um tráfego de vídeo, no ambiente controlado, entre a máquina RSVP1 e a RSVP2 passando pela RSVP-GW. Neste teste a aplicação transmissora estava localizada em RSVP1 e aplicação receptora em RSVP2. A máquina RSVP-GW atuou como roteadora. Conforme a organização do arquivo **/etc/cbq.conf** em RSVP-GW, a configuração de classes (inclusive banda e prioridade) foi feita como mostrado na figura 5.4:

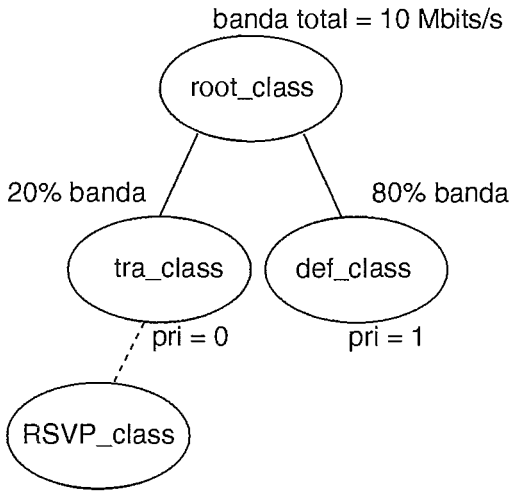


Figura 5.4: Organização de classes do CBQ.

A classe *def_class*, sendo a classe *default* e com uma configuração de 80% da banda total da classe pai *root_class*, encaminha todos os fluxos que saíam por esta interface de RSVP-GW; recebeu uma prioridade de valor 1. A classe de trabalho *tra_class* com a configuração de 20% da banda total e prioridade 0 (maior) permitiu a presença de classes dinâmicas. A classe *RSPV_class* é criada dinamicamente, tão logo a

reserva seja estabelecida. Esta classe "herda" todas as características da classe pai, no caso a *tra_class*.

Para a reserva, foi requisitado pelo RSVP uma QoS de 1,5 M*bits*/s de banda para a fonte de vídeo, cujo *throughput* médio fica em torno de 1,5M*bits*/s. Este parâmetro será usado para limitar a banda da classe *RSVP_class* (que no caso poderia ter requisitado até 2M*bits*/s). Vejamos um resumo do que o analisador dinâmico do CBQ mostrou para as classes durante a realização do experimento:

```
Default Class for Interface ed1:
        priority: 1 depth: 0 offtime: 1045 [us] wrr_allot: 1500 bytes
        nsPerByte: 1025 (7.80 Mbps),    Measured: 3.00 [Mbps]
        pkts: 1529389,  bytes: 1198316108
        overs: 0,       overactions: 0
        borrows: 0,     delays: 0
        drops: 0,       drop_bytes:0
        QCount: 0,      (qmax: 30)
        AvgIdle: 212 [us],      (maxidle: 220 minidle: -49 [us])

Class 1 on Interface ed1:
        priority: 0 depth: 0 offtime: 21061 [us] wrr_allot: 500 bytes
        nsPerByte: 5333 (1.50 Mbps),    Measured: 1.48 [Mbps]
        pkts: 6787,     bytes: 8091896
        overs: 1524,    overactions: 0
        borrows: 912,   delays: 0
        drops: 0,       drop_bytes: 0
        QCount: 0,      (qmax: 30)
        AvgIdle: 285 [us],      (maxidle: 17394 minidle: -250 [us])
```

Podemos observar pelas informações contidas no campo "*Measured*" , o *throughput* medido para as fontes de tráfego, contidas na classe Default (*def_class*). A fonte de vídeo, representada pela Classe 1 (*RSVP_class* dinâmica) não perdeu nenhum pacote até o instante de observação, como podemos constatar através do parâmetro "*drops*". Assim sendo, verificamos que a configuração, contida no arquivo do CBQ, atenderia a essa fonte de vídeo. Maiores informações podem ser obtidas em [25, 28]

5.2 Experimentos realizados no ambiente controlado

Realizamos diversos experimentos com fontes de voz e vídeo em um ambiente com reserva de recursos (RSVP e ALTQ) e um ambiente sem reserva de recursos. Foram utilizadas fontes de tráfego extras para forçar situações de congestionamento. Os seguintes cenários foram considerados:

1. Fonte de voz com tráfego extra espalhado e com tráfego extra em rajadas;
2. Filme de ação de curta duração diante de tráfego extra espalhado e com tráfego extra com rajadas;
3. Filme de uma palestra diante de tráfego extra espalhado e com tréfego extra em rajadas;
4. Filme de ação de longa duração em presença de tráfego extra espalhado.

A seguir descreveremos os resultados dos experimentos realizados.

5.2.1 Fonte de voz

Para esse experimento, foram realizados 5 ensaios com a fonte de voz, cada um com duração aproximada de cinco minutos. Esta fonte possui um *throughput* médio informado em torno de 20K*bits*/s. O expectro da fonte de voz, como podemos observar na figura 5.5, é composto de uma série de rajadas, cada uma representando um período ativo. O tempo de permanência no período ativo e de silêncio utilizados para a fonte implementada foram respectivamente 650ms e 352ms, diferentemente do valor empírico recomendado em [30].

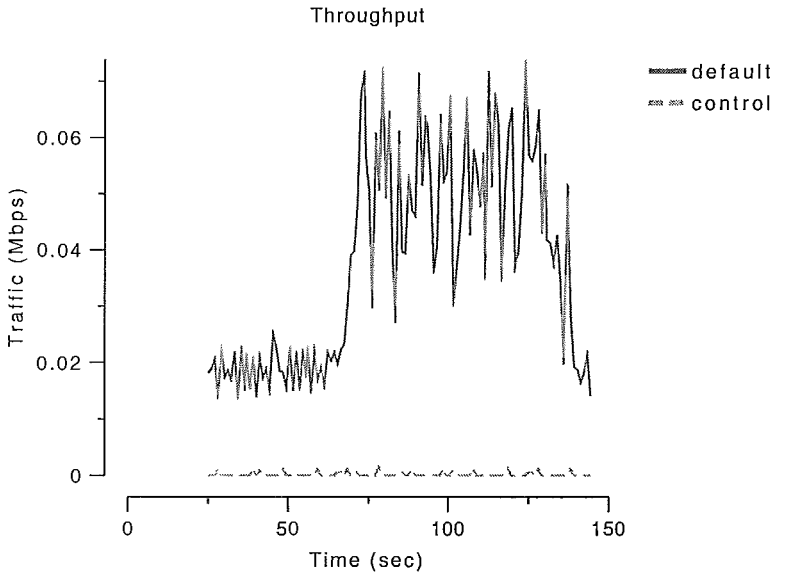
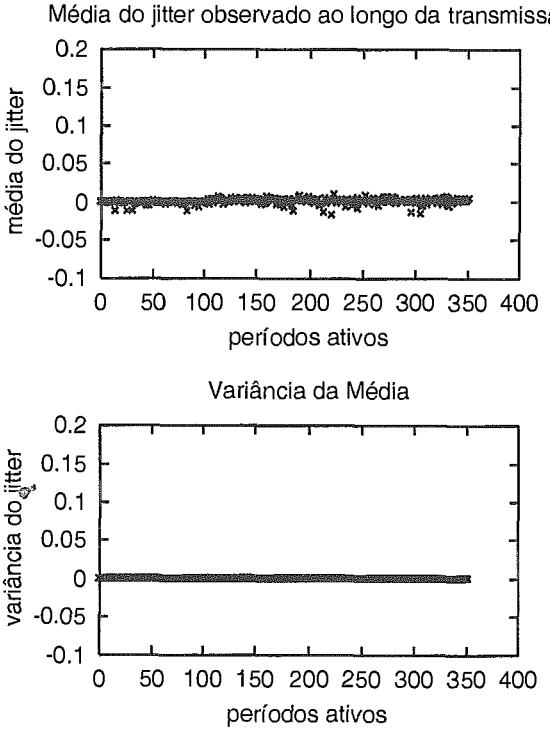


Figura 5.5: Representação do *throughput* da fonte de voz.

O tráfego de voz, fluindo entre RSVP1 (origem) e RSVP2 (destino), sofreu grande congestionamento ao passar por RSVP-GW, que fez o papel de roteador. O congestionamento foi produzido com dois tipos de tráfego distintos, determinístico espalhado e com rajadas. O tráfego extra foi gerado no roteador RSVP-GW de forma a simular uma situação de congestionamento em um roteador da Internet. Utilizou-se 3 fontes determinísticas de tráfego espalhado cada uma com taxa média de 2M*bits*/s e também três fontes de tráfego com rajadas com a mesma configuração. A descrição do tráfego gerado nas fontes está na seção **5.1.2**.

O ambiente com reserva de recursos foi configurado da seguinte forma: o módulo de controle de tráfego ALTQ foi executado no roteador RSPV-GW. A política de escalonamento usada foi o CBQ com a mesma configuração de classes utilizada na seção 5.1.3. A banda pedida ao RSVP foi de 100K*bits*/s em todos os experimentos. Vejamos os resultados:

1) Fonte de tráfego espalhado, sem reserva de recursos:



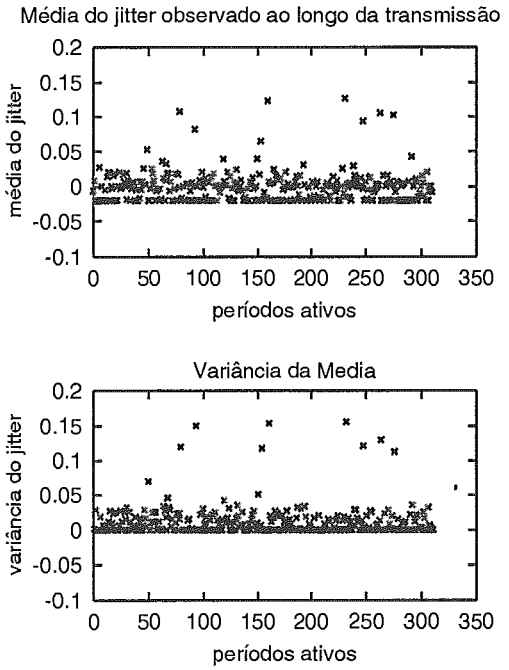
Média e variância do jitter observadas ao longo da recepção da fonte de áudio, durante aproximadamente cinco minutos. Nenhum controle das comunicações foi empregado.

O valor médio do jitter e sua variância nos 351 períodos ativos foram:

Média = 5.288e-4
Variância = 7.472e-5

Foram enviados 9485 pacotes e recebidos 10514 pacotes, representando uma perda de 9.79%

2) Fonte de tráfego espalhado, com reserva de recursos:



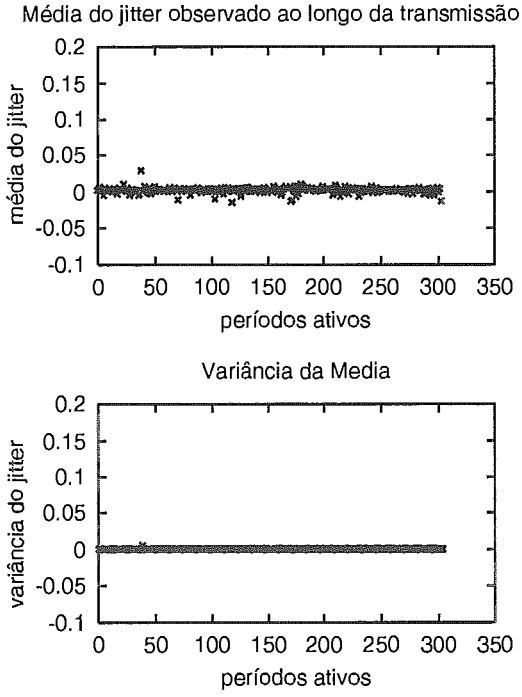
Média e variância do jitter observadas ao longo da recepção da fonte de áudio, durante aproximadamente cinco minutos. Foi solicitada uma banda de100kbps.

O valor médio do jitter e sua variância nos 310 períodos ativos foram:

Média = -1.009e-3
Variância = 1.253e-2

Foram enviados 10080 pacotes e recebidos 9212 pacotes, representando uma perda de 8.68%

3) Fonte de tráfego com rajadas, sem reserva de recursos:



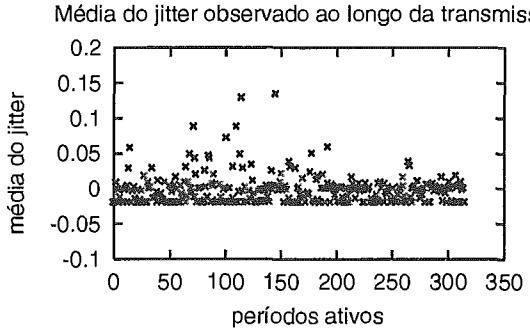
Média e variância do jitter observadas ao longo da recepção da fonte de áudio, durante aproximadamente cinco minutos. Nenhum controle das comunicações foi empregado.

