AVDNet6 - Framework de gerenciamento de aplicações distribuídas baseadas no protocolo IPv6

Leopoldo Wilson Malacrida Dundes

AVDNet6 - Framework de gerenciamento de aplicações distribuídas baseadas no protocolo IPv6

Leopoldo Wilson Malacrida Dundes

Orientador: *Prof. Dr. Ronaldo Celso Messias Correia*

Monografia apresentada ao Departamento de Matemática, Estatística e Computação (DMEC), da Faculdade de Ciências e Tecnologia (FCT), UNESP, como parte das atividades da disciplina Trabalho de Conclusão de Curso, necessária para a para obtenção do título de Bacharel em Ciência da Computação.

FCT-Unesp – Presidente Prudente
Janeiro/2008

Dedico essa Monografia ao Larry Page e Sergey Brin,
que trouxeram o conhecimento a um clique.
Aos meus contatos do Messenger
que contribuiram com artefatos de software,
dicas e ferramentas.
A minha vida, que acima de tudo,
é feita de conhecimento.

AGRADECIMENTOS

Agradeço a UNESP - Campus de Presidente Prudente, em especial ao Departamento de Matemática, Estatística e Computação (DMEC) por todas as condições oferecidas para a realização dessa Monografia.

Em especial as pessoas que estiveram diretamente envolvidas no desenvolvimento:

Ronaldo Celso Messias Correia,

Rogério Eduardo Garcia,

Milton Hirokazu Shimabukuro.

E não menos importante, a todos os professores do DMEC que contribuíram para minha formação intelectual:

Aylton Pagamisse,

Erwin Doescher,

Klaus Schlunzen Júnior,

Marco Antonio Piteri.

A todos os inúmeros amigos ao longo da vida acadêmica, pelos momentos de estudo e alegria compartilhados.

E de forma muito especial agradeço a minha Família, por sempre me apoiar nas escolhas da vida.

Índice

Lista de Figuras

Resumo

Essa Monografia tem como objetivo objetivo principal o desenvolvimento de um *framework*, nomeado como AVDNet6, que utilize funcionalidades do protocolo IPv6 para aumento de desempenho no processo de comunicação em Sistemas Distribuídos.

A melhoria no processo de comunicação é uma possibilidade de contribuir para o desenvolvimento de Sistemas Distribuídos mais eficientes.

A extrema facilidade e eficiência com que o IPv6 lida com sistemas distribuídos multi-usuários faz com que seja utilizado eficientemente em ambientes distribuídos, permitindo que múltiplos usuários interajam em tempo real num ambiente compartilhado. Seu uso pode aumentar substancialmente a taxa de transferência e suprir as necessidades de criação e gerenciamento de aplicações distribuídas que necessitem suportar uma grande quantidade de usuários conectados ao ambiente.

Para a elaboração desta Monografia, procedeu-se uma pesquisa bibliográfica que consistiu na consulta a livros e artigos publicados sobre os conceitos envolvidos no desenvolvimento.

PALAVRAS-CHAVE: IPv6. Sistemas Distribuídos. Ambientes Virtuais Distribuídos. Realidade Virtual.

Abstract

The ojective of this Monograph is the development of a framework, named AVDNet6, using IPv6 features to increase the performance in the communication process in networked systems.

The improvement in the communication is a chance to contribute for the development of efficient networked systems.

The extreme ease and efficiency wich the IPv6 deals with multi-user networked systems make it efficiently used in networked virtual environments, allowing multiple users interact at real time in a shared environment. The protocol can increase substantially the transfer rate and the necessity of creation and management of networked systems that need suport for many users connected in the environment.

For the development of this Monograph, a great literature search happens, including books and published articles based on the concepts of development.

KEY WORDS: IPv6. Networked Systems. Networked Virtual Environments. Virtual Reality.

CAPÍTULO 1

Introdução

1.1 Motivação

Sistemas distribuídos apresentam meios de comunicação para transportar dados entre as estações envolvidas. Se estiverem conectadas centenas ou milhares de pessoas a estes sistemas, técnicas de manutenção dos recursos de rede devem ser implementadas para que a largura de banda disponível seja o suficiente para um processamento distribuído eficaz (Eraslan et al., 2007).

Ambientes de realidade virtual simulam um ambiente real e permitem aos participantes interagirem com o mesmo, permitindo às pessoas visualizarem e manipularem representações extremamente complexas por intermédio de efeitos gráficos em 3 dimensões, som estéreo e suporte à colaboração entre múltiplos usuários espalhados pelo mundo (Netto et al., 2002).

Em sistemas distribuídos, um ambiente que proporciona interatividade entre as estações é chamado de AVD (*Ambiente Virtual Distribuído*). Distingue-se dos ambientes convencionais por fatores como compartilhamento da sensação de espaço, presença e tempo (Balikhina et al., 2002).

AVDs em larga escala devem satisfazer uma variedade de características, dentre elas: resposta rápida a novos requisitos de sistemas; manutenibilidade; interação em tempo real; fidelidade da inserção do usuário no mundo virtual; alta taxa de quadros por segundo; escalabilidade; reusabilidade; portabilidade; ajuste a novas interfaces de dispositivos de visualização (Balikhina et al., 2002).

Participantes em um AVD confiam na potencialidade da rede para que a troca de informações seja efetuada com sucesso. Um usuário, ao se mover no ambiente, deve transmitir atualizações a todos os participantes para que seja

visualizada a sua nova posição no ambiente. Similarmente, se um usuário interage com um objeto, ele deve notificar que o objeto está sendo manipulado por ele. Assim, a rede é usada para transmitir informações, sincronizar objetos compartilhados no ambiente e fazer com que o participante obtenha a máxima sensação de realidade.

Satisfazer todos esses requisitos e suportar uma grande quantidade de participantes simultaneamente é um problema complexo considerando a estrutura da Internet, baseada no protocolo de comunicação IPv4. O protocolo de comunicação utilizado não garante escalabilidade e eficiência a AVDs em larga escala, pois a entrada de novos usuários aumenta a quantidade de dados compartilhados e o nível de interação no ambiente, exigindo mais recursos de rede do que os proporcionados pelo IPv4 para manter os dados atualizados e disseminar as interações entre os participantes (Correia, 2005). A velocidade e escalabilidade nesses ambientes ficam limitados já que o protocolo IPv4 não implementa de forma nativa o *multicast* como forma de comunicação.

Muitas técnicas têm sido desenvolvidas para diminuir o tráfego de pacotes enviados à rede, porém poucas têm sido desenvolvidas para gerenciar o fluxo de informações de forma eficiente ao longo da rede. Dessa forma, essa Monografia tem motivação na necessidade de aprimorar as características inerentes a AVDs por intermédio de novas funcionalidades promovidas pela padronização do IPv6 como protocolo de comunicação. Pois, a capacidade de rede é um recurso limitado, logo deve ser cuidadosamente alocada aos fluxos gerados no ambiente.

As funcionalidades do IPv6 se adaptam melhor a sistemas distribuídos, e seu uso, pode aumentar a escalabilidade nesses tipos de ambientes (Eraslan et al., 2007).

1.2 Objetivos

Uma solução ideal de uso em AVDs é a utilização do IPv6 como protocolo de comunicação. O IPv6 é um protocolo que se ajusta perfeitamente aos requisitos de AVDs. Implementa diversas funcionalidades, sendo uma solução mais adequada se comparada ao IPv4 (Eraslan et al., 2007), dentre elas destacam-se: *multicast* nativo, roteamento mais eficiente, *anycast*, maior faixa de endereçamentos, IPSec (*IP Security Protocol*) integrado, QoS (*Quality of Service*) padronizado (Kurose and Ross, 2006).

Essa Monografia tem como objetivo principal o desenvolvimento de um *framework*, nomeado como AVDNet6, que utilize funcionalidades do IPv6 para aumento de desempenho no processo de comunicação em AVDs. O objetivo do *framework* é o desenvolvimento de um módulo de extensão da camada

de Comunicação da arquitetura AVDNet, desenvolvida por (Correia, 2005). Contudo, a generalidade do *framework* AVDNet6 permite que seu uso seja expandido a sistemas distribuídos não específicos.

Assim, pretende-se contribuir de maneira efetiva nessa área, já que AVDs são utilizados em diversos contextos, e com demanda crescente de uso (Serin, 2003), a melhoria no processo de comunicação é uma possibilidade de contribuir para o desenvolvimento de AVDs mais eficientes.

1.3 Organização da Monografia

O texto está estruturado como segue:
O Capítulo 2 aborda conceitos e características sobre Ambientes Virtuais Distribuídos, apresentando sua definição, as vantagens e as desvantagens nas topologias de comunicação usadas em AVDs, bem como meios de otimização de tráfego pela rede. No Capítulo 2 também é apresentado a arquitetura AVDNet (Correia, 2005).

No Capítulo 3 são apresentadas as características e funcionalidades do protocolo IPv6 e mudanças em relação ao protocolo IPv4. Nesse Capítulo, as funcionalidades do IPv6 são apresentadas de acordo com os objetivos de utilização e desenvolvimento do *framework* AVDNet6.

No Capítulo 4 é apresentado o *framework* AVDNet6. Seu uso, mecanismos, e modelo de desenvolvimento são detalhados nesse Capítulo.

Por fim, no Capítulo 5 são apresentadas as conclusões abragendo o trabalho como um todo e oportunidades para o desenvolvimento de possíveis trabalhos futuros sendo que então expõe-se as considerações finais.

CAPÍTULO 2

Ambientes Virtuais Distribuídos

2.1 Considerações Iniciais

O objetivo principal deste capítulo é explicitar conceitos sobre Ambientes Virtuais Distribuídos envolvidos na criação do *framework* AVDNet6, apresentando características que podem ser decisivas no desempenho destes ambientes: topologias de comunicação e otimização do fluxo de tráfego.

A arquitetura de software AVDNet (Correia, 2005) desenvolvida para criação e gerenciamento de aplicações de realidade virtual distribuída é apresentada neste capítulo.

Uma explicação dos conceitos envolvidos é de fundamental importância para a justificação das escolhas metodológicas adotadas no desenvolvimento do *framework*. Assim, nas seções a seguir, são explicados dois conceitos fundamentais envolvidos em Ambientes Virtuais Distribuídos: Sistemas Distribuídos e Realidade Virtual.

2.1.1 Sistemas Distribuídos

O uso de redes locais e da Internet está amplamente difundido, mesmo para uso doméstico (Kurose and Ross, 2006). Mas para que os recursos de rede sejam aproveitados da melhor forma possível é preciso fornecer suporte adequado de software.

Sistemas Distribuídos são constituídos por um conjunto de processos executando em computadores independentes e sem o compartilhamento de memória física entre eles (Comer, 2006). A ausência de memória compartilhada obriga a uma interação entre processadores de uma forma distinta do ambientes cen-

tralizado: ao invés de variáveis ou arquivos compartilhados, utiliza-se troca de mensagens (Abreu, 2006) .

A computação distribuída, por intermédio dos sistemas distribuídos, permite o compartilhamento de recursos, como impressora ou *scanners*; possibilita distribuir tarefas entre computadores participantes; decompõe um sistema complexo num conjunto de sistemas mais simples (Abreu, 2006). Essas características motivam, cada vez mais, investimentos nessa área. Pois, com essas características, a computação distribuída propicia diversas vantagens comparadas à computação centralizada.

Dentre as vantagens de uso de sistemas distribuídos, destacam-se a possibilidade de crescimento incremental desses sistemas, aumentado o poder computacional do ambiente distribuído por meio de inclusão de novos computadores ao sistema; a flexibilidade de uso em sistemas heterogêneos, permitindo que vários usuários em diferentes arquiteturas compartilhem dados entre si; a tolerância a falhas, permitindo que o ambiente distribuído mantenha a consistência em caso de falha de uma única máquina (Tanenbaum, 1995).

Além das vantagens citadas, tem-se, um custo inferior, que ao invés de se utilizar um computador mais potente, utilizam-se várias máquinas de menor potência para atender a demanda. E com uma quantidade maior de computadores, pode-se conseguir um melhor desempenho, implantando paralelismo na execução do ambiente distribuído (Tanenbaum, 1995).

Porém, sistemas distribuídos necessitam de requisitos específicos. Há a necessidade de sincronizar o acesso e gerenciar a concorrência entre os sistemas distribuídos, de forma a manter a coerência e consistência do ambiente (Abreu, 2006). Para isso, utiliza-se técnicas de sincronização e controle de concorrência, como ordenação de eventos, filas, relógios, prioridades, e diversos outros algoritmos para tratar desse requisito, mas que não são abordados nessa Monografia.

2.1.2 Realidade Virtual

O termo Realidade Virtual (RV) é creditado a *Jaron Lanier*, fundador da *VPL Research Inc.*, que o criou, no início dos anos 80, para diferenciar as simulações tradicionais feitas por computador de simulações envolvendo múltiplos usuários em um ambiente compartilhado (Netto et al., 2002).

O termo RV é bastante abrangente, e acadêmicos, desenvolvedores de software e pesquisadores tendem a definí-lo com base em suas próprias experiências, gerando diversas definições na literatura. Pode-se dizer, de uma maneira simplificada, que RV é a forma mais avançada de interface do usuário com o computador até agora disponível (Hancock, 1995). Trata-se de uma interface que simula um ambiente real e permite aos participantes interagirem com o

mesmo (Latta and Oberg, 1994), permitindo às pessoas visualizarem, manipularem e interagirem com representações extremamente complexas (Aukstakalnis and Blatner, 1992). RV é um paradigma pelo qual usa-se um computador para interagir com algo que não é real, mas que pode ser considerado real enquanto está sendo usado (Hand, 1994).

RV permite que o usuário navegue e observe um mundo tridimensional, em tempo real e com seis graus de liberdade (6DOF) (Latta and Oberg, 1994). Isso exige a capacidade do software de definir seis tipos de movimento: para frente/para trás, acima/abaixo, esquerda/direita, inclinação para cima/para baixo, angulação à esquerda/à direita e rotação à esquerda/à direita. Na essência, a RV é uma simulação da realidade física, na qual o indivíduo existe em três dimensões, tem a sensação do tempo real e a capacidade de interagir com o mundo ao seu redor.

A interface em RV envolve um controle tridimensional altamente interativo de processos computacionais. O usuário entra no espaço virtual das aplicações e visualiza, manipula e explora os dados da aplicação em tempo real, usando seus sentidos, particularmente os movimentos naturais do corpo humano. A grande vantagem é que o conhecimento intuitivo do usuário sobre o mundo físico pode ser transportado para o mundo virtual (Latta and Oberg, 1994). Para suportar esse tipo de interação o usuário utiliza dispositivos não convencionais, como capacetes de visualização e controle, e luvas de dados, chamadas *datagloves*. O uso desses dispositivos dá ao usuário a impressão de que a aplicação está funcionando no ambiente tridimensional real, permitindo a exploração do ambiente e a manipulação natural dos objetos com o uso das mãos (Netto et al., 2002) .

O termo Mundo Virtual é usado para denotar o mundo digital criado a partir de técnicas de Computação Gráfica (Netto et al., 2002). Uma vez que é possível interagir e explorar esse mundo por meio de dispositivos de entrada e de saída, ele se transforma em um ambiente virtual, ou ambiente de Realidade Virtual.

O desenvolvimento de um sistema de RV requer estudos e recursos ligados a percepção sensorial, hardware, software, interface com o usuário, fatores humanos e aplicações (Netto et al., 2002).

2.2 Ambientes Virtuais Distribuídos

Ambiente Virtual refere-se à tecnologia capaz de transportar um indivíduo para um ambiente diferente do real, sem movê-lo fisicamente (Capin et al., 1999), com auxílio de interfaces de Realidade Virtual. Por meio de manipulação dos orgãos sensoriais humanos e por intermédio de

um modelo computacional capaz de descrever fisicamente o ambiente virtual (Netto et al., 2002), esta tecnologia é capaz de criar qualquer ambiente possível (Capin et al., 1999).

Além, Ambientes Virtuais Distribuídos (AVDs) são sistemas que permitem que múltiplos usuários interajam em tempo real num ambiente compartilhado, mesmo que os usuários estejam distribuídos ao redor do mundo, utilizando Sistemas Distribuídos (Singhal and Zyda, 1999). AVDs lidam com questões acerca de como interagir com um objeto, uma aplicação, e até com outros usuários, simulando espaços e atividades reais por meio do uso incorporado de computação gráfica 3D e som estéreo, criando um sistema de experiência imersiva (sensação de estar dentro de um ambiente sintético criado por computador). Os objetos imaginários criados por software podem ser perfeitamente sentidos e manipulados, conduzindo o usuário a sensações multisensoriais.

Os AVDs fazem parte da junção de três tipos de áreas de aplicações da computação: Sistemas Distribuídos, Computação Gráfica e Realidade Virtual (Netto et al., 2002). Um AVD é caracterizado como (Singhal and Zyda, 1999):

Sensação de espaço compartilhado: todos os usuários têm ilusão de estarem situados em um mesmo lugar ou ambiente. O ambiente pode ser real ou fictício, desde que todos os usuários tenham a mesma sensação de características relativas ao ambiente: temperatura, clima, acústica, etc.

