Reserva de Recursos e Qualidade de Serviço em Redes de Computadores de Débito Elevado

Elisabete Paulo, Elisabete Reis, Filipe Araújo, Joana Urbano, Nuno Pimenta,
Paulo Mendes, Gonçalo Quadros, Edmundo Monteiro

Grupo de Comunicações e Serviços Telemáticos
Departamento de Engenharia Informática
Universidade de Coimbra
Polo II - Pinhal de Marrocos
3030 Coimbra - Portugal
(proj_qos@mercurio.uc.pt)

Resumo

Este trabalho insere-se na área das redes de comunicação de alta velocidade com garantia de qualidade de serviço (QoS), sendo especialmente focados alguns aspectos importantes nesta área, concretamente, o controlo de congestão, a reserva de recursos e a especificação de requisitos de qualidade de serviço por parte das aplicações.

É descrita a criação de um ambiente experimental cujos objectivos são o desenvolvimento e a integração de mecanismos de garantia de qualidade de serviço (QoS) em protocolos de rede e de transporte, operando sobre tecnologias de elevado débito binário. Concretamente, é implementada uma pilha protocolar com capacidade de reserva em todas as suas camadas, composta pelo protocolo de transporte XTP *(Xpress Transport Protocol)*, pelo protocolo IP *(Internet Protocol)* associados ao protocolo RSVP *(Resource ReSerVation Protocol)* e implantados sobre uma sub-rede de tecnologia ATM *(Asynchronous Tranfer Mode)*.

1. Introdução

O aparecimento de tecnologias de comunicação de alta velocidade, associado ao desenvolvimento de novas aplicações com elevadas necessidades em termos de largura de banda e de atraso, tem evidenciado a necessidade crescente de garantia de qualidade de serviço fornecida às aplicações pelas redes subjacentes, bem como de protocolos adequados à comunicação em redes de elevado débito.

O conjunto de protocolos que actualmente suporta a *Internet* (pilha protocolar TCP/IP) revela-se ineficaz no transporte de tráfego proveniente de aplicações em tempo-real, sendo necessária a sua extensão, que passará pela criação de um modelo com capacidade de oferecer garantias de QoS e pelo desenvolvimento de protocolos de sinalização para reserva de recursos, como o RSVP.

Neste trabalho é descrita a criação de um ambiente experimental cujos objectivos são o desenvolvimento e a integração de mecanismos de qualidade de serviço em protocolos de rede e de transporte, operando sobre tecnologias de elevado débito binário.

Concretamente, é experimentada uma pilha protocolar com capacidade de reserva de recursos em todas as suas camadas. A pilha experimentada é composta pelo protocolo XTP (*Xpress Transport Protocol*) [XTPForum95] na Camada de Transporte e pelo protocolo IP *(Internet Protocol)* associado ao protocolo RSVP (*Resource ReSerVation Protocol*) [Braden96] na Camada de Rede. Estes protocolos são, implantados sobre uma sub-rede de tecnologia ATM *(Asynchronous Tranfer Mode)*.

A escolha desta pilha protocolar é motivada pela atenção que, a nível internacional, tem sido votada aos protocolos XTP e RSVP.

2. Ambiente experimental

O ambiente experimental construído é baseado num comutador ATM interligando vários computadores pessoais e estações de trabalho. Nos PCs foi instalado o sistema operativo Unix BSD, tendo sido necessário o desenvolvimento de um *driver* de controlador ATM para este sistema operativo.

Num sistema com funções de *router* foram desenvolvidos e instalados módulos de *controlo de admissão*, *filtragem* e *escalonamento*, para levar a cabo a integração de serviços com diferentes requisitos de QoS e o protocolo RSVP, que permitem uma reserva efectiva de recursos, mesmo quando nas camadas inferiores da pilha são usadas tecnologias que tradicionalmente não oferecem garantias de QoS, como por exemplo a *Ethernet*.

Em todas as máquinas foi implantado o protocolo de transporte XTP sobre o protocolo IP (associado ao RSVP), tendo sido desenvolvido um *módulo de mapeamento* entre os parâmetros de QoS das sessões XTP e a negociação dos fluxos RSVP.

Sobre o protocolo XTP foi implantada uma interface de desenvolvimento de aplicações *(API XTP)* com capacidade de suporte para especificação e negociação de um conjunto variado de parâmetros de qualidade de serviço.

O ambiente experimental aqui descrito é ilustrado na Figura 1. Os seus detalhes mais relevantes serão abordados nas secções seguintes.

