MARCUS ANTONIO ALMEIDA RODRIGUES

UM FRAMEWORK PARA A PROVISÃO DE SERVIÇO DE MULTICAST EM AMBIENTES GENÉRICOS DE COMUNICAÇÃO DE DADOS

DISSERTAÇÃO DE MESTRADO

Departamento de Informática

Rio de Janeiro, 21 de Maio de 1999

MARCUS ANTONIO ALMEIDA RODRIGUES

UM FRAMEWORK PARA A PROVISÃO DE SERVIÇO DE MULTICAST EM AMBIENTES GENÉRICOS DE COMUNICAÇÃO DE DADOS

Dissertação de Mestrado apresentada ao Departamento de Informática da PUC/RJ, como parte dos requisitos para obtenção do título de Mestre em Informática: Ciência da Computação

Orientador: Luiz Fernando Gomes Soares

Departamento de Informática

Pontifícia Universidade Católica do Rio De Janeiro

Rio de Janeiro, 21 de Maio de 1999

D

Este trabalho é dedicado

A minha família, pelo esforço de me proporcionar as melhores condições para o meu desenvolvimento intelectual, e pelo amor e carinho a mim sempre dedicados.

Agradecimentos

- Ao meu Orientador, Professor Luiz Fernando Gomes Soares, pelo exemplo profissional, pela paciência e palavras de apoio nos momentos de dificuldade (pessoais e profissionais), e acima de tudo pelo amigo que tem sido.

- Ao Laboratório TeleMídia, pelo ótimo ambiente de trabalho, em especial aos amigos Sérgio Colcher, pela atenção, ajuda e orientação indispensáveis para a conclusão deste trabalho, Rogério Rodrigues e Débora Muchaluat, pela ajuda e atenção sempre presente.

- Ao Acadêmicos do TeleMídia®, pela amizade, acolhida, e pela honra e alegria de ter aprendido que nascemos e estamos prontos pra morrer. **Caminha TeleMídia!**

- Ao professores e funcionários do Departamento de Informática da PUC-Rio, pela formação e ajuda na elaboração desta dissertação, em especial a Professora Noemi Rodriguez e Professor Hugo Fuks.

- Aos professores da Universidade Estadual do Ceará, em especial a José Everardo Bessa Maia, Marcos Negreiros, Marcelo Aragão e José Neuman de Souza pelos conhecimentos transmitidos, e palavras de incentivo na escolha do meu caminho. Gostaria também de fazer um agradecimento especial ao professor e amigo Mauro Oliveira (“Aaaaêêêê!”), pela amizade, confiança, e pelo despertar da investigação científica.

- À República Federativa da Farinha, pela amizade, e por me trazer um pedacinho do Ceará nas várias mensagens trocadas em nossa lista. Obrigado a todos!

- A todos os meus amigos, em especial aos "irmãos" Roberto Façanha, Antônio Tadeu Azevedo Gomes, Romulo Martins de Menezes, Gabriel Paillard, Bruno Muniz, Cláudio Prata Santiago e Rafael Castro de Andrade, que me acompanharam nas diversas fases da elaboração deste trabalho, pela paciência nos momentos de estresse e imaturidade, pela força sempre presente nos momentos de dificuldade, e pala alegria constante nos momentos de alegria... E acima de tudo, por termos formado uma família. Gostaria também de agradecer a Família Martins de Menezes (Sr. Rafael V. de Menezes, D. Leda Martins de Menezes, Romulo, Denise e Rafael Jr.), por sempre me receberem em sua casa como membro da família, e palas agradáveis reuniões de família.

- A Andréa Serpa, pelo carinho sempre presente, apoio incondicional, e força na decisão de buscar meus sonhos. Obrigado, Andréa! Dedico também a ti esse trabalho!

- A toda minha Família, especialmente ao meu Painho (Gerardo Bandeira Rodrigues), Mainha (Maria de Almeida Rodrigues), Maninhos (Gerardo Bandeira Rodrigues Júnior e Ana Christine Almeida Rodrigues), e minhas sobrinhas (Edianne Christine Rodrigues Moura, Germana Rodrigues Moura e Luiza Helena Almeida Rodrigues), pelo apoio sempre presente e confiança incondicional, que foram importantíssimos durante todas as fases do meu trabalho, e pelo amor infinito. Vocês foram fonte de paz, estímulo, força e inspiração. **Amo vocês!**

- Ao CNPq, Embratel e PUC pelo suporte financeiro dado a este trabalho, sem o qual sua realização não teria sido possível.

- A Deus, pela luz e por ter tornado possíveis todos os agradecimentos anteriores.

"Escrevi demais por não ter tempo de escrever menos…"

C

“Good frameworks are usually the result of many design interactions and a lot of hard work.”

Wirfs-Brock and Johnson, 1990

Resumo

Reconhecendo que a maioria das soluções que têm sido apresentadas como candidatas à implementação do serviço de multicast tem sido projetadas tendo em mente determinadas condições da infra-estrutura utilizada para a distribuição de mensagens, ou requisitos de aplicações de mais alto nível, esse trabalho apresenta a proposta de um framework para a definição de um serviço de multicast cuja principal característica é a sua generalidade e independência em relação aos possíveis sistemas de comunicação, e aplicações. Partindo-se desse framework, é possível implementar serviços de multicast específicos de uma forma organizada e rápida através do reuso de toda a estrutura genérica apresentada, aliada a configuração de determinados componentes do serviço. O framework proposto é composto por dois serviços básicos: um serviço de gerenciamento de grupo, e um serviço de suporte a construção da infra-estrutura de distribuição multicast. A utilização do framework é ilustrada na implementação de um agente mediador para comunicação de grupo centralizado, utilizado em um serviço de distribuição de vídeo.

Palavras-Chave: Framework, Serviço Multicast, gerenciamento de grupo, roteamento multicast, especificação de protocolo.

Abstract

Since most of the solutions available as candidates to the implementation of a multicast service have been designed to fit specific infrastructure conditions or service characteristics to which they are aimed, or, still, to fit management politics for user groups, the present work introduces a framework for designing a multicast service whose main features are its generality and independence toward possible communication systems and applications. This framework allows the implementation of specific multicast services in a fast and organized way, by reusing the whole generic structure presented, in addition to the configuration of some service components. The framework consists of two basic services: a group management service, and a support service to the construction of the multicast distribution infrastructure. Its use is illustrated by the implementation of broker agent for centralized group communication, applied to a video distribution service.

Keywords: Framework, Multicast Service, group management, multicast routing, protocol specification.

Conteúdo

Índice de Figuras

Capítulo 1

Introdução

O gradual aumento de disponibilidade das redes de alta velocidade vem possibilitando cada vez mais o desenvolvimento de aplicações multimídia, como videoconferência, transferência de documentos, trabalho cooperativo, ensino a distância, correio eletrônico multimídia, vídeo sob demanda e distribuição de vídeo e/ou Áudio. Essas aplicações, além de exigirem a manipulação de objetos de dados não convencionais (que exigem taxas de transferência altas e contínuas e apresentação sincronizada dos dados) têm como requisito comum a necessidade de transmissão de dados para múltiplos usuários [Soares **et al 95**]. Essas características requerem atributos dos sistemas de comunicações[1] que ainda não estão plenamente disponíveis nas tecnologias mais comuns em utilização, como a eficiência na comunicação de grupos de multicast[2].

---

[1] O sistema de comunicação compreende os meios de transmissão e as regras empregadas para a interligação das entidades participantes.

[2] O termo em inglês "multicast" é utilizado no contexto de todo este trabalho, por não encontrarmos uma boa tradução para português do mesmo, e por ter seu uso amplamente difundido na área de redes de computadores.

Comunicação de grupo em sistemas de computação corresponde à troca de dados de diferentes mídias entre múltiplas entidades. Essas entidades podem utilizar diferentes sistemas de comunicação para a sua comunicação: sistemas operacionais e redes de computadores são alguns exemplos. Em comunicação de grupo, unidades de dados idênticas de um ou mais transmissores devem ser transmitidas para um grupo de receptores. Nesse contexto, multicast pode ser utilizado como um eficiente mecanismo de transferência de dados entre entidades do grupo.

Na literatura, pode-se encontrar um conjunto de definições de serviço de multicast. Kompella [Kompella et al 93] e Deering [Deering **et al 90**], por exemplo, definem multicast como a transmissão simultânea de dados para múltiplos usuários. Partridge [Partridge 93], define multicast como um serviço inter-rede que permite que origens enviem uma única cópia de um datagrama (ou célula) para um endereço de grupo, de modo que o datagrama seja entregue a múltiplos receptores. Nesse trabalho, um *serviço de multicast* é definido como um conjunto de procedimentos e interfaces que permitem enviar mensagens para um grupo de participantes em um ambiente de processamento e comunicação. A definição de um serviço multicast pode ser realizada de forma independente da implementação da infra-estrutura de distribuição de mensagens e das aplicações específicas às quais o serviço é destinado. Um serviço de multicast provê uma forma primitiva de comunicação de grupo que pode ser utilizado como base para o oferecimento de serviços mais complexos, destinados a diversas áreas específicas como trabalho cooperativo, por exemplo. Unicast pode ser definido como um caso particular de multicast, onde existe apenas um transmissor e um receptor, caracterizando assim uma comunicação ponto-a-ponto. Outro caso particular de multicast é quando temos uma transmissão para todos os participantes, caracterizando uma transmissão por difusão (broadcast).

A arquitetura genérica de um serviço de multicast pode ser dividida em duas partes: o gerenciamento de grupo e a construção de uma infra-estrutura de distribuição. Um grupo é definido como um subconjunto de usuários para o qual é possível a transmissão de mensagens, estando associado a um endereço de multicast

[Deering **et al 90**]. O conceito de grupo permite que várias entidades sejam definidas como se fossem uma só, pois têm um nome e endereço únicos, denominados respectivamente de nome do grupo e de endereço de grupo. A existência de um grupo é independente de qualquer instância de comunicação, ou seja, o grupo existe, independente de haver troca de informação [ISO **95a**]. O gerenciamento de grupo diz respeito a todas as ações relacionadas a composição do grupo, como manipulação de informações sobre os seus participantes e o controle sobre a entrada e saída de participantes ao grupo. A construção da infra-estrutura de distribuição está relacionada à forma de coordenação de recursos de processamento e comunicação para que a distribuição das mensagens possa ser efetuada. Em geral, a construção dessa infra-estrutura é efetuada de forma a tentar minimizar as replicações de mensagens desnecessárias e aumentar o desempenho dos recursos. Protocolos de roteamento são, em geral, responsáveis por grande parte desse trabalho no que concerne ao sistema de comunicação propriamente dito. Além disso, a criação dessa infra-estrutura está intimamente ligada aos mecanismos de provisão da qualidade de serviço (Quality of Service — QoS) solicitada pelos usuários.

A maioria das soluções que têm sido apresentadas como candidatas à implementação do serviço de multicast foram projetadas tendo em mente determinadas condições específicas de infra-estrutura para distribuição de mensagens, ou características do serviço às quais são destinadas, ou ainda, a forma de gerenciamento dos grupos de usuários. Com relação à infra-estrutura de distribuição, serviços tem sido implementados, em geral, de forma totalmente dependente de fatores como a “topologia virtual” do sistema comunicação. Em relação às características do serviço às quais são destinadas, muitas propostas são dependentes de características ou propósitos específicos tais como confiabilidade, ou garantias de retardo máximo e variação de retardo. Em relação ao gerenciamento dos grupos de usuários, a forma de distribuição e armazenamento das informações relativas aos grupos determina, muitas vezes, toda a arquitetura e implementação do serviço.

Técnicas de orientação a objetos (OO) têm sido recentemente aplicadas com sucesso ao projeto e à implementação de software de comunicação e no

desenvolvimento de sistemas distribuídos. Diferentes abordagens têm sido utilizadas durante os últimos anos, correspondendo a diferentes visões ou enfatizando diferentes aspectos de utilização da tecnologia OO. Todas essas visões são baseadas na idéia genérica de que técnicas de implementação e projeto orientado a objetos podem ser utilizadas para aumentar a modularidade e extensibilidade através da definição de interfaces estáveis, que encapsulam os detalhes da implementação [Schimidt **93**].

Aliados à tecnologia de OO, designs patterns representam soluções documentadas para problemas de desenvolvimento dentro de um contexto particular [Gamma **et al 95**]. Design patterns capturam experiências consagradas de desenvolvimento e ajudam a promover uma boa prática de projeto. Frameworks [Gamma **et al 95**] são aplicações reusáveis "semi-completas" que podem ser especializadas para produzir aplicações particulares. Designs patterns e frameworks têm sido aplicados em conjunto para melhorar a qualidade de software de comunicação, como exemplificado no framework ACE [Schimidt **97**].

1.1 Objetivo

Este trabalho visa a apresentar a especificação de um framework para a implementação de serviços de multicast cuja principal característica é a generalidade e independência em relação aos possíveis sistemas de comunicação, aplicações e formas de armazenamento e gerenciamento de grupos. Partindo-se desse framework, é possível implementar serviços de multicast específicos de uma forma organizada e rápida através do reuso de toda a estrutura genérica apresentada, aliada a configuração de determinados componentes do serviço. Esses pontos de configuração do framework são denominados pontos de flexibilização (ou hot spots) [Pree **95**], conforme apresentados no Capítulo 4.

Os pontos de flexibilização do framework para o gerenciamento de grupo dizem respeito à escala de distribuição das informações do grupo, além da estrutura do esquema de endereçamento utilizado pelo sistema de comunicação específico, como descrito na Seção 4.4. No serviço de roteamento, descrito na Seção 4.5, utiliza-se uma

única estrutura para a construção do protocolo de roteamento, a partir da qual aplicações ou serviços específicos podem configurar a estratégia de construção da infra-estrutura de distribuição multicast.

O framework proposto deverá poder ser utilizado em qualquer nível de comunicação, quer seja no nível de objetos dentro de uma mesma thread, quer seja de threads dentro de um processo, quer seja de processos em um nó, ou de entidades em uma camada de protocolo de uma rede e inter-redes, e assim por diante. Objetiva-se que o framework seja genérico e recursivo, de forma que possa ser empregado independente do nível das entidades participantes. A Figura 1.1 mostra possíveis cenários de aplicação do framework. A figura exemplifica a estruturação recursiva (a existência do serviço de multicast em um nível permite a sua utilização nos níveis adjacentes), uma vez que o framework pode ser aplicado para construção do sistema de comunicação no nível de objetos em uma thread, e no nível de threads em um processo, e assim em diante.

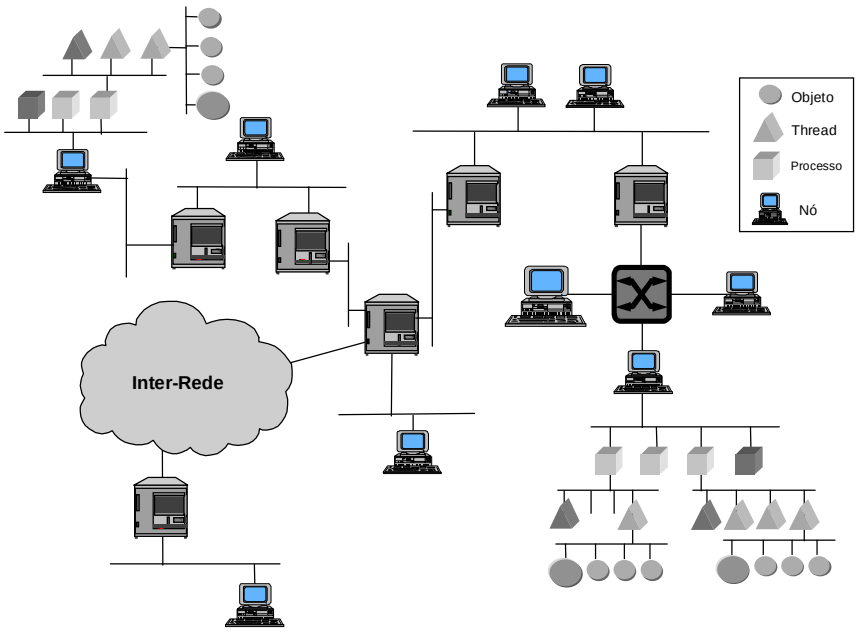

Figura 1.1: Aplicação do Framework

É importante salientar que o framework especificado não propõe inovações no roteamento, ou gerenciamento e alocação de endereços de grupo, limitando-se à

descrição dos design patterns necessários para a provisão de um serviço de multicast genérico.

Este trabalho faz parte do Modelo de Referência Unificado [**Colcher et al 98**] em desenvolvimento no Laboratório TeleMídia da Pontifícia Universidade Católica do Rio de Janeiro, cuja finalidade principal é oferecer suporte à construção de arquiteturas de protocolos de comunicação e aplicações distribuídas[3]. Maiores detalhes sobre o Modelo de Referência Unificado podem ser encontrados no APÊNDICE B, e em [**Colcher et al 98**].

1.2 Organização da Dissertação

O presente trabalho é organizado com se segue. No Capítulo 2 são apresentadas as características gerais de um ambiente que oferece um serviço de multicast, além de um conjunto de termos, princípios, definições e características do serviço. Nesse capítulo, é apresentado, com base em um conjunto de princípios relacionados à provisão do serviço de multicast, um modelo genérico de operações de grupo e associações de grupo, bem como os procedimentos básicos para o estabelecimento de uma comunicação multicast, baseado em um conjunto de recomendações definidas pelo ITU-T [**ITU-T X.6**] e pela ISO [**ISO 95a**, **ISO 95b**]. Com base nessa terminologia de comunicação multicast, são traçados os requisitos mínimos para a provisão de um serviço de comunicação multicast, divididos em duas categorias: gerenciamento de grupo e transmissão multicast. A partir desses requisitos foi extraído o núcleo dos serviço providos pelo framework proposto nesse trabalho.

---

[3] O trabalho encontra-se inserido na **Experiência de Campo de Faixa Larga do Projeto RAVEL - Redes de Alta Velocidade**. Esse projeto é baseado no convênio de pesquisa existente entra a PUC-Rio e a Empresa Brasileira de Telecomunicações (EMBRATEL), cujos objetivos incluem o estudo de novos serviços para redes de banda larga.

No Capítulo 3 é feito um levantamento dos principais serviços e protocolos relacionados a provisão do serviço de gerenciamento de grupo e roteamento multicast. É feito também nesse capítulo uma revisão geral sobre redes ativas, e de alguns serviços implementados para a provisão de multicast, de acordo com a filosofia de redes ativas.

A seguir, no Capítulo 4, é feita uma breve introdução a conceitos de orientação a objetos, e aplicação de patterns e frameworks no desenvolvimento e implementação de sistemas de comunicação e aplicações distribuídas. Em seguida, considerando os vários trabalhos apresentados no Capítulo 3, e nos requisitos descritos na Seção 2.2, é apresentada uma arquitetura genérica para a provisão do serviço de multicast. Finalmente, ainda nesse capítulo, é feita a descrição do Framework para Provisão do Serviço de Multicast. Essa descrição é feita detalhando-se os serviços e patterns componentes do serviço de multicast. Esse capítulo apresenta ainda, para cada componente do serviço, um exemplo de aplicação do framework. A arquitetura do framework é apresentada através da utilização da *Linguagem de Modelagem Unificada* (UML - Unified Modeling Language) [**Booch et al 96**].

No Capítulo 5, é apresentada a aplicação do framework como um todo, juntamente com o Framework para Provisão de Serviço de Qualidade de Serviço (QoS) [**Gomes et al 99**], para a implementação de um serviço de transmissão de vídeo para um grupo de assinantes com diferentes requisitos de qualidade de serviço, em uma rede ATM.

Finalmente, no Capítulo 6 são apresentadas algumas conclusões do trabalho, e suas principais contribuições, bem como pontos ainda em aberto, que poderão ser tratados em trabalhos futuros.

A dissertação apresenta ainda um apêndice com detalhes de UML relevantes a descrição do framework, utilizado no Capítulo 4.

Capítulo 2

Serviço de Multicast

Neste capítulo são apresentadas as características gerais de um ambiente que oferece um serviço de multicast, além de definições e características do serviço. Após essas definições, serão apresentados os requisitos mínimos para a provisão do serviço.

2.1 Modelo de Serviço Multicast

O serviço de multicast é responsável por prover os meios para que uma única mensagem transmitida pela origem alcance todos os participantes do grupo. O modelo de serviço multicast é ilustrado na Figura 2.1.

Define-se um *ambiente de processamento e comunicação* como sendo formado por um conjunto de entidades usuárias do serviço e por um provedor de serviço, responsável pelo processamento dessas entidades, assim como pela comunicação entre elas. Esse ambiente pode ser tão simples quanto um ambiente para a troca de mensagem entre objetos, presente em uma linguagem de programação OO, ou tão complexo quanto um ORB [**OMG 96**], um ambiente de processamento distribuído, formado por um conjunto de ambientes locais de execução de objetos e pela infra-estrutura de comunicação entre esses ambientes.

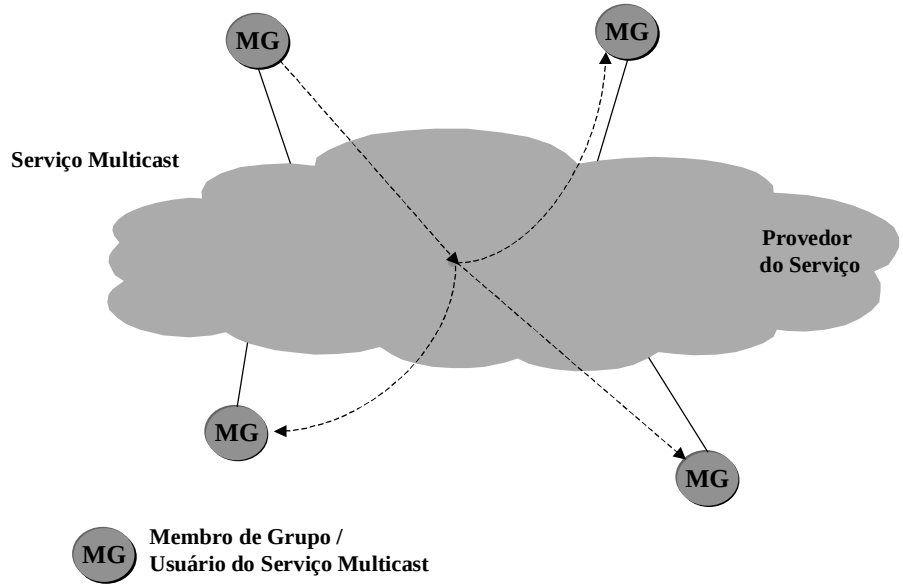

Figura 2.1: Modelo do Serviço de Multicast

O modelo de multicast descreve o serviço como uma entidade lógica única. Na prática, esse serviço pode ser provido por entidades internas ao provedor de serviço de modo centralizado ou distribuído. Estas entidades podem pertencer ou não ao sistema de comunicação utilizado. Estas entidades podem ainda pertencer ao mesmo sistema de comunicação, ou a diferentes sistemas de comunicação. A Figura 2.2 ilustra alguns exemplos de implementação do serviço de multicast, descritos a seguir. Esses exemplos não pretendem ser exaustivos, uma vez que outras implementações são possíveis.

A provisão do serviço de multicast depende muito do suporte à comunicação multicast oferecido pelo sistema de comunicação utilizado. Em sistemas que não possuem capacidade de comunicação multicast, entidades internas ao provedor de serviço podem se responsabilizar por prover o encaminhamento dos dados aos membros do grupo, ocultando do usuário todos os procedimentos necessários para o encaminhamento dos dados (*resolução de endereço, criação e manutenção da infra-estrutura de distribuição,* etc.), como pode ser visto na Figura 2.2 a). Por outro lado, se o sistema de comunicação já oferecer um suporte à comunicação multicast (ou broadcast), cabe ao provedor do serviço apenas mapear os mecanismos de multicast exigidos pelo usuário, para os mecanismos oferecidos pelo sistema de comunicação (*nativos*), como por

exemplo, mapear um endereço de grupo do nível do usuário para um endereço de grupo equivalente do nível do sistema de comunicação. Dessa forma, cabe às entidades internas ao sistema de comunicação, prover os mecanismos do serviço de multicast, como mostrado na Figura 2.2 b).

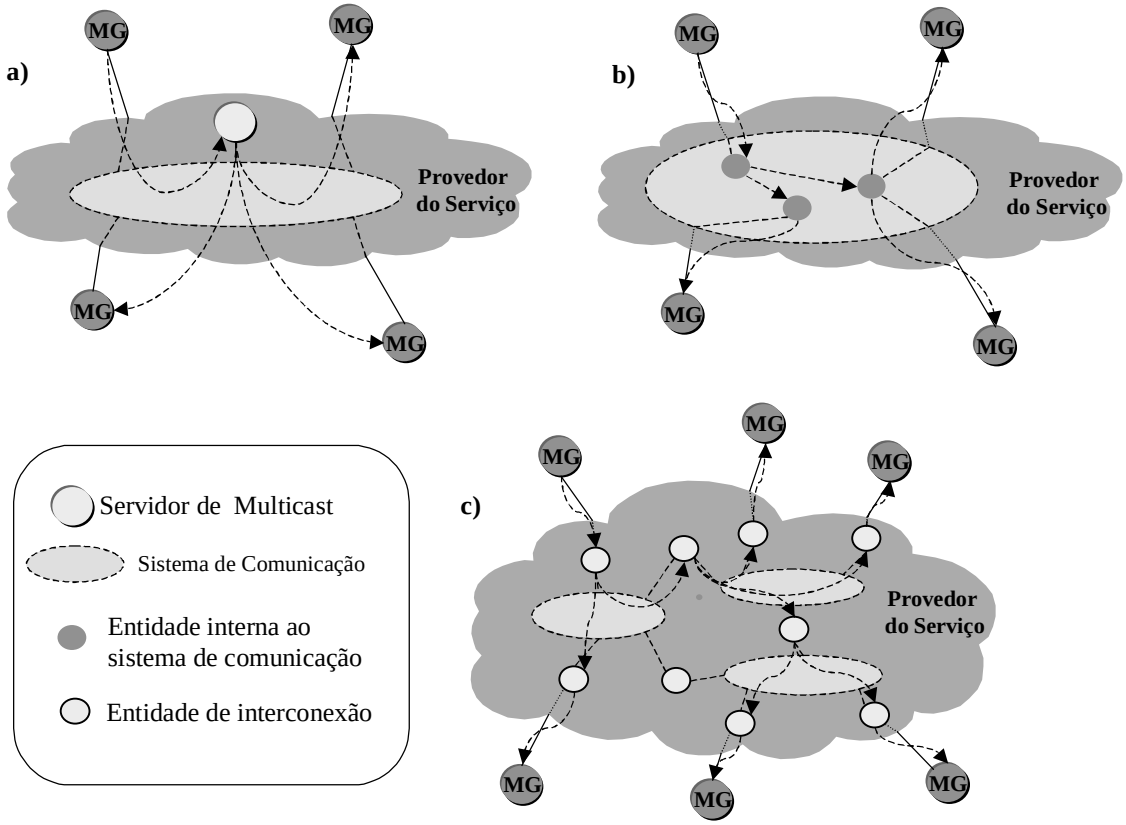

Figura 2.2: Exemplo de implementações do Serviço de multicast

O serviço de multicast pode requerer uma capacidade de comunicação entre entidades internas do provedor de serviço ligadas a diferentes sistemas de comunicação, como ilustrado na Figura 2.2 c). A comunicação entre as entidades internas ao provedor de serviço é provida de tal forma que usuários externos (membros de grupo) vêem o serviço como provido por uma única entidade lógica.

O modelo de serviço multicast aqui descrito é baseado em um conjunto de recomendações definidas pelo ITU-T [**ITU-T X.6**] e pela ISO [**ISO 95a**, **ISO 95b**].

2.1.1 Multicast, Multiponto ou Multiway ?

É importante notar a diferença entre os termos **multicast, multiponto** e ***multiway***. Na terminologia IP, a comunicação entre um grupo de participantes é refenciado como multicast [**Deering et al 90**]. O ATM Forum tem adotado o termo *multiway* uma vez que multicast pode ser confundido com comunicação ponto-a-multiponto. Algumas pesquisas [**Mauthe et al 95**, **Pasquale et al 98**, **Szyperski et al 93**] argumentam que uma vez que o termo multicast foi visto originalmente como uma restrição de comunicação *broadcast* em um meio de transmissão por difusão, não é apropriado para uso em situações que se utilizem de infra-estruturas que não forneçam suporte a comunicação por difusão, como é o caso das redes ATM. Por isso, argumentou-se que tanto os termos multiponto como multiway deveriam ser usados para evitar esse tipo de confusão. Multiponto é uma generalização de ponto a ponto para indicar múltiplos participantes de uma mesma comunicação, e multiway é um conceito similar que enfatiza que múltiplos participantes podem comunicar-se entre sim ao mesmo tempo.

Nesse trabalho, independente do suporte a comunicação por difusão dado pelo sistema de comunicação, multicast será usado para descrever o serviço de suporte a comunicação de grupo. Mas em alguns pontos os três termos serão usados alternadamente para preservar os termos usados em algumas referências. O termo *ponto a multiponto* será usado para descrever conexões ou canais de comunicação em que existe apenas um transmissor enviando dados para um grupo de receptores.

2.1.2 Grupos

O conceito de grupo permite que várias entidades sejam definidas como se fossem uma só, pois têm um nome e endereço únicos denominados respectivamente de *nome do grupo* e de *endereço de grupo*. A existência de um grupo é independente de qualquer instância de comunicação, ou seja, o grupo existe independente de haver troca de informação [**ISO 95a**].

Uma instância de um grupo é composta de *n* entidades, chamadas *membros*. A criação de um grupo é necessária para identificar o subconjunto de membros que estejam participando de uma comunicação. Essas entidades podem ser alcançadas pelo endereço (ou nome) do grupo, após terem passado pelos processos de *registro*, e *adesão*, explicados mais adiante.

Grupos de multicast podem ser classificados de acordo com algumas caraterísticas, como a dinâmica de seus membros, ou seu grau de distribuição. Abaixo são listadas algumas caraterísticas que definem os grupos:

- *Dinâmico* – a sua população (membros) muda dinamicamente;
- *Estático* – a população do grupo não muda;
- *Definido* – quando é possível definir a lista completa de participantes. Um grupo definido é dito completamente conhecido se todos os participantes se conhecem, caso contrário é dito parcialmente conhecido;
- *Indefinido* – quando não é possível determinar uma lista completa de participantes[4].

De acordo com a distribuição de seus membros através do sistema de comunicação, grupos multicast podem ser divididos em três categorias [**Deering et al 90**].

Grupos difundidos possuem membros em todos ou quase todos os enlaces ou redes na inter-rede. Exemplos de grupos difundidos são serviços de diretório distribuídos. Tais grupos tendem a ter um tempo de vida muito longo, tendo endereços multicast bem conhecidos.

---

[4] Dentro de um grupo indefinido, um participante pode ser conhecido por apenas alguns dos outros participantes. Nesse caso, os membros são chamados de *membros conhecidos*.

Grupos esparsos têm membros em apenas um pequeno número de enlaces. Exemplos de grupos esparsos são conferências de tempo-real, ou bancos de dados replicados. Tais grupos podem ter um tempo de vida longo ou passageiro.

Grupos locais possuem membros em apenas um único enlace ou um conjunto de enlaces dentro de um subdomínio administrativo da inter-rede. Exemplos de grupos locais são aplicações paralelas distribuídas ou jogos executados dentro de um único domínio[5]. Tais grupos tendem a ser passageiros, existindo apenas enquanto necessário para a execução de um único programa.

Para um grupo difundido, uma facilidade de *broadcast* eficiente que transporte um pacote multicast para todos os enlaces através de uma árvore de entrega de menor caminho pode oferecer um custo de entrega significantemente menor e retardos de entrega menores do que enviar pacotes unicast individuais para cada membro do grupo.

Para alguns grupos difundidos, tais como grupos de servidores de diretório, é essencial que um pacote multicast não seja entregue para todos os membros do grupo, mas apenas para vizinhos mais próximos. Isso é obtido pela facilidade de controle de escopo[6], que permite que um transmissor limite a distância de propagação de um pacote multicast. Não apenas é crucial para a escalabilidade de serviços difundidos, tais como serviço de diretório, como também permite que estações evitem ser bombardeadas por respostas de grupos "populares" [**Deering et al 90**].

A distinção entre grupos difundidos e esparsos é baseada no custo relativo de entrega *broadcast*, comparada a entrega multicast seletiva. Entrega *broadcast* implica no custo de entrega para todos os enlaces, independente se membros do grupo estão presentes ou não nesse enlaces (caro para grupos esparsos), ao passo que entrega seletiva

---

[5] Um domínio está associado a um espaço homogêneo de endereçamento, além de definir o espaço de transmissão broadcast [**Cecilio 97**].

[6] Um exemplo de controle de escopo pode ser conseguido através do campo TTL de um pacote IP.

recai no custo de tráfego de controle no qual os roteadores “aprendem” onde os membros do grupo estão localizados (caro para grupos difundidos). O custo exato depende da topologia particular da inter-rede, as distribuições dos tráfegos multicast, e os algoritmos de roteamento em uso. Assim, o limite entre grupos difundidos e esparsos varia de inter-rede para inter-rede.

Grupos locais são eficientemente tratados por um serviço de entrega multicast seletiva. Se todos os transmissores para um grupo local estão na mesma localidade, controle de escopo combinado com entrega *broadcast* pode prover multicast com baixo *overhead*.

Independente do tipo de grupo, é importante que as características de entrega de pacotes multicast sejam as mesma para pacotes unicast. Em particular, um pacote multicast deve ser entregue para cada membro de seu grupo destino (dentro do escopo especificado) com probabilidade e retardo muito próximos de um pacote unicast enviado para aquele mesmo membro. Essa propriedade dá aos protocolos de mais alto nível uma base para o tratamento de perda de pacotes por retransmissão. Baixo retardo é uma importante propriedade para um número de aplicações multicast, tais como conferências distribuídas, computação paralela e reserva de recursos. Além do retardo de entrega básico, a facilidade de multicast deve minimizar o retardo entre o tempo que uma estação une-se a um grupo e o tempo que ele começa a receber pacotes endereçados para aquele grupo, conhecido como *latência de união*[7]. Baixa latência de união é importante para minimizar a perda de pacotes após a união e para facilitar sincronização com a entrega unicast.

[7] Em uma rede local multiponto, por exemplo, esse tempo é geralmente apenas o necessário para atualizar o filtro de endereço local.

2.1.3 Associações de Grupo e Conversação

A partir do estabelecimento de um grupo, pode-se fazer associações entre membros deste grupo, denominadas de *associações de grupos*[8], com o propósito de transferir dados [**ISO 95a**]. O conceito de associação de grupo engloba ambos os modos de transmissão, com ou sem conexão. Os membros que aceitam participar de uma associação de grupo são denominados de *participantes* desta associação.

Conversações são efetivamente os canais criados num grupo para que haja troca de informações entre membros do grupo [**ISO 95a**]. Um grupo pode ter várias conversações ocorrendo simultaneamente, e cada membro pode também realizar diferentes conversações ao mesmo tempo. Conversações podem ser classificadas em *simplex multicast*, *duplex multicast*, *N-plex multicast*, ou *unicast*. Em conversações simplex multicast, um membro manda informações para todos os outros membros do grupo, que só recebem dados. Em conversações duplex multicast um único membro pode mandar mensagens para todos os outros membros da conversação, e esses membros podem enviar dados de volta somente para esse membro central. Conversações N-plex multicast permitem que todos os membros mandem e recebam dados dos outros membros, isto é, todo mundo se comunica com todo mundo. Conversações em que apenas um membro comunica-se com um outro, um a um, são conversações unicast.

