

1.1 Transmissão multimídia em redes

Pode-se dividir a parte de transmissão multimídia em redes de computadores como mostra a figura 1, ou seja, a parte de conferência (que requer interatividade) e a parte de transmissão de vídeo (que envolve apenas um lado transmitindo e vários clientes recebendo). Ambas possuem necessidades diferentes para funcionarem a contento, por exemplo, as aplicações de conferência normalmente possuem necessidades mais rígidas em relação ao atraso da rede, enquanto que a transmissão unidirecional pode trabalhar com um atraso maior.

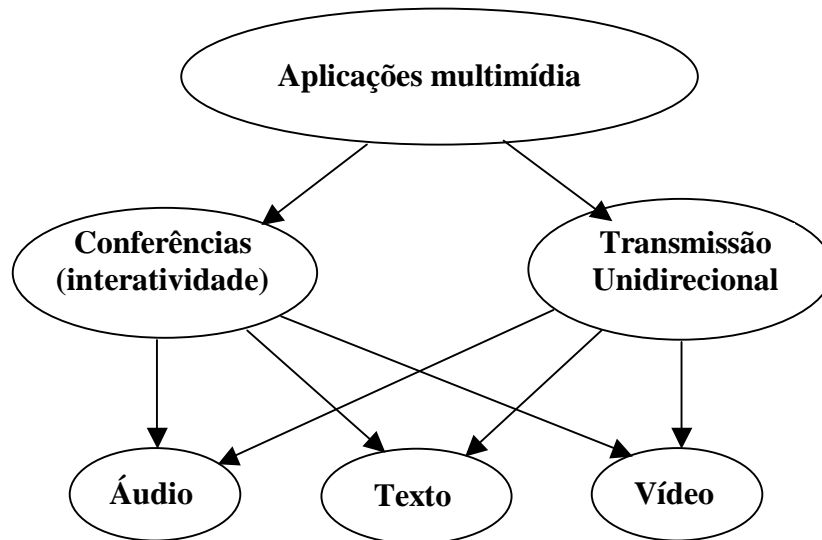


Figura 1: transmissão multimídia em rede

Nas próximas páginas serão analisados com maiores detalhes os conceitos e padrões envolvidos em transmissão multimídia em redes de computadores.

1.2 Necessidades das aplicações

Atualmente existe uma tendência de convergência de aplicações em um único meio físico, ou seja, voz, vídeo, dados, imagens, músicas, e tudo que possa ser transformado em bits utilizando o mesmo meio físico. Entretanto, as aplicações tem características e necessidades bem diferentes umas das outras, como por exemplo voz, que exige latência e *jitter* baixos, dados, que não tem tanta preocupação com latência e *jitter*, e videoconferência, que além de exigir latência e *jitter* baixos, ainda necessita de *skew* baixo, a fim de manter sincronizados voz e vídeo. A seguir será feita a definição desses termos, e depois será mostrada uma tabela comparativa das necessidades das aplicações.

1.2.1 Latência

Em redes de computadores, latência é o tempo que um pacote leva da origem ao destino. Caso esse atraso seja muito grande, prejudica uma conversação através da rede,

tornando difícil o diálogo e a interatividade necessária para certas aplicações. Segundo /PAS 97a/, que se baseou em muitos registros, um atraso confortável para o ser humano fica na ordem de 100ms.

Suponha duas pessoas conversando através da Internet. À medida que o atraso aumenta, as conversas tendem a se entrelaçar, ou seja, uma pessoa não sabe se o outro a ouviu e continua falando. Após alguns milissegundos vem a resposta do interlocutor sobre a primeira pergunta efetuada, misturando as vozes. Num atraso muito grande, as pessoas devem começar a conversar utilizando códigos, tipo “câmbio”, quando terminam de falar e passam a palavra ao outro.