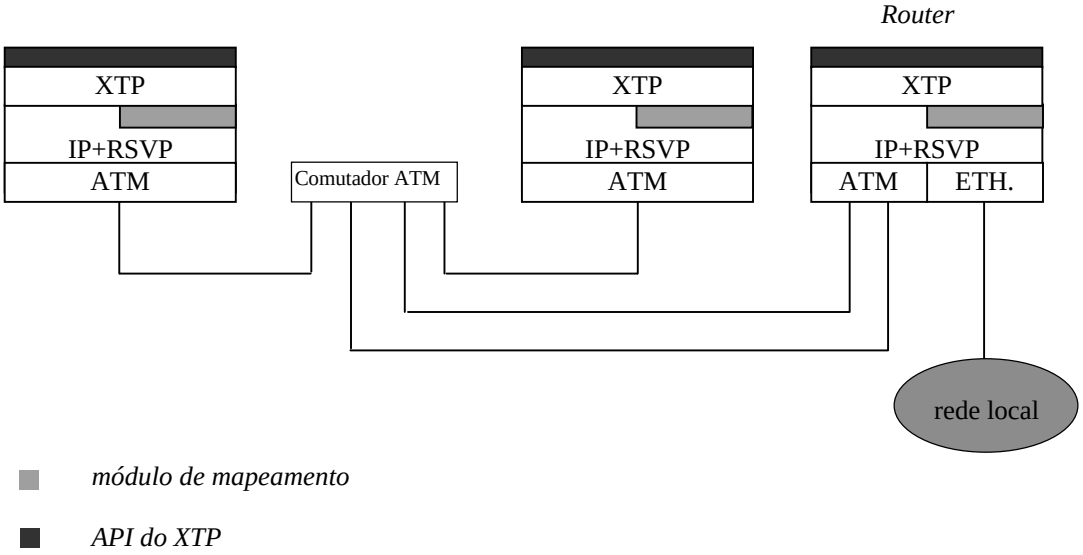

Figura 1 - Ambiente experimental

3. IP/RSVP sobre ATM

Um dos problemas levantados por esta arquitectura reside no facto de o RSVP não poder, isoladamente, por ser apenas um protocolo de sinalização, oferecer garantias de QoS às aplicações. Uma das formas de ultrapassar o problema passa, tal como foi referido, pela adição de módulos complementares ao Sistema Operativo. Outra hipótese, passa pela utilização de uma tecnologia que seja capaz de oferecer garantias de QoS nos níveis mais baixos da pilha protocolar: o ATM, por exemplo.

É certo que o ATM poderia desempenhar um papel mais importante numa pilha protocolar capaz de garantir múltiplos parâmetros de QoS às aplicações. Além disso, tudo aponta para que a tecnologia ATM venha a ter uma forte implantação num curto prazo de tempo. O problema, porém, prende-se com a base já instalada: há milhões de máquinas ligadas à rede pelos protocolos TCP/IP e parece mais razoável supor que a transição de serviços do tipo *best-effort* para serviços baseados em parâmetros de QoS seja feita por uma evolução dos protocolos TCP/IP, através da integração de novos protocolos, como o RSVP e novas tecnologias como o ATM do que esperar uma revolução em que a pilha TCP/IP dá lugar ao ATM associado a um novo protocolo de transporte e a uma nova API, ainda que esta segunda solução apresentasse possibilidades muito atraentes.

A integração do ATM na pilha TCP/IP já está feita. Há várias formas de o fazer, nomeadamente, por emulação de LANs, IP clássico e mais

recentemente NHRP (*Next Hop Resolution Protocol*). O estudo destes métodos está fora do âmbito deste documento. Veja-se, para o efeito, respectivamente [ATMForum], [Laubach93] e [Katz]. A emulação de LANs e o IP clássico sobre ATM, pelo menos, ultrapassaram já as fronteiras da teoria e são métodos normalmente incluídos pelos sistemas ATM disponíveis comercialmente.

3.1. Implantação do RSVP sobre ATM

No que toca a integrar o RSVP e o ATM na mesma pilha protocolar, as coisas não estão tão adiantadas. Pelo contrário, resta ainda muito por fazer. O trabalho que está descrito neste documento, também não constitui excepção. Com efeito, outros problemas assumiram maior prioridade, em particular o desenvolvimento de *device drivers* para placas ATM sem os quais não é possível, sequer, integrar o ATM na pilha TCP/IP. Vamos, portanto, indicar apenas alguns das questões que se levantam à operação do RSVP sobre o ATM:

- *utilização de circuitos virtuais.* No suporte do IP sobre ATM está prevista, naturalmente, apenas a utilização de serviços *best-effort*. Para alargar o suporte ao RSVP será necessário criar mecanismos que façam a correspondência entre os modelos do *Integrated Services*[1] e os modelos de QoS disponíveis no ATM, sempre que for estabelecido um circuito virtual;
- *suporte da heterogeneidade do RSVP*. Uma vez que os receptores dentro de uma mesma sessão RSVP podem solicitar diferentes parâmetros de QoS, há que encontrar um meio de permitir uma diferença efectiva nas reservas feitas pelo ATM, já que nem o UNI[2] 3.x, nem o UNI 4.0 suportam directamente receptores heterogéneos numa única ligação ponto-multiponto. Um caso particularmente importante será o de haver numa mesma sessão RSVP receptores *best-effort* em simultâneo com receptores que requeiram garantias de QoS. Por um lado, as exigências dos segundos têm que ser respeitadas, mas por outro, os primeiros não devem ter que pagar serviços que não pediram;
- *renegociação de QoS.* Enquanto no RSVP as reservas são voláteis, no ATM são estabelecidas no início de uma ligação e assumem um carácter definitivo. Alterar uma reserva significa estabelecer uma nova ligação e eliminar a antiga, em caso de sucesso;
- *suporte das características multiponto-multiponto do RSVP no ATM*. Enquanto no RSVP as sessões *multicast* têm uma natureza *connectionless* multiponto-multiponto, no ATM, as ligações *multicast* têm natureza ponto-a-multiponto.