Sensação de presença compartilhada: os usuários conseguem identificar e perceber a presença de outros usuários por intermédio de *avatares* (representação gráfica de alguma estrutura representativa: um animal, uma pessoa, etc).

Sensação de tempo compartilhado: participantes dos AVDs devem estar preparados para perceber comportamentos de outros participantes, ou mesmo de objetos do ambiente, no tempo em que realmente eles ocorrem.

Meio de comunicação: os participantes devem ser capaz de se comunicar, seja por meio de gestos, voz ou texto. A comunicação proporciona um avançado senso de realismo ao ambiente (Leite Júnior, 2000).

Meio de compartilhamento: a principal funcionalidade de um AVD está em permitir aos participantes a possibilidade de interagir com o ambiente realisticamente, não só com o outro participante, mas também com o próprio ambiente: pegando, manipulando, movendo ou colidindo em objetos do ambiente, ou até mesmo oferecendo objetos aos outros participantes.

Ambientes virtuais têm sido usados nas mais diversas áreas do conhecimento. A variedade de características faz com que os ambientes resultem em sistemas muito complexos a medida que proporcionam uma sensação mais intensa de realidade aos participantes. São úteis em diversos contex-

tos (Leite Júnior, 2000): Experiência de estar presente (Terapia de fobias, Estética, Entretenimento); Treinamento em áreas de conhecimento específico (Cirurgia, Treinamento militar, Manutenção de equipamentos); Visualização de objetos inacabados (Arquitetura, Escoamento de fluídos, Nanosuperfícies, Modelos Tridimensionais, Urbanização).

2.3 Topologias de Comunicação

Existem esquemas específicos para organizar as possíveis conexões entre os diversos computadores participantes de uma rede. A esses esquemas dá-se o nome de topologia.

A topologia utilizada em um AVD é de fundamental importância para o gerenciamento do fluxo de dados gerado pelo AVD (Eraslan et al., 2007). A topologia de um AVD define como os dados irão trafegar entre os participantes. Assim, a topologia torna-se uma solução básica para a otimização do tráfego de AVDs (Leite Júnior, 2000) .

Os dados deve fluir com extrema consistência em AVDs, pois como todo sistema de tempo real, é necessario que toda informação seja atualizada constantemente e cada usuário deve ter uma cópia fiel de todo o estado do ambiente virtual (Netto et al., 2002). Na fase de inicialização, quando o usuário se conectar ao ambiente, o ambiente deve dispor de mecanismos para constatar a existência desse novo usuário, e o usuário deve certificar-se da existência de todos os objetos do ambiente (Eraslan et al., 2007).

Para comparação entre taxa de transmissão de diferentes topologias de comunicação, pode ser usado como medida o PDU (*Protocol Data Unit*). O PDU é uma unidade básica de transmissão de dados em um AVD. É um pacote que contém informações necessárias para gerenciar a comunicação entre os participantes do AVD. Não existe um único um formato padronizado de PDU, há PDUs para cada serviço específico em um AVD (Singhal and Zyda, 1999), por exemplo em ambientes virtuais de simulações militares, existem PDUs específicos para representar detonação, disparo de projéteis ou marcação de campos minados.

O conteúdo de um PDU é definido pelo tipo de serviço implementado no AVD. Seus campos determinam opções essenciais para o eficiência na execução do serviço, por exemplo, um PDU que representa o movimento de um tanque de guerra contém a identificação do emissor do pacote, a sua localização, sua velocidade, aceleração, vetor de direção, e outros dados dinâmicos específicos ao ambiente para que a consistência seja mantida (Singhal and Zyda, 1999).

A seguir, algumas topologias de comunicação e suas respectivas vanta-

gens e desvantagens são apresentadas: *Topologia Cliente-Servidor*, *Topologia Peer-to-Peer*, *Topologias Híbridas*.

2.3.1 Cliente-Servidor

Nesta topologia existe um servidor central que administra todo o ambiente. Não há envio de mensagens diretamente para os outros participantes do sistema, pois todo o tráfego deve passar pelo servidor central, como apresentado na Figura 2.1. Quando um novo cliente entra no ambiente, o servidor se encarrega da tarefa de incluí-lo ao ambiente. E quando uma mensagem é enviada, quem é responsável em distribuí-la é o próprio servidor.

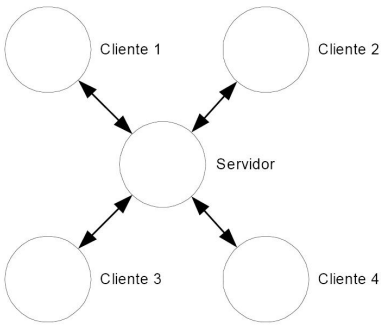

Figura 2.1: Topologia Cliente-Servidor

A principal vantagem no uso dessa topologia é a possibilidade do servidor atuar como intermediário em todas as trocas de mensagens e assim efetuar algum tipo de processamento sobre as mensagens antes de propagá-las aos demais clientes (Eraslan et al., 2007).

Assim, os servidores podem reduzir o tráfego de mensagem aplicando alguns filtros de tratamento de datagramas e informações, comprimindo múltiplos pacotes em um único ou reduzindo o tráfego redudante de mensagens. No entanto, este tipo de sistema não é escalável, não é capaz de sustentar e manter o ambiente se o número de usuários aumentar drasticamente. Nesse caso, independentemente da capacidade de processamento do servidor, o número de participantes aceitos em uma estrutura *Cliente-Servidor* é bastante inferior àquele obtido com a topologia *Peer-to-Peer* (Leite Júnior, 2000).

Além disso, as mensagens não percorrem o caminho mais curto de comunicação, pois o servidor representa um caminho a mais para o pacote percorrer até que se chegue ao destino. Por exemplo, se N usuários estão conectados ao ambiente e atualizam suas localizações enviando mensagens ao servidor, logo N*(N-1) mensagens são enviadas por atualização para que o servidor receba e repasse as atualizações a todos os outros participantes do ambiente. Assim a complexidade da topologia é $O(N^2)$ (Eraslan et al., 2007).

De qualquer forma, a abordagem cliente-servidor é relativamente simples de ser implementada e oferece uma perfeita solução para sistemas distribuídos que são baseados em poucas trocas de mensagens (Neto, 2005).

2.3.2 Peer-to-Peer

Essa topologia é caracterizada por permitir a comunicação direta entre qualquer um dos elementos da rede. Assim, qualquer mensagem de atualização específica é mandada diretamente para os outros participantes, Figura 2.2.

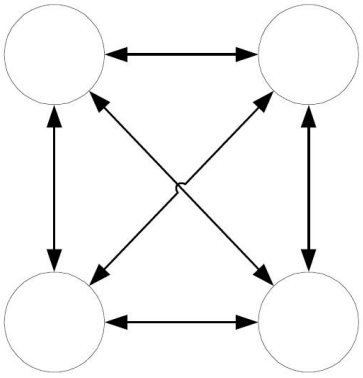

Figura 2.2: Topologia Peer-To-Peer unicast

Assim, assumindo N participantes, e se cada participante se comunica diretamente com N-1 participantes, a topologia *Peer-to-Peer* possui complexidade $O(N^2)$ (Leite Júnior, 2000). Devido ao grande número de conexões entre os participantes, a tarefa de sincronização de cópias do ambiente virtuais torna-se uma tarefa muito onerosa, além de exigir um grande esforço de busca caso o participante deseje obter informações sobre outro participante. O conjunto de medidas relacionadas a entrada e a saída de usuários de um determinado sistema é bastante complexo, pois cada participante deve estabelecer comunicação com cada outro participante.

Multicast e técnicas de roteamento mais rápidas são essenciais em sistemas de aplicações de realidade virtual distribuídas para garantir escalabilidade e confiança na comunicação estabalecida (Correia, 2005).

Uma rede baseada na topologia *Peer-to-Peer* que utilize comunicação *multicast* faz com que o número de mensagens enviadas para sincronização do ambiente virtual seja bem menor do que uma topologia unicamente *Peer-to-Peer* (Leite Júnior, 2000). A Figura 2.3 apresenta um modelo de comunicação *Peer-to-Peer multicast*.

O *multicast* em AVDs *Peer-to-Peer* evita a duplicação de pacotes desnecessários.

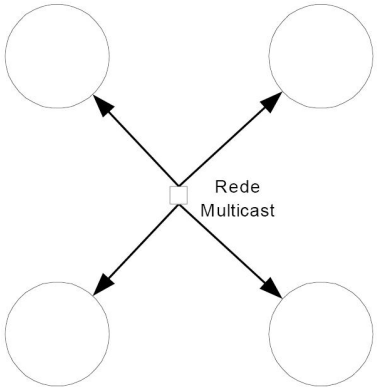

Figura 2.3: Topologia Peer-To-Peer multicast

Primeiramente, quando os usuários entram no sistema, devem se inscrever em um grupo *multicast* para se tornarem membros do ambiente, assim todos os dados do ambiente destinados a eles são mandadas para o respectivo endereço *multicast*.

Esse tipo de topologia pode suportar uma grande quantidade de usuários e controlar a banda disponível eficientemente, já que a complexidade na troca de mensagens é de O(N) (Eraslan et al., 2007), N representa o número de participantes de um AVD.

Porém, já que nessa topologia não há um repositório central, um novo usuário não pode obter conhecimento de todos os usuários presentes no ambiente. Os usuários devem gerar constantemente mensagens periódicas de *keep-alive* (constante checagem em relação ao funcionamento normal de um elemento de rede).

2.3.3 Híbridas

Topologias puramente *multicast* não são eficientes para administrar AVDs eficientemente (Leite Júnior, 2000). Primeiramente, os membros que entram no ambiente não têm conhecimento da existência dos outros usuários até que se envie uma mensagem *keep-alive* (Capin et al., 1999). Esses tipos de mensagens consomem uma banda significativa da rede e, portanto, não são eficientes e não são uma boa alternativa para serem implantados (Leite Júnior, 2000).

Além disso, usuários que se desconectam do sistema só são reconhecidos como ausentes quando uma mensagem de confirmação chegar a eles (Eraslan et al., 2007). Portanto, enquanto não chega a confirmação, permacem como usuários fantasmas (não enviam mensagens, porém recebem mensagens do ambiente).

O ideal é o uso de *Topologias Híbridas*, pois combinam as melhores car-

acterísticas das topologias descritas anteriormente (Correia, 2005). Há duas abordagens principais em relação ao emprego de *Topologias Híbridas* (Capin et al., 1999):

1- Subdivisão do mundo virtual: consiste na partição do mundo virtual em subespaços, lugares especificados, onde cada subespaço é gerenciado por um servidor específico. Os servidores estabelecem a comunicação por meio do *multicast*, assim como os clientes de cada servidor. Assim, quando um participante percorrer o ambiente, ele está na verdade se conectando e se desconectando de servidores responsáveis pela administração e gerenciamento de cada subespaço.

2- Subdivisão de participantes: consiste na geração de cópias completas do AVD em vários servidores no ambiente. O participante do AVD tem a liberdade de escolher a qual servidor se conectar, o ideal é o servidor de mais fácil acesso (Eraslan et al., 2007). Sua grande vantagem é a possibilidade dos servidores apresentarem um número igual de participantes conectados por intermédio de técnicas de balanceamento.

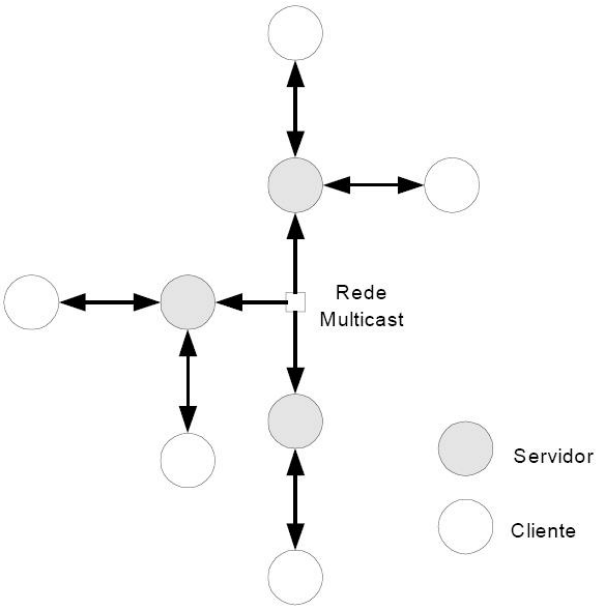

Figura 2.4: Topologia Híbrida

O uso de *Topologias Híbridas* contorna o problema típico do uso de um único servidor central (*Topologia Cliente-Servidor*) e utiliza uma estrutura integrada de comunicação que possibilita realizar a distribuição das conexões dos diversos servidores, acrescentando uma grande escalabilidade ao sistema (Eraslan et al., 2007). A representação de uma *Topologia Híbrida* de comunicação é apresentada na Figura 2.4.

O mérito do AVD nesse tipo de topologia se concentra especificamente na capacidade de processamento de cada servidor, já que eles são os respon-

sáveis pelo compartilhamento do ambiente virtual, manutenção dos objetos, comunicação e interação entre os participantes. O maior problema reside na exigência de uma estratégia de balanceamento de tráfego para tornar toda a estrutura de comunicação empregada realmente escalável (Capin et al., 1999).

2.4 Mensagens de Comunicação

Os mais conhecidos protocolos que oferecem padronização nos meios de comunicação são: IP(*Internet Protocol*), TCP(*Transmission Control Protocol*), UDP(*User Datagram Protocol*), RTP(*Real-Time Protocol*). Além dos protocolos de comunicação, para que ocorram trocas de informações entre os diversos partipantes de um AVD faz-se necessário que ocorra alguma padronização nas mensagens de comunicação utilizadas entre os participantes e o ambiente. Esses padrões devem levar em consideração todos os recursos disponíveis nos AVDs, sempre aproveitando eficientemente todos os recursos de todas as aplicações que fazem parte da rede de comunicação. Na evolução dos AVDs, dois modelos de troca de mensagens merecem destaque (Leite Júnior, 2000):

1- *DIS protocols*: o protocolo DIS (*Distributed Interactive Simulation*) consiste em um grupo de padrões desenvolvidos pelo Departamento de Defesa Americano e por entidades da indústria Americana voltados para arquiteturas de comunicação, formato e conteúdo de dados, interação e informação sobre entidades virtuais, gerenciamento de simulações, etc (Singhal and Zyda, 1999). Apesar das áreas de aplicações serem diversas, o principal emprego do protocolo destina-se a construção de simuladores militares, mais especificamente para campos de batalha. Isso só é possível graças ao amplo suporte oferecido para simulações, permitindo a utilização simultânea de diversas aplicações específicas com a análise de milhares de variáveis (Singhal and Zyda, 1999).

Um problema no protocolo DIS, em consequência direta da finalidade do protocolo para fins militares, é sua não preocupação com a interação e conversação entre participantes humanos, não existindo estruturas específicas para o suporte a este tipo de funcionalidade (Capin et al., 1999).

2- *Game protocols*: conhecidos como SRMPs (*Scalable Reliable Multicast Protocols*), são inspirados em jogos multiusuários de computadores. Busca-se a existência de cópias aproximadas do mundo virtual nos computadores de cada um dos participantes do AVD. Esses protocolos geralmente apresentam PDUs pequenas, capazes de transmitir somente as informações essenciais ao correto funcionamento do AVD, o que o torna bastante eficaz, já que produz um sincronismo entre todas as cópias do ambiente virtual distribuído

(Leite Júnior, 2000).

Os *Game Protocols* não necessitam transmitir definições completas do estado de um objeto, podendo apresentar diversos tipos de PDUs especializadas para cada operação realizada por uma determinada entidade virtual, ao contrário do protocolo DIS.

2.4.1 Otimização do Tráfego de PDUs

Um conjunto de técnicas pode ser usado para otimizar o tráfego de PDUs pela infra-estrutura de comunicação nos AVDs. Dentre várias, um destaque seria o *dead reckoning*, que é uma técnica utilizada para a predição e correção da posição e do movimento de objetos. O termo é originado da navegação marítima e sugere que a posição atual de um objeto possa ser prevista a partir da velocidade, posição e aceleração em um determinado instante (Singhal and Zyda, 1999). A trajetória pode ser prevista com um ótimo grau de precisão. Em AVDs, essa técnica foi adaptada para permitir a substancial diminuição da quantidade de informações sobre o reposicionamento de entidades que participam do ambiente virtual, diminuindo o tráfego de rede e evitando congestionamentos (Correia, 2005).

