COPPE/UFRJ

RECOLLVE: UM AMBIENTE VIRTUAL VOLTADO PARA A REPRESENTAÇÃO DE ATIVIDADES COLABORATIVAS

José Valentim dos Santos Filho

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Elétrica.

Orientadores: Aloysio de Castro Pinto Pedroza

Jean-Pierre Courtiat

Rio de Janeiro

Março de 2009

RECOLLVE: UM AMBIENTE VIRTUAL VOLTADO PARA A REPRESENTAÇÃO DE ATIVIDADES COLABORATIVAS

José Valentim dos Santos Filho

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Aloysio de Castro Pinto Pedroza, Dr.

Prof. Jean-Pierre Courtiat, Dr. d'État.

Prof. José Ferreira de Rezende, Dr.

Profa. Luci Pirmez, D.Sc.

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

Prof. Roberto Willrich, Dr.

RIO DE JANEIRO, RJ – BRASIL

MARÇO DE 2009

dos Santos Filho, José Valentim

RECOLLVE: UM AMBIENTE VIRTUAL VOLTADO PARA A REPRESENTAÇÃO DE ATIVIDADES COLABORATIVAS/José Valentim dos Santos Filho. – Rio de Janeiro: UFRJ/COPPE, 2009.

XVIII, 118 p.: il.; 29, 7cm.

Orientadores: Aloysio de Castro Pinto Pedroza

Jean-Pierre Courtiat

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2009.

Referências Bibliográficas: p. 100 – 108.

1. Sistemas Colaborativos. 2. Ambientes Virtuais. 3. Agentes Inteligentes. 4. Percepção. I. Pedroza, Aloysio de Castro Pinto *et al.*. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Dedico esta Tese ao meu filho

Pedro.

Agradecimentos

A Deus, pelo fantástico dom da vida.

Ao meu Pai (*in memorian*), por me ensinar o valor do trabalho.

À minha Mãe Jerusa, que muito embora não tenha tido oportunidade de concluir seus estudos, me ensinou a valorizar o estudo e a educação desde sempre.

À minha esposa Alessandra pelo apoio, carinho e amor em todas as horas.

Ao Prof. Aloysio, que me acolheu no GTA, me orientou, e que sempre foi um amigo e uma inesgotável fonte de estímulo, principalmente nos momentos mais difíceis.

Ao Prof. Jean-Pierre pelo apoio e amizade que muito facilitaram minha adaptação em terras Francesas.

Ao Prof. Otto por, juntamente com o prof. Aloysio, ter aberto as portas do GTA para mim, e a quem eu aprendi a admirar ao longo destes anos de convívio no GTA.

Aos Profs. Roberto Willrich, Luci Pirmez, José Rezende e Otto Duarte, pela presença na banca e contribuições à tese.

Aos amigos do GTA: Kléber, Miguel, Lafs, Pedro, Doc, Paulo André, Arthur, Rubi, Luis Henrique, Gardel, Saulo, Márcio e Igor.

Aos amigos do CECOMP/UNIVASF pelo apoio e liberação da minha carga horária durante seis meses para que eu pudesse me dedicar à conclusão desta Tese: Ana, Anderson, Edmundo, Fábio, Leonardo, Marcelo, Marcus e Max. Agradeço também

aos novos membros do CECOMP/UNIVASF pelo apoio: Juracy, Ricardo, Críston, Jorge e Eliane. Agradeço ainda ao nosso assistente administrativo Neto, por todo o apoio operacional.

Agradecimento especial aos meus alunos de Iniciação Científica, Rodrigo Bacurau e Alisson Amorim, pela ajuda no desenvolvimento da biblioteca de animações e criação de algumas cenas virtuais que serviram como estudos de caso.

À equipe da secretaria do PEE: Dani, Maurício, Rosa e todos os demais; pelo suporte operacional.

Aos amigos de Toulouse, que tornaram a minha estada na França extremamente agradável: Roberta, Magnos, Guillermo, Francisco, Eduardo Loures, Vilemar, Lena, Valter, Carlos, Leandro, Luizão, Florence, Alexandre, Chyntia, David (o maestro) e Joandre.