Existem já disponíveis propostas no sentido de implementar o RSVP sobre o ATM, resolvendo alguns destes problemas. Veja-se por exemplo, *RSVP clássico sobre ATM com atalhos* [Birman96] e *RSVP sobre ATM através de RCEs (RSVP Coordination Entity)* [Onvural96].

3.2. Desenvolvimento de um driver

Um dos problemas que surgiu na constituição do ambiente experimental foi a falta de *device drivers* para placas ATM *Fore PCA-200EPC*, para o sistema operativo *FreeBSD 2.1.0*, adoptado no ambiente experimental. Com vista ao desenvolvimento do *device driver* foram estabelecidos contactos com o fabricante que, mediante a assinatura de um contrato, forneceu código fonte para o sistema operativo SunOS 4.1.x. A grande vantagem de ter este código disponível reside na possibilidade da sua reutilização já que muitos dos mecanismos internos dos sistemas operativos em causa são semelhantes.

É de referir que uma placa adaptadora ATM, capaz de operar em débitos tão elevados como 155 Mbit/s, exige, não só, um computador à altura em termos de desempenho, mas também um sistema operativo (onde se inclui o *device driver*) cuidadosamente programado. Em termos de concepção do *device driver* estas condições impõem, naturalmente, algumas exigências, nomeadamente em termos de quantidade de memória RAM necessária.

Outra condição a ter em conta é que, actualmente, a maioria dos sistemas operativos correntes, nomeadamente o *FreeBSD*, não apresenta suporte genérico para integração do ATM na pilha TCP/IP. Quer no método de emulação de LANs, quer no IP clássico sobre ATM, esse suporte tem que ser incluído no *software* fornecido pelo fabricante em soluções proprietárias. Isto tem várias consequências. A mais óbvia é que o desenvolvimento do *device driver* e módulos associados se torna mais complicado e mais moroso. Não menos importante, porém, será o facto, mais do que provável, de soluções diversas não permitirem a interligação de equipamento proveniente de fabricantes distintos, pelo menos enquanto as

[1] O *Integrated Services Working Group* é o grupo de trabalho da IETF (*Internet Engineering Task Force*) responsável pela extensão do modelo de serviço da *Internet* a um modelo de serviços integrados.

[2] *User-Network Interface*. Define a interface entre o utilizador e a rede.

normas que regem o ATM não estiverem completamente estabilizadas e adoptadas.

Num futuro mais ou menos breve, prevê-se que os sistemas operativos venham a incluir módulos que permitam fazer a emulação de LANs e/ou IP clássico sobre ATM. Quando esse dia chegar, programar um *device driver* para uma placa ATM poderá ser mais parecido com aquilo que é hoje programar um *device driver* para uma placa *Ethernet*.

Outro aspecto importante prende-se com a utilização de aplicações ATM nativas. O eventual sucesso do ATM como elemento fundamental duma pilha protocolar autónoma passará em muito pela quantidade de aplicações específicas que sobre ele venham a ser desenvolvidas e que sejam capazes de aproveitar plenamente as suas potencialidades. Daí, que outro módulo a acrescentar a um sistema operativo, seja uma API capaz de utilizar directamente o ATM e que terá portanto, de conseguir comunicar directamente com o *device driver* da placa. Mais uma vez, essa API é fornecida, actualmente, apenas pelos fabricantes, sendo o seu futuro — como parte integrante dos sistemas operativos — um pouco mais incerto.

Os pormenores relativos ao desenvolvimento do *device driver* podem ser encontrados em [Araújo96].

4. Sistema intermediário

Como foi já referido, o RSVP, por si só, é incapaz de realizar tarefas como a reserva de recursos e o controlo e gestão dos fluxos de uma rede. No entanto, o RSVP apresenta interfaces com módulos internos ao sistema operativo capazes de efectuar essas tarefas.

Uma das fases deste trabalho foi dedicada ao projecto e desenvolvimento de três módulos de gestão de tráfego: um *módulo de controlo de admissão*, um *módulo de classificação de pacotes* ou *filtragem* e um *módulo escalonador*.

As alterações efectuadas na pilha protocolar TCP/IP situaram-se na camada de rede — protocolo IP.

A construção destes módulos foi dirigida para a sua instalação num *router* de modo a este poder oferecer aos fluxos um serviço muito semelhante ao serviço *Controlled-Load* [Wroclawski96] do *Integrated Services*.

A especificação dos módulos inclui estruturas desenvolvidas pelo MIT (*Massachusetts Institute of Tecnology*)[3], tendo envolvido ainda a criação de:

- *uma tabela de interfaces*, com informação acerca das interfaces de comunicação presentes no sistema, tal como a largura de banda total, a largura de banda ocupada e informação sobre os fluxos com reservas nessas interfaces;
- *uma tabela de filtros*, com a identificação unívoca de cada fluxo ao qual foi associada a reserva de recursos e com uma estrutura utilizada para a gestão das filas de prioridades por fluxo;
- *filas de prioridades por fluxo* — os fluxos sem qualquer reserva de recursos estão associados a uma fila *best-effort*, única para todas as interfaces.