O *dead reckoning* consiste no envio periódico de posições e direções absolutas de objetos para os participantes do AVD, e cada participante receptor tem que calcular a posicão resultante do objeto baseado nos caminhos absolutos do ambiente virtual (Correia, 2005). O envio de *dead reckoning* é apenas para informar que houve alteração no estado dinâmico corrente do objeto, e enquanto o sistema não receber as informações de atualização, o participante receptor vai predizer os atributos apenas com base nas últimas informações de atualização. Quando um nova informação de atualização chegar, o participante receptor atualiza seus estados e atributos (Singhal and Zyda, 1999).

Porém, o *dead reckoning* tem algumas limitações. Não garante que todos os participantes compartilharão o mesmo estado de cada objeto do ambiente (Correia, 2005). Assim, os participantes devem estar preparados para tolerar e adaptar possíveis discrepâncias no estado dinâmico dos objetos, para prevenir visualizações em atraso e perder o dinamismo inerente ao ambiente. Quando uma nova atualização chega ao participante, ele deve adaptar os novos estados e atributos do objeto sem que a mudança se torne brusca demais, ou seja, deve prover mecanismos de suavização da visualização e dinâmica dos objetos envolvidos no *dead reckoning* (Leite Júnior, 2000).

Além do uso de *dead reckoning*, cada um dos microcomputadores participantes do AVD pode armazenar uma cópia do ambiente virtual compartilhado (Capin et al., 1999). Dessa forma, o processamento de ocorrências do AVD pode ser descentralizado, sendo realizado localmente pelos participantes.

Esse tipo de processamento acontece de forma simultânea nos participantes, garantindo a sincronia entre todos os participantes e o ambiente virtual. Assim, um ou mais PDUs devem ser definidos para que ocorram disparos de eventos, e rotinas ou *scripts* são executados para controlar localmente ações, movimentos e mudanças de estados de diversos objetos (Leite Júnior, 2000). Além disso, esse tratamento descentralizado de processamento de atualizações evita o envio de PDUs que indiquem somente estados intermediários, substituindo esse grupo por um único PDU bem definido (Singhal and Zyda, 1999).

Assim como o *dead reckoning* e a descentralização do processsamento dos eventos, a compressão pode ser usada como uma forma de diminuir o tamanho das PDUs usadas no ambiente, por meio da retirada de possíveis elementos redundantes ou da utilização de padrões alternativos de representação de seus valores (Serin, 2003).

A filtragem torna-se uma característica fundamental em AVDs, já que nem todos os participantes necessitam receber determinadas alterações de estados em alguns objetos (Eraslan et al., 2007). Isso exige a adoção de derminados padrões de seleção de receptores, podendo ser utilizado a distância entre a representação virtual do participante e o objeto que recebeu a atualização de estado. Uma filtragem bem sucedida auxilia na diminuição da utilização da rede e também reduz a carga computacional utilizada para o processamento de AVDs, incluindo até mesmo o processo de geração gráfica e as simulações envolvidas (Correia, 2005). Isso acontece por causa do menor número de participantes envolvidos nas atualizações de estados de objetos.

Uma possível filtragem é a divisão do ambiente virtual distribuído em regiões ou células na qual cada célula represente a interação dentro de um espaço comum. A divisão está relacionada com a comunicação espacial entre os objetos que compartilham a mesma área de interesse (Correia, 2005). Cada divisão do ambiente pode ser processada em paralelo. Dessa forma a troca de PDUs somente acontece em um subconjunto de participantes remotos que se encontram nas mesmas partes do mundo virtual.

A partição geográfica do ambiente acontece com a divisão em áreas específicas, podendo ser modeladas como hexágonos regulares, retângulos regulares, ou até mesmo formas geométricas arbitárias (Leite Júnior, 2000). Porém a alternativa mais interessante é a utilização do formato de hexágono regular, pois apresenta as regiões bem mais interligadas do que as retangulares (Correia, 2005), e ainda, mais bem definidas do que as formas arbritárias. Uma região hexagonal apresenta até 6 vizinhos imediatos, enquanto regiões quadradas apresentam somente até 4 vizinhos, como pode ser visto na Figura 2.5. Quanto mais vizinhos próximos, menos associações e desassociações

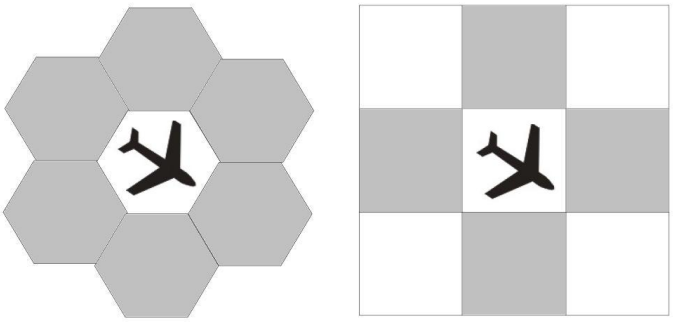

Figura 2.5: Partição geográfica de regiões: Hexagonal (Correia, 2005) e Retangular Regular

ocorrerão quando o participante trocar de região ao percorrer todo o ambiente virtual (Leite Júnior, 2000).

2.5 AVDNet

A AVDNet (Correia, 2005) é uma arquitetura de software consistente e flexível para criação e gerenciamento de aplicações de realidade virtual distribuída não específicas, capaz de suportar uma grande quantidade de participantes conectados ao ambiente, por intermédio da infra-estrutura disponível na Internet. Desenvolvida por (Correia, 2005), permite implementar a interação e a detecção de colisão de objetos em tempo real, mantendo a sincronia e consistência do estado dos objetos. Possibilita também o tratamento dos aspectos envolvidos na colaboração entre os participantes do mundo virtual. Ela identifica e sustenta os requisitos funcionais e não funcionais relacionados ao desenvolvimento e gerenciamento de AVDs, tais como: comunicação e interação entre os objetos e participantes do ambiente virtual em tempo real; tráfego gerado por sistemas desse tipo; impactos causados por atrasos adicionais que podem ser gerados pela distribuição dos dados (latência); modelagem dos objetos tridimensionais e seus comportamentos; sincronização da visualização do ambiente entre os participantes e resposta rápida a novos requisitos do sistema (Correia, 2005).

A escalabilidade em ambientes virtuais distribuídos é determinada pelo número de participantes simultâneos que eles podem suportar (Correia and Pellegrino, 2007). Para isso, métodos e técnicas que gerenciam recursos de largura de banda de rede e capacidade de processamento foram propostos e integrados na AVDNet, tais como: gerenciamento de áreas de interesse ou regiões, predição da posição dos objetos por meio do *Dead reckoning*, nível de detalhamento da imagem com base na distância do avatar, agregação e compressão de pacotes de atualização, balanceamento da carga por intermédio de

servidores adicionais, e em especial a proposta do protocolo de comunicação *multicast* em nível de aplicação, que permite a utilização das aplicações por qualquer *host* da Internet, independente dos roteadores suportarem ou não *multicast* nativo (Correia and Pellegrino, 2007).

A organização da arquitetura distribui os requisitos em vários componentes inter-relacionados, destacando-se três módulos de acordo com suas funcionalidades: Modelagem, Gerenciamento e Comunicação. A Figura 2.6 ilustra a estrutura da AVDNet e seus módulos. Para cada um dos módulos, destacam-se os componentes, que oferecem e utilizam serviços uns dos outros e especificam os padrões e procedimentos para a realização de suas funções (Correia, 2005).

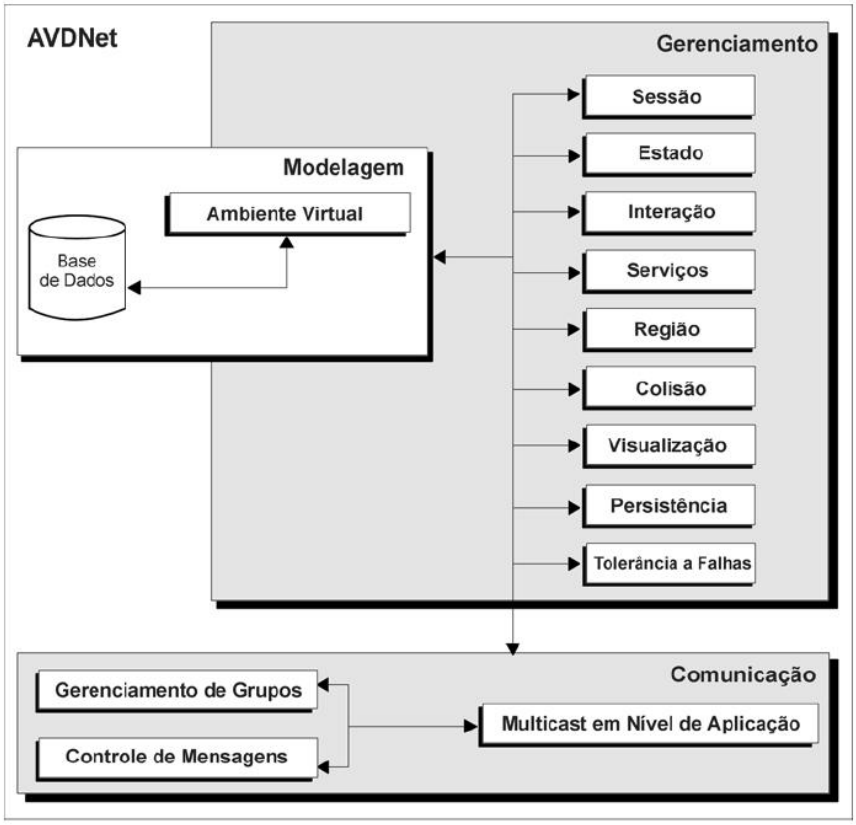

Figura 2.6: Estrutura da AVDNet (Correia, 2005)

Uma das principais características dos AVDs é garantir que todos os participantes ativos tenham visão consistente do ambiente, dessa forma, qualquer mudança de estado dos objetos, ocasionados por interações dos participantes ou por outras ações de outros objetos do ambiente, devem ser identificadas e distribuídas aos demais. Para tanto, o módulo de Gerenciamento, por meio do inter-relacionamento de seus componentes, é responsável em oferecer tais características (Correia, 2005).

O módulo de Comunicação permite que as estações participem efetivamente em sessões de comunicação em grupo, fornecendo um canal de co-

municação entre os grupos de participantes, de forma a otimizar e gerenciar as mensagens enviadas e recebidas entre os participantes (Correia, 2005).

A AVDNet, baseada na infra-estrutura atual da Internet, propõe o protocolo de comunicação *multicast* em nível de aplicação como uma técnica alternativa para o gerenciamento da comunicação em grupo em aplicações de realidade virtual distribuída.

O protocolo de comunicação *multicast* em nível de aplicação, também conhecido como *multicast* baseado na estação, se refere à capacidade de comunicação multicast implementada na camada de aplicação em vez da camada de rede, em outras palavras, uma rede virtual é construída com capacidade *multicast* sobre uma rede que transmite somente *unicast* (Correia and Pellegrino, 2007). Esse protocolo integra o conceito de conexão ponto-a-ponto entre entidades, com o conceito adicional de entidades que remetem um fluxo recebido para outros receptores, evitando que a entidade origem envie a todos os seus receptores (Correia, 2005).

CAPÍTULO 3

Internet Protocol version 6 - IPv6

3.1 Considerações Iniciais

Nesse Capítulo, uma breve descrição de características e funcionalidades são apresentadas sobre o protocolo designado como sucessor do IPv4, o IPv6.

3.2 Conceituação

O IP versão 6 (IPv6) é o protocolo de comunicacão designado como sucessor do IPv4 (RFC3697, 2004). Assim como o IPv4, ele também pertence à camada de rede da arquitetura OSI (*Open Systems Interconnection*). A Figura 3.1 apresenta a arquitetura OSI e os protocolos IP.

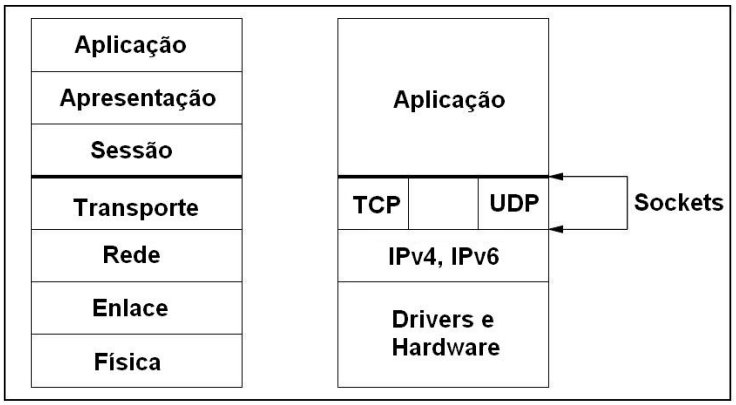

Figura 3.1: Modelo OSI e protocolos IP

Apesar do tempo de uso do IPv4 (surgiu no final da década de 1970), o principal motivo para a atualização do IP é a limitação no espaço de endereços

proporcionado pelo aumento potencial dos aparelhos conectados a Internet, principalmente aparelhos móveis: celulares, PDAs, câmeras, etc (Neto, 2005).

Os benefícios do IPv6 se sobressaem quando comparados ao IPv4 (Eraslan et al., 2007). Apesar de muitas semelhanças conceituais entre os dois potocolos, os projetistas aproveitaram a oportunidade para ajustar e ampliar aspectos do IPv4 com base na experiência operacional acumulada sobre esse protocolo (Comer, 2006).

3.3 Mudanças em relação ao IPv4

Os padrões IPv6 (RFC2460, 1998) e IPv4 foram projetados para atuar no mesmo contexto: no processo de comunicação entre computadores. Porém, o IPv6 é mais abundante em características e funcionalidades do que o IPv4 (Eraslan et al., 2007).

Uma das principais características é a capacidade de endereçamento expandida do IPv6, pois a representação aumenta o tamanho do endereço IP de 32 bits para 128 bits para suportar mais hierarquias de endereçamento e um número maior de endereços atribuídos a usuários (Silva and Faria, edes).

Para suportar novos paradigmas de comunicação entre computadores, entre eles a necessidade de protocolos para lidar com sistemas distribuídos mais eficientemente (RFC2460, 1998), novas características estão presentes no IPv6. O cabeçalho básico foi aprimorado, em comparação ao IPv4, para o tamanho padrão de 40 bytes.

O IPv6 usa um formato de cabeçalho novo e incompatível com o IPv4. Alguns campos foram incluídos para suportar novas funcionalidades, e outros, originados do cabeçalho IPv4, foram retirados no novo protocolo para reduzir o custo de processamento na passagem do datagrama por toda a estrutura da rede (de Ensino e Pesquisa RNP, html).

Além do cabeçalho básico de 40 bytes, também é possível a extensão e a inclusão de opções ao cabeçalho. Mudanças no modo como as opções do cabeçalho são codificadas permitem um melhor fluxo de datagramas na rede e uma grande flexibilidade para introduzir novas características e funcionalidades ao protocolo no futuro. Opções avançadas permitem que um datagrama inclua informações de controle opcionais (Neto, 2005).

A autoconfiguração do IPv6 permite que computadores em uma rede isolada atribuam endereços locais a si mesmos automaticamente por intermédio de seus próprios endereços MAC (*Media Access Control*) (RFC2462, 1998).

Para obter um tratamento específico para diferentes seviços estabelecidos, conhecido como QoS - *Quality of Service*, no IPv6 é possível habilitar controles de fluxos e especificar serviços diferenciados.

Dentre as funcionalidades que contribuem para o sucesso do IPv6 em sistemas distribuídos, uma em especial, é a capacidade de estabelecer comunicações por meio do *multicas* nativo. Endereços *multicast* facilitam para que as interfaces de rede recebam dados conjuntamente por intermédio de um único endereço IP.

Além do *multicast*, comunicações *unicast* são estabelecidas e direcionadas para comunicação direta entre interfaces de rede. O IPv6 acrescentou um novo de tipo de conexão denominada *anycast* (RFC1546, 2003), que permite que um datagrama seja entregado a qualquer integrante de um grupo definido por meio de um único endereço IP (Comer, 2006).

Para manter a confiabilidade dos dados (opcional) e para que os datagramas sejam autenticados, extensões são especificadas no IPv6 (RFC4303, 2005) com a utilização opcional de algoritmos de criptografia DES/3DES (*Data Encryption Standard*) para cifrar as mensagens durante todo o trajeto dos pacotes (RFC2460, 1998).

3.4 Transmissão de pacotes IPv6 na rede Ethernet

3.4.1 Formato do Frame

Os datagramas IPv6 são encapsulados dentro de um frame *Ethernet*. O frame Ethernet possui um cabeçalho *Ethernet* contendo o endereço do remetente e do destinatário e um campo de área de dados onde todo o pacote IPv6 é encapsulado. Assim, para uma rede subjacente, todos os pacotes de dados são reconhecidos pelo mesmo padrão . A rede física trata o pacote inteiro, incluindo o cabeçalho como dados (Comer, 2006).

