UMA PLATAFORMA ESCALÁVEL PARA SISTEMAS DISTRIBUÍDOS DE REALIDADE VIRTUAL

por

Renata Cruz Teixeira

UFRJ

Tese submetida para a obtenção do título de

Mestre em Ciências em Engenharia Elétrica

ao Programa de Pós-Graduação de Engenharia Elétrica

da COPPE/UFRJ

UMA PLATAFORMA ESCALÁVEL PARA SISTEMAS DISTRIBUÍDOS DE REALIDADE VIRTUAL

Renata Cruz Teixeira

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Otto Carlos Muniz Bandeira Duarte, Dr.Ing.

Prof. Aloysio de Castro Pinto Pedroza, Dr.Ing.

Prof. Bruno Feijó, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

JULHO DE 1999

TEIXEIRA, RENATA CRUZ

Uma Plataforma Escalável para Sistemas Distribuídos de Realidade Virtual [Rio de Janeiro] 1999

XIV, 69 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia Elétrica, 1999)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Realidade Virtual Distribuída
2. Protocolos Multidestinatários
3. Escalabilidade

I. COPPE/UFRJ II. Título (Série)

Ao vô Romeu.

Agradecimentos

Ao Flavio, pelo companheirismo.

Aos meus pais, pelo apoio incondicional.

Ao meu orientador, Professor Otto Carlos Muniz Bandeira Duarte, pela sinceridade e pelo apoio.

À Cacá, à Aninha e à Vó Toninha.

Aos demais membros da banca examinadora, profs. Aloysio de Castro Pinto Pedroza e Bruno Feijó.

Ao Ronaldo, ao Rubizinho e ao Rezende, pela paciência.

Aos estrangeiros: LuisH, Baiano, Daniel, Michel, Célio e Márcio.

Aos amigos do GTA: Marcos Tadeu (valeu pela máquina), Jonny (com suas dicas providenciais), Paulo André, Fred, Bill, Artur, Prof. Mauros, Belém, Emerson, Antguria, Andréa, Hermes, Sílvia, Vidal, Alexandre, Iniciações Científica (Pedro, Doc, Bernardo, Tadeu, Fred, Éric), Jucele, Valentim, Miranda, Ezequiel, Duboc e Gardel.

Aos Profs. Valmir, Aguiar, Leão, Richard e Lanza.

Ao pessoal de potência: Tony, Evandro, Luís Oscar e Pedro.

Ao pessoal da secretaria: Solange, Bia, Evelyn e Wilson.

Aos amigos: Ana, Tarzan, Sino, Enylton, Eduardo (Senna), Mônica, Glads, Lúcia, Ratton e Carlos.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UMA PLATAFORMA ESCALÁVEL PARA SISTEMAS DISTRIBUÍDOS DE REALIDADE VIRTUAL

Renata Cruz Teixeira

Julho de 1999

Orientador: Otto Carlos Muniz Bandeira Duarte
Programa: Engenharia Elétrica

Este trabalho propõe e analisa o desempenho de uma plataforma para sistemas distribuídos de realidade virtual. São apresentados a arquitetura e o protocolo propostos para esta plataforma junto com os princípios que guiaram seu desenvolvimento. Os principais objetivos da plataforma são: a escalabilidade, para possibilitar sua utilização por um grande número de usuários, e a flexibilidade, para permitir vários tipos de aplicações de realidade virtual distribuída.

A escalabilidade e a flexibilidade da plataforma são avaliadas através de simulações variando o número de participantes, o particionamento do ambiente virtual, a velocidade de deslocamento dos participantes e o tamanho do ambiente virtual. Estas simulações mediram os atrasos para entrar no mundo e trocar de grupos, o número de trocas de grupos e o número de mensagens trocadas pelo sistema de comunicação. Os resultados indicam que o particionamento possibilita um bom desempenho à plataforma proposta, permitindo o aumento do número de usuários e da velocidade dos participantes sem prejudicar a interatividade.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A SCALABLE PLATFORM FOR DISTRIBUTED VIRTUAL REALITY SYSTEMS

Renata Cruz Teixeira

July 1999

Advisor: Otto Carlos Muniz Bandeira Duarte
Department: Electrical Engineering

This work describes and analyzes the performance of a platform for distributed virtual reality systems on the Internet. We present the principles that have guided the platform design, together with an architecture and a protocol for this platform. The main goals of this platform are: scalability, to enable its use by a large number of users, and flexibility, to be suitable for various kinds of distributed virtual reality applications.

We investigate the scalability and flexibility of the platform via simulation, varying the number of participants, the partition of the virtual environment, the participants' speed, and the size of the virtual environment. These simulations have assessed the delay both to enter the virtual world and to exchange groups, the number of group exchanges, and the number of messages exchanged by the communication system. The results show that the division of the virtual environment enables a good performance of the proposed platform. This partition allows increasing both participants' speed and the number of participants, assuring good interactivity.

Palavras-chave

Realidade Virtual Distribuída

Protocolos Multidestinatários

Escalabilidade

Glossário

ACK : Reconhecimento positivo (*acknowledgement*);

BD : Base de dados;

CORBA : *Common Object Request Broker Architecture*;

DIS : Simulação interativa distribuída (*Distributed Interactive Simulation*);

IP : *Internet Protocol*;

MBone : *Multicast Backbone*;

NACK : Reconhecimento negativo (*Negative acknowledgement*);

ns : *Network Simulator*;

PDU : Unidade de dados do protocolo (*Protocol Data Unit*);

SRM : *Scalable Reliable Multicast*;

TCP : *Transmission Control Protocol*;

UDP : *User Datagram Protocol*;

VRML : *Virtual Reality Modeling Language*;

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

OS SISTEMAS de realidade virtual buscam a imersão de seus usuários em um ambiente tri-dimensional gerado por computador. Neste ambiente, também chamado de mundo virtual, os usuários podem interagir entre si e com o próprio ambiente de forma similar à interação no mundo real ou através de outros tipos de interação que não são viáveis na realidade. Assim, a realidade virtual permite uma interação homem-máquina mais natural e poderosa, tornando-se uma ferramenta bastante atrativa.

Alguns autores diferenciam o termo realidade virtual de ambiente virtual. Neste caso, o termo realidade virtual significa uma combinação de tecnologias cujas interfaces com os usuários humanos podem dominar seus sentidos de forma que eles interajam intuitivamente com um ambiente gerado por computador imersivo e dinâmico. Enquanto ambiente virtual é uma representação tri-dimensional gerada por computador que apenas sugere um espaço real ou imaginário, não tendo o realismo fotográfico e o senso total de imersão como objetivos principais. Neste trabalho os termos realidade virtual e ambiente virtual são utilizados como sinônimos com o significado de um ambiente que não tem o compromisso com a imersão total do usuário.

A realidade virtual pode ser utilizada em diversas aplicações [1, 2] como: educação [3], treinamento [4, 5], telemedicina, arquitetura, trabalho cooperativo [6, 7], visualização científica entre outras. Estas aplicações podem ser classificadas de acordo com: o tipo de ambiente, se o ambiente é puramente virtual ou se há alguma interação com o mundo real; a participação do usuário, se o usuário é ativo – modificando o ambiente – ou passivo; a distribuição do sistema, se o sistema está concentrado em uma única estação ou se está divido em estações distintas; e o número de usuários participando simultaneamente do mundo virtual. Esta classificação visa determinar os tipos de aplicações de realidade virtual, diferentemente da classificação apresentada em [8] que determina os tipos de ambientes virtuais.

Os ambientes virtuais distribuídos, ou os sistemas distribuídos de realidade virtual, são aqueles que possibilitam o compartilhamento do mundo virtual por usuários geograficamente dispersos. Este compartilhamento é importante por duas razões principais [9]. Primeiro, a capacidade de interação com outros usuários em um mesmo ambiente virtual aumenta o envolvimento e a credibilidade do usuário neste ambiente. Segundo, este compartilhamento permite ampliar o alcance das aplicações de realidade virtual, possibilitando sua utilização nas mais diversas áreas do conhecimento.

Em educação à distância pode-se obter a interação entre alunos e professores, enriquecendo o aprendizado através de um ambiente relacionado ao tema da aula. Para treinamento, pode-se usar o ambiente virtual tanto para treinar situações de perigo ou em lugares de difícil acesso, como para treinar, em conjunto, membros de uma equipe que estão em lugares diferentes. Na área de trabalho cooperativo há a possibilidade de realização de tarefas em comum utilizando a interação com o ambiente como ferramenta (análise de um protótipo, a visualização do interior de um ambiente sendo projetado). Para entretenimento, além de diversos tipos de jogos, pode-se utilizar a realidade virtual para o turismo.

Idealmente, os sistemas de realidade virtual ofereceriam uma interação com o usuário completamente imersiva. Porém, ainda existem muitas restrições que devem ser contornadas para que esta sensação de imersão seja satisfeita. Os sistemas distribuídos de realidade virtual são sistemas complexos, que representam vários sistemas agregados em um só [10]. Eles são sistemas distribuídos, precisando gerenciar a utilização dos recursos da rede de comunicação, contornando as perdas de dados, as falhas na rede e a concorrência. São também aplicações gráficas, precisando manter uma taxa de exibição de quadros em tempo real e dividir o tempo de processamento entre a renderização e as outras tarefas. Além disto, são aplicações interativas, precisando processar em tempo real as entradas dos usuários.

O enfoque deste trabalho são as limitações impostas pelo sistema de comunicação e como otimizar a utilização dos recursos deste sistema visando viabilizar o desenvolvimento de ambientes virtuais distribuídos. A rede de computadores possui limitações [11] como: banda passante, atraso e perdas de pacotes. A banda passante disponível determina o tamanho e a riqueza do ambiente virtual. A interatividade e o dinamismo deste ambiente são influenciados pelo atraso imposto pela rede. As perdas de dados podem causar inconsistências na percepção do mundo virtual, por isso, devem ser controladas através de protocolos confiáveis, que introduzem mais atraso na comunicação.

A Internet é a plataforma ideal para os ambientes virtuais distribuídos devido ao grande número de usuários que a utilizam em todo mundo. A infra-estrutura necessária para a representação dos mundos virtuais está sendo disponibilizada nos navegadores (*browsers*) da *World Wide Web* (WWW). Por exemplo, a linguagem VRML (*Virtual Reality Modeling Language*) [12] permite aos usuários carregar modelos tri-dimensionais interativos pela WWW. O padrão *Living Worlds* [13] agrega a capacidade de acesso multi-usuário aos modelos VRML.

Em redes de longa distância, como a Internet, as restrições do sistema de comunicação são agravadas [14, 11]. Nestas redes o atraso é maior devido à distância e aos congestionamentos e a banda passante em geral é menor devido ao seu compartilhamento por um número maior de usuários. Além disso, o número de usuários potencial de um mundo virtual em uma rede de longa distância é maior devido a sua abrangência. Atualmente, os sistemas distribuídos de realidade virtual visam o desenvolvimento de ambientes escaláveis para um grande número de usuários [15, 14, 11, 16], ou seja, ambientes que suportem um grande número de usuários sem a degradação de seu serviço. Para isto, é importante explorar as características destes ambientes.

1.1 Características dos Ambientes Virtuais

Um mundo virtual é composto por objetos. Os usuários deste mundo são representados por um objeto chamado avatar. Os avatares são imagens tri-dimensionais – que podem incluir um vídeo – e servem como entrada padrão para o usuário que o controla. Um avatar não é necessariamente a imagem de um ser humano. Em uma aplicação militar, por exemplo, um usuário pode estar controlando um avião e, portanto, seu avatar é a representação deste avião.

Nem todos os objetos de um mundo virtual são controlados pelos usuários. Existem objetos, chamados agentes ou robôs, que são controlados por simulações de computador e operam de forma relativamente autônoma. Agentes que exibam comportamento similar ao humano são importantes para a criação de ambientes mais complexos. Eles permitem a sensação de um número maior de participantes, sem o custo de um ambiente com muitos usuários [15]. Além destes dois tipos de objetos, há também objetos fixos que ajudam a compor o ambiente virtual.

Estes objetos estão representados em um ambiente compartilhado que deve apresentar as mesmas características para todos os usuários. Ao entrar em um mundo virtual, um participante pode ver o avatar dos outros que estão neste espaço e os outros podem ver o seu avatar, percebendo o comportamento uns dos outros imediatamente. As informações que descrevem um objeto e seu comportamento são o seu estado. Como um participante pode se movimentar pelo mundo virtual, interagir com ele (criando ou modificando objetos) e interagir com outros participantes, seu estado não permanece fixo. Por isto, torna-se necessário que os sistemas, para suportar este tipo de aplicação, notifiquem as modificações de estado a todos os participantes do ambiente virtual, garantindo que todos os usuários tenham uma visão consistente deste ambiente compartilhado.

Assim, um requisito importante dos sistemas distribuídos de realidade virtual é a manutenção de consistência do mundo virtual. Por exemplo, em um ambiente virtual para treinamento de motoristas. Todos os motoristas devem perceber os movimentos de todos os objetos imediatamente para controlar o veículo de forma correta. As inconsistências poderiam acarretar atropelamentos e batidas sem que o motorista tivesse meios para reagir evitando estes acidentes.

Outro requisito destes sistemas é o controle de concorrência no acesso aos objetos do ambiente. Além de atualizar o estado de todos os objetos constantemente, um sistema distribuído de realidade virtual é responsável por controlar o acesso a estes objetos. Se dois participantes tentarem acessar simultaneamente um mesmo objeto, apenas um deve ser capaz de concluir a ação. Suponha um ambiente virtual que representa um jogo de guerra em que dois avatares representando seres humanos lutam por uma arma, por exemplo. Para que a representação do mundo não fique inconsistente, apenas um deles deve conseguir a arma. Logo, o ambiente virtual deve ter alguma política de acesso aos objetos, mantendo a exclusão mútua.

Além destes dois requisitos, também é importante para este tipo de sistema a viabilidade de comunicação direta entre os participantes. Esta comunicação pode ser realizada através de textos, gestos ou voz. A capacidade de comunicação adiciona uma sensação de realismo necessária para qualquer ambiente simulado [10].

Para atender a estes requisitos de uma representação consistente do mundo virtual, os sistemas distribuídos de realidade virtual devem trocar mensagens pela rede contendo os estados dos objetos e as informações de acesso aos objetos. Como estas mensagens são trocadas entre todos os participantes, o volume de dados trafegando na rede é muito grande. Porém, se forem considerados alguns aspectos conhecidos da informação a ser distribuída, pode-se reduzir tanto a quantidade de mensagens enviadas quanto a freqüência destas mensagens. Alguns sistemas aproveitam-se deste conhecimento para amenizar as limitações impostas pela rede. No próximo capítulo são apresentados alguns destes sistemas e os mecanismos usados por eles para otimizar a utilização do sistema de comunicação.

1.2 Objetivo e Organização do Trabalho

Este trabalho propõe uma plataforma que utiliza os serviços dos protocolos de transporte disponíveis na Internet, visando atender aos requisitos das aplicações distribuídas de realidade virtual. A plataforma é responsável pela distribuição dos dados, mantendo a consistência do ambiente virtual. Para suportar um maior número de usuários, é feita a divisão dos participantes em grupos multipontos. O desempenho da plataforma é analisado através da simulação de aplicações de realidade virtual com requisitos diferentes. Foram gerados cenários de simulação buscando representar alguns requisitos mais críticos para o desempenho da plataforma: o particionamento do ambiente virtual, a velocidade de deslocamento dos participantes, o número de usuários e o tamanho do mundo virtual. Estes cenários foram utilizados para estabelecer o atraso, tanto para entrada quanto para troca de grupos, o número de troca de grupos, decorrentes da movimentação dos participantes no mundo virtual, e o número de mensagens trocadas.

Este trabalho está organizado da seguinte forma. O capítulo 2 apresenta os mecanismos encontrados na literatura para otimizar a utilização dos recursos do sistema de comunicação. Além disto, são apresentados neste capítulo alguns sistemas distribuídos de realidade virtual e discutidas suas principais características e fundamentos. Baseada em alguns conceitos introduzidos por estes sistemas, é proposta uma plataforma para ambientes virtuais distribuídos. O capítulo 3 apresenta as principais características desta plataforma e propõe uma arquitetura e um protocolo para esta plataforma. O capítulo 4 descreve a infra-estrutura e o modelo adotados para as simulações realizadas e analisa o desempenho da plataforma em condições diferentes, considerando os requisitos de aplicações diferentes. A seção 5 apresenta as conclusões.

Capítulo 2

Ambientes Virtuais Distribuídos

VÁRIOS grupos de pesquisa vêm propondo mecanismos para amenizar as restrições impostas pelo sistema de comunicação aos ambientes virtuais distribuídos. Neste capítulo são apresentados os principais mecanismos propostos por estes grupos. Além disto, são descritos alguns sistemas distribuídos de realidade virtual encontrados na literatura.

2.1 Mecanismos de Otimização

A manutenção de consistência do mundo virtual é uma tarefa muito custosa para o sistema de comunicação. O grande volume de dados trocados na rede pode inviabilizar a utilização do ambiente virtual quando número de usuários cresce. A fim de possibilitar um maior número de usuários, foram propostas os mecanismos de *dead-reckoning*, representação de multidões e filtragem de dados. Estes mecanismos visam a otimização da utilização dos recursos do sistema de comunicação através da redução do número de mensagens trocadas na rede.