Uma associação pode ser vista como composta de várias sub-associações, cada qual composta por *n* conversações.

---

[8] Associação de grupo, segundo a recomendação proposta pelo ITU-T [**ITU-T X.6**], é conhecida também como *conexão de grupo*, ou *chamada multicast*, com a diferença de que o serviço previsto pelo ITU é orientado a conexão.

2.1.4 Modelo de Serviço de Operação de Grupo

O modelo de serviço de operação de grupo define um conjunto de operações (e as respectivas operações inversas) necessárias para a constituição e manutenção de grupos (controle de entrada e saída de membros ao grupo) [**ISO 95a**].

2.1.4.1 Estabelecimento do Grupo

Esta primeira operação refere-se a fase de criação de um grupo, sendo realizada através da criação de estruturas de grupo. Este procedimento permite a um *criador de grupo*[9] especificar um conjunto de regras que definem o perfil dos futuros membros do grupo.

2.1.4.2 Registro de Grupo

Procedimento de iniciação que permite a uma entidade vir a ser instanciada como membro de um grupo. A entidade se torna conhecida pela gerência do grupo, o que permite depois passar por uma operação de adesão e assim vir a se tornar um membro deste grupo. Antes de ser registrada, a entidade tem que se submeter as regras que definem o grupo. Para isso, a entidade precisa apresentar suas caraterísticas ao grupo, por exemplo, fornecendo o endereço para ser incluído nos diretórios e tabelas do grupo, e os parâmetros que caracterizam suas capacidades para que sejam aplicadas a ela em comunicações posteriores.

2.1.4.3 Adesão ao Grupo

Torna uma entidade já registrada membro de um grupo. Pode-se fazer uma analogia entre a operação de adesão com a facilidade de um recurso passar de off-line a on-line (e vice-versa), sem que o sistema ache que o recurso deixou de existir. Um grupo é portanto composto por entidades que passaram pelas fases de registro e adesão.

---

[9] O criador de grupo definido pelo ITU-T [**ITU-T X.6**] é chamado de *Controlador de grupo de multicast*.

Adesões a grupos podem, opcionalmente, gerar eventos de notificação para indicar a entrada do novo participante ao grupo. Essas notificações podem ser utilizadas, por exemplo, para que transmissores que tenham criado uma associação possam adicionar o novo membro do grupo à associação.

2.1.4.4 Estado do Grupo

O estado do grupo provê informações sobre o grupo de multicast.

As seguintes informações podem ser providas pelos procedimentos de consulta ao estado do grupo:

- Lista de membros do grupo de multicast;
- Informações de mapeamento de endereço de grupo, quer seja para um endereço de grupo do sistema de comunicação do nível inferior, quer seja para os endereços unicast de todos os membros do grupo;
- Que membro é o controlador do grupo;
- Atributos de qualidade de serviço do grupo, ou de membros do grupo;
- Associações de grupo ativas.

2.1.5 Modelo de Serviço de Operação da Associação de Grupo

O modelo de serviço de operação de associação define um conjunto de operações para a criação e manutenção de associações para a transferência de informações entre os membros de um grupo de multicast.

2.1.5.1 Estabelecimento e Liberação

Na operação de estabelecimento, um membro de um grupo cria uma associação entre os membros desse grupo. Nessa operação são definidas todas as características da associação, como por exemplo, a qualidade de serviço (QoS) da

associação. O membro que inicia o estabelecimento da associação é chamado de *iniciador*. A operação de estabelecimento da associação é feita através de um pedido aos membros do grupo, podendo estes aceitarem ou não.

Define-se como *condições de estabelecimento*, as condições mínimas especificadas pelo iniciador para que a associação seja criada, tendo que ser coerente com as caraterísticas do grupo. Se as condições mínimas forem alcançadas, uma nova associação é criada e os participantes dessa associação formam um *grupo ativo*.

A operação de liberação é utilizada para encerrar uma associação, podendo ser iniciada pelo usuário ou pelo provedor (quando as condições de integridade da associação não mais forem alcançadas).

2.1.5.2 Adesão (Join) e Abandono (Leave)

A operação de adesão a uma associação permite que uma entidade se junte a uma associação de acordo com as regras da associação. Existem dois tipos de adesão:

- Adesão por convite – onde os participantes podem convidar uma entidade a se juntar à associação;
- Adesão por chamada – a entidade requisita a participação numa associação existente. Essa operação assume que a entidade sabe que a associação foi previamente estabelecida.

A operação de abandono de uma associação permite que o participante deixe a associação, podendo ser iniciado pelo participante que quer deixar a associação ou por outro participante. A operação também pode ser iniciada pelo provedor do serviço, quando ele não puder mais satisfazer à qualidade do serviço negociada durante o procedimento de estabelecimento da associação.

Opcionalmente, notificações podem ser feitas para indicar aos outros membros do grupo sobre a adesão ou saída de um membro a uma dada associação.

2.1.5.3 Transferência de Dados

Transferência de *(N-1)*SDUs entre os participantes de uma associação do nível *N* de acordo com alguns parâmetros de qualidade de serviço (QoS) negociados entre os participantes da associação. A transferência de dados só é permitida se as condições de integridade da associação são satisfeitas.

2.1.6 Procedimentos para o estabelecimento de uma comunicação multicast

Na comunicação multicast, um usuário manda dados para um determinado grupo de usuários, e também pode receber dados dos mesmos. Para isso são formados grupos de multicast. Nesta seção são expostos os procedimentos necessários para a transferência de dados multicast, considerando que o grupo já foi criado.

2.1.6.1 Diagrama de Estados

Para que uma entidade possa participar de uma comunicação multicast, é necessário que se associe um grupo já formado com o qual ela deseja trocar dados. Considerando que o grupo já esteja formado, e registrado, temos duas fases no diagrama de estados: (i) a fase de adesão (onde a entidade se une a determinado grupo), e (ii) a fase de transferência de dados.

O primeiro estado, no início da fase de adesão, é o estado de grupo criado. Nesse estado, assume-se que:

- Já existe uma associação criada entre os membros do grupo;
- As condições de integridade da associação estão determinadas e não são negociáveis;
- Existe uma lista de conversações no grupo;

- Existe uma lista de QoS, descrevendo a qualidade de serviço de cada conversação em curso.

A Figura 2.2 apresenta o diagrama de estados de uma comunicação multicast, cuja descrição é realizada no próximo item. As transições do diagrama estão descritas na Tabela 2-1.

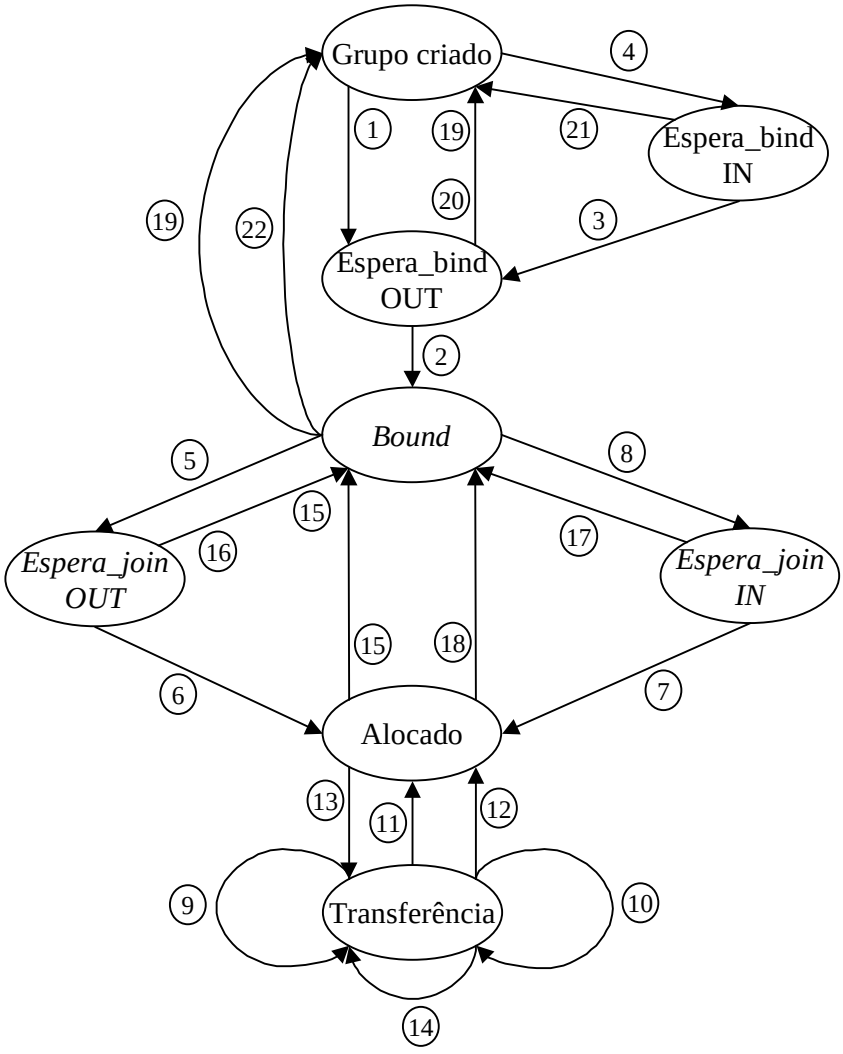

Figura 2.3: Diagrama de Estados

2.1.6.2 Tabela de Primitivas e Parâmetros

Tabela 2-1: Tabela de Primitivas para o estabelecimento de uma comunicação multicast

	Fase	Primitiva de Serviço	Parâmetros
1	Adesão	BIND.requestor.submit	Endereço chamador, end. Chamado, condições, QoS
2		BIND.requestor.deliver	Endereço chamador, end. Chamado, condições, QoS
3		BIND.acceptor.submit	Endereço chamador, end. Chamado, condições, QoS
4		BIND.acceptor.deliver	Endereço chamador, end. Chamado, condições, QoS
5		JOIN.requestor.submit	Endereço chamador, end. Chamado, condições, QoS
6		JOIN.requestor.deliver	Endereço chamador, end. Chamado, condições, QoS
7		JOIN.acceptor.submit	Endereço chamador, end. Chamado, condições, QoS
8		JOIN.acceptor.deliver	endereço chamador, end. Chamado, condições, QoS
9	Dado	DATA.requestor.submit	referência chamador, ref. Chamado, dados do usuário
10		DATA.acceptor.deliver	referência chamador, ref. Chamado, dados do usuário
11		PAUSE.requestor.submit	endereço chamador, razão
12		PAUSE.acceptor.deliver	endereço chamador, razão
13		READY.acceptor.submit	endereço chamador
14		REPORT.acceptor.deliver	endereço chamador
15	Saída	LEAVE.requestor.submit	endereço chamador, endereço chamado, razão
16		LEAVE.requestor.deliver	endereço chamador, endereço chamado, razão
17		LEAVE.acceptor.submit	endereço chamador, endereço chamado, razão
18		LEAVE.acceptor.deliver	endereço chamador, endereço chamado, razão
19		UNBIND.requestor.submit	endereço chamador, endereço chamado
20		UNBIND.requestor.deliver	endereço chamador, endereço chamado
21		UNBIND.acceptor.submit	endereço chamador, endereço chamado
22		UNBIND.acceptor.deliver	endereço chamador, endereço chamado

As primitivas *requestor* são iniciadas pelo próprio usuário, e as primitivas *acceptor* são iniciadas pelo provedor do serviço. Existem quatro estados principais: grupo criado, *bound*, alocado, e transferência.

2.1.6.3 Descrição dos Procedimentos do Diagrama de Estados

Fase de Adesão

- *BIND:*

No caso de um usuário iniciar o pedido de entrada na associação:

1. Quando um usuário deseja iniciar ou se juntar a uma associação, ele manda ao provedor um *BIND.requestor.submit(endereço chamador, endereço chamado, condições de associação, QoS)*, e entra no estado *espera_Bind OUT*. A associação é identificada pelo endereço chamado (passado como parâmetro da primitiva). Essa primitiva também manda a QoS, pois ela pode ter sido modificada pelo usuário em algum momento depois que ele se tornou integrante do grupo.

2. Em resposta a um *BIND.requestor.submit*, o provedor obtém informações sobre essa associação através de troca de PDUs internas com os participantes da associação. Em seguida, o provedor responde ao usuário com um *BIND.requestor.deliver*, levando como parâmetros as condições de integridade e QoS mais recentes na associação, e fazendo com que o usuário passe para o estado *bound*.

No caso do provedor pedir ao usuário que se associe

1. O provedor manda ao usuário2 uma primitiva *BIND.acceptor.deliver* (ao iniciar uma associação após o pedido feito por um usuário1). Essa primitiva contém os parâmetros de QoS que o usuário1 mandou ao fazer o pedido, ou a QoS modificada de acordo com a capacidade do provedor.

2. O usuário2 vai então para o estado *espera_bind IN*. Se o usuário2 aceitar as condições, ele manda ao provedor uma primitiva *BIND.acceptor.submit* com a QoS que ele pode manter e entra no estado *espera_bind OUT*. Nesse estágio, o usuário2 ainda não sabe qual será a QoS final negociada.

3. O Provedor passa ao usuário2 a QoS final, conhecida através de negociações com os outros usuário da associação, através da primitiva *BIND.requestor.deliver* e o usuário2 passa ao estado *Bound*.

- *JOIN*

No caso do usuário iniciar o pedido:

1. Quando o usuário está no estado *Bound* e aceita as condições de integridade e QoS definidas pelo provedor, ele se une à associação mandando ao provedor um *JOIN.requestor.submit*, sem mais negociações de QoS e passa para o estado *espera_join OUT*.

2. O provedor troca PDUs internas com os componentes da associação, para saber das atuais condições de integridade da associação. Caso sejam satisfeitas, ele responde ao usuário com uma primitiva *JOIN.requestor.deliver*, contendo o valor da QoS final negociada. O usuário passa então para o estado *Alocado*.

No caso do pedido iniciado pelo provedor:

1. O provedor convida um usuário a entrar na associação mandando uma primitiva *JOIN.accept.deliver* e o usuário entra no estado *espera_join IN*.

2. Se o usuário aceita as condições, manda uma primitiva *JOIN.acceptor.submit* e passa para o estado *alocado*.

A negociação de QoS é feita durante o BIND, e a negociação das condições de integridade do grupo feita durante o JOIN.

O estado final da fase de alocação é o estado *alocado*. Desse estado, são iniciadas as tentativas de transferência de dados. Nesse estado podem ou não existir conversações abertas. Nesse caso, provedor é quem decide estabelecer conexões multiponto para uma conversação dependendo das condições de QoS.

Fase de Liberação

- *UNBIND*

Pode acorrer mediante as seguintes condições: (i) no caso do usuário querer remover o seu *binding*; (ii) no caso do provedor querer remover o *binding* do usuário ou (iii) no caso do provedor indicar ao usuário que seu *binding* à associação de grupo não é possível.

Procedimentos para a condição (i):

1. Caso o usuário queira remover o seu *binding* (após ter enviado a primitiva *BIND.requestor.submit*) e passado ao estado *espera_bind OUT*, ele envia ao provedor a primitiva *UNBIND.requestor.submit* e retorna ao estado de grupo criado.

2. Caso o usuário não queira fazer o *binding* após receber do provedor o *BIND.acceptor.deliver*, no estado *espera_bind IN*, ele envia ao provedor o *UNBIND.acceptor.submit* e retorna ao estado de grupo criado.

3. Caso o usuário queira fazer um *UNBIND* quando se encontra no estado *bound*, ele envia uma primitiva

UNBIND.requestor.submit ao provedor e retorna ao estado de grupo criado.

Procedimentos para a condição (ii):

1. Se o usuário estiver no estado *espera_bind OUT*, tanto devido a uma iniciativa do usuário através do envio da primitiva *BIND.requestor.submit*, tanto por iniciativa do provedor através do envia da primitiva *BIND.acceptor.deliver*, (e confirmado pelo usuário através de um *BIND.acceptor.submit*), o provedor pode enviar uma primitiva *UNBIND.requestor.deliver*, indicando que não foi possível o *bind* à associação, provocando um retorno ao estado grupo criado.

Procedimentos para a condição (iii):

1. Se o usuário estiver no estado *bound*, o provedor pode solicitar o seu *UNBIND* enviando uma primitiva *UNBIND.acceptor.deliver*, provocando uma transição ao estado de grupo criado.

- *LEAVE*

Pode ocorrer de acordo com as seguintes condições: (i) quando um usuário desejar sair da associação; (ii) no caso do provedor querer remover algum usuário, e (iii) quando o provedor avisa ao usuário que não foi possível realizar o JOIN à associação.

Procedimentos para a condição (i):

1. Caso o usuário queira remover o seu pedido de adesão à associação a partir do estado *espera_join OUT* (alcançado após o envio da primitiva *JOIN.requestor.submit*), ele envia ao

provedor a primitiva *LEAVE.requestor.submit* e retorna ao estado *bound*.

2. Caso o usuário não queira fazer o JOIN após receber do provedor um *JOIN.acceptor.deliver* (estando no estado *espera_join IN*), ele envia ao provedor o *LEAVE.acceptor.submit* e retorna ao estado *bound*.

3. Caso o usuário queira fazer um *LEAVE* quando se encontra no estado *alocado*, ele envia uma primitiva *LEAVE.requestor.submit* ao provedor, e retorna ao estado *bound*.

Procedimentos para a condição (ii):

1. Se o usuário estiver no estado *espera_join OUT*, alcançado depois de enviar a primitiva *JOIN.requestor.submit* ao provedor, o mesmo pode enviar uma primitiva *LEAVE.requestor.deliver*, indicando que não foi possível a adesão à associação, retornando o usuário ao estado *bound*.

Procedimentos para a condição (iii):

1. Se o usuário estiver no estado *alocado*, o provedor pode solicitar o sua saída enviando a ele a primitiva *LEAVE.acceptor.deliver*, passando então ao estado *bound*.

Fase de Transferência de Dados

1. Quando o provedor determina que a transferência de dados pode ser feita de uma forma segura e as condições de integridade do grupo são satisfeitas, ele manda ao usuário uma primitiva *READY.acceptor.deliver* e o usuário entra no estado de *transferência*.

2. No estado de transferência, o usuário requisita transferência de dados através da primitiva *DATA.requestor.submit*.

3. O provedor então envia os dados para o destino usando *DATA.requestor.deliver*.

4. Quando ocorre uma falha nas condições de integridade do grupo ou QoS, o provedor envia aos usuários uma primitiva *PAUSE.acceptor.deliver* e suspende a transferência de dados. O usuário volta ao estado alocado.

5. O usuário pode suspender a transferência de dados enviando ao provedor uma primitiva *PAUSE.requestor.submit*. O usuário volta ao estado alocado.

6. Se os valores de *threshold* de um ou mais parâmetros de QoS são alcançados, o provedor avisa o usuário através da primitiva *REPORT.acceptor.deliver*.

2.2 Requisitos e Características de um Serviço de Multicast

Aplicações de grupo adicionam uma série de novas funções às arquiteturas de sistemas de computação (incluindo sistemas operacionais, redes e seus protocolos de comunicação, etc.). Embora os requisitos de aplicações sejam muitas vezes específicos, certas funções genéricas para o suporte a comunicação de grupo podem ser identificadas. Nessa sessão, serão listados alguns desses requisitos, visando a especificação de um serviço genérico de multicast.

2.2.1 Gerenciamento de Grupo

Nesse trabalho, o termo *gerenciamento de grupo* é utilizado para descrever todas as ações relacionadas à composição do grupo, como manipulação de informações sobre os seus participantes, e o controle de acesso ao grupo, isto é, controle sobre a entrada e saída de participantes ao grupo.

Grupos devem ser administrados por um gerenciador de grupo. O gerenciador de grupo deve ser distribuído por todas as camadas da arquitetura do sistema (abstrações do sistema de comunicação), isto é, cada nível da arquitetura deve possuir um gerenciador. Em cada nível o gerenciador pode ser centralizado, com um gerenciador de grupo executando todas as funções relacionadas à administração dos grupos, ou descentralizado.

O gerenciador de grupo é ativado pelo *iniciador* da comunicação de grupo. Ele deve negociar um único identificador de grupo que será associado ao grupo. Dependendo da localização da comunicação de grupo, isso pode envolver outros gerenciadores de grupo e gerenciadores de endereço. Se o grupo for destruído, o gerenciador de grupo deve liberar o endereço de grupo e possivelmente notificar outras entidades que o grupo não mais existe.

O gerenciamento de grupo envolve o gerenciamento de seus integrantes, seus papéis e estados, além do gerenciamento das características do grupo. Para grupos dinâmicos, deve ser possível adicionar e remover membros ao grupo, alterar seus papéis, e as características do grupo.

O gerenciamento de grupo deve oferecer um serviço de diretório, isto é, prover um serviço de informações sobre o grupo e seus integrantes. Informações sobre endereços, estado e características do grupo e dos membros do grupo devem ser fornecidas aos membros do grupo e entidades que não pertençam ao grupo.

O gerenciamento do grupo deve também gerenciar uma lista de notificações. Notificações sobre adesões e abandonos de grupo é uma tarefa do gerenciador do grupo.

As ações relacionadas ao gerenciamento de grupo dependem da natureza da aplicação. No entanto, algumas funções podem ser consideradas genéricas. Dentre elas podemos destacas:

- Criação e extinção de grupo e a associação das características de grupo são funções básicas necessárias. Durante a criação do grupo, um único identificador de grupo deve ser associado ao mesmo.
- Gerenciamento de grupos e membros do grupos, e manutenção de suas características também se fazem necessárias. Isso inclui o suporte a adesões e saída de membros de grupo, e alterações dinâmicas das características do grupo e seus membros. Nesse contexto, um requisito chave para aplicações de grupo diz respeito a sua escalabilidade.
- Gerenciamento de informações sobre os membros do grupo. A instância de gerenciamento deve ser capaz de fornecer informações sobre o grupo e responder a consultas a respeito de seus membros. Isso inclui o mapeamento do endereço do grupo e aos vários endereços dos participantes do grupo.

2.2.2 Transmissão Multicast

Para um serviço de transmissão multicast, o provedor do serviço pode utilizar diversos sistemas de comunicação interligados e estruturados em diversos níveis de abstração, como mostrado na Figura 2.2c). Uma outra visão dessa arquitetura pode ser vista na Figura 2.4. Nesse contexto, é necessário a definição de um eficiente mecanismo de resolução de endereços entre níveis adjacentes, bem como para a construção da infra-estrutura de distribuição e roteamento.

Resolução de endereço é definida como o mapeamento entre endereços de níveis adjacentes de uma arquitetura de comunicação, isto é, o mapeamento entre um endereço de nível *N* e o endereço do nível *N-1* associado a uma entidade do nível *N* [**X200 97**]. É função da entidade do nível N fazer esse mapeamento.

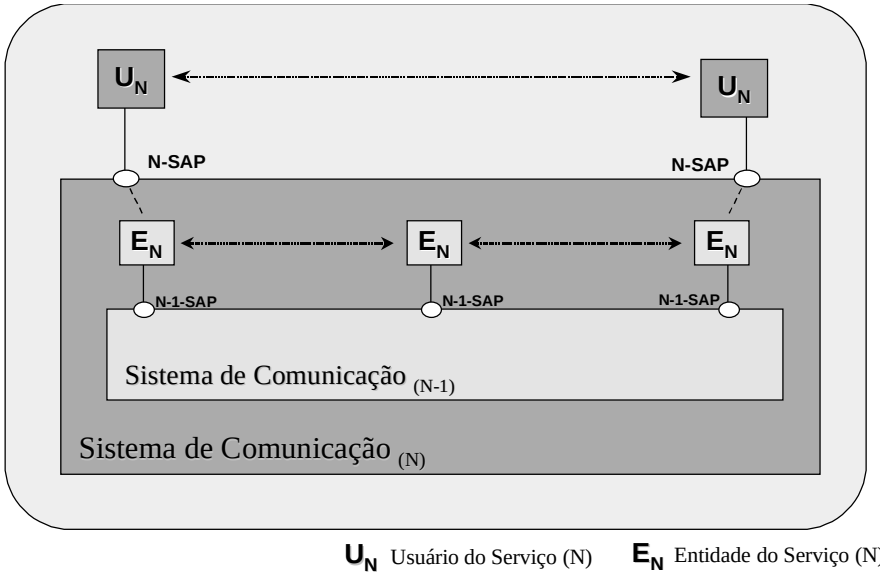

Figura 2.4: Arquitetura do Sistema de Comunicação

Os sistemas de comunicação podem ser formados pela interconexão de diferentes sistemas de comunicação. Para permitir a comunicação entre entidades ligadas a diferentes sistemas de comunicação pode-se fazer uso de entidades intermediárias. Para isso, é necessário definir um caminho através dessas entidades intermediárias, formando assim uma *árvore de distribuição*. Uma árvore de multicast (ou canal de comunicação multicast) é definida por uma comunicação *1 x n* entre uma entidade origem e *n* destinos.

Um protocolo de roteamento multicast é responsável pela construção de árvores de distribuição multicast e por habilitar a transmissão de dados multicast [Kompella et al 93]. Existem algumas técnicas exploradas por protocolos de roteamento. *Flooding* e Árvore Geradora [Maufer et al 97] são dois algoritmos que podem ser usados para construir protocolos de roteamento primitivos. As técnicas são primitivas porque tendem a desperdiçar banda passante ou requerem uma grande quantidade de recursos computacionais entre as entidades de roteamento envolvidas. Técnicas de construção de

árvores baseadas na origem do fluxo de dados [Maufer et al 97] mostram-se mais eficientes, uma vez que fundamentam-se na construção de uma árvore para cada transmissor de um grupo. A diferença entre as técnicas baseadas na origem encontra-se na eficiência de construção da árvore, na demanda de banda passante e de recursos das entidades de roteamento requeridos para construí-las. As técnicas de transmissão *multicast* mais recentes baseiam-se em uma árvore de entrega compartilhada [Ballardie 97a]. Diferentemente dos algoritmos de árvore de menor caminho [Maufer et al 97], que constroem uma árvore para cada origem ou par (*origem, grupo*), algoritmos de árvore compartilhada constroem uma única árvore de entrega que é compartilhada por todos os membros do grupo. No capítulo seguinte, veremos em mais detalhes alguns desses protocolos.

Após a criação da infra-estrutura de comunicação, o processo de transmissão pode ser iniciado, de acordo com os parâmetros da qualidade de serviço exigidos pelas entidades.

2.3 Resumo

Neste capítulo, a terminologia básica a ser usada na descrição do serviço de comunicação multicast foi apresentado. Também foi apresentado, com base em um conjunto de princípios relacionados à provisão do serviço de multicast, um modelo genérico de operações de grupo e associações de grupo, bem como os procedimentos básicos para o estabelecimento de uma comunicação multicast.

Com base nessa terminologia de comunicação multicast, foram traçados os requisitos mínimos para a provisão de um serviço de comunicação multicast, divididos em duas categorias: gerenciamento de grupo e transmissão multicast.

Capítulo 3

Trabalhos Relacionados: Soluções Presentes

3.1 Serviços e Protocolos de Multicast

Serviços de multicast são oferecidos em diferentes sistemas e diferentes níveis de arquitetura. Usuários desses serviços também variam de acordo com a característica do ambiente em que o serviço é oferecido. Em um sistema operacional, por exemplo, um grupo pode ser definido como um conjunto de processos, que se comunicam através da troca de mensagens [Tanenbaum **92**]. No sistema ANSA (Advanced Network Systems Architecture), *grupos de objetos* são introduzidos como uma abstração para comunicação de grupo [Mauthe **et al 95**]. Grupos possuem uma interface de gerenciamento que oferece funções para a adesão e abandono de grupo, adição e remoção de categorias de policiamento, etc. Em sistemas de comunicação, multicast é oferecido por protocolos de uma camada a entidades das camadas imediatamente superiores.

Em redes locais, a comunicação de grupo depende, em geral, da capacidade da rede subjacente para a difusão de mensagens. Os protocolos padronizados pelo IEEE 802.x e FDDI suportam transmissão multicast [Soares **et al 95**].

3.1.1 IP Multicast

O trabalho de Deering , *IP Multicast* [Deering **89**], foi um dos primeiros a considerar o suporte a multicast num ambiente composto por diversas redes. Essencialmente, Deering projetou um método para roteadores determinarem, dinamicamente, como datagramas IP devem ser transmitidos para *grupos de multicast*. Quando um roteador recebe um pacote endereçado a um grupo, envia-o por todas as interfaces que levam a membros do grupo. Se um grupo multicast estende-se por múltiplas redes, informações de membros de grupo são trocadas entre os roteadores pelo Protocolo de Gerenciamento de Grupo Inter-rede (IGMP – Internet Group Management Protocol) [Fenner 97].

Modificações que permitem uma estação transmitir IP Multicast não são difíceis. O software IP deve permitir que um programa de aplicação especifique um endereço de multicast como endereço de destino IP, e o software de interface de rede deve ser capaz de mapear um endereço IP Multicast para o endereço de multicast do hardware correspondente (ou usar broadcast se o hardware não suportar multicast).

Extensões ao software da estação para receber datagramas IP Multicast são um pouco mais complexas. O software IP da estação deve ter uma interface que permita a um programa de aplicação declarar que ele quer fazer parte de um grupo particular de multicast, ou abandoná-lo. Desta forma, a interface do serviço IP deve prover duas novas operações: *JoinHostGroup(group_address, interface*), que requisita que a estação se torne membro do grupo identificado por "*group_address*" na interface de rede dada; e *LeaveHostGroup(group_address, interface)*, que requisita que estação abandone o grupo. Se múltiplos programas de aplicação unem-se a um mesmo grupo, o software IP deve passar a cada um deles uma cópia dos datagramas recebidos destinados a este grupo. Se todas os programas de aplicação deixarem o grupo, a estação não mais deve fazer parte

do grupo. Além disso, como será visto na próxima sessão, a estação deve executar um protocolo que informe os roteadores de multicast locais o seu estado de participação em grupos.

Muito da complexidade da recepção vem da idéia básica de que "*estações unem-se a grupos de multicast IP específicos em redes específicas*". Isto é, uma estação com múltiplas conexões de rede pode unir-se a um grupo de multicast particular em uma rede (interface), e não em uma outra[10]. Porque associação a grupos é relacionada com redes particulares, o software deve manter listas separadas de endereços de multicast para cada rede a qual a máquina está ligada. Além disso, um programa de aplicação deve especificar uma rede particular quando requisita uma adesão ou saída de um grupo de multicast. Para suportar a recepção de datagramas de IP Multicast, o módulo IP deve ser estendido para manter uma lista de membros de grupos associados a cada interface de rede.

Em estações com mais de uma interface de rede, se um datagrama chegar por uma interface, destinado a um grupo que a estação pertence associado a uma outra interface, o datagrama é descartado, da mesma forma que um datagrama destinado a um grupo que a estação não pertence.

A lista de membros de grupos (protocolos de alto nível) é atualizada em resposta as requisições *JoinHostGroup* e *LeaveHostGroup* dos protocolos dos níveis superiores. Na primeira requisição para o "*join*", ou a última para o "*leave*" a um grupo em uma dada interface, o módulo de rede local para aquela interface é notificada de modo a atualizar seus filtros de recepção multicast.

---

[10] Note que é possível usar IP Multicast entre conjuntos de máquinas locais. A estação pode querer usar uma aplicação multicast para interagir com máquinas em uma rede física, mas não com máquinas em uma outra.

O módulo IP deve também ser estendido para implementar o protocolo **IGMP** (*Internet Group Management Protocol*), usado para manter os roteadores de multicast vizinhos informados sobre as estações membros dos grupos em uma rede particular. Para suportar IGMP, toda estação (de nível 2) deve unir-se ao grupo "*all-hosts*" (224.0.0.1) em cada interface de rede em tempo de iniciação, e permanecer enquanto a estação estiver ativa.

Para permitir que o módulo IP diga ao módulo de rede local quais pacotes multicast receber, a interface de serviço de rede local deve ser estendida para prover duas novas operações: *JoinLocalGroup(group_address)*, que requisita ao módulo de rede que receba e entregue ao módulo IP pacotes destinados aos grupo endereçado por *group_adress*; e *LeaveLocalGroup(group_address)*, que requisita ao módulo de rede que pare de receber pacotes destinados ao grupo endereçado por *group_address*.

3.1.2 Backbone Multicast da Internet (MBone)

O Backbone Multicast da Internet (**MBone - *Multicast Backbone***) [**Kumar 96**] é um conjunto interconectado de subredes e roteadores que suportam a entrega de tráfego IP Multicast. O objetivo do MBone é construir um *testbed* de IP multicast para habilitar o desenvolvimento de aplicações multicast sem esperar pelo desenvolvimento de roteadores com capacidade de multicast na Internet.

MBone é uma rede virtual situada acima das sessões da Internet física. É composta de ilhas de capacidade de roteamento multicast conectadas a outras ilhas por enlaces ponto-a-ponto virtuais chamados "*túneis*". Os túneis permitem tráfego multicast através de algumas partes da Internet sem capacidade de multicast. Pacotes de IP Multicast "entunelados" são encapsulados como IP-sobre-IP[11] de modo que eles parecem pacotes IP unicast normais para os roteadores. O encapsulamento é adicionado na entrada

[11] O número do protocolo é configurado para 4.

de um túnel, e retirado na saída. Esse conjunto de roteadores multicast, suas subredes diretamente conectadas, e os túneis de interconexão compreendem o MBone.

Uma vez que o MBone e a Internet tenham diferentes topologias, roteadores multicast executam um protocolo de roteamento separado para decidir como transmitir pacotes multicast. A maioria dos roteadores MBone atualmente usam o protocolo de roteamento multicast **DVMRP** (*Distance Vector Multicast Routing Protocol*), embora algumas porções do MBone executem OSPF Multicast (**MOSPF**) ou Multicast Independente de Protocolo (**PIM**).

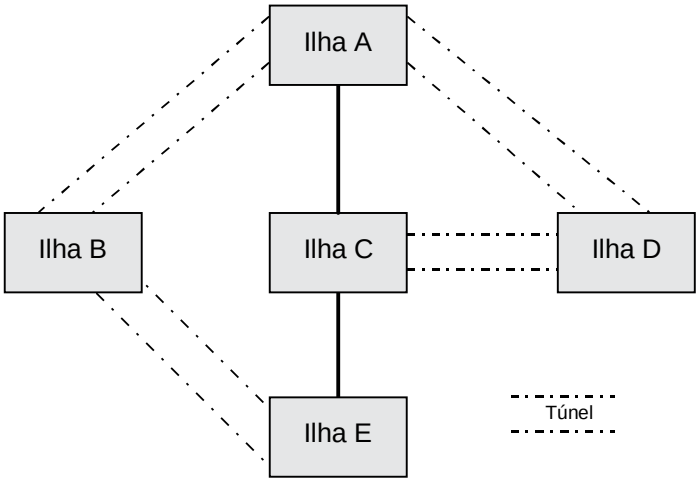

Figura 3.1: Backbone Internet Multicast

3.1.3 Suporte a IP Multicast em Redes ATM

Em redes ATM, a implementação do serviço de multicast e broadcast não é trivial. Um transmissor precisa obter os endereços de todos os receptores para o estabelecimento de uma conexão ponto-a-multiponto. Para solucionar este problema, a IETF definiu, no documento RFC 2022 [Armitage **96**], um mecanismo de registro e resolução de endereços e distribuição de informações de membros de grupo que permite oferecer suporte a um serviço de multicast sobre redes ATM baseadas nas versões UNI 3.0/3.1. O mecanismo é baseado na existência de *Servidores de Resolução de Endereços de Multicast* (MARS – Multicast Address Resolution Server), que associam endereços de grupos de multicast a interfaces ATM representando membros dos grupos. Entidades de

resolução de endereços nos sistemas finais solicitam a um MARS um conjunto de endereços ATM (que formam um grupo), no momento em que um endereço de grupo de multicast precisa ser resolvido. Os sistemas finais, por sua vez, mantêm o MARS informado de sua entrada ou saída em um grupo (o esquema de gerenciamento de grupo é mostrado adiante).