3.4.2 Unidade Máxima de Transmissão

A MTU *(Maximum Transmission Unit)* refere-se ao máximo datagrama que pode se enviado pela rede. Se o tamanho do datagrama for maior que a MTU da rede, o datagrama tem que ser fragmentado em mútiplos datagramas menores.

O tamanho para pacotes IPv6 na rede Ethernet é limitado pelo tamanho da MTU, que é de 1500 octetos (RFC2464, 1998). O datagrama nem sempre trafega usando o tamanho máximo em relação a MTU, o datagrama pode ser fragmentado por um aviso de algum roteador intermediário indicando um valor de MTU menor e obrigando o emissor a diminuir o tamanho do datagrama, ou por configuração manual em cada interface de rede.

Quando um host tem uma grande quantidade de dados a ser enviada por meio de uma rede, os dados serão transmitidos em séries de múltiplos datagramas. No IPv4, para evitar a fragmentação excessiva e aumentar a eficiência

da rede, os datagramas devem ser fragmentados no maior tamanho possível para que possam trafegar sem serem fragmentados novamente ao longo do caminho da rede entre o remetente e o destinatário.

No entanto, no IPv6, para evitar que os datagramas sejam fragmentados, uma técnica foi especificada (RFC1191, 1998) para que um caminho seja escolhido dinamicamente de modo que seja um bom caminho a ser trafegado pelos datagramas, técnica conhecida como *Path MTU Discovery - PMTU*. Porém, nem sempre este melhor caminho escolhido é o melhor a ser percorrido. Se o caminho entre o remetente e o destinatário não estiver situado na mesma rede, provavelmente nesse caminho existe uma subrede que tenha menor MTU do que as outras, e assim o tamanho do datagrama é limitado pela menor MTU das redes presentes no caminho do datagrama. Isso significa um enorme desperdício de recursos nas subredes com MTU maiores (Kurose and Ross, 2006).

3.5 Funcionalidades IPv6

A extrema facilidade e eficiência com que o IPv6 lida com sistemas distribuídos multi-usuários torna-se uma melhor alternativa para o uso em AVDs (Eraslan et al., 2007). Seu uso pode aumentar substancialmente a taxa de transferência e suprir as necessidades de criação e gerenciamento de aplicações distribuídas que necessitem suportar uma grande quantidade de participantes conectados ao ambiente. A seguir são apresentadas algumas funcionalidades presentes no IPv6 para aumentar a eficiência no processo de comunicação.

3.5.1 Multicast

Multicast é um ótimo e poderoso método para implementar aplicações distribuídas. Provê um mecanismo para distribuir mensagens a um grande número de usuários com um esforço mínimo e baixa utilização dos recursos de rede.

É um serviço que permite a rede suportar comunicações ponto-multiponto, ou seja, um pacote pode ser endereçado para um grupo de participantes que possuem endereço único, assim o roteador é o responsável em replicar os pacotes e encaminhá-los para um ou mais participantes presentes na subrede (Filipe and Martins, 2007).

Portanto, para que conexões *multicast* sejam estabelecidas, é necessário que os roteadores dêem suporte a este tipo de conexão (Correia, 2005). Esta tecnologia já está disponível em grande parte dos equipamentos de rede mais recentes, e a maioria dos novos roteadores já vem com suporte a IPv6 por intermédio do *dual-stack*.

No entanto, o problema situa-se nos roteadores já instalados e ativos, já

que na rede mundial de computadores é inviável prever um determinado caminho na qual existam somente roteadores habilitados com *dual-stack* e *multicast* (Eraslan et al., 2007). Nesses casos é necessário o uso de túneis para que pacotes IPv6 possam trafegar sobre uma rede IPv4 (Neto, 2005) .

Determinadas topologias de AVDs em conjunto com o *multicast* podem aumentar significamente a eficiência do ambiente virtual distribuído, já que a topologia define como a informação trafega entre os participantes da AVD (Leite Júnior, 2000).

3.5.2 Quality of Service

A especificação IPv6 (RFC2460, 1998) determina uma estrutura mais aprimorada e mais simples que no IPv4, todo cabeçalho contém um campo *Traffic Class* de 8 bits e um *Flow Label* de 20 bits. O suporte a QoS no IPv6 implementa um mecanismo confiável de colaboração entre os objetos em um AVD.

A *Traffic Class* distingue as diferentes classes e prioridades nos pacotes IPv6. O que permite obter controle sobre os tipos de tráfegos especiais para determinadas aplicações, obtendo escalabilidade e processamento especial a cada nó no caminho do tráfego (Eraslan et al., 2007). Todo o controle de QoS e repasse de pacotes pode estar situado no roteador, que pode ser o responsável em rotear as determinadas classes de tráfego aos seus respectivos destinos, e que portanto é necessário um roteador que suporte tal redirecionamento baseado em QoS (de Ensino e Pesquisa RNP, html).

O *Flow Label* é usado para rotular pacotes que pertecem a fluxo particulares para os quais o remetente requisita tratamento especial. Esse campo é selecionado pela origem e nunca é alterado na rede (Eraslan et al., 2007). Exemplo: mensagens de interação no ambiente entre os participantes são muito mais importantes do que um pacote ICMP (*Internet Control Message Protocol*) que foi enviado ao ambiente.

3.5.3 Anycast

Um mesmo endereço *anycast* pode ser atribuído a múltiplas interfaces de rede simultaneamente. Isso permite que um pacote destinado a um endereço *anycast* seja enviado para a melhor interface de rede de acordo com o protocolo de roteamento (RFC1546, 2003).

O *anycast* é geralmente usado como uma maneira de fornecer a disponibilidade elevada a determinado serviço ou servidor específico, além de ser usado também como uma forma de balanceamento de carga para serviços (Eraslan et al., 2007), como exemplo o acesso a dados replicados em AVDs, onde a dinâmica inerente a estes ambientes faz com que os participantes nunca es-

tejam numa mesma rota de tráfego (Serin, 2003).

Porém, a tecnologia requer uma infra-estrutura de controle mais complexa do que as requeridas em conexões *unicast*. Determinadas topologias em AVDs podem se beneficiar do uso do *anycast*, como topologias *Cliente-Servidor* de múltiplos servidores permitem que o participante possa encontrar o melhor servidor para conexão inicial ao ambiente (Eraslan et al., 2007).

3.5.4 Fragmentação e Remontagem

O IPv6 não permite a fragmentação e remontagem de datagramas IP em roteadores intermediários, ao contrário do IPv4. Essas operações podem ser realizadas somente pela fonte e pelo destino do tráfego. Assim, se um datagrama for maior que a MTU da rede, o roteador simplesmente o descarta e envia uma mensagem ICMP para a origem, que então o fragmenta em múltiplos datagramas menores (Kurose and Ross, 2006). A montagem e a fragmentação de datagramas IP são processos que consomem muito tempo, a execução de tais tarefas apenas nos sistemas finais acelera consideravelmente o repasse de pacotes e, conseqüentemente melhora a eficiência em AVDs (Neto, 2005).

3.5.5 Autoconfiguração de endereço

O IPv6 define dois modos de autoconfiguração de endereço: *stateful address autoconfiguration* e *stateless address autoconfiguration* (RFC2462, 1998). O mecanismo *stateless* permite que o *host* possa gerar seu próprio endereço IP usando uma combinação entre o endereço MAC (*Media Access Control*) e informações enviadas pelos roteadores. Enquanto os roteadores determinam o prefixo que identifica a subrede associada a conexão, a interface de rede determina um identificador único que identifica o *host* na subrede (RFC2462, 1998). Um endereço IP autoconfigurado é formado pela combinação de ambas as informações.

O método de derivação de endereço por meio do endereço MAC tem a função de preservar a unicidade do host na sua respectiva subrede possível. Porém, não há proteção quanto à duplicação de endereços, seja por acidente ou por fraude (RFC2464, 1998).

Já no modelo *stateful address autoconfiguration*, os *hosts* obtêm os parâmetros de configuração do endereço IP por meio de um servidor central que mantém domínio sobre os endereços alocados a cada host. Também conhecido como DHCPv6 (*Dynamic Host Configuration Protocol for IPv6*)(RFC4649, 2006).

3.6 Formato de um cabeçalho básico do IPv6

O cabeçalho IPv6 é mais simplificado do que o IPv4, embora tenha que acomodar maiores representações de endereços IP, os campos foram escolhidos de forma que o tráfego de datagramas seja explorado da melhor forma pelos roteadores e por toda a estrutura da rede, já que no IPv4 alguns campos representam funcionalidades que não são necessárias.

Assim, os cabeçalhos foram escolhidos de forma que aumente significativamente a taxa de transferência. Em redes de amplo tráfego de informações, como um ambiente virtual distribuído (Correia, 2005), isso pode ser decisivo para o sucesso do ambiente, já que com um cabeçalho mais eficiente a velocidade do tráfego pode aumentar e todas os dados podem chegar ao destino mais rapidamente.

O cabeçalho básico, que no IPv4 é de tamanho variável, no IPv6 é de tamanho fixo de 40 bytes. A Figura 3.2 apresenta o formato do geral do IPv6.

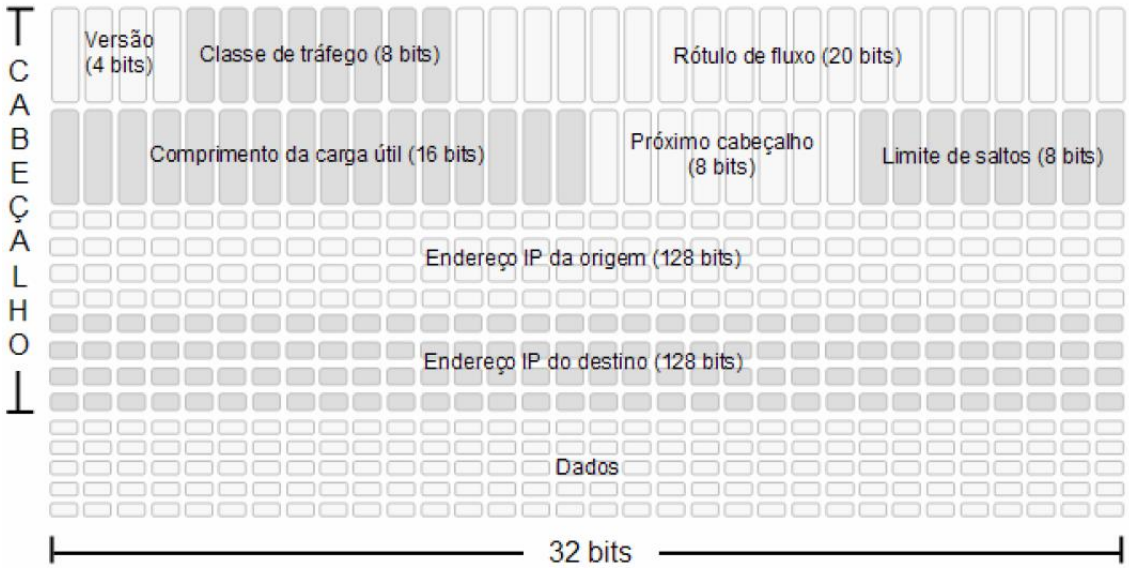

Figura 3.2: Formato geral do datagrama IPv6 (Neto, 2005)

Os seguintes campos estão presentes em um cabeçalho IPv6 básico (Kurose and Ross, 2006):

Versão: refere-se ao tipo de versão do protocolo IP. Versão 4 para IPv4 e Versão 6 para IPv6.

Classe de Tráfego: tem função semelhante ao campo TOS *Type of Service* do IPv4. Esse campo classifica o pacote em uma classe de serviço ou prioridade, que pode ser usado para diferenciar serviços e requerer tratamento especial ao pacote.

Rótulo de fluxo: não existe no IPv4. Criado para marcar pacotes de um fluxo específico para que sejam diferenciados na camada de rede da arquite-

tura OSI. Com esse rótulo o roteador pode identificar a qual tipo de fluxo o datagrama pertence sem que seja preciso verificar sua aplicação, o que facilita a pratica do *QoS*.

Comprimento de carga útil: informa o número de bytes do datagrama IPv6. Não é preciso informar o tamanho do cabeçalho, já que nessa nova versão os cabeçalhos possuem tamanho fixo de 40 bytes.

Próximo cabeçalho: equivale ao *Campo de Protocolo* no cabeçalho IPv4. Identifica o tipo de informação que segue o cabeçalho básico IPv6. Essa informação pode ser o protocolo usado na camada de transporte, o UDP (*User Datagram Protocol*) ou TCP (*Transmission Control Protocol*), ou também pode se referir a um cabeçalho de extensão.

Limite de saltos: equivale ao *Tempo de Vida* do protocolo IPv4. A cada roteador*(hop)* que o datagrama trafega esse campo é decrementado em uma unidade. Se esse campo possuir valor zero, o datagrama é descartado.

Endereço de fonte e destino: carregam, respectivamente, os endereços de 128 bits do remetente e do destinatário.

Dados: contém os dados das camadas superiores a camada de rede da arquitetura OSI e da mensagem propriamente dita.

3.7 Cabeçalhos de extensão

A escolha do uso de cabeçalhos de extensão gera um conflito entre generalidade e eficiência. Enquanto o cabeçalho básico IPv6 de 40 bytes fixos dispõe de campos que são usados na maioria das situações de conexão entre computadores, os cabeçalhos de extensão propõem um uso mais eficaz e dirigido, já que nem todas as situações usam os campos descritos no cabeçalho básico IPv6. Cada datagrama inclui cabeçalhos de extensão somente para as funcionalidades que o datagrama utiliza (Kurose and Ross, 2006).

Um datagrama IPv6 pode carregar zero, um, ou mais pacotes de extensão (RFC2402, 1998). Não há um tamanho fixo, e podem variar de acordo com o tipo de cabeçalho de extensão ou o mesmo tipo podem ter tamanhos diferentes. Para isso, os cabeçalhos de extensão possuem o campo *Extension Header Length* que indica seu tamanho.

Outro campo do cabeçalho de extensão é o *Next Header*. Nele é identificado qual o tipo de cabeçalho que o sucede por intermédio do seu valor. No último cabeçalho de extensão, o campo *Next Header* indica o protocolo da camada de transporte que é usado nesse pacote. A Figura 3.3 apresenta a disposição dos cabeçalhos de extensão no IPv6.

Cabeçalhos de extensão não são analisados ou processados durante o repasse

de pacotes no caminho de tráfego até que cheguem ao destino especificado no campo *Destination Address*. Eles devem ser analisados estritamente na ordem em que chegam no pacote, ou seja, o destinatário não pode receber o pacote e escaneá-lo a procura de um tipo particular de cabeçalho e processá-lo com prioridade sobre os outros cabeçalhos de extensão (RFC2460, 1998), com uma exceção, o cabeçalho de extensão *Hop-by-Hop Options Header* deve ser processada por todos os roteadores no caminho do pacote, inclusive pelo remetente e pelo destinatário. Porém, os destinatários estão preparados para receber os cabeçalhos de extensão em qualquer ordem de chegada.

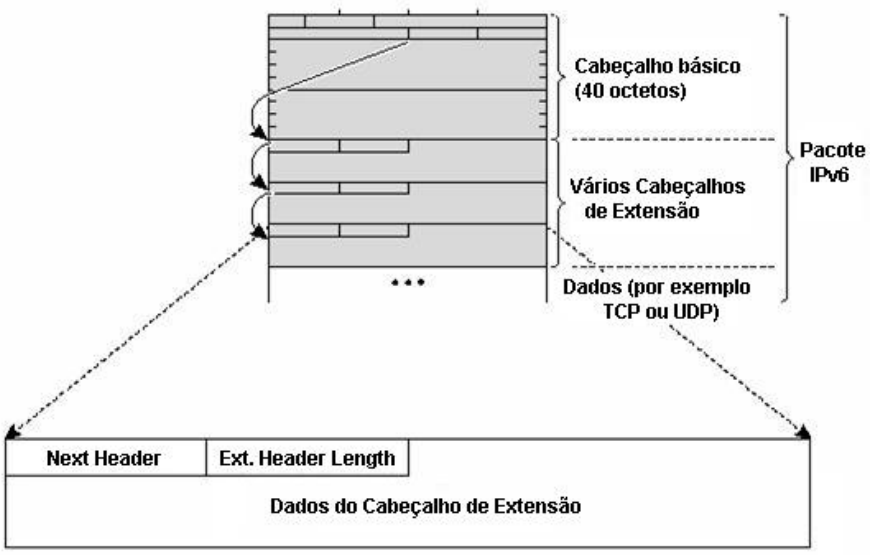

Figura 3.3: Formato dos cabeçalhos de Extensão IPv6 (Neto, 2005)

A implementação do IPv6 inclui os seguintes cabeçalhos de extensão (RFC2460, 1998):

Hop-By-Hop Options Header: Esse cabeçalho requer tratamento especial e deve aparecer sempre após o cabeçalho básico IPv6. É usado para transportar informação opcional ou adicional que deva ser analisada por todos os dispositivos intermediários no caminho do tráfego de dados. Ele é identificado pelo valor 0 no campo *Next Header* do cabeçalho básico IPv6.