2.1.1 *Dead-Reckoning*

O *dead-reckoning* é um mecanismo utilizado para a predição da posição e do movimento dos participantes entre a recepção de mensagens de atualização de estado. Seu conceito foi introduzido no sistema SimNet (ver seção 2.3.1), desenvolvido no início dos anos 80 pelas forças armadas americanas. O termo é originado da navegação marítima e sugere que a posição atual de um objeto pode ser extrapolada de sua posição e velocidade anteriores.

Durante a simulação, um simulador calcula sua próxima posição e orientação baseadas em um algoritmo detalhado que leva em consideração diversos fatores, entre eles a posição e a velocidade anteriores e os comandos do usuário. Ele calcula sua posição novamente, levando em consideração um número limitado de entradas. Se a diferença entre o primeiro cálculo detalhado e segundo mais limitado exceder um certo limite, então o simulador envia mensagens contendo sua posição atualizada baseada no cálculo detalhado. Se a diferença entre os dois cálculos for menor que o limite, o simulador não gera uma nova mensagem na rede. Desta forma, os participantes enviam mensagens somente quando necessário para manter a acurácia da simulação. O simulador extrapola a posição de todos os usuários remotos na simulação, usando o mesmo conjunto limitado de entradas. Como todos os simuladores seguem este procedimento, o número de mensagens trocadas pela rede é reduzido drasticamente [17].

O *dead-reckoning* foi primeiramente utilizado para ambientes de simulações militares, onde o número de objetos é restrito e todos os simuladores possuem um banco de dados completo de objetos. Nesta situação, as informações que precisam ser enviadas para a atualização do objeto são um identificador do objeto, sua posição, sua velocidade e sua orientação. Portanto, além da freqüência de mensagens ser reduzida pela característica de predição do *dead-reckoning*, o volume de informação enviada por mensagem também diminui.

A idéia do *dead-reckoning* pode ser estendida para representar objetos mais complexos, que contêm articulações, como seres humanos. Ao entrar em um ambiente virtual, um participante deve especificar o algoritmo de *dead-reckoning* que ele vai utilizar. Além do vetor velocidade do objeto, pode-se transmitir uma informação de comportamento, no caso de seres humanos [18], andar, caminhar ou dançar. Estas informações são suficientes para que um objeto remoto possa ser simulado. No caso das estações participantes não possuirem um banco de dados com todos os objetos do mundo, o objeto seria enviado inicialmente e depois apenas as informações de movimento precisariam ser enviadas periodicamente.

Existem duas abordagens para a escolha do algoritmo de *dead-reckoning* [15]. Uma abordagem, a adotada em sistemas militares, que extrapola a posição e a orientação usando mensagens contendo a localização, a velocidade e aceleração de um objeto. A outra utiliza o histórico de posições do objeto para extrapolar sua posição atual [19].

O *dead-reckoning* provoca um grau de inconsistência no estado do ambiente. O estado real de cada objeto é apenas aproximado pelas estações remotas. Este grau de inconsistência depende da escolha do algoritmo de *dead-reckoning* e do limite para a diferença entre os cálculos. Quando este limite decresce, o número de mensagens trocadas para um mesmo número de participantes aumenta. Quando o limite aumenta, o salto na posição de um participante devido a recepção de uma mensagem de atualização é maior.

2.1.2 Representação de Multidões

Aplicações como espetáculos, concertos, palestras e conferências têm o potencial de envolver um grande número de usuários. Nestas aplicações, no entanto, alguns participantes estão em destaque – músicos, apresentadores, palestrantes – enquanto os outros – pertencentes à platéia – participam em bloco. Os participantes do bloco percebem o comportamento dos participantes externos ao bloco, porém, estes últimos não conseguem identificar comportamentos individuais no bloco. O comportamento dos participantes em bloco é percebido de forma agregada.

Com base neste tipo de aplicação, Benford *et al.* [20] definiram o conceito de multidão, para representar os participantes em bloco. Uma multidão é uma abstração de um grupo de objetos que permite que estes objetos sejam tratados como um todo em algumas circunstâncias (para interação à distância) mas como indivíduos em outras (para interação entre os membros).

Um objeto multidão reflete a percepção agregada de um conjunto de objetos. As formas de fazer esta agregação dependem da aplicação e da mídia considerada. As multidões devem ter um efeito assimétrico. Vistos do lado de fora, indivíduos na multidão estão escondidos e são substituídos por uma visão agregada da multidão como um todo. Dentro da multidão, indivíduos podem interagir normalmente uns com os outros. Normalmente, eles também vão ter conhecimento dos objetos fora da multidão individualmente.

A representação de multidões é uma noção importante para o desenvolvimento de ambientes virtuais escaláveis. Como um grupo de objetos é representado por um único objeto multidão, as mensagens de atualização que seriam enviadas por cada um dos objetos do grupo vão ser substituídas por mensagens referentes a apenas um objeto. Isto diminui consideravelmente o tráfego na rede e possibilita um número bem maior de participantes.

2.1.3 Filtragem de Dados

A filtragem de dados é um mecanismo para reduzir o volume de informações que trafegam no sistema de comunicação. Este mecanismo explora o fato de um participante não interagir com todos os objetos de um mundo virtual simultaneamente. Neste mecanismo os participantes do mundo virtual são agrupados definindo escopos de interação, de forma que as interações entre participantes fiquem confinadas nestes escopos. O escopo de interação pode ser visto como um filtro para as informações a serem recebidas.

Estes escopos podem ser definidos por afinidades espaciais, temporais ou funcionais [21]. A filtragem espacial agrupa participantes que compartilham a mesma região do mundo virtual. A filtragem temporal agrupa objetos que necessitam de taxas de atualização similares. A filtragem funcional agrupa objetos de uma mesma classe funcional, por exemplo, em uma simulação militar, uma classe funcional seria o grupo de controle aéreo.

O tipo mais comum de filtragem é a espacial. Para este tipo de filtragem deve ser considerado se o espaço é aberto, por exemplo cidades, parques, praias, ou confinado, edifícios, escritórios [22]. No caso de espaços confinados pode ser utilizada a filtragem por oclusão, ou seja, se existir algum bloqueio entre dois objetos um não vai receber informações sobre o outro. Um exemplo seria um apartamento, onde os avatares que estão no quarto não podem interagir com os que estão na sala. No caso de espaços abertos pode ser usada tanto a filtragem por oclusão, quanto a filtragem por distância – que verifica a distância entre os objetos para saber se eles podem interagir e com que nível de detalhe.

A filtragem de dados limita a troca de mensagens a um escopo de interação, reduzindo o número total de mensagens trocadas no sistema de comunicação. A implementação deste mecanismo pode ser feita através da comunicação multidestinatária, mapeando-se os escopos de interação em grupos multipontos.

A participação nos grupos não é fixa, pois os participantes podem se movimentar pelo mundo, trocando, assim, de um escopo para outro. Porém, os usuários percebem o mundo como uma unidade, um ambiente contínuo. Eles não têm conhecimento do particionamento do mundo. Para isto, as mudanças de grupo devem ser realizadas suavemente, de forma que não haja interrupção na percepção do ambiente. Esta característica exige do sistema distribuído de realidade virtual e, conseqüentemente, do protocolo multidestinatário a possibilidade de trocas de grupo dinâmicas, com atraso mínimo.

2.2 Utilização de Protocolos Multidestinatários

Os protocolos multidestinatários permitem a transmissão de um pacote de dados para um conjunto de destinos em uma única operação. O termo multidestinatário considera tanto a comunicação ponto-a-multiponto, uma fonte e vários destinos, quanto multiponto-a-multiponto, várias fontes e vários destinos. Os protocolos multidestinatários podem ser utilizados em mundos virtuais com dois objetivos principais: restringir o número de participantes em um mesmo escopo de recepção de mensagens, fazendo um mapeamento entre os escopos de interação e os grupos multipontos, e economizar banda passante.

Um grande benefício da comunicação multiponto é a diminuição do custo de envio da mensagem. Alguns sistemas podem suportar multiponto em *hardware* de uma forma paralela. Nas redes em que os roteadores suportam multiponto, o custo para o transmissor de enviar um pacote multiponto é praticamente o mesmo de enviar um pacote ponto-a-ponto. A responsabilidade de difundir os pacotes é assumida pela rede que pode utilizar o paralelismo inerente à topologia da rede para buscar maior eficiência na comunicação multidestinatária. Assim, o número de estações receptoras pode aumentar sem sobrecarregar o transmissor.

Outro benefício da comunicação multidestinatária é a melhor utilização dos recursos da rede. Em redes que não utilizam multiponto o aumento na carga da rede é proporcional ao número de destinos. Com a transmissão multidestinatária, um determinado pacote só estará presente em um segmento de rede uma vez.

As comunicações multidestinatárias na Internet foram possibilitadas com a extensão do IP para suportar multiponto, surgindo o protocolo IP multiponto [23]. O IP multiponto possibilita o envio de um datagrama IP para um grupo de estações, um conjunto de uma ou mais estações identificadas por um único endereço destino IP. A rede virtual dentro da Internet que utiliza o IP multiponto é conhecida como MBone (*Multicast Backbone*) [24].

O MBone é chamado de rede virtual porque compartilha o mesmo meio físico que a Internet. A propagação de mensagens com endereçamento de grupo entre sub-redes é feita por roteadores que suportam comunicação multidestinatária, chamados roteadores multipontos. Quando uma sub-rede com suporte a roteamento de grupo está isolada das outras por roteadores convencionais, há necessidade de utilização de “tunelamento”. Desta forma, os pacotes IP multipontos são encapsulados dentro de pacotes IP convencionais.

Não existe nenhuma restrição em relação à localização ou ao número de membros de um grupo. Uma estação pode participar de um ou mais grupos ao mesmo tempo e não precisa ser membro de um grupo para enviar datagramas para ele. A participação dos membros em um grupo de estações é dinâmica, ou seja, as estações podem aderir ou se desligar de um grupo a qualquer momento.

O dinamismo da participação dos membros em grupos IP multipontos pode ser utilizado em ambientes virtuais em que os participantes estejam subdivididos em grupos multipontos. Porém, existe um atraso associado a troca de grupos no IP Multiponto que pode causar algumas inconsistências quando um participante muda de um escopo para outro do ambiente. Este atraso corresponde ao tempo necessário para que os roteadores retirem o participante de um grupo e coloquem em outro grupo.

2.2.1 Abordagens para Alocação de Grupos Multipontos

O mapeamento dos escopos de interação em grupos multipontos está associado à aplicação. O escopo ideal para a filtragem deveria ignorar apenas os objetos que não poderiam ser percebidos ou influenciados em nenhuma situação e enxergá-los com qualidade reduzida somente quando esta redução fosse escondida por limitações dos dispositivos ou da própria percepção do usuário. No entanto, este escopo ideal não seria suficientemente reduzido e, portanto, os grupos devem ser alocados buscando um compromisso entre o tamanho do escopo no mundo virtual e o número de grupos.

Para filtragens funcionais e temporais os grupos são praticamente estáticos e não há muitas opções para o mapeamento dos grupos de filtragem nos grupos multipontos. Porém, para filtragens espaciais existe um grande dinamismo nos grupos, devido a possibilidade de movimentação dos objetos pelo mundo. Por isso, neste trabalho apenas a filtragem espacial é considerada. É importante ressaltar que a alocação dos participantes nos grupos é determinada pela proximidade destes participantes no mundo virtual, não há nenhuma relação entre a posição real destes participantes e o seu grupo. Por exemplo, dois participantes em uma mesma sala no mundo virtual vão estar interagindo e, portanto, pertencer ao mesmo grupo multiponto, independentemente da sua localização no mundo real. Os usuários podem pertencer a mais de um grupo ao mesmo tempo, dependendo da subdivisão do mundo virtual.

Greenhalgh [25] propõe várias abordagens para realizar a alocação dos objetos do mundo virtual em grupos multiponto.

Um grupo por fonte

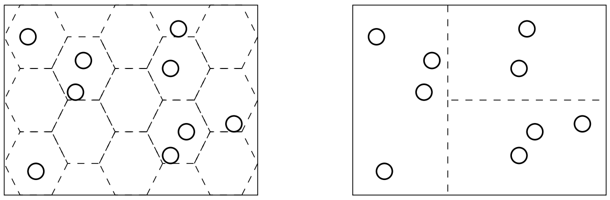
Esta abordagem possibilita que o escopo efetivo e o ideal coincidam. Assim, um usuário não entra em contato com fontes em que ele não esteja interessado só porque elas compartilham um mesmo grupo multiponto que fontes em que ele está interessado. Como resultado, o número de fontes é o mínimo possível para o escopo ideal em questão. As desvantagens desta abordagem são ter que considerar todas as fontes e observadores individualmente ao verificar interseções, além da necessidade de um grupo multiponto por fonte. Isto não seria escalável para sistemas grandes. O tratamento de grupos multiponto é mais custoso nos roteadores porque não existe uma organização hierárquica dos grupos multipontos atrelada às sub-redes como existe para endereços convencionais.

Um grupo por mundo

Esta alternativa elimina a filtragem dentro dos limites de um mundo, ou seja, todas as fontes enviam suas mensagens para um mesmo grupo multiponto e todos os observadores recebem seus pacotes através deste grupo. A adesão e o desligamento do grupo ocorrem apenas quando as fontes e os observadores entram e saem do mundo. É necessário que os observadores possam tratar o tráfego gerado em excesso. Além disto, esta abordagem é melhor aplicada quando o tamanho do mundo é limitado.

Grupos baseados em localização

Nesta abordagem o mundo é subdividido em áreas fixas, conforme ilustrado na figura 2.1(a). As fontes pertencem a um único grupo multiponto baseado em sua localização e trocam de grupo conforme atravessam seus limites. Os observadores monitoram cada grupo multiponto que se sobrepõe a seu escopo, o que pode mudar com a movimentação do observador. Desta forma, o escopo efetivo de um observador é estendido e ele é mantido informado de um número de fontes em que ele não está interessado. Tornando o tamanho do espaço que cada grupo vai representar maior possibilita a utilização de menos grupos multipontos, mas o escopo efetivo cresce para todos os usuários. Para usar esta abordagem deve ser possível pré-determinar um tamanho da área a ser utilizado de acordo com a aplicação em questão.



(a) (b)

Figura 2.1: Alocação com grupos compartilhados.

Grupos associados ao conteúdo

Esta abordagem é similar a de divisão em áreas fixas descrita anteriormente, com exceção de que os grupos são relacionados a organização espacial (a estrutura) do mundo virtual. Por exemplo, na figura 2.1(b) é representada a planta de um apartamento. A divisão dos objetos nos grupos multiponto segue a disposição dos cômodos, refletindo a tendência de que as interações dentro de um espaço fechado envolvam o conteúdo deste espaço. A descrição do mundo deve prover algumas diretivas explícitas para o sistema de comunicação sobre a possibilidade de uma região ser utilizada como um grupo. Já a abordagem baseada em localização pode ser feita em um espaço arbitrário. Além disto, esta abordagem não pode ser utilizada quando não existe um conteúdo estruturado para o espaço.

Grupos dinâmicos

As duas abordagens de grupos compartilhados apresentadas anteriormente são estáticas e imutáveis. Esta abordagem final é dinâmica, respondendo à disposição de fontes e observadores no espaço [26]. A figura 2.2 mostra um exemplo desta abordagem. O mundo começa com um único grupo multiponto (Grupo 1). Conforme a população aumenta este grupo vai sendo dividido em múltiplos grupos (Grupo 1 e Grupo 2), cada um cobrindo uma parte do espaço. Esta abordagem pode ser considerada um complemento para a abordagem de área fixa para mundos que são imprevisíveis e mutáveis. Com grupos dinâmicos, as descontinuidades são mais aparentes conforme novos grupos multiponto são adicionados ao escopo ou são feitas divisões e recombinações.

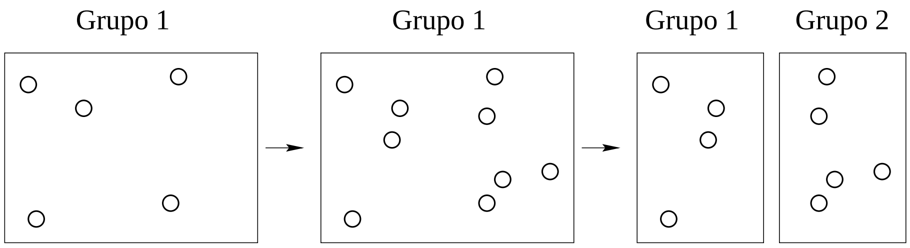


Figura 2.2: Alocação com grupos dinâmicos.

2.2.2 Multiponto com Confiabilidade

Os protocolos multidestinatários possibilitam que os mundos virtuais cresçam mantendo a carga nas fontes e otimizando a utilização dos recursos da rede de comunicação. Porém, estes protocolos não são escaláveis quando há necessidade de confiabilidade. Abordagens viáveis para oferecer multiponto confiável na Internet estão começando a surgir [27]. O problema do multiponto confiável em ambientes virtuais é relativo ao tamanho do mundo e seu crescimento, que torna o tratamento dos reconhecimentos (ACKs) um desafio conhecido como implosão de ACKs.