Ao PEE/COPPE, pelas instalações e equipamentos utilizados.

Ao LAAS/CNRS, também pelas instalações e equipamentos utilizados.

Agradeço à Comissão de Aperfeiçoamento de Pessoal Docente (CAPES) pelo suporte financeiro durante 18 meses.

"Os rios não querem chegar a nenhum lugar,

eles só querem ficar mais largos e profundos."

João Guimarães Rosa

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

RECOLLVE: UM AMBIENTE VIRTUAL VOLTADO PARA A REPRESENTAÇÃO DE ATIVIDADES COLABORATIVAS

José Valentim dos Santos Filho

Março/2009

Orientadores: Aloysio de Castro Pinto Pedroza

Jean-Pierre Courtiat

Programa: Engenharia Elétrica

A presente Tese apresenta RECOLLVE: um ambiente virtual voltado para a representação de atividades colaborativas, assim como o gerenciamento das aplicações envolvidas na atividade de colaboração. RECOLLVE é composto por uma arquitetura multiagentes que o torna flexível o suficiente para representar, na cena virtual, a atividade de colaboração que acontece no mundo real, assim como permite ao usuário gerenciar as aplicações do mundo real a partir de ações realizadas na cena virtual.

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

RECOLLVE: A VIRTUAL ENVIRONMENT FOCUSED ON REPRESENT COLLABORATIVE APPLICATIONS

José Valentim dos Santos Filho

March/2009

Advisors: Aloysio de Castro Pinto Pedroza

Jean-Pierre Courtiat

Department: Electrical Engineering

This Thesis work presents RECOLLVE: a virtual environment focused on to represent collaborative activities. RECOLLVE is composed by a multi-agents architecture which makes it flexible enough to represent, into the virtual scene, the collaboration activity realized in the real world, such as, it allows user to manager collaborative applications of real world from actions made into the virtual scene.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

ACL	*Agent Communication Language*, p. 31
ADVICE	*Ambiente Virtual Colaborativo para o Ensino a Distância*, p. 21
AVC	*Ambientes Virtuais Colaborativos*, p. 2
BSI	*Browser Script Interface*, p. 33
CSCW	*Computer Supported Cooperative Work*, p. 1
CVE	*Collaborative Virtual Environments*, p. 12
DIVE	*DIstributed Virtual Environments*, p. 18
EAI	*External Authoring Interface*, p. 33
EVE	*Education Virtual Environment*, p. 19
FIPA	*Foundation for Intelligent Physical Agents*, p. 31
H-Anim	*Humanoid Animation Working Group*, p. 50
INVITE	*Intelligent Distributed Virtual Training Environment*, p. 19
JADE	*Java Agent DEvelopment framework*, p. 31

LEICA	*Loosely-coupled Environment for Integrating Collaborative Applications*, p. 52
LW	*LIVING WORLDS*, p. 36
MUVE	*Multi User Virtual Environment*, p. 13
NETICE	*NETwork Intelligent Collaborative Environment*, p. 22
RECOLLVE	*REpresenting COLLaboration in Virtual Environments*, p. 28
SAI	*Scene Access Interface*, p. 51
SMA	*Sistemas Multiagentes*, p. 31
STEVE	*Soar Training Expert for Virtual Environment*, p. 24
TCSC	*Trabalho Colaborativo Suportado por Computador*, p. 1
VIP	*VRML Interchange Protocol*, p. 39
VRENG	*Virtual Reality Engine*, p. 23
VRML	*Virtual Reality Modeling Language*, p. 32
X3D	*Extensive 3D Graphics*, p. 50

Capítulo 1

Introdução

1.1 Contexto Geral

Vivemos a era da globalização, onde é cada vez mais comum a existência de pessoas de diferentes nacionalidades espalhadas pelo mundo, inclusive equipes de trabalho de diversas organizações. Essa realidade tem provocado uma crescente demanda por comunicação e colaboração entre pessoas distantes geograficamente. O cenário descrito, combinado com o cada vez maior poder de processamento dos computadores pessoais e das redes de computadores de alta velocidade, provocou uma enorme demanda por aplicações do tipo *TCSC*, ou do Inglês *CSCW*.

