MCF: incorporando escalabilidade, mobilidade e novas estratégias de implementação de alto desempenho*

Marcelo Dias Nunes
bill@gta.ufrj.br

Otto Carlos M. B. Duarte
otto@gta.ufrj.br

Grupo de Teleinformática e Automação – GTA
COPPE/EE – Programa de Engenharia Elétrica
Universidade Federal do Rio de Janeiro
Caixa Postal 68504 – CEP 21945-970 – Rio de Janeiro – RJ – Brasil

Resumo

Os serviços de IP Multiponto e MBone na Internet possibilitaram o surgimento das mais diversas aplicações multidestinatárias. Enquanto seu padrão de utilização não for bem definido, somente os protocolos de alto desempenho que suportarem grupos dos mais variados tamanhos serão bem sucedidos. Para que isso seja possível, é preciso levar em consideração os problemas de implementação eficiente de protocolos, implosão de reconhecimentos e escalabilidade. Além disso, torna-se cada vez mais importante garantir conectividade contínua para estações móveis. Portanto, este artigo discute duas estratégias de implementação, ALF e ILP, as principais classes de protocolos multidestinatários, questões de mobilidade e, finalmente, extensões e modificações ao protocolo MCF são propostas, visando melhorar seu desempenho e escalabilidade.

Abstract

IP Multicast and MBone services in the Internet have made possible the outcome of a wide range of multicast applications. While their utilization pattern is not well defined, only those high performance protocols that support large groups will succeed. The problems of efficient protocol implementation, acknowledgement implosion and scalability must be thoroughly analyzed. Besides, there has been a strong trend towards guaranteeing continuous connectivity for mobile stations. Thus, this paper discusses two implementation strategies, ALF and ILP, the main multicast protocol classes, mobility issues and, finally, it proposes extensions and modifications to the MCF protocol, in order to improve its performance and scalability.

---

* Este trabalho foi realizado com recursos da UFRJ, FUJB, PROTEM-CC, CNPq, FAPERJ, CAPES e COFECUB.

1 Introdução

O MCF (Protocolo Multidestinatário Configurável com Gerenciamento de Acesso por Fichas) [9] é um protocolo multiponto-a-multiponto projetado para redes de alta velocidade. O protocolo MCF foi implementado primeiramente em microcomputadores PC-compatíveis, sendo posteriormente portado para plataformas Sun SPARC, sob sistema operacional Solaris 2.x. Para atender ao maior número possível de aplicações, o MCF é completamente configurável. *Checksum*, controle de fluxo e mecanismos de recuperação de erros são opcionais. Neste artigo, o trabalho está direcionado no sentido de melhorar o desempenho do MCF e transformá-lo em um protocolo escalável e robusto, visando redes de longa distância. Portanto, três assuntos principais serão abordados:

- novas estratégias de implementação: **ALF** (*Application Level Framing*) e **ILP** (*Integrated Layer Processing*);
- confiabilidade e escalabilidade em conexões multidestinatárias. Nesse sentido, o problema da implosão de reconhecimentos é atacado;
- mobilidade. Cada vez mais, torna-se importante garantir conectividade para estações móveis, permitindo difusão eficiente de informações nesse tipo de rede.

Este artigo está organizado da seguinte forma: na seção 2, as estratégias de ALF e ILP são estudadas mais detalhadamente. Na seção 3, são analisadas as principais classes de protocolos multidestinatários confiáveis, dando ênfase à escalabilidade. A seção 4 aborda algumas questões com respeito à mobilidade em conexões multidestinatárias. Na seção 5, uma breve descrição do protocolo MCF é apresentada. Na seção 6, são apresentadas as principais extensões ao MCF e, finalmente, a seção 7 apresenta as conclusões dessa etapa do trabalho.

2 Novas técnicas de implementação de protocolos

Implementações eficientes de protocolos devem evitar cópias de memória, pois o acesso à memória é lento. Se a cópia de dados é inevitável, uma solução possível é a utilização de memórias *cache*, bem mais rápidas, porém mais caras. Para tentar contornar esse gargalo, duas estratégias foram propostas: *Application Level Framing* (ALF) e *Integrated Layer Processing* (ILP) [1][4].