Duas abordagens podem ser utilizadas para a construção da infra-estrutura de distribuição para redes ATM: *malhas de VCs ponto-a-multiponto* ou *servidores de multicast* **(**MCSs – Multicast Servers). Na abordagem da malha de VCs, cada origem estabelece um VC ponto-a-multiponto com o conjunto de destinos que são membros do grupo com o qual ela deseja se comunicar, criando assim uma malha de VCs. Na abordagem baseada em servidores de multicast, cada origem estabelece um VC ponto-a-ponto com um nó intermediário, denominado de MCS, responsável pelo estabelecimento e gerenciamento de uma única VC ponto-a-multiponto com os membros do grupo ao qual ele gerência.

A arquitetura do MARS permite que tanto malhas de VCs quanto MCSs sejam usados simultaneamente para grupos diferentes. Apesar da especificação do MARS ser independente do protocolo utilizado, ele apresenta as mesmas características e interfaces providas pelo IGMP, implementado, porém, de modo centralizado.

3.1.4 Transporte Multicast

Os protocolos de transporte mais tradicionais como OSI-TP4 e TCP (Transmission Control Protocol) não suportam comunicação de dados multiponto. Novos protocolos como VMTP (Versatile Message Transaction Protocol) [Cherinton **88**] e XTP (Xpress Transport Protocol) oferecem suporte ao serviço de multicast. O XTP [Rezende **et al 96**] é um protocolo de transporte[12] que oferece suporte a um serviço de transmissão

[12] Alguns autores classificam o XTP como um *protocolo de transferência*, uma vez que ele incorpora funcionalidades tanto da camada de rede como da camada de transporte do modelo de referência OSI.

com confiabilidade limitada. XTP é um protocolo orientado a conexão, mas suporta também um serviço não orientado a conexão. Não há nenhum mecanismo para o estabelecimento e gerenciamento de grupos multicast.

RMTP (Reliable Multicast Transport Protocol) [Paul **et al 96**] é um protocolo de transporte multicast confiável baseado em uma estrutura hierárquica, na qual receptores são agrupados em *regiões locais* (também denominados *domínios*), nas quais um receptor especial (*receptor designado*) é responsável por enviar reconhecimentos periódicos ao transmissor, processar reconhecimentos de receptores daquele domínio, e retransmitir pacotes perdidos para os receptores apropriados. RMTP é um protocolo bastante geral, uma vez que ele pode ser construído sobre qualquer rede, seja ela orientada ou não à conexão. Apesar disso, o RMTP espera que a rede subjacente seja capaz de gerar e estabelecer uma árvore de multicast do transmissor para os receptores.

3.2 Gerenciamento de Grupo de Multicast

O termo *gerenciamento de grupo* é utilizado para descrever todas as ações relacionadas à composição do grupo, como manipulação de informações sobre os seus participantes, e o controle de acesso ao grupo, isto é, controle sobre a entrada e saída de participantes do grupo.

Os esquemas de gerenciamento de grupo podem ser classificados em dois tipos: o *gerenciamento de grupo distribuído*, no qual as informações e o controle dos grupos estão distribuídos pelo sistema de comunicação, como é o caso do IGMP [Fenner 97]; e o *gerenciamento de grupo centralizado*, no qual existe a figura de um *gerenciador de grupo centralizado*, que controla todas as atividades de gerenciamento do grupo, como acontece com o MARS [Armitage **96**].

3.2.1 Gerenciamento de Grupo Distribuído

Para participar do IP Multicast em uma rede, uma estação deve ter software que permita enviar e receber datagramas multicast. Para participar do multicast que pode espalhar-se por várias redes, a estação deve informar os roteadores de multicast de cada rede. Os roteadores por sua vez contatam outros roteadores, passando informações de grupo e estabelecimento de rotas. A idéia é bastante similar a propagação de rota local através de roteadores inter-redes convencionais.

Antes que um roteador multicast possa propagar informações dos membros de grupos de multicast, é preciso determinar que estações em sua rede fazem parte de um grupo. Para isso, roteadores multicast e estações que implementam multicast devem usar o **Protocolo de Gerenciamento de Grupo Inter-rede** (**IGMP**) [Fenner 97] para comunicar as informações de membros dos grupos.

IGMP é análogo ao ICMP. Como ICMP, ele usa datagramas IP para carregar mensagens. Também como ICMP, ele provê um serviço usado pelo IP. Portanto, embora IGMP use datagramas IP para carregar mensagens, *IGMP é considerado parte integral do IP, não um protocolo a parte*. Além disso, IGMP é um padrão para TCP/IP; isto é, ele é necessário em todas as máquinas que participam do IP Multicast no nível 2[13].

Conceitualmente, IGMP tem duas fases. Na fase 1, quando uma estação junta-se a um novo grupo de multicast (*JoinLocalGroup*), ela envia uma mensagem IGMP para o endereço de grupo "*all hosts*" declarando sua entrada no grupo[14]. Roteadores de multicast locais recebem a mensagem e estabelecem as rotas necessárias propagando as informações do membro do grupo para outros roteadores multicast através da Internet. Na

---

[13] Uma estação participa do IP Multicast de um dos três níveis possíveis. No nível 2, uma estação pode tanto transmitir como receber IP Multicast.

[14] Para cobrir a possibilidade do relatório inicial ser perdido ou corrompido, é recomendado que a mensagem seja repetida uma ou duas vezes após um pequeno retardo.

fase 2, devido à dinamicidade dos grupos, roteadores de multicast locais periodicamente questionam as estações na rede local para determinar que estações permanecem como membros de quais grupos. Cada estação mantém uma lista dos grupos de IP multicast que ele tenha aderido. Quando um roteador multicast envia uma consulta IGMP para um endereço, cada estação inicia um processo para envio de respostas para cada um dos endereços que ela esteja associada. Essas respostas são enviadas para o endereço de multicast, para que outros membros do mesmo grupo na mesma rede não precisem mais enviar respostas para esse grupo. Se nenhuma estação responder para um determinado grupo, após algumas solicitações de informação, o roteador de multicast assume que nenhuma estação na rede permanece no grupo, e para então de anunciar as informações do grupo para outras estações. Na versão 2 do IGMP [Fenner 97], foi criada a mensagem *IGMP_leave*, cuja função é a de informar sobre o desligamento de uma estação de um determinado grupo. Assim, o roteador multicast fica sabendo imediatamente se deve continuar a replicar as mensagens multicast para a rede a qual a estação que acaba de se desligar do grupo.

O esquema de gerenciamento de grupo distribuído apresenta uma grande flexibilidade no gerenciamento de membros de um grupo. No IGMP, por exemplo, esse esquema facilita a implementação de um esquema de endereçamento multicast flexível, que permite que qualquer estação envie mensagens para um grupo sem necessitar conhecer todos os participantes. Por outro lado, esse esquema torna a verificação de segurança dos membros do grupo mais complexa. Além disso, o IGMP não possui um esquema de alocação de endereços onde endereços são associados ou por uma autoridade externa, ou por cada aplicação. Isso pode levar a contenção de endereços entre várias aplicações.

3.2.2 Gerenciamento de Grupo Centralizado

O esquema de gerenciamento centralizado tem sido adaptado por vários protocolos multicast, como no PIM [Deering **et al 96**] de Deering, onde o *Rendezvous Point* (RP) é usado pelo transmissores para anunciar sua existência e pelo receptores para

conhecer os novos transmissores de um grupo. Recentemente, vários trabalhos foram propostos na implementação de protocolos multicast em redes baseadas em ATM [Armitage **96**, Guerin et al 96, Szyperski **et al 93**, Talpade **et al 96**], usando gerenciamento centralizado.

O mapeamento de mecanismos de multicast e broadcast em redes ATM não é trivial, uma vez que especificações recentes do *ATM Forum* (UNI 3.0 [ATMF **93**] e UNI 3.1 [ATMF **94**]) não provêm uma abstração flexível para endereços de multicast. A limitação reside no fato de que a estação origem deve conhecer a priori todas as estações destino, para o estabelecimento de uma VC ponto-a-multiponto.

Em redes IP convencionais, mecanismos de multicast são providos através do protocolo IGMP, como mostrado na seção anterior. Este protocolo possibilita uma abstração dos mecanismos de multicast nativos (geralmente encontrados nos níveis de enlace das LANs convencionais) para multicast inter-redes, através da utilização de um subconjunto dos endereços IP de classe D (224.0.0.0 à 239.255.255.255), reservados para grupos de multicast. A comunicação multicast IP entre estações pertencentes a redes distintas é feita através de roteadores multicast. Nas LANs convencionais, os meios de comunicação são geralmente compartilhados por todas as estações, facilitando a definição de mecanismos de multicast no nível de enlace. Por isso, o IGMP pressupõe o uso de meios de comunicação compartilhados. Como se sabe, este conceito não existe em ATM.

Para solucionar esse problema, Armitage propôs um esquema para suporte a IP multicast sobre redes ATM [Armitage **96**]. Nesse trabalho, um **Servidor de Resolução de Endereço de Multicast** (**MARS**) age como um gerenciador de registro e resolução de endereço de grupo. O conceito de MARS é uma extensão análoga ao conceito de servidor **ATMARP** (*ATM Address Resolution Protocol*) [Laubach **94**]. Ele associa endereços IP de classe D (ou no caso de qualquer outro protocolo de nível de rede, identificadores de grupos de multicast) a *interface*s ATM representando membros dos grupos. Enquanto o servidor ATMARP mantém uma tabela de pares de endereços (IP, ATM) para todos os sistemas finais IP pertencentes a uma subrede lógica IP, o

MARS mantém tabelas de endereços (IP classe D; ATM.1, ATM.2, ..., ATM.n). Um MARS pode residir em qualquer sistema final que possa ser diretamente endereçado via ATM pelos sistemas finais que requerem o serviço de multicast. Estes, por sua vez, devem ser configurados com o endereço ATM do nó onde o MARS (ao qual o agrupamento desejado está ligado) reside.

As mensagens de controle dos MARSs suportam a distribuição de informações sobre grupos de multicast IP entre outros MARSs e sistemas finais (estações ou roteadores). Entidades de resolução de endereços dos sistemas finais perguntam a um MARS (*MARS_REQUEST*) quando um endereço IP precisa ser resolvido em um conjunto de endereços ATM que formam um grupo. Os sistemas finais, por sua vez, mantém os MARSs informados da entrada ou saída deles de um grupo. Após receber informações sobre os membros de um grupo (*MARS_MULTI*), o transmissor pode estabelecer uma conexão ponto-a-multiponto com todos os membros do grupo, ou usar um servidor de multicast para distribuir os pacotes multicast.

A entrada e saída de participantes de um grupo é obtida através de mensagens *MARS_JOIN* e *MARS_LEAVE*, respectivamente. A mensagem de adesão contém o endereço de grupo de multicast e o endereço unicast do participante. Quando a mensagem é recebida pelo MARS, ele adiciona o endereço ATM especificado a entrada na tabela para o grupo especificado. A exclusão de um participante de um grupo é processada removendo-se o endereço ATM especificado da tabela para o endereço de grupo especificado. As mensagens *MARS_JOIN* e *MARS_LEAVE* são retransmitidas para todos os membros do *agrupamento* para garantir que as alterações do grupo serão refletidas em todas as conexões existentes para o grupo. Para prover mensagens de notificação de mudança na composição de um grupo a todas as estações (dentro e fora do grupo modificado), os MARSs gerenciam conexões ATM ponto-a-multiponto (*VC de Controle de Agrupamento*) com todas as estações que desejam suportar multicast, definindo *agrupamentos* de multicast.

Gerenciamento de grupo centralizado tem pontos fracos que precisam ser solucionados. Primeiro, todas as atividades de gerenciamento de grupo são conduzidas por uma única entidade de genciamento, o que pode introduzir retardos significantes de comunicação. Além disso, uma única entidade de gerenciamento se torna um ponto de falha em todo o esquema. Uma solução óbvia para esses problemas seria criar várias cópias de entidades de gerenciamento de grupo.

3.3 Roteamento Multicast

Um protocolo de roteamento multicast é responsável pela construção da infra-estrutura de distribuição multicast. Roteamento multicast é facilitado em algumas redes locais, tais como Ethernet que provê um esquema eficiente de entrega por difusão [Deering **et al 90**]. No entanto, para prover um serviço de entrega multicast, é necessário definir um mecanismo eficiente de roteamento multicast.

Várias estratégias de roteamento e implementações tem sido propostas. Dentre elas, a **Árvore Baseada na Origem** (SBT – Source Based Tree) [Deering **et al 90**, Deering et al 97, Moy 94b, Pusateri 97] e a **Árvore Baseada em um Núcleo** (CBT – Core Based Tree) [Ballardie 97a] têm recebido muita atenção em trabalhos recentes. Desenvolvimentos recentes na Internet tem enfatizado a importância de suportar ambos os tipos de árvores [Deering et al 97]. Outros tipos de esquemas de roteamento multicast são também propostos, como árvores de Steiner [Kompella et al 93].

3.3.1 Árvore Baseada na Origem

Árvores baseadas na origem (SBT) são árvores de entrega de menor caminho criadas para cada grupo, entre o transmissor e os correspondentes receptores do grupo. Para construir árvores baseadas na origem vários algoritmos tem sido propostos. Entre eles, o roteamento multicast baseado em um vetor de distância e o roteamento multicast baseado no estado do enlace tem recebido maior atenção.

O roteamento baseado em um vetor de distância tem sido usado em muitas redes como algoritmo de roteamento unicast [Comer **95**]. Roteadores que usam o algoritmo de roteamento baseado em um vetor de distância mantêm uma tabela de roteamento contendo uma entrada para cada destino alcançável na rede. Cada roteador envia periodicamente pacotes de roteamento para seus vizinhos. Ao receber um pacote de roteamento de um roteador vizinho, o roteador (receptor) pode atualizar sua tabela de roteamento se o vizinho oferece uma caminho mais curto para um dado destino, ou se o vizinho não mais oferece uma rota usada pelo roteador. Dessa forma, roteadores podem manter rotas mais curtas para todos os destinos na rede.

Dois algoritmos de roteamento multicast baseados no roteamento baseado em um vetor de distância são **Transmissão por caminho reverso** (**RPF**) e **Difusão por caminho reverso** (**RPB**). Esses algoritmos são implementados pela difusão de mensagens através da árvore de distribuição de menor caminho baseada na origem, cabendo a cada receptor selecionar os pacotes realmente destinados a eles. Para cada origem, se um pacote chegar por um enlace que o roteador acredite ser o menor caminho de volta para a origem do pacote, então o roteador transmite o pacote por todas as interfaces, exceto a interface por onde chegou o pacote. Se o pacote chegar por uma interface que não é a mais curta de volta para a origem, o pacote é descartado. Uma das principais limitações desses algoritmos é que eles não levam em consideração os membros de um grupo multicast na construção de uma árvore de distribuição para uma origem. Como resultado, datagramas podem ser desnecessariamente transmitidos para subredes que não tem membros em um grupo destino.

Uma abordagem mais sofisticada pode ser encontrada no **Protocolo de Roteamento Multicast baseado em um Vetor de Distância** (**DVMRP**), que utiliza um algoritmo RPF modificado para prover um esquema de poda por demanda da árvore de multicast de menor caminho. No DVMRP, o primeiro pacote multicast é transmitido por difusão através da árvore de difusão de menor caminho para todos os membros da inter-rede. Quando um pacote alcança um roteador no qual todos os enlaces filhos são folhas e nenhum deles possui membros do grupo destino, uma mensagem de poda (NRM – Non

Membership Report) é enviada para o roteador em direção a origem. Se um roteador receber uma mensagem de poda de todos os seus roteadores filhos, e se ele não possui nenhum membro em qualquer uma de suas subredes, a mensagem de poda é retransmitida ao seu roteador pai. Dessa forma, uma árvore de poda é criada e mensagens subsequentes para aquele grupo não mais serão enviadas por caminhos que não levam a membros do grupo. Após um período de tempo, o estado de poda para cada par (*origem, grupo*) expira para reforçar um atualização do estado de poda (de grupos que não mais permanecem ativos). Se estes grupos ainda estão em uso, um datagrama do par (*origem, grupo*) será transmitido por toda a rede através de todos os roteadores "*downstream*". Essa inundação resultará em novo conjunto de mensagens de poda, regenerando a árvore de menor caminho para esse par (*origem, grupo*).[15]

DVRP também implementa um mecanismo para rapidamente incluir de volta um ramo previamente podado de uma árvore de entrega para um grupo. Se um roteador que tenha mandado uma mensagem de poda para o par (*origem, grupo*) descobre novos membros do grupo em uma subrede folha, ele envia uma mensagem de inclusão para o roteador em direção a essa origem. Quando um roteador acima recebe uma mensagem de inclusão, ele cancela a mensagem de poda previamente recebida. Mensagens de inclusão são propagadas (de modo confiável) salto-a-salto em direção a origem, até que elas encontrem o ramo ativo mais próximo da árvore de entrega. Dessa forma, ramos previamente podados são rapidamente restaurados em uma árvore de entrega multicast.

Um outro algoritmo para a construção de árvores multicast é baseado no roteamento baseado no estado de enlace. Nesse algoritmo, todos os roteadores monitoram o estado de cada um de seus enlaces diretamente conectados. Sempre que o estado de um enlace é alterado, o roteador distribui o novo estado do enlace para todos os roteadores da rede usando um protocolo de inundação. Dessa forma, todos os roteadores recebem

---

[15] Na implementação atual do RPF, mensagens de poda são transmitidas de modo não confiável. Assim o tempo de vida da poda deve ser pequeno o suficiente para compensar a perda de mensagens de poda.

informações de topologia de todos os roteadores na rede. Baseado nessas informações de topologia, cada roteador pode calcular a árvore de espalhamento de menor caminho usando, por exemplo, o algoritmo de menor caminho de Dijkstra. OSPF (*Open Shortest Path First*) [Moy 94a] é baseado em algoritmos de estado de enlace que permitem cálculos de rotas rápidas com um mínimo de tráfego de protocolo de roteamento. O algoritmo de roteamento baseado na origem pode facilmente ser estendido para dar suporte a árvores de multicast de menor caminho. As extensões multicast ao OSPF (**MOSPF** - *Multicast Extensions to OSPF*) são definidas na RFC 1584 [Moy 94b]. MOSPF provê mecanismos para transmitir datagramas multicast de uma rede IP para outra (através de roteadores inter-redes), utilizando a base de dados de estados de enlaces OSPF. Sempre que um novo identificador de grupo é associado a um enlace (ou desassociado) o estado desse enlace é distribuído pelo seu roteador para todos os outros roteadores. Uma vez que roteadores possuem informações completas da topologia da rede, cada roteador calcula a árvore de multicast de menor caminho de qualquer origem para qualquer grupo usando o algoritmo de Dijkstra.

MOSPF herda as vantagens do OSPF, tais como cálculo de rota localizada, rápida convergência a alterações dos estados dos enlaces, etc. Entretanto, informações de membros de grupo são enviadas através da rede, incluindo enlaces que não estão no caminho direto aos destinos do grupo multicast. Assim, como DVMRP, ele é mais apropriado para inter-redes pequenas, como um mecanismo de roteamento intra-domínio, uma vez que esses algoritmos possuem uma baixa escalabilidade.

3.3.2 Árvore Baseada em um Núcleo

As mais recentes adições ao conjunto de técnicas de transmissão multicast são baseadas em uma árvore de entrega compartilhada. Diferente dos algoritmos de árvore de menor caminho, que constróem uma árvore baseada na origem para cada origem, ou cada par (*origem, grupo*), algoritmos de árvore compartilhada constróem uma única árvore de entrega que é compartilhada por todos os membros de um grupo. Estações que desejem receber tráfego para um grupo multicast devem explicitamente unir-se a árvore de

entrega compartilhada. Tráfego multicast para cada grupo é enviado e recebido pela mesma árvore de entrega, independente da origem.

No **Core Based Tree** (**CBT**) [Ballardie et al 93], um roteador ou um conjunto de roteadores são escolhidos para formarem o núcleo de distribuição de pacotes multicast, a partir do qual emanam ramos. Esses ramos são formados por outros roteadores, conhecidos como roteadores não-núcleo (*non-core*), que formam o menor caminho entre uma estação membro diretamente conectado ao roteador, e o núcleo. Um roteador na extremidade de um ramo é conhecido como roteador *folha* de uma árvore.

Duas fases distintas podem ser identificadas na entrega de dados CBT. Primeiramente, uma rota unicast é usada para encaminhar pacotes multicast para o núcleo da árvore de multicast especificada. Isso é alcançado usando-se o endereço unicast do núcleo no campo de destino de um pacote multicast. Em seguida, uma vez que o pacote multicast tenha alcançado um roteador da árvore, o endereço do núcleo no campo de endereço de destino do cabeçalho IP é descartado, e o identificador do grupo no campo de opção é colocado no campo de endereço de destino quando então, o pacote é retransmitido através de todas as interfaces que correspondem a árvore compartilhada.

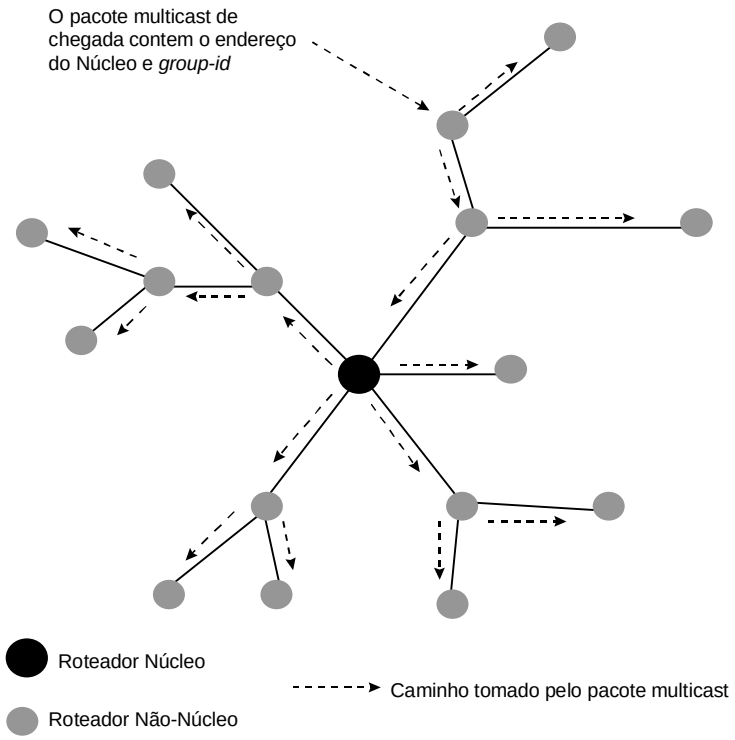

Figura 3.2: Árvore CBT

Uma vantagem do CBT é que apenas roteadores no caminho entre o núcleo e os potenciais membros do grupo são envolvidos no processo. A formação da árvore baseada em um núcleo e o fluxo de pacotes são desacoplados do roteamento unicast sobrejacente. A principal desvantagem é que pacotes não atravessam o menor caminho da origem para os seus destinos. O desempenho desse protocolo em geral depende da colocação dos núcleos e a coordenação entre eles. A concentração de tráfego nos enlaces ligados ao núcleo é um outro problema. Há também uma dependência por entidades da rede (como por exemplo, em outros domínios administrativos) para a reserva de recursos e policiamento do roteamento.

Recentemente, Deering *et. al.* desenvolveram um esquema de **Multicast Independente de Protocolo** (**PIM**) [Deering **et al 96**]. PIM recebe esse nome porque ele não depende de qualquer mecanismo provido por qualquer protocolo de roteamento unicast particular. Entretanto, qualquer implementação que suporte PIM requer a presença de um protocolo de roteamento unicast para prover informações de tabelas de roteamento e para adaptar as alterações de topologia.

PIM faz uma clara distinção entre um protocolo de roteamento multicast que é projetado para ambientes densos, e outro que é projetado para ambientes esparsos. Modo-Denso se refere a um protocolo que é otimizado para ambientes onde os membros do grupo são relativamente agrupados, e banda passante é abundante. Modo-Esparso refere-se a um protocolo otimizado para ambientes onde os membros de grupos são distribuídos através de muitas regiões e a banda passante não é necessariamente largamente disponível. É importante notar que o modo-esparso não implica que o grupo tenha poucos membros, pelo fato de ser largamente disperso através da Internet.

PIM é único no sentido de que ele suporta tanto árvores baseadas na origem (SBT), como árvore compartilhadas baseadas em um núcleo (CBT). PIM é formado por dois protocolos: PIM - Modo Denso (PIM-DM) [Deering et al 97] e PIM - Modo Esparso (PIM-SM) [Estrin et al 97]. Embora os dois protocolos compartilhem parte de seus nomes, e usem mensagens relacionadas, eles são dois protocolos completamente diferentes.

PIM-DM usa uma técnica na qual um datagrama multicast é transmitido na árvore de entrega, para todas as outras interfaces, se a interface pela qual o datagrama foi recebido é aquela usada para transmitir datagramas unicast para a origem do datagrama. PIM-DM constrói árvores acíclicas baseadas na origem (SBT). PIM-DM assume que todos os sistemas abaixo na árvore de distribuição desejam receber datagramas multicast. Se algumas áreas da rede não possuem membros do grupo, PIM-DM podará os ramos da árvore de entrega.

PIM-SM difere dos protocolos de modo denso existentes em alguns aspectos chaves.

Roteadores com membros adjacentes, ou abaixo na árvore de distribuição, devem explicitamente juntar-se a uma árvore de distribuição de modo esparso transmitindo mensagens de adesão a um grupo (*join*). Se um roteador não aderir a uma árvore de distribuição pré-definida, ele não receberá tráfego multicast endereçado para

aquele grupo. As estações devem possuir configurado o “roteador designado” (DR), que é um roteador pertencente à mesma sub-rede IP com o maior endereço IP.

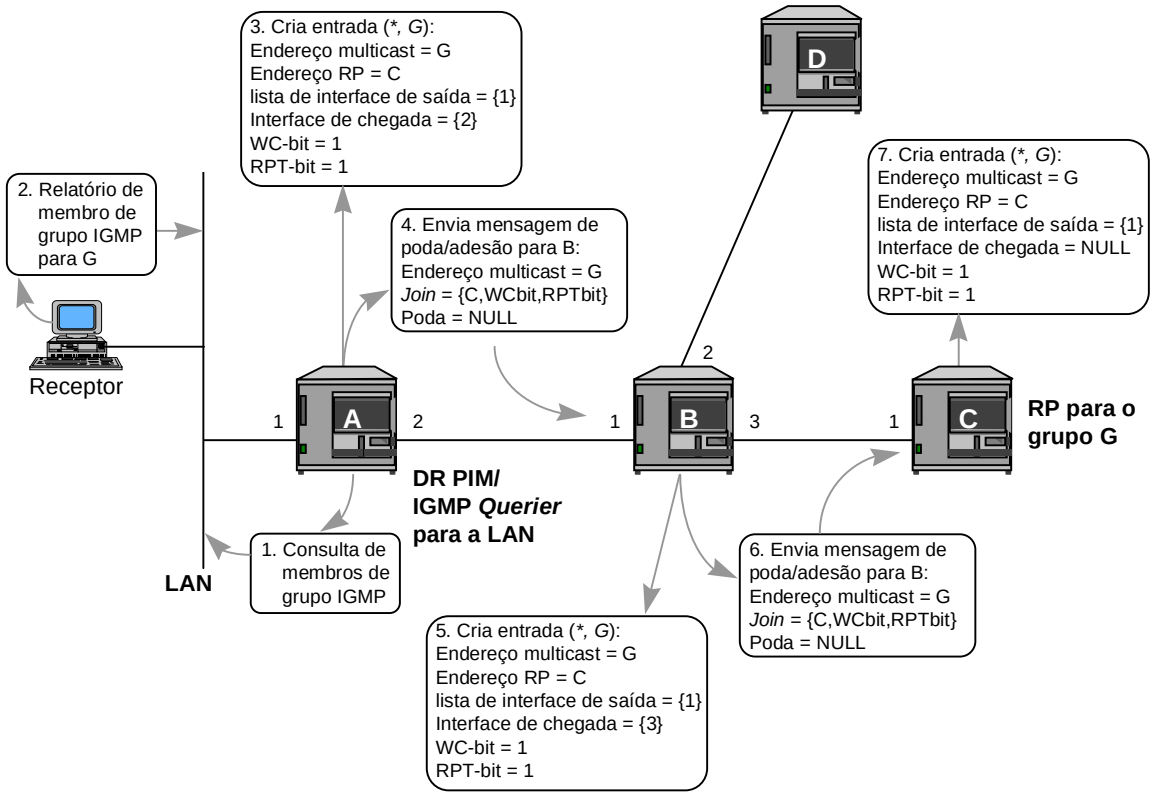

Figura 3.3: Adesão a um Grupo de multicast e construção da árvore de entrega compartilhada

Em contraste, protocolos de modo denso assumem que os roteadores a baixo na árvore de distribuição pertencem ao grupo, e transmitem tráfego multicast nos enlaces a baixo, até que mensagens de poda explícitas sejam recebidas. Assim, a ação de transmissão *default* dos protocolos de roteamento de modo denso é transmitir todo o tráfego, enquanto a ação *default* de um protocolo de modo esparso é bloquear o tráfego, a menos que ele seja explicitamente requisitado.

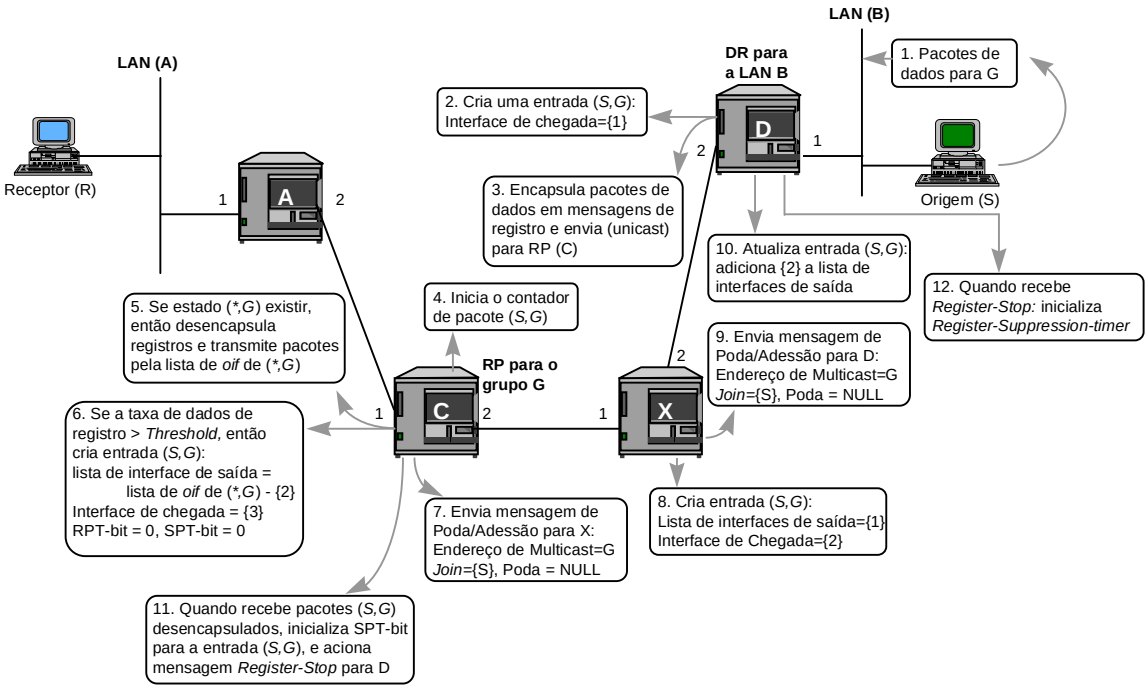

Figura 3.4: **Exemplo de transmissão a um grupo**

PIM-SM evoluiu da abordagem de Árvores Baseada em um Núcleo (CBT - *Core-Based Trees*), uma vez que ele emprega o conceito de um núcleo (ou **ponto de *rendezvous* - RP** -, na terminologia PIM-SM) onde receptores “encontram” transmissores. Apesar da árvore RP ser suficiente para a entrega de pacotes multicast, o aumento do número de participantes pode sobrecarregá-la com tráfego excessivo. Por isso, o PIM-SM fornece um procedimento alternativo utilizando árvores de menor caminho para alguns ou todos os receptores. Assim, os roteadores que implementam o PIM-SM utilizam a árvore RP, mas também têm a opção de transmitir através de árvores de menor caminho entre uma fonte e os destinatários.

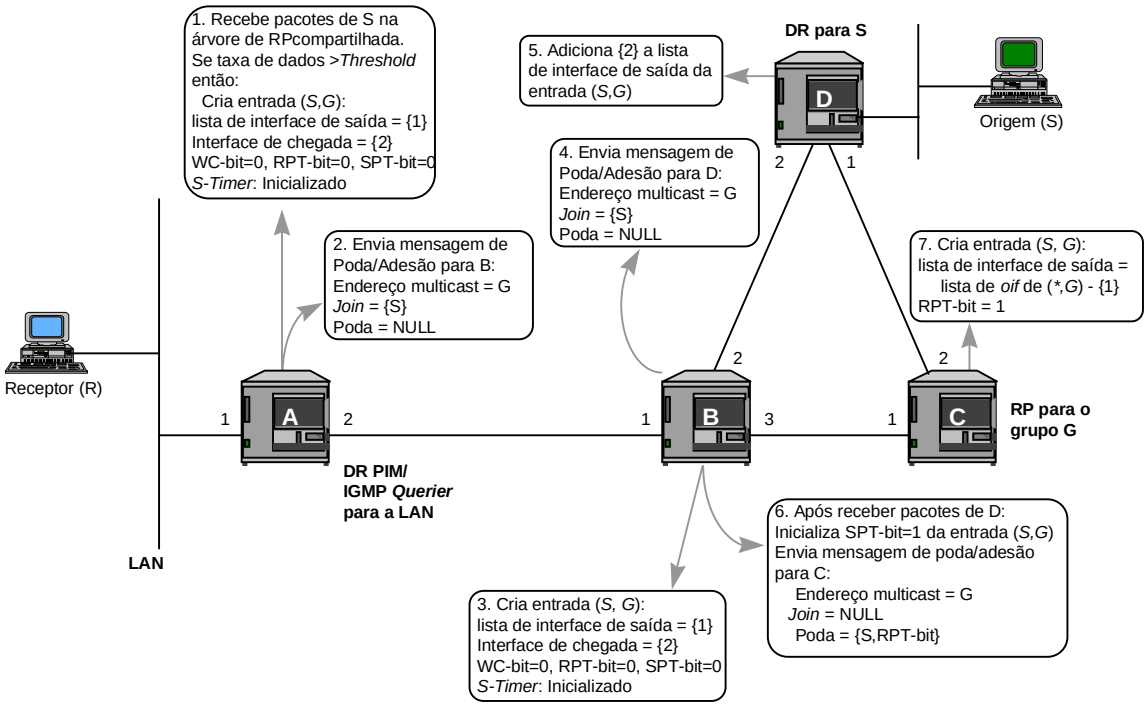

Figura 3.5: Exemplo de comutação de uma árvore compartilhada para uma árvorfe de menor caminho

3.4 Redes Configuráveis

Serviços tradicionais providos por redes de comunicação podem ser divididos em duas categorias: serviços de voz e vídeo (serviço telefônico, radiotransmissão, distribuição de vídeo) e serviços de dados (terminal remoto, correio eletrônico, transferência de arquivos), cada um com diferentes requisitos de taxa de transferência, probabilidade de perda, retardo de transmissão, e relacionamento entre participantes. Esses diferentes requisitos levam a diferentes projetos e tecnologias de redes de comunicação específicos (voltados para as necessidades específicas da aplicação). Redes de dados tradicionais transportam bits passivamente de um lado para o outro, sem se importar com o valor semânticos desses dados. O processamento em tais redes limita-se ao processamento de cabeçalho, em redes comutadas por pacotes, e sinalização em redes orientadas a conexão.