Destination Options Header: É usado para transportar informações opcionais que devam ser analisadas somente pelo destinatário final, e é identificado pelo valor 60 no campo *Next Header* no imediato cabeçalho precedente.

Ambos os cabeçalhos *Destination Options Header* e *Hop-by-Hop Options Header* foram projetados no mesmo formato e reúnem informações simples que não necessitam de mais um cabeçalho de extensão. Eles possuem um campo *Type* que indica o tipo de dado que segue no cabeçalho, os seus 2 bits de mais alta ordem indicam qual ação deve ser tomada caso o processamento

do datagrama IPv6 não reconheça o tipo de dado do cabeçalho de extensão enviado.

Routing Header: Esse cabeçalho é usado para mostrar um ou mais caminhos intermediários para serem visitados pelo pacote até chegar ao destino final. Desta forma o remetente IPv6 pode traçar um caminho alternativo de roteamento mesmo que este caminho não seja o caminho indicado pelos protocolos de roteamento. É identificado pelo valor de 43 no campo *Next Header* em seu cabeçalho precedente.

Como alternativa, o *Routing Header* pode ser seguido pelo *Hop-by-Hop Options Header*, que significa que todos os nós intermediários devem analisar o cabeçalho em busca de informações adicionais. Caso o caminho referido no cabeçalho de roteamento possua uma MTU - *Maximum Transmition Unit* menor que o tamanho do pacote, o nó deve descartar o pacote e enviar um *ICMP Packet Too Big* para o remetente IPv6 indicando que o pacote é muito grande para trafegar na rede por este caminho.

Fragment Header: É usado para enviar um pacote que seja maior que a MTU do caminho até o destino. Esse cabeçalho é identificado pelo valor 44 no campo *Next Header* de seu cabeçalho precedente.

Para cada pacote fragmentado, o remetente gera um *Identification Value*, que corresponde a um campo específico do *Fragmentation Header*. Essa identificação deve ser única e diferente de qualquer outro pacote enviado recentemente com o mesmo remetente e destinatário (medido com um contador de tempo de vida (RFC2460, 1998)). Cada fragmento possui uma cabeçalho base seguido de pelo menos um de Extensão do tipo *Fragment Header*, cujo processamento só ocorre no destino, onde eles são remontados e transformados no pacote que os originaram.

O remetente que é o responsável pela fragmentação dos pacotes, tem duas escolhas: ele pode usar a MTU mínima garantida de 1500 octetos (Comer, 2006) ou realizar um *Path MTU Discovery*, procedimento de descoberta do tamanho máximo do pacote que irá trafegar entre a origem e o destino. Assim, basta que o fragmento seja menor que a MTU esperada do caminho.

Authentication Header: Esse cabeçalho é usado dentro do serviço IPSec (*IP Security Protocol*) para prover autenticação e garantia de integridade aos datagramas IPv6 (RFC2402, 1998). Esse cabeçalho é identificado pelo valor 51 no campo *Next Header* de seu cabeçalho precedente.

O IPSec não restringe o usuário a um único algoritmo de criptografia ou autenticação, o IETF (*Internet Engineering Task Force*), criador do IPSec, escolheu tornar o sistema flexível e extensível, permitindo assim que as extremidades em comunicação escolham os parâmetros e os algoritmos de segurança a serem usados na comunicação (Comer, 2006).

Encapsulating Security Payload - ESP: Esse cabeçalho também é usado dentro do serviço IPSec. Porém, lida com a privacidade e também autenticação entre as extremidades em comunicação, sendo mais complexo do que um *Authentication Header*. Um valor de 50 no campo *Next Header* de seu cabeçalho precedente informa a um destinatário que o datagrama transporta o ESP (RFC4303, 2005).

3.8 Endereçamento IPv6

A notação IPv6 utiliza 128 bits ao invés dos 32 bits do protocolo IPv4 (Bieringer, 2006). Essa nova representação exige uma notação diferente da representação decimal separada por pontos usada no IPv4. Logo, temos:

2^{128}-1: 340282366920938463463374607431768211455 possibilidades de representação de endereços IPv6

Tal número de possibilidades em representação decimal é impossível de ser memorizado. Assim, ao invés de representar um endereço em notação decimal, uma notação mais prática é a utilização da notação em hexadecimal. Esse formato reduz o tamanho do endereço IPv6 para 32 caracteres. Além do mais, para se tornar mais conveniente (Neto, 2005), os projetistas do IPv6 escolheram separar cada 16 bits do endereço IPv6 com o uso de ':' (dois pontos). Exemplo de um endereço IPv6:

2001:0db8:0100:0000:0210:a4ff:fee3:0168

Por simplificação, zeros a esquerda em cada bloco de 16 bits podem ser omitidos, assim como blocos formados por zero podem ser substituídos por um único zero:

2001:db8:0:0:210:a4ff:fee3:168

Grupos de zeros seguidos podem ser substituídos por '::'. Essa notação só deve ser utilizada uma única vez em cada endereço, senão é impossível de se descobrir a quantidade de bits abreviada. Assim, o exemplo resulta:

2001:db8::210:a4ff:fee3:168

Assim como a máscara de subrede e a notação CIDR (*Classless Inter-Domain Routing*) utilizadas no IPv4, o IPv6 também apresenta um conceito de prefixo para indicar a parte referente à identificação da rede. Os 64 bits mais altos representam o endereço da rede, e os 64 bits mais baixos representam o endereço do host (Bieringer, 2006). Notação CIDR: 2001:db8::210:a4ff:fee3:168/16.

3.8.1 Endereços e Prefixos Especiais

No IPv4 alguns endereços recebem denominações especiais baseados no seu modo de transmissão de dados, como *unicast* e *broadcast*. O *unicast* é us-

ado quando o destino representa um único computador, ou seja, os dados são enviados a uma única interface de rede. O *broadcast* é um endereço especial usado para enviar dados a todas as interfaces presentes na rede ou subrede.

Porém, o IPv6 aboliu o uso de endereços *broadcast* e em seu lugar entram os endereços *multicast*. O IPv6 também especificou um novo tipo de endereço, o endereço *anycast* (RFC1546, 2003).

O endereçamento *anycast* pode ser útil para detecção de um determinado serviço ou servidor. Já que de forma simplificada é possível encontrar o servidor mais próximo que ofereça um recurso específico, reduzindo a carga excessiva de tráfego de dados e permitindo que os usuários se conectem mais rapidamente ao servidor mais próximo (Eraslan et al., 2007). A Figura 3.4 apresenta o espaço de endereçamento especificado pelo IPv6 para alguns tipos de roteamento.

Alocação	Prefixo binário	Prefixo hexadecimal
Reservado	0000 0000	::0/128
Reservado para alocação NSAP	0000 001	
Reservado para alocação IPX (removido no RFC mais recente)	0000 010	
Endereço unicast Global	001	
Endereço unicast Link-local	1111 1110 10	FE80::/10
Endereço unicast Site-local	1111 1110 11	FEC0::/10
Endereço multicast	1111 1111	FF00::/8

Figura 3.4: Espaço de endereçamento IPv6 (Neto, 2005)

No IPv6 os endereços *unicast* possuem tipos definidos (RFC2460, 1998):

Global Unicast Address: pode ser roteado em toda e estrutura da rede e também na Internet sem escopo local.

Unspecified Address: definido como '::' indica a ausência de um endereço e nunca deverá ser representado por algum host. Um exemplo seria a utilização nos endereços de origem em estações não inicializadas que ainda não tenham aprendido seus endereços. Esse endereço não pode ser usado como destino em pacotes IPv6.

Loopback Address: definido como '::1', simula uma interface virtual e é usado somente para enviar mensagens para si mesmo. Um pacote com endereço de destino *Loopback* nunca deve ser repassado por um roteador.

Embedded IPv4 Addresses: trata-se de um endereço IPv6 com IPv4 embutido, denominado *IPv4-compatible IPv6 Address*. É formado anexando-se um prefixo nulo (96 bits zeros) a um endereço IPv4, como por exemplo '::172.15.20.32'. Foi incluído como forma para que os hosts façam o tunelamento de pacotes

IPv6 sobre roteamento IPv4 (Silva and Faria, edes). Para hosts que não possuam suporte IPv6 foi definido o *IPv4-mapped IPv6 Address* que é representado da seguinte forma '::FFFF:172.16.25.32'.

Local-Use IPv6 Address: existem dois endereços para o uso local: *Link-local* e *Site-local*. O *Link-local* é definido para uso interno em um único enlace. Podem ser usados em estações não configuradas para autoconfiguração de endereços, descoberta do vizinho (RFC2461, 2003) ou quando não há roteador. O *Site-local* pode ser usado em organizações que não se conectam a Internet. Os roteadores não devem repassar pacotes que tenham o campo *Remetente* ou *Destinatário* como endereços *Site-local*.

Os endereços *anycast* são alocados no mesmo espaço de endereçamento do *unicast*. Os dois endereços não possuem diferenças sintáticas. Quando um endereço *unicast* é alocado em várias estações simultaneamente, ele se torna um endereço *anycast* (o host deve ser explicitamente configurado para reconhecer o endereço).

Os endereços *multicast* identificam um grupo de interfaces ou hosts. Porém, uma interface pode pertencer a mais de um grupo *multicast* (Silva and Faria, edes). Os endereços IPv6 *multicast* são sempre iniciados por 1111 1111, ou seja, iniciam como 'ffxx', xx identifica o escopo do endereço, ou seja, o alcance do tráfego.

Escopo é um parâmetro que especifica a distância máxima que um pacote *multicast* pode trafegar a partir de uma entidade emissora (interface de rede). Os seguintes escopos *multicast* são definidos:

FFx1: *node-local*, pacotes nunca deixam o nó origem.

FFx2: *link-local*, pacotes nunca são roteados, assim nunca deixam o link especificado.

FFx5: *site-local*, pacotes nunca saem do nó de rede, sendo roteado apenas na rede interna, sem propagação na Internet.

FFx8: *organization-local*, pacotes nunca saem de uma organização definida(difícil implementação (Bieringer, 2006)).

FFxE: *global*, escopo mundial, toda a Internet.

Tipos *multicast* especiais:

All Nodes Address: endereça todos os hosts no nó local (FF01::1) ou o link (FF02::1).

All Routers Address: endereça todos os roteadores no nó local (FF01::2), no link (FF02::2) ou na rede interna (FF05::2).

CAPÍTULO 4

O Framework AVDNet6

4.1 Considerações Iniciais

Nesse Capítulo é apresentado o *framework* AVDNet6, proposto como objetivo da Monografia. Assim como, suas características, vantagens, e arquitetura do *framework*.

Posteriormente, são apresentados os resultados obtidos com testes aplicados ao *framework* AVDNet6, demonstrando sua eficiência e aplicabilidade no uso em Sistemas Distribuídos.

4.2 Introdução

A popularização do uso dos computadores nos mais diversos segmentos da sociedade e a expansão da Internet são alguns dos fatores que têm impulsionado a pesquisa e o desenvolvimento de tecnologias computacionais cada vez mais elaboradas e parecidas ao cotidiano das pessoas. Observa-se, por exemplo, que o uso de recursos multimídia deixou, há algum tempo, de ser um privilégio ou recurso adicional de um sistema de computação e passou a ser um componente indispensável a um grande número de aplicações e casos (Júnior et al., 1999). Por outro lado, as facilidades de acesso a redes de comunicação são responsáveis em expandir o uso do computador na sociedade (Comer, 2006).

Neste contexto, tecnologias são criadas, transformadas e melhoradas constantemente, o que torna relevante a apresentação e a elaboração de um *framework* que utilize tecnologias que possam aumentar a eficiência do processo de comunicação em sistemas distribuídos.

A especificação do AVDNet6 utiliza o conceito de *framework*, que é uma ar-

quitetura reutilizável que fornece estrutura e comportamento genéricos para uma classe específica de software, podendo ser adaptado ou estendido para atender variadas necessidades (Júnior et al., 1999).

O desenvolvimento com *frameworks* traz benefícios. *Frameworks* diminuem o esforço para entender e manter a aplicação. *Frameworks* incentivam o reuso, pois reúnem o conhecimento de desenvolvedores em determinado domínio. Além disso, *frameworks* são expansíveis por construção e, por fim, o uso de inversão de controle simplifica o desenvolvimento de aplicações (Correia, 2005).

Apesar dos benefícios decorrentes do uso de *frameworks*, também são impostos desafios ao desenvolvimento de *frameworks* e de aplicações que usem os *frameworks* (Júnior et al., 1999). Dentre eles, o esforço de desenvolvimento imposto aos projetistas de *frameworks*; a curva de aprendizagem, a integração e a eficiência, que afetam principalmente os desenvolvedores de aplicação; a manutenção, validação e remoção de defeitos, que afetam tanto desenvolvedores de aplicação quanto mantenedores de *frameworks*; e o desafio da falta de padrões que afetam a todos envolvidos no desenvolvimento e uso de *frameworks*.

Os *frameworks* requerem um enorme esforço de desenvolvimento e representam um grande investimento (Correia, 2005). Não são aplicações completas: falta a necessidade específica de uma aplicação. Eles implementam funcionalidades comuns a diversas aplicações, e declaram pontos de extensão que devem ser preenchidos pela aplicação que faz a instância do *framework*. O que também permite que uma aplicação possa ser construída a partir de um ou mais *frameworks*, inserindo-se as funcionalidades ausentes em suas lacunas.

4.2.1 Tecnologias Utilizadas

Na fase de implementação do *framework* AVDNet6 utilizou-se a linguagem de programação Java. A linguagem Java é considerada uma tecnologia para a construção otimizada de aplicações distribuídas, permitindo que múltiplos computadores sejam acessados por meio de uma rede e por múltiplos usuários remotamente localizados (Sun, 2007).

A linguagem Java é multiplataforma, ou seja, um programa escrito na linguagem Java pode ser executado em qualquer Sistema Operacional sem necessidade de alterações no código-fonte. Tal funcionalidade é possível devido à estrutura de linguagem interpretada que caracteriza a linguagem Java. Outra característica importante é que a linguagem Java permite a inclusão de bibliotecas (Repositório de Classes) de propósito específico por meio da definição de interfaces, denominadas *packages* (como exemplo, bibliotecas para Pro-

gramação Distribuída, Banco de Dados, Computação Grafica, IPv6, e outras) (Sun, 2007).

O protocolo de comunicação com que o AVDNet6 se comunica é o IPv6 (descrito no Capítulo 3 na página 19). Para que o IPv6 possa ser utilizado plenamente em uma rede utilizando o IPv4, devem ser atualizados, pelo menos, todos os hosts dessa rede (de Ensino e Pesquisa RNP, html), ou seja, uma mudança árdua deve ser feita em toda e estrutura da rede. Entretanto, existem mecanismos que suportam, gradualmente, a migração do protocolo IPv4 para o IPv6.

Um dos mecanismos que suportam essa migração gradual é o *dual-stack*. Nesse mecanismo é possível que *hosts* e roteadores manipulem ambos os protocolos em uma mesma interface de rede, ou seja, tanto IPv4, como IPv6. Dessa maneira, um host *dual-stack* pode receber e transmitir pacotes dos dois protocolos, fazendo com que eles coexistam em uma mesma rede (Sun, 2007).

O desenvolvimento do *framework* AVDNet6 baseou-se no Sistema Operacional GNU/Linux. O GNU/Linux possui uma modularidade inerente que permite uma ampla flexibilidade no desenvolvimento de sistemas distribuídos por meio da manipulação de opções do *kernel* (núcleo) do GNU/Linux (Neto, 2005). Além, o *kernel 2.6* do GNU/Linux é capaz de utilizar o *dual-stack* naturalmente (Organization, lorg).

A vantagem em utilizar o *dual-stack* é a habilidade em criar apenas um *socket* para interpretar ambos os protocolos IPv4/IPv6, simplificando o desenvolvimento da aplicação. Em Sistemas Operacionais que utilizam apenas uma pilha para cada protocolo, como o Windows XP (Microsoft, tcom), é necessário a criaçã de um *socket* para que cada protocolo seja interpretado em sua respectiva pilha. A Figura 4.1 ilustra como o *dual-stack* é representado na arquitetura de rede.

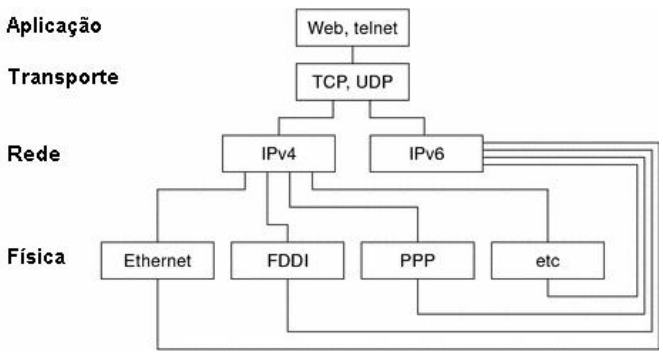

Figura 4.1: Dual-Stack (Neto, 2005)

4.3 AVDNet6 como camada de extensão da AVDNet

A AVDNet (Apresentada na Seção 2.5 na página 16) é uma arquitetura para criação e gerenciamento de aplicações distribuídas por intermédio da infra-estrutura disponível na Internet, desenvolvida por (Correia, 2005).