As duas abordagens mais populares para protocolos multidestinatários confiáveis atualmente são chamadas orientada ao transmissor e orientada ao receptor. Na abordagem orientada ao transmissor, este mantém o estado de todos os receptores para quem ele deve enviar informação e de quem ele deve receber ACKs. A cada transmissão ou retransmissão da fonte é feita uma transmissão multiponto para todos os receptores, e para cada pacote recebido corretamente o receptor envia um ACK para o transmissor. Assim, quando o número de receptores cresce muito, para cada pacote enviado, o transmissor precisa processar um grande volume de ACKS. Este fenômeno é conhecido como implosão de ACKs.

Ao contrário, na abordagem orientada ao receptor, cada receptor informa à fonte que alguma informação está errada ou faltando. O transmissor faz a difusão multiponto de todos os pacotes, dando prioridade para retransmissões e o receptor envia um reconhecimento negativo (NACK) quando detecta um erro ou um pacote perdido. Neste caso, pode ocorrer de um pacote ser perdido e vários receptores detectarem esta perda. Todos estes receptores vão, então, enviar um NACK, causando uma implosão de NACKs. Para evitar este problema, são usados mecanismos de supressão de NACKs.

Pingali *et al.* [28] apresentam uma análise que fornece uma demonstração quantitativa da superioridade da abordagem orientada ao receptor sobre a abordagem orientada ao transmissor. Os protocolos orientados ao transmissor possuem desempenho inferior em relação à escalabilidade alcançada quando comparados aos protocolos orientados ao receptor, especialmente quando neste último são utilizados esquemas de supressão de NACKs para evitar a sobrecarga da fonte com pedidos de retransmissão [27]. Portanto, para a construção de um ambiente virtual escalável e confiável deve-se utilizar protocolos multidestinatários orientados ao receptor e com supressão de NACKs.

2.3 Sistemas Existentes

Esta seção discute alguns sistemas distribuídos de realidade virtual. A descrição destes sistemas abrange as suas principais características, enfocando os mecanismos adotados para otimizar a utilização dos recursos do sistema de comunicação.

2.3.1 SimNet e DIS

O primeiro esforço em ambientes virtuais distribuídos foi o SimNet (*Simulator Networking*) [29], um sistema desenvolvido na década de 80 para o departamento de defesa americano com o objetivo de promover o treinamento militar utilizando uma rede de simuladores. O SimNet era baseado em três componentes: uma arquitetura objeto-evento, a noção de nós de simulação autônomos e o mecanismo de *dead-reckoning*.

A arquitetura objeto-evento modela o mundo virtual como uma coleção de objetos. As interações entre estes objetos são realizadas através de um conjunto de eventos, que são mensagens trocadas pela rede indicando uma alteração no estado do objeto. Os nós de simulação são autônomos porque são responsáveis pelo envio de mensagens periódicas (*heartbeats*) atualizando o estado dos objetos sob seu controle. Os receptores, por sua vez, devem receber estas mensagens e atualizar o estado dos objetos. As PDUs (*Protocol Data Units*) definidas para o SimNet são específicas para sistemas militares. Por exemplo, a PDU *fire* representa o disparo de uma arma, enquanto a PDU *impact* comunica que a arma acertou um objeto. O SimNet, no entanto, não define um conjunto de PDUs que supram os requisitos de qualquer tipo de simulação militar. Além disso, o formato das PDUs e a arquitetura não estão suficientemente documentadas.

Por isto, após o desenvolvimento do SimNet sentiu-se a necessidade de padronização do protocolo utilizado por ele, possibilitando a interoperabilidade entre qualquer tipo de participante e máquina. Esta preocupação deu origem ao desenvolvimento do protocolo para Simulações Distribuídas Interativas (*Distributed Interactive Simulation* – DIS), padrão IEEE 1278 [30]. A arquitetura objeto-evento foi projetada para suprir um maior número de requisitos de simulações militares que o SimNet.

O protocolo definido para o DIS especifica 27 PDUs, dentre as quais apenas 4 para interação entre nós no ambiente virtual: *Entity State* – enviada para a atualização de estado –, *Fire*, *Detonation* e *Collision*. A maioria dos sistemas compatíveis com o padrão DIS implementa apenas estas 4 PDUs. No DIS, a troca de pacotes é realizada através do protocolo UDP (*User Datagram Protocol*), que não é confiável. As perdas de pacotes são recuperadas na recepção do próximo pacote, para isto, mesmo objetos que não existem mais no ambiente devem enviar PDUs *Entity State*. A especificação do DIS estabelece o atraso de 100 ms para simulações fortemente acopladas e de 300 ms para simulações fracamente acopladas. As simulações implementadas com o DIS – realizadas na rede do departamento de defesa americano para simulações, *Defense Simulation Internet* (DSI) – suportam até 300 participantes simultâneos.

Porém, existem alguns problemas associados à escalabilidade do protocolo DIS [21, 15] como a necessidade de difusão periódica de atualizações de estado, mesmo para objetos estáticos, e a replicação dos modelos e banco de dados do mundo em cada simulador. Com o objetivo de tornar escaláveis os sistemas distribuídos de realidade virtual, estão surgindo novas propostas baseadas no aprendizado com o SimNet e o DIS.

2.3.2 RTI

Atualmente, a especificação do DIS está parada e o departamento de defesa americano redirecionou seus esforços para o desenvolvimento de um modelo cliente/servidor distribuído de barramento de objetos, a *High-Level Architecture/Runtime Infrastructure* (HLA/RTI). O HLA/RTI [31, 32] é um sistema que oferece serviços necessários para simulações distribuídas. O objetivo da HLA é o suporte adequado a todos os tipos de simulação do departamento de defesa americano, aumentando o grau de interoperabilidade e reutilização destas simulações. O serviço de distribuição de dados da RTI visa uma maior eficiência e flexibilidade de forma a suportar uma maior variedade de aplicações.

Os serviços da infra-estrutura RTI são descritos pela especificação de interface da arquitetura HLA. Estes serviços incluem o gerenciamento do tempo, das federações (conjunto de simulações), dos objetos, do acesso aos objetos e da declaração de dados. A distribuição e o controle dos objetos do RTI é feita através de CORBA

(*Common Object Request Broker Architecture*). Para modificar o estado de um objeto, basta alterar o atributo necessário. Apenas este atributo vai ser enviado para as outras simulações. A infra-estrutura separa os participantes em regiões, definindo as regiões em que eles têm interesse em receber ou enviar atualizações. Através das interseções nestas regiões, a RTI determina quais participantes estão interagindo, devendo trocar mensagens.

2.3.3 NPSNET-IV

O NPSNET-IV [33, 34, 21], desenvolvido pela *Naval Postgraduate School*, é um sistema de simulação para treinamento militar. As versões anteriores do NPSNET utilizavam redes locais e o protocolo SimNet para interoperabilidade. O principal objetivo desta última versão é obter um ambiente virtual mais escalável, permitindo o treinamento de muitos (mais de 100.000) usuários simultâneos. Atualmente, este sistema suporta centenas de participantes utilizando a Internet.

A fim de obter um sistema mais escalável, foram adotados o protocolo DIS, para comunicação entre simuladores independentes, e o protocolo IP multiponto, para suportar simulações distribuídas escaláveis em redes de longa distância. Assim como o DIS, o NPSNET-IV utiliza o mecanismo de *dead-reckoning* para reduzir o número de mensagens trocadas.

O NPSNET-IV utiliza um gerente de área de interesse [21], dividindo o ambiente virtual em um conjunto de partições menores (classes) para reduzir a carga computacional nas estações, minimizar a troca de informações na rede e restringir os problemas de confiabilidade. Os objetos do mundo virtual são divididos em classes espaciais, temporais e funcionais. Em relação às classes espaciais, o ambiente virtual é particionado em áreas hexagonais de tamanho fixo, chamadas células. Estas classes são associadas a grupos IP multipontos. O modelo de comunicação suporta tipos diferentes de interações [34], para garantir um certo grau de confiabilidade sem prejudicar as interações em tempo real.

O grupo que criou o NPSNET propôs, recentemente, o protocolo VRTP (Virtual

Reality Transfer Protocol) [35] com o objetivo de viabilizar ambientes virtuais na *Web*. A idéia é fazer uma extensão para o protocolo HTTP (*hypertext transfer protocol*), oferecendo um suporte de rede para a linguagem VRML. O VRTP deve oferecer suporte aos requisitos dos ambientes virtuais escaláveis, permitindo desde comunicação cliente/servidor até comunicação direta entre dois participantes.

2.3.4 MASSIVE

O sistema MASSIVE (*Model, Architecture and System for Spatial Interaction in Virtual Environments*) [36, 37] é um sistema distribuído de realidade virtual que suporta atividades colaborativas entre usuários com equipamentos heterogêneos. Ele introduz um modelo espacial de interação com diferentes níveis de percepção entre os participantes.

Neste modelo espacial, a *aura* define o volume do espaço dentro do qual a interação é possível. A interação entre dois objetos só torna-se possível quando as suas *auras* colidem. Portanto, os objetos só trocam notificações quando há uma colisão de *auras*, ou seja, é feita uma filtragem dos dados de forma que apenas objetos que estão em um mesmo escopo espacial troquem informações.

Outro componente do modelo espacial, o conhecimento, trata do controle de interação ou comunicação entre dois objetos uma vez que a interação foi permitida devido a uma colisão de *auras*. O conhecimento de um objeto sobre o outro quantifica a importância deste último objeto. O nível de conhecimento é baseado na distância relativa entre os participantes e mediados por um mecanismo de negociação, feito por um gerente de colisão. Quando o gerente de colisões identifica uma colisão os participantes são notificados e estabelecem uma conexão ponto-a-ponto. A responsabilidade de detecção de colisão é dividida entre vários gerentes de colisão, buscando melhorar a escalabilidade do sistema.

Uma nova versão do MASSIVE foi proposta [20] para gerenciamento de conhecimento e comunicação entre participantes de uma multidão. A estrutura para tratar multidões é baseada no modelo espacial de interação e numa extensão para este modelo chamada objeto mediador. Colisões de *auras* levam a estabelecimentos de conexão. A qualidade da informação que é transmitida depende do nível de conhecimento que um observador tem do observado. Um objeto mediador é um objeto independente que afeta o conhecimento entre dois outros objetos.

As multidões são uma classe específica de objetos mediadores que suportam grupos de pessoas potencialmente grandes. As multidões devem ter efeito assimétrico no conhecimento. Vistos do lado de fora, indivíduos na multidão estão escondidos e são substituídos por uma visão agregada da multidão como um todo. Dentro da multidão, indivíduos podem interagir normalmente uns com os outros. Tipicamente, eles também vão ter conhecimento dos objetos fora da multidão individualmente.

2.3.5 DIVE

O DIVE (*Distributed Interactive Virtual Environment*) [38], desenvolvido no *Swedish Institute of Computer Science*, tem como principal enfoque a interação entre os usuários. Por isso, as aplicações visadas são as que exigem colaboração entre os usuários. Basicamente, o DIVE representa uma sala de conferência para interação compartilhada à distância.

Este sistema busca a escalabilidade através da utilização de um protocolo multidestinatário confiável com reconhecimentos negativos, a replicação parcial da base de dados e um módulo simples de *dead-reckoning*. O particionamento dos participantes em grupos é feito considerando um grupo por mundo, não havendo filtragem de dados entre os participantes de um mundo.

A base de dados do DIVE é dinâmica. O sistema tem a capacidade de adicionar novos objetos e modificar a base de dados de maneira confiável. O controle de concorrência dos acessos aos objetos do mundo virtual é feito através de um mecanismo de bloqueio (*locking*) do objeto acessado. Alguns testes realizados com o DIVE, mostram que ele funciona em redes de longa distância para 20 participantes com atrasos menores que 200 ms.

2.3.6 BrickNet

O BrickNet [39] é um trabalho do *Institute of Systems Science* da Universidade Nacional de Singapura. O BrickNet é uma ferramenta orientada-a-objetos para ambientes virtuais que oferece suporte gráfico, comportamental e um modelo de rede para mundos virtuais. Seu objetivo é o auxílio ao desenvolvimento de ambientes colaborativos. Esta ferramenta tem sido utilizada para a construção de jogos distribuídos e trabalho cooperativo.

Neste sistema, o mundo virtual é particionado em vários ambientes virtuais clientes. Os pedidos de objetos são intermediados por um servidor. Interações com objetos distantes são realizadas por um barramento de pedidos de objetos (*object request broker*) em um servidor que saiba a qual cliente pertence o objeto. O estado dos objetos é sincronizado entre os clientes através de mensagens de sincronização intermediadas pelo servidor. Neste modelo de comunicação, os servidores podem ser um gargalo quando o número de clientes aumenta. O maior cenário utilizando o BrickNet permite sua utilização com poucas dezenas de participantes.

2.3.7 SPLINE

O SPLINE (*Scalable Platform for Large Interactive Networked Environment*) [40], foi desenvolvido no *Mitsubishi Electric Research Labs* (MERL). O primeiro mundo virtual construído utilizando esta plataforma foi um parque (*Diamond Park*). Vários usuários podem interagir com o parque pedalando em bicicletas controladas por computador. Quando próximos uns dos outros, os usuários são capazes de conversar.

No SPLINE, o mundo virtual é modelado por uma base de dados que contém informações sobre este mundo. Este modelo está parcialmente distribuído entre os processos SPLINEs. As aplicações interagem umas com as outras através da modificação desta base de dados e da observação das modificações realizadas pelos outros usuários.

O mundo é divido em *locales*, áreas que podem ser processadas separadamente.

Cada *locale* está associado a um grupo multiponto, possui um sistema de coordenadas próprio e seu tamanho, formato e orientação são escolhidos arbitrariamente. Cada participante possui uma cópia parcial da base de dados, de acordo com o seu *locale*. Servidores de *locale* são responsáveis por manter um registro do estado de todos os objetos em um determinado *locale*. Desta forma, um processo SPLINE que se interesse por um *locale* pode receber a informação atualizada sem contactar cada processo pertencente ao *locale*.

2.3.8 Grupo de Realidade Virtual

O Grupo de Realidade Virtual (GRV) da Universidade Federal de São Carlos especifica um sistema de realidade virtual [41] dividido em sub-áreas estruturadas como um conjunto de células hexagonais. Cada sub-área é coordenada por um servidor de ambiente responsável pelo armazenamento e gerenciamento de seu conjunto de dados, além da animação e da sincronização dos objetos que pertencem a esta sub-área. Como o sistema pode compreender várias sub-áreas, o ambiente como um todo pode ser distribuído em vários servidores de ambiente.

Outro trabalho do GRV aborda a utilização de CORBA como plataforma para suporte às aplicações distribuídas de realidade virtual [42]. A plataforma CORBA, através de seus serviços e facilidades, possibilita que os objetos gráficos sejam armazenados, gerenciados e utilizados. Assim, estes objetos podem ser compartilhados pelos usuários do ambiente virtual.

Os sistemas apresentados utilizam o mecanismo de filtragem de dados associado a protocolos multidestinatários com o objetivo de alcançar ambientes virtuais distribuídos mais escaláveis. Outro mecanismo bastante utilizado é de *dead-reckoning*. Pode-se observar também que estes sistemas buscam restringir o escopo onde é realizado o controle de consistência, distribuindo este controle por diversos servidores.

A documentação destes sistemas não é de fácil acesso e uma comparação entre todos eles seria um trabalho árduo. Como o objetivo deste trabalho é relacionado ao sistema de comunicação, é proposta uma plataforma genérica que incorpora as principais características destes sistemas. Assim, é possível mensurar o impacto dos ambientes virtuais distribuídos no sistema de comunicação através da análise desta plataforma. A plataforma proposta é apresentada no próximo capítulo.

Capítulo 3

Plataforma Proposta

A PLATAFORMA proposta [43] utiliza os serviços dos protocolos de transporte disponíveis na Internet, visando atender aos requisitos das aplicações de realidade virtual distribuída. O serviço oferecido pela plataforma é flexível, ou seja, a plataforma é configurada de acordo com a aplicação desejada. A aplicação é responsável por fornecer os cenários, as imagens para representação dos usuários (avatares) e algumas informações sobre a estrutura do ambiente, específicas para a aplicação. A plataforma baseia-se nestas informações para realizar a entrega dos dados de forma eficiente, garantindo a manutenção da consistência do mundo. Assim, para a aplicação a distribuição dos dados e o controle dos objetos do ambiente virtual são transparentes.

Além da manutenção da consistência e do controle de concorrência do acesso aos objetos do mundo, outro aspecto importante para as aplicações de realidade virtual distribuída é a possibilidade de interação direta entre os participantes através de múltiplas mídias, como o áudio e o vídeo. Esta interação aumenta a sensação de realismo no ambiente virtual. Para atender a todos estes requisitos, foi proposta uma arquitetura modular para a plataforma. Esta arquitetura está divida em funções para interação direta entre participantes (multimídia), controle de concorrência (interação com o mundo virtual) e funções do ambiente virtual (controle de consistência e envio de atualizações de estado dos participantes).