A principal idéia por trás do ALF/ILP é que a integração de todas as funções realizadas por diferentes camadas em uma única camada de protocolo resultará em um aumento significativo de desempenho. Uma distinção importante é a separação das funções de transferência de dados em dois grupos: aquelas que realmente lêem ou modificam os dados, chamadas funções de **manipulação de dados**, e aquelas que regulam a transferência, chamadas funções de **controle de transferência**. As manipulações de dados são muito mais intensas em termos de processamento, uma vez que literalmente “tocam” os octetos, possivelmente várias vezes.

2.1 ALF

Se a aplicação tiver a opção de lidar com dados perdidos, ao invés de deixar a cargo da camada de transporte (por retransmissão, por exemplo), as perdas precisam ser expressas em termos significativos. A aplicação deve segmentar os dados de acordo com suas características e as camadas inferiores devem preservar esse encapsulamento à medida que processam os dados. Nessa proposta, os segmentos são chamados de **Unidades de Dados da Aplicação** (ADUs), que substituem os pacotes como unidade de manipulação. Este é o princípio básico do ALF: cada ADU pode ser processada fora de ordem com respeito às demais ADUs. A ADU também passa a ser a unidade de recuperação de erros.

2.2 ILP

Implementações ingênuas de um sistema de protocolos em camadas realizam um processamento seqüencial de cada unidade de informação. Normalmente, é possível arrumar as operações em uma ordem mais eficiente, que atinge o mesmo resultado final. Assim, introduz-se o princípio de ILP, no qual um protocolo é estruturado de forma a permitir ao implementador a opção de realizar todos os estágios de manipulação em um ou dois *loops* de processamento integrados, ao invés de realizá-los seqüencialmente. Infelizmente, os protocolos tradicionais normalmente impõe limitações de precedência ou seqüência que impossibilitam tal otimização. O principal argumento em favor do ILP é que um *loop* de processamento integrado é mais eficiente que várias etapas diferentes que lêem os dados da memória, processam-nos e os escrevem de volta.

3 Escalabilidade em comunicações multidestinatárias confiáveis

Principalmente na Internet, é necessário que um protocolo multidestinatário seja **escalável**, ou seja, ele deve comportar grupos dos mais variados tamanhos. Visando a melhor solução em termos de escalabilidade, esta seção descreve as quatro principais estratégias de multiponto confiável.

3.1 Protocolos controlados pelo transmissor

Um protocolo multidestinatário confiável controlado pelo transmissor [10][7] é aquele que exige que o transmissor receba reconhecimentos (daqui em diante referenciados pelo seu acrônimo ACK, de *acknowledgement*) de **todos** os receptores, antes que lhe seja permitido liberar da memória os dados associados aos ACKs. Dessa maneira, o transmissor precisa conhecer o conjunto de receptores. Essa estratégia sofre claramente do problema de implosão de ACKs.

3.2 Protocolos controlados pelo receptor

Nesta categoria de protocolos [10][7], os receptores enviam reconhecimentos negativos (NAKs) ao transmissor quando uma retransmissão é necessária, devido a erros, falhas na seqüência ou estouro de temporizadores. Se vários receptores detectarem erros de transmissão, pode ocorrer, à semelhança dos protocolos controlados pelo transmissor, um problema de implosão de NAKs. Surgiram então os chamados **protocolos controlados pelo receptor com supressão de NAKs**. Essa categoria de protocolos utiliza dois algoritmos, *damping* e *slotting*.

Damping

Uma vez que os receptores difundem seus NAKs para o grupo, e não especificamente para o transmissor, todos os membros podem observar os NAKs enviados por seus pares. Na ocorrência de erros, receptores que detectassem erros similares enviariam NAKs contendo informações similares. O que o algoritmo de *damping* faz é impedir que um receptor envie seu NAK se alguma duplicata for detectada na rede. Com este esquema, espera-se que, teoricamente, apenas **um** NAK seja enviado de volta ao transmissor no caso de uma perda, para todo o conjunto de receptores. Deve ser utilizado em conjunto com a técnica de *slotting*.