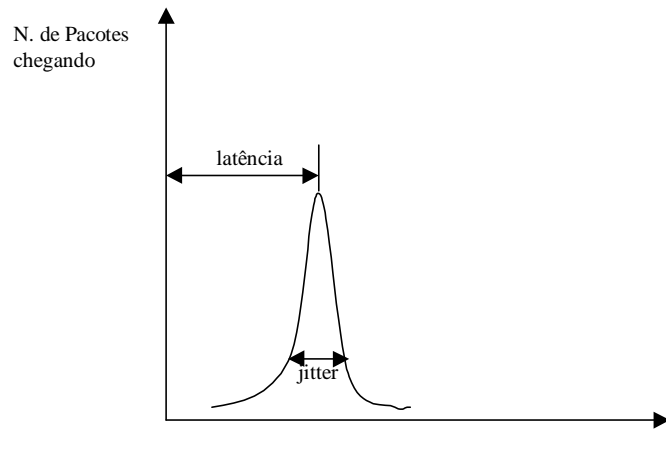
Os principais responsáveis pela latência são o atraso de transmissão, de codificação e de empacotamento, que podem ser definidos da seguinte forma:

- **Atraso de transmissão:** tempo após a placa de rede ter transmitido o pacote até ele chegar na placa de rede do computador destino. Esse tempo envolve uma série de fatores, como o atraso no meio físico (por exemplo fibra ótica, UTP, *wireless*), processamento em cada roteador ou switch intermediário (por exemplo, para trocar o TTL do pacote e decidir sua rota), fila de espera em cada roteador e switch intermediário, e assim por diante;
- **Atraso de codificação e decodificação:** sinais como voz e vídeo normalmente são codificados em um padrão, tipo PCM (G.711 a 64Kbps) para voz, ou H.261 para vídeo. Essa codificação gasta um tempo de processamento na máquina. Alguns protocolos gastam menos, como o G.711, que ocupa menos de 1ms de codificação /PAS 97a/, porém, requer 64Kbps de banda. Alguns protocolos de voz, como o G.729, requerem 25ms de codificação, mas ocupam apenas 8Kbps de banda /PAS 97a/;
- **Atraso de empacotamento e desempacotamento:** após codificado, o dado deve ser empacotado na pilha OSI a fim de ser transmitido na rede, e isso gera um atraso. Por exemplo, numa transmissão de voz a 64Kbps, ou 8000 bytes por segundo, tem-se que, para preencher um pacote de dados contendo apenas 100 bytes, vai levar 12,5ms. Mais 12,5ms serão necessários no destino a fim de desempacotar os dados.

Além da latência, a existência do *jitter* é outro fator de atraso na comunicação entre duas pessoas.

1.2.2 Jitter

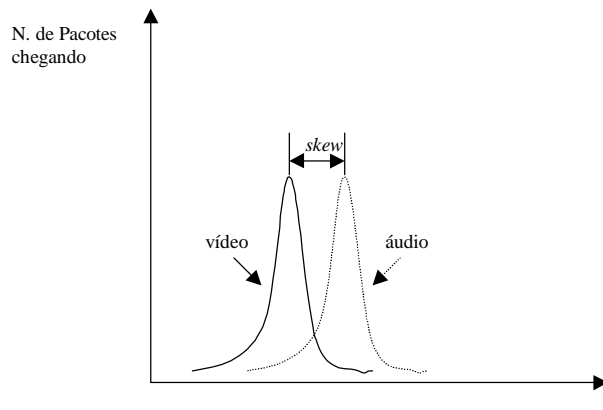
Utilizar somente a latência não é suficiente para definir a qualidade de transmissão, pois as redes não conseguem garantir uma entrega constante de pacotes ao destino. Assim, os pacotes chegam de forma variável, como mostra a figura 2, ocasionando o *jitter*, que nada mais é do que uma flutuação na latência, ou variação estatística do retardo /MOU 99/.

Figura 2: comparação entre latência e *jitter*

A consequência do *jitter* é que a aplicação no destino deve criar um *buffer* cujo tamanho vai depender do *jitter*, gerando mais atraso na conversação. Esse *buffer* vai servir como uma reserva para manter a taxa de entrega constante no interlocutor. Daí a importância de latência e *jitter* baixos em determinadas aplicações sensíveis a esses fatores, como videoconferência.

1.2.3 “Skew”

O *skew* é um parâmetro utilizado para medir a diferença entre os tempos de chegada de diferentes mídias que deveriam estar sincronizadas, como mostra a figura 3 /MOU 99/. Em muitas aplicações existe uma dependência entre duas mídias, como áudio e vídeo, ou vídeo e dados. Assim, numa transmissão de vídeo, o áudio deve estar sincronizado com o movimento dos lábios (ou levemente atrasado, visto que a luz viaja mais rápido que o som, e o ser humano percebe o som levemente atrasado em relação à visão). Outro exemplo é quando tem-se uma transmissão de áudio explicativo e uma seta percorrendo a figura associada.