Nesse contexto, *Redes Ativas* [**Tennehouse et al 96**] representam a principal ruptura, permitindo que redes forneçam serviços especializados, configurados de acordo com requisitos específicos, através de processamento dos dados do usuário. Rede ativas introduzem uma tecnologia capaz de prover um serviço integrado aberto, não em termos de qualidade de serviço, mas também comportamento funcional. Processamento específico por pacote em cada nó da rede possibilita o desenvolvimento de novos serviço de rede. No caso extremo, pacotes convencionais são substituídos por "*capsulas*", fragmentos de programas que são executados a cada roteador/comutador da rede por onde ele passa. Arquitetura ativas permitem um grande aumento na sofisticação do processamento realizado na rede. Elas habilitarão novas aplicações, especialmente aquelas baseadas em aplicações específicas de multicast, fusão de informação, e outros serviços que influenciam o processamento e armazenamento baseado em rede. Além disso, acelerarão inovações desacoplando serviços de rede da infra-estrutura subjacente de rede, e permitindo que novos serviços sejam carregados na infra-estrutura utilizada. Por exemplo, um usuário de uma rede ativa poderia enviar uma programa de compressão especializado de acordo com suas necessidades para um nó da rede (um roteador, por exemplo) e requisitar que o nó execute aquele programa ao processar seus pacotes.

Tecnologias de rede ativa são aplicadas em sistemas finais, e acima do nível de rede fim-a-fim, por exemplo, permitindo que servidores Web e clientes troquem fragmentos de programas. A inovação de redes ativas é influenciar e estender essas tecnologias para o uso em rede, de modo que usuários possam dinamicamente configurar e programas suas redes, de acordo com os requisitos de suas aplicações.

3.4.1 ANTS – Active Network Tookit

ANTS [**Wetherall et al 98**] é uma rede ativa que pode ser reprogramada para prover serviços não previstos, quando a rede foi originalmente projetada. ANTS permite o desenvolvimento de novos protocolos tanto em nós intermediários da rede, quanto em sistemas finais, utilizando-se técnicas de *código móvel*, onde pacotes especiais, denominados *cápsulas*, carregam, além dos dados do usuário, fragmentos de programas

que devem ser executados sobre os dados transportados nos nós intermediários da rede, denominados *nós ativos*. Sua arquitetura permite visualizar a rede como um sistema de programação distribuída, e provê um modelo baseado em linguagem de programação para descrever novos protocolos em termos de operações nos nós da rede.

Diferente do IP, o serviço provido pelo ANTS não é fixo. Diferentes aplicações podem introduzir novos protocolos na rede especificando as rotinas a serem executadas nos nós da rede que retransmitem as mensagens. Aplicações podem configurar o processamento da rede de acordo com suas necessidades adicionando processamento a rede.

3.4.2 AMSA - Active Multicast Service Architecture

AMSA (Active Multicast Service Architecture) [**Li et al 97**] é um framework que suporta a configuração de um serviço de multicast, em redes ATM. Essa configuração personalizada é alcançada adicionando e invocando programas específicos de usuário a recursos de rede compartilhados, tais como roteadores e comutadores, usando conceitos e tecnologia de redes ativas. Ele permite que usuários desenvolvam serviços multicast selecionando componentes básicos de serviço de rede, adicionando programas personalizados e, finalmente, agrupando-os em um novo serviço de acordo com os requisitos específicos da aplicação.

AMSA estende o conceito de rede ativa provendo dois níveis de serviços. No primeiro nível, usuários podem configurar aplicações escolhendo certos serviços padronizados providos pelo AMSA. Por exemplo, usuários podem selecionar serviços de roteamento preferidos, enfatizando os vários requisitos de transmissão, tais como confiabilidade, retardo, ou custo. No segundo nível, AMSA propõe a adição e chamada de programas específicos de usuário aos comutadores para o processamento dos dados do usuário. Isso requer a disponibilidade e suporte de comutadores ATM programáveis. AMSA utiliza um agente mediador (*broker*) – MBA (Multicast Broker Agency) – para gerenciar e controlar o processo dinâmico de configuração de usuário do serviço de multicast. O MBA contém alguns agentes responsáveis pelo monitoramento do estado dos

enlaces ATM (MA - *Monitoring Angent*), roteamento multicast (RA - *Routing Agent*), negociação de QoS e gerenciamento de grupo (GMA - *Group Management Agent*) e gerenciamento de conexões (CMA - *Connection Management Agent*). A arquitetura proposta também suporta adesão e saída de grupos dinamicamente.

O arquitetura proposta AMSA pode ser modelada como um processo de distribuição de produtos que envolve a interação entre *Produtores* (transmissores para o grupo), *Consumidores* (receptores do grupo), *Mediadores* (MBA - responsáveis pela negociação e implementação de uma conversação multicast – criação da árvore de distribuição multicast de acordo com os requisitos de QoS exigidos pela aplicação, iniciação dos programas específicos do usuário no comutadores, e estabelecimento de conexões - entre produtores e consumidores) e *Comutadores* (responsáveis pelo transporte e entrega dos dados dos produtores aos consumidores). Para coletar informações de estado dos comutadores, ou enviar sinalização para eles, o MBA mantém um conjunto de conexões bidirecionais permanentes para todos os comutadores, onde os enlaces de chegada são utilizados pelo MA para propósitos de monitoramento, enquanto que os enlaces de saída são utilizados pelo CMA para sinalização (criação de conexões, por exemplo). O MBA é conectado aos seus clientes (estações) através de um conjunto de enlaces ponto-a-ponto bidirecionais. Essas conexões são estabelecidas pelas estações que requisitam serviços de multicast do MBA. As estações podem enviar consultas ao MBA, ou receber relatórios dele.

Diferente do modelo de rede ativa proposto por Tennenhouse [**Tennehouse et al 96**, **Wetherall et al 98**], no qual pacotes de dados de usuário são encapsulados com fragmentos de programas, que viajam através de nós ativos que o interpretam, AMSA utiliza um agente mediador (ou servidor) para o processo de configuração do usuário. AMSA encapsula apenas os manipuladores dos programas (*handlers*), enquanto dados de usuário e fragmentos de programa específicos do usuário são chamados dos switchs pelo MBA, possibilitando que programas de usuário sejam invocados apenas em pontos estratégicos da rede.

Capítulo 4

Framework Para Serviço de Multicast

Nesse capítulo, é feita inicialmente uma breve introdução a conceitos de orientação a objetos, e aplicação de patterns e frameworks no desenvolvimento e implementação de sistemas de comunicação e aplicações distribuídas. Em seguida, é apresentada, considerando os vários trabalhos apresentados no Capítulo 3, e nos requisitos descritos na Seção 2.2, uma arquitetura genérica para a provisão do serviço de multicast.

Finalmente é feita a descrição do Framework para Provisão do Serviço de Multicast. Essa descrição é feita detalhando-se os serviços e patterns componentes do serviço de multicast. Esse capítulo apresenta ainda, para cada componente do serviço, um exemplo de aplicação do framework.

4.1 Patterns e Frameworks e Conceitos de Orientação a Objetos no Projeto e Configuração de Serviços

Técnicas de orientação a objetos (OO) têm sido aplicadas com sucesso ao desenvolvimento e implementação de sistemas de comunicação e aplicações distribuídas. Diversas abordagens têm sido utilizadas durante os últimos anos, correspondendo a visões diferentes ou enfatizando aspectos diferentes de utilização da tecnologia OO. Todas essas visões são baseadas na idéia genérica de que técnicas de implementação e projeto orientado a objetos podem ser utilizadas para aumentar a modularidade e extensibilidade através da definição de interfaces estáveis, que encapsulam os detalhes da implementação. Dependendo da abordagem, os benefícios do encapsulamento podem ser vistos de modo estático ou dinâmico [**Buschmann et al 95**].

A visão estática é limitada à *fase de construção*, anterior à provisão do serviço. Sua principal motivação é baseada no reconhecimento de que serviços de comunicação são caracterizados como produtos de rápida e constante evolução. Já a visão dinâmica é relacionada à provisão do serviço e fases de execução, e é usualmente empregada aliada à abordagem da visão estática. Abordagens dinâmicas são baseadas no uso de modelos OO , ambientes de objetos distribuídos (DOC) e plataformas baseadas em componentes (CORBA, Java Beans, ODP e TINA DPE) que fornecem suporte a configuração de sistemas em tempo de execução de softwares e serviços de comunicação. Configuração de sistemas de tempo real não é apenas importante para lidar com evolução constante de sistemas, mas também para prover soluções escaláveis para sistemas que apresentam uma larga variedade de requisitos determinados pela aplicação.

Durante o desenvolvimento de quaisquer sistemas, projetistas tendem a se valer de decisões de projeto tomadas anteriormente em outros sistemas [**Buschmann et al 95**]. Um *pattern* pode ser visto como uma forma de documentar conjuntos de regras que descrevem decisões de projeto recorrentes em vários sistemas, facilitando assim a reutilização de soluções já existentes [**Gamma et al 95**]. Patterns podem cobrir várias

escalas de distribuição e níveis de abstração presentes em um sistema, dependendo de quão genéricas são as descrições desses patterns, podendo variar desde diretrizes a respeito da organização de um sistema (decisões arquiteturais) até práticas de codificação da implementação desse sistema. Buschmann classifica patterns em três principais categorias: *patterns arquiteturais*, que descrevem esquemas de organização estutural ou arquitetural; *design patterns*, que fornecem um esquema para a descrição de subsistemas específicos ou componentes de um sistema, descrevendo interfaces, relacionamentos e padrões de colaboração entre as classes, componentes ou objetos; e *idiomas*, patterns de baixo nível, que especificam linguagens de programação.

Os conceitos oferecidos pela tecnologia OO também têm o potencial de aumentar significativamente a reutilização de soluções existentes, se comparada a outras abordagens de construção de sistemas. Esses conceitos são especialmente úteis na descrição de *frameworks*. Frameworks são definidos como arquiteturas semi-completas reutilizáveis que podem ser especializadas para produzir aplicações configuradas de acordo com requisitos particulares [**Pree 95**]. Frameworks permitem que não só os componentes de um sistema, mas também as decisões de projeto a ele associadas, sejam reutilizados. Neste sentido, o potencial da abordagem de patterns permite a descrição de frameworks de forma mais abstrata do que o código correspondente que os implementaria. Designs patterns e frameworks tem sido aplicados em conjunto para melhorar a qualidade de software de comunicação, como apresentado no framework ACE [**Schimidt 97**].

Em geral, frameworks padronizam sistemas para um domínio específico [**Gamma et al 95**]. Usualmente, vários aspectos de um framework relativos a esse domínio não podem ser antecipados. Essas “partes” do framework, denominadas em [**Pree 95**] de *pontos de flexibilização* (*hot spots*), devem ser suficiente genéricas para que possam ser adaptadas a cada caso de uso. Patterns mostram-se como ferramentas poderosas no delineamento desses pontos de flexibilização durante o processo de definição de um framework.

4.2 Arquitetura Genérica do Framework

De uma forma genérica, um serviço de multicast está relacionado a algum serviço de comunicação, explícito ou implícito, entre entidades que podem ser objetos, processos, entidades de protocolos, etc. Assim, o referido ambiente pode abranger sistemas de processamento e comunicação representados por camadas de protocolos, sistemas operacionais, plataformas de componentes OO (como CORBA, COM ou Java Beans), entre outros. O serviço de multicast compõe a parte específica de um serviço de comunicação responsável pela construção de uma infra-estrutura de distribuição e o gerenciamento dos grupos. O serviço não trata da transmissão propriamente dita. O serviço geral de comunicação de grupo com qualidade de serviço pode ser composto por três serviço: um serviço multicast, descrito pelo framework proposto nessa dissertação; um serviço para provisão de qualidade de serviço, modelado pelo framework descrito em [**Gomes et al 99**]; e um serviço de transporte, que trata da transmissão (seja ela multicast ou unicast) utilizando serviços providos pelos dois outros frameworks.

Para que a interface de um serviço de multicast derivado do framework proposto seja genérica, o framework é definido de forma a apresentar as seguintes características: (i) a possibilidade de definição de um *endereçamento de grupo*; (ii) a possibilidade de que o serviço seja *orientado* ou *não orientado à conexão*; (iii) o oferecimento de uma "topologia virtual" *ponto-a-multiponto* para a comunicação de seus usuários; e (iv) a possibilidade do gerenciamento de grupo ser *centralizado* ou *distribuído*.

O suporte ao serviço multicast é baseado nos conceitos de *agrupamento* e *grupo*. Agrupamento é definido como um conjunto de participantes que desejam fazer uso do serviço de comunicação multicast. A comunicação de dados no interior de um agrupamento é sempre realizada entre um membro e outro, ou entre um membro e um

grupo de usuários. Cada grupo de usuários de um agrupamento detém um endereço de grupo.[16]

A interface genérica para um serviço de multicast define um conjunto de primitivas nas quais um endereço de grupo é fornecido. Cada endereço de grupo ou *endereço multicast* identifica um conjunto de endereços, cada um correspondendo a um dos membros do grupo. O transmissor não necessita conhecer os membros do grupo nem seus endereços particulares. Além disso, o transmissor não é necessariamente membro do grupo. Um grupo deste tipo é denominado um *grupo aberto* em contraste ao *grupo fechado*, no qual apenas os próprios membros podem transmitir para o grupo [**Deering et al 90**].

O serviço de multicast previsto pelo framework é ponto-a-multiponto considerando que os outros dois arranjos, *ponto-a-ponto* e *difusão*,[17] podem ser tratados como casos particulares onde o destino é formado respectivamente por: apenas um participante (topologia virtual ponto-a-ponto), ou todos os participantes do sistema de comunicação (topologia virtual em barra ou por difusão).

A implementação de um serviço de multicast em um determinado nível depende do suporte dado pelo nível inferior, que pode oferecer ou não suporte ao endereçamento de grupo. Além disso, o serviço de comunicação do nível inferior pode ser ou não orientado à conexão e pode possuir capacidade de transmissão por difusão, ponto-a-ponto, ou ponto-a-multiponto. Assim, algumas adaptações serão necessárias para a provisão de um serviço específico dado um determinado sistema de comunicação do nível inferior. A modelagem do framework é genérica o suficiente de modo a permitir qualquer configuração.

---

[16] Casos particulares desses grupos são aqueles compostos por um só participante e aquele formado por todos os participantes do agrupamento.

[17] Nesse trabalho, não foi considerada comunicação multiponto-a-multiponto.

A estrutura do serviço, contendo seus principais componentes e interfaces, é apresentada na Figura 4.1.

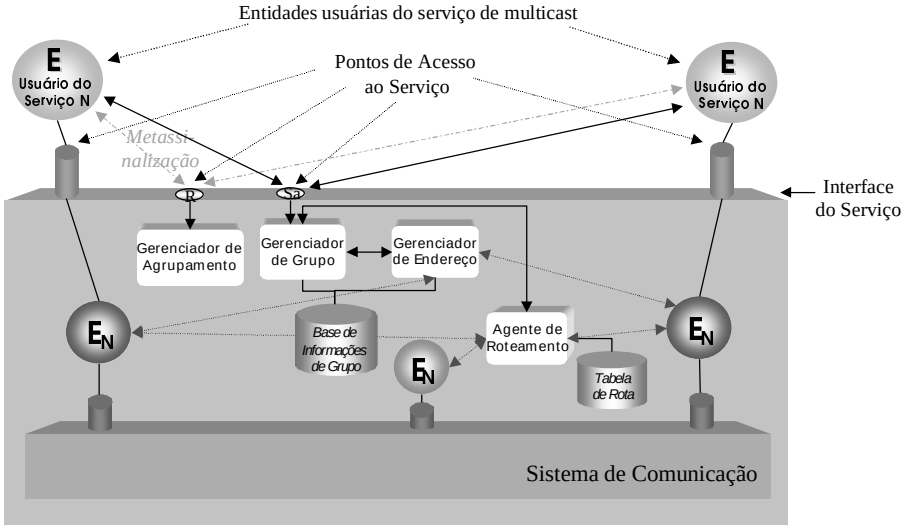

Figura 4.1: Arquitetura Geral do Serviço de Multicast

Considerou-se, a partir da análise dos vários trabalhos apresentados no capítulo de trabalhos relacionados, que a arquitetura genérica de um serviço de multicast pode ser dividida em duas partes: o *gerenciamento de grupo*, composta por três componentes principais (*gerenciador de agrupamento*, *gerenciador de grupo* e *gerenciador de endereço*), e a *construção de uma infra-estrutura de distribuição* (agente de suporte ao roteamento), que serão detalhados nas próximas subseções.

A comunicação é realizada por sistemas de comunicação de acesso múltiplo ou por sistemas formados por nós interligados através de enlaces ponto-a-ponto, interconectadas em uma topologia arbitrária por nós de comutação.

4.2.1 Interface do Serviço

A interface básica para um serviço de multicast genérico é feita através da definição de primitivas, em que é especificado um endereço de grupo com o qual se deseja iniciar uma comunicação. Cada endereço multicast identifica um *grupo de pontos de*

acesso ao sistema de comunicação[18], que deve receber dados enviados para aquele endereço. O transmissor não necessita conhecer os membros do grupo, nem seus endereços, e não necessita também, pertencer ao grupo. Este grupo é referenciado como *aberto*, como definido anteriormente. É importante salientar também que o endereço do grupo deve ser único, padronizado e genérico, de tal forma a acomodar esquemas de endereçamento atuais e futuros.

A interface do serviço não impõe restrições sobre número ou localização das entidades em um grupo. Entidades podem aderir ou abandonar o grupo, sem necessidade de sincronização ou negociação com outros membros (do grupo) ou com transmissores potenciais. Uma entidade pode pertencer a mais de um grupo.

Os pontos de acesso ao serviço podem ser classificados em três tipos distintos. O primeiro deles diz respeito à metassinalização[19] necessária para que entidades de aplicação possam registrar-se como usuárias do serviço de multicast. Esse ponto de acesso é denominado de *Manipulador de Registro* (R) e pode ser encontrado através de um endereço bem conhecido. O registro junto a serviço de multicast é alcançado através da primitiva *ContainerJoin*.

O segundo tipo de pontos de acesso engloba aqueles utilizados para a sinalização do serviço de multicast propriamente dita. Ao registrar-se, entidades de aplicação recebem e passam a utilizar estes pontos de acesso exclusivamente para sinalização. Os pontos de acesos de sinalização são denominados de *Manipuladores de Sinalização* (Sa). Cada entidade de aplicação registrada no serviço de multicast possui um manipulador de sinalização exclusivo, e passa a *usuária* do serviço. Todas as primitivas

---

[18] Ponto de acesso ao sistema de comunicação será usado nesse trabalho como referência a qualquer identificador de um membro de grupo, quer seja ele um objeto, um processo, uma estação, ou uma interface física de acesso ao sistema de comunicação.

[19] Procedimento utilizado par o estabelecimento de chamadas de sinalização.

das entidades usuárias do serviço, com exceção de *containerJoin* e primitivas de transmissão são tratadas no manipulador de sinalização.

A terceira categoria de pontos de acesso engloba os *Manipuladores de Serviço* (Sv). A tarefa dos manipuladores de serviço é prover as entidades de aplicação o serviço de comunicação de dados propriamente dito. O serviço de transmissão multicast não será tratado pelo framework multicast, uma vez que a transmissão deve levar em conta requisitos de qualidade de serviço, tratado em um outro framework.

4.2.2 Gerenciamento de Grupo e Agrupamento

Os esquemas de gerenciamento de grupo podem ser classificados em dois tipos: o *gerenciamento de grupo distribuído*, no qual as informações e o controle dos grupos estão distribuídos pelo sistema de comunicação, como é o caso do IGMP [Fenner 97]; e *gerenciamento de grupo centralizado*, no qual existe a figura de um *gerenciador de grupo centralizado*, que controla todas as atividades de gerenciamento do grupo, como acontece com o MARS [Armitage **96**].

Para que um sistema possa prover um serviço de multicast é necessário, primeiramente, a criação da infra-estrutura de gerenciamento, que implica na criação de agrupamento e na geração de um *identificador* para esse agrupamento. Adicionalmente, é criado o grupo *all-users*, que conterá todos os participantes do agrupamento em um determinado instante.

Após criada a infra-estrutura de gerenciamento, usuários podem registrar-se junto ao serviço de multicast para fazer uso de sua infra-estrutura. O registro é feito através da entrada no agrupamento (através da primitiva *containerJoin*), momento em que um usuário é automaticamente inserido no grupo all-users, e em um grupo onde apenas ele será elemento. Os endereços desses grupos de apenas um participante definem os *identificadores de usuário*. A saída do agrupamento é alcançado através da primitiva *containerLeave*.

Pertencendo ao agrupamento, usuários podem criar e destruir grupos, ou juntar-se e abandonar grupos existentes. A criação de grupo é feita através da primitiva *GroupCreate*. Ao solicitar a criação de um grupo, um usuário pode alternativamente: (i) especificar o identificador de grupo *0*, caso em que um novo identificador de grupo disponível será alocado; (ii) especificar um identificador em particular para um novo grupo a ser criado. A extinção de um grupo é feita através da primitiva *GroupDestroy*. A adesão a um grupo é alcançada através da primitiva *GroupJoin*, passando o identificador do grupo, e o identificador do usuário ao provedor do serviço. A primitiva *GroupLeave* permite a um usuário solicitar sua saída de um grupo.

O diagrama de estados do esquema de gerenciamento de grupo é ilustrado na Figura 4.2. Considerando-se que o agrupamento já esteja criado no estado inicial (*start*), temos duas fases no diagrama de estados: (i) registro, onde o usuário adere ao agrupamento, e (ii) criação de um grupo ou adesão a um grupo já existente.

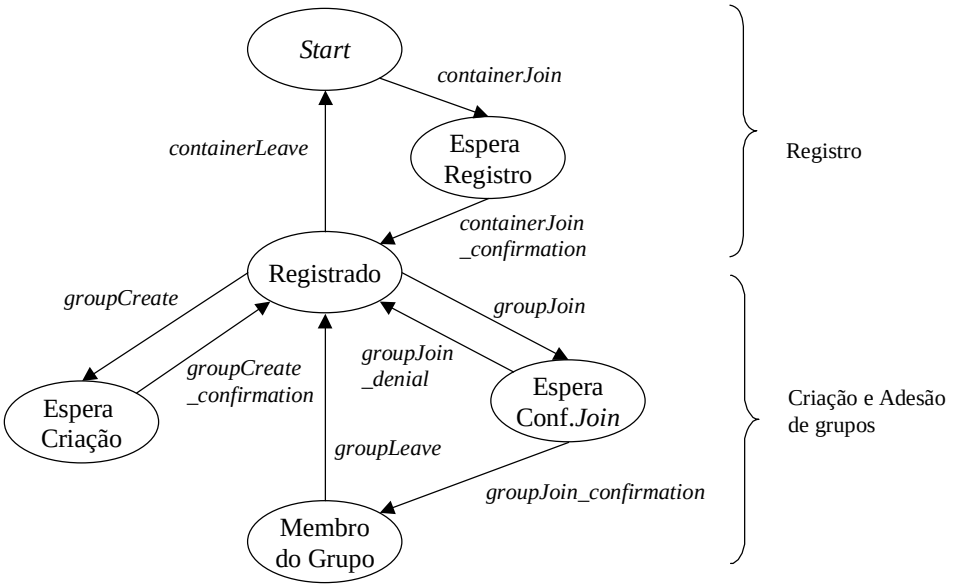

Figura 4.2: Diagrama de Estados do Gerenciamento de Grupo

As operações possíveis de serem realizadas com os grupos são a criação, destruição, entrada, saída e a obtenção de informações sobre os mesmos. As primitivas referentes a essas operações estão apresentadas na Tabela 4-1.

Tabela 4-1: Primitivas Referentes aos Serviços de Manipulação de Grupos

Primitiva	Parâmetros
containerJoin	Identificador do agrupamento, Id. do usuário, Parâmetros do agrupamento
containerLeave	Identificador do agrupamento, Id. do usuário
setContainerParameters	Lista de parâmetros e seus valores
getContainerParameters	Lista de parâmetros
groupCreate	Identificador do grupo, Parâmetros do grupo
groupJoin	Identificador do grupo, Identificador do usuário
groupLeave	Identificador do grupo, Identificador do usuário
groupDestroy	Identificador do grupo
getGroupInformation	Identificador do grupo, Identificador do usuário, lista de parâmetros

4.2.3 Gerenciamento de Endereço

Iniciar um grupo de multicast exige primeiro a associação de um endereço de grupo multicast. A partir de então, todo o tráfego multicast é entregue a esse endereço, o que implica que todos os participantes do grupo devem esperar por dados enviados para esse endereço.

A atribuição de endereços de grupos deveria ser feita de modo independente da aplicação, isto é, deveria ser possível uma atribuição dinâmica de endereços de grupos a aplicações que desejassem usufruir de um serviço de multicast.

Outras funções importantes, relacionadas ao gerenciamento de endereços são a resolução de endereço entre níveis adjacentes de uma arquitetura de sistema de comunicação, e o mapeamento de endereços de grupos nos endereços dos respectivos pontos de acesso ao sistema de comunicação que fazem parte do grupo. Endereços de grupos devem ser mapeados aos endereços reais dos nós participantes de um grupo. Esse mapeamento, bem como a manutenção das informações necessárias a esse mapeamento é de responsabilidade do *gerenciador de endereços* do serviço de multicast.

A resolução de endereços pode ser feita através do *mapeamento direto* ou *vinculação dinâmica* [**Soares et al 95**]. No mapeamento direto uma entidade sabe como realizar o mapeamento através de funções ou tabelas de conversão. Para evitar o uso de funções ou tabelas de mapeamento, um endereço de grupo também pode ser vinculado

dinamicamente a um outro endereço de grupo do nível inferior, ou a vários endereços individuais dos membros pertencentes ao grupo, através de algum protocolo de resolução [**Soares et al 95**]. A resolução de endereço pode ser distribuída ou centralizada. No contexto da solução centralizada enquadram-se os *servidores de resolução de multicast*. Servidores de resolução são entidades responsáveis pelo mapeamento de um endereço de grupo aos vários endereços dos membros do grupo, através de requisições ditadas por algum protocolo de resolução.

Para tornar a definição do serviço genérica o suficiente, deve-se utilizar uma definição abstrata de endereço, de tal sorte que permita sua adaptação a qualquer esquema utilizado nos sistemas de comunicação. Define-se simplesmente que um endereço pode ser um endereço simples, ou um endereço composto, que por sua vez pode ser um endereço simples ou um aninhamento de endereços compostos.

4.2.4 Construção da Infra-Estrutura de Transmissão Multicast

A função de roteamento em um sistema de comunicação é responsável por, dado um endereço de destino, escolher uma rota a ser utilizada para o encaminhamento do tráfego a ela destinado. O roteamento nem sempre é realizado de forma atômica, podendo estar espalhado ao longo do tempo pelas diversas fases da operação de um sistema de comunicação. Em sistemas com serviço orientado a conexão implementado através de circuito virtual, por exemplo, rotas são estabelecidas a priori, no momento do estabelecimento da conexão, de forma separada do encaminhamento das mensagens propriamente dito. Em outros tipos de serviço, rotas são estabelecidas dinamicamente conforme cada mensagem é enviada ao longo do caminho. Os algoritmos utilizados para escolha de rotas também são os mais variados, como já apresentado no Capítulo 3, podendo levar em consideração diversas métricas para a escolha do caminha mais adequado como, por exemplo, a QoS [Kompella et al 93].

Independente dos vários algoritmos e escalas de tempo nas quais eles operam, para que uma função de roteamento possa funcionar de forma adequada, é necessária a criação de uma infra-estrutura de transmissão, i.e., os elementos do sistema

que cooperam para a escolha de rotas devem ser alimentados de informações de alcançabilidade e disponibilidade de recursos que sirvam de base para a tomada das decisões de roteamento.

A parte do framework de multicast responsável pela criação da infra-estrutura de transmissão cuida do fornecimento e distribuição das informações de alcançabilidade dos componentes endereçáveis do sistema. O formato e a maneira com que essas informações devem ser distribuídas depende do tipo de infra-estrutura que o ambiente ou sistema de comunicação inferior fornece. Em ambientes cujo suporte à transmissão é ponto-a-ponto, por exemplo, serviço de multicast pode ser implementado através do envio de várias cópias da mensagem para os vários destinos, sem o conhecimento do usuário ("*Multicast by unicast*"). Outra abordagem, utilizada quando a infra-estrutura provê suporte à transmissão por difusão, é o envio de uma única cópia de cada mensagem endereçada ao grupo para todos os usuários, cabendo a cada receptor aceitar apenas as mensagens direcionadas ao grupo que ele pertence ("*Multicast by broadcast*"). Finalmente, quando a infra-estrutura oferece suporte direto à comunicação ponto-a-multiponto, o serviço multicast pode se utilizar dessa capacidade, e a própria infra-estrutura encarrega-se de fazer as devidas replicações, evitando que cópias de uma mesma mensagem percorram um mesmo caminho mais de uma vez ("*Multicast by multicast*"). Essa última estratégia é também comumente associada à existência de uma topologia virtual em árvore.

A parte do framework responsável pelo gerenciamento de grupo está diretamente relacionada à criação da infra-estrutura de transmissão, uma vez que, de acordo com a entrada ou saída de participantes de um grupo, as informações sobre rotas devem ser modificadas para refletir a nova estrutura de distribuição. Por conseguinte, o framework de transporte (e também o de QoS) têm os respectivos comportamentos influenciados, já que alterações nos participantes de um grupo podem demandar o estabelecimento de novas conexões ou a adição de novos participantes em conexões já existentes.

4.3 Descrição do Framework

A definição do serviço através de um framework segue uma abordagem adotada em alguns trabalhos recentes [**Tennehouse et al 96**, **Tennehouse et al 96**, **Tennehouse et al 96**, **Colcher et al 98**], na qual um serviço pode ser configurado de acordo com as necessidades do usuário, ou do ambiente disponível. O framework aqui proposto, conforme repetidamente mencionado, pode ser aplicado independentemente da escala de distribuição de seus usuários (em um mesmo processo, em processos distintos de uma mesma máquina, etc.), assim como do tipo de sistema de comunicação utilizado para a troca de mensagens.

O framework, conforme também já mencionado, é dividido em duas partes: (i) **Serviço de Gerenciamento de Grupo** (*Group Management Service*), responsável por manter uma base de dados de informações sobre grupos (Group Information Base) e suas composições, e utilizando-se dessas informações, dar suporte, por exemplo, ao mapeamento de endereços de grupos aos endereços de seus participantes; e (ii) **Serviço de Suporte ao Roteamento** (*Routing Support Service*), responsável por criar e gerenciar a infra-estrutura de distribuição multicast. Os serviços de gerenciamento de grupo e suporte ao roteamento funcionam como meta-arquiteturas, uma vez que eles esboçam os principais mecanismos de provisão do serviço de multicast (e os pontos de flexibilização associados) a serem modelados durante o processo de definição da arquiteturas de gerência de grupo e suporte ao roteamento multicast.

A arquitetura do framework é apresentada através da utilização da *Linguagem de Modelagem Unificada* (UML - Unified Modeling Language) [**Booch et al 96**], que provê uma combinação de semântica, sintaxe, notações e um meta-modelo para o projeto de sistemas orientados a objeto. Esta linguagem prevê a utilização de múltiplas, diferentes e concorrentes visualizações para a arquitetura de um sistema, cada uma delas se concentrando em uma abstração específica. Duas destas visualizações serão utilizadas na descrição do framework: a visualização de *Casos de Utilização* (Use Case) e a *Visualização Lógica*, utilizada para a descrição dos serviços providos pelo framework.

Particularmente neste trabalho, foi adotada uma notação diferenciada para as classes de objetos que compõem a estrutura básica dos elementos do framework (classes hachuradas), e as que simbolizam possíveis especializações dos pontos de flexibilização desses elementos (classes brancas). Nesta dissertação, a descrição do framework leva em consideração apenas os relacionamentos de dependência, especialização, agrupamento, e associação entre componentes associados a provisão do serviço de multicast. Outros tipos de relacionamentos, como os de concorrência e sincronização, embora importantes em uma descrição mais formal, não serão abordados.

Maiores detalhes sobre a metodologia e a notação podem ser encontrados no APÊNDICE A, e em [**Booch et al 96**, **Quatrani 98**, **Rational 97**].

4.3.1 Casos de Utilização

A visualização de Casos de Utilização [**Booch et al 96**] descreve o serviço de multicast como visto do ponto de vista externo, ou seja, focalizando as transações específicas realizadas pelo serviço para atender às necessidades de seus usuários. O refinamento dos casos de utilização leva aos Diagramas de Seqüência e Colaboração, que definem as colaborações entre os objetos do modelo, tendo o eixo do tempo como referência, para a implementação das soluções aos casos de utilização.

Os casos de utilização do serviço de multicast estão apresentados na Figura 4.3. Os atores (entidades externas) são os usuários do serviço (entidades de aplicação), e outros serviços, como por exemplo, um serviço de transmissão, que utiliza-se do serviço de resolução de endereço para iniciar transmissão para um grupo de usuários, ou do suporte a construção da infra-estrutura de transmissão multicast fornecido pelo framework.

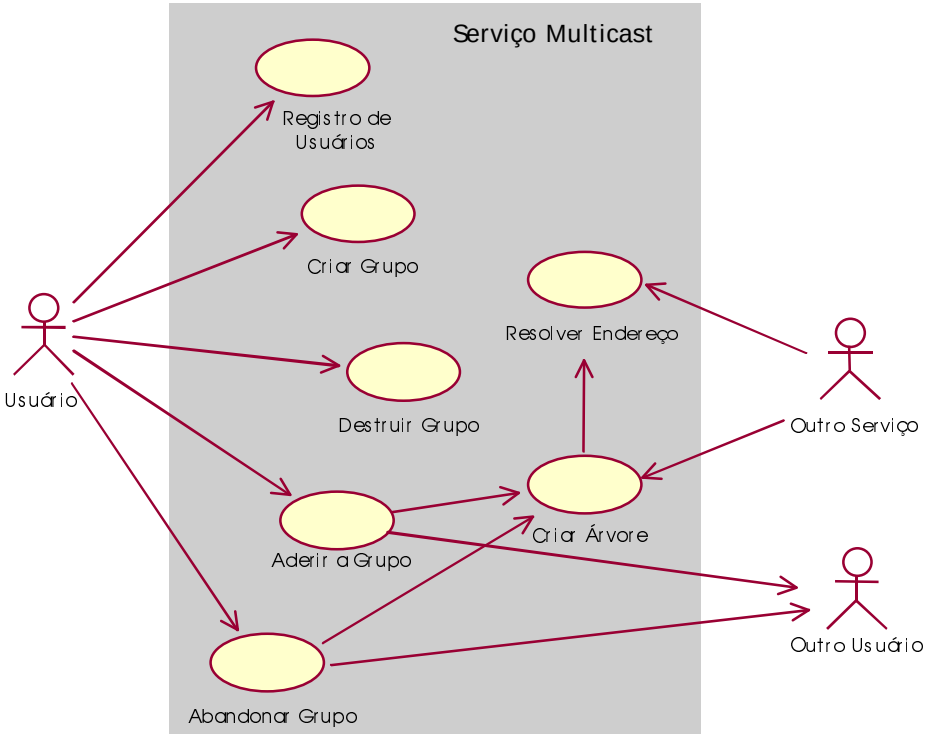

Figura 4.3: Casos de Utilização do Serviço de Multicast

Os casos de utilização representam os principais serviços providos pelo framework de serviço multicast, explicados adiante.