A AVDNet contribui para a idealização de um sistema distribuído de realidade virtual como um ambiente integrado de geração de múltiplos mundos virtuais, que suporta a construção e simulação de diferentes aplicações, a construção e integração de objetos estáticos e dinâmicos com vários níveis de complexidade, que colaboram e interagem em tempo real entre eles, além de permitir o acesso de múltiplos usuários dispersos fisicamente. Isto tudo combinado à minimização dos recursos de rede na tentativa de contornar os problemas de utilização de largura de banda, distribuição, latência e confiabilidade, mantendo a sincronia e consistência do ambiente virtual (Correia, 2005).

Porém, a estrutura atual da Internet, baseada no protocolo IPv4, não garante níveis de perfomance satisfatórios, pois existem funcionalidades que permitem aumentar a taxa de transferência e obter melhor eficiência, porém não são implementadas no IPv4: suporte a QoS, *multicast* de forma nativa e segurança em termos de autenticação e privacidade. Estes mecanismos são usados principalmente para distribuir mensagens de atualização em larga-escala aos participantes do AVD de forma eficiente. Quando um participante se move de um ponto a outro de um ambiente virtual é necessário o uso de um protocolo de comunicação que possa prover esses mecanismos de forma nativa, além de garantir escalabilidade ao ambiente virtual distribuído.

O IPv6 é uma solução para garantir eficiência. O protocolo IPv6 já foi implementado para um grande número de plataformas, e a maioria dos roteadores vêem sendo implementados com *dual-stack* para lidar com pacotes IPv4 e IPv6. Assim, os serviços oferecidos acima da camada de rede da arquitetura ISO/OSI continuam os mesmos (Kurose and Ross, 2006).

O desenvolvimento do *framework* AVDNet6 teve motivação na necessidade de prover a sistemas distribuídos existentes uma camada de gerenciamento mais eficaz e otimizar o uso dos limitados recursos de rede em AVDs de larga escala, aumentando a taxa de transferência do ambiente virtual respectivo.

O AVDNet6 é utilizado como uma camada de extensão da camada de Comunicação da AVDNet. Além da AVDNet6 contribuir para que a AVDNet suporte IPv6, o *framework* pode ser facilmente estendido para uso em AVDs não específicos. A Figura 4.2 representa a extensão que o *framework* AVDNet6 proporciona a AVDNet.

Assim, o AVDNet6 proporciona à AVDNet um módulo mais eficaz de geren-

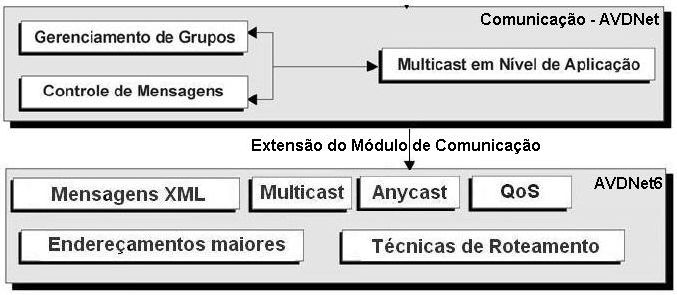

Figura 4.2: Extensão do Módulo de Comunicação da AVDNet

ciamento da Comunicação estabelecida no Ambiente Virtual Distribuído. Por intermédio das funcionalidades implementadas no IPv6, o *framework* disponibiliza Endereçamentos Maiores, Técnicas de Roteamento mais eficientes, comunicações *Multicast* e *Anycast* nativas, QoS, e padronização das mensagens de comunicação com XML.

4.4 A Arquitetura

A organização da estrutura do *framework* AVDNet 6 é baseada em módulos separados de acordo com suas funcionalidades: *Cliente* e *Servidor*. Para cada módulo é especificado componentes que interagem entre si e declaram padrões e procedimentos para a realização das respectivas funções. A AVDNet6 administra todas as conexões efetuadas pelos participantes com o uso de *threads* de controle. Sistemas *multithreads* permitem que múltiplas linhas de execuções sejam estabelecidas e consequentemente fornecem agilidade e a funcionalidade necessária a sistemas que manipulem muitas informações concorrentes assim como os AVDs em larga escala.

O módulo *Cliente* é o responsável em realizar todos os mecanismos necessários para que a mensagem possa ser encaminhada por meio do ambiente virtual distribuído ao destinatário especificado, realizando as transformação necessária para uma mensagem se tornar um documento XML válido, aplicando todas as modificações inerentes ao protocolo IPv6, e enviando por meio de uma *thread* e com o uso de *sockets* ao destinatário. A Figura 4.3 apresenta a classe *AVD6Cliente*, que é a responsável em enviar as mensagens e realizar as transformações necessŕias no pacote de dados.

Já no módulo *Servidor*, a classe *AVD6Servidor* (Figura 4.4) quando ativada sua função de servidor, fica a espera de uma conexão por meio de um *socket* que está situado em uma *thread*. Quando este *socket* receber uma conexão, o servidor cria uma outra *thread* instanciada como referência a classe *AVD6ServidorExec* (Figura 4.5) para manipular as informações enviadas por intermédio do *socket* de conexão, liberando a *thread* associada a

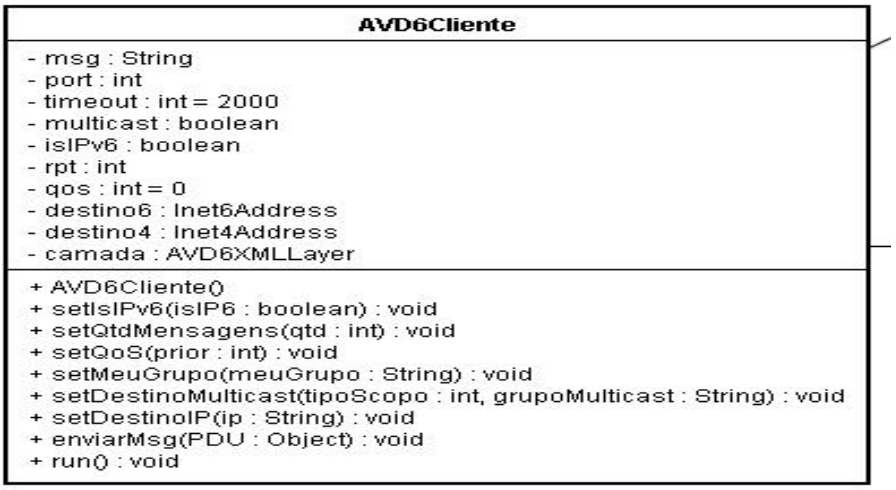

Figura 4.3: Classe AVD6Cliente

classe *AVD6Servidor* para ficar em modo de 'espera' para uma nova conexão.

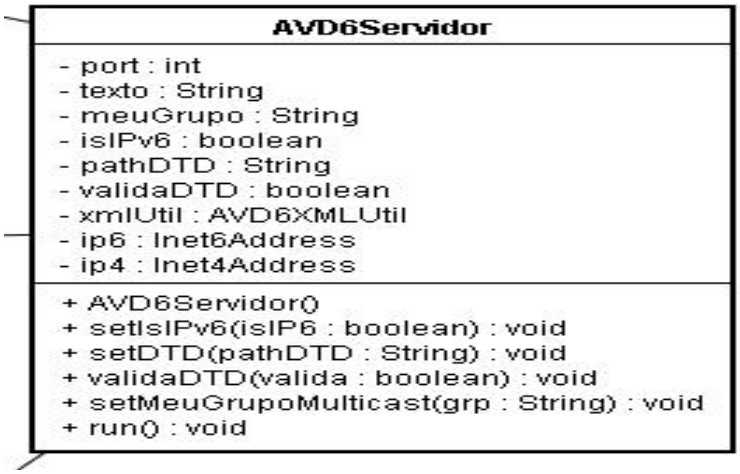

Figura 4.4: Classe AVD6Servidor

A modularidade implementada pelo AVDNet6 permite que cada módulo atue independentemente, tanto que garante recursos de usabilidade tanto para participantes que atuam como servidor, cliente, ou ambos.

A simplicidade de utilização do *framework* garante que o AVDNet6 seja utilizado nas mais diversas arquiteturas de sistemas distribuídos com alta confiabilidade. A curva de aprendizagem de usabilidade do *framework* é relativamente pequena, pois acima de tudo, o esforço de desenvolvimento na manutenção de parâmetros como simplicidade, integração e eficiência tornam-se fundamentais.

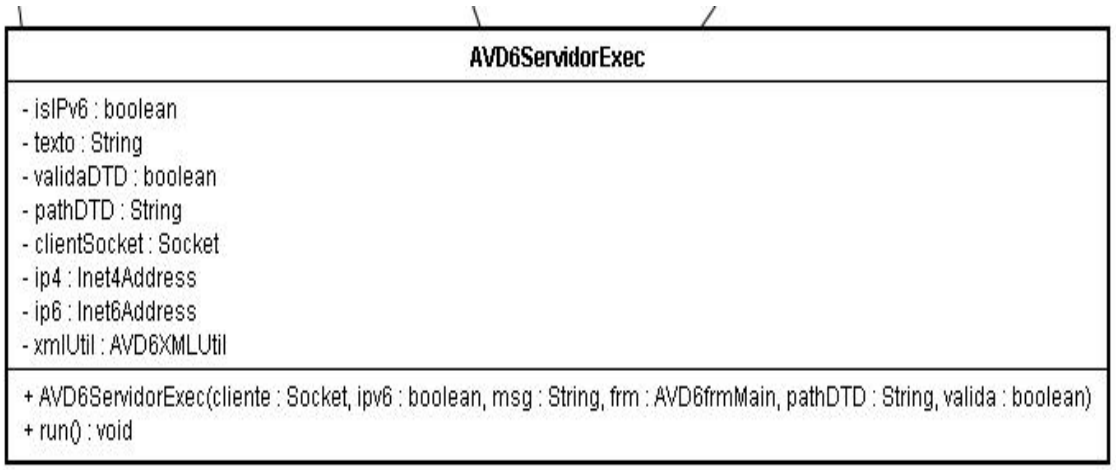

Figura 4.5: Classe AVD6ServidorExec

4.5 Troca de Mensagens

Historicamente, padrões de mensagem que são usados em AVDs são compilados por software e todas as entidades que participam do ambiente devem implementar esses padrões para que seja possível a interação com outros participantes.

Além do mais, tradicionalmente, AVDs podem operar somente com padrões de comunicação, comportamentos e objetos que estão presentes quando o AVD inicia, ou seja, se algum novo tipo de comunicação, objeto ou comportamente ncessite ser adicionado à arquitetura, o ambiente tem que ser parado, compilado e reiniciado para que as novas alterações tenham efeito.

Para que o *framework* AVDNet6 seja utilizado em AVDs não específicos, o PDU, unidade de transferência entre os participantes do AVD, deve ser manipulado de maneira padronizada pelo *framework* AVDNet6.

Como o *framework* AVDNet6 foi desenvolvido para atuar em diversas aplicações distribuídas, necessitamos de um padrão que possa ser utilizado em todas elas. A AVDNet6 utiliza o XML (*Extensible Markup Language*) para encapsular todas as informações de troca entre os participantes do AVD.

Duas Classes do AVDNet6, *AVD6XMLLayer* e *AVD6XMLUtil*, são as responsáveis em manipular mensagens XML e fazer as transformações entre PDU e XML, necessárias para que o *framework* AVDNet6 torne-se um sistema que possa ser utilizado em todos os AVDs possíveis.

A Classe *AVD6XMLLayer* pertence ao módulo *Cliente*, e possui um único método, cuja responsabilidade é transformar um PDU criado pelo sistema que utiliza o *framework*, e de formato desconhecido, em uma mensagem XML padronizada para o Ambiente Virtual Distribuído. A Figura 4.6 apresenta a Classe *AVDXMLLayer*.

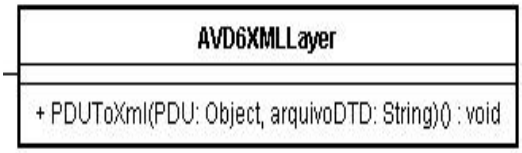

Figura 4.6: Classe AVD6XMLLayer

Já a Classe *AVD6XMLUtil* pertence ao módulo *Servidor*. A Classe possui diversos Métodos para manipular mensagens XML, dentre eles: validação de sintaxe e semântica; padronização; transformação de uma mensagem XML, utilizada no Ambiente Virtual Distribuído por meio do AVDNet6, para um PDU de formato conhecido ao sistema que utilize o AVDNet6. A Figura 4.7 apresenta a respectiva Classe.

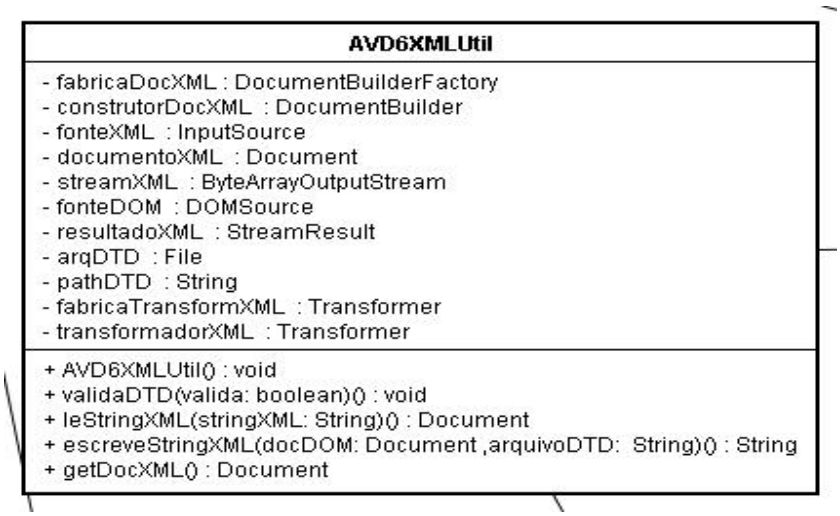

Figura 4.7: Classe AVD6XMLUtil

4.5.1 XML - *Extensible Markup Language*

A especificação da linguagem XML (W3C, 2007) define uma forma padronizada para a adição de marcações a documentos contendo informação estruturada.

A linguagem XML, um dialeto mais simples da SGML *(Standard Generalized Markup Language)*, foi projetada para facilitar a publicação de informaçẽs em ambientes distribuídos heterogêneos. Assim, qualquer sistema que possa suportar XML pode fazer parte desse ambiente heterogêneo, e inclusive interagir com ele.

A liberdade de criação dá à linguagem a flexibilidade e a capacidade de extensão para que ela se adpate a qualquer documento, sem restrições. Além do mais, XML se encaixa nas mais diversas plataformas de software, o que

auxilia uma transferência de dados padronizada entre elas (W3C, 2007).

Os principais benefícios do XML em sistemas distribuídos são:

XML permite que vários sistemas heterogêneos se comuniquem, sem a necessidade de compilar protocolos de comunicação específicos para o uso em sistemas distribuídos.

XML adapta-se bem a ambientes virtuais nas quais a necessidade de trabalhar com rápidas mudanças entre os participantes é fundamental. Inclusive em ambientes virtuais desconhecidos - de arquitetura diversas.

XML promove o reuso de informações e meios para localizá-las facilmente por meio de buscas semânticas, o que permite que as informações possam fluir entre aplicações diversas.

Assim, padrões de comunicação em um AVD podem ser criados e modificados em tempo de execução. A facilidade com que as mensagens são manipuladas permitem que possam explorar recursos específicos de um determinado ambiente obtendo uma melhor performance da rede, resultando em um AVD altamente extensível e dinâmico.

Para que o AVDNet6 possa suportar diversas aplicações distribuídas, o *framework* dispõe de um mecanismo capaz de transformar um objeto PDU, gerado pelo sistema que utiliza o *framework* e de estrutura desconhecida, em um documento XML que é posteriormente encapsulado no protocolo IPv6 para o envio aos participantes do AVD.

No AVDNet6 a classe responsável em fazer o mapeamento PDU para XML é a *AVD6XMLLayer* por meio do método *pduToXML(PDU,DTD)*. A classe *AVD6XMLLayer* atua como uma camada intermediária responsável em mapear objetos PDUs de arquitetura desconhecida em arquivos XML, que podem ser validados por arquivos DTD *(Document Type Definition)*.