O TCSC é o domínio que estuda a concepção, implementação e utilização de aplicações colaborativas, também chamados de *Groupwares*. Os *Groupwares* são sistemas cooperativos que permitem a vários usuários trabalharem em conjunto por intermédio de uma infraestrutura de redes de computadores. Sistemas de áudio e vídeoconferência, *whiteboard*, *chats*, sistemas de navegação WEB cooperativos e

ambientes virtuais colaborativos (AVC) são exemplos de aplicações colaborativas. Esses sistemas estão disponíveis como produtos acadêmicos e comerciais.

Sistemas TCSC são um tema complexo e multidisciplinar. Segundo Gomes [1]:

"...O dominio TCSC é multidisciplinar porque trata diferentes tipos de problemas. No plano técnico, é preciso considerar que um sistema TCSC é uma aplicação interativa, multiusuário e distribuída. Assim, diferentes aspectos tais como acesso concorrente para compartilhar recursos, controle de acesso de usuários e sincronização da comunicação em rede, entre outros, devem ser considerados por esse tipo de sistema.

No plano sociológico, os sistemas TCSC devem considerar os aspectos mais humanos do trabalho cooperativo. É preciso considerar, por exemplo, como iniciar uma atividade de colaboração, i.e., como os usuários iniciam o processo de colaboração. O gerenciamento de papéis (do Inglês, *roles*) e a percepção de grupo são igualmente importantes."

Também são encontrados na literatura Groupwares como o sistema *PLATINE* [2], que agrupa um conjunto de ferramentas específicas, tentando prever as necessidades dos usuários em termos de funcionalidades que o sistema deve oferecer. No entanto, segundo *Hummes et al* [3], *Groupwares* como *PLATINE*, encontram dificuldades para serem aceitos por grupos de trabalho reais, porque é praticamente impossível prever a maneira como as pessoas trabalham em conjunto. Como consequência, ainda segundo *Hummes et al.*, é praticamente impossível para desenvolvedores predefinir as reais necessidades e artefatos necessários para o desenvolvimento de trabalho em grupo. Neste contexto, as ferramentas de integração desempenham um papel muito importante, pois elas permitem integrar diferentes sistemas colabo-

rativos para suportar trabalho colaborativo. Este fato explica o crescente número de sistemas voltados para a integração de aplicações colaborativas, como o sistema LEICA [4], por exemplo. Esse tipo de sistema oferece aos usuários uma grande flexibilidade para estabelecer sessões colaborativas, permitindo-os adequar às suas necessidades e preferências.

No entanto, o emprego de sistemas TCSC esbarra em um importante problema natural no desenvolvimento de trabalho em grupo: ter uma real noção de presença dos outros usuários com os quais um usuário colabora e ter noção de quais ações esses usuários estão realizando em um dado momento. É neste contexto que está situada a problemática deste trabalho de Tese.

1.2 Problemática Abordada

Um dos problemas mais críticos no emprego de aplicações cooperativas é fornecer ao usuário uma real noção da presença de todos os usuários com os quais ele coopera. A realidade virtual, permitindo representar o mundo real por meio de metáforas, torna esta noção de presença mais natural, com as possiblidades de interação bem próximas daquelas do mundo real. Neste contexto, os Ambientes Virtuais Colaborativos (AVCs) desempenham um papel bastante importante, pois eles permitem à usuários separados geograficamente se comunicar e interagir em um ambiente 3D compartilhado, frequentemente chamado de "cena virtual". Um dos principais desafios na concepção de um Ambiente Virtual Colaborativo (AVC) é que ele seja capaz de representar e gerenciar a dinâmica de trabalho em grupo de usuários distantes geograficamente.

Em toda equipe de trabalho, pessoas desempenham funções diferentes, com pode-

res e responsabilidades diferentes, assim, o gerenciamento dos papéis desempenhados pelos usuários é fundamental, assim como o gerenciamento dos direitos de acesso dos membros de um mesmo grupo de trabalho. Como consequência disto, torna-se obrigatório representar no interior da cena virtual os protocolos sociais utilizados pelas pessoas no mundo real.