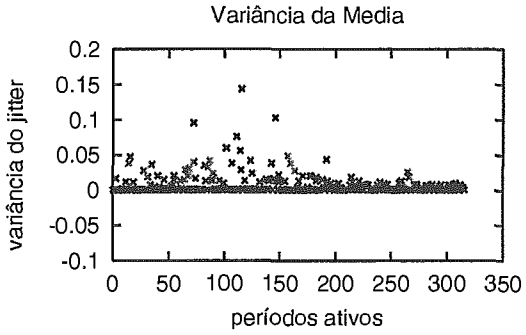
O valor médio do jitter e sua variância nos 303 períodos ativos foram:

Média = 1.605e-3
Variância = 9.011e-5

Foram enviados 8517 pacotes e recebidos 9811 pacotes, representando uma perda de 13.18%

4) Fonte de tráfego com rajadas, com reserva de recursos:





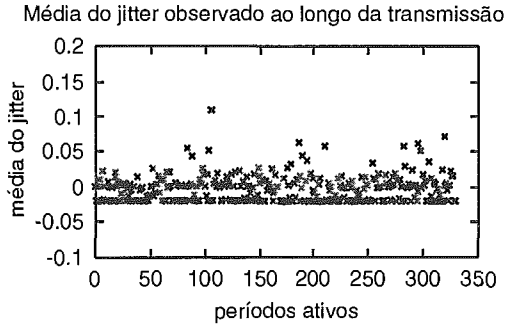
Média e variância do jitter observadas na trasmissão da fonte de áudio ON-OFF durante cinco minutos. Foi solicitada 100kbps de banda.

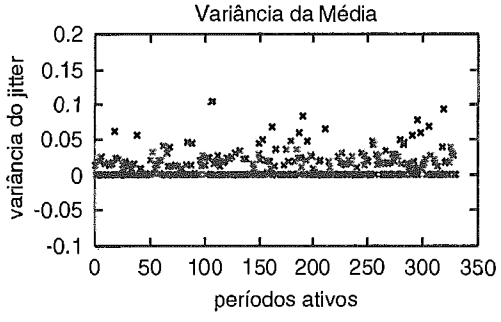
O valor médio do jitter e sua variância nos 344 períodos ativos foram:

Média = -6.564e-4
Variância = 7.974e-3

Foram enviados 8873 pacotes e recebidos 8509 representando uma perda de 4,01%

5) Fonte de tráfego espalhado, dobrando o tamanho do *buffer*:





Média e variância do jitter observadas ao longo da transmissão da fonte de áudio ON-OFF durante aproximadamente cinco minutos. Foi solicitada uma reserva de banda de 100kbps e foi dobrado o tamanho do buffer.

O valor médio do jitter e sua variância nos 334 períodos ativos foram:

Média = -5.801e-4
Variância = 1.573e-2

Foram enviados 9952 e recebidos 9822 pacotes, representando uma perda de 1,3%.

Quadro comparativo dos resultados com fonte de voz

Tabela 5.1: Comparação entre os diversos ensaios com a fonte de voz.

	tráfego extra		CBQ	Sem reserva
	espalhado	per. ativos	310	351
	espalhado	méd. do jitter(ms)	-1.009e-3	5.288e-4
	espalhado	var. do jitter(ms)	1.253e-2	7.472e-5
	espalhado	perda(%)	8.68	9.79
	em rajadas	**per. ativos**	**344**	**303**
	em rajadas	**méd. do jitter(ms)**	**-6.564e-4**	**1.605e-3**
	em rajadas	**var. do jitter(ms)**	**7.974e-3**	**9.011e-5**
	em rajadas	**perda%**	**4.01**	**13.18**
buff. dobrado	espalhado	per. ativos	329	x
buff. dobrado	espalhado	méd. do jitter(ms)	-5.801e-4	x
buff. dobrado	espalhado	var. do jitter(ms)	1.573e-2	x
buff. dobrado	espalhado	perda%	1.3	x

Conforme podemos observar pela tabela 5.1, o fato de colocar uma reserva que cria fila e banda diferenciadas para o aplicativo de voz não melhora significativamente a QoS (no exemplo com o tráfego espalhado). Optamos então por aumentar o tamanho do *buffer* da aplicação de voz. O tamanho de *buffer* de transmissão para a aplicação é o segundo parâmetro passado no pedido de QoS enviado pelo RSVP (através do parâmetro TOKEN_BUCKET_TSPEC), porém a interface do programa CBQ utilizado não interpreta essa informação, conforme explanado na seção 4.2.

Para alterar tal parâmetro, foi necessária recompilação do programa *rsvpd*, alterando especificamente tal variável, dobrando seu valor no caso de 30 para 60 possíveis posições de entrada. Como efeito de tal alteração, o quinto ensaio foi produzido e foi verificado níveis muito baixos para o parâmetro perda. Utilizando a estatística do CBQ, foram observadas raras perdas.

O fato do aumento do tamanho do *buffer* melhorar a taxa de perda do voz deve-se ao algoritmo implementado pelo escalonador/estimador. Neste algoritmo o tamanho do *buffer* de cada classe é calculado dinamicamente como uma função do tamanho máximo de *buffer* especificado e dos últimos valores do tamanho da fila da classe. Portanto, após um período de silêncio da fonte (início do período ativo), o valor do tamanho do *buffer* será pequeno, ocasionando perdas no próximo período ativo.

5.2.2 Fonte de vídeo

Na realização dos experimentos de vídeo, optou-se por criar um túnel lógico entre a máquina RSVP-GW e ARARUAMA, em uma rede local próxima. Tal fato deveu-se pelo pequeno desempenho mostrado pela máquina RSVP1 (de menor capacidade) na transmissão do vídeo. Cabe ressaltar que os experimentos foram realizados em horários onde os roteadores que fazem parte do túnel lógico estavam sub-utilizados, visando não comprometer a validade dos resultados. A configuração está representada na figura 5.6 abaixo:

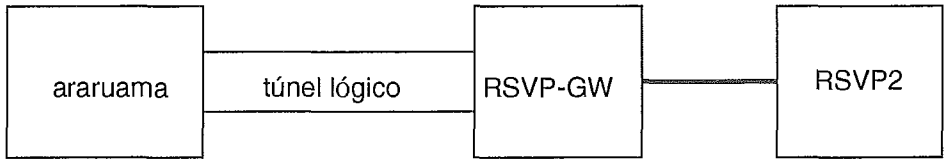


Figura 5.6: Experimento da fonte de vídeo.

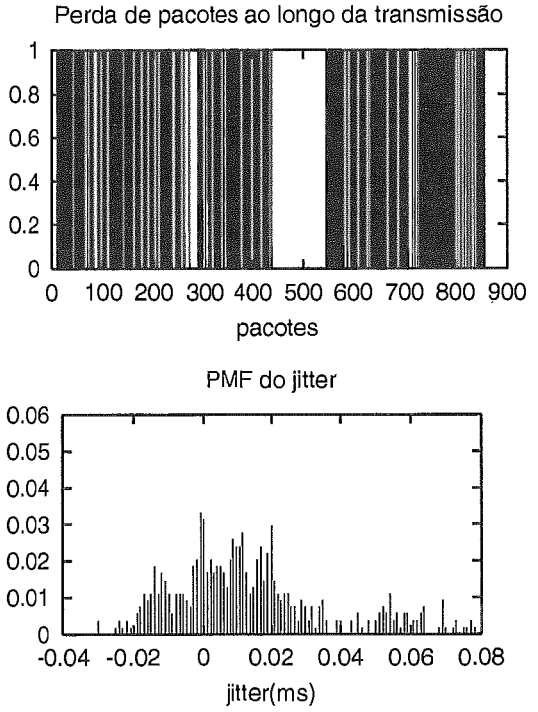
Para esse experimento foram usadas três seqüências distintas de vídeo codificadas em padrão MPEG. Procuramos utilizar seqüências com características de tráfego diferentes, ou seja, duas seqüências que possuem rajadas (filmes de ação) e a outra, por ser uma palestra, apresenta variações menores em torno da taxa média. Todos os arquivos utilizados para a geração das seqüências estão disponíveis em [29].

Cada seqüência requer condições específicas de banda, de acordo com a taxa de transmissão. A primeira seqüência, chamada **Discovery**, contém um filme de ação com duração de 30s. Sua taxa média de saída ficou em torno de 1.5M*bits*/s. A segunda seqüência transmitida, chamada de **Tangram2**, contém uma palestra. Sua taxa média de saída gira em torno de 450K*bits*/s e possui a duração de 4min30s. A terceira seqüência transmitida, contém todo o filme **Starwars**, codificado em preto e branco. Durante a transmissão do filme, a taxa de envio variou entre valores de 300K*bits*/s até 1M*bits*/s. Sua duração média é de 85 minutos.

O tráfego extra foi gerado no roteador RSVP-GW. Foram utilizadas três fontes de tráfego extra em cada experimento, gerando uma taxa média total de 3.0M*bits*/s para as fontes de tráfego espalhado e 4.3M*bits*/s para as fontes de tráfego em rajadas. Foi requisitado através do RSVP, 1.5M*bits*/s para seqüência **Discovery** e **Tangram2**, correspondente à taxa média e de pico para as respectivas seqüências. Para a seqüência **Starwars**, foi produzido somente um resultado utilizando a fonte de tráfego espalhado com taxa média de 2.4M*bits*/s. A reserva foi feita pelo RSVP

requisitando 1M*bits*/s, sendo esta a taxa de pico produzida pelo filme. Não houve alteração na configuração do escalonador CBQ em relação ao item anterior. Vejamos o comportamento do jitter, sua média e variância ao longo da transmissão das seqüências, apresentado os comentários ao final da seção.

1) Seqüência Discovery: sem reserva e fonte de tráfego espalhado:

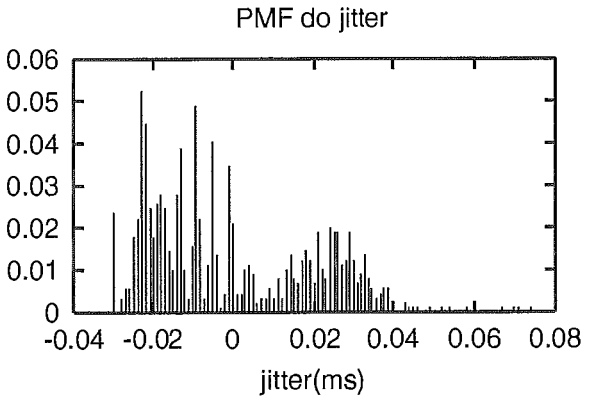


Foi utilizada a fonte de vídeo com a sequência Discovery e nenhum controle foi empregado sobre as comunicações.

Média do jitter: 2,027e-2 ms
Variância: 1,224e-3

Pacotes recebidos: 540 em 900 enviados, resultando em perda de 40%.

2) Sequência Discovery: com reserva e fonte de tráfego espalhado:

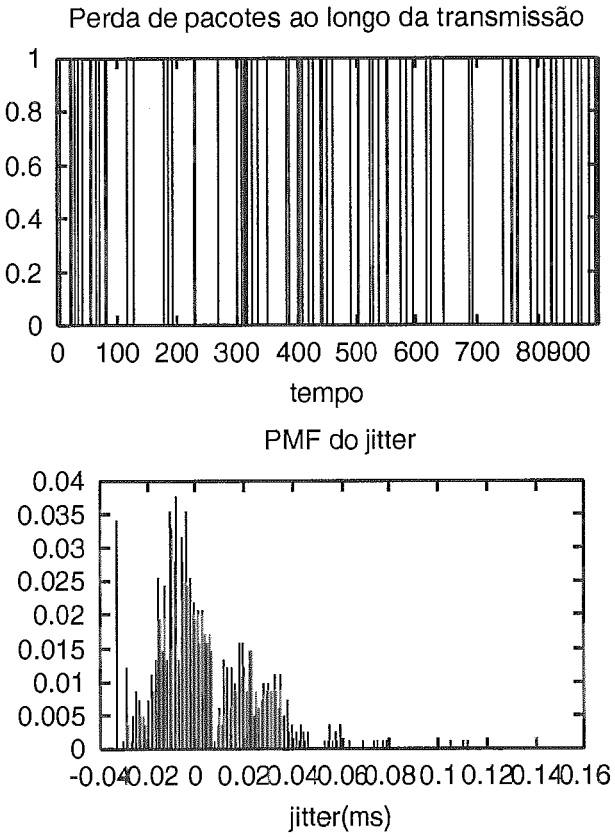


Foi utilizada a sequência Discovery, tendo sido requisitada uma reserva de 1500kps de banda

Média do jitter: 2,486e-4 ms
Variância: 4,499e-4

Pacotes recebidos: 897 em 900 enviados, resultando em perda de 0,33%.