4.4 Serviço de Gerenciamento de Grupo

A parte do framework responsável pelo serviço de gerenciamento de grupos oferece os mecanismos para a manter a base de informações dos grupos e seus componentes, além das funções e informações necessárias para resolução de endereços de grupo utilizadas pelos mecanismos de transmissão. A resolução de endereços é responsável por dar suporte ao mapeamento entre um endereço de nível *N* para um ou mais endereços de nível *N-1*. A maneira com que as informações de grupo estão distribuídas e a forma de resolução de endereços são dois aspectos intimamente relacionados.

4.4.1 Componentes do Serviço de Gerenciamento de Grupo

A partir das análises sobre as possíveis formas de resolução e gerenciamento, que englobam inclusive as formas de resolução utilizadas para endereços unicast, chegou-se a uma modelagem genérica que constitui a parte do framework relativa ao gerenciamento de grupo, ilustrada na Figura 4.4.

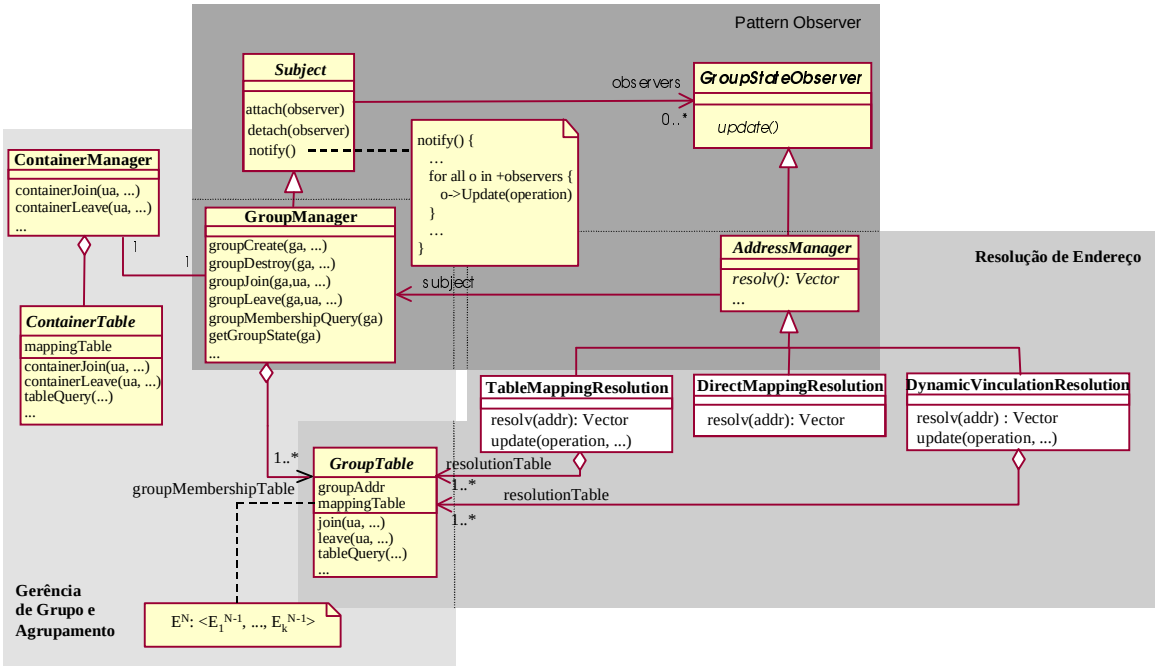

Figura 4.4: Modelo de Gerenciamento de Grupo

O serviço de gerenciamento de grupo é composto por um pattern para manutenção de informações de grupo, um modelo para gerência de grupos e agrupamento, e um modelo para gerência e resolução de endereços de multicast.

4.4.1.1 Base de Informações de Grupo

O gerenciamento de grupo gira em torno da manutenção de uma base de informação de grupo. Essa base é composta por duas tabelas principais, que relacionam endereços de um nível *N*, sejam eles de grupo ou não, a endereços do nível *N-1*. Por apresentar uma estrutura única, essas tabelas são implementadas a partir de uma classe comum (*GroupTable*), a partir da qual são criadas uma instância para o gerenciamento de

grupo, que mapeia um grupo a todos os seus participantes, e outra, para a resolução de endereço, que mapeia um endereço de grupo do nível N a um (ou mais) endereço de grupo do nível N-1, ou aos vários endereços N-1 dos participantes do grupo.

A manutenção de consistência entre as tabelas que compõem a base de dados do grupo é alcançado através do pattern *Observer* [**Gamma et al 95**]. Esse pattern define uma dependência entre objetos de modo que quando um objeto altera seu estado, todos os outros objetos "dependentes" são notificados e atualizados automaticamente. Isso garante que sempre que mudanças na composição de grupos ocorrem, elas são imediatamente refletidas nas tabelas de resolução de endereços, se isso for necessário. Isso é particularmente interessante no modelo de gerenciamento de grupo centralizado, onde sempre que um novo membro adere a um grupo (ou o abandona), a tabela de resolução de endereços deve ser modificada para refletir a atual composição do grupo. Os objetos chaves desse pattern são *Subject*, que provê uma interface para a associação (*attach(observer)*) e exclusão (*detach(observer)*) de objetos observadores, além da notificação de alterações (*notify()*), e *GroupStateObserver*, que define uma interface de atualização para objetos que devem ser notificados sobre alterações no grupo.

Um *Subject* pode ter um número de observadores dependentes. Todos os observadores são notificados sempre que há alguma alteração na composição do grupo. A notificação é implementada pelo método *notify()*, que dispara o procedimento de atualização (*update(operation)*) em todos os observadores dependentes. Em resposta, cada observador enviará uma requisição de consulta ao *Subject*, através do método *getGroup()*, para a sincronização de seus estados. A Figura 4.5 ilustra o diagrama de colaboração do pattern *Observer*, no caso de uma adesão ao grupo.

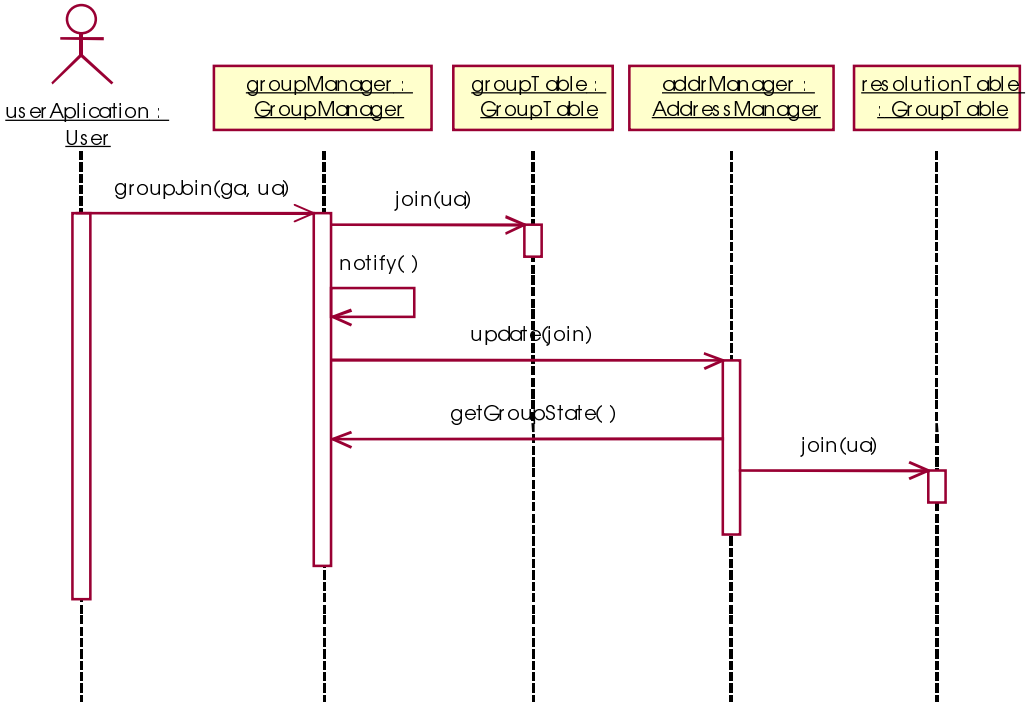

Figura 4.5: Diagrama de Colaboração do pattern Observer para a adesão a grupo

4.4.1.2 Modelo de Gerenciamento de Grupo

O principal componente do serviço de gerenciamento de grupo é o *GroupManager*, onde são definidas as operações básicas necessárias para a constituição e manutenção de um grupo (*groupCreate*, *groupDestroy*, *groupJoin*, e *groupLeave*). Essas operações possuem uma estrutura única, tanto para o modelo de gerenciamento centralizado como distribuído. Todas essas operações são formadas, basicamente, por uma atualização da base de dados local do grupo e por uma notificação da alteração a alguma entidade de gerenciamento remota. O único ponto de configuração dessa classe está exatamente na definição do endereço utilizado para a notificação das alterações, obtido através da função *getReportAddr()*. Assim a estrutura é a mesma para qualquer abordagem de implementação.

Quando um usuário do serviço solicita uma operação, a notificação será feita através do envio de uma mensagem especial, determinada pelo tipo de operação (criação de grupo, destruição de grupo, adesão a grupo, ou saída de grupo) para um endereço configurado de acordo com o modelo do serviço. No modelo centralizado,

adotado pelo MARS, essa notificação será enviada para o servidor de gerenciamento de grupo (MARS). No servidor, a base de dados centralizada dos grupos é alterada. Mais uma vez, o esquema de notificação é utilizado. Desta vez, a notificação é enviada para todos os participantes do agrupamento (all-users), ou para o servidor de multicast (MCS), para que a operação seja refletida nas conexões existentes para os membros do grupo (como por exemplo, para que um usuário ao aderir ao grupo, seja inserido em todas as conexões ponto-a-multiponto existentes para o grupo). No modelo distribuído, a notificação do usuário será enviada para o endereço do grupo, para que outros membros do grupo tomem conhecimento da alteração sofrida no grupo. Esse é o caso implementado pelo IGMP. No IGMP essas notificações (relatórios) são enviados em resposta a consultas feitas pelo roteador da rede. A mensagem é enviada para o grupo, de modo que outros membros do mesmo grupo na mesma rede possam receber o relatório, e assim não precisem responder a requisição feita pelo roteador.

Em ambas as abordagens, as informações de notificação serão também utilizadas para a atualização das informações de resolução de endereço, tanto nos clientes, como no servidor de multicast.

Registro e Saída de Agrupamento

Para se tornar um membro de um agrupamento, e assim ter acesso ao serviço de multicast, um sistema final deve se registrar junto ao provedor de serviço. Registro é um procedimento de inicialização que permite a uma entidade vir a ser instanciada como membro de um grupo.

O registro junto ao provedor do serviço é feito através da mensagem *containerJoin(…)*. É importante observar que não é especificado aí nenhum endereço de grupo. A Figura 4.6 ilustra o diagrama de sequência para o registro de agrupamento. O registro no agrupamento permite a uma entidade se tornar conhecida pela gerência do grupo, o que permite depois passar por uma operação de adesão e assim vir a se tornar um membro de um grupo. Antes de ser registrada, a entidade tem que se submeter as regras que definem o agrupamento. Para isso, a entidade precisa apresentar suas caraterísticas ao

agrupamento, como por exemplo, fornecendo seu endereço para ser incluído nos diretórios e tabelas do agrupamento, e os parâmetros que caracterizam suas capacidades para que sejam aplicadas a ela em comunicações posteriores.

A interface para o registro de agrupamento é definida na classe *ContainerManager*.

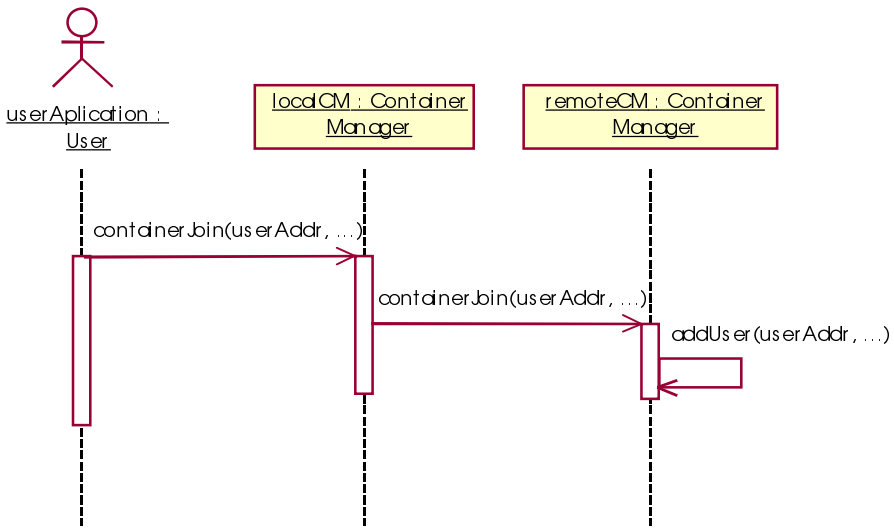

Figura 4.6: Diagrama de sequência de registro de grupo

No modelo de gerenciamento centralizado, o gerenciador de agrupamento local notifica ao gerenciador central sobre a entrada (ou saída) do membro ao agrupamento.

Caso um sistema final peça a sua saída do agrupamento através da mensagem *containerLeave(…)*, quaisquer referências a ele nas tabelas do provedor do serviço são apagadas.

Criação do Grupo

Esta operação, definida através da primitiva *groupCreate(grpAddr, …)* refere-se a fase de criação de um grupo, sendo realizada através da criação de estruturas de grupo (instanciação das tabelas de grupo - *GroupTable*), em uma base de informação de grupos, que pode estar distribuída ou centraliza. Este procedimento permite ainda a um

criador de grupo especificar um conjunto de regras que definem o perfil dos futuros membros do grupo, como por exemplo, requisitos mínimos de qualidade de serviço.

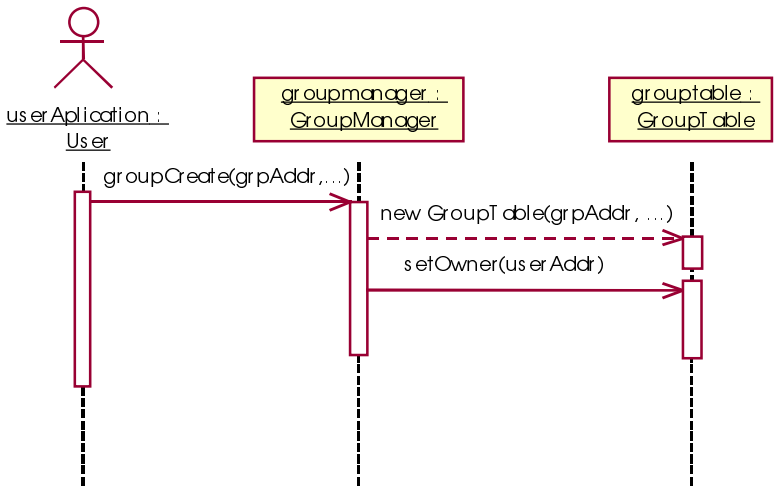

Figura 4.7: Diagrama de sequência da criação de grupo

É importante destacar que, no modelo de gerenciamento centralizado, ainda existe uma outra troca de mensagens entre um gerenciador de grupo local e um outro gerenciador de grupo remoto (servidor de gerenciamento de grupo), exatamente igual a troca de mensagens ocorrida entre o usuário e o gerenciador de grupo local, mostrada na Figura 4.7, isto é, a instância do gerenciador de grupo local passa a cliente de uma outra instância central do gerenciador de grupo, que mantém a base de informações sobre todos os grupos gerenciados no agrupamento.

Adesão e Saída de Grupo

Torna uma entidade já registrada membro de um grupo. A adesão a um grupo é obtida através da primitiva *groupJoin(grpAddr, userAddr, ...)*, passando-se o endereço do grupo (*grpAddr*) e o endereço do usuário (*userAddr*). Outras informações sobre o usuário também podem ser fornecidas, como por exemplo parâmetro de qualidade de serviço. Operações de adesão a um grupo levam a uma atualização da base de dados do grupo (*GroupTable*) através da primitiva *join(userAddr, ...)*.

Adesões a grupos podem, opcionalmente gerar eventos de notificação para indicar a entrada do novo participante ao grupo. Notificações são realizadas através do envio de mensagens de adesão (*groupJoin(grpAddr, userAddr, ...)*) a gerenciadores de grupo remotos. Essas notificações podem ser utilizadas, por exemplo, para que o servidor de resolução de endereço multicast (MARS [**Armitage 96**]), no modelo de gerenciamento centralizado adotado para redes ATM, possa atualizar sua base de informações dos grupos do agrupamento gerenciado, ou ainda, para que transmissores para um grupo, ainda no modelo centralizado, que tenham estabelecido conexões ponto a multiponto com os membros do grupo possam adicionar o novo membro do grupo à conexão, como já falado anteriormente. Num modelo distribuído, como no caso do IP Multicast sobre Ethernet (usando o IGMP como protocolo de gerenciamento de grupo) [**Deering 89**], essa notificação pode ser utilizada para informar ao roteador da rede local que uma de suas estações aderiu a um grupo, para que ele possa assim dar início ao processo de adesão a árvore multicast para o grupo, se necessário. Nesse último caso, mais uma vez o pattern *Observer* será utilizado, como explicado mais adiante.

A Figura 4.8 ilustra o diagrama de sequência para a adesão a um grupo.

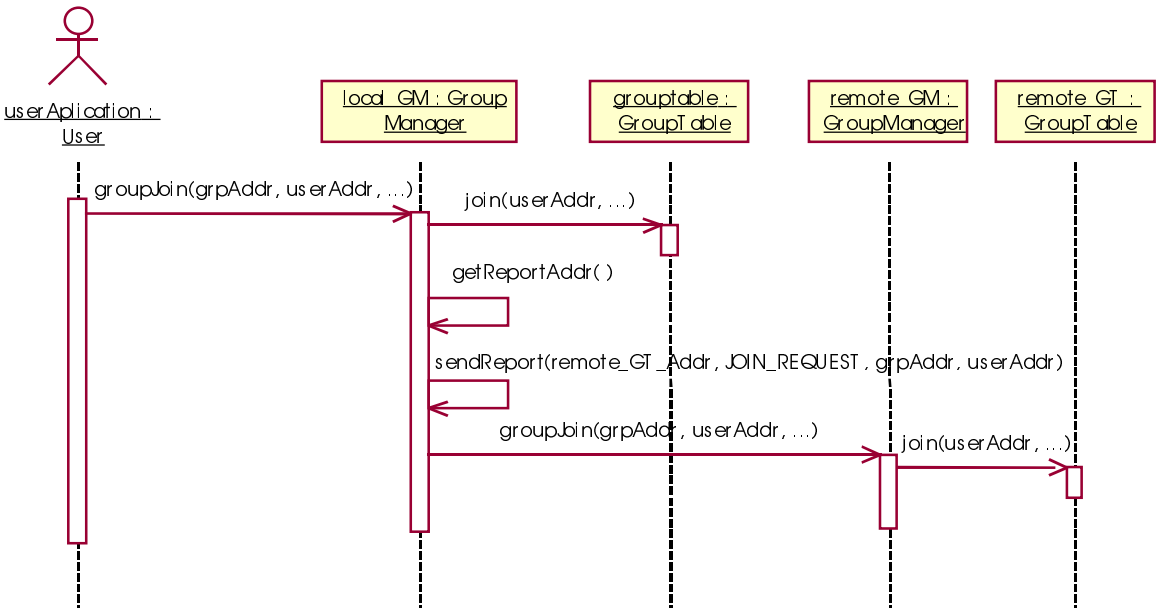

Figura 4.8: Diagrama de sequência da adesão de grupo

O processo de saída de um grupo é bem semelhante a adesão, obtida através da primitiva *leaveGroup(grpAddr, userAddr)*. O pedido de saída do grupo provocará a exclusão do endereço do usuário da tabela de grupo identificada por *grpAddr*. Da mesma forma que na adesão, a saída de um grupo também poderá provocar uma notificação a entidades remotas de gerenciamento de grupo.

4.4.1.3 Modelo de Gerenciamento e Resolução de Endereço

O modelo para resolução de endereço é formado basicamente por uma operação abstrata *resolv(Address)*, definida em *AddressManager*, que especifica a interface de resolução de endereço para um dado grupo, a ser implementada de acordo com a estratégia de resolução de endereço adotada.

A resolução pode ser feita de forma direta ou por intermédio de protocolos de resolução. Na forma direta, o protocolo da camada *N* é capaz de traduzir localmente o endereço para o endereço de nível *N-1*, podendo utilizar-se de uma tabela ou de uma função de mapeamento. No caso do IP Multicast sobre Ethernet, por exemplo, o mapeamento é bastante simples: os 23 bits menos significativos do endereço IP de classe D devem ser colocados nos 23 bits menos significativos do endereço de multicast Ethernet. A Figura 4.9 ilustra como o endereço de grupo multicast 234.138.8.5 (ou EA-8A-08-05 expressa em Hex) é mapeado em um endereço de multicast IEE-802. Note que os nove bits mais significativos do endereço IP são totalmente ignorados, não sendo mapeados no endereço de multicast de nível MAC.

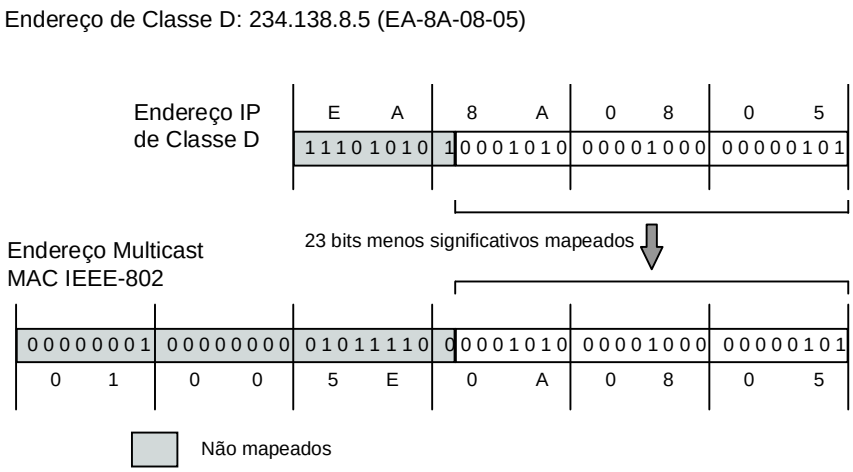

Figura 4.9: Mapeamento entre endereço de classe D e e endereço multicast IEEE-802

Note que o mapeamento não é único, isto é, o mapeamento pode colocar múltiplos grupos IP em um mesmo endereço IEEE-802, devido ao "descarte" dos cinco bits mais significativos do identificador de grupo do endereço IP de classe D. Assim, existe uma razão de 32:1 de endereços IP de classe D para endereços de multicast MAC válidos. Na prática, existe uma pequena chance de colisões, isto é, o conjunto de endereços é grande o suficiente de modo que as chances de dois grupos escolherem endereços com o mesmo conjunto dos 23 bits menos significativos é muito pequena. Por outro lado, a consequência desse projeto é que alguns datagramas multicast podem ser recebidos por estações que não deveriam recebê-los. Assim o software IP deve cuidadosamente checar endereços em todos os datagramas recebidos, e descartar qualquer datagrama não esperado.

Na resolução através de protocolos, a entidade da camada *N* faz uma requisição solicitando que alguma outra entidade (ou entidades) retornem os endereços de nível *N-1* correspondentes. O protocolo ARP [**Plummer 82**] é um exemplo de protocolo de resolução para endereços IP unicast.

Quando o nível *N-1* já oferece um serviço com endereçamento multicast, a resolução direta pode ser realizada através do mapeamento de cada endereço de multicast de nível *N* em um endereço de multicast do nível *N-1*, que corresponde, exatamente, à abordagem seguida pelo IP multicast sobre Ethernet. Caso não exista endereçamento

multicast no nível inferior, as seguintes abordagens podem ser utilizadas, dependendo da distribuição das informações de grupo:

1. Quando as informações de grupo estão centralizadas em um *servidor para resolução de endereços* (SRE), a resolução através de protocolo pode ser utilizada requisitando-se desse servidor a resolução desejada; o SRE é responsável apenas pela resolução de endereços e não pela distribuição das mensagens para os integrantes do grupo. O transmissor, após receber os endereços traduzidos, é responsável por distribuir mensagens para os integrantes do grupo.

2. Quando as informações de cada grupo estão centralizadas em um *servidor para distribuição de mensagens* (SDM), a resolução direta pode ser utilizada, mapeando-se cada endereço de grupo ao endereço *N-1* do SDM correspondente. Assim, ao receber as mensagens, o SDM imediatamente as distribui aos integrantes do grupo. Dessa forma, clientes desconhecem o fato de que o sistema de nível *N-1* não possui endereçamento multicast já que o endereço do SDM passa a servir como o endereço de multicast sob o ponto de vista desses clientes.

3. Quando o nível inferior tem capacidade de difusão e as informações de grupo estão distribuídas, a resolução pode ser feita através de um protocolo no qual a requisição de resolução é difundida e as entidades capazes de responder enviam ao cliente os endereços mapeados. A forma mais óbvia de distribuição é aquela na qual cada participante tem as informações dos grupos aos quais ele próprio pertence. Dessa forma, todos os componentes de um grupo respondem a requisições de resolução de endereços daquele grupo, cada um com seu próprio endereço de nível *N-1*.

A primeira abordagem foi utilizada pelo IETF na definição do MARS [**Armitage 96**] para a resolução de endereços de grupo em redes IP Multicast sobre ATM. A definição do MARS prevê ainda que a segunda abordagem pode ser utilizada em conjunto com o servidor de resolução, na qual o SDM (denominado Multicast Server - MCS) atua na parte da distribuição e o MARS propriamente dito na parte de resolução. Assim, o MARS é responsável por resolver os endereços de grupo alternativamente: nos endereços *N-1* dos componentes do grupo, ou no endereço *N-1* de um MCS. No caso em que não há MCS, o cliente, após a resolução, é responsável por criar uma conexão ponto-a-multiponto com todos os integrantes. Havendo um MCS, o cliente deve estabelecer com ele uma conexão ponto-a-ponto; o MCS, por sua vez, é responsável por manter uma conexão ponto-a-multiponto com todos os integrantes do grupo em questão.

A combinação da segunda abordagem com a primeira é conveniente, pois o uso da segunda abordagem de forma isolada dificulta a criação e destruição de grupos de forma dinâmica. A terceira abordagem, apesar de viável, não é utilizada na prática para resolução de endereços de grupo. Porém, para a resolução de endereços unicast (ou considerando cada grupo como sendo formado por um só elemento), essa abordagem corresponde exatamente ao ARP tradicional.

As subclasses de *AddressManager* especializam o método de resolução de endereço específico para cada abordagem. A subclasse *DirectMappingResolution* é a responsável pelo mapeamento direto através de funções (como no caso do IP multicast sobre Ethernet), enquanto que *TableMappingResolution* e *DynamicVinculationResolution* cuidam, respectivamente, da resolução local (utilizada por exemplo em servidores de resolução) e resolução por protocolo (utilizada pelo usuário que requisita a resolução). Um usuário em um ambiente sem mapeamento direto, envia uma requisição para um endereço que pode ser configurado, obtido através da função *getResolverAddr()*. Esse endereço será o endereço de um servidor, no caso do modelo de gerenciamento centralizado, ou o endereço all-users, no caso distribuído. No servidor, uma instância da classe *TableMappingResolution* será utilizada para a resolução do endereço, como mostrado na Figura 4.10. Ao receber a resposta, o cliente pode atualizar uma tabela local

que funciona com cache de resolução. A Figura 4.11 ilustra a sequência de troca de mensagens adotada no protocolo de resolução de endereço por vinculação dinâmica.

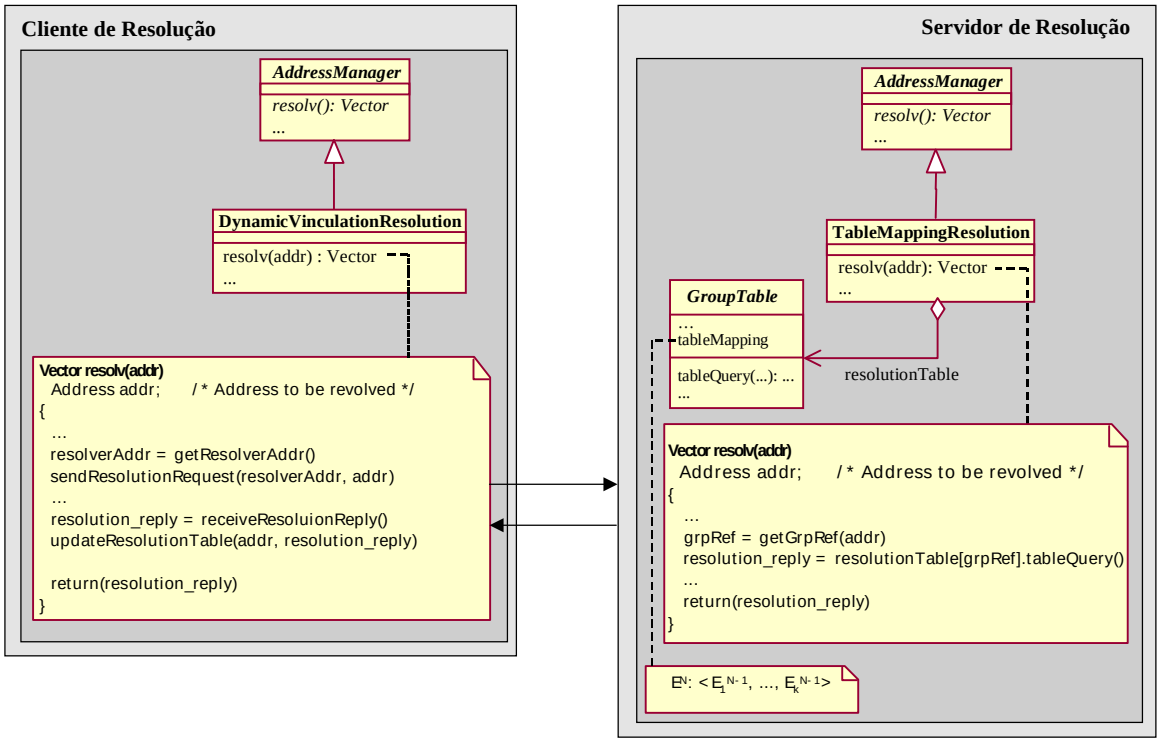

Figura 4.10: Exemplo de especialização do método *resolv*() para resolução por vinculação dinâmica e por tabela de mapeamento

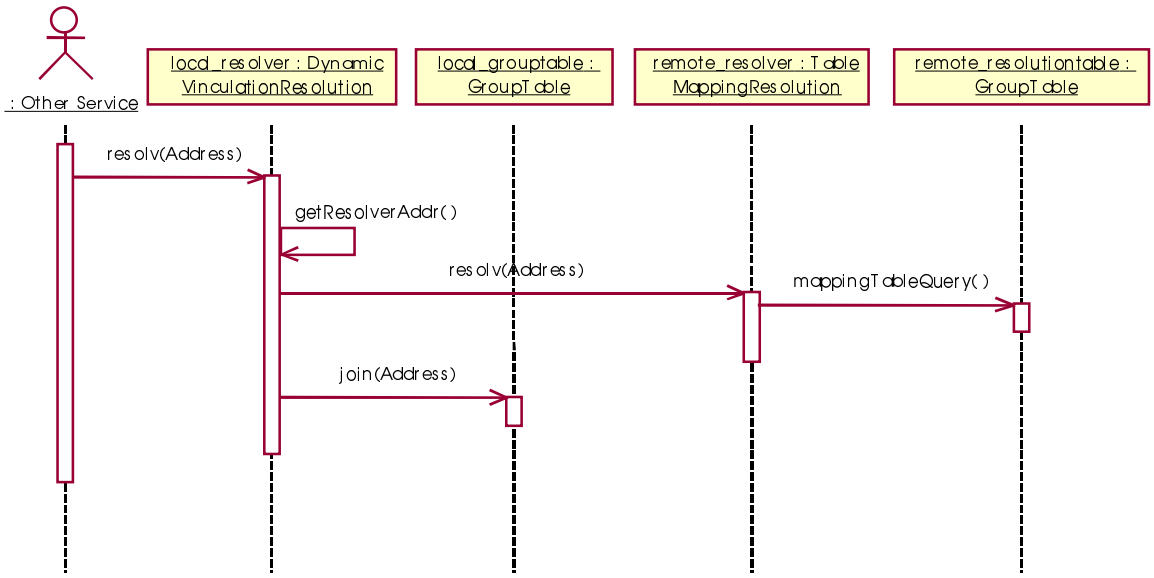

Figura 4.11: Diagrama de sequência de resolução de endereço por vinculação dinâmica

4.4.2 Exemplos de Aplicação do Framework de Serviço de Gerenciamento de Grupo

Essa Seção apresenta dois caos de aplicação do framework para a definição de um serviço de gerenciamento de grupo e resolução de endereço. O primeiro, mostra a aplicação do framework para a definição de um serviço centralizado. A seguir é ilustrado um caso de aclimação para gerenciamento distribuído.

4.4.2.1 Configuração de um Serviço de Gerenciamento de Grupo e Resolução de Endereço Centralizado

A configuração de um serviço de gerência e resolução de grupo centralizado é ilustrada na Figura 4.12. Um servidor que concentra todas as informações dos grupos do agrupamento deve ser configurado. Esse servidor deve possuir uma instância de *GroupManager* e uma de *AddressManager*, responsáveis por gerenciar e manter todas as informações do grupo. A configuração em *GroupManager* diz respeito somente ao endereço utilizado para a notificação da alteração (obtido através da função *getNotifyAddr()*) que, nesse caso, por não possuir servidor de multicast, será o endereço *all-users* (todos os usuários do agrupamento precisam tomar conhecimento das alterações ocorridas no agrupamento.