Figura 3: definição do *skew* entre mídias diferentes

1.2.4 Tabela comparativa

A tabela 1 mostra algumas aplicações típicas em rede, bem como seus fatores críticos.

Tabela 1 – fatores críticos em aplicações numa tendência de convergência nas redes

	Telefone	Download	TV	Videoconferência
Latência	Sensível	insensível	insensível	sensível
Jitter	Sensível	insensível	sensível	sensível
Skew	-	insensível	sensível	sensível
Velocidade (larg. banda)	baixa	depende	alta	alta

Aplicações de telefonia são sensíveis a latência e *jitter*. Caso estiverem associadas a sincronismo em alguma figura, como por exemplo um áudio explicativo associado a uma seta se movendo numa figura, o áudio também é sensível a *skew*. Possuem velocidade baixa, de 64Kbps no padrão G.711, o mais comum em telefonia atualmente, mas pode-se chegar a apenas 8Kbps, usando a compressão no padrão G.729.

Aplicações de *download* de dados são insensíveis a latência, *jitter* e *skew*, podem variar em necessidades de velocidade, e possuem taxa variável. Entretanto, na maior parte das vezes esse tipo de mídia não pode sofrer perdas. Pode-se imaginar o problema que pode acontecer de perdas de pacotes numa transação bancária.

Já em transmissões unilaterais de áudio e vídeo, como por exemplo TV, a latência não é tão importante, visto que não vai fazer muita diferença se a transmissão demorar 5 segundos para começar a passar. Entretanto, uma vez que começou, deve se manter até o final e com sincronismo entre áudio e vídeo, daí a necessidade de *jitter* e *skew* baixos.

Aplicações de videoconferência são muito parecidas com aplicações de voz em termos de latência e *jitter*, entretanto, possuem alta largura de banda e devem manter um baixo *skew*, pois necessitam sincronização entre áudio e vídeo.

1.3 Garantia de qualidade de serviço

Somente o aumento na largura de banda não é suficiente para garantir a qualidade do serviço à aplicação, pois em se tratando de redes compartilhadas por múltiplos usuários e muitas vezes a longas distâncias, podem haver congestionamentos, provocando atrasos inadmissíveis em certas aplicações sensíveis, como por exemplo voz e videoconferência.

Existem algumas formas de prover qualidade de serviço às aplicações críticas: serviços integrados, serviços diferenciados /MOU 99a/, prioridade relativa e *label switching*. Alguns desses protocolos serão detalhados brevemente a seguir e com mais detalhes na tese.

1.3.1 QoS através de serviços integrados

O grupo de trabalho de serviços integrados do IETF desenvolveu um modelo que inclui serviços de **melhor esforço** e de **tempo real**, descrito na RFC 1633. O serviço de tempo real é utilizado para prover qualidade de serviço em aplicações multimídia sobre

redes IP. O protocolo RSVP (*Resource Reservation Protocol*) permite uma reserva de recursos ao longo do caminho entre origem e destino, e será analisado a seguir.

1.3.1.1 RSVP – Resource Reservation Protocol

RSVP é um protocolo de controle que roda sobre IP, ocupando o lugar do protocolo de transporte, da mesma forma que o ICMP, IGMP ou protocolos de roteamento /BRA 97/.

As aplicações utilizam RSVP para reservar e manter durante a conexão uma determinada qualidade de serviço até a aplicação destino, e o RSVP faz isso criando um caminho entre origem e destino, perguntando a todos os nós intermediários se eles suportam a qualidade desejada, e reservando as necessidades daquela aplicação. Para tanto, todos os nós no meio do caminho devem suportar esse protocolo.

O protocolo RSVP utiliza outros protocolos para efetuar roteamento e transmissão. Seu objetivo único é a reserva, manutenção e liberação de recursos quando solicitado. Assim, pode operar em unicast, multicast, Ipv4, Ipv6, e outros. A figura 4 mostra os módulos necessários em uma implementação RSVP, e em seguida tem um resumo deles.