Este capítulo apresenta a plataforma proposta para aplicações de realidade virtual distribuída. A seção 3.1 discute algumas características da plataforma: quais são seus principais objetivos, como são alcançados, como são tratadas as restrições do sistema de comunicação. Na seção 3.2 é apresentada uma arquitetura para esta plataforma. O protocolo proposto para a comunicação entre as entidades da plataforma é especificado na seção 3.3.

3.1 Projeto da Plataforma

A plataforma foi desenvolvida com dois objetivos principais: a flexibilidade, podendo ser adotada por diferentes aplicações de realidade virtual distribuída, e a escalabilidade, permitindo que o número de participantes cresça sem prejudicar a interatividade entre eles. Para prover escalabilidade, a plataforma adota o mecanismo de filtragem de dados, dividindo os participantes em escopos de interação. Estes escopos são mapeados em grupos multipontos, que são o domínio para realização do controle de consistência.

A divisão do mundo virtual em escopos é definida pela aplicação de acordo com sua estrutura. Esta divisão pode ser feita por conteúdo, quando o ambiente tiver uma estrutura bem definida, ou por áreas de tamanho fixo, para ambientes abertos. Na filtragem por áreas de tamanho fixo cabe à aplicação definir as áreas e o tamanho de cada área. Este particionamento em escopos não varia no decorrer de uma sessão.

A consistência do ambiente é mantida por uma estação especial chamada líder. A estação líder é eleita entre uma das estações participantes comuns, acumulando as funções de líder. A eleição do líder é realizada como proposta por Macedonia *et al.* [21], ou seja, o líder é o participante mais antigo do escopo. São utilizadas estampas de tempo para determinar o tempo de permanência de uma estação num escopo.

Cada escopo de interação possui exatamente um líder. Quando um participante troca de um escopo de interação para outro, mudando conseqüentemente de grupos multipontos, diz-se que ele efetua um *handoff*. O líder é responsável pelo controle das entradas, saídas e *handoffs* no seu escopo de interação. Além disto, o líder controla o acesso aos objetos do mundo virtual.

A divisão dos participantes em grupos multipontos reduz a carga no sistema de comunicação, realizando a filtragem das mensagens de atualização trocadas. Conforme os participantes movimentam-se no mundo virtual, seu escopo de interação modifica-se e, conseqüentemente, seu grupo multiponto, havendo a necessidade de *handoffs* dinâmicos. Este dinamismo é importante para que não haja interrupções na percepção do ambiente virtual com a realização de um *handoff*. Por isto, o atraso para sinalização do *handoff* e para a troca efetiva de grupos deve ser mínimo.

O modelo de dados utilizado é distribuído, ou seja, os dados do mundo virtual estão particionados entre os participantes do ambiente. Cada participante possui uma cópia dos dados referentes a seu escopo de interação, havendo replicação dos dados de um escopo entre os seus participantes. Para que as réplicas dos dados em cada participante de um escopo permaneçam consistentes entre si, é necessário que as mensagens sinalizando alterações no escopo sejam transmitidas de forma confiável. Logo, seria interessante que o protocolo de transporte utilizado oferecesse um serviço multidestinatário confiável.

Porém, como apresentado na seção 2.2.2, os protocolos multidestinatários apresentam restrições à escalabilidade quando é necessária confiabilidade. Além disto, protocolos confiáveis introduzem atrasos para garantir a entrega dos pacotes sem erros e ordenados, não sendo adequados para a transmissão de dados que necessitem de entrega em tempo real [11].

Sendo assim, propõe-se a separação das operações do ambiente virtual em operações críticas, que precisam de confiabilidade, e operações não críticas, que precisam ser entregues em tempo real. Todas as comunicações com o líder para entrar no mundo, sair, trocar de grupos são consideradas críticas. Estas mensagens não podem ser perdidas porque causariam situações de inconsistência no ambiente, que não poderiam ser recuperadas. Já as operações de atualização de estado dos participantes e as interações multimídias entre participantes, que possuem como maior restrição o atraso, são trocadas sem confiabilidade. A perda de uma mensagem não crítica pode ser recuperada pela mensagem seguinte e pelo mecanismo de *dead-reckoning*. O único efeito indesejável desta perda é um salto no comportamento do participante.

A divisão das operações causa a distinção entre quatro tipos de comunicação utilizados na plataforma:

- mensagens não críticas multidestinatárias – devem ser recebidas por todos os participantes em um escopo com o menor atraso possível, são as mensagens de atualização de estado;
- mensagens não críticas ponto-a-ponto – utilizadas para troca de dados multimídias entre dois participantes, permitindo a interação direta entre eles;
- mensagens críticas ponto-a-ponto – são necessárias para permitir a troca de dados confiável entre dois participantes, a entrega do banco de dados com as informações do mundo virtual para um novo participante utiliza este tipo de mensagem;
- mensagens críticas multidestinatárias – devem ser enviadas com confiabilidade para um grupo de participantes, podem ser utilizadas, por exemplo, no envio de um avatar para um grupo.

Com a finalidade de atender a todos estes requisitos, é necessária a utilização de três protocolos de transporte diferentes em conjunto com um serviço de rede que permita a entrega de pacotes para grupos multipontos. O IP Multiponto [23] permite a adesão e o desligamento de estações a grupos a qualquer momento, ou seja, a participação em um grupo é dinâmica. Então, o IP Multiponto aliado ao UDP é adotado para a transmissão das mensagens não críticas multidestinatárias. Para as mensagens não críticas ponto-a-ponto pode ser utilizado o protocolos UDP/IP. Para as mensagens críticas multidestinatárias o protocolo de transporte utilizado é o SRM (*Scalable Reliable Multicast*) [44], um protocolo de transporte confiável com reconhecimento negativo que visa a escalabilidade. O TCP é usado para a troca de mensagens confiáveis entre dois participantes.

Resumindo, a escalabilidade é alcançada através da distribuição do controle de consistência, realizado pelos líderes, e da redução da troca de mensagens com confiabilidade, através da divisão das operações.

3.2 Arquitetura

A arquitetura proposta para a plataforma visa a uniformidade entre todas as estações que a executam, inclusive as estações líderes. Assim, qualquer participante pode ser eleito líder durante uma sessão. A única distinção entre os líderes e os participantes é o fato destes primeiros realizarem, além das funções comuns, funções de controle. A arquitetura é apresentada na figura 3.1. Pode-se observar que as funções referentes aos líderes estão em um módulo separado, sendo acionadas apenas quando o participante torna-se líder. As funções da estação líder são totalmente transparentes ao usuário que participa e interage com o mundo virtual como qualquer outro participante conectado a uma estação comum (não-líder). Além disto, é importante notar que para uma maior simplicidade e flexibilidade da plataforma, foram criadas interfaces com a aplicação e com sistema de comunicação, tornando a plataforma fracamente acoplada.

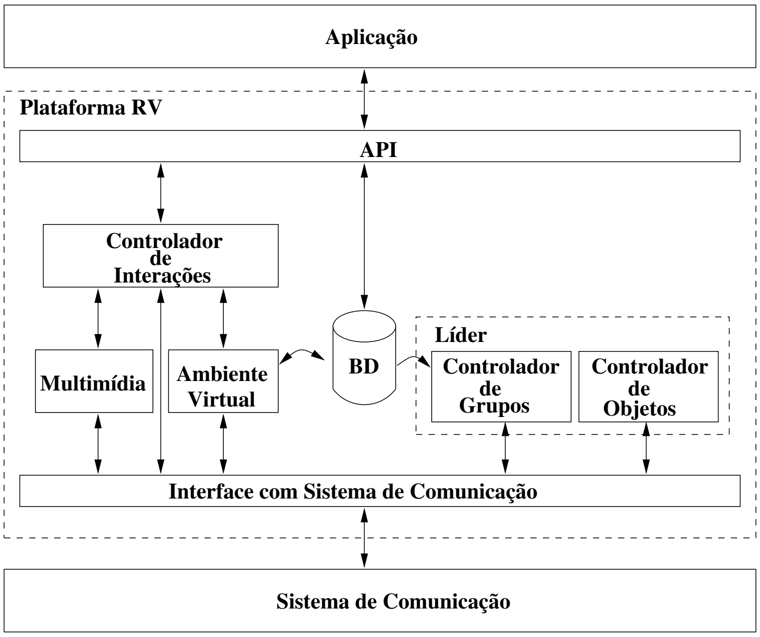


Figura 3.1: Arquitetura da plataforma.

Como a divisão do ambiente em escopos não varia no decorrer de uma sessão, tendo-se a posição inicial de um participante no ambiente virtual, tem-se o escopo a que ele pertence e, conseqüentemente, seu líder. Os dados do ambiente virtual estão parcialmente replicados no banco de dados (BD) de acordo com esta divisão. O BD contém, além da representação de cada objeto, sua posição e comportamento, as informações sobre o cenário do mundo virtual e sua divisão em escopos. Quando ocorre alguma modificação no usuário local, a aplicação atualiza o BD através da interface com a aplicação.

O módulo ambiente virtual é responsável por manter atualizadas as informações do mundo virtual. Ele monitora o BD para verificar as modificações que devem ser enviadas para os outros participantes e recebe mensagens de atualização dos outros participantes, fazendo as modificações no BD. Este módulo pode executar um algoritmo de *dead-reckoning*, visando reduzir a freqüência das mensagens de atualização. O algoritmo deve ser escolhido pela aplicação, considerando os tipos de avatares utilizados. O ideal seria que cada avatar tivesse seus movimentos aproximados por um algoritmo de predição diferente, adequado ao seu tipo de comportamento.

O módulo controlador de interações gerencia tanto as interações diretas entre participantes, quanto as interações entre participantes e objetos do mundo. Nas interações entre participantes, o controlador de interações é responsável por estabelecer a conexão entre os participantes interagindo. Esta conexão é usada pelo módulo multimídia, que promove a interação dos participantes através de diversas mídias. As interações entre participantes e objetos precisam ser realizadas mantendo a exclusão mútua no acesso aos objetos. Para isto, o controlador de interações precisa verificar a permissão de acesso aos objetos.

O líder, além de atuar como um participante comum através dos módulos já apresentados, utiliza mais dois módulos: o controlador de grupos e o controlador de objetos. O módulo controlador de grupos é responsável por gerenciar a adesão e o desligamento dos usuários no escopo de interação do líder. O módulo controlador de objetos contém as permissões de acesso para todos os objetos do escopo do líder, evitando, assim, conflitos entre participantes acessando os mesmos objetos.

Esta arquitetura é abrangente, refletindo os requisitos dos ambientes virtuais distribuídos. O enfoque deste trabalho, no entanto, são os problemas relacionados à escalabilidade destes ambientes. Por isto, as funções de interação multimídia e de controle dos objetos do mundo virtual não foram especificadas. O protocolo especificado na próxima seção engloba apenas as operações críticas e as mensagens de atualização de estado.

3.3 O Serviço e o Protocolo

Um dos objetivos da plataforma é a flexibilidade. Para alcançá-lo o protocolo definido para esta plataforma deve ser genérico, sem incorporar características específicas de nenhuma aplicação como, por exemplo, o protocolo do DIS que foi especificado para aplicações militares. Assim, as unidades de dados do protocolo (PDUs) e as primitivas de serviço especificadas podem ser utilizadas por qualquer tipo de aplicação de realidade virtual distribuída, inclusive para aplicações militares.

3.3.1 Primitivas de Serviço e PDUs

Neste trabalho são apresentadas apenas as PDUs e as primitivas necessárias para realizar o controle de consistência. Este controle engloba as funções determinantes para a escalabilidade do sistema – entrada e saída do mundo, atualizações de estado e *handoffs*. Por isto, as tabelas 3.1 e 3.2 mostram as primitivas e as PDUs especificadas para realizar o controle de consistência na plataforma. Também não são especificadas as PDUs para eleição de líder, assumindo-se que as estações líderes são conhecidas.

As primitivas de serviço apresentadas na tabela 3.1 podem ser dividas em três grupos: entrada no mundo virtual, saída do mundo virtual e atualização. Na fase de entrada no mundo virtual a aplicação do novo participante envia uma primitiva RV_Con.Req, pedindo para receber o banco de dados com a informação do ambiente virtual e recebe uma primitiva RV_Con.Conf avisando que o banco de dados está disponível e que seu avatar foi enviado aos outros participantes do es-

Tabela 3.1: Primitivas.

Primitiva	*Função*
RV_Con.Req	Pedido de entrada no mundo
RV_Con.Ind	Indicação de que um novo avatar aderiu ao grupo
RV_Con.Conf	Notificação de que os dados do ambiente estão disponíveis
RV_Dis.Req	Pedido para sair do mundo
RV_Dis.Ind	Aviso aos participantes da saída de um objeto
RV_Update.Req	Movimentação do participante deve ser atualizada no BD
RV_Update.Ind	Indicação de que algum objeto foi modificado no BD

copo. Os outros participantes percebem a entrada do novo avatar ao receber uma primitiva RV_Con.Ind. Durante a permanência de um usuário em uma sessão, seus movimentos vão gerar o envio de primitivas RV_Update.Req. Estas primitivas nem sempre vão gerar o recebimento pelos outros participantes do escopo de uma primitiva RV_Update.Ind. Isto ocorre porque os pedidos de atualização podem ser filtrados por um algoritmo de *dead-reckoning*. As primitivas para a saída do mundo virtual são RV_Dis.Req e RV_Dis.Ind, que indicam à plataforma que esta deve retirar o avatar do mundo virtual. É importante notar que não existem primitivas para realização de *handoffs* porque a divisão em grupos e, conseqüentemente, a troca de grupos é transparente para a aplicação durante uma sessão. A aplicação define a divisão em grupos na fase de configuração do mundo virtual.

As PDUs do protocolo, especificadas baseando-se nos protocolos DIS [30] e do NPSNET-IV [21], são apresentadas na tabela 3.2. As PDUs utilizadas para a entrada no mundo são JR, JI e JC. Para a saída do mundo é enviada uma PDU LR. Os *handoffs* são realizados através das PDUs de entrada e saída no mundo aliadas a PDU FNL, que indica a um participante que ele deve iniciar um *handoff*. As atualizações de estado são notificadas através do envio de PDUs ES.

Tabela 3.2: Unidades de dados do protocolo.

PDU	*Função*
JR	Pedido de adesão a um grupo
JI	Indicação de que um novo avatar aderiu ao grupo
JC	Confirmação de um pedido de adesão, envio do BD
LR	Desligamento do grupo
FNL	Indicação de que um participante deve iniciar um *handoff*
ES	Atualização de estado de um avatar

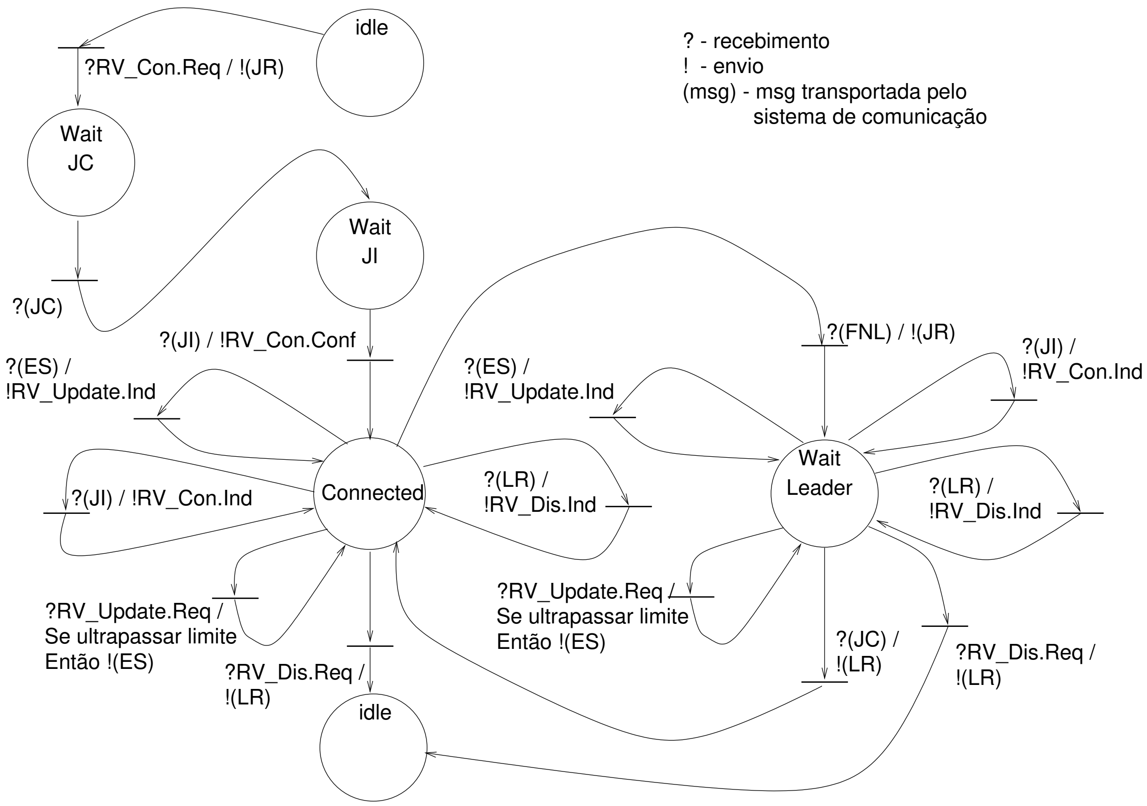
3.3.2 Funcionamento do Protocolo

O funcionamento do protocolo modelado através de Redes de Petri Predicado-Ação [45][1] é mostrado na figura 3.2. A figura 3.2(a) apresenta a Rede de Petri que modela o comportamento de um participante comum. A figura 3.2(b) mostra as funções extras desempenhadas por um participante líder.