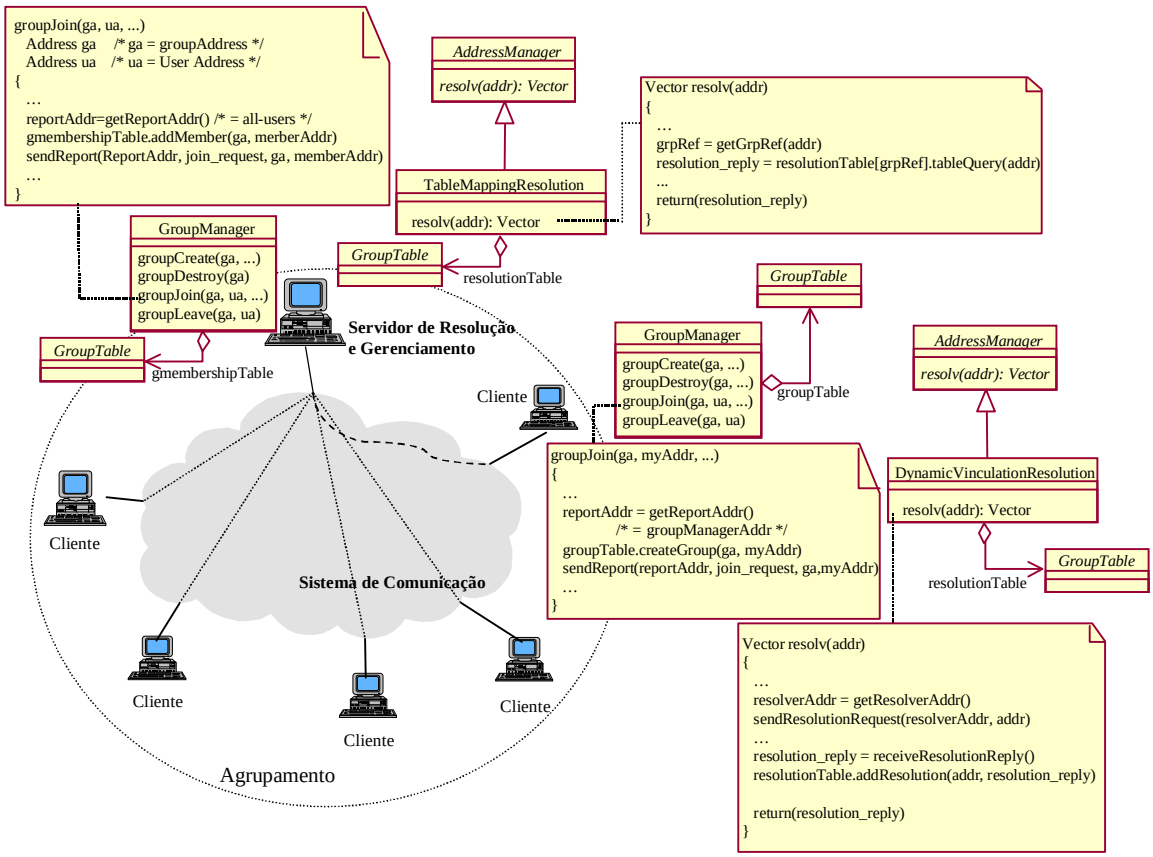

Figura 4.12: Exemplo de configuração do serviço de gerenciamento de grupo centralizado

O cliente do serviço deve possuir uma instância de *GroupManager*, onde devem ser configuradas as funções de criação e manutenção de grupo, e uma instância de *AddressManager*, responsável pela interface de resolução de endereços. No cliente, o endereço de notificação da alteração do grupo deve ser configurado como sendo o endereço do servidor de gerenciamento de grupo. Da mesma forma, deve ser configurado, no cliente, o endereço do gerenciador de resolução a ser utilizado para o mapeamento do endereço. Ao receber uma requisição de resolução de endereço de um usuário do serviço, o cliente verifica em um cache local se já existe um mapeamento para o endereço requisitado. Se já existir, esse mapeamento será devolvido ao usuário requisitante. Caso contrário, uma requisição de resolução será enviada para o servidor de resolução (no endereço obtido por *getResolverAddr()*), e então o cliente passa a um estado de “espera” pela resposta a sua requisição (que pode ser um endereço de grupo do nível inferior, um conjunto de endereços do nível inferior dos vários membros do grupo, um conjunto de

endereços dos servidores de distribuição multicast do grupo, ou ainda um conjunto vazio de endereços, quando o mapeamento não existe ou o grupo está vazio). O servidor, ao receber a requisição, consulta suas tabelas de resolução, e retorna o mapeamento relacionado ao endereço consultado. O resultado da resolução é então mantida em um cache de resolução no cliente.

4.4.2.2 Configuração de um Serviço de Gerenciamento de Grupo e Resolução de Endereço Distribuído

A Figura 4.13 ilustra um exemplo de aplicação do framework de gerenciamento de grupo na configuração de um esquema de gerenciamento distribuído, tal como é implementado pelo IP Multicast sobre redes Ethernet, usando o protocolo IGMP.

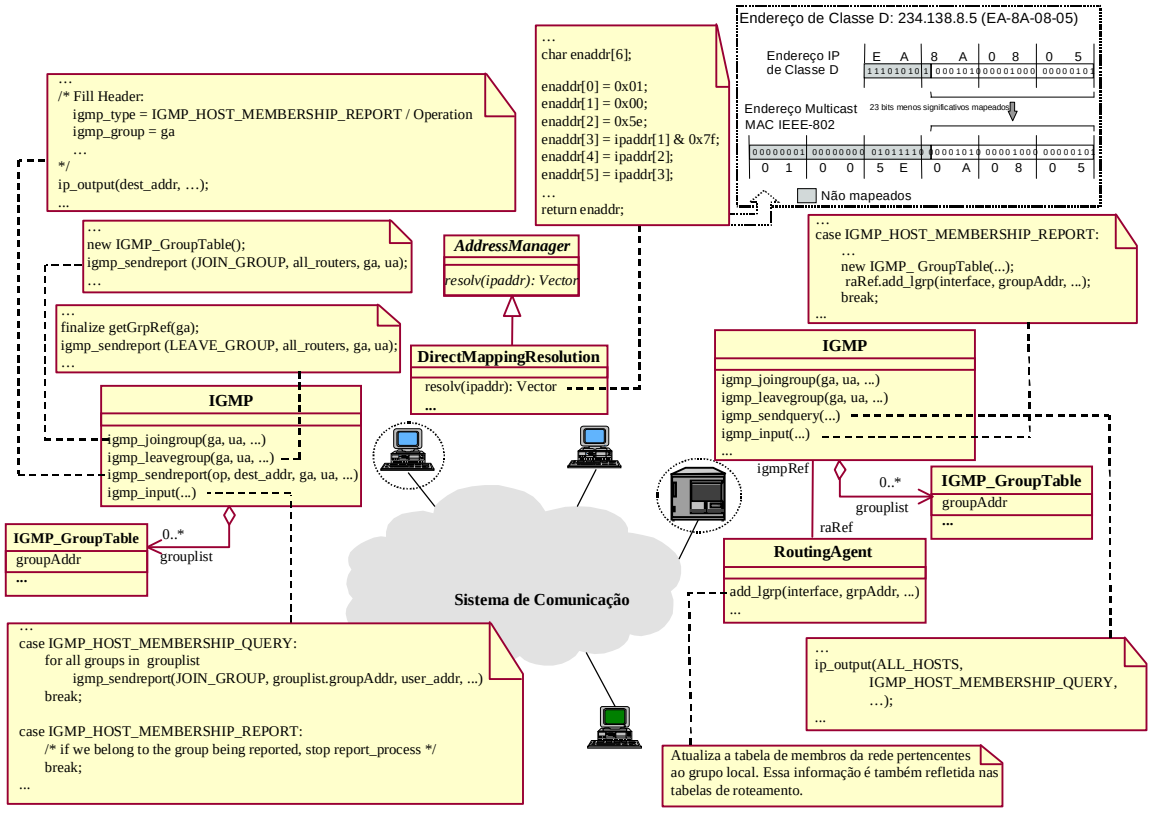

Figura 4.13: Exemplo de configuração do serviço de gerenciamento de grupo distribuído

No esquema distribuído, as informações do grupo estão distribuídas, cabendo a cada participante de grupo guardar apenas informações locais dos grupos a que faz parte.

A classe *IGMP* representa o elemento responsável pelo gerenciamento de grupo distribuído (*GroupManager*). Nessa classe são definidos os métodos de adesão e saída do grupo (*igmp_joingroup(…)* e *igmp_leavegroup(…)*), e o método de notificação de alteração do grupo (*igmp_sendreport(…)*). A configuração dessa classe é feita de forma similar, tanto no roteador, quanto nas estações da rede. Roteadores multicast enviam mensagens de consulta de membros de grupos (IGMP_HOST_MEMBERSHIP_ QUERY) para descobrir que grupos de estações tem membros nas suas redes locais diretamente ligadas. A consulta é feita através do método *igmp_sendquery(…)*. Consultas são endereçadas para o grupo "*all hosts*". Estações respondem a consulta gerando mensagens de relatório de membros de grupos (IGMP_HOST_MEMBERSHIP_ REPORT), informando cada grupo de estações que elas pertencem na interface de rede através da qual a consulta foi recebida.

A estratégia para resolução de endereço adotada é a mapeamento direto, como já descrita anteriormente.

4.5 Construção da Infra-estrutura de Distribuição

O serviço de roteamento é baseado em dois patterns principais: *Facade* (*RoutingSupportAgent*) e *Strategy* (*RoutingProtocol*) [**Gamma et al 95**]. A Figura 4.14 apresenta a estrutura do serviço de roteamento multicast. Na figura, contornado, pode-se ver a estrutura genérica do serviço, enquanto na parte inferior, pode-se ver algumas especialização dos pontos de flexibilização do serviço.

O serviço de roteamento pode ser implementado de modo centralizado, ou distribuído. No modo centralizado, as funções de roteamento são concentradas em um

único ponto do sistema de comunicação. Além das funcionalidades, também estão concentradas em um ponto todas as informações de roteamento no sistema. No modo distribuído, cada entidade do sistema de comunicação é responsável por manter e manipular suas próprias informações de roteamento.

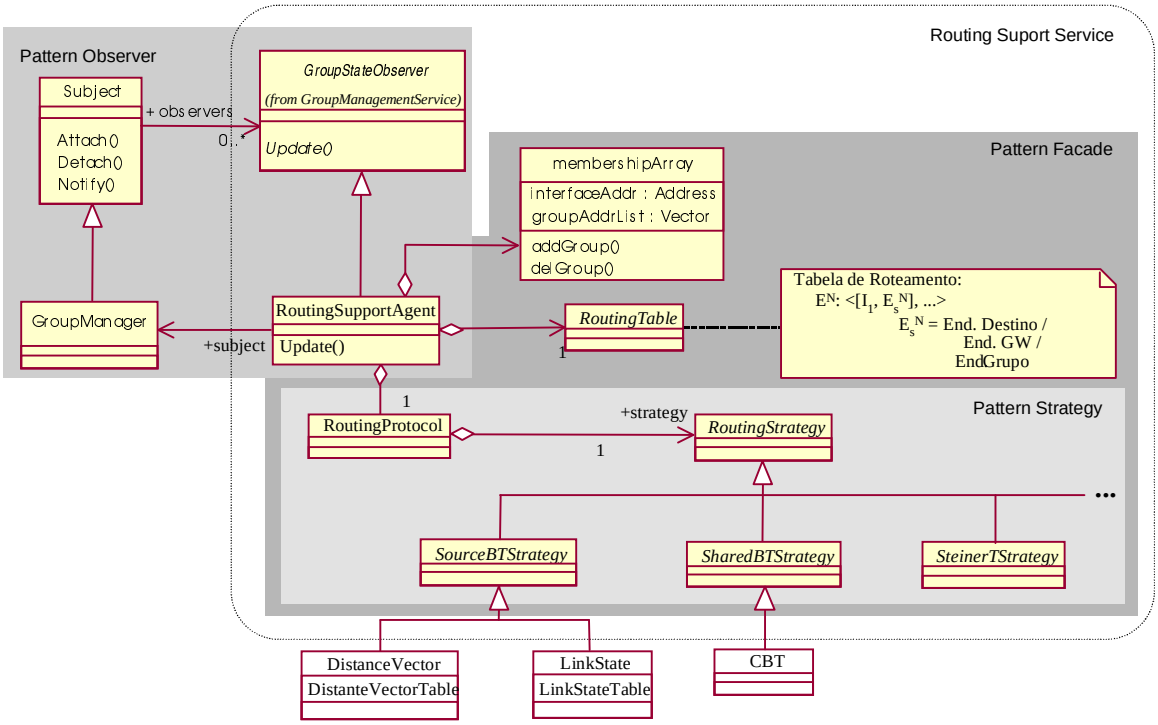

Figura 4.14: Framework do Serviço de Roteamento Multicast

A base de dados contida na tabela de roteamento (*RoutingTable*) representa uma abstração lógica da infra-estrutura de distribuição multicast. Basicamente, essa tabela mapeia endereços do nível *N* a tuplas contendo interface de saída de dados, e endereços do próximo destino, que pode ser um usuário individual, um gateway, ou um grupo.

O protocolo de roteamento (*RoutingProtocol*) é responsável por manter as informações contidas na tabela de rotas do agente de roteamento. A escolha da heurística adotada pelo protocolo de roteamento é fundamentada nas características da aplicação. O framework de serviço de roteamento provê facilidades para se configurar um protocolo de roteamento, através da escolha e especialização da estratégia de roteamento

(*RoutingStrategy*), baseado nas mais diferentes características das aplicações, e do sistema de comunicação utilizado.

4.6 Resumo

Esse capítulo, apresentou uma breve introdução a conceitos de orientação a objetos, e aplicação de patterns e frameworks no desenvolvimento e implementação de sistemas de comunicação e aplicações distribuídas. Em seguida, foi apresentada, considerando os vários trabalhos apresentados no Capítulo 3, e nos requisitos descritos na Seção 2.2, uma arquitetura genérica para a provisão do serviço de multicast. Finalmente, ainda nesse capítulo, foi feita a descrição do Framework para Provisão do Serviço de Multicast. Essa descrição foi feita detalhando-se os serviços e patterns componentes do serviço de multicast. Esse capítulo apresentou ainda, para cada componente do serviço, um exemplo de aplicação do framework. A arquitetura do framework foi apresentada através da utilização da *Linguagem de Modelagem Unificada* (UML - Unified Modeling Language) [**Booch et al 96**].

Capítulo 5

Aplicação do Framework: Implementação de um Agente Mediador de um Serviço de Multicast

O atual padrão de sinalização ATM [**ATMF 94**] permite apenas a configuração e finalização de conexões multicast iniciadas pela origem do fluxo de dados, o que se torna bastante restritivo, em comparação com esquemas de multicast que suportam a adesão e saída de participantes iniciada pelos receptores. A reserva de recursos para os canais virtuais multicast deve ser realizada pela origem, e todos os receptores devem ter os mesmos requisitos de qualidade de serviço, incluindo a taxa de dados a ser recebida. Esse suporte multicast rudimentar não permite que receptores heterogêneos possuam diferentes capacidades de recepção ou exigências de QoS [**Li et al 97**].

Em virtude dessas limitações, propomos aqui a implementação de um serviço configurável de transmissão de vídeo para um grupo de participantes com

exigências diferentes de QoS, baseado nos frameworks para a provisão de serviço de multicast, apresentado no capítulo anterior, e um framework para a provisão de qualidade de serviço [**Gomes et al 99**]. O serviço é orientado a conexão e iniciado pelo receptor. O serviço é configurável no sentido de permitir a criação de uma infra-estrutura de distribuição multicast heterogênea, de acordo com os requisitos de QoS exigidos pelos receptores e suportados pelo transmissor.

Utilizou-se de recursos de simulação para a prototipação do serviço. A plataforma de simulação consiste de quatro estações de trabalho conectadas a um comutador ATM IBM 8260, como descrito posteriormente.

A seguir, será descrito brevemente o serviço proposto. Em seguida, será descrita a arquitetura do serviço, bem como os principais componentes do serviço. Será também mostrada a aplicação do framework na construção dos serviços de gerenciamento de grupo, e suporte ao roteamento multicast. Finalmente, será descrita a plataforma de prototipação implementada.

5.1 Descrição do Serviço

A Figura 5.1 apresenta uma visão geral do serviço de distribuição de vídeo e seus componentes. Do ponto de vista dos usuários do serviço, podemos ver claramente a distinção de dois papeis: servidor de vídeo, no papel de transmissor (ou produtor, como classificado posteriormente), e assinantes, no papel de receptores (ou consumidores). Cada canal de vídeo será criado como um grupo, onde existe apenas um transmissor por grupo. Cabe aos receptores assinarem o canal, e posteriormente iniciar a recepção de dados, através da seleção do canal.

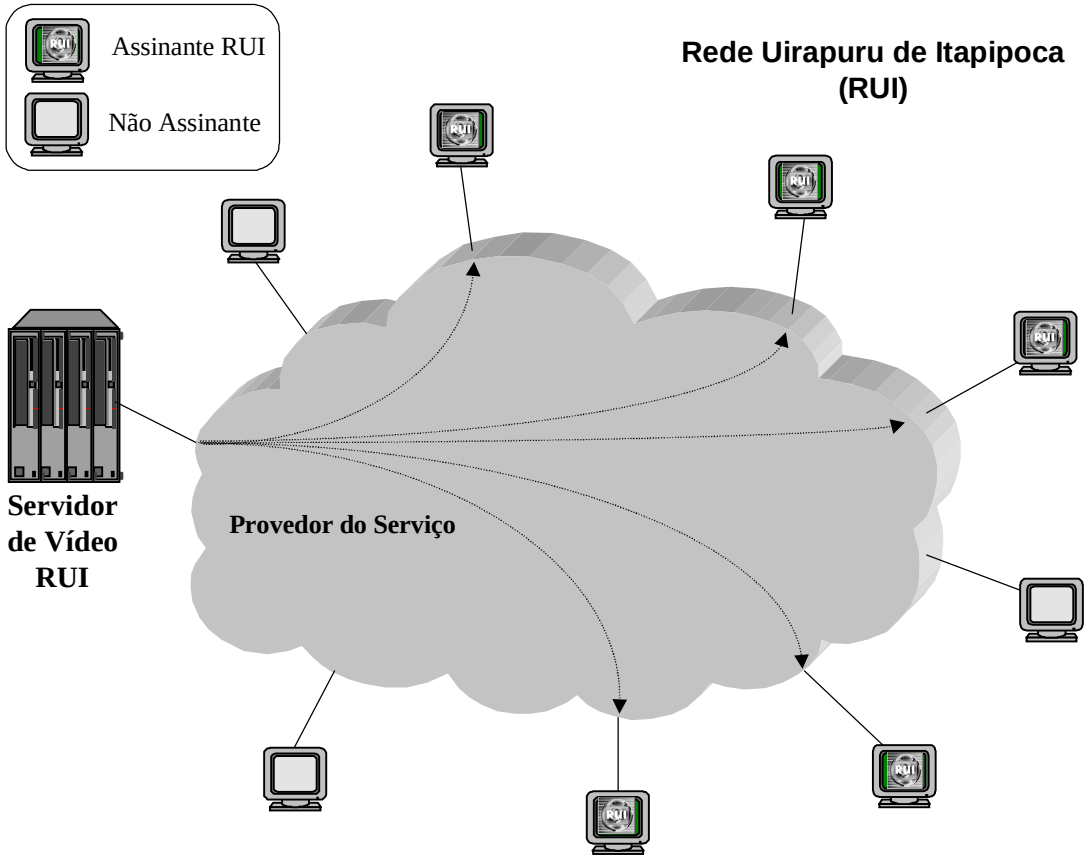

Figura 5.1: Exemplo de Serviço de Transmissão de vídeo

De acordo com essa classificação, pode-se ver a distinção entre duas interfaces de acesso ao provedor do serviço. Uma destinada ao servidor de vídeo, e outra destinada ao conjunto de receptores (clientes). As *primitivas dos servidores* permitem aos servidores de vídeo criar, publicar e encerrar fluxos (*canais*) de distribuição. Já as *primitivas dos clientes* permitem aos clientes obter informações sobre canais de distribuição, assinar e selecionar um canal de distribuição específico.

A seguir, será mostrada resumidamente a descrição de cada primitiva.

5.1.1 Publicar Canais

A publicação de canais permite a servidores de distribuição divulgarem canais de transmissão. A divulgação cria um canal de distribuição. Nesse instante, é feita

uma negociação entre o servidor e o provedor do serviço. Essa negociação garantirá que o serviço publicado e oferecido pelo servidor será cumprido pelo provedor.

5.1.2 Consulta de Canais

Permite que usuários, já devidamente registrados no serviço, possam consultar os canais ativos no momento. No momento da consulta, um cliente pode opcionalmente informar uma qualidade de serviço, ou um intervalo de qualidade de serviço requerido. O resultado da consulta será uma lista de tuplas {*canal, QoS*}, onde *canal* representa o identificador do canal, e *QoS* representa a qualidades de serviço associada ao canal fornecido pelo servidor.

5.1.3 Assinar um Canal

Permite a um usuário aderir a um grupo de assinantes de um canal, e assim ter direito de recepção do canal. Na assinatura, o assinante especifica a qualidade de serviço requerida. Nesse instante, é feita uma primeira negociação entre assinante e servidor, intermediada pelo provedor de serviço, para verificar se a qualidade requerida pelo assinante pode ser provida pelo servidor. Nenhuma negociação é feita ainda entre receptor, transmissor e provedor do serviço.

5.1.4 Seleção de Canais

A seleção permite a um assinante receber o fluxo de dados gerado no canal. Nesse momento, o assinante será efetivamente conectado ao canal, sendo assim necessário uma negociação entre o assinante e o provedor do serviço. A partir desse instante, o provedor de serviço encarrega-se de garantir a qualidade de serviço negociada e devidamente paga pelo assinante.

5.2 Arquitetura do Serviço

A arquitetura do serviço proposto pode ser modelada como um processo de distribuição de um produto, que envolve as interações entre *produtores*, *consumidores*, *mediadores* e *comutadores*. Desta forma, o modelo pode ser expresso como uma tupla {*P, C, M, S*}; onde *P* é o conjunto de produtores origem de informações distribuídas nas sessões; *C* é o conjunto de consumidores destino das informações produzidas pelos produtores; *M* é o conjunto de mediadores responsáveis pela implementação do esquema de gerenciamento de grupos e negociação da comunicação multicast entre produtores e consumidores; e *S* é o conjunto de comutadores que transportam e entregam o fluxo de dados dos produtores aos consumidores. A relação entre esses componentes pode ser vista na Figura 5.2. Na figura pode-se ver os quatro principais personagens do serviço: um gerenciador do serviço de transmissão de grupo (mediador), um transmissor (produtor - servidor de vídeo), um grupo de receptores (consumidores), e o conjunto de elementos intermediários do sistema de comunicação (comutadores).

O *gerenciador do serviço* é composto por dois componentes: um *mediador do serviço de multicast* (MSB – *Multicast Service Broker*), responsável por gerenciar as informações sobre a composição dos grupos e controlar o acesso aos canais publicados, e um *gerenciador de negociação de QoS*, responsável por verificar se a QoS requerida pelo assinante é compatível com a QoS provida pelo servidor. O MSB possui ainda três componentes, instanciados a partir dos componentes descritos no serviço de gerenciamento de grupo, apresentado na Seção 4.4: uma *Agência de Gerenciamento de Grupo* (GMA), uma *Agência de Gerenciamento de Endereço* (AMA), e um *Gerenciador do Agrupamento* (CM). O gerenciador do serviço funciona como uma base centralizada de informações de gerência, onde as interfaces de serviço dos clientes e servidores (localizadas nas estações finais) podem publicar e obter informações sobre a configuração de canais (através das agências de gerenciamento de grupo – *GMA* - e endereço - *AMA*). Por intermédio do *gerenciador de negociação da QoS* (*QoSNM*), é feita a negociação da QoS entre os clientes e os servidores.

O serviço de suporte a criação da infra-estrutura de roteamento é distribuído pelos elementos intermediários do sistema de comunicação (comutadores).

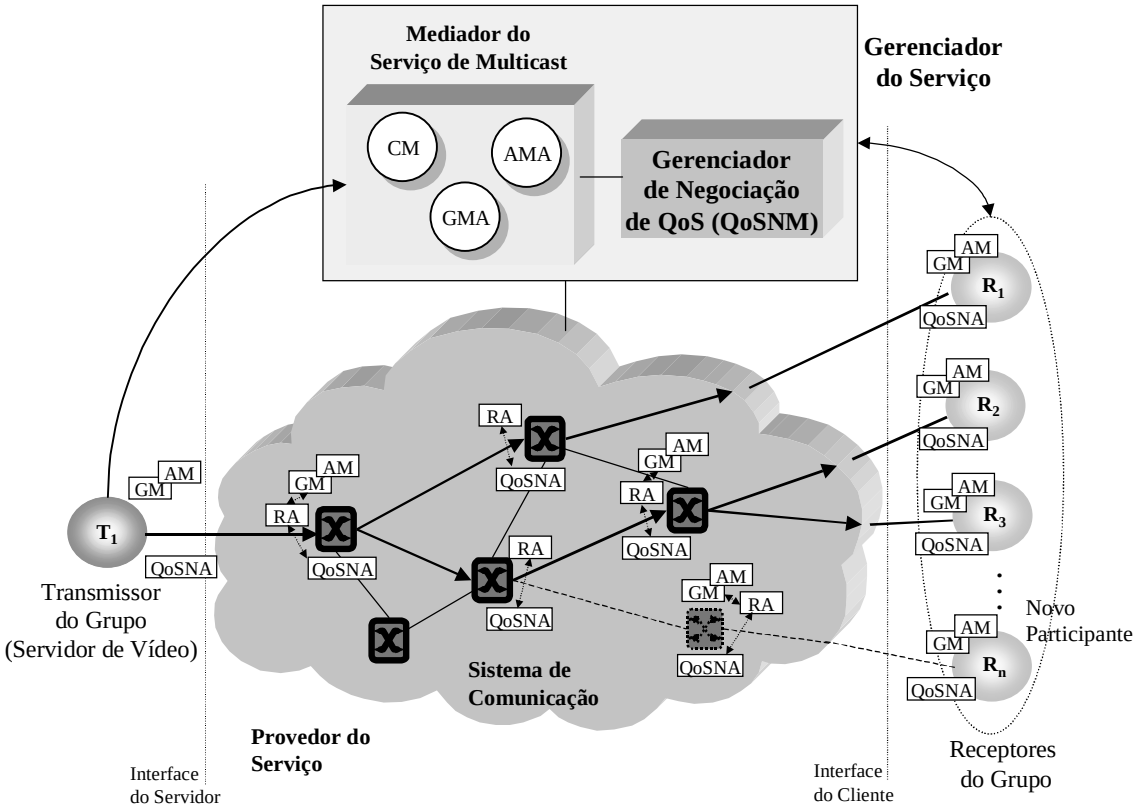

Figura 5.2: Arquitetura de aplicação do framework

Cada estação final é dotada de um *gerenciador de grupos* e *de endereçamento* (*GM* e *AM*), e *um agente de negociação da QoS* (*QoSNA*). Os dois primeiros componentes atuam basicamente como gerenciadores de caches de informações gerenciadas, respectivamente, pelo GMA e pelo AMA, enquanto o último é responsável por gerenciar os recursos da estação final, e por iniciar o processo de reserva e alocação de recursos na rede, que é executado pelos *agentes de negociação da QoS* presentes em cada um dos comutadores ATM. O suporte ao roteamento nesta rede, em cada um de seus comutadores, é dado pelo *agente de roteamento* (*RA*).

A configuração das classes do GM, GMA, AM e AMA é feita como mostrado na Seção 4.4.2.1, adotando uma abordagem centralizada para a provisão do serviço. O RA é responsável por manter e gerenciar as informações de roteamento

multicast. A estratégia adotada pelo protocolo de construção da árvore foi a de árvore compartilhada.

O gerenciador do serviço mantém conexões bidirecionais ponto a ponto com todo os seus clientes, isto é, transmissores e receptores. Essas conexões são estabelecidas pelos clientes no momento do seu registro no serviço de transmissão multicast. Os clientes podem enviar requisições, ou receber respostas do gerenciador através dessas conexões.

Alguns dos componentes introduzidos até agora serão apresentados em maiores detalhes nas subseções a seguir.

5.2.1 Definição do Serviço

Nessa subseção, serão mostradas mais detalhadamente as primitivas de interface do serviço e os serviços componentes de gerenciamento de grupo e de canais, onde será mostrada a aplicação do framework para a especificação dos componentes do serviço de multicast implementado.

5.2.1.1 Primitivas de Interface

As classes *ClientInterface* e *ServerInterface*, ilustradas na Figura 5.3, representam, respectivamente, os conjuntos de primitivas de interfaces dos clientes e dos servidores. Essa interface é definida segundo o Pattern Facade [**Gamma et al 95**]. A cada cliente ou servidor presente no sistema, está associada uma instância de uma dessas classes. A classe abstrata *ServiceInterface*, da qual as duas classes anteriores são especializadas, oferece o método *registry(...)*, que permite aos clientes e servidores ingressarem no sistema de distribuição. No momento do registro, é estabelecida uma conexão com o gerenciador do serviço para o envio da mensagem de registro e recepção de notificações exclusivas ao cliente, vindas do gerenciador. No gerenciador do serviço, o gerenciador do agrupamento (representado na Figura 5.6 através da classe *CentralCM*) atualiza sua base de informações, adicionando o endereço do cliente a uma tabela de

membros do agrupamento. Se não for informado nenhum endereço do cliente, o gerenciador de agrupamento criará um endereço para o cliente, e o devolverá, como reposta a sua requisição de registro.

A classe *ServerInterface* oferece aos servidores o método *publishChannel(...)*, que permite a publicação, no gerente de serviços, de novos canais. A publicação de um canal implica na criação de um grupo, junto à agência de gerenciamento de grupo, no gerenciador do serviço. Para isso, o endereço usado na notificação da operação, como definido em 4.4.1.2, é configurado como o endereço do gerenciador do serviço. No momento da criação do grupo, é informado ao gerenciador do serviço a categoria de serviço[20] adotada pelo servidor, bem como o endereço do *comutador núcleo* (*core switch*) usado para a distribuição multicast. A criação do canal (e do grupo) junto ao gerenciador do serviço só é realmente efetivada após a confirmação que o comutador núcleo é capaz de suportar o tráfego gerado pelo servidor. *publishChannel(…)* é responsável também pela iniciação do processo de configuração de fluxos de dados entre os servidores e os comutadores núcleos destes canais, efetivada através do método *request(...)*, definido em *EndSystemQoSN*. A cada canal, corresponde um núcleo responsável pelo repasse do fluxo de unidades de informação, referente ao canal, até os clientes, adotando, dessa forma, uma topologia em árvore única baseada em um núcleo como infra-estrutura de distribuição multicast. Os detalhes de como fluxos são configurados entre servidores e núcleos, bem como as vantagens e implicações desta abordagem, são relacionados à gerência e provisão de qualidade de serviço e, portanto, não são abordados em detalhes neste trabalho, sendo modelados pelo Framework para Provisão de QoS, descrito em [**Gomes et al 99**]. Serão discutidos, no entanto, os detalhes e vantagens da abordagem de árvore baseada em núcleo para a distribuição multicast.

[20] Categoria de serviço descreve um conjunto qualquer de parâmetros de qualidade de serviço.

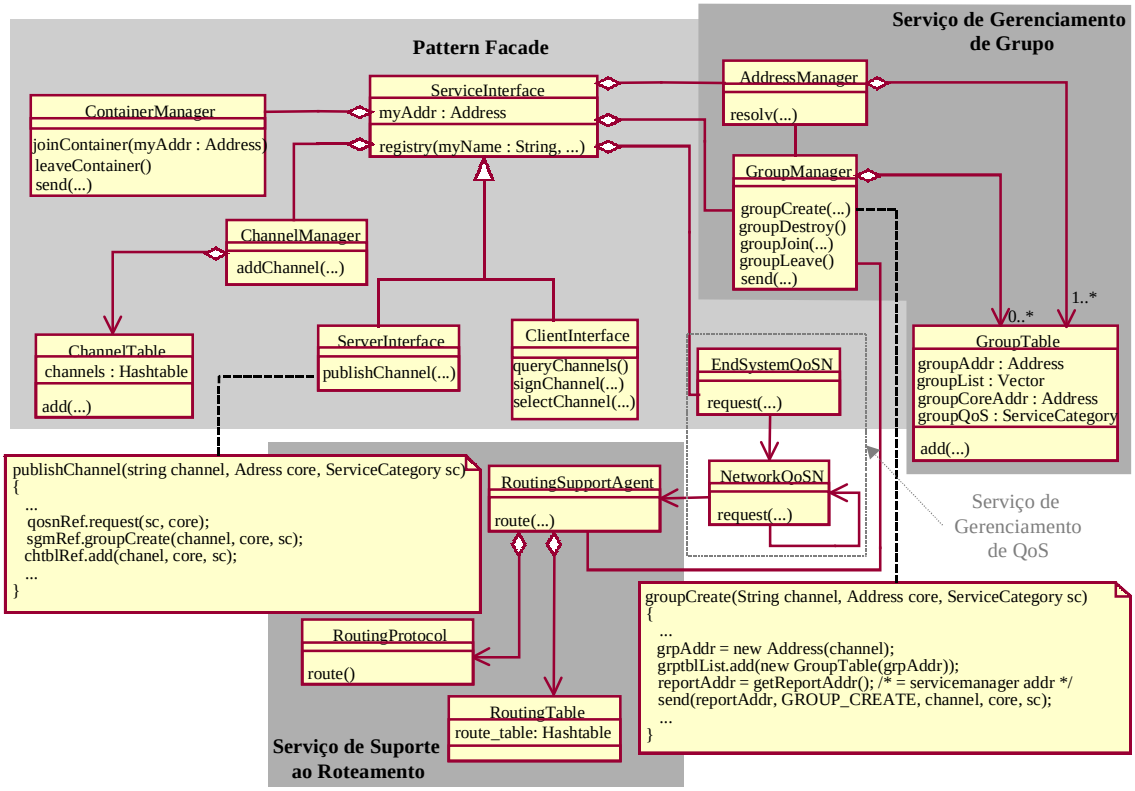

Figura 5.3: Hierarquia de classes do serviço para o transmissor e receptores do grupo

Após a confirmação da publicação do canal, são atualizados os caches locais de informações de grupo (*GroupTable*) e canais (*ChannelTable*), gerenciados, respectivamente, por *GroupManager* e *ChannelManager*, através dos métodos *groupCreate(...)* e *add(...)*. A Figura 5.4 apresenta o diagrama de sequência da publicação de canais.

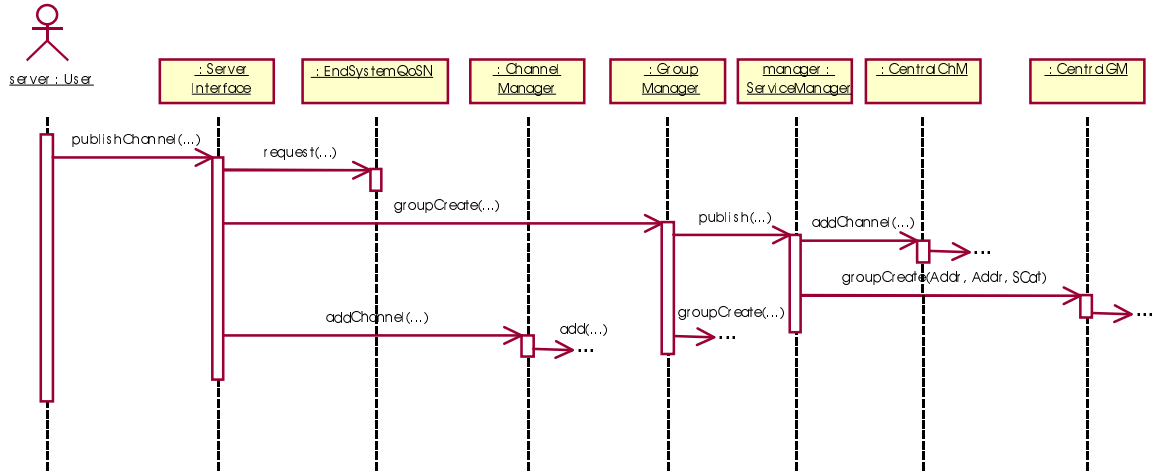

Figura 5.4: Diagrama de sequência da publicação de canais

A classe *ClientInterface* oferece aos clientes um conjunto de métodos para consulta, assinatura e seleção de canais. O método *queryChannels(...)* permite a um cliente consultar os canais, oferecidos pelo sistema de distribuição, que atendam a um determinado conjunto de restrições. No contexto deste trabalho, a única restrição que pode ser definida é a da categoria de serviço associada aos canais.