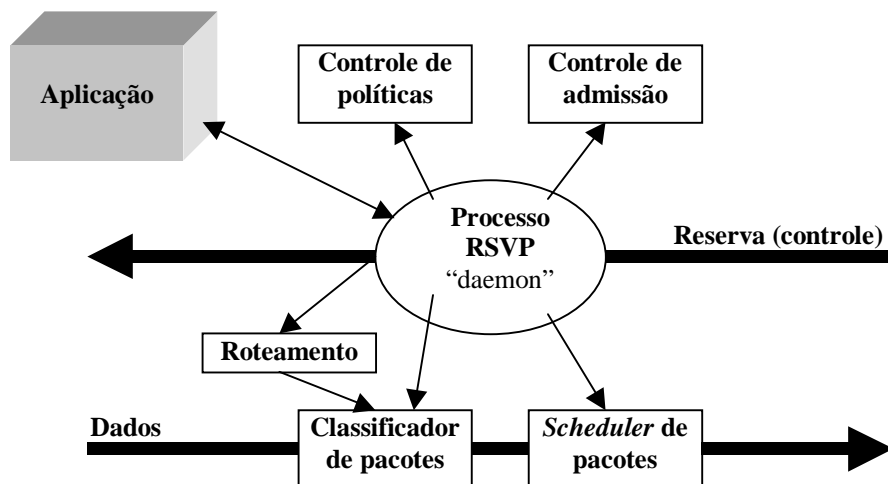


Figura 4: módulos necessários em uma implementação RSVP

- **Controle de admissão:** utilizado no início da chamada para verificar se o nó tem recursos suficientes para atender a qualidade de serviço solicitada;
- **Controle de políticas:** determina se o usuário tem permissão administrativa para fazer a reserva;
- **Classificador e scheduler de pacotes:** o classificador de pacotes determina a classe de QoS. Quando a solicitação passa pelo controle de admissão e políticas, são configurados alguns parâmetros nesses módulos, a fim deles reconhecerem os pacotes para ordenar corretamente na saída, dando a necessária qualidade de serviço para cada fluxo;
- **Fluxo de dados:** RSVP é simplex, ou seja, faz reservas para fluxos unidirecionais.

O processo do RSVP também se comunica com as rotinas de roteamento para determinar o caminho das solicitações de reserva. Isso causa um problema no caso de mudança de uma rota na tabela de roteamento (que é dinâmica), gerando uma necessidade de reserva (feita automaticamente) através do novo caminho. Para conseguir isso, o caminho estabelecido é do tipo “*soft state*”, necessitando mensagens periódicas para se manter. Na ausência de tais mensagens (por uma mudança de rota ou saída de cliente), a interface dá *time-out* e a reserva é liberada.

A reserva de QoS é feita na ordem reversa, ou seja, do receptor ao transmissor. Essa solicitação de reserva se repete até que chegue no transmissor ou encontre um nó com as mesmas necessidades. Tais reservas são implementadas através de dois tipos de mensagens: PATH e RESV /LIU 98/.

- **PATH:** mensagens enviadas periodicamente pelo transmissor ao endereço multicast. Contém a especificação de fluxo (formato de dados, endereço fonte, porta fonte) e características de tráfego. Essa informação é utilizada pelos receptores para achar o caminho reverso ao transmissor e determinar quais recursos devem ser reservados. Os receptores devem se cadastrar no grupo multicast a fim de receber mensagens PATH;
- **RESV:** mensagens geradas pelos receptores contendo parâmetros de reserva, como especificação de fluxo e de filtro. O filtro determina quais pacotes no fluxo de dados devem ser usados no classificador de pacotes. A especificação de fluxo é usada no *scheduler*, que procura manter a necessidade do receptor.

1.3.2 QoS através de serviços diferenciados

Ao contrário dos serviços integrados, vistos no item anterior, os padrões que utilizam serviços diferenciados preenchem um campo específico no pacote para dizer a prioridade que a aplicação tem. Assim, cada nó da rede pode priorizar ou não o encaminhamento do pacote. O diffserv, visto a seguir, é um exemplo dessa arquitetura.

1.3.2.1 Differentiated Services (*diffserv*)

O objetivo da arquitetura diffserv é melhorar o protocolo IP a fim de obter qualidade de serviço de uma forma escalável, ou seja, sem depender de protocolos de sinalização em cada nó ou reserva de recursos baseada em fluxo /FER 99/, NIC 98/. Para isso, é utilizado o campo DS (*Differentiated Services*) do cabeçalho IP, que nada mais é do que o campo TOS (*Type Of Service*) do cabeçalho IPv4 (RFC 791) ou o campo “classe de tráfego” do IPv6 (RFC 2460).

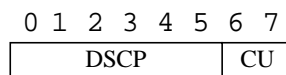
A configuração do campo DS pela aplicação vai determinar o comportamento do pacote na rede e seu tipo de serviço, ou seja, se ele vai ser mais ou menos prioritário.