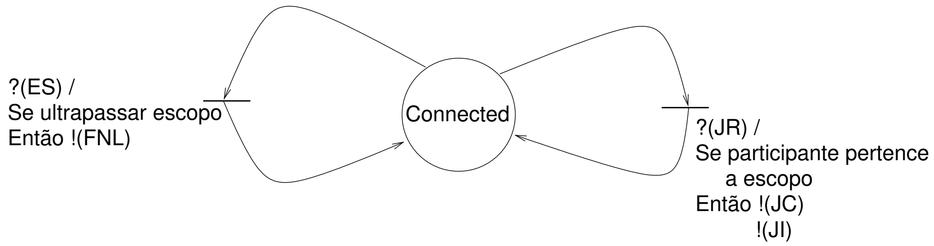
Todo participante do mundo virtual é membro de um grupo multiponto. Este grupo é determinado de acordo com a sua posição no mundo virtual. Os líderes participam de um grupo multiponto de controle, com endereço conhecido. Assim, os participantes enviam mensagens para este endereço de controle para descobrir qual líder é responsável por seu escopo.

Para começar uma sessão, i.e., para entrar no mundo virtual, a aplicação deve receber as informações do escopo em que está ingressando e deve enviar seu avatar para este escopo. Então, a aplicação na estação participante deve enviar uma primitiva RV_Con.Req para a plataforma que envia uma mensagem multiponto confiável com uma PDU JR para o endereço de controle pedindo para aderir ao grupo. A PDU JR contém a posição inicial e as informações do novo avatar. Ao receber uma PDU JR, os líderes verificam se a posição requisitada está dentro do seu escopo de interação. O líder responsável pela posição responde ao pedido enviando uma

[1] As Redes de Petri são compostas de lugares e transições, onde os lugares representam o estado do sistema e as transições as trocas de estado. Além disto, nas redes Predicado-Ação, cada transição está associada a uma expressão "predicado/ação".



(a) Participante comum.



(b) Líder.

Figura 3.2: Funcionamento do protocolo modelado com Redes de Petri.

PDU JC diretamente para o novo participante com o BD contendo o estado atual do escopo.

Após a entrega do BD, o líder envia uma mensagem multiponto confiável com a PDU JI para todos os participantes de seu escopo, contendo as informações do novo avatar. Ao receber uma PDU JI com o seu avatar, a plataforma no novo participante, que já recebeu o BD, sinaliza à aplicação que as informações do escopo já estão atualizadas e que seu avatar já foi inserido no ambiente através de uma primitiva RV_Con.Conf. Assim, este participante pode começar a interagir com o ambiente. Nos outros participantes do escopo, o recebimento pela plataforma de uma PDU JI gera o envio de uma primitiva RV_Con.Ind, informando a presença de um novo usuário.

Após a fase de adesão, os participantes atualizam seus estado através das primitivas RV_Update.Req e RV_Update.Ind que geram a troca de mensagens não críticas multidestinatárias com a PDU ES no seu escopo de interação. O líder, assim como todos os participantes de seu escopo, recebe as atualizações de estado de todos os outros membros de seu escopo. Quando o líder detecta que um participante está cruzando os limites de seu escopo, ele envia uma mensagem crítica ponto-a-ponto com a PDU FNL para este participante, sinalizando que um *handoff* deve ser iniciado. Assim, o participante envia uma mensagem multiponto com uma PDU JR e reinicia a fase de adesão. Após receber uma PDU JC de seu novo líder, o participante desliga-se de seu grupo antigo através do envio de uma mensagem crítica multiponto com a PDU LR para este grupo. É importante ressaltar que os *handoffs* são controlados pelos módulos da plataforma sem o conhecimento da aplicação.

Para terminar uma sessão, um participante envia uma primitiva RV_Dis.Req. Ao receber esta primitiva, a plataforma envia uma mensagem crítica multiponto com uma PDU LR para o escopo do participante, pedindo a remoção de seu avatar do ambiente. Os outros participantes do escopo são notificados através de uma primitiva RV_Dis.Ind.

O funcionamento da plataforma foi simulado visando avaliar o impacto dos ambientes virtuais distribuídos no sistema de comunicação. O próximo capítulo descreve as simulações realizadas e discute os resultados obtidos para diferentes requisitos das aplicações distribuídas de realidade virtual.

Capítulo 4

Desempenho da Plataforma

A FIM de avaliar o desempenho da plataforma foram realizadas simulações em uma topologia similar a estrutura encontrada na Internet. O comportamento da plataforma foi avaliado visando verificar o impacto de aplicações com requisitos diferentes. Estas simulações compreenderam as operações críticas – troca de grupos, entrada e saída do mundo – e as atualizações de estado. Na seção 4.1 é discutida a implementação da plataforma utilizando uma ferramenta de simulação. Esta seção também apresenta a topologia onde a plataforma foi simulada. O modelo de simulação definido para obter os resultados é apresentado na seção 4.2. Os parâmetros medidos nas simulações e os resultados obtidos são discutidos nas seções 4.3 e 4.4, respectivamente.

4.1 Infra-estrutura de Simulação

A ferramenta escolhida para simular a plataforma foi o *ns* (*Network Simulator*) [46, 47]. O *ns* é um simulador de eventos discretos desenvolvido para pesquisa em redes de comunicação. Ele oferece uma infra-estrutura para simulação de protocolos de rede com abstrações para nós e enlaces. Esta infra-estrutura também compreende alguns algoritmos de roteamento utilizados na Internet, incluindo roteamento multiponto. Além disto, o *ns* engloba vários modelos de protocolos usuais na Internet, como o TCP e o UDP, e outros protocolos mais novos que poderão ser usados na Internet, como o SRM. O *ns* permite a integração de novos módulos a sua estrutura. Estes módulos podem representar um protocolo, uma aplicação, uma fonte de tráfego. Para simular a plataforma foi necessária a criação de um novo módulo no *ns* que implemente o protocolo apresentado na seção 3.3. Mais detalhes sobre o *ns* podem ser obtidos no apêndice A. As simulações foram realizadas em um computador com processador Pentium-200Mhz com 128 MB de memória RAM e sistema operacional *linux*.

4.1.1 Implementação da Plataforma no *ns*

A plataforma foi implementada no *ns* como um agente, que foi denominado agenteRV. Um agente no *ns* gera e consome pacotes, sendo ideal para a simulação do protocolo da plataforma. O agenteRV utiliza os serviços oferecidos por outros agentes disponíveis no *ns* para o transporte dos dados. Este agente apenas simula o comportamento do protocolo da plataforma. Sendo assim, as PDUs do protocolo não carregam dados, que são representados apenas por seu tamanho. Os pacotes gerados pelo agenteRV carregam uma identificação da PDU, o endereço destino (neste caso um número que identifica o agente destino no *ns*), o endereço fonte e o tamanho dos dados. Os agentes que representam os protocolos de transporte no *ns* apenas simulam o envio de um pacote do tamanho determinado.

A fim de realizar as simulações foi necessária a escolha dos algoritmos de roteamento, tanto multiponto quanto ponto-a-ponto, e dos protocolos de transporte que serviriam ao agenteRV. A infra-estrutura oferecida pelo *ns* abrange modelos para os principais protocolos encontrados na Internet, o que facilitou a adoção de um ambiente mais próximo ao real.

O roteamento dos datagramas IP foi realizado através do algoritmo SPF (*Shortest Path First*) de Dijkstra [48]. Neste algoritmo as rotas são calculadas usando uma matriz de adjacência e custo dos enlaces para todos os enlaces da topologia. Nas simulações realizadas não foram simuladas mudanças na topologia, por isso, a estratégia adotada foi a estática, em que o algoritmo de computação de rotas é executado somente uma vez, antes do início da simulação.

O protocolo de roteamento multiponto do *ns* utilizado foi o modo denso estático. Este protocolo do *ns* é baseado no DVMRP (*Distance Vector Multicast Routing Protocol*) [49], o protocolo utilizado no MBone (*Multicast Backbone*) [48]. Porém, o modo denso estático não se adapta a mudanças na rede, diferentemente do DVMRP. É importante notar que a utilização do protocolo encontrado na Internet permite que as medidas de atraso obtidas na simulação já incluam os atrasos introduzidos pelo protocolo de roteamento para trocar um membro de grupos.

Os protocolos de transporte adotados foram o UDP (*User Datagram Protocol*) [50], o TCP (*Transmission Control Protocol*) [50] e o SRM (*Scalable Reliable Multicast*) [44]. O agente UDP no *ns* recebe dados com tamanho variável da aplicação e os segmenta caso necessário. A implementação do UDP no *ns* recebe da aplicação apenas o tamanho dos dados a serem enviados e simula a transmissão de um pacote deste tamanho. Assim, para simular o protocolo da plataforma foi necessário modificar a função do agente UDP no *ns* que envia os dados, de forma que o cabeçalho do protocolo da plataforma possa ser acrescentado ao pacote de dados a ser transmitido. O pacote no destino é entregue diretamente pela camada de rede para a plataforma, pois a implementação do UDP no *ns* não possui entidade destino.

O *ns* possui vários agentes que implementam variações do protocolo TCP, modelando diferentes tipos de controle de congestionamento. No entanto, apenas o agente FullTcp é bidirecional, ou seja, é capaz de receber e enviar dados. Assim, este agente aproxima-se do comportamento do TCP como utilizado na Internet e, por isso, foi escolhido para o transporte das mensagens críticas ponto-a-ponto. O FullTcp modela o comportamento do TCP Reno [50], que utiliza os mecanismo de retransmissão e recuperação rápidas. Na simulação da plataforma, o FullTcp foi configurado para operar com tamanho de segmento padrão igual a 536 octetos e janela de transmissão de 100 pacotes, que corresponde a uma janela no TCP de 53600 octetos. Foi utilizada a opção de enviar dados no pacote de abertura de conexão e para cada pacote recebido foi gerado um reconhecimento.

O protocolo SRM é um protocolo multidestinatário confiável que utiliza a abordagem orientada ao receptor com supressão de NACKs, como apresentado na seção 2.2.2. Através desta abordagem o SRM oferece um serviço de comunicação confiável sem prejudicar a escalabilidade. Neste protocolo os receptores podem aderir e desligar-se de um grupo dinamicamente, sem afetar a transmissão dos demais participantes deste grupo. O agenteRV utiliza os serviços do agente SRM do *ns* para a transmissão das mensagens críticas multidestinatárias.

Foram necessárias algumas modificações no agente SRM para adequá-lo aos requisitos da plataforma. No SRM, quando um novo usuário entra numa sessão, ele recebe todos os pacotes já enviados nesta sessão [44]. Porém, em um ambiente virtual distribuído, um usuário ao aderir a um grupo deseja receber apenas o estado atual do escopo associado a este grupo. A recepção de todas as mensagens já enviadas naquele grupo acarretaria em uma grande quantidade de mensagens desnecessárias que desperdiçariam os recursos do sistema de comunicação e o processamento para descarte dessas mensagens no novo usuário. Por isto, o módulo do SRM no *ns* foi alterado para não fazer a retransmissão das mensagens desde o início da sessão.

4.1.2 Geração das Topologias

A topologia para a simulação foi gerada através do gerador de topologias GT-ITM (*Georgia Tech Internetwork Topology Models*), visando um formato mais próximo do encontrado atualmente na Internet. O GT-ITM utiliza um método de geração híbrido [51], o modelo *Transit-Stub*, capaz de criar grandes grafos compostos de grafos randômicos menores. Assim, este modelo permite a criação de topologias utilizando uma estrutura de domínios parecida com a da Internet.

Os grafos gerados possuem várias sub-redes (*stubs*) interconectadas por núcleos (*transits*), que representam os *backbones* da Internet. Para a simulação foi usada uma topologia com quarenta nós – oito no núcleo e mais oito sub-redes com quatro nós. A topologia utilizada é apresentada na figura 4.1. As latências nos enlaces variam entre 0.3 e 3.8 ms e a banda passante é 2 Mbps. Nas simulações, a plataforma não é executada no nós do *backbone*, só sendo executada nos 32 nós das sub-redes. Logo, o número máximo de estações simultâneas participando do mundo virtual para esta topologia é 32.

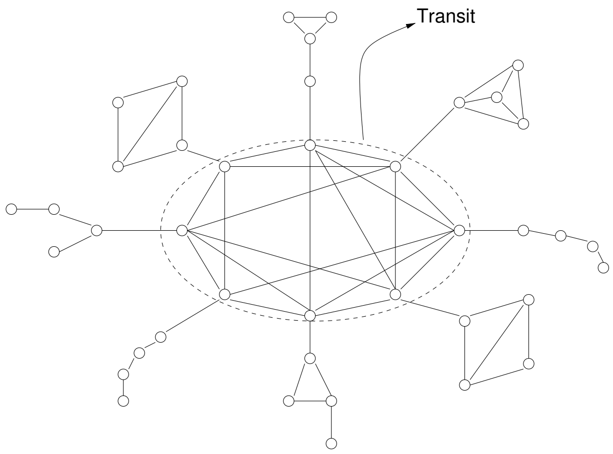


Figura 4.1: Topologia utilizada nas simulações.

4.2 Modelo de Simulação

O agenteRV implementado no *ns* modela o comportamento da plataforma. Para que uma simulação seja executada utilizando este agente, é necessário realizar um *script* que simule o comportamento dos usuários participando da sessão simulada. Este *script* configura os parâmetros definindo as características do ambiente virtual, em que nós da topologia estão os usuários, os participantes que atuam como líderes, o comportamento dos usuários etc.

Todas estas características são mapeadas em parâmetros do modelo de simulação. A aplicação precisa configurar uma base de dados contendo o cenário inicial do mundo virtual, esta base de dados é mapeada em um parâmetro do modelo que define o tamanho inicial da base de dados. A definição do avatar é mapeada num parâmetro que representa o tamanho de um avatar. Nas simulações realizadas, assume-se que todos os avatares têm o mesmo tamanho.

Estes dois parâmetros são usados na adesão e no desligamento dos participantes do mundo. Quando um novo usuário adere a um grupo, a base de dados cresce, pois o tamanho do avatar do novo usuário é acrescentado à base de dados. De forma similar, quando um participante deixa um escopo, seu avatar é retirado da base de dados, decrementando o tamanho da base de dados do tamanho do avatar.

Nas simulações, o ambiente virtual é divido em áreas de tamanho fixo, modelando a filtragem por distância. O espaço do mundo virtual, ou seja, o número de posições possíveis, é mantido fixo, enquanto o número de grupos varia. Logo, com o aumento do número de grupos, a área de cada grupo diminui. Esta divisão é feita entre todos os líderes, de forma que não haja interseção entre os escopos dos diferentes líderes e que a união de todos os escopos resulte na totalidade do mundo virtual. A escolha dos líderes é feita aleatoriamente[1] entre todos os possíveis participantes. A posição inicial dos participantes também é escolhida aleatoriamente entre todas as posições possíveis no ambiente.

Não é simulada a eleição de líderes durante uma sessão. Os líderes são escolhidos no início da simulação e são os primeiros a entrar no mundo virtual. Eles permanecem fixos, ou seja, no mesmo grupo, durante toda a sessão e saem no final, após todos os outros participantes. O tempo de entrada e saída do mundo de cada participante não líder é gerado aleatoriamente.

A movimentação dos participantes através do mundo está associada a uma probabilidade de movimento (P_m). Assim, com probabilidade P_m um participante movimenta-se uma posição para frente, ou seja, incrementa em uma unidade sua posição. Esta probabilidade está relacionada a velocidade do participante, logo, participantes mais velozes têm maior probabilidade de movimento. Neste trabalho, assume-se uma aplicação em que todos os participantes têm a mesma probabilidade de movimento. A cada mudança de posição é enviada uma mensagem de atualização de estado para o grupo multiponto (não foi modelada a utilização de nenhum algoritmo de *dead-reckoning*). Quando a posição do participante ultrapassa os limites de seu grupo, o líder inicia o *handoff* deste participante.

As simulações foram realizadas baseadas neste modelo utilizando a topologia des-

[1] Todos os valores aleatórios foram gerados utilizando as variáveis aleatórias disponíveis no *ns*.

crita na seção 4.1.2. Os parâmetros de simulação utilizados para obter os resultados são apresentados na tabela 4.1. O tamanho inicial do banco de dados, que representa o escopo de interação de um usuário é o maior tamanho. Um escopo simples pode ser representado por 10 k octetos. Foram considerados avatares não muito complexos representados por 1,5 k octetos. As mensagens de atualização contêm apenas informações de comportamento que são representadas com poucos octetos. Para este modelo estas mensagens são representadas com um tamanho de 100 octetos. A probabilidade de movimento utilizada é de 20 %, que representa avatares não muito velozes, como por exemplo seres humanos correndo. O tempo de simulação de cada sessão utilizando a plataforma foi de cinco minutos.