3) Sequência Discovery, sem reserva e fonte de tráfego em rajadas:

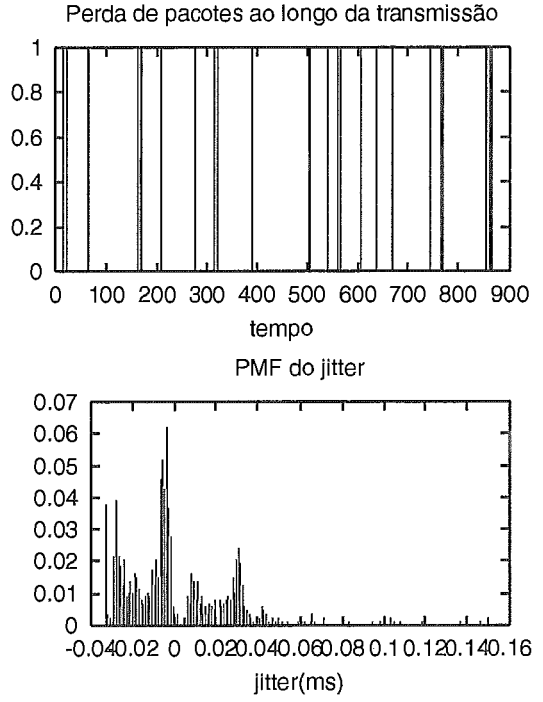


Foi utilizada a fonte de vídeo com a sequência Discovery sem nenhum controle de QoS requisitado.

Média do jitter: 3,244e-3 ms
Variância: 4,440e-4

Pacotes recebidos: 819 em 900 enviados, resultando em perda de 9%.

4) **Sequência Discovery, com reserva e fonte de tráfego em rajadas:**



Foi utilizada a fonte a sequência Discovery, sendo requisitada uma reserva com 1500kps de banda para a sequência.

Média do jitter: 1,880e-3 ms
Variância: 6,779e-4

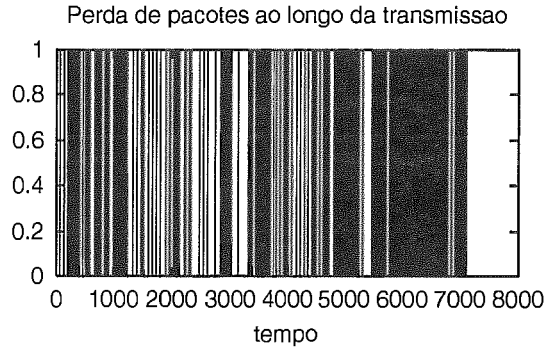
Pacotes recebidos: 869 em 900 enviados, resultando em perda de 3,4%.

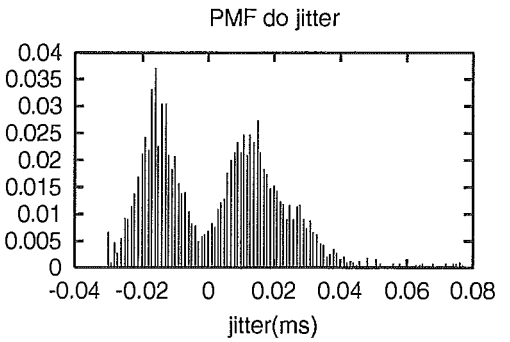
Quadro comparativo seqüência Discovery

Tabela 5.2: Quadro comparativo das experiências com a seqüência Discovery.

		CBQ	Sem reserva
	Méd. jitter(ms)	**2.486e-4**	**2.027e-2**
espalhado	**Var. jitter**	**4.499e-4**	**1.2242-3**
	Perda(%)	**0.33**	**40**
	Méd. jitter(ms)	1.880e-3	3.244e-3
em rajadas	Var. jitter	6.779e-4	4.440e-4
	Perda(%)	3.4	9

5) Seqüência Tangram2, sem reserva e fonte de tráfego espalhado:



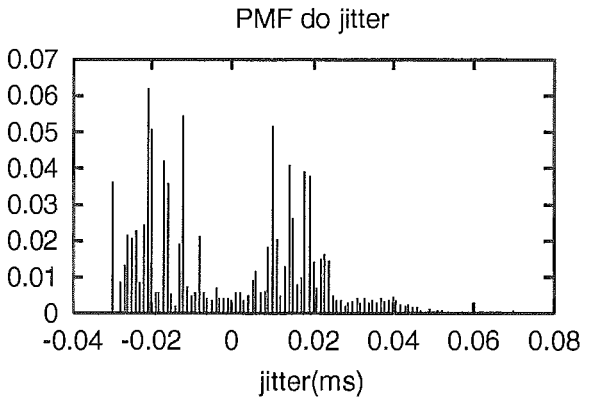


Foi utilizada a sequência Tangram2 sem nenhum controle sobre as comunicações.

Média do jitter: 4,063e-3 ms
Variância: 4,406e-4

Pacotes recebidos: 790 em 7115, resultando em perda de 11,1%.

6) Seqüência Tangram2, com reserva e fonte de tráfego espalhado:

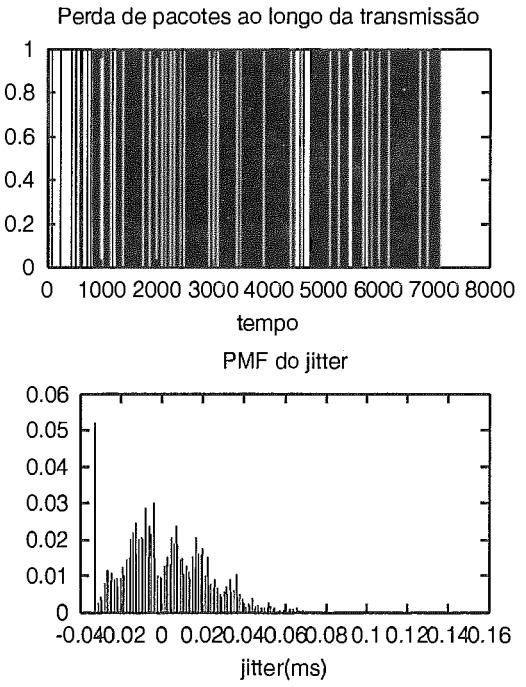


Foi utilizada a fonte a sequência Tangram2 requisitando de 1500Mpbs de banda à rede.

Média do jitter: 4,530e-4 ms
Variância: 4,845e-4

Pacotes perdidos: 4 em 7115, resultando em perda de 0,056%.

7) Seqüência Tangram2, sem reserva e fonte de tráfego em rajadas:

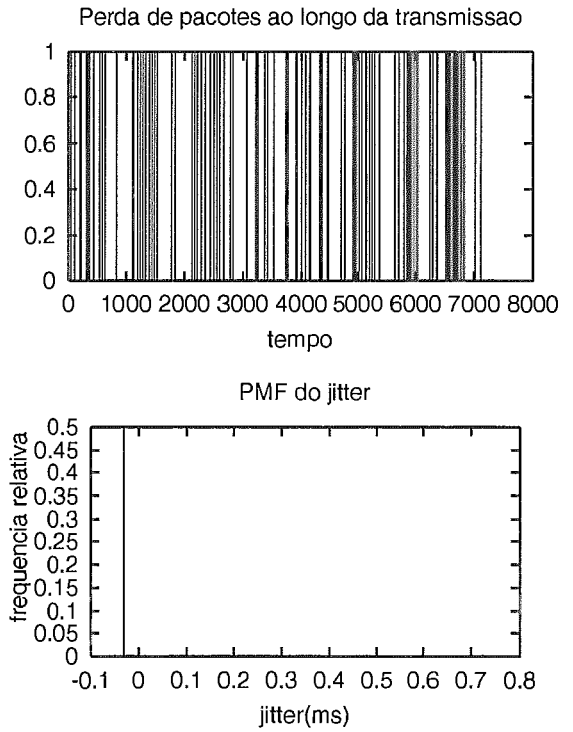


Foi utilizada a sequência Tangram2, Foi requisitada uma reserva de 1500kps para o aplicativo.

Média do jitter: 2.756e-3 ms
Variância: 4.884e-4

Pacotes recebidos: 6573 em 7115 enviados, resultando em perda de 7.6%.

8) Seqüência Tangram2, com reserva e fonte de tráfego em rajadas:



Foi utilizada a sequência Tangram2 e reservada uma banda de1500kps para o aplicativo.

Média do jitter: 8.962e-4 ms
Variância: 4.371e-3

Pacotes recebidos: 6932 em 7115 enviados, resultando em perda de 2.57%.

Tabela 5.3: Quadro comparativo da seqüência Tangram2 - ambiente controlado.

		CBQ	Sem Reserva
	Méd. jitter(ms)	**4.530e-3**	**4.063e-3**
espalhado	**Var. jitter**	**4.845e-4**	**4.406e-4**
	Perda(%)	**0.056**	**11.1**
	Méd. jitter(ms)	8.962e-4	2.756e-3
em rajadas	Var. jitter	4.371e-3	4.884e-4
	Perda(%)	2.57	7.6

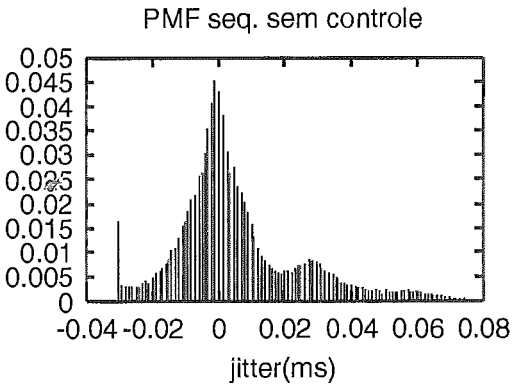
5) Seqüência Starwars

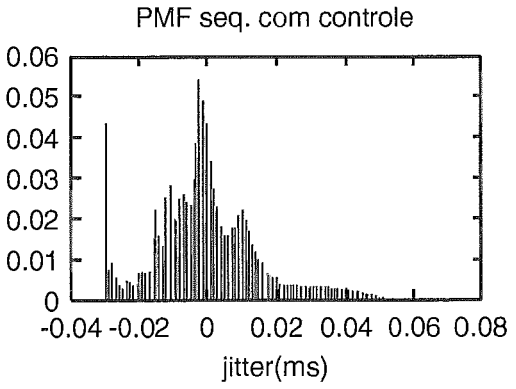
A seqüência Starwars utilizada conta com 174049 frames. Nesse cenário, foram instaladas três fontes de tráfego extra determinístico com taxa média total de 2.4M*bits*/s. Quando em um ambiente com reserva de recursos, foi alocada uma banda de pico de 1M*bits*/s para a seqüência. Os resultados dos experimentos estão apresentados a seguir.

Tabela 5.4: Resumo do experimento com a seqüência Starwars.

	sem reserva	com reserva
média jitter(ms)	1.645e-2	3.995e-5
var. jitter	2.937e-0	2.564e-4
perda de pacotes (%)	34.87	0

PMF do jitter para transmissão do filme Starwars:





Constatamos em todos os testes realizados com as seqüências de vídeo que a qualidade de serviço em termos de perda de pacotes e média e variância do jitter teve uma melhora significativa quando o ambiente com reserva de recursos foi utilizado.

A média e a variância do jitter diminuíram de uma até três ordens de grandeza. Observamos que o tráfego extra acarretou em uma percentagem de perda menor do que quando o tráfego espalhado foi utilizado para o ambiente sem reserva. Por exemplo tivemos 40% de perda (tráfego extra espalhado) e 9% de perda (tráfego extra em rajadas) no ambiente sem reserva, no ensaio realizado para a fonte Tangram2.

Pudemos constatar através da observação da fila de saída do roteador que cada vez que a fonte extra emitia uma forte rajada, os *buffers* ficavam cheios. Contudo, durante os períodos de silêncio, o *buffers* ficavam praticamente dedicados à seqüência de vídeo ocasionando menor perda na saída do transmissor.

5.3 Ambiente remoto

A arquitetura do ambiente remoto está apresentada na figura 5.7. Foi configurado um túnel lógico entre duas máquinas FreeBSD localizadas em dois laboratórios do campus da UFRJ: uma no NCE (ARARUAMA) e outra no CT - Bloco I (LAM-ROUTER). Os experimentos realizados consideraram a transmissão das seqüências entre os dois laboratórios.

Figura 5.7: Ligação física entre os roteadores do NCE e CT-Bloco I.

Conforme a figura 5.7, observamos que o tráfego trocado entre os dois laboratórios atravessa outros roteadores, além de compartilhar o anel FDDI da Universidade. Cabe ressaltar que o controle dos recursos foi realizado somente entre as duas pontas do túnel lógico criado, pois não houve possibilidade de acesso aos outros roteadores. Porém, os testes foram realizados em horários nos quais os roteadores estavam sub-utilizados. Vejamos as características da configuração montada.

5.3.1 Descrição do ambiente

Configuração de rede:

As máquinas utilizadas eram os respectivos roteadores de suas sub-redes, tendo sido unidas via *mrouted*. Elas contavam com o seguinte *hardware*:

- ARARUAMA - Pentium 100 com 64M*bytes* de RAM.
- LAMROUTER - Pentium 166 com 32M*bytes* de RAM.

Configuração do RSVP/ALTQ:

Os pacotes RSVP/ALTQ foram instalados. Somente o roteador LAMROUTER possui o módulo de controle de tráfego, logo, em todos os experimentos, a transmissão iniciou-se no Bloco I, chegando até ARARUAMA no NCE, onde foi observada a QoS do tráfego recebido.