Outro aspecto importante no que diz respeito a implementação de AVCs, é que a complexidade na sua concepção e implementação tem levado os desenvolvedores a optar por soluções proprietárias, sem permitir uma integração com outras ferramentas existentes e limitando assim as possibilidades de colaboração ao interior da cena virtual.

1.3 Principais Contribuições do Trabalho de Tese

As contribuições a serem destacadas neste trabalho são:

- a concepção de RECOLLVE: um ambiente virtual baseado em agentes voltado para a representação da dinâmica de atividades colaborativas;
- um mecanismo de controle das aplicações externas a partir de ações na cena virtual;
- um mecanismo que viabiliza a construção de mundos VRML multiusuários;
- a implementação de um protótipo de RECOLLVE.

1.4 Organização do Texto

A presente Tese está organizada em 6 capítulos, sendo o texto organizado da seguinte forma: no Capítulo 2, é apresentado um estado da arte no que diz respeito à pesquisa em ambientes virtuais colaborativos. São apresentadas definições e conceitos relacionados aos sistemas colaborativos. Os AVCs são abordados em seus diferentes aspectos. É realizada também uma avaliação dos AVCs encontrados na literatura, procurando discutir sua importância para o desenvolvimento desta Tese.

No Capítulo 3, são descritas as soluções tecnológicas utilizadas na concepção, modelagem e implementação de RECOLLVE. É realizado um estudo sobre agentes inteligentes e sua importância em aplicações como ambientes virtuais, além de descrever brevemente a plataforma JADE, utilizada para simular a arquitetura multiagentes definida para RECOLLVE. Em seguida, é realizado um estudo sobre as linguagens padrão (VRML/X3D) utilizadas para descrição das cenas virtuais e dos avatares e suas animações (H-Anim). Finalizando o capítulo, o ambiente de integração *LEICA* é apresentado.

O Capítulo 4 descreve formalmente uma sessão virtual colaborativa no sistema RECOLLVE e sua arquitetura multiagentes. São apresentados os agentes da sociedade multiagentes, sua descrição, seu comportamento e como eles interagem para alcançar os objetivos traçados para RECOLLVE.

O Capítulo 5 descreve os detalhes da implementação de RECOLLVE. Além disso, apresenta os estudos de caso desenvolvidos mostrando o potencial de RECOLLVE como plataforma de suporte ao trabalho colaborativo.

Por fim, o Capítulo 6 apresenta as conclusões deste trabalho de tese e perspectivas para o desenvolvimento de trabalhos futuros.

Capítulo 2

Estado da Arte

2.1 TCSC - Conceitos e Classificações

A utilização da informática para gerenciar trabalho em grupo constituiu uma revolução no que diz respeito ao trabalho cooperativo [5]. Esta revolução está na origem do domínio intitulado "TCSC - Trabalho Cooperativo Suportado por Computador", ou do Inglês, *CSCW* (Computer Supported Cooperative Work).

O objetivo do TCSC, em essência, é adaptar as tecnologias existentes às necessidades dos usuários envolvidos no trabalho em grupo. Os aplicativos de software, elaborados dentro do contexto de pesquisa em TCSC, que permitem a um grupo de usuários colaborar com um objetivo comum, recebem o termo de *Groupware*. Mas são encontradas também na literatura as denominações "Sistemas Colaborativos" e "Aplicações Colaborativas". Nós usaremos sempre os termos "Aplicações Colaborativas" ou "Sistemas Colaborativos" para designarmos esses aplicativos de software.

A multidisciplinaridade do TCSC torna difícil a definição desses sistemas. Uma definição bastante utilizada para sistemas TCSC foi dada por [6]:

"Aplicativo de software que apóia um grupo de pessoas trabalhando em uma tarefa comum e fornece uma interface para um ambiente compartilhado".

A seguir são apresentados alguns importantes conceitos concernentes ao domínio do trabalho colaborativo.