Tabela 4.1: Parâmetros utilizados na simulação.

Parâmetros	*Valores*
Tamanho inicial BD	10 k octetos
Tamanho avatar	1,5 k octetos
Tamanho atualizações	100 octetos
P_m	20 %

4.3 Medidas Realizadas

O modelo de simulação apresentado na seção 4.2 foi adotado para avaliar o desempenho da plataforma para aplicações de realidade virtual distribuída com requisitos diferentes. Os parâmetros utilizados para mensurar o comportamento da plataforma foram: os atrasos para entrada no mundo virtual e para realização de *handoffs*, o número de mensagens trocadas pela plataforma em uma sessão e o número total de *handoffs* realizados pelos participantes de uma sessão. Para cada configuração do modelo de simulação foi calculada a média aritmética destes parâmetros obtidos em cada sessão simulada.

O atraso para entrada e para realização de um *handoff* em cada sessão foram calculados obtendo-se a média aritmética do atraso de todos os participantes de uma sessão. O atraso para entrada no ambiente virtual é composto pelos seguintes elementos:

1. envio de um pedido de adesão (PDU JR) pelo SRM ao grupo dos líderes;
2. resposta do líder responsável pelo escopo ao pedido adesão;
3. estabelecimento da conexão TCP;
4. envio do banco de dados pelo TCP.

Qualquer participante ao entrar no mundo virtual está sujeito aos atrasos impostos pelos ítens 1 e 3, independentemente da carga do sistema. Porém, o atraso referente ao ítem 2 acontece apenas quando o líder responsável por responder ao novo participante está sobrecarregado, ou seja, já está respondendo a algum outro participante. É importante ressaltar que no *ns* não são considerados atrasos de processamento. Assim, o atraso referente ao ítem 2 representa apenas o tempo de espera na fila de pedidos pendentes enquanto outros participantes estão sendo atendidos. Já o atraso relativo ao ítem 4 é dependente do tamanho dos dados a serem transmitidos, variando com o tamanho em octetos do mundo virtual.

Outros fatores que influenciam o atraso são a topologia utilizada e a posição do líder e do novo participante nesta topologia. A topologia, como descrito na seção 4.1.2, é formada por sub-redes interligadas por núcleos. O caminho entre duas sub-redes distintas pode passar por um ou mais núcleos. Quanto maior o número de núcleos entre as sub-redes, maior o número de saltos que um pacote precisará percorrer e, conseqüentemente, maior o atraso. A topologia também determina a latência nos enlaces e a banda passante, fatores que influenciam diretamente o atraso na transmissão de um pacote entre duas estações. A posição das estações participante e líder na topologia escolhida também influenciam o atraso porque determinam o número de enlaces que os pacotes percorrerão. Se, por exemplo, as duas estações estiverem na mesma sub-rede, o número de saltos é menor e, provavelmente, o atraso também é menor do que se as estações estiverem em sub-redes distintas.

O atraso para realização de um *handoff* é composto pelos mesmos elementos que o atraso para entrada no mundo virtual acrescentado do atraso para o líder sinalizar ao participante que ele deve iniciar um *handoff*. Então, o primeiro componente deste atraso é o envio pelo TCP de uma PDU FNL e os outros são os mesmos do atraso de entrada no mundo virtual.

Outra medida realizada foi o número de mensagens trocadas pela plataforma, visando analisar a inundação do sistema de comunicação. As mensagens são contadas por enlace percorrido. Assim, cada mensagem enviada por uma estação final (líder ou participante) que gera mais de uma mensagem nos enlaces - por ser uma mensagem multiponto, por percorrer um caminho com mais de um salto ou por ser segmentada pelo TCP - é contada mais de uma vez. As mensagens foram dividas em dois grupos: mensagens de controle, que representam as mensagens críticas (PDUs JR, JC, JI, FNL, LR), e mensagens não críticas, ou seja, as mensagens de atualização (PDU ES). O número total de mensagens trocadas em uma sessão é igual a soma das mensagens controle com as mensagens não críticas.

Por último, foi medido o número de *handoffs* realizados em uma sessão. Este número representa o número total de *handoffs* realizados por todos os participantes juntos durante o decorrer de uma sessão. Cada *handoff* tem um atraso associado, representando uma interrupção na percepção que o usuário tem do ambiente virtual. Logo, quanto maior o número de *handoffs* maior o número de interrupções. Diferente das medidas anteriores, o número de *handoffs* não depende da topologia, este número está relacionado apenas ao comportamento e à distribuição dos avatares no mundo virtual e ao particionamento deste mundo em escopos. Esta medida também é sensível ao número de usuários, pois para um mesmo número de *handoffs* por usuário, o número total vai aumentar proporcionalmente ao número de usuários.

4.4 Resultados Obtidos

A plataforma tem como principais objetivos a escalabilidade e a flexibilidade. Por isso, a avaliação do seu desempenho foi realizada observando estes dois aspectos. As simulações realizadas mediram os parâmetros discutidos na seção anterior para verificar o impacto do particionamento em escopos, da velocidade dos participantes, do número de participantes e do tamanho do mundo virtual no funcionamento da plataforma.

4.4.1 Particionamento do Ambiente Virtual

O particionamento do ambiente virtual em escopos deve ser realizado de acordo com as características especificadas pela aplicação. Para verificar o impacto deste particionamento na plataforma foram realizadas simulações variando-se o número de grupos.

Quando o número de grupos é igual a um, tem-se apenas um líder para todos os participantes. Neste caso, a plataforma atua como um sistema centralizado e não há filtragem de mensagens entre os participantes do mundo. Como o mundo não é particionado, não há trocas de grupo e, conseqüentemente, também não há interrupções na percepção do mundo. Os participantes sofrem apenas o atraso para entrar no mundo virtual.

Por outro lado, quando o número de grupos é maior do que um, o controle de consistência passa a ser distribuído entre os líderes. Quanto maior o número de grupos, menor o número de participantes a ser controlado por cada líder, maior o nível de distribuição do controle entre estes líderes e menor é o número de mensagens de atualização trocadas em cada grupo. Porém, o número de interrupções na percepção do ambiente cresce, devido às trocas de grupo (*handoffs*) mais freqüentes.

As simulações foram realizadas para 32 participantes, utilizando os parâmetros definidos na seção 4.2 (tamanho inicial do BD de 10 k octetos, tamanho do avatar de 1,5 k octetos, tamanho das atualizações de 100 octetos e P_m igual a 20 %). A figura 4.2 mostra o atraso médio para a entrada de participantes no mundo virtual e para *handoffs*. É importante ressaltar que não existe valor de atraso de *handoffs* quando o valor do número de grupos é igual a um.

A diferença entre o atraso de *handoff* e o de entrada reflete o atraso necessário para o líder sinalizar ao participante que ele deve sair do grupo e este participante iniciar o processo para entrar no novo grupo. Isto significa o atraso para o líder iniciar o *handoff* (o líder pode estar respondendo a algum pedido de outro participante ao detectar a necessidade de realização do *handoff*) mais o atraso para estabelecimento de uma conexão TCP e o envio de uma PDU FNL por esta conexão. Observa-se que esta diferença de tempos diminui com o número de grupos (65 ms para 2 grupos e de 49 ms para 6 grupos). Isto se deve a uma menor sobrecarga do líder que como consequência inicia mais rapidamente os *handoffs*, uma vez que os tempos para estabelecimento da conexão e para o envio da PDU FNL não variam para uma determinada topologia.

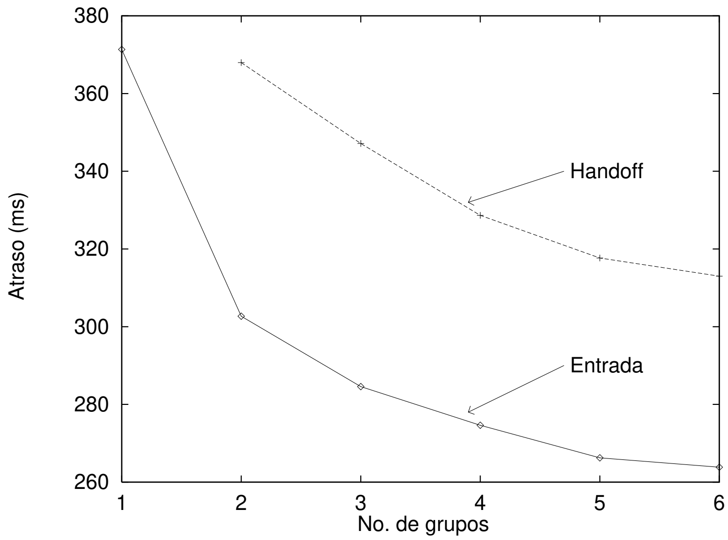


Figura 4.2: Impacto do particionamento em escopos no atraso de entrada e de *handoff*.

Ao aumentar o número de grupos observa-se uma queda tanto no atraso para entrada no mundo quanto no atraso de *handoffs*. Esta melhora do atraso é causada pela repartição entre os líderes da responsabilidade de enviar a base dados para os participantes e de detectar os *handoffs*. Com a menor quantidade de participantes em cada escopo, os líderes ficam menos sobrecarregados e podem responder mais rapidamente aos pedidos de adesão. Esta distribuição é importante para a escalabilidade do ambiente, pois quando o número de participantes cresce, a divisão em grupos mantém o processamento em cada líder aproximadamente constante. O principal componente destes atrasos é a entrega do banco de dados pelo protocolo TCP, causado pelo atraso de transmissão. Assim, um algoritmo de compressão poderia ser utilizado para manter este banco de dados menor possível.

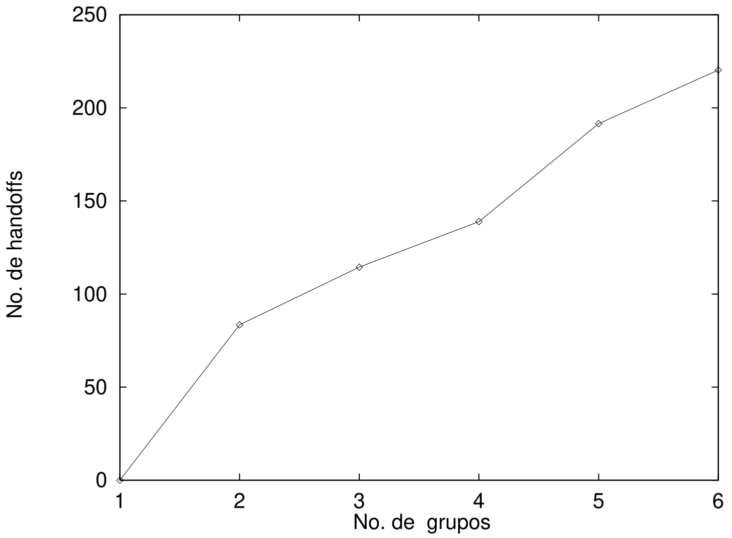


Figura 4.3: Impacto do particionamento em escopos no número de *handoff*.

A figura 4.3 apresenta o número total de *handoffs* realizados por todos os participantes durante o período de simulação. Pode-se observar no gráfico que com o aumento do número de grupos, aumenta a freqüência com que as trocas de grupo são realizadas. Este efeito pode ser facilmente explicado pelo fato do aumento do número de grupos corresponder a uma menor área para cada grupo. Assim, para a mesma probabilidade de movimento, isto significa um maior número de *handoffs*.

A figura 4.4 apresenta o número total de mensagens trocadas. Nota-se que o número total de mensagens trocadas, mensagens de controle mais mensagens de atualização (ES), reduz com o aumento do número de grupos. Este aumento do número de grupos significa escopos de interação menores, logo, as mensagens de atualização são enviadas para um grupo menor de participantes, otimizando o uso dos recursos do sistema de comunicação. Paralelamente, a freqüência de *handoffs* aumenta, aumentando o número de mensagens de controle trocadas para realizar estes *handoffs*. Porém, a redução nas mensagens de atualização é mais significativa que o aumento na troca de mensagens de controle.

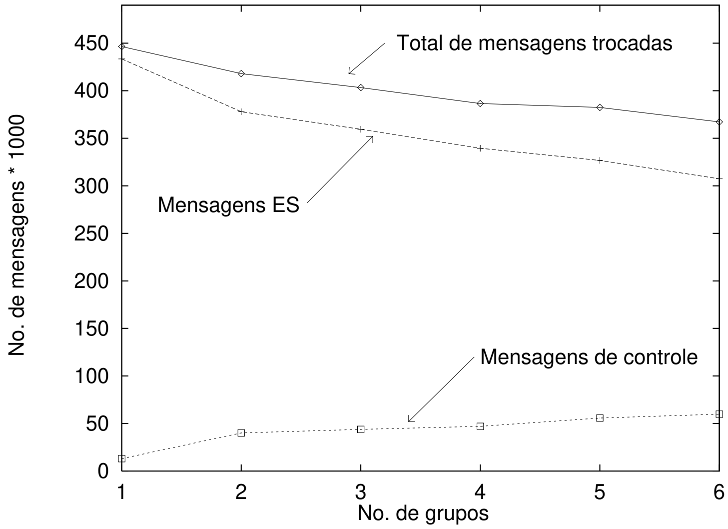


Figura 4.4: Número de mensagens trocadas.

4.4.2 Velocidade dos Participantes

As aplicações de realidade virtual distribuída podem ser povoadas por diferentes tipos de avatares. Em aplicações militares, os avatares são navios, aviões. Em jogos os participantes podem estar representados como pessoas normais, super heróis voadores, animais etc. Cada um destes tipos de avatares possui características próprias que determinam seu comportamento no ambiente virtual. Uma das características de maior influência no desempenho da plataforma é a velocidade com que estes avatares movimentam-se.

Para verificar o impacto da velocidade dos avatares no funcionamento da plataforma, foram realizadas simulações variando-se a probabilidade de movimento (P_m), parâmetro que determina a velocidade dos participantes. Os outros parâmetros foram mantidos como apresentados na seção 4.2 e o número de usuários simulados foi 16. As simulações foram realizadas tanto para o número de grupos igual a 4, quanto para o número de grupos igual a 8.

A figura 4.5 apresenta o número de *handoffs* realizados por todos os participantes durante toda uma sessão. Pode-se reparar que o número de *handoffs* cresce linearmente com a probabilidade de movimento. Com o aumento da velocidade dos avatares, os participantes trocam mais de grupos, realizando mais *handoffs* por sessão. O número de *handoffs* é maior para as simulações com o mundo virtual divido em 8 grupos do que as com 4 grupos. Isto ocorre devido a redução da área de cada grupo, como discutido na seção 4.2.

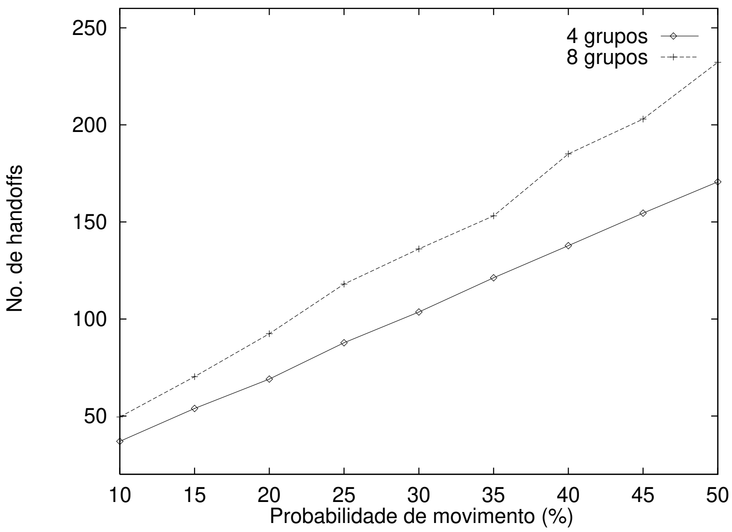


Figura 4.5: Número de *handoffs* com a variação da velocidade dos participantes.

No entanto, o aumento no número de trocas de grupos, gerado por probabilidades de movimento maiores, não tem grande influência no atraso para os participantes entrarem no mundo virtual e nem para realizaram *handoffs*. Isto pode ser verificado na figura 4.6. Pode-se ressaltar que os atrasos para a simulação com 8 grupos são menores do que os da simulação com 4 grupos. Assim, as interrupções causadas pelos *handoffs* são mais curtas quando o ambiente está mais particionado.

A figura 4.7 verifica o impacto da variação da velocidade dos participantes na carga da rede, ou seja, no número de mensagens trocadas. Com o aumento no número de *handoffs*, aumenta também o número de mensagens de controle trocadas. Além disto, como os participantes movem-se com maior velocidade, o número de mensagens de atualização (PDU ES) enviadas também aumenta. Na figura 4.7(a) nota-se que as mensagens de atualização são o principal componente do número total de mensagens. Por isto, a utilização de um algoritmo de dead-reckoning poderia reduzir substancialmente a carga na rede. A divisão em grupos também é importante para esta redução, como pode-se reparar na figura 4.7(b), que compara o número total de mensagens trocadas com 4 e 8 grupos multipontos.