Para que no *framework* AVDNet6 fosse possível manipular uma instância de um objeto de uma classe estruturalmente desconhecida, foi utilizado o conceito de *Reflexão Java*, ou também conhecido como *Introspecção Java*. *Reflexão* é a capacidade de um objeto de conhecer seus próprio dados, ou seja, obter o nome de todos os seu membros, como atributos e métodos, bem como executar um método usando a *Introspecção*. Porém, a *Reflexão* necessita de privilégios na execução, o que não pode funcionar se o *framework* AVDNet6 estiver rodando sob uma administração severa de segurança (Sun, 2007).

4.5.2 Processamento de PDUs em XML

Para que mensagens em XML possam ser manipuladas de acordo com o padrão W3C (*World Wide Web Consortium*) (W3C, 2007) é necessário a existência de uma analisador, também conhecido como *parser*, que dê suporte as funcionalidades necessárias para percorrer as estruturas destes documen-

tos: seus elementos e atributos.

Para o processamento XML, as APIs Java incluem tecnologias para manipulação desses documentos por intermédio de interfaces: DOM *Document Object Model* e SAX *Simple API for XML*. Por meio da interface DOM a leitura do documento é feita de uma só vez para a criação de uma representação na memória em forma de árvore, o que requer um uso extensivo da memória. Já na interface SAX o documento XML é lido por partes, enquanto aciona um método para cada elemento encontrado.

Para o processamento de PDUs em XML entre os participantes de um AVD utilizando o AVDNet6, a interface DOM foi escolhida. Como a maioria dos PDUs que são usados em AVDs são simples, geralmente de tamanho pequeno (Singhal and Zyda, 1999) e todas as informações presentes nos PDUs terão quer lidas e manipuladas, a representação por meio de uma árvore na memória se consolidou como uma solução ideal às informações contidas no PDU.

Os documentos XML podem ser classificados em duas categorias: os bem-formados e os válidos. Os bem-formados significam que seguem as especificações XML básicas: apenas um elemento raiz, todas as *tags* abertas devem ser fechadas, sem atributos repetidos, elementos aninhados corretamente, conter declaração XML na primeira linha do documento, elementos *case sensitive*. Naturalmente, a AVDNet6 dispõe de recursos que permitam que apenas documentos bem-formados XML possam trafegar em todo o ambiente virtual. Por intermédio do seu *parser*, documentos XML que não atendam tais requisitos de boa-formação são rejeitados pelo AVDNet6 e consequentemente não são recebidos pelo AVD. Por definição, se um documento não é bem-formado ele não é um documento XML (Santos et al., 2003).

Documentos XML válidos significam que cada instância do documento precisa ser bem-formado e estar de acordo com uma gramática de documentos XML. A AVDNet6 utiliza um DTD *(Document Type Definition)* para definir uma gramática de validação de documentos XML. Um DTD pode ser declarado dentro do próprio documento XML ou como referência externa.

No *framework*, o tarefa de criação do DTD é de responsabilidade do projetista do AVD. Permitindo um desenvolvimento flexível, utilizando um meio não proprietário, robusto, verificável e padronizado para armazenamento e transmissão de dados por meio de uma rede de comunicação (W3C, 2007).

No AVDNet6 o DTD é exclusivamente referenciado externamente e pode ser utilizado quando uma mensagem XML necessite ser validada para que tenha efeito no AVD, como por exemplo quando um participante envia a primeira mensagem para entrar no AVD, esta mensagem tem que ser validada, pois tem possibilidade de vir de qualquer sistema que manipule XML e pode estar

fora do padrão de PDU definido para tal fim.

O sistema distribuído que faz uso do AVDNet6 tem total controle sobre quando a mensagem XML é validada por meio do método *validaDTD(boolean)* e a definição do caminho do arquivo DTD por meio do método *setDTD(String)*, ambos métodos fazem parte da classe *AVD6Servidor* no módulo *Servidor*.

4.6 Testes e Resultados

A seguir são apresentadas as descrições dos testes efetuados com o uso do AVDNet6, e posteriormente são divulgados os resultados sobre o uso do *framework* como gerenciador de aplicações distribuídas.

4.6.1 Sistema de Apoio ao AVDNet6

Para a avaliação do *framework* AVDNet6, é importante que o *framework* seja instanciado em algum AVD. Porém, uma variedade de métodos e técnicas são implementados no desenvolvimento de AVDs para garantir a gerência de recursos, como largura de rede e capacidade de processamento, compressão e agregação de pacotes, servidores adicionais, gerenciamento de interesse, etc. Entretanto, a avaliação do *framework* não pode depender do processamento de algum AVD ou outro sistema específico, já que todo o processamento e gerência de recursos podem ser centralizados no AVD, não apresentando confiabilidade nos resultados apresentados nos testes com o *framework* AVDNet6.

Para o uso do *framework* AVDNet6 nas simulações foi necessária a criação de um sistema de apoio para fornecer funcionalidade ao *framework* e simular o comportamento de um AVD.

O sistema de apoio utiliza o *framework* AVDNet6 para simular as ações que são realizadas no uso de um Ambiente Virtual Distribuído. Por meio de envio de vários pacotes, com opções de roteamento definidas, os pacotes IPv6 são enviados ao destino especificado. O sistema de apoio é capaz de utilizar vários mecanismos do *framework* AVDNet6 que são utilizados no tratamento de pacotes e roteamento IPv6. A Figura 4.8 apresenta a GUI (*Graphical User Interface*) do Sistema de Apoio que utiliza o protocolo IPv6 para definir mecanismos utilizados no AVDNet6.

Para que o protocolo IPv6 apresente suas vantagens de uso, é necessário algum meio de verificação para que os resultados sejam comparados. Assim, é necessário efetuar testes com o protocolo IPv4 para que uma comparação consciente seja efetuada. A Figura 4.9 apresenta opções para simulação de envio de pacotes IPv4 pela rede para que sejam efetuadas comparações com o IPv6.

O sistema de apoio é capaz de gerar PDUs dinamicamente para que operações que envolvam troca de pacotes e avaliação de desempenho desses

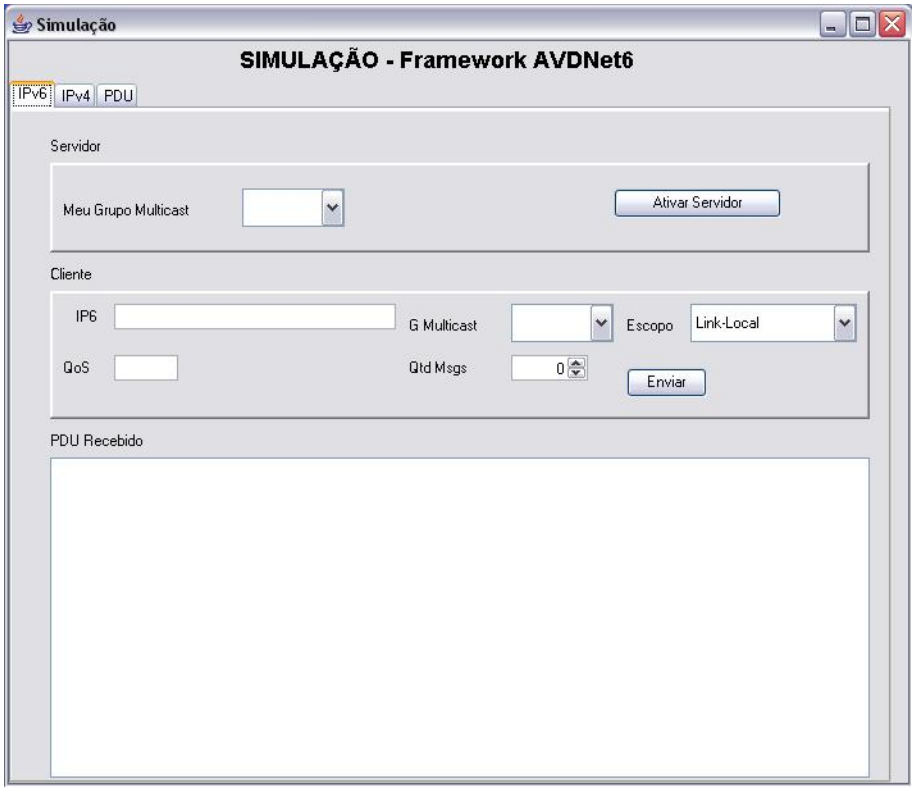

Figura 4.8: Sistema de Apoio IPv6 que utiliza o Framework AVDNet6 para testes

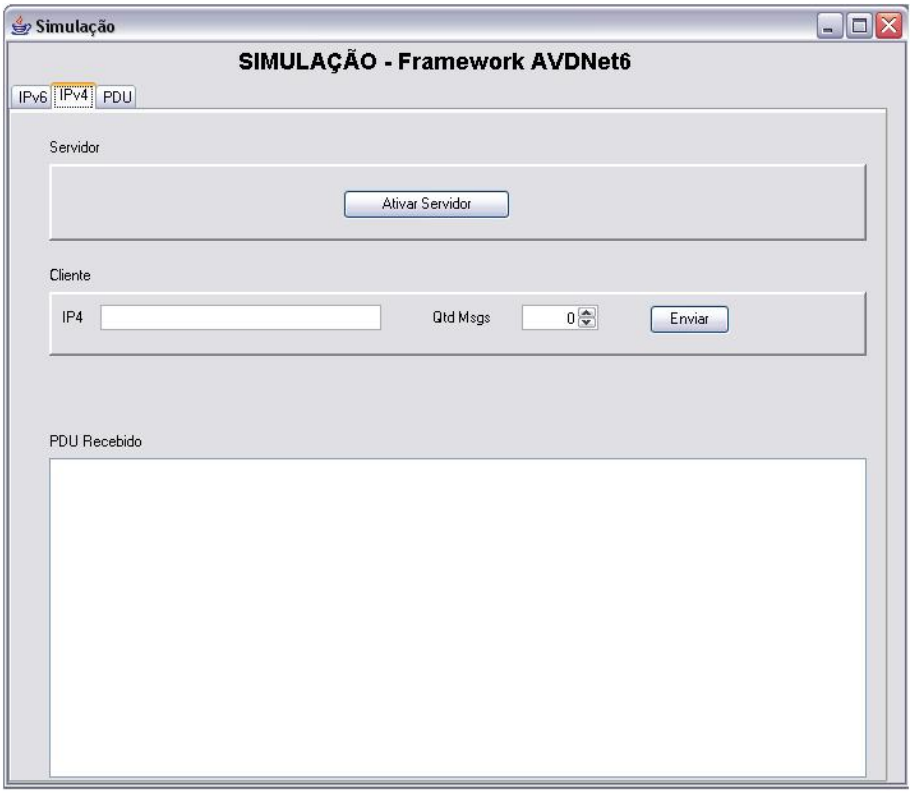

Figura 4.9: Sistema de Apoio IPv4 para testes

pacotes na rede de comunicação sejam avaliadas. Os PDUs são criados de acordo com a especificações definidas no arquivo DTD, e de forma dinâmica por meio das definições das variáveis do PDU.

Os PDUs são criados de forma dinâmica pois simulam um Ambiente Virtual Distribuído, na qual existem PDUs de vários formatos, e consequentemente o *framework* AVDNet6 deve ser capaz de manipular todos os PDUs gerados em um Ambiente Virtual Distribuído. A Figura 4.10 apresenta a GUI que gera o PDU de acordo com o formato especificado pelo DTD e tipos de variáveis.

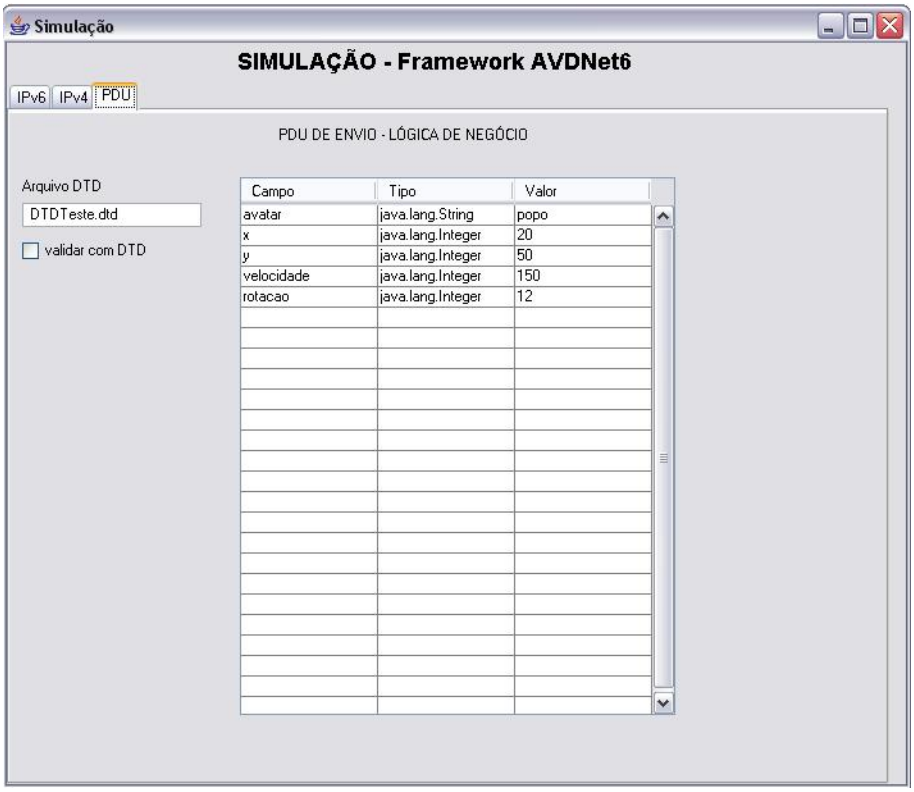

Figura 4.10: Geração de PDU dinamicamente para testes

4.6.2 Estrutura e Ferramentas Utilizadas

A estrutura da rede *Ethernet* de comunicação utilizada para as simulações foi feita com o uso de dois computadores utilizando o Sistema Operacional *GNU/Linux* e com um *switch* interligando-os. A rede foi configurada para aceitar ambos os tipos de protocolos: IPv4 e IPv6.

Para análise dos pacotes, utilizou-se um analisador de pacotes para que todo tipo de tráfego IPv6 ou IPv4 sejam capturados, chamado de *Ethereal* (Ethereal, Icom).

O *Ethereal* é um analisador de protocolos de rede, com uma interface gráfica de usuário. Este programa permite o gerenciamento e leitura de dados em tempo real em uma rede. Além disso, o *Ethereal* possibilita a criação de

filtros, para mostrar o que se deseja, com sintaxe aprimorada em relação aos outros analisadores de tráfego de rede (Ethereal, lcom). A Figura 4.11 apresenta o *Ethereal* em um processo de captura de tráfego em uma rede IPv6 de dois computadores.

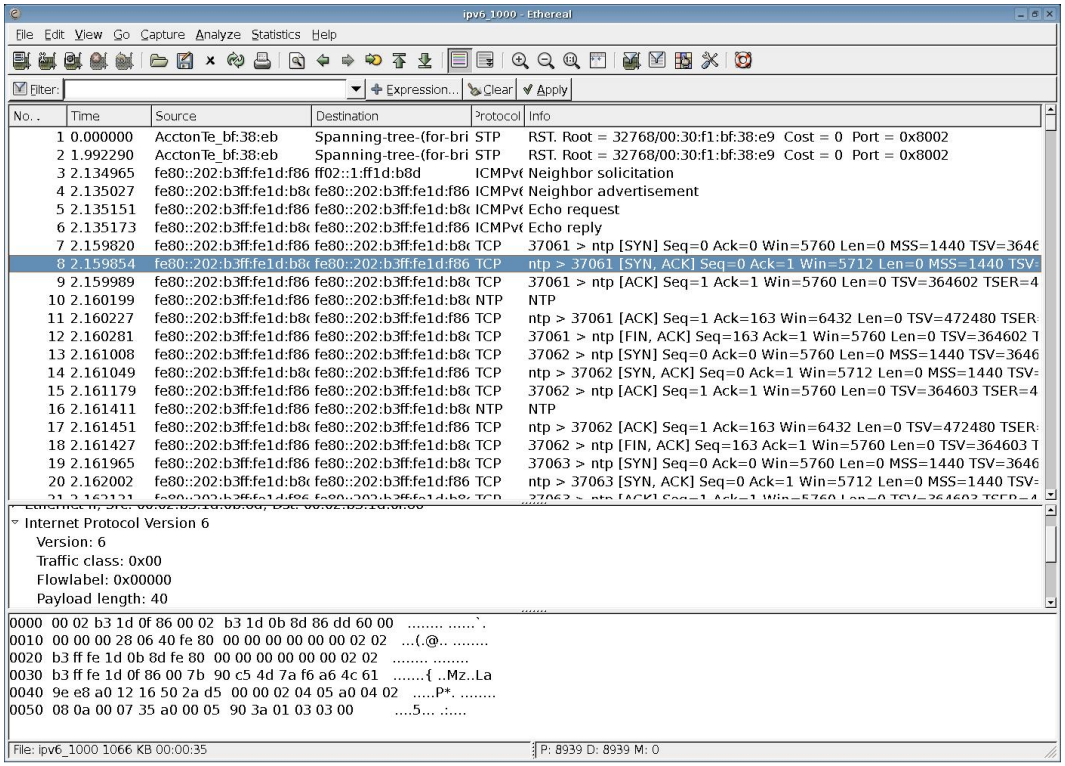

Figura 4.11: Ethereal

A partir dos dados do tráfego capturados pelos testes efetuados com o uso do *Ethereal*, é possível gerar análises sobre o *framework* AVDNet6.