4.1. Módulo de controlo de admissão

O serviço *Controlled-Load* não requer a oferta de quaisquer garantias em relação a atrasos sofridos nas filas de saída do *router*, pelo que optámos pela construção de um módulo de controlo de admissão baseado em capacidade de largura de banda, bastante simples, cujo único elemento de decisão são os recursos disponíveis no elemento da rede no momento do pedido de reserva.

No momento da decisão de aceitação/recusa, o Controlo de Admissão possui informação acerca da interface para a qual se pretende efectuar a reserva, assim como o especificador do fluxo requerente (*flowspec*[4]).

Os recursos que o Controlo de Admissão considera são a largura de banda disponível na interface e o espaço de memória existente para a criação da fila de prioridades[5].

A reserva de largura de banda é efectuada por limiares múltiplos de uma largura de banda base, sendo efectivamente sujeito à decisão de reserva o limiar superior mais próximo do parâmetro *token_rate* (r) do *flowspec*.

O parâmetro *token_depth* do *Tspec* [6] define o tamanho da fila associada ao fluxo.

---

[3] Estruturas distribuídas no ficheiro *tc_isps.h*.

[4] Tal como em *tc_isps.h*.

[5] A capacidade computacional do sistema será um factor a considerar numa futura versão.

[6] Tal como em *tc_isps.h*.

4.2. Módulo de classificação

O módulo de classificação de pacotes entra em acção sempre que a placa notifica a recepção de um pacote pela activação de uma interrupção de *software*.

A sua acção consiste na verificação do cabeçalho de cada pacote IP que se encontra na fila de pacotes IP — *ipintrq* —, comparando a informação aí obtida (endereços origem e destino, protocolo e portos de nível protocolar superior[7]) com informação contida na tabela de filtros.

Caso haja uma correspondência de valores, conclui-se que o pacote em análise pertence a um fluxo para o qual foi feita reserva de recursos, sendo o seu endereço colocado na fila apropriada. Caso contrário, o endereço do pacote é colocado na fila *best-effort*.

4.3. Módulo de escalonamento

O Escalonador vai escolher o próximo pacote a ser processado, de entre os que já foram separados pelo Classificador.

Este módulo segue um esquema aproximado ao *round-robin* pesado: o escalonador verifica periodicamente as filas por fluxo com os endereços dos pacotes IP, que têm diferentes prioridades, tirando de cada fila o número de pacotes equivalente à sua prioridade. Uma fila terá uma prioridade tão mais elevada quanto maior for o número de pacotes retirados pelo escalonador em cada iteração do ciclo.[8]

A remoção dos pacotes propriamente ditos da fila *ipintrq* é feita de acordo com os endereços que vão sendo seleccionados das filas dos fluxos, sendo por isso uma remoção selectiva e não sequencial, como acontecia originalmente no código IP.

Embora não haja nenhum módulo exclusivo para o policiamento dos fluxos, ele é feito implicitamente através da divisão do tráfego em filas de prioridade por fluxo.

[7] Os portos de nível protocolar superior são obtidos por decomposição do pacote IP. Embora este procedimento constitua uma violação ao princípio da distinção entre camadas protocolares, ele é estritamente necessário para a distinção de fluxos de diferentes sessões RSVP.

[8] Uma nova versão deste módulo deverá considerar não apenas o número mas também o tamanho dos pacotes a retirar.

5. XTP sobre IP/RSVP

Nesta secção é feita uma descrição do método utilizado para permitir utilizar o XTP sobre IP com a possibilidade de reserva de recursos mediante a utilização do RSVP.

Como base deste trabalho foi utilizada a implementação do XTP dos *Sandia National Laboratories*, Califórnia (EUA), *SandiaXTP Release 1.3.3* [XTP] e a implementação do RSVP da ISI (*Information Sciences Institute — University of Southern California*), *RSVP Release 4.0a7* [RSVP].

A comunicação em XTP é feita pela associação de contextos[9]. Esta associação é iniciada com o envio de um pacote FIRST[10] por parte de um contexto. Os contextos que receberem este pacote fazem uma análise da especificação de tráfego e enviam ao contexto emissor um pacote TCNTL com a especificação de tráfego possível, sendo igual à do pacote FIRST caso esta seja aceite.

Depois de concluída a negociação de tráfego, os contextos estão aptos a trocar dados entre si, em ambos os sentidos.

Em qualquer momento da associação, uma aplicação pode requerer um novo contrato de tráfego que é negociado pelos contextos através da troca mútua de pacotes de controlo de tráfego (TCNTL).

Ao terminar a associação, os contextos nela envolvidos passam ao estado quieto.