O método *signChannel(…)* possibilita a assinatura de um canal, o que envolve a adesão a um grupo junto ao gerenciador do serviço e a negociação entre a QoS desejada pelo cliente e a QoS oferecida pelo servidor do canal. A Figura 5.5 ilustra o diagrama de sequência de troca de mensagens desencadeada na assinatura de um canal. Ao requisitar a assinatura de um canal, o usuário deve informar, dentre outras coisas, o canal que ele deseja assinar e a categoria de serviço desejada, a ser utilizada na negociação da QoS entre cliente e servidor. A adesão ao grupo foi implementada de acordo com o modelo de gerenciamento de grupo centralizado, acrescido da fase da negociação de QoS. Após verificar que o servidor consegue prover o serviço com a qualidade requisitada pelo assinante, de acordo com o resultado da consulta dos canais, o assinante envia uma mensagem de assinatura (*sign(…)*) ao gerenciador do serviço, que desencadeará o processo de atualização de suas bases de dados. Esse processo envolve uma negociação simplificada da QoS, para verificar se a qualidade requerida pelo assinante pode ser provida pelo servidor, e por um pedido de adesão ao grupo associado ao canal

(*groupJoin(channelAddr, …)*). Ao receber a confirmação de que a adesão foi bem sucedida, o assinante atualiza seu cache local com as informações do canal assinado e do grupo aderido.

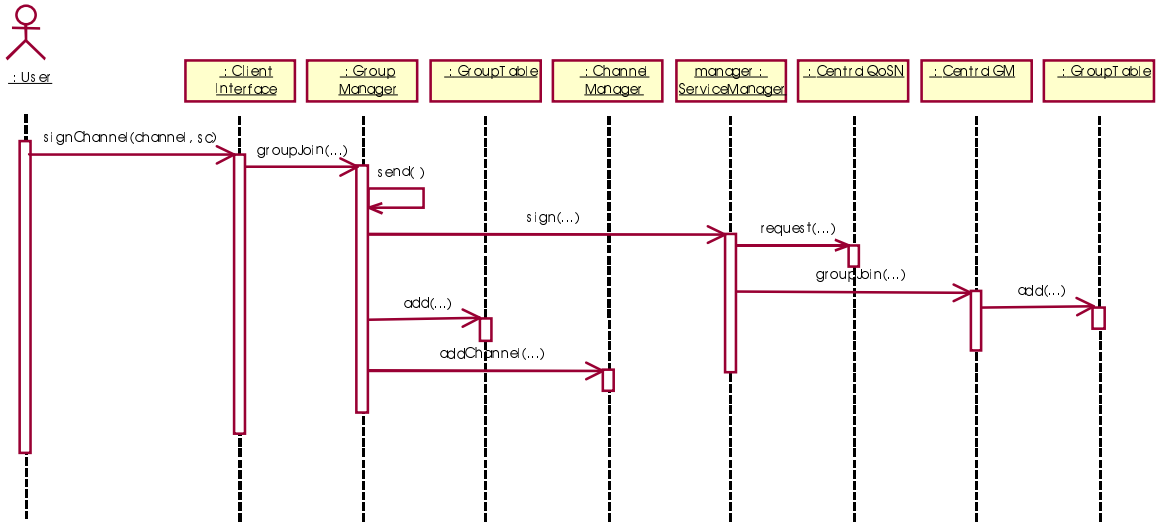

Figura 5.5: Diagrama de sequência da assinatura de um canal

O método *selectChannel(channel, ...)* possibilita ao cliente requisitar a criação de um fluxo que leve até ele as informações transmitidas pelo servidor de um determinado canal. Este método envolve a adesão à árvore de distribuição compartilhada e a configuração de um fluxo entre o núcleo do canal em questão e o cliente. A adesão à árvore ocorre segundo a estratégia adotada pelo CBT [Ballardie 97b]. Essa abordagem é especialmente interessante quando se deseja endereçar aspectos de escalabilidade no suporte a aplicações multicast através de uma inter-rede pública. Uma árvore baseada em um núcleo envolve em ter um único nó (um roteador), conhecido como núcleo da árvore (*core*), do qual emanam ramos. Esses ramos são formados por outros roteadores, conhecidos como roteadores não-núcleo (*non-core*), que formam o menor caminho entre uma estação membro, diretamente conectado ao roteador, e o núcleo. Os detalhes da configuração do fluxo não serão aqui apresentados.

O método *selectChannel(channel, ...)* consulta junto ao gerenciador de canal local qual o endereço do núcleo e a categoria de serviço negociada durante a fase de assinatura. De posse dessas informações, é feita uma requisição ao gerenciador de

negociação de QoS local para requisitar a criação de um fluxo (*request(…)*). Essa requisição será encaminhada ao gerenciador de negociação de QoS do comutador da rede (*request(…)*). Essa mensagem informa a seu comutador local que a estação deseja receber tráfego gerado no canal. Essa requisição desencadeará uma sequência de requisições aos vários comutadores entre o cliente e o núcleo da árvore. Durante esse encaminhamento, o agente de suporte ao roteamento de cada comutador intermediário será requisitado, para determinar qual o próximo comutador a seguir, em direção ao núcleo. Essa consulta é feita através do método *route(coreAddr)*. A determinação do caminho é feita de acordo com informações de alcançabilidade existente em tabelas de roteamento unicast previamente configuradas[21]. Assim, a requisição de criação do fluxo e adesão à árvore de distribuição do canal é encaminhada, salto-a-salto, em direção ao núcleo da árvore, de acordo com a tabela de transmissão unicast. Foi utilizado assim o algoritmo RPF (*Reverse Path Forwarding*) [Maufer et al 97] para o encaminhamento das requisições ao núcleo da árvore. A resposta a toda essa sequência de requisições volta à origem do pedido, isto é, ao assinante que deseja selecionar um canal. Quando uma confirmação da requisição de adesão a árvore e criação do fluxo é recebida por um comutador intermediário, ele adiciona a interface através da qual a mensagem foi recebida a uma entrada existente do cache de transmissão, ou então, cria uma nova entrada, se esta ainda não existir, para o grupo de multicast. Nessa volta, as tabelas de roteamento multicast vão sendo atualizadas. Quando um comutador recebe um pacote de dados endereçado ao grupo de multicast, ele simplesmente retransmite o pacote através de todas as interfaces de saída, como especificado pela entrada do cache de transmissão para o grupo.

A principal caraterística dessa estratégia de construção da infra-estrutura é que ele é “iniciado pelo receptor”, isto é, receptores que desejam unir-se a um canal buscam um ramo da árvore (ou o seu núcleo) e ligam-se a ele, sem qualquer participação dos transmissores.

---

[21] A configuração do ambiente é feita previamente durante a inciação do serviço de multicast. Isto é, informações de roteamento unicast existentes são aqui utilizadas pelo método de roteamento multicast.

5.2.1.2 Gerenciamento de Canais

O gerenciamento de canais foi modelado, na arquitetura do sistema de distribuição, pela classe *ChannelManager*, como mostra a Figura 5.3. Tanto nos servidores, quanto nos clientes, instâncias desta classe atuam junto aos gerenciadores de grupo, sendo responsáveis pelo armazenamento de informações sobre os canais oferecidos, no caso dos servidores, e canais assinados pelo cliente. No gerenciador do serviço, como mostrado na Figura 5.6, o componente responsável pelo gerenciamento de canal é representado pela classe *CentralChM*. Essa classe atua junto a agência de gerenciamento de grupo, sendo responsável por armazenar informações sobre todos os canais oferecidos no sistema. As informações armazenadas por essas instâncias correspondem a um conjunto de tuplas do tipo {*canal, endg, enc, cs*}, onde *canal* é o nome do canal, *endg* é o endereço de grupo associado ao canal, *enc* é o endereço do núcleo do canal, e *cs* é a categoria de serviço associada ao canal. Os métodos *addChannel(…)* e *removeChannel(…)* permitem a inclusão e exclusão de tuplas do conjunto gerenciado pela instância de *ChannelManager*, enquanto o método *getChannelInfo()* permite a obtenção de subconjuntos de tuplas com base em algumas informações, como por exemplo, informação sobre a categoria de serviço associada aos canais.

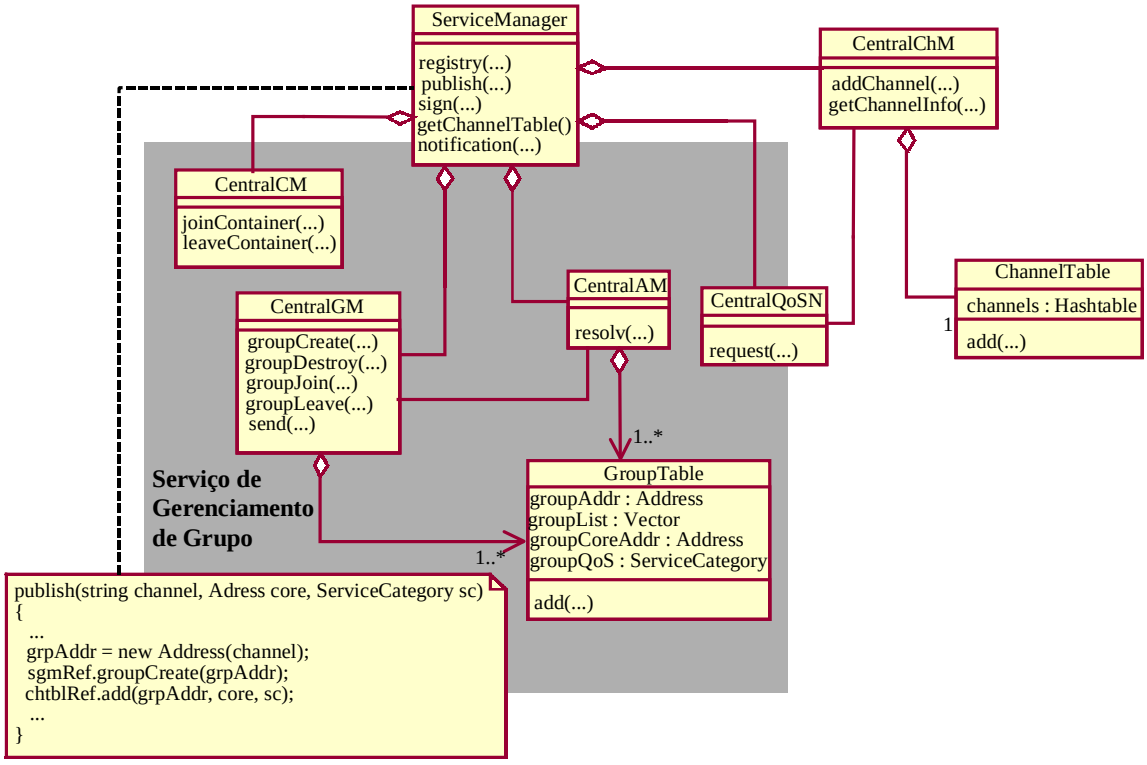

Figura 5.6: Hierarquia de classes do gerenciador do serviço

5.2.1.3 Gerenciamento de Grupo

Como já mencionado, o modelo de gerenciamento de grupo adotado foi o centralizado. O MSB centraliza as informações de composição do grupo e resolução de endereço. GMA provê as funções para o gerenciamento do grupo, controlando o acesso às informações do grupo, bem como a entrada e saída de participantes ao grupo. Como na implementação do MARS em uma rede IP multicast sobre ATM [**Armitage 96**], o MSB mantém uma base de dados para o gerenciamento e resolução de endereços de multicast. A classe *CentralGM* representa a modelagem dos serviços de gerência de grupo. A resolução de endereço é modelada pela classe *CentralAM*, através do método *resolv(…)*. A estratégia adotada para resolução no gerenciador do serviço foi a resolução por mapeamento direto através de tabelas de conversão.

5.3 Plataforma de Prototipação

Devido ao número relativamente pequeno de comutadores ATM instalados em nosso *test-bed*, recorremos a uma técnica de prototipação usando comutadores virtuais implementados como objetos simulando o comportamento de comutadores reais, tal como proposto em [**Li et al 97**]. Um comutador virtual, responsável por simular as funções lógicas de um comutador programável, é representado por uma thread em um servidor RMI (Remote Method Invocation) [**Sridharan 97**]. Esse comutador virtual, como mostrado na Figura 5.7, consiste de uma tabela de roteamento, uma interface de controle, utilizada para sinalização de controle entre o gerenciador do serviço e os comutadores, uma unidade de controle, responsável por interpretar os pedidos de estabelecimento de conexão, adicionar novas rotas a tabela de rotas, e iniciar os programas de usuário, nas portas de saída adequadas, um repositório de programas (por exemplo, um banco de filtros) e duas interfaces de sinalização (uma para as portas de entrada, e outra para as portas de saída), responsável por fornecer toda a interface de sinalização entre os comutadores virtuais (*connect/accept, request, join, …*).

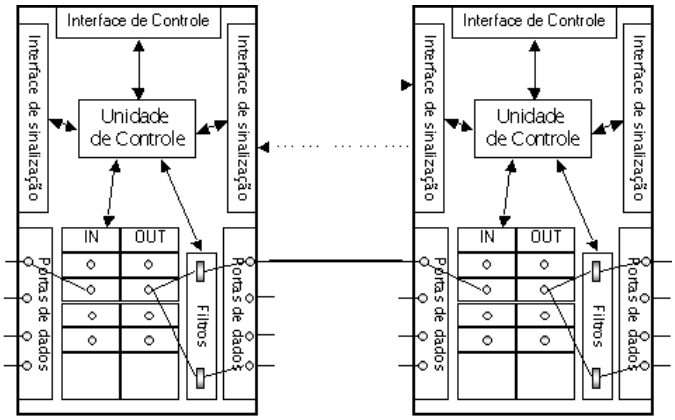

Figura 5.7: Estrutura dos comutadores virtuais

A plataforma de simulação é criada em um ambiente distribuído com quatro estações conectadas a um comutador ATM IBM (IBM 8260), como mostrado na Figura 5.8. A razão de usar quatro estações é fundamentada na teoria de que qualquer região em um mapa pode ser rotulada por quatro cores, de tal sorte a garantir que nenhuma região

adjacente possua a mesma cor. Portanto, para simular qualquer topologia de rede (grafo planar), os comutadores virtuais podem estar distribuídos em quatro estações[22]. Assim, é garantido que quaisquer dois comutadores vizinhos são simulados em estações diferentes e conectadas por circuitos virtuais permanentes (PVC) através do comutador ATM. Esta estratégia de simulação facilita a emulação de reserva de QoS para cada enlace em uma árvore de multicast. Como indicado na figura, o enlace entre os nós 1 e 2 é mapeado no PVC que liga o comutador virtual 1, na estação salgueiro-cip e o comutador 2 na estação tijuca-cip.

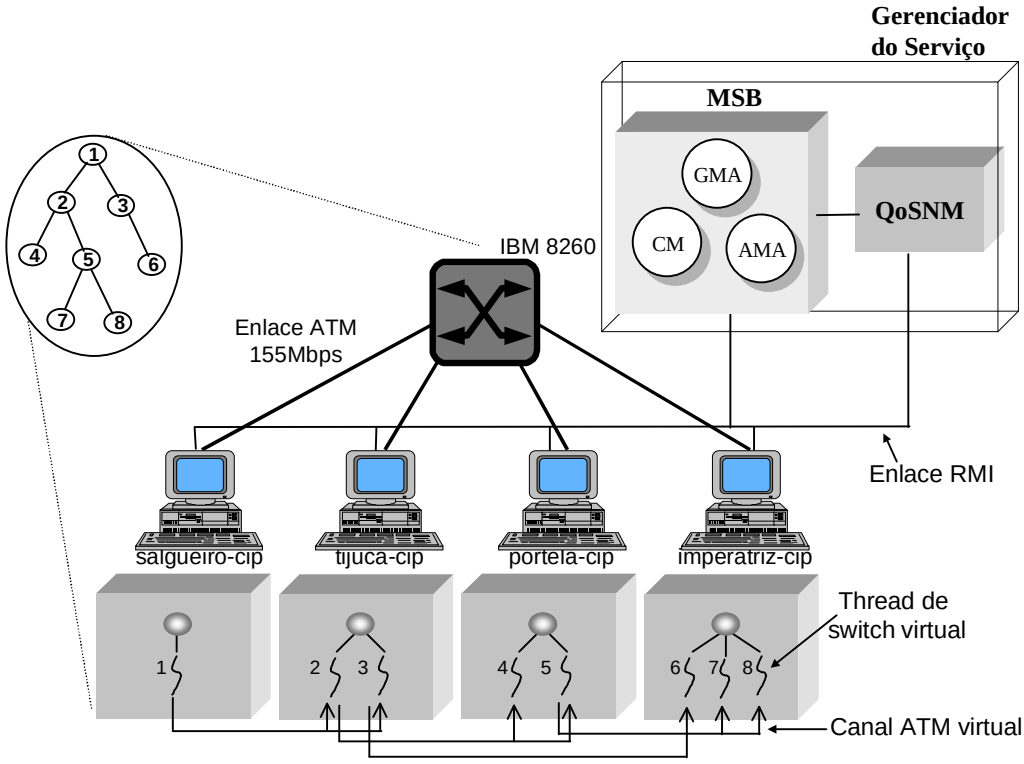

Figura 5.8: Plataforma de Simulação

Os quatro elementos funcionais da arquitetura são modelados como objetos RMI. Os componentes do Gerenciador do Serviço (GMA, CM, AMA e QoSNM) são

[22] A configuração da topologia da rede e a distribuição do comutadores virtuais nas quatro máquinas é feita previamente.

modelados como servidores RMI. Esses componentes receberão requisições de assinatura e seleção de canais dos receptores do grupo, e requisições de publicação de canais dos servidores de vídeo (clientes RMI). Os comutadores do sistema de comunicação são modelados como clientes e servidores RMI. O QoSNA (*QoS Negociator Agent*), localizado nos receptores do grupo (modelado como um cliente RMI), juntamente com o GM (também modelado como um cliente RMI) são responsáveis por iniciar o processo de estabelecimento da árvore de distribuição multicast. O processo continuará através dos objetos (RA e QoSNA) localizados nos comutadores virtuais, modelados ora como clientes RMI, e ora como servidores, até que se chegue ao núcleo da árvore ou a um comutador que já faça parte da árvore de transmissão multicast, como já explicado anteriormente. Ao receber uma requisição de adesão à árvore, o QoSNA (modelado como um servidor RMI), envia uma requisição a outro servidor RMI (RA) para o cálculo do próximo salto em direção ao núcleo da árvore. Localizado o próximo salto, o QoSNA (agora no papel de cliente) envia uma requisição de adesão ao QoSNA (servidor RMI) do próximo salto, quando o processo se repete. Assim, a conexão é estabelecida salto a salto em direção ao núcleo da árvore.

Capítulo 6

Conclusões

A maioria das soluções que têm sido apresentadas como candidatas à implementação do serviço de multicast foram projetadas tendo em mente determinadas condições específicas de infra-estrutura para distribuição de mensagens, ou características do serviço às quais são destinadas, ou ainda, a forma de gerenciamento dos grupos de usuários.

Nessa dissertação apresentou-se um estudo sobre a provisão de um serviço de comunicação de grupo. O estudo envolveu a proposição de uma especificação de um framework para a implementação de serviços de multicast, cuja principal característica é a generalidade e independência em relação aos possíveis sistemas de comunicação, aplicações e formas de armazenamento e gerenciamento de grupos. Partindo-se desse framework, foi possível configurar serviços de multicast específicos através do reuso da estrutura genérica apresentada, aliada à configuração de determinados componentes do serviço. Mostrou-se como o framework pode ser utilizado para a definição de um serviço de gerência e resolução de endereço, adotando-se tanto o modelo centralizado, como o modelo distribuído como exemplos. Mostrou-se também a definição de um serviço de construção de uma infra-estrutura de distribuição multicast.

A definição da estrutura básica do framework foi possível graças a uma análise dos vários cenários possíveis de configuração do serviço de gerência de grupo e suporte a construção da infra-estrutura de distribuição multicast, isolando-se o modelo básico, e verificando sua validação nos mais diferentes casos de utilização.

O framework foi testado na implementação de um protótipo de serviço de comunicação multicast. Um protótipo de um agente mediador de um serviço de multicast, juntamente com um framework para a provisão de qualidade de serviço [**Gomes et al 99**], foi utilizado para a provisão de um serviço de transmissão de vídeo. O agente mediador foi responsável pelos serviços de gerência e resolução de endereço, adotando-se o modelo centralizado como exemplo, e pelo serviço de suporte a construção da infra-estrutura de distribuição, adotando-se o modelo distribuído. Um ponto importante para a implementação do ambiente de distribuição multicast foi a utilização de uma plataforma de simulação baseada em comutadores virtuais, construída sobre um ambiente de comunicação de objetos (RMI) utilizando enlaces ATM.

6.1 Contribuição

A principal contribuição desta dissertação foi a **Especificação do framework para provisão do serviço de multicast.** A especificação do framework definiu a arquitetura básica de um serviço de comunicação multicast, com a descrição de cada um dos serviços componentes (gerenciamento de grupo e resolução de endereço, e suporte à construção da infra-estrutura de distribuição multicast) a partir de suas funções básicas. Foi descrita toda a estrutura do framework e seus patterns componentes utilizando-se a Linguagem de Modelagem Unificada (UML) [**Booch et al 96**]. Foram descritos ainda, como parte da descrição do framework, os principais serviços providos pelo framework (criação e destruição de grupo, adesão e saída de grupo, resolução de endereço, e construção da infra-estrutura de distribuição), através de diagramas de sequência UML.

Para a definição do serviço de multicast, foi feito um estudo sobre a provisão do serviço de comunicação multicast. A definição do serviço de multicast englobou a descrição das características gerais de um ambiente que oferece um serviço de multicast, além de um conjunto de termos, princípios, definições e características do serviço. Nessa dissertação foi apresentado, com base em um conjunto de princípios relacionados à provisão do serviço de multicast, um modelo genérico de operações de grupo, bem como os procedimentos básicos para o estabelecimento de uma comunicação multicast, baseado em um conjunto de recomendações definidas pelo ITU-T [**ITU-T X.6**] e pela ISO [**ISO 95a**, **ISO 95b**]. Com base nessa terminologia de comunicação multicast, foram traçados os requisitos mínimos para a provisão de um serviço de comunicação multicast, divididos em duas categorias: *gerenciamento de grupo* e *transmissão multicast*. A partir desses requisitos foi extraído o núcleo dos serviços providos pelo framework proposto nesse trabalho.

6.2 Extensões e Trabalhos Futuros

Como trabalho futuro, alguns pontos importantes devem ser levados em consideração. Primeiramente, o framework deve ser reavaliado com relação a restrições de qualidade de serviço. O processo de reavaliação deve consistir na integração deste framework, com um outro, especificado em [**Gomes et al 99**], para o suporte à provisão de QoS. Aliado a essa integração, uma extensão ao framework seria a inclusão de um serviço de transporte multicast, visto que a transmissão de dados deve levar em consideração alguns requisitos de qualidade de serviço exigidos pela aplicação. Assim, esse serviço seria responsável pela transmissão multicast de acordo com diferentes requisitos de qualidade de serviço. De modo geral, a principal funcionalidade exigida por um serviço de transporte multicast, incluem o suporte à troca de dados entre grupos de transmissores e grupos de receptores, de acordo com diferentes exigências de qualidade de serviço das aplicações (requisitos de qualidade dos dados transmitidos podem variar e dependem da aplicação e dos receptores).

O transporte dos dados multicast poderia ser feito de acordo com duas estratégias: garantido, ou de melhor esforço. No serviço de transporte garantido as especificações da qualidade de serviço exigidas pela aplicação seriam totalmente obedecidas, podendo ser classificado em transporte de tempo real, onde garantia de retardos mínimos é de fundamental importância, ou transporte confiável, utilizado por aplicações onde os requisitos de confiabilidade na transferência são mais importantes que os requisitos relacionados ao retardos sofridos na transmissão dos dados. No serviço de melhor esforço ("*Best Effort*"), nenhuma garantia seria provida ao transporte multicast, e a aplicação trabalha em função dos recursos disponíveis no sistema. A estratégia de transmissão multicast seria outro ponto de configuração do framework

Outra extensão ao framework seria a especificação de uma estrutura de configuração conjunta, isto é, um modelo para a configuração relacionada dos vários componentes do framework para a provisão do serviço de multicast. Embora os pontos de flexibilização do framework lhe concedam uma característica altamente desejável de configurabilidade, tais pontos não são totalmente independentes entre si, o que dificulta a utilização do framework por um projetista de sistemas. Por isso, um modelo para configuração do serviço como um todo seria altamente desejável.

Apesar de, neste trabalho, ter-se tido a preocupação constante em apresentar o Framework para Provisão do Serviço de Multicast de maneira precisa, os padrões de estrutura e comportamento delineados pelo framework não foram formalmente descritos. Além disso, a notação UML é limitada no que se refere à distinção entre descrições de famílias de arquiteturas e descrições de configurações específicas de uma arquitetura, sendo mais adequada a essas últimas. Esta característica é especialmente crítica na modelagem do framework proposto, onde se tem em mente exatamente a descrição de famílias de arquiteturas. Embora a notação UML ofereça elementos que permitam a descrição, através de diagramas, de alternativas de projeto com relação a uma arquitetura específica modelada por esses frameworks, isto não significa que esses diagramas capturam adequadamente todas as variedades de arquiteturas modeladas por eles. A utilização de uma linguagem formal de especificação como, por exemplo, a

linguagem de especificação de arquiteturas Wright [**Allen 97**], em conjunto com a descrição apresentada neste trabalho, permitiria uma maior precisão na descrição dos padrões de estrutura e comportamento definidos pelo Framework para Provisão de Serviço de Multicast.

Finalmente, reconhecendo o fato comum de que bons frameworks são resultado de muita interação e refinamentos sucessivos, é necessário a aplicação e implementação de mais casos de uso para uma melhor validação e refinamento do framework proposto. É importante destacar aqui a necessidade de aplicação do framework no desenvolvimento de um serviço multicast em um ambiente real de comunicação, uma vez que até o momento, ele só foi aplicado ao desenvolvimento de um protótipo construído sobre uma plataforma de simulação.

Apêndice A

Linguagem de Modelagem Unificada (UML)

UML (*Unified Modeling Language*) [**Booch et al 96**] é uma linguagem usada para especificar, visualizar, e documentar os artefatos de um sistema orientado a objetos em desenvolvimento. UML representa a unificação das notações de Booch, OMT, e Objectory, bem como as melhores idéias de um número de outras metodologias. Unificando as notações usadas por esses métodos orientados a objetos, UML provê a base para um padrão de fato no domínio de análise projeto orientado a objetos [**Quatrani 98**].

UML provê uma combinação de semântica, sintaxe, notações e um meta-modelo para o projeto de sistemas orientados a objeto. Esta linguagem prevê a utilização de múltiplas, diferentes e concorrentes visualizações para a arquitetura de um sistema, cada uma delas se concentrando em uma abstração específica. O modelo de visualização 4+1 [**Quatrani 98**] organiza a descrição da arquitetura de um software usando 5 visões concorrentes, cada uma voltada a um conjunto específico de interesses. São elas: Visualização de Uso de Caso (*Use Case View*), Visualização Lógica (*Logical View*), e Visualização de Componentes (*Component View*), Visualização de Processos (*Process*

View), e Visualização de Desenvolvimento (*Deplymentt View*). Descreveremos rapidamente as duas primeiras visualizações por serem mais usadas nesse trabalho.

A primeira descreve o Serviço como visto do ponto de vista externo, ou seja, focalizando as transações específicas realizadas pelo Serviço para atender às solicitações de seus usuários. O refinamento dos Casos de Utilização leva aos *Diagramas de Seqüência*, que definem as colaborações entre os objetos do modelo, tendo o eixo do tempo como referência, para a implementação das soluções aos Casos de Utilização.

A segunda visualização, a Lógica, é voltada para os requisitos funcionais do sistema (o que o sistema deve prover em termos de serviços para seus usuários). A visualização lógica descreve o Serviço como um conjunto de pacotes de classes e seus relacionamentos. Seu refinamento leva à hierarquia de classes do Serviço.

Mostraremos a seguir os dois principais modelos de diagramas UML utilizados nesse trabalho. Maiores detalhes sobre a metodologia e a notação podem ser encontrados e em [**Booch et al 96**, **Quatrani 98**, **Rational 97**].

A.1 Diagrama de classes

Diagramas de classes descrevem classes, suas estruturas internas e seus relacionamentos com outras classes. A Figura A.1 mostra a notação UML para classes abstratas e concretas. Um *classe* é denotada por um retângulo dividido em três partes, com o nome da classe localizado na parte superior do mesmo. Os *atributos* da classe aparecem abaixo do nome da classe, e os *métodos* aparecem abaixo dos atributos. Os tipos dos atributos, bem como dos argumentos e valores de retorno dos métodos, são opcionais. Na notação UML, o nome do tipo aparece sempre após o nome do método (no caso de valores de retorno), atributo ou argumento. Classes e métodos *abstratos* são grafados em itálico. Especificamente no presente trabalho, são utilizadas reticências para representar os atributos, métodos, argumentos e tipos considerados como não essenciais para a compreensão dos frameworks e patterns apresentados.

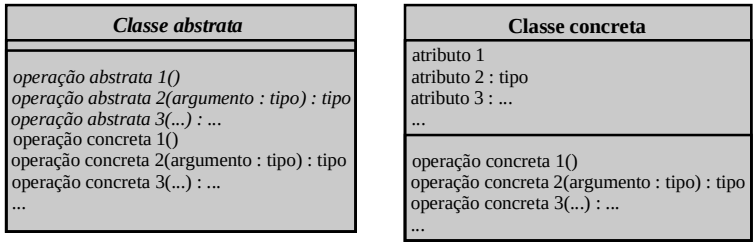

Figura A.1: Classes UML

A Figura A.2 mostra os vários tipos de relacionamentos entre classes. A herança entre classes (*relacionamento de especialização*) é denotada por uma seta com a ponta em triângulo, conectando a subclasse à classe pai. Uma linha simples é utilizada para representar um *relacionamento de associação* entre classes. O relacionamento de associação é usado para indicar que as instâncias das classes associadas podem possuir referências umas às outras, o que permite a troca de mensagens entre as instâncias dessas classes. Um *relacionamento de agregação* (também conhecido como "*parte-todo*") é representado por uma linha com um losango próximo à classe agregadora. O relacionamento de agregação é usado para indicar que as instâncias da classe agregadora são fisicamente construídas a partir de instâncias da classe agregada, ou que a classe agregadora contém logicamente (através de referências) instâncias da classe agregada. A distinção semântica entre os relacionamentos de agregação e de associação é, por vezes, muito tênue. Para tentar melhor diferenciar ambos os relacionamentos, o presente trabalho adota uma representação em que instâncias de uma classe agregada só podem ser referenciadas por uma única instância da classe agregadora. Já nos relacionamentos de associação, múltiplas instâncias de uma classe podem referenciar uma mesma instância da outra classe do relacionamento. Casos particulares de relacionamentos de associação e agregação são os *relacionamento recursivos*, que envolvem instâncias de uma mesma classe. Outro relacionamento de interesse é o *relacionamento de dependência*, representado por uma linha tracejada, com uma seta simples apontando para a classe provedora, da qual a outra classe do relacionamento depende para prover certos serviços. Tais serviços incluem, por exemplo, a criação de instâncias da classe provedora, ou o uso da classe provedora como tipo de um argumento ou valor de retorno de um método da classe dependente.

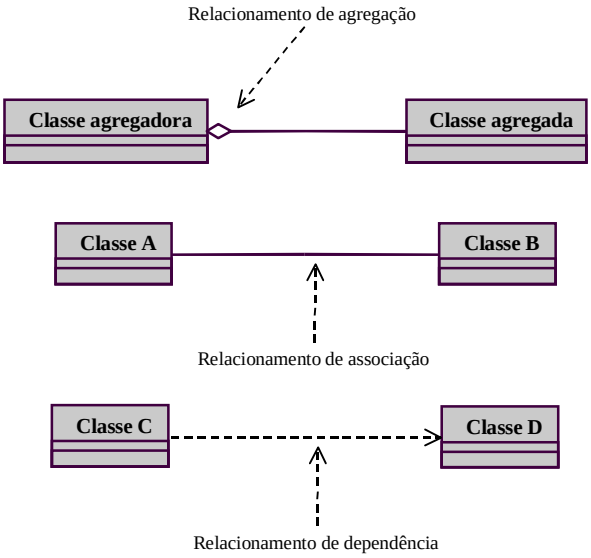

Figura A.2: Relacionamentos UML

Como ilustra a Figura A.3, relacionamentos de associação e agregação podem ser definidos de forma mais precisa, através do uso de *adornos*. O adorno de *cardinalidade* especifica quantas instâncias de uma classe podem estar relacionadas com uma única instância de uma outra classe. Este adorno fica sempre próximo à classe da qual se quer especificar a cardinalidade. Alguns exemplos de adornos de cardinalidade são:

- 1 Exatamente um
- 0..* Zero ou mais
- 1..* Um ou mais
- 0..1 Zero ou um
- 1..* Um ou mais
- 5..8 Intervalo específico (5, 6, 7 ou 8)
- 4..7,9 Combinação (4, 5, 6, 7, ou 9)

Outro adorno bastante utilizado neste trabalho é o de *navegabilidade*. Um adorno de navegabilidade indica a direção de um relacionamento, e é representado por

uma seta simples junto ao mesmo. Este adorno fica sempre próximo à classe cujas instâncias são referenciadas pelas instâncias da outra classe do relacionamento. A ausência de adornos de navegabilidade indica que o relacionamento é recíproco, ou seja, instâncias de ambas as classes podem se referenciar mutuamente.

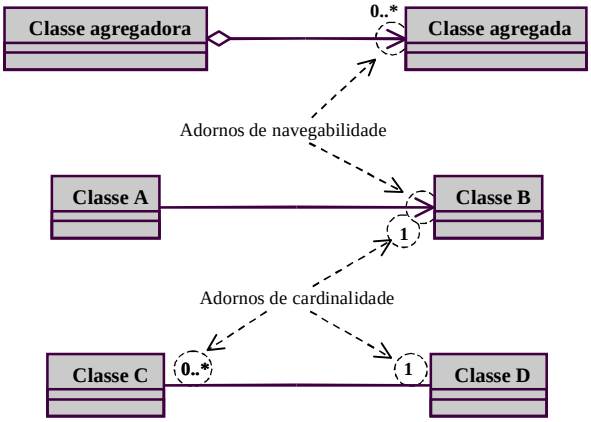

Figura A.3: Adornos UML

Outros elementos secundários podem ser encontrados em um diagrama de classes que segue a notação UML; dentre eles, *nomes de papéis* e *anotações* são os de maior interesse para o presente trabalho. As extremidades de um relacionamento (onde as classes se conectam) são denominadas, na notação UML, de *papéis* do relacionamento. *Nomes de papéis* podem ser usados para denotar o propósito pelo qual uma classe se relaciona com outra. O nome de um papel é colocado próximo à classe que exerce este papel em um relacionamento. Uma *anotação* é um campo livre que pode conter qualquer tipo de informação, desde dicas até trechos de código. Essas anotações podem ser ligadas a outros elementos de um diagrama através de *âncoras de anotação*, representadas por linhas tracejadas. Um exemplo de um diagrama de classes que utiliza todos os elementos apresentados nesta seção é apresentado na Figura A.4.