O CBQ respeitou a mesma configuração instalada no ambiente controlado. Além do CBQ, a política WFQ disponibilizada pelo pacote ALTQ foi usada. Sua configuração é bastante simples. Dado que para o tráfego dos aplicativos foi utilizado o grupo 224.9.9.10 e que para o tráfego das outras fontes geradoras o grupo 224.9.9.9, os seguintes comandos foram usados, habilitando a configuração dos pesos para ambos os tráfegos e atribuindo o pesos de 30/60 respectivamente para o tráfego dos aplicativos e 60 para o tráfego das demais fontes.

```
lam1:wfq-tools# ./wfqd
> help
Usage : interface enable [nqueues]
 interface disable
 interface getqid ipaddr(XX.XX.XX.XX)
 interface getstats qid
 interface setweight qid weight
>
> ed1 enable
> ed1 getqid 224.9.9.9
queue ID = 233 for dest (224.9.9.9)
> ed1 getqid 224.9.9.10
queue ID = 234 for dest (224.9.9.10)
> ed1 setweight 233 30
> ed1 setweight 234 60
> ed1 getstats 233
Bytes = 0-
Weight = 30
Sent : packets = 0, bytes = 0
Drop : packets = 0, bytes = 0
> quit
```

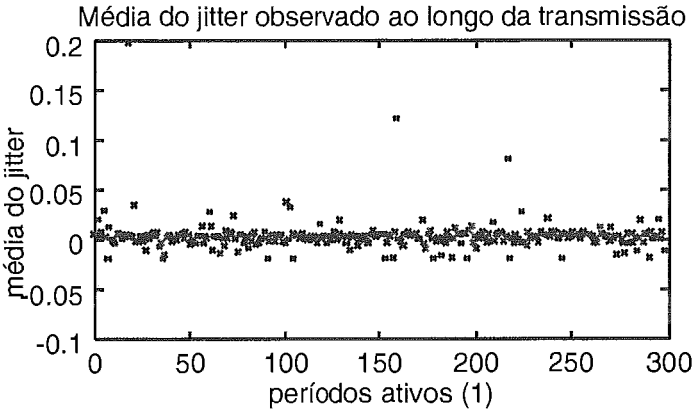
5.4 Experimentos realizados

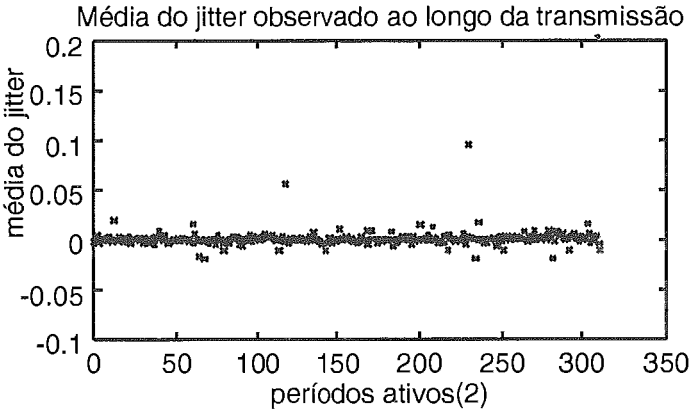
Realizamos os experimentos para os mesmos cenários descritos no ambiente controlado. No ambiente remoto realizamos um experimento a mais (para todos os cenários) onde consideramos também a política WFQ do pacote ALTQ. A seguir apresentaremos uma série completa de ensaios realizados para as fontes de voz e vídeo implementadas.

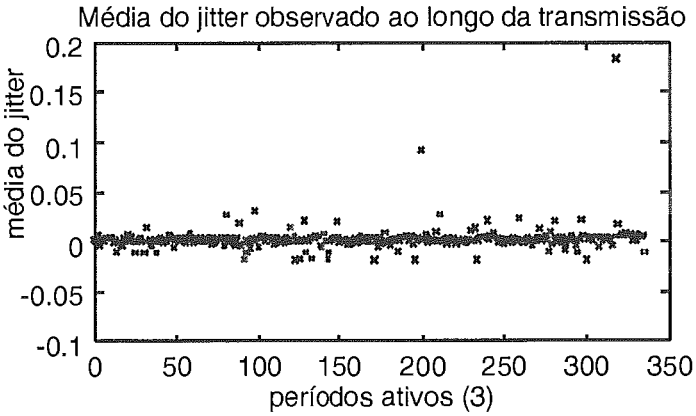
5.4.1 Fonte de voz

Nestes experimentos utilizamos seis fontes extras gerando tráfego em LAMROUTER. A carga de tráfego extra para o experimento foi em média de 4800K*bits*/s. O CBQ foi configurado através do RSVP. Foi solicitado 100K*bits*/s de banda alocada média para o CBQ e o WFQ foi configurado conforme a seção 5.3.1, utilizando pesos 1/89, respectivamente para o tráfego da fonte de voz e fontes de tráfego extra.

1) Três ensaios de voz para o tráfego extra espalhado:







Média do jitter observada na transmissã da fonte de áudio ON-OFF durante cinco minutos.

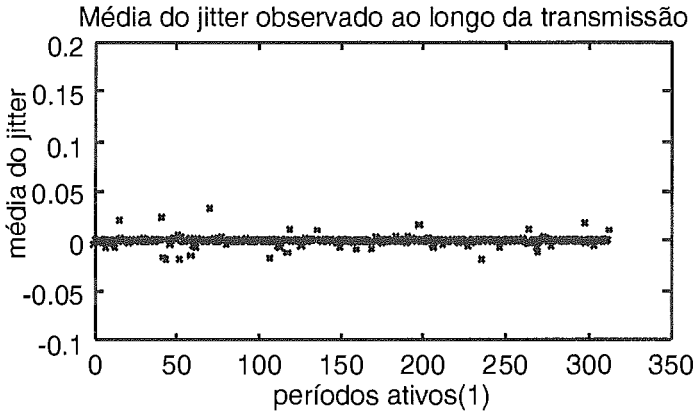
O primeiro gráfico (1) apresenta medidas quando foi usado o escalonamento WFQ.

No (2), foi utilizado o escalonamento CBQ.

O terceiro(3) mostra a média do jitter sem controle das comunicações.

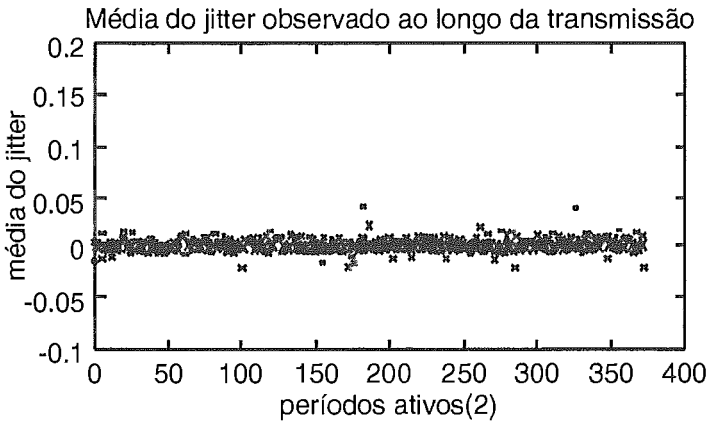
Mais detalhes estão na tabela ao final da seção.

2) Três ensaios com fonte de voz com tráfego extra em rajadas:



Média do jitter observada na transmi
da fonte de áudio ON-OFF durante
cinco minutos.

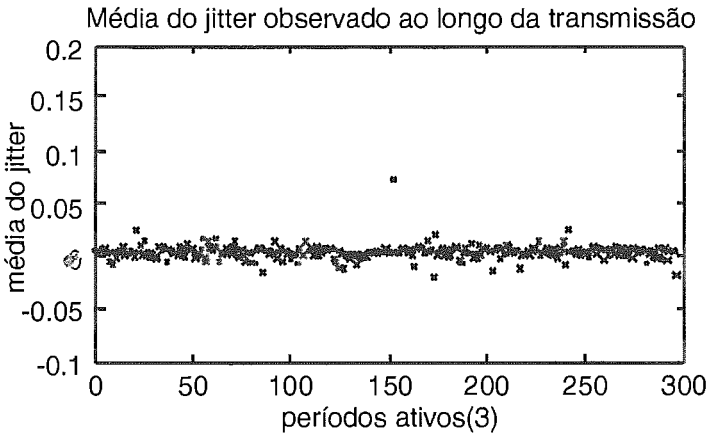
O primeiro gráfico (1) apresenta
medidas quando foi usado o
escalonamento WFQ.



No (2), foi usado o escalonamento C

O terceiro(3) mostra a média do
jitter sem controle das comunicaçõe

Mais detalhes estão
na tabela ao final da seção.



5.4.2 Resumo das medidas realizadas

Conforme podemos observar pela tabela 5.5, o parâmetro que obteve uma melhora mais significativa foi a perda de pacotes, quando a política utilizada foi WFQ. Como observado na seção 5.2.1, a perda de pacotes não apresenta uma melhora significativa quando a política CBQ é utilizada devido ao comportamento da fonte (períodos ativos e de silêncio). Já quando a política WFQ é selecionada, podemos observar uma melhora significativa na perda. O jitter obteve uma diminuição de uma ordem de grandeza.

Tabela 5.5: Quadro comparativo dos experimentos de voz.

		CBQ	WFQ	S. Controle
	Núm.Per.Ativos.	**377**	**319**	**304**
	pac. enviados	**10747**	**9943**	**9661**
Espalhado	**pac. recebidos**	**9207**	**9305**	**7412**
	média do jitter(ms)	**1.362e-3**	**-1.691e-3**	**3.432e-3**
	var. do jitter	**3.583e-4**	**1.909e-5**	**4.906e-4**
	perda(%)	**14.32**	**6.42**	**23.88**
	Núm.Per.Ativos	314	342	339
	pac. enviados	9051	8973	9588
Em rajadas	pac. recebidos	8436	8401	8417
	média do jitter(ms)	6.999e-3	2.789e-4	1.536e-3
	var. do jitter	6.369e-4	3.351e-4	6.605e-4
	perda(%)	6.79	6.37	12.21

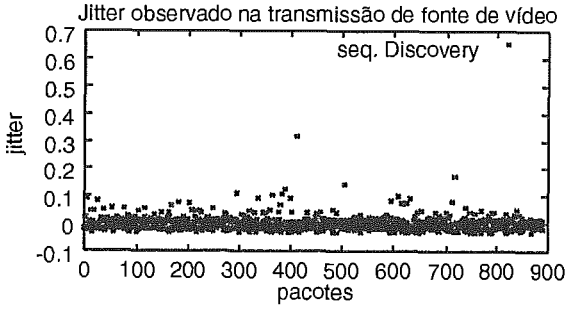
5.4.3 Fonte de vídeo

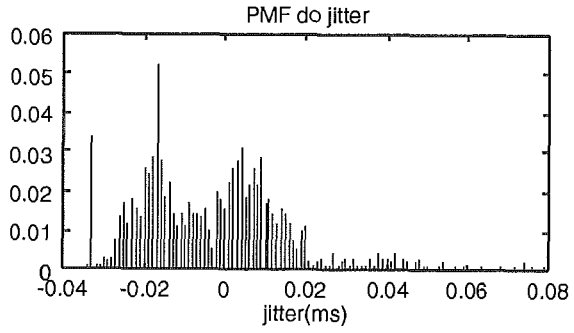
As experiências com tráfego de vídeo, também para o ambiente remoto, contaram com a utilização das três seqüências anteriormente descritas [29]. Inúmeros foram os ensaios produzidos, com a alteração de vários parâmetros. O tráfego extra foi gerado no roteador LAMROUTER. O número de fontes geradoras variou em cada seqüência. Para a seqüência **Discovery** foram usadas 5 fontes com taxa média de 6M*bits*/s no total, tanto para tráfego espalhado como para o em rajadas. Para a seqüência **Tangram2**, foram 6 fontes geradoras de tráfego extra, perfazendo uma taxa média de 4.8 M*bits*/s. Para o filme **Starwars**, foram 4 fontes, apenas de tráfego determinístico espalhado, com 6.4M*bits*/s de taxa média. A configuração do CBQ foi idêntica à realizada no ambiente controlado, i.e., duas classes principais requisitando

20% da banda para a transmissão das seqüências e 80% para os demais fluxos. Através do RSVP, foi requisitada uma reserva com 1.5M*bits*/s de banda alocada para as seqüências **Discovery** e **Tangram2**, correspondendo às taxas média e de pico das seqüências respectivamente e 1.0M*bits*/s para o filme Starwars, sendo esta sua taxa de pico. Para o WFQ manteve-se a mesma relação de pesos do item anterior: 60/30, respectivamente fontes extras/fontes de vídeo. Vejamos alguns gráficos com os resultados mais significativos e ao final, mais um quadro comparativo com um resumo dos resultados obtidos.

1) Seqüência Discovery, com tráfego extra espalhado:

CBQ





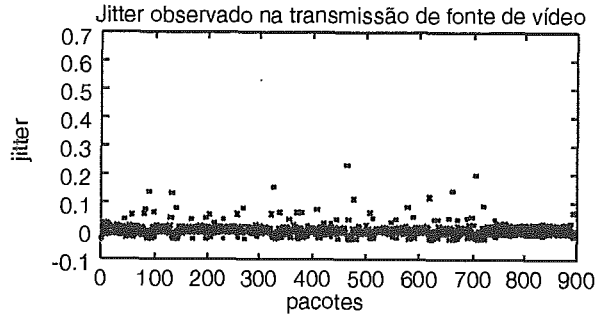
Jitter observado na transmissão da da sequência Discovery.