5.1. Integração XTP/RSVP

Sendo o contexto XTP a entidade que suporta a especificação de tráfego da comunicação, os endereços origem e destino bem como os portos utilizados, sendo esta entidade a responsável por iniciar e terminar uma associação, negociar uma especificação de tráfego, enfim, gerir a comunicação entre utilizadores, não poderia deixar de ser o contexto XTP o responsável por solicitar a reserva de recursos à rede, para o cumprimento do contrato de tráfego estabelecido.

Assim, toda a integração do código RSVP foi efectuada nas funções específicas do contexto XTP - classe *XTPcontext*. Para além das funções implementadas nesta classe, foram criadas novas funções necessárias à integração deste código.

[9] A cada processo, está associado um contexto que contém toda a informação útil relativa à sua comunicação.

[10] O XTP, ao longo de uma ligação, utiliza vários tipos de pacotes consoante a função que pretende desempenhar.

Na Figura 2 está esquematizada a integração destes dois protocolos.

Sendo a comunicação em XTP possível em ambos os sentidos (*full-duplex*), para que seja possível estabelecer reservas de recursos também nos dois sentidos, terão de ser criadas duas sessões RSVP.

Depois de registadas ambas as sessões e feitos os respectivos pedidos de reserva de recursos à rede, ambos os contextos estão aptos a enviar e a receber dados.

Para fazer o pedido de reserva de recursos à rede, é necessário fazer um mapeamento dos parâmetros do especificador de tráfego da associação XTP para as respectivas estruturas de QoS do RSVP. São feitas equivalências, assumindo determinados pressupostos e seguindo determinadas regras, de parâmetros do especificador de tráfego de cada contexto XTP, como a quantidade máxima de dados de um pacote XTP, tamanho máximo dos *bursts* de entrada e saída, débito médio de entrada e saída, etc, para parâmetros do RSVP como o tamanho máximo dos pacotes IP, débito médio e profundidade do *Token Bucket*, etc. Este mapeamento é efectuado quer para a sessão de envio quer para a sessão de recepção.