2.1.1 Conceitos Fundamentais

Colaboração

A noção de colaboração é caracterizada por três funções principais: comunicação, coordenação e produção [7] [8]. A comunicação corresponde à troca direta de informação entre os membros de um grupo. A coordenação consiste em definir as regras de interação para controlar a participação dos membros do grupo que realizam um trabalho comum. A produção é caracterizada pelo compartilhamento de um espaço onde os membros podem armazenar, processar e compartilhar vários objetos comuns. Nós utilizaremos os termos "trabalho colaborativo" ou "atividade colaborativa" para designarmos qualquer reunião entre usuários na qual as noções de comunicação, coordenação e produção estejam presentes.

A Telepresença

A telepresença, ou noção de presença, representa a consciência (do inglês, *awareness*) comum em um trabalho de grupo que permite definir o contexto no qual o trabalho se realiza. Essa consciência de grupo corresponde à compreensão que cada um tem sobre: (i) com quem o usuário trabalha; (ii) a atividade de cada um dos usuários

do sistema e (iii) a maneira como as ações de cada um dos usuários interagem [9]. Em um sistema colaborativo, diferentemente de uma aplicação monousuária, várias pessoas podem agir ao mesmo tempo sobre os mesmos artefatos de trabalho. Nos ambientes compartilhados, os atores não têm conhecimento das ações dos outros usuários. Assim, a consciência das atividades individuais e das atividades de grupo se torna uma informação crítica para tornar possível o sucesso da colaboração. Segundo *Gutwin e Greenberg*[10], a telepresença apresenta as seguintes vantagens:

- redução do esforço de coordenação;
- antecipação das ações dos outros usuários;
- ajuda às pessoas a se integrarem nas atividades.

Segundo *Dix* [11], a noção de telepresença se debruça sobre três formas:

- consciência da presença dos membros do grupo e sua disponibilidade no trabalho cooperativo;
- consciência das ações realizadas pelos membros do grupo;
- consciência dos efeitos consecutivos às suas ações.

Assim, a telepresença, sob suas diferentes formas, permite controlar os comportamentos de cada participante. Para poder trabalhar em conjunto, é indispensável estar consciente da presença dos outros participantes e das suas ações.

A Sessão Colaborativa e os Artefatos de Trabalho

O conceito de Sessão é geralmente utilizado em TCSC para estruturar e organizar o trabalho em grupo. Em [12], *Haake et al.* definem uma sessão por:

- grupo de usuários;
- espaço de trabalho comum no qual os usuários agem e manipulam artefatos de trabalho;
- um modo de cooperação específico explorado por esses usuários.

Assim, a aplicação colaborativa deve fornecer operadores para gerenciar uma sessão, tais como criar e destruir uma sessão; participar e deixar uma sessão; selecionar ou modificar os modos de cooperação, assim como os modos de manipulação dos artefatos de trabalho.

A manipulação dos artefatos evoca a manipulação de objetos físicos que podem ser deslocados ou modificados em um ambiente de trabalho real. Este conceito visa determinar como os sistemas colaborativos implementam o compartilhamento de artefatos. O conceito de "Espaço Compartilhado" corresponde a uma das noções mais difundidas na organização de artefatos de trabalho, tais como documentos e ferramentas compartilhadas necessárias às tarefas a serem executadas pelo grupo [9][10][13].

Certas aplicações colaborativas devem oferecer funcionalidades para controlar o acesso concorrente aos artefatos de trabalho compartilhado. Frequentemente este controle é implementado por um mecanismo de passagem de mão, ou do inglês *floor control* [14] que permite a usuários controlarem uma aplicação compartilhada.

2.1.2 Classificação das Aplicações Colaborativas

O domínio TCSC apresenta uma grande diversidade de aplicações, o que torna difícil definir uma taxonomia unificada para estas aplicações. As duas classificações

mais importantes encontradas na literatura são as seguintes: a classificação espaço-tempo - diz respeito às características temporais e espaciais, e a categoria funcional da aplicação.

A Classificação Espaço-Tempo

O espaço e o tempo constituem as dimensões mais comuns na classificação de aplicações colaborativas [6]. O espaço diz respeito à distância geográfica separando os usuários da aplicação. Por exemplo, os membros de uma reunião podem estar situados na mesma sala ou lugares diferentes. A dimensão temporal caracteriza o tipo de interação entre os usuários. Os membros de um grupo podem interagir simultaneamente, o que define uma colaboração síncrona. Nestes casos, o problema de sincronização e gestão da concorrência estão presentes. Essa categoria de problema foi bastante estudada no domínio do TCSC, onde problemas de ordem humana se agregam a problemas de ordem técnica [15][16].