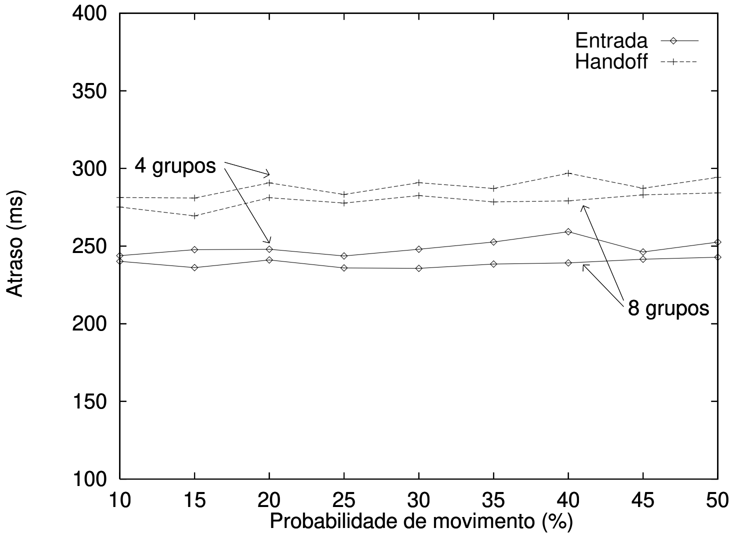
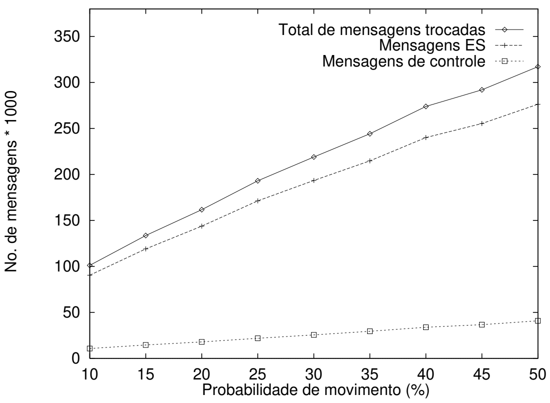
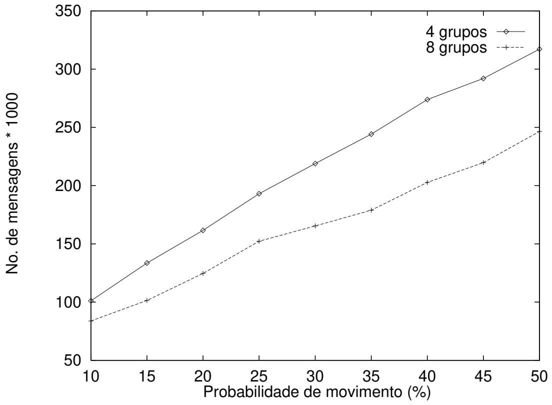


Figura 4.6: Atraso de entrada e de *handoffs* com a variação da velocidade dos participantes.



(a) Número de mensagens para 4 grupos.



(b) Comparação entre 4 e 8 grupos.

Figura 4.7: Impacto da variação de velocidade na troca de mensagens.

4.4.3 Número de Participantes

A plataforma deve suportar aplicações com números diferentes de participantes. Para verificar como a variação no número de participantes é refletida no desempenho da plataforma foram realizadas simulações para 4 e 8 grupos. Os parâmetros utilizados são os apresentados na seção 4.2.

A figura 4.8 mostra a variação do atraso quando o número de participantes cresce. Tanto o atraso para entrar no mundo quanto o atraso para realização de um *handoff* crescem com o aumento do número de usuários. Este aumento ocorre porque o número de líderes foi mantido, assim, quando o número de usuários aumenta, o número por líder também aumenta. Os líderes ficam mais sobrecarregados, demorando mais a responder aos pedidos de adesão e a detectar os *handoffs*. Porém, nota-se no gráfico que o crescimento do atraso desacelera com o aumento do número de usuários e que as simulações realizadas com 8 líderes impõem um atraso menor.

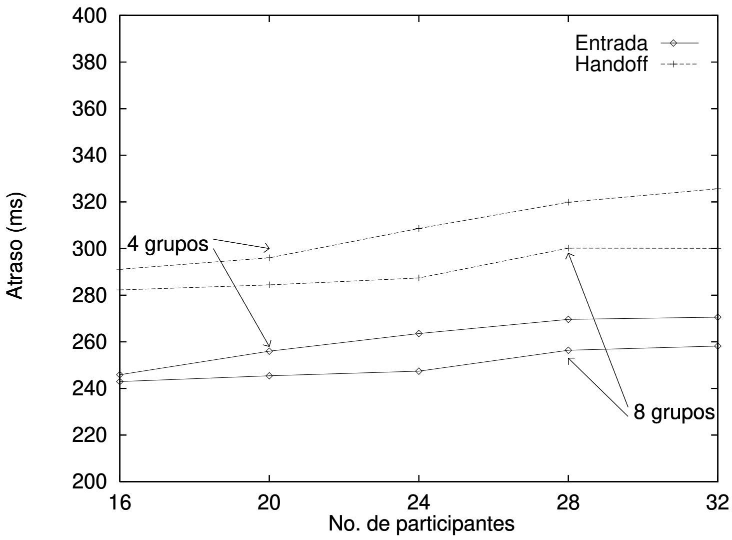
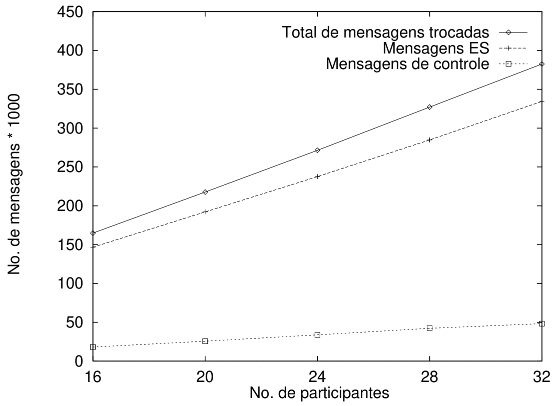
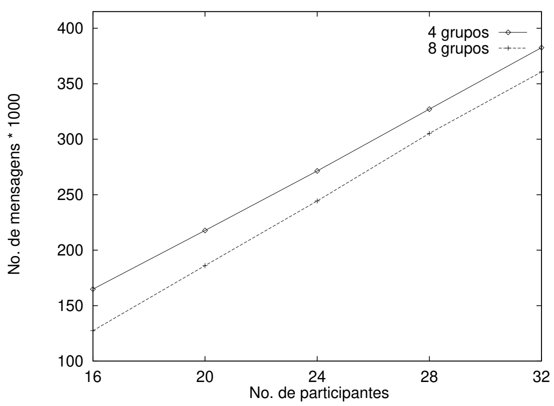


Figura 4.8: Impacto do aumento do número de participantes no atraso.

Quanto maior o número de usuários maior o número de mensagens devem ser trocadas tanto para notificar o estado destes participantes quanto para controlar adesões e desligamentos de escopos. Este crescimento no número de mensagens é mostrado na figura 4.9(a). A figura 4.9(b) apresenta uma comparação entre o número de mensagens trocadas para a simulação com os participantes divididos em 4 e em 8 grupos, onde a simulação com 8 grupos sobrecarrega menos o sistema de comunicação.



(a) Número de mensagens para 4 grupos.



(b) Comparação entre 4 e 8 grupos.

Figura 4.9: Impacto do aumento do número de participantes no número de mensagens trocadas.

4.4.4 Tamanho do Mundo Virtual

Conforme observado na seção 4.3 o tamanho do banco de dados que representa um escopo de interação é um dos componentes determinantes no atraso sofrido por um participante ao entrar no mundo ou ao realizar *handoffs*. Este tamanho pode variar entre aplicações diferentes dependendo da riqueza com que o mundo virtual é representado.

As simulações para avaliar o comportamento da plataforma para escopos com tamanhos diferentes foram realizadas para 32 participantes em configurações com 4 e com 8 grupos. Os outros parâmetros foram mantidos como apresentados na seção 4.2. É importante ressaltar que o valor para o parâmetro tamanho é apenas o tamanho inicial da representação de um escopo. Ao entrarem no escopo, os participantes acrescentam o tamanho de seu avatar a este valor. Então, o tamanho efetivamente transmitido é este valor inicial mais o tamanho de todos os avatares presentes no grupo.

A figura 4.10 apresenta os resultados obtidos para o atraso com a variação do tamanho da base de dados referente a um escopo. É interessante notar que existe um limite inferior para os valores dos atrasos tanto de entrada como de *handoff* para a topologia apresentada na seção 4.1.2. Este atraso mínimo pode ser verificado quando o banco de dados ocupa apenas 1 k octeto. Nesta configuração os componentes com maior impacto no atraso são os tempos para envio da PDU JR e para a abertura da conexão TCP. Pode-se reparar no gráfico que com o aumento do tamanho da base de dados, o atraso aumenta de maneira menos acelerada. Esta desaceleração ocorre devido a segmentação realizada pelo TCP, que permite maior paralelismo no envio dos segmentos de uma mensagem.

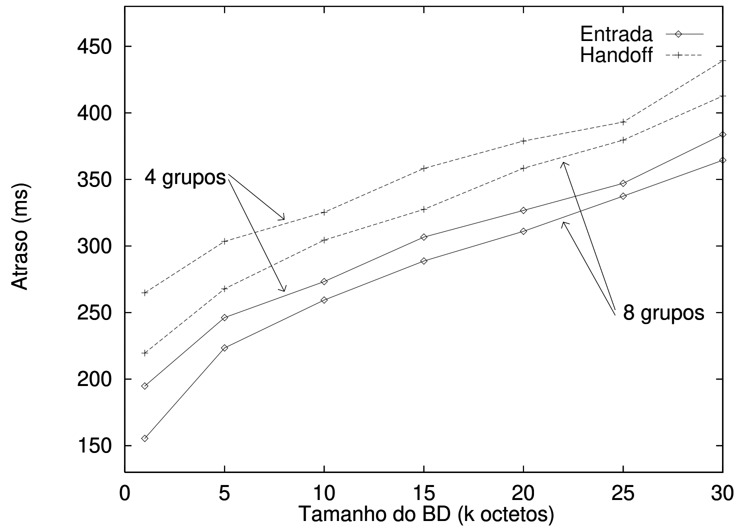


Figura 4.10: Impacto da variação do tamanho do BD no atraso de entrada e de *handoffs*.

A figura 4.11 apresenta a variação do número de mensagens trocadas. As mensagens de atualização não apresentam praticamente nenhuma variação, visto que estas mensagens não dependem do tamanho do banco de dados. Já as mensagens de controle apresentam um crescimento devido ao maior número de segmentos referentes às PDUs JC, que transportam o banco de dados. Estas mensagens, porém, não são muito significativas no número total de mensagens, que mostra um crescimento modesto.

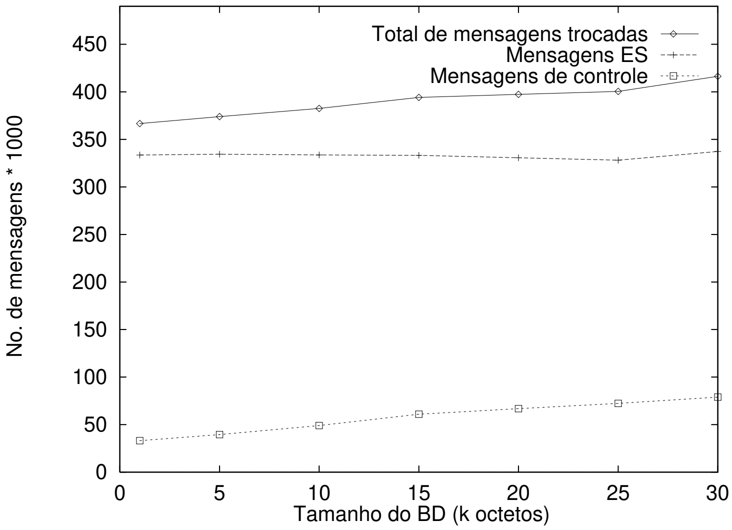


Figura 4.11: Impacto da variação do tamanho do BD no número de mensagens trocadas (4 grupos).

Capítulo 5

Conclusões

A PLATAFORMA proposta neste trabalho oferece um serviço de distribuição e controle de consistência para aplicações de realidade virtual na Internet. Os principais objetivos desta plataforma são a flexibilidade e a escalabilidade. A flexibilidade é alcançada através da proposta de um protocolo genérico, permitindo que a plataforma seja utilizada por qualquer aplicação de realidade virtual distribuída. A escalabilidade é alcançada pela utilização da técnica de filtragem de dados aliada a comunicação multidestinatária, pelo controle de consistência distribuído entre os líderes e pela redução da transmissão de mensagens confiáveis.

A simulação da plataforma abrangeu as funções para a manutenção de consistência do ambiente virtual com objetivo de verificar o impacto no comportamento da plataforma de requisitos de diferentes aplicações. Nas simulações foram medidos: os atrasos de entrada no mundo virtual e de *handoff*, que permitem avaliar a interatividade do sistema e a duração das interrupções causadas pelos *handoffs*; o número de *handoffs*, que analisa a freqüência destas interrupções na percepção do ambiente virtual; e o número de mensagens, que mede a inundação do sistema de comunicação com os diversos tipos de mensagens trocadas pela plataforma.

Nas simulações realizadas pode-se reparar que as mensagens que mais influenciam o número total de mensagens trocadas, sobrecarregando a rede de computadores, são as mensagens de atualização (PDU ES). Estas mensagens correspondem a 81%

do total para a simulação com o tamanho inicial do banco de dados igual a 30 k octetos e a 91% para a simulação com tamanho igual a 1 k octeto. Esta diferença no percentual de mensagens de atualização é devida ao aumento do número de mensagens de controle quando o tamanho do banco de dados aumenta. Pode-se reparar que mesmo quando o tamanho do banco de dados é 30 k octetos o percentual de mensagens de atualização é muito elevado. Por isso, a importância de utilizar algoritmo de *dead-reckoning*, minimizando a troca destas mensagens.

Os resultados mostram a importância do particionamento em escopos para viabilizar o desenvolvimento de ambientes virtuais escaláveis. Esta divisão provoca uma diminuição no atraso para as operações de entrada no mundo e *handoffs*, que beneficia a interatividade entre os participantes e reduz a possibilidade de inconsistências. Há um compromisso entre este ganho em interatividade e as interrupções geradas pelos *handoffs* quando o número de grupos aumenta. Porém, para todos os valores de número de grupos simulados, as interrupções não foram significativas, representando menos de 1% do tempo total da sessão de um participante. Assim, a maior carga na rede devido ao aumento no número de participantes pode ser compensada em aplicações que permitam um ambiente mais particionado.

Nas aplicações em que os avatares movimentam-se rapidamente, o número de interrupções devido a *handoffs* e a carga na rede são maiores. Para diminuir a carga na rede é interessante particionar mais o ambiente. Considerando o pior caso, quando P_m igual a 0.5, a redução do número de mensagens quando aumenta-se o número de grupos de 4 para 8 foi de 23.19%. Porém o particionamento aumenta o número de interrupções na percepção do ambiente virtual. Para P_m igual a 0.5, as interrupções representaram 2.69% da sessão de um participante para a simulação com 8 grupos. Enquanto na simulação com 4 grupos, as interrupções representam 1.37%. Novamente, o tempo gasto em interrupções devido aos *handoffs* não foi significativo. É importante ressaltar também que o atraso, tanto para entrada no mundo quanto para *handoffs*, é mantido abaixo dos 300 ms estabelecido pelo DIS para todas as velocidades consideradas. Isto garante boa interatividade mesmo para os participantes mais velozes.

Uma boa interatividade também foi garantida com a variação do número de usuários. Apesar dos atrasos apresentarem um ligeiro aumento, este aumento não acompanhou linearmente o crescimento do número de participantes. Isto ocorreu devido à distribuição dos usuários em grupos, que garante uma boa escalabilidade da plataforma. Já o aumento no número de mensagens de atualização cresceu linearmente com o aumento do número de usuários, sobrecarregando o sistema de comunicação. O maior particionamento do sistema reduz o número total de mensagens, mas este número mantém um crescimento linear com o número de participantes. Por isto, ao configurar a plataforma para uma determinada aplicação, deve-se ter uma noção do número de participantes a serem atendidos e escolher um particionamento em escopos adequado a este número de usuários.

Em relação ao tamanho do ambiente virtual, as simulações foram realizadas com o mundo virtual ocupando entre 4 k octetos (1 k octeto para cada escopo com 4 grupos) e 240 k octetos (30 k octetos para cada escopo com 8 grupos). Este tamanho dita a riqueza com que o cenário do mundo virtual é expresso. Nas simulações realizadas percebe-se que o particionamento permite uma maior riqueza nesta representação, reduzindo os atrasos. Apesar disto, o tamanho do mundo virtual exerce grande influência nos atrasos e deve ser minimizado, para não causar interrupções prejudicando a interatividade. Nas condições simuladas com a divisão em 8 grupos, consegue-se obter boa interatividade com um mundo virtual de até 120 k octetos. Há algumas opções para que a plataforma possa suportar cenários maiores: a realização de algum tipo de compressão na base de dados ou a utilização de redes com banda passante mais larga, como as redes ATM (*Asynchronous Transfer Mode*), o que reduziria sensivelmente o atraso para a transmissão da base de dados.