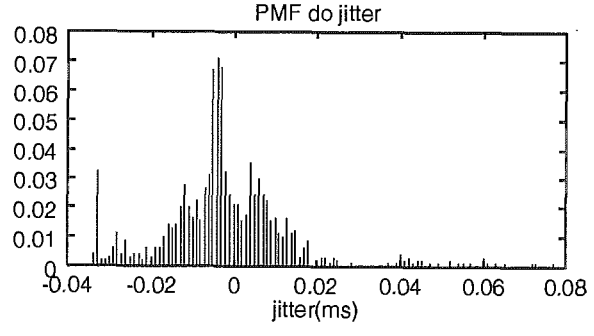
Nesse exemplo foi empregado um controle pelo escalonamento CBQ. Através do RSVP, requisitou-se 500kbps de banda para a aplicação.

Média do jitter: 2,178e-4 ms
Variância: 6,450e-4

Foram perdidos 7 pacotes em 900, correspondendo a uma perda de 0,77%

WFQ



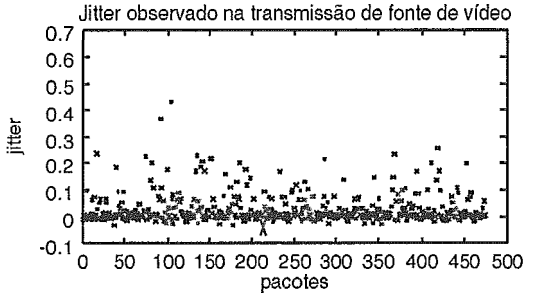


Jitter observado na transmissão da fonte de vídeo, sequência Discovery, utilizando a política de escalonamento WFQ.

Média do jitter: 1,368e-4 ms
Variância: 4,916e-4

Foram perdidos 2 pacotes em 900, com perda de 0,22%

Sem escalonamento da saída

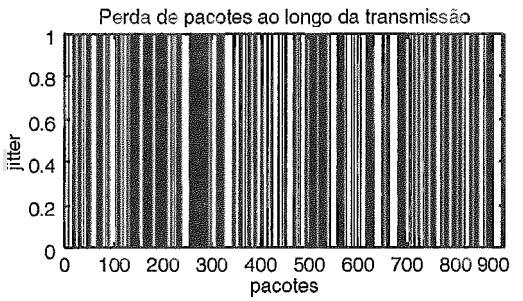


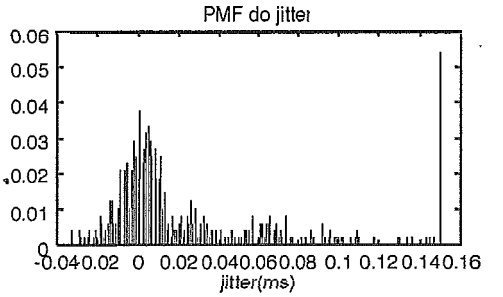
Jitter observado na transmissão da fonte de vídeo.

Nenhum controle adicional foi requisitado.

Média do jitter: 1,956e-1 ms
Variância: 1,833e-2

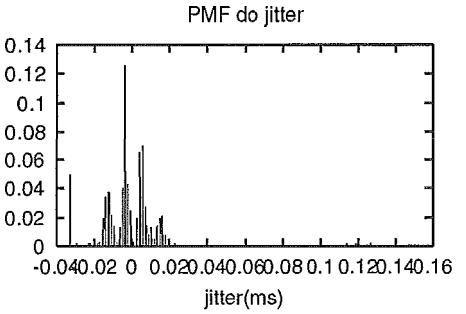
Foram perdidos 420 pacotes em 900 enviados: perda de 46,67%





2)Seqüência Discovery, com tráfego extra em rajadas:

CBQ

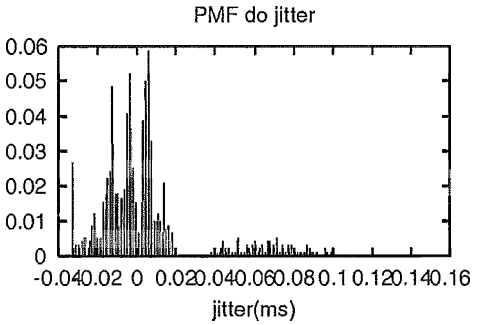


PMF do jitter na sequência Discovery, escalonamento da saída segundo política CBQ.

Média do jitter = 9.538e-4 ms
Variância do jitter = 4.957e-4

Houve registro de perda de 1 pacote.

WFQ

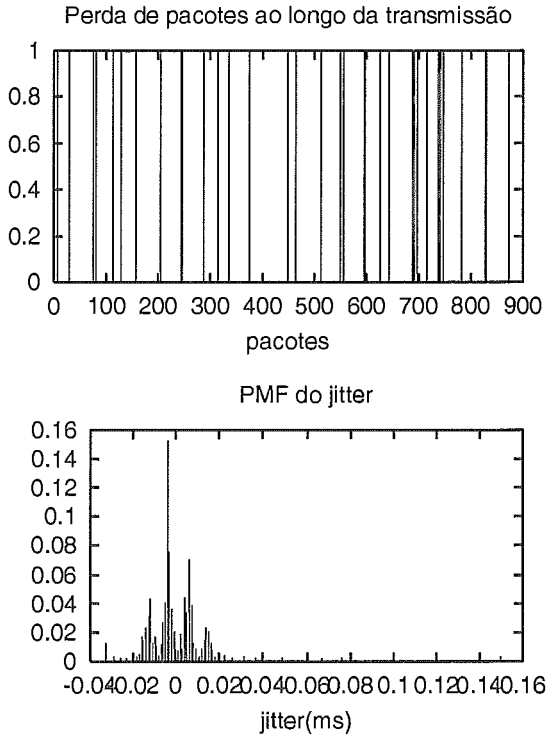


PMF da sequência Discovery. Foi utilizado escalonamento WFQ da saída.

Média do jitter = 3.955e-3 ms
Variância do jitter = 6.348e-4

Não houve perda de pacotes.

Sem escalonamento



Nenhuma qualidade de serviço foi requisitada à rede.

Média do jitter = 2.793e-3 ms
Variância do jitter = 4.834e-4

Pacotes recebidos: 856 em 900.

Perda: 4,88%

Quadro comparativo dos resultados fonte Discovery

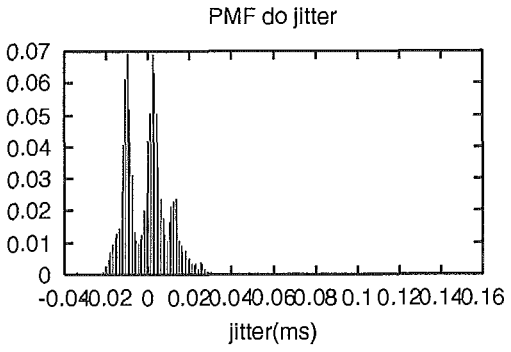
Tabela 5.6: Resumo dos resultados obtidos para fonte de vídeo Discovery.

		CBQ	WFQ	S. Controle
	Méd. do jitter(ms)	**2.178e-4**	**1.368e-4**	**1.956e-1**
	Var. do jitter(ms)	**6.450e-4**	**4.916e-4**	**1.833e-2**
Espalhado	**Pac. enviados**	**900**	**900**	**900**
	Pac. recebidos	**893**	**898**	**580**
	perda(%)	**0.77**	**0.22**	**46.67**
	Méd. do jitter(ms)	9.538e-4	3.995e-3	2.793e-3
	Var. do jitter(ms)	4.957e-4	6.348e-4	4.834e-4
Em rajadas	Pac. enviados	900	900	900
	Pac. recebidos	899	900	856
	perda(%)	0.11	0	4.88

Conforme a tabela 5.6, observamos que houve melhora em todos os experimentos realizados em relação aos mesmos experimentos no ambiente sem controle.

3) Seqüência Tangram2, com o tráfego extra espalhado:

CBQ

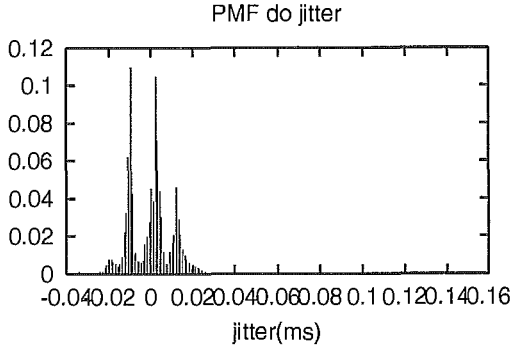


PMF do jitter observada na recepção do tráfego da fonte de vídeo.

Média do jitter = -3.897e-6 ms
Variância do jitter = 1.090e-4

Não houve perda de pacotes.

WFQ

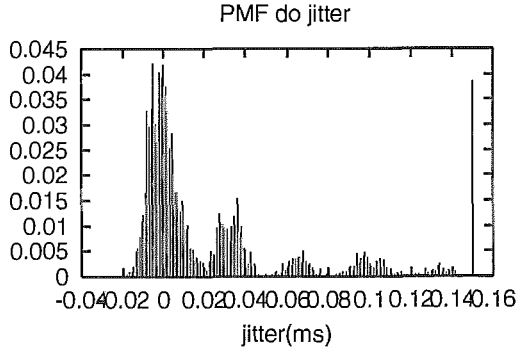


PMF do jitter observado na recepção do tráfego da fonte de vídeo, com saída usando política de escalonamento WFQ

Média do jitter = -1.097e-5 ms
Variância do jitter = 1.086e-4

Não houve perda de pacotes.

Sem controle



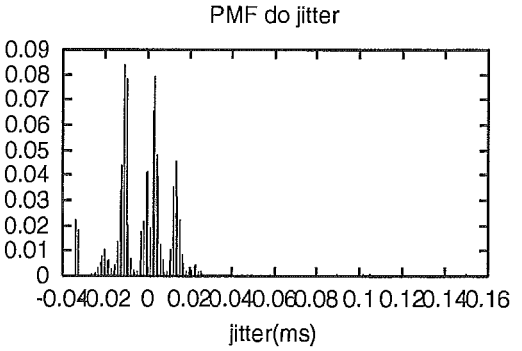
PMF do tráfego da fonte Tangram2. Nenhum controle das comunicações foi solicitado à rede.

Média do jitter = 2.614e-2 ms
Variância do jitter = 2.468e-3

Houve perda de 43.98%.

4) Seqüência Tangram2, com tráfego extra em rajadas

CBQ

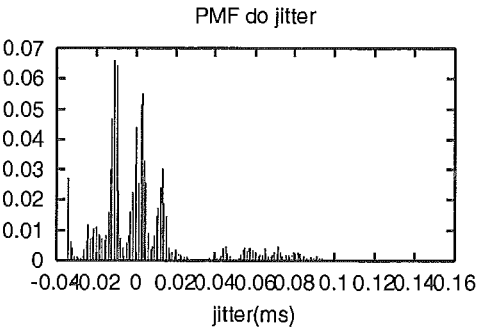


PMF da sequência Tangram2 na recepção.

Média do jitter = 1.029e-3 ms
Variância do jitter = 4.562e-4

Houve perda de 4 pacotes em 7115 enviados.

WFQ

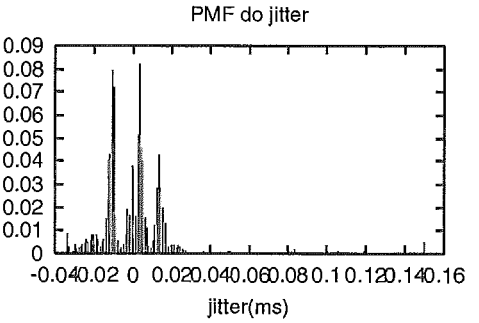


PMF da sequência Tangram2 observada na chegada.

Média do jitter = 3.914e-3 ms
Variância do jitter = 6.264e-4

Não houve registro de perdas na recepção.

Sem controle



PMF do jitter na chegada. Nenhum controle das comunicações foi solicitado à rede.

Média do jitter = 3.916e-3 ms
Variância do jitter = 7.352e-4

Foi registrada a chegada de 6729 pacotes de 7115 enviados, resultando em perda de 5,43%.

Quadro comparativo dos resultados fonte Tangram2

Através dos ensaios produzidos, conforme a tabela 5.7, observamos melhora da QoS significativa em todos os ensaios.

Tabela 5.7: Resumo dos experimentos com a fonte Tangram2.

		CBQ	WFQ	Sem Controle
	Méd. do jitter(ms)	**-3.897e-6**	**-1.097e-5**	**2.614e-2**
	Var. do jitter	**1.090e-4**	**1.086e-4**	**2.468e-3**
Espalhado	**Pac. enviados**	**7115**	**7155**	**7155**
	Pac. recebidos	**7115**	**7115**	**3986**
	perda(%)	**0**	**0**	**43.98**
	Méd. do jitter(ms)	1.029e-3	3.914e-3	3.916e-3
Em rajadas	Var. do jitter	4.562e-4	6.264e-4	7.352e-4
	Pac. enviados	7115	7115	7115
	Pac. recebidos	7111	7115	6729
	perda(%)	0.056	0	5.43

5) Starwars

A seqüência do filme foi transmitida com um tráfego extra determinístico espalhado. O gráfico à esquerda refere-se à PMF com a utilização da reserva e o da direita sem nenhum controle.