Além da colaboração síncrona, os membros de um grupo podem igualmente agir em instantes diferentes, ou seja, as ações são espaçadas no tempo. Trata-se então de uma colaboração assíncrona. Neste caso é importante que o estado da atividade seja conservado permanentemente, para que os membros do grupo sejam capazes de observar o histórico das ações que foram efetuadas antes de sua chegada.

Categorias Funcionais das Aplicações TCSC

As aplicações colaborativas são frequentemente classificadas por domínio de aplicação ou por categorias funcionais [17][18]. A seguir é apresentada uma lista não exaustiva das categorias de aplicações colaborativas mais referenciadas:

- Correio eletrônico - esta categoria de sistema colaborativo é o meio de co-

municação assíncrono mais difundido atualmente; o correio fornece o serviço de transporte de mensagens via Internet para a caixa de mensagens do destinatário escolhido pelo emissor;

- Mensagem instantânea e fórum de discussão - assim como o correio eletrônico, estes dois sistemas são dedicados à comunicação por troca de mensagens. No primeiro caso, contrariamente ao correio eletrônico, a comunicação foi concebida para ser instantânea, ou seja, síncrona; o segundo tipo de sistema representa os lugares (normalmente páginas Web) onde os indivíduos podem compartilhar suas opiniões por troca de mensagens.

- Sistema de auxílio à tomada de decisão - esses sistemas foram concebidos para facilitar a tomada de decisão implementando uma espécie de sala de reunião eletrônica, oferecendo uma série de aplicativos (por exemplo, urna de votação e blocos de notas)[19]; o anonimato e o direito à palavra são mecanismos fornecidos ao usuário para encorajá-lo a se engajar no processo de tomada de decisão.

- Ferramenta de compartilhamento de aplicação - este tipo de software permite a vários usuários (trabalhando em computadores diferentes) utilizar simultâneamente uma aplicação hospedada em um Servidor. Esta funcionalidade é frequentemente implementada exportando a janela compartilhada para as máquinas dos outros usuários.

- Editor compartilhado - estes sistemas de edição conjunta permitem a um grupo de usuários editar coletivamente um documento compartilhado; eles podem ser utilizados de maneira síncrona ou assíncrona; mas o principal problema desses

sistemas diz respeito ao gerenciamento de tarefas concorrentes [20], quando dois ou mais usuários modificam o mesmo documento simultâneamente.

- Quadro branco compartilhado - este tipo de sistema coloca a disposição dos usuários uma superfície de desenho acessível para vários usuários, permitindo assim de trabalhar de maneira síncrona sobre documentos 2D.

- Áudioconferência - as ferramentas de áudio conferência permitem aos usuários falar a várias pessoas simultâneamente; a qualidade da transmissão desempenha um papel fundamental para a compreensão da comunicação, além de não consumir uma grande banda passante; o áudio é um dos meios de comunicação mais ricos, mas a utilização de áudio conferência com vários participantes como o único meio de comunicação pode apresentar dificuldades na identificação dos interlocutores [21].

- Plataformas colaborativas - esta categoria especial de sistemas colaborativos agrupam em um mesmo ambiente diferentes ferramentas colaborativas associadas a várias categorias.

- Ambientes virtuais colaborativos (AVCs) - ou do inglês, *CVE*, representam uma importante categoria de sistemas TCSC e oferecem facilidades de colaboração por intermédio da implementação de um espaço virtual compartilhado; frequentemente o espaço virtual é baseado em ricas cenas 3D compartilhadas; a utilização da realidade virtual é encorajada graças à sua capacidade importante de modelagem e interatividade, permitindo aos AVCs resolver vários problemas concernentes às aplicações TCSC: (i) a noção de presença; (ii) a representação das metáforas do mundo real (gestos humanos); (iii) custo de

transmissão através da rede reduzido, sobretudo quando comparados com os sistemas de vídeoconferência.