Slotting

O algoritmo de *slotting* impõe um atraso aleatório antes da transmissão dos NAKs. Assim, o envio de NAKs é distribuído no tempo e não mais concentrado, resolvendo o problema da implosão de NAKs. A distribuição dos NAKs no tempo aumenta a possibilidade do exercício da técnica de *damping*.

3.3 Protocolos em árvore

Os protocolos baseados em árvore [7] dividem o conjunto de receptores em grupos, chamados grupos locais [6]. Cada grupo consiste em não mais do que N filhos e um líder. Filhos também podem ser líderes de um grupo. ACKs, nos protocolos em árvore, são conhecidos como reconhecimentos hierárquicos (HACKs).

Uma árvore N-ária é composta de três tipos de nós: fontes, folhas e *hops*. A **fonte** é a originadora de um novo pacote e tem no máximo N filhos dos quais deve processar HACKs e para os quais deve enviar retransmissões. **Folhas** estão na base da árvore e não têm filhos. ***Hops*** são líderes de grupos entre a fonte e as folhas. Eles enviam HACKs aos seus próprios líderes de grupo, um nível acima na árvore, e coletam HACKs de seus filhos, retransmitindo quando necessário.

Fonte e *hops* não liberam os dados da memória até que todos os seus filhos tenham reconhecido corretamente a transmissão.

Existe ainda uma segunda versão de protocolo baseado em árvore, que utiliza supressão de NAKs (no caso, HACKs) e realiza varredura nos grupos locais.

O tipo de rede que melhor se adapta a essa categoria de protocolos são as redes em malha, como redes ATM. Sua implementação sobre redes em barramanto é visivelmente impraticável.

3.4 Protocolos em anel

Protocolos em anel para multiponto confiável foram desenvolvidos originalmente para oferecer suporte a aplicações que exigem **ordenação** das transmissões em todos os receptores [7]. A idéia básica é ter um receptor, designado *token-site*, responsável pela difusão de ACKs para o grupo. Se não receber um ACK do *token-site* dentro de um certo intervalo de tempo, o transmissor retransmite o pacote. Caso contrário, poderá dealocá-lo de sua memória. O ACK enviado pelo *token-site* serve basicamente a três propósitos:

- atribuir um número de seqüência **global** ao pacote reconhecido, garantindo sua ordenação em todos os receptores;
- possibilitar aos demais receptores a detecção de perdas de pacotes. Os receptores enviam NAKs ao *token-site*, que é responsável pela retransmissão seletiva de pacotes perdidos. Por isso, o *token-site* somente poderá liberar os dados da memória quando a ficha for passada à frente;
- passar a ficha ao próximo membro do anel de receptores, caso este esteja apto a recebê-la. Para isso, o candidato a novo *token-site* deverá ter recebido corretamente todos os pacotes que o anterior recebeu.

4 Multiponto e mobilidade

Cada vez mais, com o advento dos computadores portáteis, tem se tornado fundamental a garantia de conectividade contínua a estações móveis, não obstante alterações de sua localização. O objetivo do projetista é obter transparência nas comunicações à medida que as estações se movem, sem interrupções, preservando o roteamento e o endereçamento. O termo **móvel** significa “capaz de mover-se sem perder suas conexões de rede”. Uma estação com essa característica é chamada **estação móvel** (*mobile host* – MH). As estações que se comunicam diretamente com os MHs são chamadas **estações de suporte móveis** (*mobile support station* – MSS). Uma **célula** é uma área lógica ou geográfica coberta por um MSS. A qualquer instante de tempo, um MH pertence a apenas uma célula.

Normalmente, uma rede móvel consiste em uma rede estática, ou fixa, que conecta os MSSs e demais estações fixas, e uma rede sem-fio que conecta os MSSs aos MHs, permitindo comunicação direta entre um MSS e os MHs presentes em sua célula. Um MH está associado a um endereço fixo, independentemente de sua localização na rede.

O problema da confiabilidade na difusão de dados multiponto em redes móveis pode ser resolvido garantindo-se que uma mensagem seja entregue uma – e apenas uma – vez para estações móveis. Se maiores cuidados não forem tomados, estações podem receber a mesma mensagem duas vezes ou nem chegar a recebê-la.