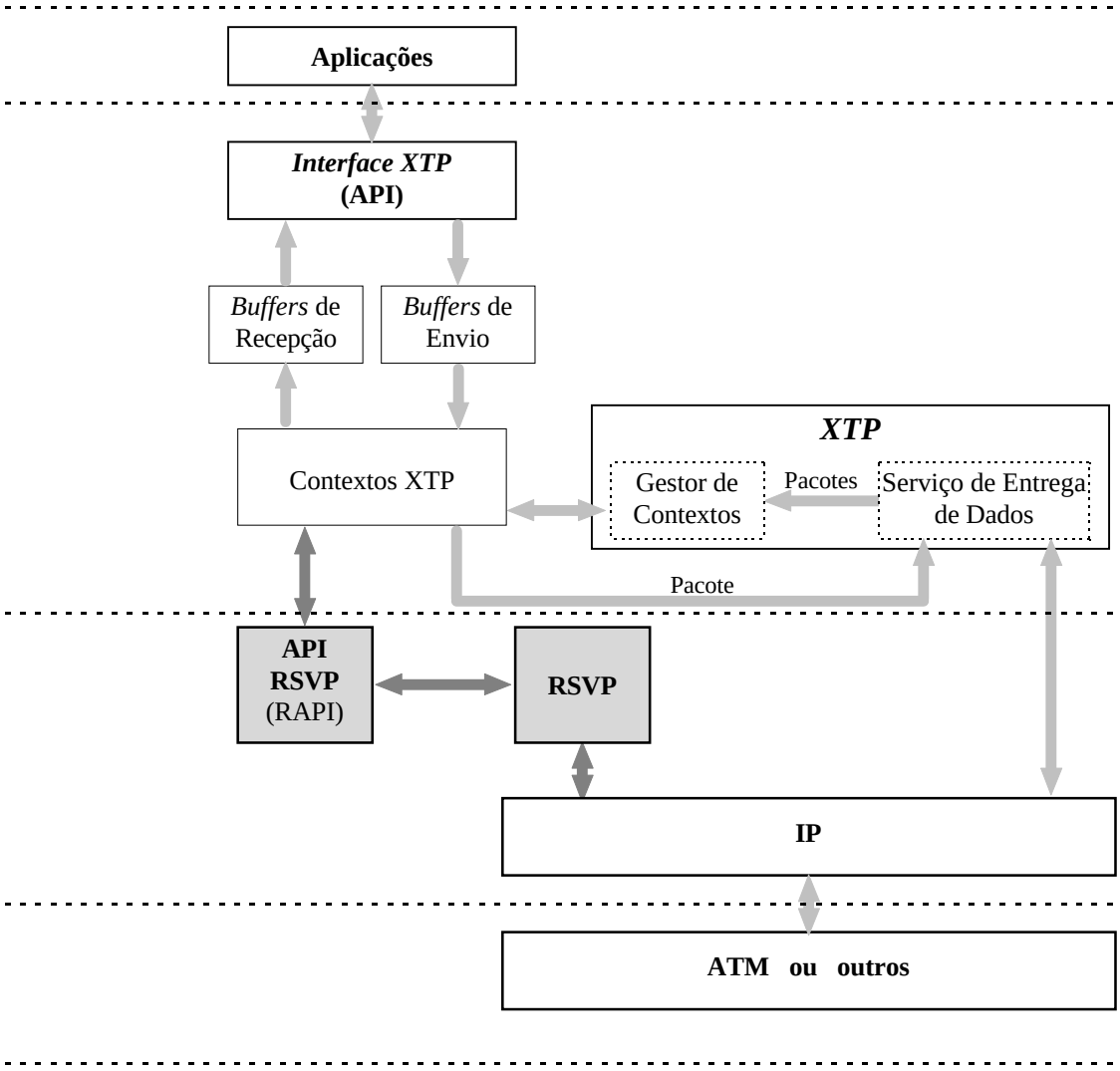

Figura 2 - Integração do XTP/RSVP

Quando um contexto passa ao estado quieto (chamada da função *XTPcontext:: go_quiescent()*), ambas as sessões são fechadas, mediante a chamada da função *rapi_release()*, eliminando assim as reservas efectuadas.

Feita a integração do código destes dois protocolos, foram concebidos dois pequenos

programas de teste (cliente/servidor), utilizando as potencialidades do novo XTP sobre IP/RSVP.

Estes programas de teste permitem efectuar a troca de dados entre si, dando ao utilizador, em qualquer momento da associação, a possibilidade de propor novas especificações de tráfego que serão renegociadas e, consequentemente, efectuar novos pedidos de reserva de recursos através do RSVP.

5.2. API para XTP

Para uma melhor utilização do XTP por parte das aplicações, foi especificada uma API (*Application Programming Interface*) que consiste num conjunto de rotinas que permitem ao utilizador interactuar directamente com um protocolo de transporte (XTP) através das aplicações.

Esta interface tem como principal objectivo permitir ao utilizador requerer uma QoS de acordo com as necessidades da aplicação em causa.

Esta API teve como base o manuseamento de parâmetros relacionados com os mecanismos disponíveis no XTP, de modo a obter uma determinada QoS.

Neste âmbito, foi adicionado um conjunto de parâmetros a ser tratado quer pelo XTP quer pelas camadas subjacentes.

A API é definida por rotinas que permitem a criação de contextos, o estabelecimento de ligações, leitura e escrita de dados e terminação das ligações.

Para uma especificação mais detalhada veja-se [Paulo96].

6. Evolução do ambiente

As aplicações de comunicação têm diferentes necessidades quanto ao desempenho (ou qualidade de serviço) que lhes é fornecido pelos sistemas de comunicação que as suportam. Designadamente, quanto à largura de banda disponível, tempo de transporte dos dados, variação desse tempo, fiabilidade, etc.. Os sistemas de comunicação têm de ser capazes de transportar convenientemente e de forma integrada o tráfego gerado. Esse tráfego pode ser de diferente natureza (por exemplo, vídeo, áudio, ou dados de computador).

Nesta secção vão ser analisados alguns aspectos de evolução do ambiente experimental desenvolvido.

Em primeiro lugar é abordada a caracterização das aplicações de tempo-real em termos das suas necessidades de QoS.

Em segundo lugar, partindo do conhecimento destas necessidades é identificado um conjunto de caracteristicas necessárias a uma API da camada de transporte para suporte de aplicações de tempo-real no ambiente experimental desenvolvido.

Finalmente, são focados alguns aspectos relacionados com a necessidade de funções de monitorização da qualidade de serviço, no funcionamento dos módulos protocolares do sistema de comunicação.

Os aspectos de evolução aqui abordados correspondem a trabalhos em curso no Grupo de Comunicações e Serviços Telemáticos, no âmbito do estudo das arquitecturas protocolares baseadas no paradigma da *reserva de recursos*.

6.1. Caracterização das aplicações

Segundo o modelo *Integrated Services (IS)* proposto como percurso de evolução para a Internet [Braden94] as aplicações, reflectindo a natureza do tráfego que produzem, podem ser divididas em:

- *aplicações de tempo-real* — exigentes no que diz respeito ao tempo de transporte dos seus dados, ou
- *aplicações elásticas* — sem exigências no que diz respeito ao tempo de transporte dos seus dados.

As primeiras, por sua vez, consoante sejam mais ou menos rigorosas nas suas exigências sobre o desempenho do sistema de comunicação, podem ser classificadas em: *adaptativas e tolerantes*; ou *intolerantes*. Uma boa parte das aplicações tempo-real são adaptativas e tolerantes, o que possibilita uma utilização mais eficiente dos recursos de comunicação.

Independentemente do modelo utilizado para fornecimento de qualidade de serviço às aplicações (para além do modelo *IS* existem outros [Alfano96], [Nahrstedt95], [Campbell94], [Banerjea94]), são os utilizadores do sistema de comunicação que terão de especificar a qualidade de serviço desejada. Assim, ao nível da interface das aplicações com o sistema de comunicação, têm de ser especificados os requisitos de QoS. Essa é uma importante função da responsabilidade da API utilizada.