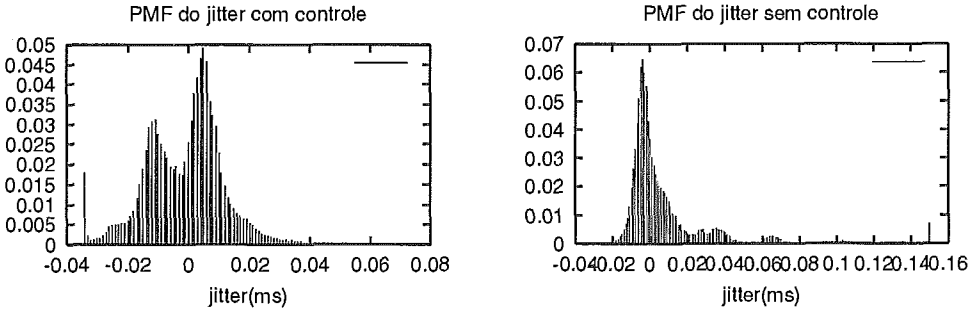


Tabela 5.8: Experimentos com o filme Starwars.

	CBQ	Sem Controle
Méd. do jitter(ms)	1.812e-6	9.771e-3
Var. do jitter	2.017e-4	9.626e-4
Pacotes enviados	174049	174049
Pacotes recebidos	174048	134616
Perda(%)	0	22.65

Podemos observar a grande variação para os parâmetros amostrados na tabela 5.8. Nesse experimento, a presença da reserva de recursos garantiu uma transmissão com uma QoS significativamente melhor em termos dos parâmetros verificados (jitter, sua média e perda de pacotes).

5.5 Ambiente Integrado

Nessa seção foram reunidos os testes utilizando ambos os ambientes acima descritos. Tivemos dois objetivos com a realização desses experimentos: testar o comportamento do ALTQ quando estilos de reserva diferentes são pedidos pelo RSVP e testar a QoS obtida pelas aplicações. Consideramos os seguintes cenários:

- Vários fluxos usando a mesma sessão com estilos de reserva diferentes;
- Vários fluxos usando sessões diferentes.

Além das máquinas cujas configurações estão previamente descritas, a máquina TRINDADE, fisicamente na mesma rede local de ARARUAMA foi utilizada. Essa máquina é composta de um PENTIUM 166MHz com 48M*Bytes* de memória. Também está configurada com o sistema operacional FreeBSD (v 3.1) e nela foi instalado o RSVP com módulo de controle de tráfego. A representação esquemática da figura 5.8, mostra esse ambiente integrado.

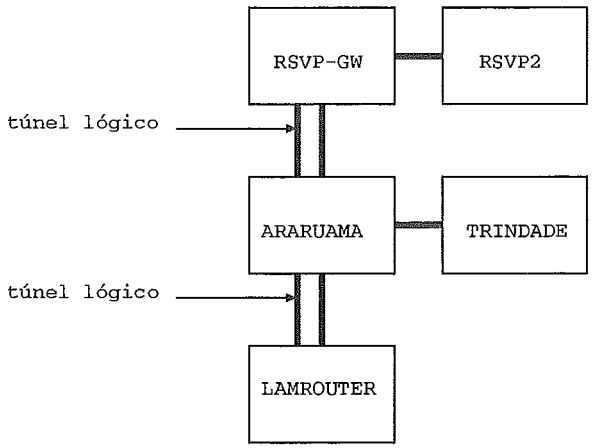


Figura 5.8: Representação esquemática do ambiente integrado utilizado.

A seguir apresentamos um resumo dos experimentos realizados.

5.5.1 Teste de estilos de reserva (mesma sessão RSVP)

Até agora, todos os testes realizados foram produzidos usando o estilo WF. Vejamos o comportamento das aplicações com os outros estilos. A seqüência **Tangram2**,

anteriormente utilizada, foi enviada segundo uma árvore multiponto formada pelos elementos da figura 5.8. A coleta dos dados no receptor mostra o comportamento dos parâmetros de QoS dos aplicativos, quando os fluxos dos transmissores foram compartilhados ou em situações de prioridade.

1) Estilo FF

Foram usados dois transmissores e um receptor, respectivamente TRINDADE, LAMROUTER e RSVP2. Para a realização dos eventos, houve alteração das bandas alocadas para as classes em RSVP-GW, uma vez que os fluxos necessariamente passariam por este elemento roteador. No arquivo de configuração, 40% da banda foi alocada para as classes dinâmicas e 60% para os demais fluxos. A representação de classes do experimento pode ser observada na figura 5.9, a seguir.

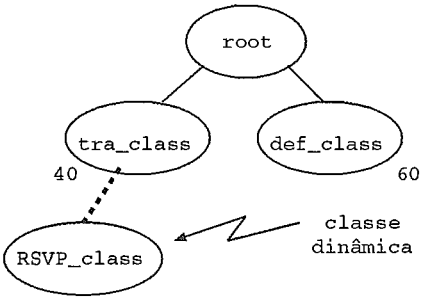


Figura 5.9: Representação de classes no elemento RSVP-GW.

Através do estilo FF, o receptor RSVP2 optou em receber o fluxo vindo de TRINDADE de maneira diferenciada, com 1.5Mbits/s de banda alocada. O fluxo vindo de LAMROUTER foi recebido passando pela classe *default*, sofrendo influência da presença de um tráfego determinístico espalhado com 4.3M*bits*/s de taxa. Foi observado o correto processamento das mensagens de PATH e RESV em todos os pontos do percurso de dados, bem como a criação de classes dinâmicas no CBQ à medida que a mensagem de RESV percorria seu trajeto de RSVP2 em direção à TRINDADE. O fluxo saído de LAMROUTER não obteve nenhum controle diferenciado. Foram reunidas na tabela 5.9 os resultados obtidos com o experimento acima. O significado das colunas é o seguinte:

1 aplicação entre TRINDADE e RSVP2, com reserva;

2 aplicação entre LAMROUTER e RSVP2 pela *default*;

3 aplicação entre TRINDADE e RSVP2, sem reserva e

4 aplicação entre LAMROUTER e RSVP2, sem reserva;

Tabela 5.9: Resultados obtidos para o experimento com o estilo FF.

parâmetro de QoS	1	2	3	4
Média jitter	2.228e-3	2.558e-3	3.457e-3	3.352e-3
Var. da média	8.332e-4	3.099e-3	1.066e-3	1.147e-3
Perda (%)	3.78	6.58	6.58	7.64

Podemos verificar através dos resultados da tabela 5.9, que a seqüência obteve a melhor QoS foi a número 1, ou seja, aquela que usou a classe com maior prioridade.

2) Estilo SE

O experimento para testar o estilo SE utilizou três transmissores e um receptor, respectivamente TRINDADE, LAMROUTER, ARARUAMA e RSVP2. A mesma hierarquia de classes foi usada para esse estilo, conforme a figura 5.9, contudo as bandas do escalonamento em RSVP-GW foram alteradas para 50/50, uma vez que o tráfego aumentaria para a classe RSVP dinâmica. Os dois primeiros transmissores foram escolhidos para compartilhar a reserva, tendo esta sido pedida pelo RSVP com 4.0M*bits*/s de banda. O tráfego gerado em ARARUAMA foi escoado pela classe *default* da máquina RSVP-GW, sofrendo influência de uma fonte de tráfego determinístico espalhado com 4.3M*bits*/s de banda. Foi observado o correto envio/recebimento das mensagens de PATH/RESV, configurando o sistema conforme o esperado. Na tabela 5.10, a seguir, teremos os resultados do experimento. O significado das colunas é o seguinte:

1 Aplicação entre TRINDADE e RSVP2 com reserva;

2 Aplicação entre LAMROUTER e RSVP2 com reserva;

3 Aplicação entre ARARUAMA e RSVP2 pela *default*;

4 Aplicação entre TRINDADE e RSVP2 sem reserva;

5 Aplicação entre LAMROUTER e RSVP2 sem reserva e

6 Aplicação entre ARARUAMA e RSVP2 sem reserva.

Tabela 5.10: Experimento com estilo SE.

	1	2	3	4	5	6
Méd. jitter	2.038e-3	8.899e-4	1.186e-3	4.549e-3	3.135e-3	3.871e-3
Var. Média	1.000e-3	8.022e-4	9.444-4	7.26e-4	7.04e-4	1.159e-3
Perda(%)	1.37	1.64	2.61	9.71	8.63	9.34

3) Estilo WF

O mesmo experimento anterior foi repetido para obtermos os resultados comparativos do estilo WF. Novamente as porções de banda alocada em RSVP-GW foram alteradas, agora para 60/40. A mesma organização do experimento anterior será repetida neste ítem, com exceção da banda pedida pelo RSVP que foi de 3M*bits*/s, porém teremos apenas os resultados utilizando a reserva.

1 Aplicação entre TRINDADE e RSVP2 com reserva;

2 Aplicação entre LAMROUTER e RSVP2 com reserva e

3 Aplicação entre ARARUAMA e RSVP2 sem reserva.

A comparação deverá ser realizada com as colunas 4, 5 e 6 do ítem anterior.

Figura 5.10: Experimento com o estilo WF.

	1	2	3
Méd. do jitter	2.145e-3	1.452e-3	1.080e-3
Var. do jitter	8.601e-4	9.11e-4	7.552e-4
Perda(%)	3.68	3.43	3.58

5.5.2 Sessões RSVP distintas

Para testarmos se a configuração de sessões RSVP distintas está realmente funcionando, foram realizados dois experimentos simples. Utilizando três transmissores e um receptor, novamente TRINDADE, LAMROUTER, ARARUAMA e RSVP2, foram organizados dois grupos, onde criou-se duas reservas distintas. O primeiro grupo formado pelos transmissores TRINDADE e LAMROUTER, enviou pacotes a RSVP2. A reserva, partindo de RSVP2 deu-se segundo o estilo WF. O segundo grupo foi criado tendo ARARUAMA como transmissora e também RSVP2 como

receptor, este utilizando o estilo de reserva FF. Todos os fluxos foram roteados por RSVP-GW. Foi observado o correto envio/recebimento de mensagens RESV e PATH entre os participantes, além da esperada configuração do CBQ em RSVP-GW, conforme a figura 5.11 abaixo:

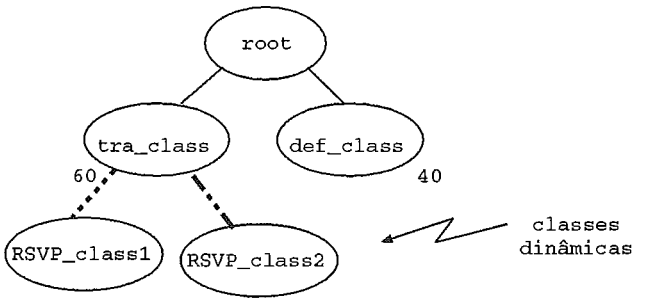


Figura 5.11: Representação de duas sessões RSVP distintas para o CBQ.

O primeiro experimento foi realizado com uma fonte de tráfego extra de 4.3M*bits*/s em RSVP-GW sendo escoada pela classe *default*. Através do RSVP, foi requisitada uma reserva de recursos de 2M*bits*/s de banda para o grupo com dois transmissores e uma reserva de 1M*bits*/s entre os participantes do segundo grupo. A configuração do escalonador CBQ colocou 60% da banda à disposição para a criação das classes dinâmicas e o restante foi entregue à classe *default*. O cenário do segundo experimento foi o mesmo, porém sem a presença da fonte de tráfego extra em RSVP-GW. Apresentaremos a seguir os resultados resumidos ena tabela 5.11. Temos os seguintes significados para as colunas:

1 Aplicação entre TRINDADE e RSVP2, com reserva na sessão 224.9.9.10/8500

2 Aplicação entre LAMROUTER e RSVP2, com reserva na sessão 224.9.9.10/8500

3 Aplicação entre ARARUAMA e RSVP2, com reserva na sessão 224.9.9.10/9500

Os experimentos 4,5 e 6 possuem o mesmo significado, porém representam as transmissões sem a presença de carga extra passando pela classe *default* de RSVP-GW.

Pelos resultados obtidos nos experimentos realizados no ambiente integrado, observamos que o programa RSVP, juntamente com seu módulo de controle de tráfego, consegue realizar corretamente o processamento das mensagens de RESV/PATH em todos os pontos do percurso nos quais o programa está instalado e observamos uma

Tabela 5.11: Resultados comparativos com duas sessões RSVP.

	1	2	3	4	5	6
Méd. jitter	1.497e-3	2.877e-3	2.349e-3	9.522e-4	1.872e-3	9.167e-4
Variancia	8.159e-4	1.323e-3	1.063e-3	6.753e-4	8.505e-4	7.778e-4
Perda	4.34	4.36	3.56	2.83	2.82	2.06

pequena melhora, principalmente do parâmetro de desempenho perda de pacotes em todos os ensaios realizados.