5 O protocolo MCF

O protocolo MCF (**M**ultidestinatário, **C**onfigurável e baseado em **F**ichas) [9] é um protocolo multiponto-a-multiponto desenvolvido pelo Grupo de Teleinformática e Automação (GTA) da COPPE-UFRJ, nos moldes de uma arquitetura de implementação especializada para otimizar o gerenciamento de memória (alocação/dealocação, manipulação de ponteiros) e o escalonamento de chamadas de procedimentos [8]. Implementado em linguagem C, sua primeira versão rodava sob sistema operacional DOS, em rede local. Devido às conhecidas limitações desse sistema operacional, o MCF não tardou a ser portado para UNIX, rodando especificamente sob sistemas operacionais Solaris 2.x, em estações de trabalho Sun SPARC. O MCF acessa os serviços oferecidos pelo UDP através da biblioteca *sockets* do sistema operacional.

Visando atender à maior variedade possível de aplicações, o protocolo MCF é completamente **configurável**, oferecendo como opcionais mecanismos que na maioria das vezes são impostos ao usuário. O cálculo de *checksum* pode ser desabilitado. Os mecanismos de controle de fluxo (janela e taxa) e recuperação de erros (*go-back-N* e retransmissão seletiva) são opcionais.

O protocolo MCF é um protocolo multiponto-a-multiponto que utiliza fichas, cuja concessão se concentra em um **servidor de fichas**, para implementar essa característica. A interface com o MCF se dá através de primitivas de serviço. As tabelas 1 e 2 resumem essas primitivas, e também as mensagens (PDUs) trocadas entre entidades do MCF.

O mecanismo de gerenciamento de fichas original do MCF é simples. Primeiramente, um servidor de fichas deve se registrar junto a um roteador multiponto e obter um endereço IP classe D, utilizando o protocolo IGMP. Quando uma aplicação deseja participar de um grupo, ela invoca uma primitiva T-MULTIPOINT.Request, desencadeando uma série de eventos. Em primeiro lugar, o IGMP contata o roteador multiponto com o nome do grupo, para obter o endereço IP correspondente. O candidato a novo membro transmite então uma PDU JOIN com endereço de grupo. Apenas o servidor de fichas responderá ao pedido, com uma PDU ACCEPT, que contém seu endereço. O novo membro pode então configurar seu dispositivo de rede local para receber os dados destinados àquele grupo.

Qualquer estação que deseje transmitir dados para o grupo deve enviar primeiramente ao servidor (ponto-a-ponto) um pedido de ficha. Se tiver alguma ficha disponível, o servidor a concede e dispara um temporizador, o TTIMER (de *token timer*), que representa o tempo máximo com o qual a estação pode permanecer com a ficha. Caso não disponha de nenhuma ficha, o endereço da estação é armazenado em uma fila FIFO interna. Uma vez em poder da ficha, a estação pode iniciar a transmissão de dados. Ao final da transmissão, a estação devolve a ficha ao servidor, que interrompe o TTIMER associado àquela ficha e a concede automaticamente ao próximo endereço da fila de pedidos (caso haja algum). Entretanto, se TTIMER estourar e a ficha não tiver sido devolvida, o servidor dispara um novo temporizador, o WTIMER (de *wait timer*), e envia um pedido de devolução **imediata**. Esse temporizador, de duração menor que TTIMER, é reiniciado três vezes até a ficha ser considerada perdida pelo servidor (nesse caso, um contador local é incrementado). Caso contrário, WTIMER é interrompido.

6 Extensões ao protocolo MCF

Nesta seção, procurar-se-á aplicar os conceitos descritos nas seções anteriores ao protocolo MCF, no sentido de torná-lo escalável e melhorar seu desempenho. Assim, algumas propostas de extensões são apresentadas.

6.1 Aplicando os conceitos de ALF e ILP

Se todos os opcionais do MCF forem habilitados (*checksum*, controle de fluxo e controle de erros), o MCF se torna um protocolo relativamente “pesado”. O princípio básico do ALF dita que a camada aplicação, se possível a única camada de protocolo existente, realizaria controle de fluxo e retransmissões de suas ADUs. Nesse caso, evidentemente, o MCF perderia sua utilidade no sistema. Seu processamento deveria, nesse sentido, ser incorporado ao código da aplicação.