6.2. Evolução da API de transporte

As APIs são uma componente fundamental para os sistemas de comunicação aptos a fornecer qualidade de serviço às aplicações. De facto, é

através das APIs que estas estabelecem as condições para que os fluxos de dados que geram obtenham o desempenho que necessitam. Designadamente, é através das APIs que é requisitada a qualidade de serviço pretendida, as características do tráfego e ainda o comportamento relacionado com o fornecimento da QoS desejada.

Idealmente, uma API deve ser muito flexível, por exemplo, deve permitir:

- definir um intervalo de variação aceitável para cada característica de QoS, e eventualmente valores objectivos; é ainda interessante possibilitar não só a especificação de valores absolutos mas também de valores estatísticos;
- definir a natureza das especificações, ou ainda, a forma como elas devem ser interpretadas (o tipo de serviço): como seja, garantido, previsível, ou melhor esforço;
- definir limiares e acções a executar em caso de ultrapassagem desses limiares;
- antecipar o estabelecimento de ligações.

Ainda idealmente, todas as especificações devem ser ortogonais a cada uma das características de QoS; deve ser possível a negociação entre as aplicações e o sistema de comunicação da qualidade de serviço de facto estabelecida; e esse estabelecimento não deve ser estático mas sim dinâmico, ou seja, deve reflectir a evolução das condições operacionais dos sistemas de comunicação e das necessidades de quem o utiliza.

No âmbito deste trabalho foi desenvolvida uma API simples cujo único objectivo foi criar condições para testar o ambiente protocolar implementado (referida na secção 5). No seguimento deste trabalho tem vindo a ser trabalhada um proposta de API para suporte de serviços de comunicação com necessidades de tempo-real e qualidade de serviço.

A API em desenvolvimento permite a utilização, de forma transparente, de protocolos de transporte em tempo-real, de reserva de recursos e de monitorização da qualidade de serviço requerida, realizando a ligação entre as funções postas à disposição da aplicações e as funções inerentes a estes protocolos.

Como os parâmetros de qualidade definidos para as aplicações e que os utilizadores podem ajustar, com base em perfis de QoS, não são os mesmos processados pelos protocolos, ao nível do sistema de comunicações, a API realiza a conversão, bidireccional, entre estes dois níveis de parâmetros. Apesar deste mapeamento ser flexível, no presente trabalho, ele está a ser concebido tendo em consideração a sua utilização no âmbito do *Integrated Services* [Mendes97].

Especificada a QoS torna-se necessário disponibilizar os meios para que ela possa de facto ser fornecida. Tal como referido na secção 4, são necessários mecanismos de controlo de admissão, através dos quais o sistema de comunicação avalia a possibilidade de fornecer o que lhe está a ser requisitado.

Em caso afirmativo, são essenciais mecanismos que possibilitem a reserva dos recursos necessários para tal (protocolos como o RSVP são utilizados neste contexto). São ainda muito importantes os mecanismos de gestão de recursos que permitam maximizar a eficiência com que o sistema de comunicação é utilizado[11].

Outro aspecto fundamental para o bom funcionamento dos sistema de comunicação capazes de fornecer qualidade de serviço, é a sua monitorização. Este aspecto será abordado na secção seguinte.

6.3. Métrica de monitorização de QoS

O funcionamento dos sistemas de comunicação capazes de disponibilizarem qualidade de serviço e a manutenção de bons níveis de eficiência na utilização dos seus recursos, depende substancialmente da existência de informação acerca do seu comportamento. Tal é assumido pelos diversos modelos propostos para dotar os sistemas de comunicação de capacidades no domínio da qualidade de serviço. Apesar disso, existe ainda pouco trabalho nesta área específica de investigação.

O presente trabalho pretende também promover as condições necessárias para a implementação e teste de uma arquitectura de monitorização de QoS, que tem vindo a ser desenvolvida em paralelo e se espera poder vir a ser testada a curto prazo no modelo do *Integrated Services*.

O conceito nuclear desta proposta para medir qualidade de serviço é o da *métrica de QoS* concebida para esse efeito [Quadros97]. Esta métrica, tem como objectivo possibilitar a

[11] Note-se que todo este processo tem de ser concretizado nos diferentes níveis tecnológicos que tipicamente constituiem um sistema de comunicação; isto implica a existência de mecanismos de translação, ou mapeamento, de qualidade de serviço entre as diversas tecnologias. Um exemplo disso é tratado no capítulo 5 deste documento – mapeamento de QoS entre XTP e RSVP.

quantificação, de forma integrada e normalizada[12], do desempenho do sistema de comunicação.

Pretende-se, naturalmente, que o desempenho seja avaliado nas suas diversas facetas (como sejam, largura de banda, atraso no transporte dos dados, *jitter*, fiabilidade) e que tal avaliação contemple diferentes pontos de vista do funcionamento do sistema de comunicação (o que implica, entre outras coisas, a necessidade de medir em diferentes escalas de tempo).

A ideia, é que as medições obtidas utilizando a métrica referida, facilitem e optimizem a gestão, controlo e manutenção do sistema de comunicação, fornecendo indicações que possibilitem a tomada das melhores decisões por parte de quem tem a responsabilidade de gerir, e a optimização do funcionamento dos mecanismos do sistema de comunicação relacionados com a disponibilização da qualidade de serviço.