Capítulo 6

Conclusões

O objetivo do trabalho realizado foi investigar a QoS obtida para aplicações de tempo real quando estas utilizassem um ambiente com reserva de recursos e controle de tráfego.

O ambiente estudado foi composto de um agente gerenciador de recursos e de um módulo para fazer o controle do tráfego. O gerenciador de recursos escolhido foi o protocolo RSVP por ser o padrão proposto para uso na Internet. O módulo de controle de tráfego usado foi o software ALTQ que implementa classificação de pacotes, políticas de escalonamento (CBQ e WFQ) e controle de admissão. O ALTQ foi escolhido por dois motivos principais: possui interface com o RSVP e está implementado na plataforma FreeBSD.

Os experimentos foram realizados em um ambiente controlado onde uma ou mais máquinas continham as aplicações fontes, uma máquina atuava como roteadora e uma atuava como receptora. O RSVP foi instalado em todas as máquinas do ambiente de forma que a reserva pudesse ser feita ao longo do caminho. Para criar situações de congestionamento no roteador, foram utilizadas fontes geradoras de tráfego "artificial".

A partir dos resultados obtidos neste trabalho, podemos concluir qual a QoS que seria obtida na Internet caso o protocolo RSVP fosse usado pelas aplicações e caso fosse implementada uma política de classificação e escalonamento de fluxos (CBQ e WFQ) nas filas de saída dos roteadores.

Utilizamos dois tipos de aplicação nos testes: uma aplicação de transmissão de voz e uma aplicação de vídeo. Os parâmetros de desempenho medidos foram a distribuição

do jitter, sua média e variância e a proporção de pacotes perdidos.

Os experimentos realizados com a aplicação de voz utilizando o ambiente com controle (política CBQ) não evidenciaram uma melhora significativa na perda de pacotes, em relação ao ambiente sem controle. Isto decorreu do fato da política CBQ computar o tamanho do *buffer* dinamicamente como uma função do valor máximo especificado e do valor medido para o *buffer*. Portanto, no início do período ativo da fonte, o tamanho do *buffer* é muito pequeno, ocasionando perda de pacotes. O fato de aumentarmos o tamanho máximo permitido para o *buffer* melhorou a perda de pacotes. Quando utilizamos a política WFQ, obtivemos melhora significativa de todos os parâmetros da QoS.

Para a aplicação de vídeo, utilizamos três seqüências MPEG com características de tráfego diferentes nos experimentos. Duas eram filmes de ação (duração curta e longa) e uma continha uma palestra. Em todos os experimentos realizados (usando a política CBQ e WFQ) obtivemos melhora da QoS. A média e a variância do jitter diminuíram de uma até três ordens de grandeza. A perda caiu de valores em torno de 50% para valores abaixo de 1% evidenciando a melhora significativa no ambiente com controle.

Comparando os resultados obtidos com as seqüências de vídeo e voz quando a política CBQ foi utilizada, podemos fazer as seguintes observações:

- No caso da fonte de tráfego apresentar períodos de silêncio e períodos ativos (fonte de voz) não basta reservar um percentual da banda que seja proporcional à taxa média da aplicação. O fator que determina a melhora na QoS em termos do paràmetro perda de pacotes é o tamanho máximo especificado para o *buffer*.
- No caso de fontes de tráfego que apresentem uma taxa variável no tempo (ex. as seqüências de vídeo) porém os valores da taxa não cheguem próximos do zero, uma reserva do percentual da banda proporcional à taxa média da fonte garante uma melhora significativa da QoS.

Como conclusão dos experimentos, podemos dizer que a política CBQ é bastante complexa no que diz respeito à configuração dos parâmetros e ao algoritmo implementado para escalonar os pacotes (são usadas diversas variáveis calculadas dinamicamente a partir da configuração dos parâmetros e do estado das filas), enquanto que a política WFQ é mais simples e oferece praticamente a mesma QoS do que a obtida com o CBQ.

O uso da política CBQ é interessante no caso de necessidade de uso de uma estrutura hierárquica de classes. Para ilustrar esse caso, foi realizado um experimento maior, com várias máquinas criando uma arquitetura mais complexa, onde foi possível realizar reserva diferenciada para vários fluxos, usando os estilos de reserva do RSVP. Todos os testes produziram resultados compatíveis com o esperado, tanto em relação ao processamento das mensagens RSVP pelos participantes, quanto pela configuração das classes apresentada pelo CBQ. No que diz respeito aos parâmetros de QoS, foi observada uma melhora pequena, principalmente em relação ao parâmetro desempenho de perda de pacotes, para aqueles fluxos que possuíam reserva.

Sem dúvida, a maior dificuldade para a realização do trabalho foi a montagem do laboratório de testes. Nessa tarefa, foram configurados os programas MROUTED, RSVP e ALTQ (e suas respectivas interfaces), criação de filtros para permitir a passagem dos pacotes através dos *firewalls*, além da montagem do equipamento (*hardware*) em si. Em todas essas ações houve necessidade de conhecimento da senha de superusuário das máquinas utilizadas, por si só uma situação delicada, uma vez que envolve questões de segurança do *sites*. O ambiente de testes ideal para experimentos mais elaborados deve contar com a presença de mais máquinas em pontos distantes, permitindo a utilização do próprio tráfego normal da Internet como produtor de congestionamento e um grupo de pessoas, cada qual responsável por sua sub-rede.

Podemos citar como possíveis trabalhos futuros, no que diz respeito ao controle de tráfego, implementar mecanismos de controle de admissão mais adequados para trabalhar em conjunto com o gerenciador de recursos, implementação de outras políticas de escalonamento para a saída de pacotes, como por exemplo o algoritmo do *Virtual Clock*[32], testes específicos que permitam averiguar qual o impacto sobre as aplicações quando a QoS pedida é variável, estudo das mensagens de advertência transportadas pelo protocolo RSVP, a fim de observar a composição dos parâmetros de QoS ao longo do caminho e o desenvolvimento da interface entre as diversas aplicações de tempo real existentes no nosso grupo de pesquisa (Ferramenta para a transmissão de voz na Internet, Whiteboard e o Servidor de VoD) e o módulo de gerência de recursos (RSVP).

Referências Bibliográficas

[1] R. Braden, D. Clark, S. Shenker, *Integrated Services in the Internet Architecture: an Overview*, RFC 1633, julho 1994.

[2] J. Wroclawski, *Specification of the Controlled-Load Netwok Element Service*, Internet-Draft, novembro 1996. http://www.aciri.org/sally/admit.html

[3] S. Shenker, C. Partridge, R. Guerin, *Specification of Guaranteed Quality of Service*, Internet-Draft, agosto 1996.

[4] S. Shenker, J. Wroclawski, *General Characterization Parameters for Integrated Service Network Element*s, Internet-Draft, outubro 1996.

[5] C. Diot, W. Dabbous, J. Crowcroft, *"Multipoint Communication: A Survey of Protocols, Functions, and Mechanisms"*, *IEEE J. Select. Areas Commun.*, vol 15, pp. 277-290, abril 1997.

[6] S. Deering, *Host Extensions for IP Multicastin*g, RFC1112, agosto 1989.

[7] T. Pusateri, *Distance Vector Multicast Routing Protocol*, Internet-Draft, setembro 1996.

[8] L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zappala, "*RSVP: A New Resource ReSerVation Protocol*", *IEEE Network*, pp. 8-17, setembro 1993.

[9] B. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, *Resource ReSerVation Protocol - Version 1 Functional Specification*, setembro 1997, http://www.isi.edu/div7/rsvp/rsvp.html.

[10] J. Wroclawski, *The use of RSVP with IETF Integrated Services*, Internet-Draft, outubro 1996.

[11] L. Delgrossi, L. Berger, *Internet Stream Protocol Version 2 (ST2) Protocol Specification - Version ST2+*, RCF1819 Experimental, agosto 1995.

[12] D. Mitzel, D. Estrin, S. Shenker, L. Zhang, "*An Architectural Comparison of ST-II and RSVP*", *Proceedings of IEEE* Infocom 94, junho 1994.

[13] S. Deering, D. Cheriton, *"Multicast Routing in Datagram Internetworks and Extended LANs"*, *ACM Trans. Computer Systems*, Vol. 8, pp 85-110, maio 1990.

[14] J. Moy, *OSPF version 2*, RFC 1247, julho 1991.

[15] S. Deering et. all, *" An Architecture for Wide-Area Multicast Routing"*, *ACM SIGCOMM*, Londres, 1994, pp. 126-135.

[16] T. Ballardie, P. Francis, J. Crowcroft "*Core Based Trees (CBT) An Architecture for Scalable Inter-Domain Multicast Routing*", *ACM SIGCOMM*, Ithaca, pp 85-95, 1993.

[17] S. Shenker, L. Breslau, "*Two Issues in Reservation Establishment*". *Proceedings of ACM SIGCOMM' 95*, Cambridge, MA, agosto 1995.

[18] D. Clark, "*The Design Philosophy of the DARPA Internet Protocols*", *Proceedings of ACM SIGCOMM' 88*, agosto 1988.

[19] Z. Wang, J. Crowcroft, *QoS Routing for Supporting Resources Reservation*, outubro, 1994. http://www.cs.ucl.ac.uk/staff/J.Crowcroft/jons-papers.html

[20] D. Zappala, *RSRR: A Routing Interface for RSVP*, Internet-Draft, novembro, 1996.

[21] Y. Bernet et all., *A Framework for Use of RSVP with Diff-serv Networks*, Internet-Draft, junho, 1998.

[22] S.Floyd, V. Jacobson, "*Link-sharing and Resource Management Models for Packet Network*", *IEEE/ACM Transactions on Networking*, vol. 3 No. 4, agosto 1995.

[23] Ian Wakeman, et all., *Implementing Real Time Packet Forwarding Policies using Streams*, novembro, 1994. http://www.aciri.org/sally/cbq.html

[24] S. Floyd, *Comments on Measurement-based Admissions Control for Controlled-Load Services*, draft, julho, 1996. http://www.aciri.org/sally/cbq.html

[25] K. Cho, *"A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers"*, Proceedings of USENIX 1998, Annual Conference, New Orleans LA, June 1998.

[26] D. Bertsekas, R. Gallager, *Data Networks*, Prentice-Hall International Editions.

[27] D.W.Wall, *Mechanisms for Broadcast and Selective Broadcast*, Tese de PhD, Universidade de Stanford, Departamento de Eng. Elétrica, 1980.

[28] K. Cho, ALTQ - *TIPS.txt*, draft contido no pacote ALTQv.1.1.3.

[29] Seqüências de vídeo MPEG consultar em *ftp:ftp.land.ufrj.br/pub/rsvp/seqs* (anonymous)

[30] LI, S., "*Study of Information Loss in Packet Voice Systems*", IEEE Transactions on Communications, v.37, N.11, pp.1192-1202, November 1989.

[31] H. Zhang, "*Services Disciplines for Guaranteed Performance Service in Packet-Switching Networks*", Proceedings of the IEEE, 83(10), outubro 1995.

[32] L. Zhang, Virtual Clock: "*A New Traffic Control Algorithm for Packet Switching Networks*", Proc. of ACM SIGCOMM'90, pp. 19-29, setembro 1990.

[33] Partridge, Craig, Gigabit Networking, Addison Wesley.

Apêndice A

Mensagens RSVP

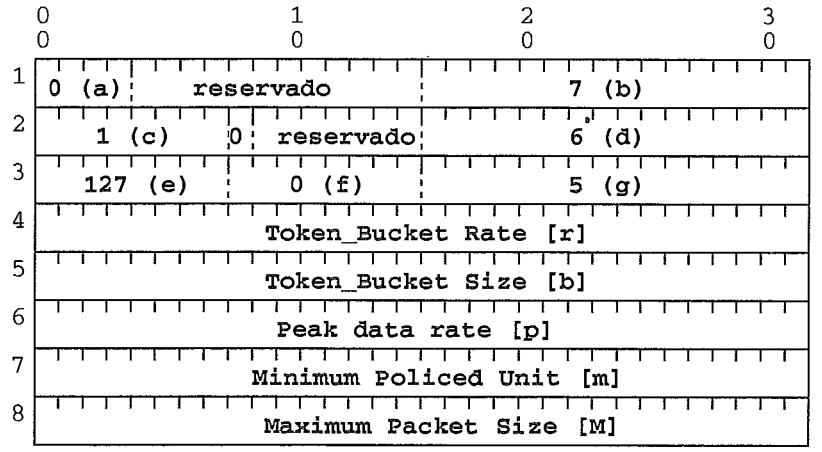
Serão mostrados três principais objetos RSVP em um ambiente utilizando serviços de QoS de carga controlada e serviços garantidos. A definição de outros tipos de serviços de QoS definirão um novo conjunto de regras para os objetos RSVP, uma vez que eles são opacos para o protocolo. Todos os objetos RSVP são antecedidos por um cabeçalho comum, descrito na figura 3.4.