A aplicação do conceito de ILP é mais complexa. Tradicionalmente, as funções de manipulação de dados de maior *overhead* são as funções realizadas pela camada de apresentação. Na maioria dos trabalhos publicados, como por exemplo em [5] e [3], aplica-se ILP a implementações que realizam intensivo processamento de apresentação, como transformações de formatos de vídeo, no primeiro caso, encriptação e *marshalling* no segundo. A única função de manipulação de dados identificada no processamento do MCF é o cálculo de *checksum*. As demais funções realizadas pelo MCF são funções de controle: controle de fluxo e recuperação de erros por retransmissão. Não existe, portanto, um número suficiente de funções de manipulação que justifiquem a integração em um *loop* único de processamento.

Outro ponto importante atacado pelo ILP é a cópia de dados. Em [3], a implementação ILP sobre TCP em nível de usuário realiza duas cópias de dados: da aplicação para o *buffer* do TCP e do *buffer* para o *kernel* do sistema. No MCF, garantindo-se que não haja segmentações, realiza-se apenas duas cópias de dados, sem implementação ILP: uma cópia para o *buffer* do UDP e uma cópia desse *buffer* para o *kernel* do sistema. Na recepção, haveria também duas cópias: uma do *kernel* para o *buffer* do UDP e daí para a área de dados do MCF (a implementação ILP em [3] também realiza duas cópias). Mas isso somente é possível porque a arquitetura de implementação do GTA é otimizada exatamente para evitar cópias. Em um sistema em camadas, os dados são alocados apenas uma vez, (na camada mais alta ou mais baixa) sendo que de uma camada para outra apenas **ponteiros** são transferidos.

Ao invés de se tentar incorporar conceitos de ILP ao protocolo MCF, outras estratégias podem ser adotadas no sentido de melhorar seu desempenho. Uma delas consiste em se evitar chamadas de rotinas. Uma solução mais eficiente consiste em se embutir macros no código do protocolo (*macro inlining*), o que não aumenta significativamente o tamanho do código executável.

Outra solução, que atualmente já se encontra em andamento por alguns membros do GTA, consiste na implementação de protocolos em *kernel* do sistema operacional. Essa implementação utilizaria a interface STREAMS do sistema; dessa forma, o MCF acessaria diretamente o dispositivo de rede. Apenas uma cópia seria realizada: do usuário para o *kernel* , na transmissão, e vice-versa, na recepção.

6.2 Incorporando escalabilidade

O conceito de servidor único e centralizado de fichas pode ser criticado por não ser dinâmico, pois o servidor é sujeito a falhas, prejudicando assim a escalabilidade do MCF. Deve-se ter em mente que o MCF foi projetado inicialmente para redes locais. As modificações que serão apresentadas a seguir visam reduzir o problema da implosão de ACKs e tornar o protocolo MCF mais flexível e escalável para utilização em redes de longa distância.

Considerando um grupo espalhado fisicamente por várias LANs interconectadas por roteadores, a primeira proposta consiste em subdividir o grupo, mantendo um servidor de fichas em cada subgrupo. Para entender melhor esse procedimento, imagine-se como exemplo a seguinte situação:

Um servidor de fichas no Rio de Janeiro obtém um endereço IP classe D de um roteador multiponto e cria um novo grupo. Ao tomar conhecimento da existência desse grupo, uma estação no Rio solicita adesão, enviando uma PDU JOIN. O candidato a novo membro recebe prontamente uma PDU ACCEPT do servidor, aderindo efetivamente ao grupo. Posteriormente, uma estação em São Paulo também deseja aderir a esse grupo, mas não recebe resposta à sua PDU JOIN, indicando que não há nenhum servidor de fichas nas proximidades. Assim, essa estação se declara servidor de fichas daquele subgrupo. A partir daí, no decorrer da existência do grupo, os dados são transmitidos para todo o grupo, não importando a localização dos participantes. Entretanto, as fichas são obtidas junto ao servidor local. Isso é importante para evitar situações em que um membro muito distante do originador do grupo seja impedido de transmitir seus dados devido ao longo atraso em sua linha de comunicação. Em suma, o serviço de concessão de fichas passa a ser distribuído.