Serviços são definidos em /NIC 98/ como necessidades dos usuários, tanto fim a fim como dentro de um mesmo domínio. Tais necessidades podem ser de largura de banda ou baseadas em desempenho relativo (uso de classes). O contrato de serviço entre o cliente e o provedor de serviços é conhecido como *Service Level Agreement* (SLA). Os serviços podem ser implementados de acordo com as seguintes diretivas:

- Marcar o campo DS do cabeçalho IP de acordo com o tipo de serviço. Usado nos limites da rede (limites de sistemas autônomos, limites administrativos internos ou *hosts*);
- Usar o campo DS para determinar a prioridade com que os pacotes serão encaminhados através dos nós da rede;
- Condicionar os pacotes nos limites da rede de acordo com os requerimentos de cada serviço, como por exemplo monitoramento, política e conformação de tráfego.

Dessa forma, cada pacote vai ter um comportamento num determinado nó, e isso é conhecido como “*per-hop behaviors*”, ou PHB. Muitos padrões de PHB estão em desenvolvimento no IETF /FER 99/, como por exemplo o *Expedited Forwarding* (EF), ou “encaminhamento agilizado” /JAC 99/, e *Assured Forwarding* (AF), ou “encaminhamento garantido” /HEI 99/.

O campo DS é mostrado na figura 5. Como pode ser visto, seis bits são usados como código para determinar o PHB que determinado pacote vai ter em cada nó da rede. Dois bits não são usados ainda.



DSCP: *differentiated services codepoint*

CU: *Currently Unused*

Figura 5: estrutura do campo DS

O campo DS visto acima é incompatível com a estrutura definida no campo TOS do IPv4, podendo provocar comportamentos de encaminhamento de pacotes indesejáveis caso o pacote seja IPv4 original. Para eliminar esse problema, está definido em /NIC 98/ um método de utilização do DS onde existe uma compatibilização ao campo TOS.

1.3.3 QoS através de Prioridade relativa

No modelo de prioridade relativa, a aplicação configura uma determinada prioridade (ou precedência) para o pacote, e os nós ao longo do caminho aplicam essa regra na hora de encaminhar o quadro. O comportamento que pode ser configurado é de atraso relativo ou prioridade de descarte.

A arquitetura Diffserv pode ser considerada um refinamento desse modelo, pois especifica com maiores detalhes a importância dos domínios de tráfego, bem como os condicionadores de tráfego.

Alguns exemplos desse tipo de QoS /BLA 98/ são o modelo de precedência do IPv4 definido na RFC 791, a prioridade das redes Token Ring (IEEE 802.5) e a interpretação das classes de tráfego dada no protocolo IEEE 802.1p, que será analisado a seguir.

1.3.3.1 Protocolo IEEE 802.1p/Q

O protocolo IEEE 802.1p é uma técnica para priorização de tráfego em redes locais, sendo especificado na norma IEEE 802.1D – LAN Bridges /CON 99/. Através dessa técnica, é possível utilizar aplicações sensíveis a tempo em ambientes LAN.

No IEEE 802.1p, estão definidas 8 classes de tráfego, como mostra a tabela 2. Como os pacotes Ethernet não possuem campos para priorização de tráfego, a norma 802.1p recomenda a utilização da extensão Ethernet para reconhecimento de VLANs, definida na norma 802.1Q. Essa norma adiciona 4 bytes ao pacote Ethernet a fim de reconhecimento de VLANs, e desses 4 bytes, 3 bits são reservados para priorização de tráfego.

Tabela 2: classes de prioridade 802.1p e tipos de tráfego associados

Prioridade	Binário	Tipos de tráfego
7	111	Controle da rede
6	110	Voz interativa
5	101	Multimídia interativa
4	100	Aplicações de carga controlada (<i>streaming</i>)
3	011	Esforço excelente
0	000	Melhor esforço (<i>default</i>)
2	010	Econômico
1	001	<i>Background</i>

1.3.3.2 Classes de serviço no ATM

No ATM, a qualidade de serviço está especificada na camada AAL (*ATM Adaptation Layer*). Como as aplicações possuem necessidades diferentes, como visto no item 1.2, o ITU definiu grupos de aplicações com requisitos semelhantes, baseado em três critérios:

- **Temporização entre origem e destino:** necessária ou não necessária;
- **Taxa de bit:** constante ou variável;
- **Modo de conexão:** orientado à conexão ou não.

Isso será estudado mais a fundo no capítulo dedicado ao ATM.