4.6.3 Testes e Análise

Uma comparação da largura de banda consumida entre as conexões *unicast*, *multicast*, e o *multicast* em nível de aplicação proposto na AVDNet (Correia, 2005) é ilustrada na Figura 4.12. O aumento da quantidade de usuários no Ambiente Virtual Distribuído aumenta a Largura de Banda utilizada nos três meios de conexão.

A partir do Sistema de Apoio, um único PDU de informações foi gerado e repassado para o AVDNet6 que posteriormente foi enviado pela rede de comunicação em formato XML por meio de comunicação *unicast*. O tempo que o pacote IPv6 demorou para percorrer todo o caminho entre os dois computadores foi menor, e consequentemente um melhor resultado na taxa de transferência do que o mesmo PDU empacotado pelo IPv4. A Figura 4.13 mostra a relação entre tamanho e tempo de envio dos pacotes IPv6 e IPv4.

O pacote IPv6, apesar de apresentar uma maior cabeçalho de dados do que

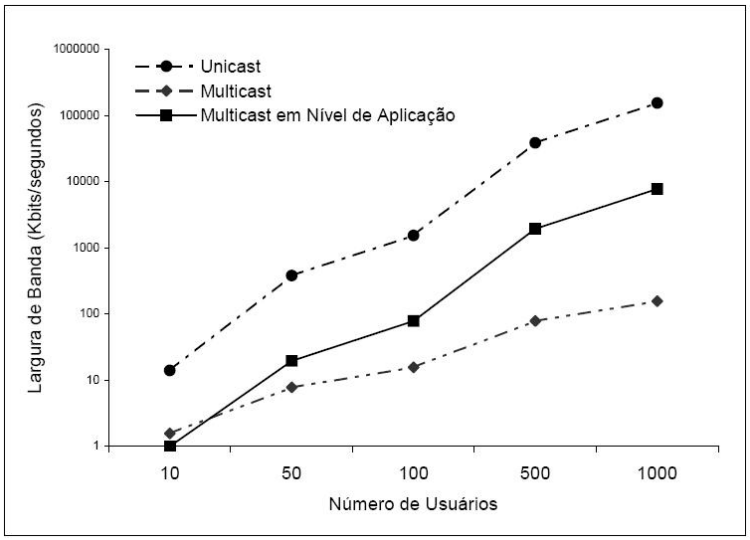

Figura 4.12: Tráfego Unicast x Multicast x Multicast em nível de aplicação

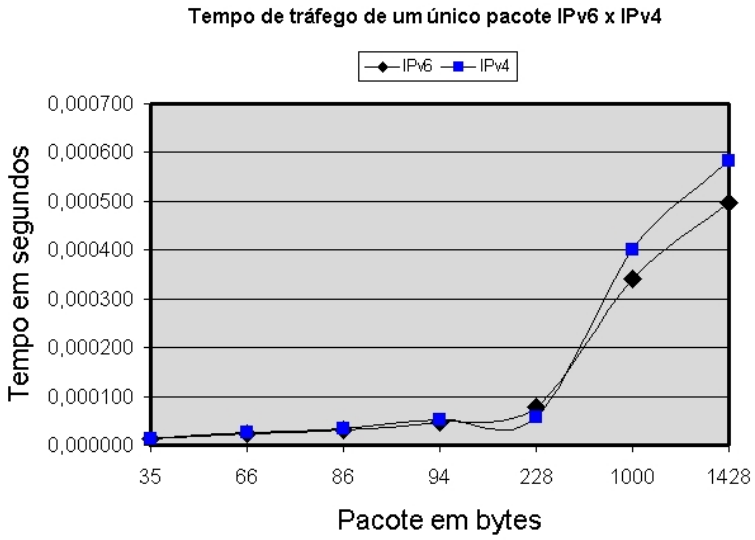

Figura 4.13: Relação entre tamanho e tempo no IPv6 e IPv4

o IPv4, obtêm um ganho significativo no roteamento pelo *switch*.

Entretanto no modelo de tráfego de AVD, as estações geram uma grande quantidade de pacotes. As taxas de ativação desses serviços estão geralmente relacionadas com as ações realizadas pelos participantes do ambiente.

Os PDUs de atualização e estado dos participantes são frequentes e de extrema importância em AVDs. Toda interação criada no AVD gera um PDU correspondente de atualização ou estado, como por exemplo: entrada de um novo participante no sistema, interações com objetos pelos participantes, movimentação pelo ambiente, etc.

CAPÍTULO 5

Conclusão

5.1 Considerações Iniciais

A Monografia teve como objetivo apresentar o *framework* AVDNet6, acompanhando a evolução dos vários cenários de redes, e apresentou algumas tecnologias, técnicas e conceitos de Ambientes Virtuais Distribuídos disponíveis. Logo a seguir, apresentou também os conceitos básicos relativos ao protocolo IPv6, analisando o impacto de sua implementação em sistemas distribuídos. Após a verificação detalhada de alguns mecanismos IPv6 individualmente, constata-se que, da utilização do IPv6, aparecem vários desafios técnicos no que concernem à sua interoperabilidade com as atuais ferramentas de gerenciamento e administração de redes de computadores.

A seguir, uma Conclusão geral é apresentada sobre a Monografia desenvolvida. Após, são apresentados os Trabalhos Futuros a serem desenvolvidos.

5.2 Conclusão

Como caracterizado ao longo dessa Monografia, a concepção de Ambientes Virtuais Distribuídos escaláveis envolve conceitos de várias áreas distintas, como sistemas distribuídos, redes de computadores, computação gráfica. As soluções apresentadas pelas arquiteturas de Realidade Virtual Distribuída existentes ou foram projetadas levando-se em consideração determinadas condições de infra-estrutura ou foram definidas para dar suporte a aplicações específicas. Além disso, suportam um número limitado de usuários.

O IPv4, versão atual do protocolo IP, não antecipou e, conseqüentemente, não contempla as necessidades atuais de sistemas distribuídos. O problema

mais evidente é o crescimento exponencial da Internet e resultante ameaça de exaustão do espaço de endereçamento IPv4. Logo, técnicas como NAT e CIDR foram desenvolvidas para resolver paliativamente este problema.

Aliado ao problema de endereçamento IP, a necessidade de configuração simplificada, a ausência de mecanimos de segurança mais robustos e a necessidade de suporte melhorado para entrega de dados em tempo-real estimularam a *Internet Engineering Task Force* (IETF) a desenvolver um conjunto de protocolos e padrões conhecido como IP versão 6 (IPv6).

A necessidade do IPv6 é premente, e as ferramentas necessárias para sua utilização estão prontamente disponíveis. As iniciativas de utilização, e ampla pesquisa, surgem a todo momento: vários provedores de serviços importantes disponibilizam acesso via IPv6, diversos estudos são desenvolvidos sobre o IPv6, softwares já implementam IPv6 nativamente. Enfim, existem boas documentações e administradores de rede capacitados em IPv6.

O *framework* AVDNet6 é apresentado como uma solução para que sistemas distribuídos possam dispor de um melhor meio de comunicação e roteamento pela rede de comunicação, por meio da utilização do protocolo IPv6 nativamente.

Os resultados apresentados pelas simulações do AVDNet6 mostram a potencialidade do protocolo IPv6 em integrar Sistemas Distribuídos e conseqüentemente Ambientes Virtuais Distribuídos, demostrando que o IPv6 obtêm um melhor desempenho em sistemas de tráfego intenso pela rede. Pacotes maiores de informações IPv6 apresentam melhor tempo de envio do que pacotes de informações de tamanho menor, explorando melhor os recursos de rede e melhorando o processo de roteamento pelos equipamentos de rede.

Os resultados da avaliação da AVDNet6, com base na comunicação *multicast* nativa, mostram a capacidade de suportar um grande número de usuários conectados no ambiente e a baixa utilização de largura de banda disponível em comparação ao *multicast* em nível de aplicação proposto na AVDNet (Correia, 2005).

A extensão da camada de comunicação na AVDNet permite que alguns procedimentos e métodos possam ser retirados, movendo toda essa responsabilidade para o AVDNet6. Além da AVDNet, outros AVDs podem usufruir dessa vantagem de retirar a carga extra do sistema.

A flexibilidade do *framework* AVDNet6 permite que sistemas distribuídos com configurações heterogêneas possam utilizar o *framework* com facilidade. Estas são características requeridas pela maioria dos *frameworks*, porém alcançada por poucas.

A contribuição mais relevante é a concepção da arquitetura de software AVDNet6, especificada e implementada na estrutura de um *framework*, po-

dendo ser utilizado para o desenvolvimento de aplicações completas ou ser integrado em aplicações legadas que requerem tais características.

5.3 Trabalhos Futuros

Como padrão de mensagens para envio de dados, a linguagem XML não é um método compacto para representar dados, uma mensagem escrita em XML é muito maior que a binária equivalente (Serin, 2003). Para perspectivas futuras, técnicas de compressão XML podem ser usadas para produzirem mensagens de menor tamanho e possibilitar que menos recursos da rede de comunicação sejam utilizados.

Uma possibilidade de compressão é substituir *tags* XML por *tokens* binários (Serin, 2003). Por exemplo, quando uma árvore XML é construída, *tags* marcadas previamente são substituídas por *tokens* binários equivalentes. O resultado é uma árvore XML mais compacta e um tamanho reduzido de mensagem.

Outro aspecto que precisa ser investigado está relacionado com a comunicação *anycast*. Esse mecanismo do IPv6 pode ser muito útil para evitar sobrecarga da rede e garantir um posicionamento estratégico dos servidores em relação aos participantes de um Ambiente Virtual Distribuído.

Adicionalmente, é possível expandir a AVDNet6 explorando características e funcionalidades do *anycast* para balanceamento de carga em Ambientes Virtuais Distribuídos em larga escala e para Sistemas Distribuídos não específicos.

Referências Bibliográficas

Abreu, B. R. C. (2006). Unindo sistemas e bancos de dados distribuídos. *Trabalho de Conclusão de Curso - Ciência da Computação - Universidade Federal de Pernambuco, Recife*.

Aukstakalnis, S. and Blatner, D. (1992). Silicon mirage: the art and science of virtual reality. *Berkeley, CA*.

Balikhina, T., Ball, F., and Duce, D. (2002). Distributed virtual environments - an active future? *The 20th Eurographics UK Conference, De Montfort University*.

Bieringer, P. (2006). Linux ipv6 howto,release 0.51. *http://www.bieringer.de/linux/IPv6*.

Capin, T. K., Pandzic, I. S., Magnenat-Thalman, N., and Thalman, D. (1999). *Avatars in Networked Virtual Environments*. John Wiley and Sons.

Comer, D. E. (2006). *Internetworking with TCP/IP, v.1*. Elsevier.

Correia, R. C. M. (2005). *AVDNet-Arquitetura para Ambientes Virtuais Distribuídos Escaláveis Baseada na Infra-Estrutura Atual da Internet*. Instituto Tecnológico de Aeronáutica, São José dos Campos, SP.

Correia, R. C. M. and Pellegrino, S. R. M. (2007). *Análise de Escalabilidade em Ambientes Virtuais Distribuídos por Meio do Multicast em Nível de Aplicação*. IX Symposium on Virtual and Augmented Reality, Petrópolis, Rio de Janeiro.

de Ensino e Pesquisa RNP, R. N. (http://www.rnp.br/ipv6/ipv6-rnp.html). Ipv6 na rnp. *Acessado em 16/05/2007*.

Eraslan, M., Georganas, N. D., R., G. J., and D., M. (2007). A scalable network architeture for distributed virtual environments with dynamic qos over ipv6. *School of Information Technology and Engineering, University of Ottawa, Ontario, Canada*.

Ethereal (http://www.ethereal.com/). A network protocol analyzer. *Acessado em 15/10/2007*.

Filipe, M. J. N. and Martins, J. L. (2007). Multicasting - da teoria à prática. *Universidade de Évora e Departamento de Informática - FCT-UNL, Portugal*.

Hancock, D. (January, 1995). Viewpoint: virtual reality in search of middle ground. *IEEE Spectrum*.

Hand, C. (1994). Other faces of virtual reality. *First International Conference MHVR'94 - Lectures Notes in Computer Science n.1077, Moscow, Russia*.

Júnior, J. B. S., Bícego, E., Lago, K., Carvalho, L. B., and Souza, E. P. (1999). Um framework para o uso de novas tecnologias para compartilhamento e distribuição de informação em ambientes de ensino cooperativos distribuídos. *Ciência da Computação, UNIFENAS - Universidade José do Rosário Vellano - Minas Gerais*.

Kurose, J. F. and Ross, K. W. (2006). *Redes de computadores e a Internet*. Pearson Addison-Wesley.

Latta, J. N. and Oberg, D. J. (January, 1994). A conceptual virtual reality model. *IEEE Computer Graphics and Applications*.

Leite Júnior, A. J. M. (2000). *ATAXIA: Uma arquitetura para viabilização de NVE's voltados para a Educação a Distância através da Internet*. Dissertação de Mestrado - Ciência da Computação - Universidade Federal do Ceará, Fortaleza.

Microsoft (http://www.microsoft.com). Windows. *Acessado em 20/10/2007*.

Neto, P. F. L. (2005). 6ix linux: divulgando e intensificando a utilização do novo protocolo da internet (ipv6). *Trabalho de conclusão de curso - Engenharia da Computação - Universidade de Pernambuco, Recife*.

Netto, A. V., Machado, L. S., and Oliveira, M. C. (2002). *Realidade Virtual - Fundamentos e Aplicações*. Visual Books.

Organization, L. K. (http://www.kernel.org). The linux kernel archives. *Acessado em 21/10/2007*.

RFC1191 (http://www.ietf.org/rfc/rfc1191.txt - December 1998). Path mtu discovery. *Acessado em 15/10/2007*.

RFC1546 (http://www.ietf.org/rfc/rfc1546.txt - November 2003). Host anycasting service. *Acessado em 17/10/2007*.

RFC2402 (http://www.ietf.org/rfc/rfc2402.txt - December 1998). Ip authentication header. *Acessado em 14/10/2007*.

RFC2460 (http://www.ietf.org/rfc/rfc2460.txt - December 1998). Internet protocol, version 6 (ipv6) - specification. *Acessado em 13/10/2007*.

RFC2461 (http://www.ietf.org/rfc/rfc2461.txt - November 2003). Neighbor discovery for ip version 6 (ipv6). *Acessado em 17/10/2007*.

RFC2462 (http://www.ietf.org/rfc/rfc2462.txt - December 1998). Ipv6 stateless address autoconfiguration. *Acessado em 16/10/2007*.

RFC2464 (http://www.ietf.org/rfc/rfc2464.txt - December 1998). Transmission of ipv6 packets over ethernet networks. *Acessado em 15/10/2007*.

RFC3697 (http://www.ietf.org/rfc/rfc3697.txt - March 2004). Ipv6 flow label specification. *Acessado em 13/10/2007*.

RFC4303 (http://www.ietf.org/rfc/rfc4303.txt - December 2005). Ip encapsulating security payload (esp). *Acessado em 14/10/2007*.

RFC4649 (http://www.ietf.org/rfc/rfc4649.txt - August 2006). Dynamic host configuration protocol for ipv6 (dhcpv6) relay agent remote-id option. *Acessado em 16/10/2007*.

Santos, C. A. S., Passos, C., and Marques, M. (2003). Tecnologias para o processamento de documentos xml: Uma abordagem java. *Universidade Salvador - CEPERC - Centro de Pesquisas Interdepartamental em Redes de Computadores - Salvador,BA*.

Serin, E. (2003). *Design and Test of the Cross-Format Schema Protocol (XFSP) for Networked Virtual Environments*. Naval Postgraduate School, Monterey, California.

Silva, A. J. S. and Faria, M. R. (RNP - Rede Nacional de Ensino e Pesquisa - Boletim trimestral sobre tecnologia de redes). Hierarquia de endereços ipv6. *16 de Março de 2001, volume 5, número 2*.

Singhal, S. and Zyda, M. (1999). *Networked Virtual Environments*. Addison-Wesley.

Sun, M. (http://java.sun.com - November 2007). Sun developer network. *Acessado em 20/11/2007*.

Tanenbaum, A. S. (1995). *Distributed Operating Systems*. Prentice Hall.

W3C, W. W. W. C. (http://www.w3.org/XML/ - November 2007). Xml w3c. *Acessado em 20/11/2007*.