Outra importante modificação se refere à realocação do servidor de fichas. Para isso, é preciso fazer algumas extensões ao mecanismo de gerenciamento de fichas original: assim que devolver a ficha ao servidor local, a estação deve disparar um temporizador, o CTIMER (de *confirmation timer*). Dentro desse intervalo de tempo, a estação esperará pela confirmação de recepção da ficha pelo servidor. Se CTIMER estourar, a estação deve reiniciá-lo e enviar ao servidor um pedido de **confirmação imediata** da devolução da ficha. Se nenhuma resposta for obtida, a estação considera que aconteceu algum problema com o servidor de fichas e se declara novo servidor, difundindo uma PDU CHANGE para todos os membros do grupo e informando seu endereço. Portanto, a **primeira** estação a detectar uma falha no servidor, no grupo local, será o novo servidor.

Quanto ao problema da implosão de ACKs, duas soluções estão sendo estudadas: configuração em árvore, conforme definição na seção 2.3, e implementação dos algoritmos de *damping* e *slotting*.

Na primeira opção, o servidor de fichas, que como já foi dito seria local a um subgrupo, concentraria também os ACKs de seus "filhos". Assim, o transmissor, esteja ele onde estiver, receberia um número de ACKs igual ao de servidores de ficha. Entretanto, tal solução implica o uso de um protocolo de roteamento subjacente, como ST–II, RSVP, CBT (*Core-Based Trees*) e PIM (*Protocol Independent Multicast*). A segunda opção (*damping* e *slotting*), apesar de também não ser simples, devido ao problema da configuração da duração dos temporizadores utilizados pelo *slotting*, tem aplicação mais imediata.

6.3 Incorporando mobilidade

A subdivisão do grupo em subgrupos locais, proposta na seção anterior, encoraja a incorporação de suporte à mobilidade no protocolo MCF. O servidor de fichas já acumula, além de sua função natural, o papel de concentrador de ACKs para serem repassados ao transmissor. Nesta seção, a proposta consiste em transformar o servidor de fichas também em estação de suporte móvel (MSS). Assim, cada subgrupo seria encarado como uma **célula**, a qual estações móveis poderiam se conectar. Essa proposta está baseada no trabalho apresentado em [2].

A execução do MCF para difusão de mensagens em redes móveis consiste em três etapas distintas:

1) Inicialmente, a mensagem é transmitida, na rede fixa, pelo iniciador para todos os MSSs participantes, juntamente com uma lista dos MHs destinatários da mensagem (essa lista é ignorada pelas estações fixas que não forem servidores de ficha).
2) A seguir, os MSSs transmitem a mensagem aos MHs locais pertencentes à lista de destinatários. O MSS receptor envia então uma mensagem de resposta ao iniciador, contendo um ACK para cada MH destinatário em sua célula.
3) É necessário um mecanismo que preveja a mudança de uma estação móvel de uma célula para outra: cada servidor de fichas (ou MSS) difunde periodicamente seu endereço na célula. Quando um MH recebe uma mensagem de identificação com endereço de MSS diferente, deve responder com uma "saudação" ao novo MSS, que funciona como um ACK. Essa mensagem contém seu próprio endereço e o endereço do MSS anterior. Dessa forma, o novo MSS pode solicitar ao anterior a transferência das informações relativas ao MH que acaba de entrar em sua célula.

O MCF resolve o problema da entrega de uma, e apenas uma, cópia de cada mensagem transmitida aos MHs associando uma lista com *N* entradas a cada MH **h**. Essas listas são mantidas apenas no MSS local, e cada uma registra o número de seqüência da **próxima** mensagem multidestinatária a ser enviada por cada um dos *N* MSSs do sistema que possam incluir **h** como receptor.

7 Conclusões

Este artigo propôs modificações ao protocolo MCF para adaptá-lo à realidade das aplicações multidestinatárias, procurando melhorar seu desempenho, reduzir a implosão de ACKs e torná-lo escalável, inclusive no que tange ao suporte a comunicações móveis.