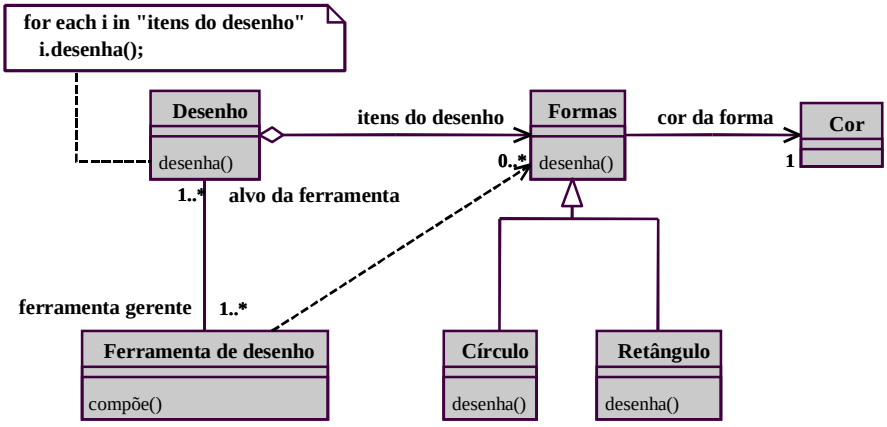

Figura A.4: Diagrama de Classes exemplo

A.2 Diagramas de Seqüência

Diagramas de seqüência mostram as interações entre instâncias de classes (através do disparo de métodos) dispostas em uma seqüência temporal. Uma instância de uma classe é representada por um retângulo contendo o nome da instância, o nome da instância e da sua classe, ou somente o nome da classe. A Figura A.5 mostra um diagrama de seqüência associado ao diagrama de classes da fig. A linha do tempo inicia-se na parte superior e segue em direção à parte inferior de um diagrama de seqüência. O disparo de métodos entre instâncias de classes é representado por setas que apontam da instância disparadora do método para a instância onde o método será disparado.

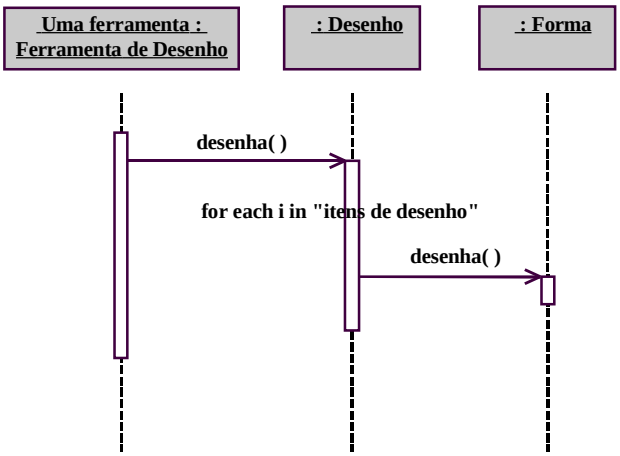

Figura A.5: Diagrama de sequência UML

Apêndice B

Modelo de Referência Unificado

Nos últimos anos, tem-se observado uma crescente demanda por serviços de telecomunicação que ofereçam suporte ao transporte de dados para múltiplos usuários. Um aspecto a ser tratado pelo serviço de multicast está relacionado ao fato de que diferentes aplicações possuem diferentes exigências de comunicação, como por exemplo, níveis de confiabilidade.

Uma alternativa que tem sido considerada em recentes propostas é a de oferecer um único serviço configurável de acordo com a necessidade das aplicações. A principal característica dessas propostas é a definição, através da tecnologia de OO, de pequenos componentes de protocolo (objetos) reutilizáveis. Através da conexão entre esses componentes é possível construir implementações particulares de protocolos e até de sistemas de comunicação inteiros, cujas estruturas são descritas através do grafo resultante da interligação dos seus componentes, denominados *grafos de protocolos*. Unidades de informação trafegam através desse grafo, e a cada unidade, é feito um processamento relacionado àquela unidade. Assim, se adequadamente configurados, os componentes podem servir como base para a configuração de protocolos e ambientes de comunicação.

Uma conseqüência direta da alternativa acima é que, como os diferentes requisitos são oriundos das aplicações ou de serviços de alto nível, parte da responsabilidade de implementação e, principalmente, configuração dos protocolos recairá sobre os analistas, projetistas e programadores das próprias aplicações. Essa "programabilidade" deverá ser uma característica básica em qualquer ambiente distribuído moderno. O projeto, implementação, e configuração de serviços de comunicação deverá se tornar tão corriqueiro quanto o projeto da aplicação em si, estando ambas as tarefas interligadas e completamente integradas.

Embora a tecnologia de OO seja reconhecidamente eficaz na modelagem e implementação tanto das aplicações quanto dos componentes de protocolos de comunicação, ainda não se encontram trabalhos que tratem detalhadamente de um modelo unificado de utilização da tecnologia OO na programação de aplicações e implementação/configuração de protocolos e ambientes de comunicação.

Com o intuito de preencher a lacuna apresentada acima é que surgiu o Modelo de Referência Unificado para Arquitetura de Protocolos e Programação de Aplicações Multimídia, que permite, através de um modelo de objetos, a construção e configuração de ambientes flexíveis no que diz respeito à adequação das aplicações e protocolos às suas características específicas de comunicação.

O restante do capítulo dedica-se à apresentação desse modelo. Porém, não serão considerados todos os conceitos pertinentes a ele, por não ser esse o escopo do trabalho. O objetivo é enfatizar a aplicação dos patterns para comunicação de grupo no modelo, disponibilizando os conceitos básicos necessários para compreensão dos capítulos seguintes.

O Modelo de Referência Unificado é dividido em duas partes. A primeira, denominada Visão Computacional, corresponde a uma especificação completamente independente de aspectos de distribuição e concorrência dos objetos, sendo a chave para a integração dos vários níveis de abstração nos quais o modelo como um todo pode ser aplicado. A segunda, denominada Visão de Engenharia, é baseada numa seqüência de

refinamentos, cada qual relacionado a um nível diferente de distribuição e concorrência dos objetos, que guiam a construção da infra-estrutura de comunicação e processamento dos níveis de abstração subseqüentes.

B.1 Visão Computacional

A Visão Computacional oferece uma descrição abstrata de alto nível, válida em quaisquer dos níveis a serem definidos, e que pode ser visualizada na Figura B.1.

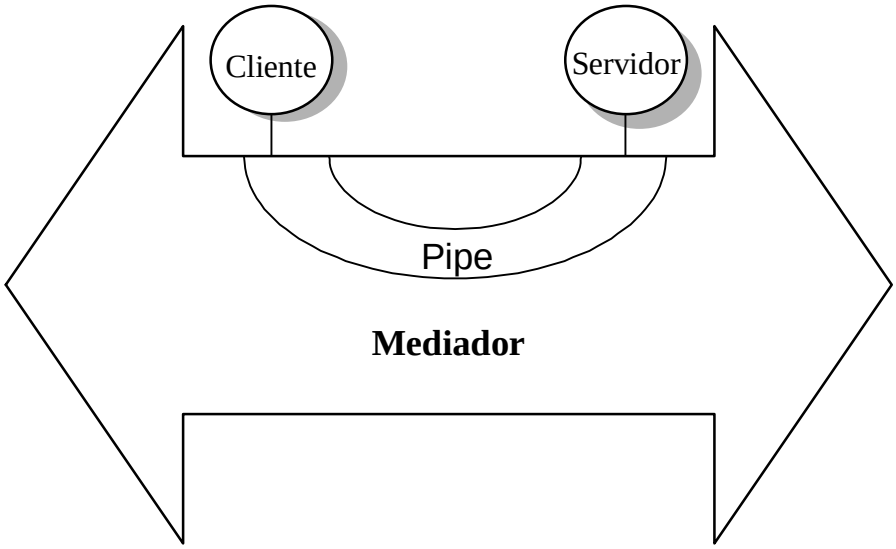

Figura B.1: Componentes da Visão Computacional.

Clientes e servidores correspondem a objetos, ou máquinas virtuais, capazes de executar instruções que modificam o estado interno dos mesmos. Estes objetos interagem através de trocas de mensagens, que constituem o mecanismo central de todo o ambiente definido pela Visão Computacional. Mensagens se prestam a várias funções no modelo, dentre elas, a de acionamento de objetos para a execução de instruções e a de transporte de valores entre objetos.

Mensagens são trocadas pelos objetos através de uma entidade abstrata denominada mediador. Considerando o caráter abstrato da Visão Computacional, trocas de mensagens são processos puramente conceituais, que poderão ser materializados por

diferentes mecanismos, dependendo do nível da Visão de Engenharia considerado, sendo o mediador a entidade responsável por essa materialização. Define-se ambiente de processamento virtual como sendo composto de um mediador e os objetos (ou máquinas virtuais) que o utilizam.

O mediador é composto de um núcleo, que permite a troca de mensagens entre objetos, e alguns objetos internos ao próprio mediador que, juntamente com o núcleo, oferecem um conjunto de serviços básicos. Esses serviços básicos podem ser obtidos por objetos clientes como se fossem serviços oferecidos por um objeto servidor comum, acima do mediador.

Dentre os serviços básicos mais importantes, encontra-se o pipe. Um pipe corresponde a uma associação explícita entre objetos para troca de informação ao estilo "memória compartilhada".

Objetos podem ser compostos, internamente, por grafos de conexões entre objetos de granularidade mais fina e pipes. Esse tipo de composição, exemplificada na Figura B.2, permite uma organização lógica e hierárquica dos protocolos de comunicação.

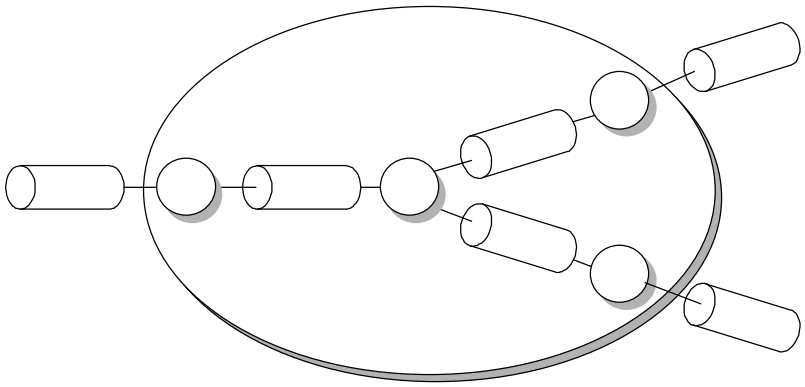

Figura B.2: Composição de objetos e *pipes*.

Considerando a necessidade de provisão de diferentes qualidades de serviço de troca de informação para os diversos tipos de mídia existentes, o mediador oferece suporte à definição de pipes especiais denominados MediaPipes. MediaPipes podem ser

estabelecidos entre cada conjunto de objetos sobre o qual se deseja exercer controle. Para haver o estabelecimento de um MediaPipe entre os diversos objetos que desejem estabelecer uma conversação é necessário um esquema de endereçamento de grupo, e possivelmente, o mapeamento desse esquema de endereçamento para os vários endereços dos objetos. Nesse contexto, um serviço de gerência e resolução de grupo do framework de multicast pode ser aplicado.

O esquema de endereçamento e consequentemente, os mecanismos de gerência e resolução de grupo são totalmente dependentes do nível de distribuição, e portanto, assuntos da visão de engenharia.

B.2 Visão de Engenharia

A Visão de Engenharia relaciona um determinado ambiente de processamento virtual a recursos computacionais disponíveis em um possível sistema de processamento e comunicação real. Na Visão de Engenharia, recursos relacionam-se às noções de thread, processo, máquina e rede.

Um determinado ambiente de processamento virtual é dito um cluster quando as atividades de todos os objetos e do mediador ocorrem todas, sempre, numa mesma thread. Quando tais atividades ocorrem todas, sempre, num mesmo processo, o ambiente de processamento virtual é chamado cápsula. Uma cápsula tem um espaço de endereçamento único e uma ou mais threads de execução. Um ambiente de processamento virtual é dito uma camada, quando objetos estão relacionados a uma mesma camada ou sub-camada de protocolo.

A troca de informação nos níveis de cluster e cápsula é realizada por intermédio de buffers entre os objetos, que juntamente com os mecanismos do mediador que permitem a passagem de mensagens entre objetos, correspondem aos pipes. Em camadas, a comunicação é sempre horizontal, sendo que, elas podem ou não ocorrer numa mesma máquina.

B.2.1 Modelo Específico do Nível de *Cluster*

Sendo os clusters associados a uma única thread, não há paralelismo interno neles. Assim, o núcleo do mediador, em geral, materializa-se no próprio ambiente de execução de uma linguagem de programação orientada a objetos, que fornece mecanismos para a execução de instruções por intermédio do disparo de métodos ao estilo de chamadas de procedimentos.

Associado a um cluster existe sempre um único objeto composto mais externo, denominado objeto cluster, que serve de base para a definição do nível de cápsula.

Como o mediador ao nível de cluster materializa-se no próprio ambiente de execução da linguagem de programação, é necessário uma infra-estrutura para o tratamento de comunicação de grupo, como gerência e resolução de endereços, visto que para o ambiente de execução da linguagem, objetos somente são referenciados individualmente. Assim, o framework pode aqui ser aplicado para o mapeamento de endereços de grupos lógicos a referências aos objetos pertencentes ao grupo.

B.2.2 Modelo Específico do Nível de Cápsula

Cápsulas são ambientes de processamento virtual que exibem paralelismo interno. A interação entre objetos no nível do cluster é desconhecida pelo mediador do nível da cápsula. Objetos cluster, nesse nível, são definidos como as unidades de concorrência da cápsula. Logo, pode-se definir uma cápsula como um conjunto de clusters e um mediador através do qual eles podem interagir, como mostra a Figura B.3.

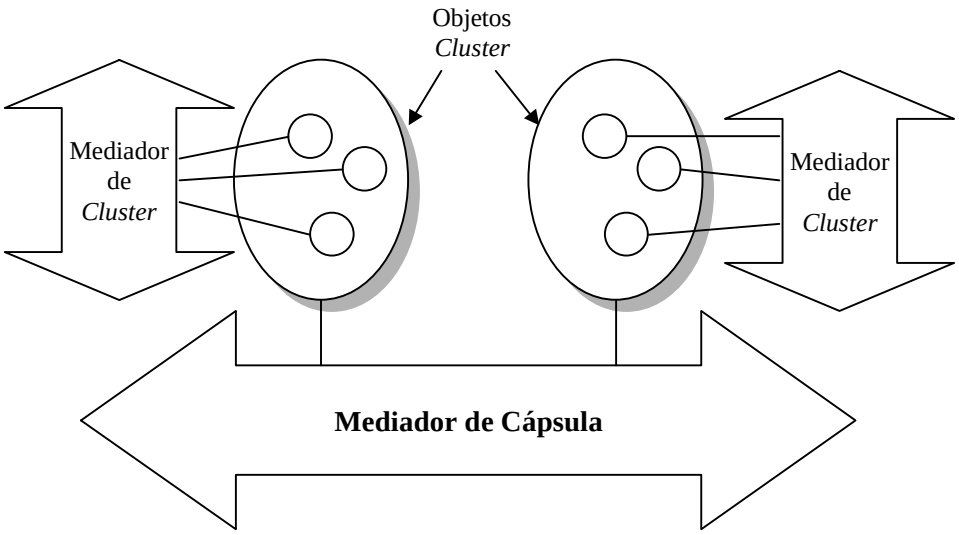

Figura B.3: Cápsula como conjunto de clusters.

Um objeto cluster pode atender, concorrentemente, a diversas solicitações de execução de instruções, criando para cada uma delas (solicitações) uma nova thread.

Pipes são, no nível de cápsula, objetos compartilhados, cujas instruções podem ser solicitadas concorrentemente pelas threads dos objetos que trocam informações através do pipe.

Mais uma vez, no nível de cápsula, é necessário um ambiente para a manipulação do endereçamento de grupo. É preciso criar um esquema de endereçamento para possibilitar a comunicação de objetos no nível de cápsula, e mais uma vez, o framework também pode ser aplicado, de acordo com os patterns de gerenciamento de grupo, mostrados no capítulo 4.

B.2.3 Modelo Específico do Nível de Camada

Nas regras gerais de estruturação em camadas definidas pelo modelo de referência OSI, a comunicação horizontal entre entidades pares é definida através da especificação dos serviços que uma camada (fornecedora) oferece a entidade do nível superior (solicitante e acolhedor). Essa especificação é baseada nas primitivas trocadas entre solicitante e fornecedor (i), e acolhedor e fornecedor (ii). As especificação de um

serviço, como um todo, contém as primitivas (i) e (ii), e seu relacionamento ao longo do tempo.

No modelo, são definidos dois níveis totalmente distintos, um relativo somente à comunicação horizontal, e outro relativo à comunicação vertical. A definição do nível de camada será tal que a comunicação horizontal ofereça uma abstração totalmente independente de primitivas trocadas verticalmente e a comunicação vertical mantém a estrutura de primitivas definidas pelos protocolos, permitindo a interoperabilidade com implementações já existentes.

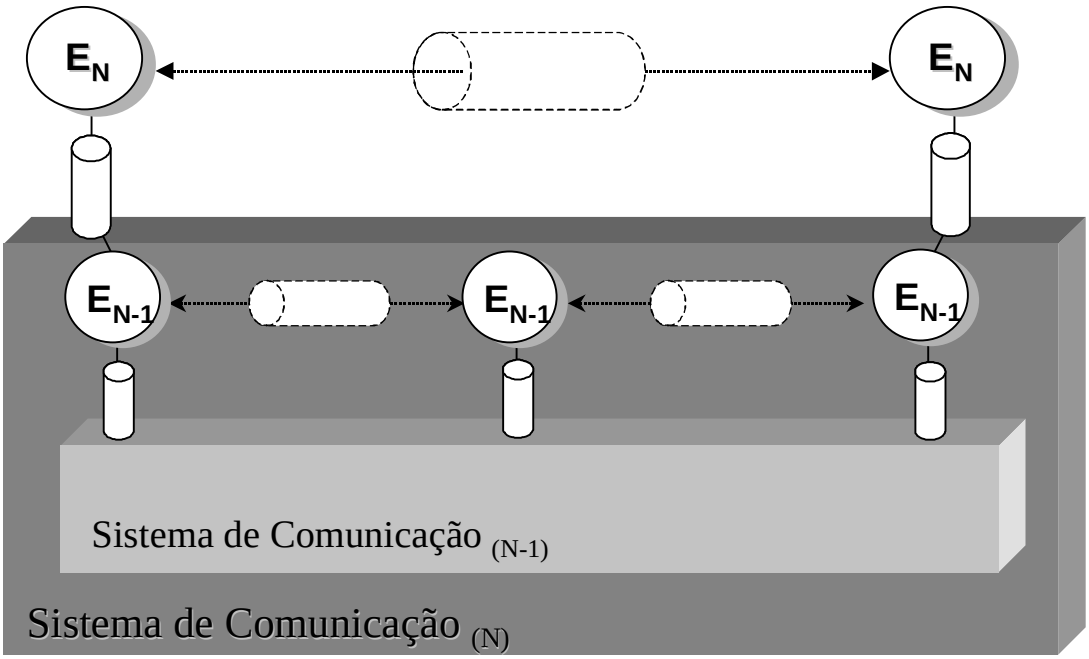

Figura B.4: Arquitetura genérica de comunicação

O estabelecimento de um *MediaPipe* horizontal na camada N é baseado no estabelecimento de uma conexão através da camada N-1, seguida da solicitação do estabelecimento de *MediaPipe* propriamente dito sobre essa conexão. O pedido de estabelecimento de conexão à camada N-1 provoca pedidos de estabelecimento de conexão à camada N-2, e assim sucessivamente, da forma tradicional.

Aqui, também de faz necessário um esquema de resolução de endereços de grupo, no momento do estabelecimento de conexões. É preciso mapear um endereço de

grupo do nível N, para o endereço correspondente do nível N-1, quer seja ele também um endereço de grupo, ou endereços dos membros do grupo, no nível N-1.

. Bibliografia

[Allen 97] Allen, R., *Formalizing Software Architecture*. PhD thesis, Carnegie Mellon School of Computer Science, 1997.

[Armitage 96] Armitage, G., "Multicast over UNI 3.0/3.1 based ATM", *Request for Comments 2022*, Novembro de 1996.

[Armstrang et al 92] Armstrong, S., Freier, A., Marzullo, K., "Multicast Transport Protocol", *Request for Comments 1301*, Fevereiro de 1992.

[ATMF 93] ATM Forum, *ATM User Netkork Interface (UNI) Specification v. 3.0*, Setembro de 1993.

[ATMF 94] ATM Forum, *ATM User Netkork Interface (UNI) Specification v. 3.1*, Setembro de 1994.

[Ballardie 97a] Ballardie, T., "Core Base Trees (CBT version 2) Multicast Routing, Protocol Specification", *Internet-Draft, draft-ietf-idmr-cbt-spec-09.txt*, Maio de 1997.

[Ballardie 97b] Ballardie, T., "Core Base Trees (CBT) Multicast Routing Architecture", *Internet-Draft, draft-ietf-idmr-cbt-arch-06.txt*, Maio de 1997.

[Ballardie et al 93] Ballardie, T., Francis, P., Crowcroft, J., "Core Base Trees (CBT): An Architecture for Scalable inter-Domain Multicast Routing", em *Proccedings of ACM SIGCOMM'93*, Págs. 85-95, Setembro de 1993.

[Bauer et al 96] Bauer, F., Varma, A., "Distributed Algorithmes for Multicast Path Setup in Data Networks", em *IEEE/ACM Transactions on Networking*, Vol. 4, Nº 2, Págs. 181-191, Abril de 1996.

[Booch et al 96] Booch, G., Jacobson, I., Rumbaugh, J. "The Unified Modeling Language for object-oriented Development", *Documentation Set Version 0.91 Addendum UML Update*, Setembro de 1996.

[Braudes et al 93] Braudes, R., Zabele, S., "Requirements for Multicast Protocols", *Request for Comments 1458*, Maio de 1993.

[Buschmann et al 95] Buschmann, F., Meunier, R., Rohnert, H., Soomerland, P., e Stal, M. *A System of Patterns*. Wiley, 1995.

[Cecilio 97] Cecilio, E.L., "Framework Genérico para comunicação de Dados Multimídia", *Relatório Técnico*, Laboratório TeleMídia, PUC-Rio, Departamento de Informática, 1997.

[Cherinton 88] Cheriton, D.; "VMTP: Versatile Message Transaction Protocol – Protocol Specification Version 0.7", Computer Science Department, Stanford University, 1988.

[Cheung et al 96] Cheung, S.Y., Ammar, M.H., "Using destination set grouping to improve the performance of window-controlled multipoint connections", em *Computer Comunications*, 19, Págs. 723-736, 1996.

[Colcher et al 98] Colcher, S., Soares, L.F.G., "Modelo de Referência Unificado para Arquitetura de Protocolos e Programação de Aplicações Multimídia", em *XVI Simpósio Brasileiro de*

Redes de Computadores (SBRC), Rio de Janeiro, 1998.

[Comer 95] Comer, D., "*Internetworking with TCP/IP, Vol 1: Principles, Protocols and Architecture*", Terceira Edição, Prentice Hall, New Jersey, 1995.

[Deering 89] Deering, S., "Host Extentions for IP Multicasting", *Request for Comments 1112*, Agosto de 1989.

[Deering et al 90] Deering, S., Cheriton, D., "Multicast Routing in Datagram Internetworks and Extended LANs", em *ACM Transactions on Computer Systems*, Vol. 8, Nº 2, Págs. 85-110, Maio de 1990.

[Deering et al 95] Deering, S., Hinder, R., "*Internet Protocol, Version 6 (Ipv6) Specification*", *Request for Comments 1883*, Dezembro de 1995.

[Deering et al 96] Deering, S., Estrin, D.L., Farinacci, D., Jacobson, V., Liu, C-G., Wei, L., "The PIM Architecture for Wide-Area Multicast Routing", em *IEEE/ACM Transactions on Networking*, Vol. 4, Nº 2, Págs. 153-162, Abril de 1996.

[Deering et al 97] Deering, S., Estrin, D.L., Farinacci, D., Jacobson, V., Helmy, A., Wei, L., "Protocol Independent Multicast Version 2, Dense Mode Specification", *Internet Draft, draft-ietf-idmr-pim-dm-05.txt*, Maio de 1997.

[Effelsberg et al 93] Effelsberg, W., Müller-Menrad, E., "Dynamic Join and Leave for Real-Time Multicast", *TR-93-056*, International Computer Science Institute, Berkeley, California. Outubro de 1993.

[Estrin et al 97] Estrin, D., Farinacci, D., Helmy, A., Thaler, D., Deering, S., Handley, M., Jacobson, V., Liu, C-G., Sharma, P., Wei, L., "Protocol Independent Multicast - Sparce Mode (PIM-SM): Protocol Specification", *Internet Draft, draft-eitf-idmr-pim-sm-specv2-00.ps*, Setembro de 1997.

[Fenner 97] Fener, W., "Internet Group Management Protocol, Version 2", *Internet Draft, draft-ietf-idmr-igmp-v2-06.txt*, Janeiro de 1997.

[Gamma et al 95] Gamma, E., Helm, R., Johnson, R., Vlissides, J. *Design Patterns: Elements of Reusable Object Oriented Software*. Addison Wesley, 1995.

[Gauthier et al 96] Gauthier, E., Le Boudec, J-Y, Oechslin, P., "SMART: A Many-to-Many Multicast Protocol for ATM", *Technical Report*, LRC, http://lrcwww.epfl.ch/PS_files/SMARTpaper.ps.gz, Agosto de 1996.

[Gomes et al 99] Gomes, A.T.A., Colcher, S., Soares, L.F.G., "Um Framework para provisão de QoS em ambientes genéricos de processamento e comunicação". Em *Anais do 17º Simpósio Brasileiro de Redes de Computadores*, Salvador, Maio de 1999.

[Guerin et al 96] Guerin, R., Kandlur, D., Willinams, D., "*MARS Extentions for RSVP*", Internet Draft, Setembro de 1996.

[Hoffman et el 96] Hoffman, D., Yavathar, R., Patki, E., "SBM (Subnet Bandwith Manager): A Proposal for Admission Control over Ethernet", *Internet Draft*, Dezembro de 1996.

[ISO 95a] ISO/IEC JTC1/SC6, *Draft text on the subject of "Multi-peer Taxonomy"*, Março de 1995.

[ISO 95b] ISO/IEC JTC1/SC6 M 9476, *First Draft of Enhanced Communications Traposrt service Definition*, Março de 1995.

[ITU-T X.6] Internation Telecomunication Union. "Multicast Service Definition", *ITU-T Recommendation X.6*, Março de 1993.

[Jia et al 96] Jia, X., Lee, C.H., Makki, K., Pissinou, N., "Efficient multicast tree algorithm in ATM networks", em *Computer Comunications*, 19, Págs. 637-644, 1996.

[Koifman et al 96] Koifman, A., Zabele, S., "RAMP: A Reliable Adaptative Multicast Protocol", http://www.tasc.com:80/simweb/papers/RAMP/full.htm, 1996.

[Kompella et al 93] Kompella, V.P., Pascale, J.C., Polyzos, G.C., "Multicasting Routing for Multimedia Communication", em *IEEE/ACM Transactions on Networking*, Vol. 1, Nº 3, Págs. 286-292, Junho de 1993.

[Kumar 96] Kumar, V., "*Mbone: Interactive Multimedia on the Internet*", New Riders Publishing, Indianapolis, 1996.

[Laubach 94] Laubach, M., "Classical IP and ARP over ATM", *Request for Comments 1577*, Janeiro de 1994.

[Li et al 97] Li, H., Pung, H.K., Ngoh, L.H. *"Active Multicast service architecture for user customized multimedia data transmission over ATM networks"*, Multimedia Systems Research Lab., National University of Singapore, 1997.

[Lu 96] Lu, G., *Communication and Computing for Distributed Multimedia Systems*, Artech House, 1996.

[Maffeis et al 95] Maffeis, S., Bischofberger, W., Mätzel, K-U, "A Generic Multicast Transport Service to Support Disconnected Operation", em *Proceedings of the 2nd USENIX Symposium on Mobile and Lacation-Independent Computing*, Abril de 1995.

[Maufer et al 97] Maufer, T., Semeria, C., "Introduction to IP Multicast Routing", *Internet Draft, draft-ietf-mboned-intro-multicast-02.txt*, Março de 1997.

[Mauthe et al 95] Mauthe, A., Hutchison, D., Coulson, G., Namuye, S. "*From Requirements to Services: Group communication Support for Distributed Multimedia Systems*", Computing Department, Lancaster University, 1995.

[Moy 94a] Moy, J., "OSPF Version 2", *Request for Comments 1583*, Março de 1994.

[Moy 94b] Moy, J., "Multicast Extensions to OSPF", *Request for Comments 1584*, Março de 1994.

[OMG 96] Object Management Group (OMG), "The Commom Object Request Broker (CORBA): Architecture and Specification". Revisão 2.0. Fevereiro de 1988.

[Partridge 93] Partridge, C., *Gigabit Networking*, Addison-Wesley, 1992.

[Pasquale et al 98] Pasquale, J.C., Polyzos, G.C., Xylomenos, G., "The Multimedia Multicasting Problem", em *ACM Multimedia Systems, Vol. 6, N 1*, 1998.

[Paul et al 96] Paul, S., Sabnani, K.K., Lin, L.C., Bhattacharyya, S., "Reliable Multicast Transport Protocol (RMTP)", em *IEEE Journal on Selected Areas in Communications, special issues on Network support for Multipoint communication*, 1996.

[Perlman 92] Perlman, R., *Interconnections: Bridges and Routers*, Addison-Wesley, 1992.

[Plummer 82] Plummer, D.C., "An Ethernet Address Resolution Protocol", *Request for Comments 826*. Network Working Group. Novembro de 1982.

[Pree 95] Pree, W. *Design Patterns for Object-Oriented Software Development*. Addison-Wesley, 1995.

[Pusateri 97] Pusateri, T., "Distance Vector Multicast Routing Protocol", *Internet Draft, draft-ietf-idmr-dvmrp-v3-04.txt*, Fevereiro de 1997.

[Quatrani 98] Quatrani, T. *Visual Modeling with Rational Rose and UML*. Addison Wesley. 1998.

[Rajagopalan 92] Rajagopalan, B., "Reliability and Scalling Issues in Multicast Communication", em *Proceedings of ACM SIGCOMM'92*, Págs. 188-198, Setembro de 1992.

[Rational 97] Rational Software Corporation; "Unified Modeling Language Notation Guide"; Version 1.0; *Rational Software Corporation*. 1997

[Reynolds et al 97] Reynolds, J., Poster, J., "Assigned Numbers", *Request for Comments 1700*, Outubro de 1997 (STD 2).

[Rezende et al 96] Rezende, J.F., Mauthe, A., Hutchison, D., Fdida, S., "*M-Connection Service: A Multicast Service for Distributed Multimedia Applications*", 1996.

[Schimidt 93] Schimidt, D.C.,, "The ADPTATIVE Communication Environment: An Object-Oriented Network Programming Toolkit for Developing communication Software", em *12th Sun User Group Conference*, San Francisco, Californica, 14-17 de Junho, 1993.

[Schimidt 97] Schimidt, D.C., "Applying Design Patterns and Frameworks to Develop Object-Oriented Communication Software*"*, em *Handbook of Programming Language*, Volume I, MacMillan Computer Publishing, 1997.

[Soares et al 95] Soares, L.F.G., Lemos, G., Colcher, S. *Redes de Computadores: das LANs, MANs e WANs às Redes ATM*. Ed. Campus, 1995. Segunda Edição.

[Sridharan 97] Sridharan, P. *Advanced Java Networking*. Prentice Hall, 1997.

[Subramaniam et al 96] Subramaniam, S., Somani, A.K., "Multicasting in ATM Networks using MINs", em *Computer Comunications*, 19, Págs. 712-722, 1996.

[Szyperski et al 93] Szyperski, C., Ventre, G.; "Efficient Multicast for Interactive Multimedia Applications", *Techinical Report TR-93-017*, International Computer Science Institute, Março de 1993.

[Talpade et al 96a] Talpade, R., Armitage, G., Ammar, M., "*Experience with Architectures for Supporting IP Multicast over ATM*",

Agosto de 1996.

[Talpade et al 96b] Talpade, R., Ammar, M., "*Single Connection Emulation (SCE): An Architecture for Providing a Reliable Multicast Transport Service*", College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, 1996.

[Talpade et al 96c] Talpade, R., Ammar, M., "*Multicast Server Archictetures for Suporting IP Multicast over ATM*", Outubro de 1996.

[Tanenbaum 92] Tanenbaum, A.S., *Modern Operating Sistemm*. Prentice-Hall, Inc., Engelwood Cliffs, 1992.

[Tennehouse et al 96] Tennehouse, D.L., Wetherall. "Towards an Active Network Architecture", Em *Keynote Session of Multimedia Computing and Networking Conference*, San Jose, CA. Janeiro de 1996.

[Thaler 97] Thaler, D., "Interoperability Rules for Multicast Routing Protocols", *draft-thaler-multicast-interop-01.txt*, Março de 1997.

[Topolcic 90] Topolcic, C., "Experimental Internet Stream Protocol: Version 2 (ST-II)", *Request for Comments 1190*, Outubro de 1990.

[Wetherall et al 98] Wetherall, J.D., Guttag, J.V., Tennehouse, D.L. "ANTS: A Tookit for Building and Dinamically Deploying Network Protocols". Em *IEEE OPENARCH'98*, San Francisco, CA. Abril de 1998.

[X200 97] ITU-T, "OSI- Basic Reference Model: the Basic Model", ITU-T Recommendation X.200, 1994.

[Zappala et al 97] Zappala, D., Braden, B., Estrin, D., Shenker, S., "Interdomain Multicast Routing Support for Integrated Services Networks", *Internet Draft, draft-zappala-multicast-routing-ar-00.ps*, Março de 1997.

[Zhang et al 93] Zhang, L., Deering, S., Estrin, D., Shenker, S., Zappala, D., "RSVP: A New Resource ReServation Protocol", em *IEEE Network Magazine*, Setembro de 1993.

***UM FRAMEWORK* PARA A PROVISÃO DE SERVIÇO DE MULTICAST EM AMBIENTES GENÉRICOS DE COMUNICAÇÃO DE DADOS**

Dissertação de Mestrado apresentada por **Marcus Antonio Almeida Rodrigues**, em 21 de maio de 1999 ao Departamento de Informática da PUC-Rio, e aprovada pela Comissão Julgadora formada pelos seguintes professores:

Luiz Fernando Gomes Soares (Orientador)
Departamento de Informática – PUC-Rio

Hugo Fuks
Departamento de Informática – PUC-Rio

Antonio Mauro Barbosa de Oliveira
CEFET-CE

Visto e permitida a impressão

Rio de Janeiro, 21 de maio de 1999

Coordenador de Programas de Pós-Graduação do
CENTRO TÉCNICO CIENTÍFICO