7. Conclusão

Neste trabalho é descrita a criação de um ambiente experimental com capacidade de reserva em todas as suas camadas.

O ambiente implementado é composto pelo protocolo de transporte XTP *(Xpress Transport Protocol)* na Camada de Transporte e pelo protocolo RSVP *(Resource Reservation Protocol)* associado ao protocolo IP na camada de Rede e implantados sobre uma sub-rede de tecnologia ATM *(Assyncronous Tranfer Mode).*

O ambiente experimental descrito neste trabalho encontra-se actualmente na fase de integração e teste dos módulos desenvolvidos, sendo esperados para breve os primeiros resultados relativos ao seu desempenho quando submetido a tráfego com características multimédia, gerado por um conjunto de aplicações de teste entretanto seleccionadas.

Em paralelo com as etapas de integração e teste dos módulos desenvolvidos, foi iniciada a caracterização das aplicações de tempo-real em termos das suas necessidades de QoS com vista à identificação um conjunto de características necessárias a uma API genérica da Camada de Transporte para suporte de aplicações de tempo-real no ambiente experimental desenvolvido.

O ambiente experimental desenvolvido pretende também promover as condições necessárias para a implementação e teste de uma arquitectura de monitorização de QoS, que tem vindo a ser desenvolvida em paralelo e se espera poder vir a ser testada a curto prazo no modelo do *Integrated Services*.

[12] Ou seja, de tal forma que as medidas obtidas possam ser comparadas.

Agradecimentos

O trabalho apresentado neste documento foi parcialmente financiado pela Junta Nacional de Investigação Científica e Tecnológica (Projecto QoS — referência PBIC/TIT/2460/95).

Referências

[Alfano96] M. Alfano, N. Radouniklis, "A Cooperative Multimedia Environment with QoS Control: Architectural and Imlpementation Issues", ICSI, Berkeley, Sept. 1996

[Araújo96] F. Araújo, J. Urbano, "QoS — Qualidade de Serviço em Redes de Alto Débito — Parte A", Relatório da Cadeira de Projecto e Dissertação, DEI, Universidade de Coimbra, Novembro de 1996.

[ATMForum] The ATM Forum, "Lan Emulation Over ATM Specification", Version 1.0.

[Banerjea94] A. Banerjea, D. Ferrari, B. Mah, M. Moran, D. Verma, H.Zhang, "The Tenet Real-Time Protocol Suite: Design, Implementation, and Experiences", Technical Report, TR-94-059, ICSI, Berkeley, CA, November, 1994.

[Birman96] A. Birman, R. Guerin, D. Kandlur, "Support for RSVP-based Service over an ATM Network", Internet Draft, <draft-birman-ipatm-rsvpatm-00.txt>, February, 1996

[Braden94] R. Braden, D. Clark, S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC1633, June, 1994.

[Braden96] R. Braden, L. Zhang, D. Estrin, S. Herzog, S. Jamin, "Resource ReSerVation Protocol (RSVP) — Version 1 Functional Specification", Internet Draft, <draft-ietf-rsvp-spec-14.txt>, November, 1996.

[Campbell94] A. Campbell, G. Coulson, D. Hutchison, "A Quality Of Service Architecture", Lancaster University, April, 1994.

[Katz96] D. Katz, D. Piscitello, B. Cole, J. Luciani, "NBMA Next Hop Resolution Protocol (NHRP)", Internet Draft, 1996.

[Laubach93] Laubach, M., "Classical IP and ARP over ATM", RFC 1577, December, 1993.

[Mendes97] P. Mendes, "Estudo do Suporte dos Requisitos de Qualidade de Serviço das Aplicações de Trabalho Cooperativo pelos serviços integrados", *Draft* de tese de mestrado, 1997.

[Nahrstedt95] K. Nahrstedt, "An Architecture for End-To-End Quality of Service Provision and its Experimental Validation", Univ. Pennsylvannia, 1995.

[Onvural96] R. Onvural, S. Keshav, "A Framework for Supporting RSVP Flows Over ATM Networks", Internet Draft, <draft-onvural-srinivasan-rsvp-atm-00.txt>, February, 1996.

[Paulo96] E. Paulo, E. Reis, N. Pimenta, "QoS — Qualidade de Serviço em Redes de Alto Débito — Parte B", Relatório da Cadeira de Projecto e Dissertação, DEI, Universidade de Coimbra, Novembro, 1996.

[Quadros97] G. Quadros, "Proposta de Métrica para a avaliação de Qualidade de Servigo em Sistemas de Comunicação", *Draft* de Relatorio Técnico, Departamento de Engenharia Informática, Universidade de Coimbra, Janeiro, 1997.

[RSVP] URL: http://www.isi.edu/ div7/ rsvp/ overview.html

[Wroclawski96] J. Wroclawski, “Specification of the Controlled-Load Network Element Service”, INTERNET-DRAFT, <draft-ietf-intserv-ctrl-load-svc-03.txt>, August, 1996.

[XTP] URL:http://www.ca.sandia.gov/xtp

[XTPForum95] XTP Forum, "Xpress Transport Protocol Specification - XTP Revision 4.0", March, 1995.