Quanto às estratégias de implementação para aumento de desempenho, conclui-se que o mecanismo de ALF não se aplica ao MCF e uma implementação ILP não traria benefícios significativos. Nesse caso, a utilização de macros embutidas no código e/ou a implementação em *kernel* do sistema operacional poderiam trazer melhores resultados.

Utilizando a idéia básica do conceito de grupos locais, que consiste na distribuição geográfica em subgrupos para utilização em redes de longa distância, a primeira proposta no sentido de aumentar a escalabilidade do MCF consiste em designar-se um servidor de fichas para cada subgrupo. Isso resolveria principalmente o problema do atraso na aquisição de fichas em um grupo heterogêneo, constituído por várias subredes interconectadas.

Quanto ao problema da implosão de reconhecimentos, os algoritmos de *slotting* e *damping* mostram-se mais adequados à implementação imediata. Sem descartar o emprego desses algoritmos, serão também pesquisados mecanismos para estruturar o protocolo MCF hierarquicamente, de maneira dinâmica, em grupos locais. Possivelmente, a solução consistirá em uma estreita interação entre o protocolo MCF e algum protocolo de roteamento, como por exemplo o ST–II ou o RSVP. O servidor de fichas seria o líder ou "pai" do subgrupo, ficando responsável pelo recolhimento de ACKs e retransmissões locais. Estudos comprovam [7] que protocolos hierárquicos mantêm a vazão (teórica) constante, independentemente do número de receptores, mesmo quando a probabilidade de perdas não é desprezível. Além disso, permitem que a fonte trabalhe com memória finita, ofereça atrasos fim-a-fim limitados e processe apenas mensagens trocadas nos grupos locais.

Finalmente, os procedimentos descritos para que o MCF gerencie a transmissão multidestinatária para estações móveis não onera as máquinas em termos de processamento, é escalável (não existe limite para o número de estações móveis que, espera-se, seja muito maior que o de estações de suporte) e garante que uma estação móvel receberá sempre uma única cópia da mensagem a ela destinada.

Referências

[1] M. B. Abbott e L. L. Peterson, "Increasing Network Throughput by Integrating Protocol Layers", *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, outubro 1993.

[2] A. Acharya e B. R. Badrinath, "Delivering multicast messages in networks with mobile hosts", in *13th International Conference on Distributed Computing Systems*, maio 1993.

[3] T. Braun e C. Diot, "Performance evaluation and cache analysis of na ILP protocol implementation", *IEEE/ACM Transactions on Networking*, junho 1996.

[4] D. D. Clark e D. L. Tennenhouse, "Architectural considerations for a new generation of protocols", in *ACM SIGCOMM*, Philadelphia, pp. 200–208, setembro 1990.

[5] M. Fry e A. Ghosh, "Integrated Layer Processing: a Non-Trivial Application", submetido para publicação no *HPN 95*.

[6] M. Hofmann, "Adding scalability to transport level multicast", in *3rd COST 237 Workshop – Multimedia Telecommunications and Applications*, Barcelona, novembro 1996.

[7] B. N. Levine e J. J. Garcia-Luna-Aceves, "A comparison of known classes of reliable multicast protocols", in *Proceedings of International Conference on Network Protocols*, Columbus, outubro 1996.

[8] M. D. Nunes, C. V. N. de Albuquerque e O. C. M. B. Duarte, "An efficient implementation architecture for layered communication systems", in *38th Midwest Symposium on Circuits and Systems*, Rio de Janeiro, agosto 1995.

[9] M. D. Nunes e O. C. M. B. Duarte, "MCF: um protocolo multidestinatário configurável com gerenciamento de acesso por fichas de permissão", in *XIV Simpósio Brasileiro de Redes de Computadores*, Fortaleza, pp. 159–177, maio 1996.

[10] S. Pingali, D. Towsley e J. F. Kurose, "A comparison of sender-initiated and receiver-initiated reliable multicast protocols", *Performance Evaluation Review*, vol. 22, pp. 221–230, maio 1994.