A.1 Objeto SENDER_TSPEC

Este transporta informações sobre o tráfego gerado pela aplicação fonte. Ele contém o parâmetro de QoS global Token_Bucket_TSpec com o par (**número_do_serviço =1** (***default***), **número_do_parâmetro=127**), como definido em [4]. O objeto TSPEC transporta informações usáveis tanto pelo serviço de QoS de carga controlada, quanto pelo serviço garantido, figura A.1.

A.2 Objeto FLOWSPEC

Este transporta informações sobre a reserva produzida no recepor. Varia conforme o serviço, para carga controlada temos o par (**número_do_serviço=5, número_do_parâmetro=127**), como definido em [4]. Para serviço garantido temos o par (**número_do_serviço=2, número_do_parâmetro=127**). Eles estão representados nas figuras A.2 e A.3 a seguir.



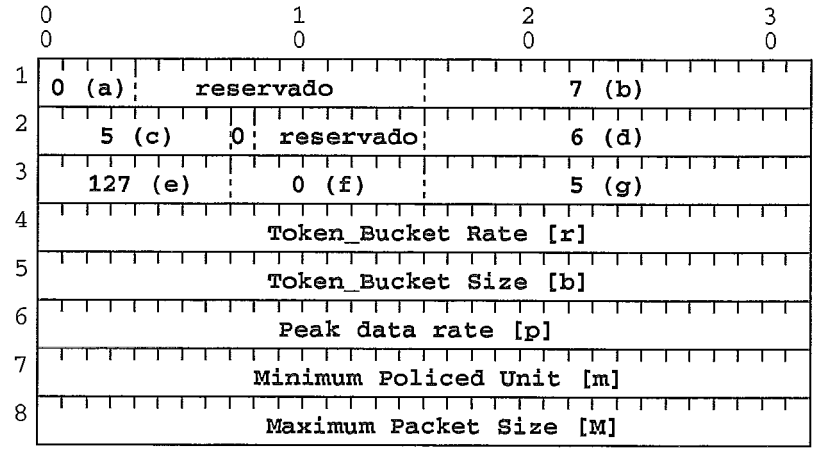
(a) Versão do formato da mensagem (valor = 0)
(b) Tamanho total (sem incluir o cabeçalho - valor = 7)
(c) Cabeçalho do serviço, serviço = 1 (informação global/default)
(d) Tamanho dos campos de dados do serviço, (valor = 6)
(e) Identificador do parâmetro de QoS (127 = Token_bucket_tspec
(f) Flags do parâmetro 127
(g) Tamanho do parâmetro (5 palavras não incluindo o cabeçalho)

Figura A.1: Organização do objeto sender_tspec.

A.3 Objeto ADSPEC

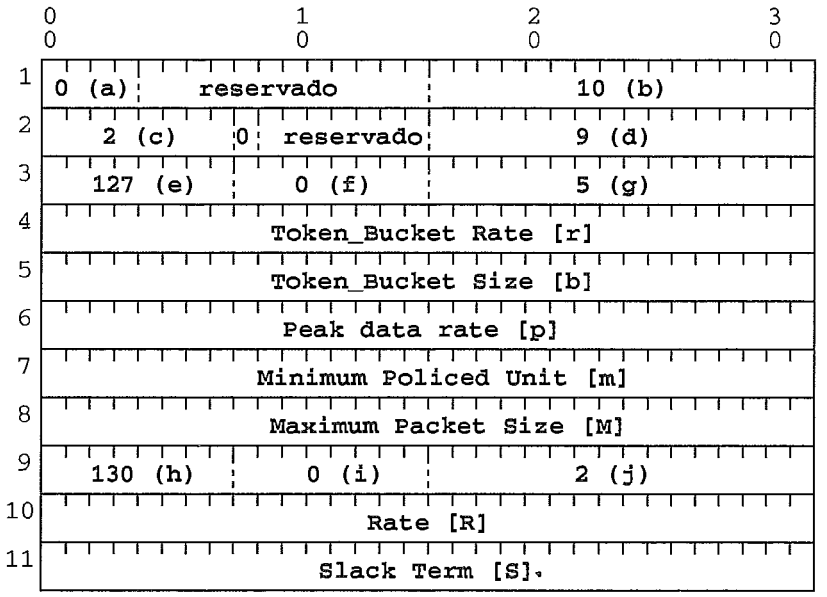
Este objeto é construído de informações sobre uma série de parâmetros relevantes em termos de QoS para aqueles elementos envolvidos em uma comunicação de dados. Ele é composto de um cabeçalho seguido por um fragmento destinado aos parâmetros gerais, seguido por fragmentos de serviço de controle de QoS que possa ser selecionado pela aplicação. Seu tamanho varia dependendo do número de parâmetros que irá representar.

A ausência de determinado parâmtro de um serviço particular indica que a aplicação transmissora não sabe ou não se importa sobre tal informação, indicando também que os nós intermediários também não acrescentaram ou atualizaram o objeto sobre esse parâmetro. Vejamos um esquema genérico em A.4. Foram incluídas em A.5, A.6 e A.7 informações sobre os fragmentos acrescentados no caso default e quando o serviço for de carga controlada e serviço garantido.



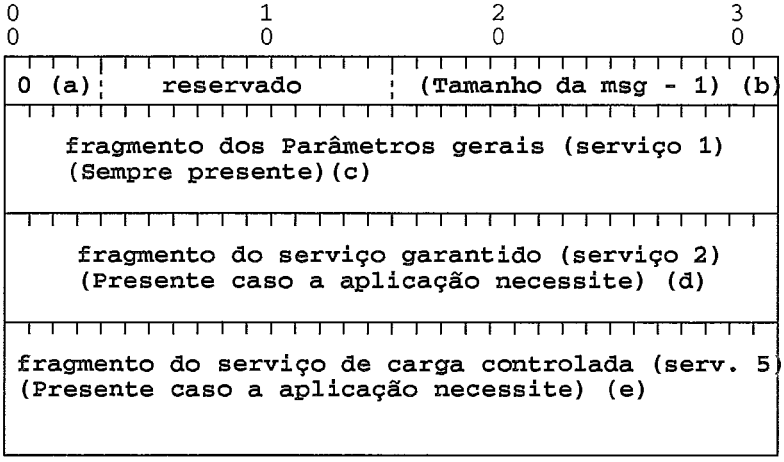
(a) Versão do formato da mensagem (valor = 0)
(b) Tamanho total (sem incluir o cabeçalho - valor = 7)
(c) Cabeçalho do serviço, serviço = 5 (carga controlada)
(d) Tamanho dos campos de dados do serviço, (valor = 6)
(e) Identificador do parâmetro de QoS (127 = Token_bucket_tspec)
(f) Flags do parâmetro 127
(g) Tamanho do parâmetro (5 palavras não incluindo o cabeçalho)

Figura A.2: Organização do objeto flowspec para o serviço de carga controlada.



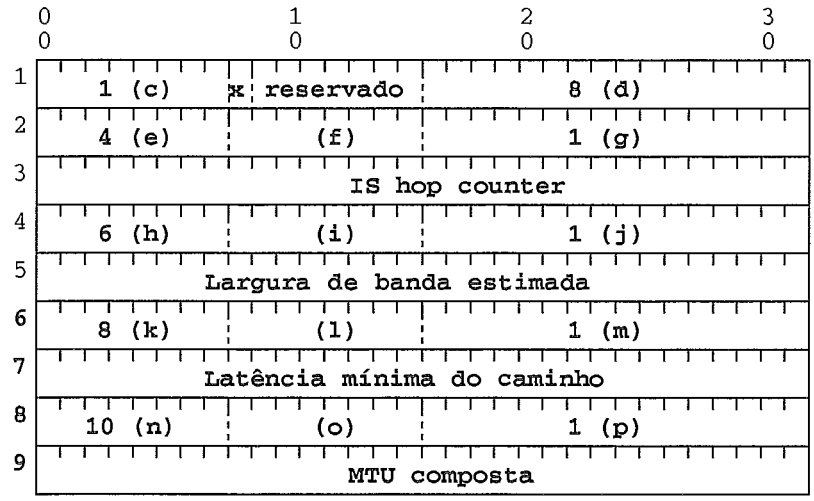
(a) Versão do formato da mensagem (valor = 0)
(b) Tamanho total (sem incluir o cabeçalho - valor = 10)
(c) Cabeçalho do serviço, serviço = 2 (serviço garantido)
(d) Tamanho dos campos de dados do serviço, (valor = 9)
(e) Identificador do parâmetro de QoS (127 = Token_bucket_tspec)
(f) Flags do parâmetro 127
(g) Tamanho do parâmetro (5 palavras não incluindo o cabeçalho)
(h) Indentificador do parâmetro de serviço garantido RSpec (130)
(i) Flags do parâmetro 130
(j) Tamanho do parâmetro 130 (valor=2, sem incluir o cabeçalho)

Figura A.3: Organização do objeto flowspec para o serviço garantido.



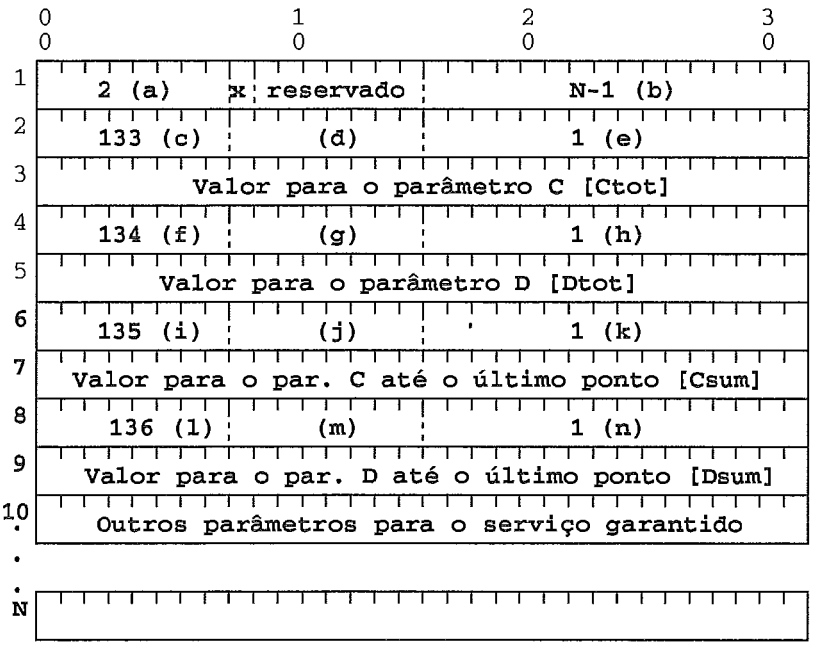
(a) Versão do formato da mensagem (valor = 0)
(b) Tamanho total (sem incluir o cabeçalho)
(c), (d) e (e) Fragmentos de dados

Figura A.4: Formato da mensagem de advertência geral.



(c) Serviço = 1 (Default)
(d) Global Break bit (marcado com x) tamanho do bloco de dados.
(e) Identificador do parâmetro (4 = IS-HOP)
(f) Flags do parâmetro 4
(g) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho)
(h) Identificador do parâmetro (6 = PATH-BW)
(i) Flags do parâmetro 6
(j) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho
(k) Identificador do parâmetro (8 = latência mínima do caminho)
(l) Flags do parâmetro 8
(m) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho
(n) Identificador do parâmetro (10 = MTU composta)
(o) Flags do parâmetro 10
(p) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho

Figura A.5: Formato do fragmento para parâmetros globais.



(a) Serviço = 2 (Garantido)
(b) Global Break bit (marcado com x) tamanho do bloco de dados.
(c) Identificador do parâmetro (133 = Composed Ctot)
(d) Flags do parâmetro 133
(e) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho)
(f) Identificador do parâmetro (134 = Composed Dtot)
(g) Flags do parâmetro 134
(h) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho
(i) Identificador do parâmetro (135 = Composed Csum)
(j) Flags do parâmetro 135
(k) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho
(l) Identificador do parâmetro (136 = Composed Dsum)
(m) Flags do parâmetro 136
(n) Tamanho do parâmetro (valor = 1) (não incluindo o cabeçalho

Caso a aplicação não utilize os serviços garantidos, o cabeçalho deverá existir com o campo do tamanho de mensagem = 0.

Figura A.6: Formato do fragmento para parâmetros do serviço garantido.

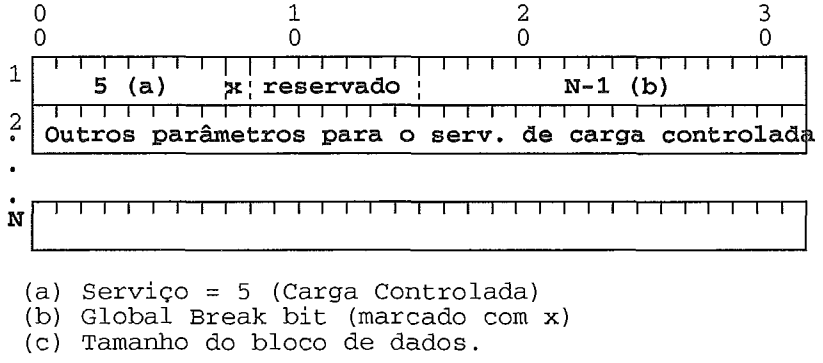


Figura A.7: Formato do fragmento para parâmetros do serviço com carga controlada.