Os resultados mostram a importância de uma configuração da plataforma otimizada para a aplicação. Aplicações que são povoadas por avatares mais velozes devem buscar reduzir o tamanho da base de dados para diminuir o impacto causado pelas interrupções mais freqüentes. Quando há a necessidade de grande detalhamento do mundo virtual e seus avatares, o ideal seria reduzir o número de usuários permitidos ou a freqüência de movimentação destes usuários. Em todos os tipos de aplicação o particionamento do mundo virtual deve ser realizado cuidadosamente, otimizando o compromisso entre o tamanho dos grupos e o número de grupos.

Pode-se concluir que a plataforma proposta, com a adoção de um modelo distribuído entre líderes e uma divisão do ambiente em vários escopos separados, permitiu uma boa interatividade sem prejudicar a continuidade da aplicação.

Como sugestão para trabalhos futuros, pode-se destacar:

- a simulação da utilização de um algoritmo de *dead-reckoning*, verificando qual a redução no número de mensagens trocadas;
- a ampliação do modelo de simulação para suportar avatares com características diferentes (tamanho e comportamento);
- a simulação da troca de grupos de participantes líderes, devendo ser realizada uma eleição de um novo líder para o escopo;
- a utilização de máquinas com maior poder computacional, possibilitando a simulação com topologias maiores e, conseqüentemente, com maior número de participantes.

Referências Bibliográficas

[1] Charles, J. Virtually There: Taking VR to Work. *IEEE Computer Graphics and Applications 17*, 6 (1997), 4 – 5.

[2] Gobel, M. Industrial Applications of VEs. *IEEE Computer Graphics and Applications 16*, 1 (1996), 10 – 13.

[3] Allison, D., et al. The Virtual Gorilla Exhibit. *IEEE Computer Graphics and Applications 17*, 6 (1997), 30 – 38.

[4] Tate, D. L., Sibert, L., and King, T. Using Virtual Environments to Train Firefighters. *IEEE Computer Graphics and Applications 17*, 6 (1997), 23 – 29.

[5] Mastaglio, T. W., and Callahan, R. A Large-scale Complex Virtual Environment for Team Training. *IEEE Computer* (julho de 1995).

[6] Lehner, V. D., and DeFanti, T. A. Distributed Virtual Reality: Supporting Remote Collaboration in Vehicle Design. *IEEE Computer Graphics and Applications 17*, 2 (1997), 13 – 17.

[7] Lamotte, W., et al. Visinet: Collaborative 3D Visualization and VR over ATM Networks. *IEEE Computer Graphics and Applications 17*, 2 (1997), 66 – 75.

[8] Zeltzer, D. Autonomy, Interaction and Presence. *Presence: Teleoperators and Virtual Environments 1*, 1 (1992), 265 – 287.

[9] Amselem, D. A Window on Shared Virtual Environments. *Presence: Teleoperators and Virtual Environments 4*, 2 (1995), 130 – 145.

[10] SINGHAL, S., AND ZYDA, M. *Networked Virtual Environments - Design and Implementation.* ACM Press, 1999.

[11] MACEDONIA, M. R., AND ZYDA, M. J. A Taxonomy for Networked Virtual Environment. *IEEE Multimedia 4*, 1 (1997).

[12] The Virtual Reality Modeling Language: International Standard ISO/IEC 14772-1:1997. http://www.vrml.org/technicalinfo/specifications/vrml97/.

[13] Living Worlds: Making VRML 2.0 Applications Interpersonal and Interoperable. http://www.vrml.org/WorkingGroups/living-worlds/draft_2.

[14] TEIXEIRA, R. C., AND DUARTE, O. C. M. B. Requisitos de Ambientes Virtuais Distribuídos de Larga Escala. In *I Workshop de Realidade Virtual* (São Carlos, novembro de 1997), pp. 26 – 35.

[15] STYTZ, M. R. Distributed Virtual Environments. *IEEE Computer Graphics and Applications 16*, 3 (maio de 1996), 19 – 31.

[16] DE ARAUJO, R. B., DE OLIVEIRA, M. A. M. S., SWENSSON, R. P., AND CRONEMBERGER, R. R. Exploração de Ambientes Virtuais Distribuídos de Grande Porte. In *I Workshop de Realidade Virtual* (São Carlos, novembro de 1997), pp. 11 – 18.

[17] PIMENTEL, K., AND BLAU, B. Teaching Your System to Share. *IEEE Computer Graphics and Applications 14*, 1 (janeiro de 1994).

[18] CAPIN, T. K., PANDZIC, I. S., THALMANN, D., AND THALMANN, N. M. A Dead-Reckoning Algorithm for Virtual Human Figures. In *VRAIS'97* (Albuquerque, USA, 1997), IEEE Press, pp. 161 – 168.

[19] SINGHAL, S. K., AND CHERITON, D. Using a Position History-Based Protocol for Distributed Object Visualization, julho de 1994.

[20] BENFORD, S., GREENHALGH, C., AND LLOYD, D. Crowded Collaborative Virtual Environments. In *ACM Computer-Human Interface CHI'97* (1997).

[21] Macedonia, M. R., Zyda, M. J., Pratt, D. R., Brutzman, D. P., and Barham, P. T. Exploiting Reality with Multicast Groups. *IEEE Computer Graphics and Applications* (setembro de 1995), 38 – 50.

[22] de Araujo, R. B., and Kirner, C. Análise dos Requisitos de Rede de um Sistema Distribuído de Realidade Virtual. In *XXIII Seminário Integrado de Software e Hardware* (1996).

[23] Deering, S. *Host Extensions for IP Multicasting.* RFC-1112, agosto de 1989.

[24] Macedonia, M. R., and Brutzman, D. P. MBone Provides Audio and Video Across the Internet. *IEEE Computer* (abril de 1994).

[25] Greenhalgh, C. Spatial Scope and Multicast in Large Virtual Environments. Tech. rep., The University of Nottingham/Department of Computer Science, setembro de 1996.

[26] Abrams, H., Watsen, K., and Zyda, M. Three tiered interest management for large-scale virtual environments. In *VRST 98* (Taipei, novembro de 1998).

[27] Levine, B. N., and Garcia-Luna-Aceves, J. A Comparison of Known Classes of Reliable Multicast Protocols. In *Proceedings International Conference on Network Protocols* (outubro de 1996).

[28] Pingali, S., Towsley, D., and Kurose, J. A Comparison of Sender-Initiated and Receiver-Iniciated Reliable Multicast Protocols. *IEEE Journal on Selected Areas in Communications 15*, 3 (abril de 1997), 398 – 406.

[29] Institute for Defense Analysis. SIMNET. maio de 1990.

[30] Standard Commitee on Interactive Simulation, IEEE Computer Society. IEEE Standard for Distributed Interactive Simulation. IEEE Std 1278.1-1995, 1995.

[31] Calvin, J. O., and Weatherly, R. An Introduction to the High Level Architecture (HLA) Runtime Infrastructure (RTI). In *DIS Workshop on Simulation Standards* (Cambridge Massachussetts, março de 1996), Massachussetts Institute of Technology.

[32] Hook, D. J. V., Rak, S. J., and Calvin, J. O. Approaches to RTI Implementation of HLA Data Distribution Management Services. In *DIS Workshop on Simulation Standards* (Cambridge Massachussetts, setembro de 1996), Massachussetts Institute of Technology.

[33] Macedonia, M. R., Zyda, M. J., Pratt, D. R., Barham, P. T., and Zeswitz, S. NPSNET: A Network Software Architecture for Large Scale Virtual Environments. *Presence: Teleoperators and Virtual Environments 3*, 4 (1994), 265 – 287.

[34] Macedonia, M. R. *A Network Software Architecture for Large Scale Virtual Environments.* PhD thesis, Naval Postgraduate School, Monterey, California, junho de 1995.

[35] Brutzman, D., Zyda, M., Watsen, K., and Macedonia, M. Virtual Reality Transfer Protocol (VRTP) Design Rationale. In *Workshops on Enabling Technology: Infrastructure for Collaborative Enterprises (WET ICE): Sharing a Distributed Virtual Reality* (Cambridge Massachussetts, junho de 1997), Massachussetts Institute of Technology.

[36] Greenhalgh, C., and Benford, S. MASSIVE: A Distributed Virtual Reality System Incorporating Spatial Trading. In *Proceedings IEEE DCS'95* (Vancouver, Canada, 1995).

[37] Greenhalgh, C., and Benford, S. MASSIVE: A Collaborative Virtual Environment for Teleconferencing. *ACM Transactions on Computer-Human Interaction 2*, 3 (setembro de 1995), 239 – 261.

[38] Hagsand, O. Interactive Multiuser VEs in the DIVE System. *IEEE Multimedia 3*, 1 (1996), 30–39.

[39] Singh, G., et al. BrickNet: A Software Toolkit for Network-based Virtual Environments. *Presence: Teleoperators and Virtual Environments 3*, 1 (1994), 19 – 34.

[40] Barrus, J. W., Waters, R. C., and Anderson, D. B. Locales: Supporting Large Multiuser Virtual Environments. *IEEE Computer Graphics and Applications 16*, 6 (novembro de 1996), 50–57.

[41] de Araujo, R. B., and Kirner, C. Especificação e Análise de um Sistema Distribuído de Realidade Virtual. In *XIV Simpósio Brasileiro de Redes de Computadores* (1996), pp. 357 – 374.

[42] Deriggi Jr, F. V., Kubo, M. M., Sementille, A. C., Brega, J. R. F., dos Santos, S. G., , and Kirner, C. CORBA Platform as Support for Distributed Virtual Environments. In *IEEE Virtual Reality 99 (VRAIS 99)* (Houston, Texas, USA, março de 1999).

[43] Teixeira, R. C., and Duarte, O. C. M. B. Uma Plataforma para Sistemas de Realidade Virtual Distribuída. In *XXIX Congresso da Sociedade Brasileira de Computação, XXVI Seminário Integrado de Software e Hardware (SEMISH'99),* (Rio de Janeiro, julho de 1999).

[44] Floyd, S., Jacobson, V., Liu, C.-G., McCanne, S., and Zhang, L. A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing. *IEEE/ACM Transactions on Networking 5*, 6 (dezembro de 1997), 784 – 803.

[45] Diaz, M. Modeling and Analysis of Communication and Cooperation Protocols Using Petri Net Based Models. *Computer Networks and ISDN Systems 6*, 6 (dezembro de 1982), 419 – 441.

[46] Fall, K., and Varadhan, K. NS Notes and Documentation. Tech. rep., The VINT Project, novembro de 1997.

[47] Bajaj, S., et al. Improving simulation for network research. Tech. Rep. 99-702, University of Southern California, março de 1999.

[48] Huitema, C. *Routing in the Internet.* Prentice-Hall, 1995.

[49] Deering, S., Partridge, C., and Waitzman, D. *Distance Vector Multicast Routing Protocol.* RFC-1075, novembro de 1988.

[50] Peterson, L. L., and Davie, B. S. *Computer Networks: A Systems Approach.* Morgan Kaufmann Publishers, San Francisco, California, 1996.

[51] Zegura, E. W., Calvert, K., and Bhattacharjee, S. A Quantitative Comparision of Graph-Based Models for Internet Topology. *IEEE/ACM Transactions on Networking 5*, 6 (dezembro de 1997), 770 – 783.

[52] Brown, R. Calendar Queues: A Fast O(1) Priority Queue Implementation for the Simulation Event Set Problem. *Communications of the ACM 31*, 10 (outubro de 1998), 1220 – 1227.

Apêndice A

Simulador *ns*

O SIMULADOR *ns* [46, 47] é um simulador de protocolos de rede em desenvolvimento no projeto *Virtual InterNet Testbed* (VINT), uma colaboração entre UC Berkeley, LBL, USC/ISI e Xerox PARC. O *ns* é um simulador orientado a objetos escrito em linguagem C++. Os usuários geram *scripts* na linguagem OTcl que são interpretados e executados pelo simulador. O simulador suporta uma hierarquia de classes em C++ e uma hierarquia de classes similar para o interpretador OTcl. Os usuários criam novos objetos do simulador através do interpretador que instancia estes objetos e gera um espelho em C++, criando um objeto correspondente.

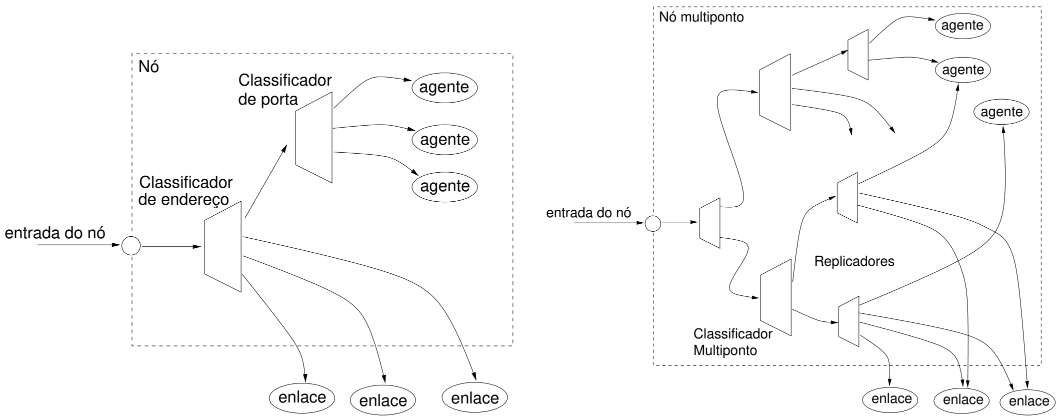
O simulador, descrito por uma classe em Tcl *class Simulator*, oferece um conjunto de interfaces para a configuração de uma simulação e para a escolha do tipo de escalonador de eventos utilizado para conduzir a simulação. Um *script* de simulação, normalmente, inicia criando uma instância desta classe e chamando métodos para a criação de nós, topologias e para a configuração de outros aspectos da simulação.

Atualmente, existem quatro tipos de escalonadores de eventos disponíveis no *ns*, cada um é implementado utilizando uma estrutura de dados diferente: lista simplesmente encadeada, *heap*, *calendar queue* e um tipo especial chamado “tempo real”. O escalonador escolhido para simulação da plataforma foi o *calendar queue* [52], que utiliza uma estrutura de dados análoga ao calendário de um ano, em que eventos num mesmo mês/dia de anos diferentes podem ser mapeados em um único dia. A função do escalonador é selecionar o próximo evento, executá-lo até o fim e retornar para executar o próximo evento. O escalonador do *ns* executa apenas um evento por vez, se mais de uma evento forem escalonados para serem executados ao mesmo tempo, estes eventos são ordenados e executados em série.

As topologias são criadas no *ns* através da criação dos nós e da conexão destes nós por enlaces. O *ns* permite a utilização de dois tipos de nós, apresentados na figura A.1. A função de um nó, quando recebe um pacote, é examinar seus campos, normalmente o endereço destino, mapeando os valores em um objeto de interface de saída que é o próximo receptor deste pacote. Esta função é executada por um objeto chamado classificador. Vários tipos de classificadores, cada um verificando um pedaço do pacote, são usados em um nó no *ns*. Um classificador deve determinar um número de *slot* associado ao pacote recebido e enviar o pacote para o objeto referenciado por este *slot*.

Nos nós ponto-a-ponto (*unicast*), como mostrado na figura A.1(a), a entrada do nó referencia o classificador de endereço. Este classificador é responsável por verificar o endereço destino e produzir um número de *slot* associado ao próximo objeto que deve receber o pacote. Nos nós multipontos (figura A.1(b)), a entrada do nó referencia um comutador que verifica o primeiro *bit* para decidir se o pacote deve ser enviado para o classificador ponto-a-ponto ou para o multiponto. O classificador multiponto classifica o pacote tanto de acordo com o endereço de destino (endereço de grupo) como com o de origem, definindo qual replicador deve receber este pacote. O replicador possui uma tabela que é construída mapeando-se o endereço fonte e destino em um *slot*, o pacote recebido deve ser copiado uma vez para cada enlace levando a nós pertencentes ao grupo.

Os enlaces no *ns* são compostos por objetos chamados conectores. Os conectores, diferentemente dos classificadores, apenas geram dados para um receptor. O pacote pode ser enviado para este receptor ou pode ser descartado. Um enlace é criado com características específicas de banda passante e atraso. Os enlaces possuem conectores do tipo *queue* que modelam as filas de saída associadas aos enlaces. O *ns* suporta diversos tipos de escalonadores de pacote, para a simulação da plataforma



(a) *Ponto-a-ponto.* (b) Multiponto.

Figura A.1: Estrutura dos nós no *ns*.

foi usado o *drop-tail queueing* (FIFO).

Os agentes, mostrados na figura A.1, representam pontos onde os pacotes são gerados ou consumidos e são usados para a implementação de protocolos de várias camadas. Existem vários agentes já implementados no *ns* entre eles: o *FullTcp* (TCP com tráfego nos dois sentidos, implementação do TCP no *ns* mais próxima da utilizada na Internet), o *UDP*, o *SRM* e o *Message/Prune* (processa as mensagens de *prune* do roteamento multiponto).