Ambientes virtuais colaborativos são o principal tema deste trabalho de Tese e será discutido com profundidade na sequência deste capítulo.

2.2 Ambientes Virtuais Colaborativos

Um ambiente virtual é um ambiente 2D/3D que representa um ambiente real ou um ambiente imaginário com o qual o usuário pode interagir em tempo real.

Ambientes 3D que suportam vários usuários conectados simultâneamente são chamados de Ambientes Virtuais Multiusuários, do Inglês, *MUVE* [22].

Ambientes Virtuais Colaborativos (AVCs) são espaços virtuais multiusuários (*MUVEs*) onde pessoas distantes geograficamente podem se encontrar, interagir e colaborar por meio de agentes e objetos virtuais [23].

Nos AVCs, os usuários são representados por avatares que habitam o mundo virtual. A palavra *avatar* é originária do termo Sânscrito [1] *avatara* que designa cada uma das diferentes incarnações do Deus Hindu *Visnu* na Terra.

A utilização de avatares reforça a sensação de presença de todos os usuários que estão simultâneamente conectados ao mesmo ambiente virtual. A possibilidade do avatar realizar algumas ações básicas, tais como gestos e movimentos aumenta a sensação de presença, de espaço compartilhado e de tempo [24].

No que diz respeito à interação entre usuário e ambiente virtual, estes podem ser

[1] (em devanagari, pronuncia-se saṃskṛtā), é uma língua clássica da Índia, uma língua litúrgica do Hinduísmo, Budismo, Jainísmo, e uma das 23 línguas oficiais da Índia, influenciou praticamente todos os idiomas ocidentais. fonte: wikipedia, Novembro de 2008.

classificados em duas categorias:

- Imersivo: são baseados no uso de equipamentos sofisticados como luvas, capacetes, óculos, cavernas e seus acessórios;
- Não-Imersivo: são baseados em equipamentos desktop, como computadores pessoais e notebooks. A imersão do usuário é apenas parcial. Os ambientes não-imersivos são os mais populares por serem mais baratos e simples de implementar.

Tipos de interação entre usuários definidas em um AVC:

- interação multi-modal entre usuários: por meio de *chat*, comunicação verbal e não verbal (gestos). Este último tipo é suportado por manipulação de objetos 3D compartilhados.
- interação usuário-sistema: comandos que o sistema oferece ao usuário para ajudá-lo na navegação, assim como na realização de uma tarefa específica por meio da manipulação de um objeto 3D compartilhado.

A seguir são apresentadas algumas arquiteturas de comunicação utilizadas nos Ambientes Virtuais Colaborativos.

2.2.1 Modelos de Comunicação para Ambientes Virtuais Colaborativos

Arquitetura Cliente-Servidor Centralizado

Neste tipo de arquitetura, a base de dados representando o mundo virtual é inteiramente gerenciada por um único servidor. Cada máquina cliente envia ao servidor

central as mensagens de atualização (deslocamento, comunicação textual, manipulação de objetos, etc) provenientes dos seus usuários, que estão conectados ao mesmo ambiente virtual. O servidor por sua vez efetua as mudanças ocorridas na sua base de dados, devolvendo para cada um dos usuários conectados o resultado das alterações feitas sobre o ambiente virtual.

Essa arquitetura tem sua topologia em forma de estrela, onde cada usuário está conectado ao computador central. O modelo de comunicação centralizado apresenta facilidade de implementação da estrutura de controle da comunicação entre os usuários. No entanto, no quesito escalabilidade, esse modelo deixa a desejar. Pois a medida que aumenta a quantidade de usuários conectados simultâneamente, a quantidade de mensagens de atualização também aumenta e o desempenho do sistema cai drasticamente [25].

Arquitetura Cliente-Servidor Distribuído

Este modelo de arquitetura é semelhante ao anterior, com uma diferença: a base de dados representando o mundo virtual é distribuída entre os clientes e apenas as mensagens trocadas entre os clientes são gerenciadas pelo servidor.

Em geral, o servidor gerencia diversos filtros para evitar que todas as mensagens que ele recebe não sejam difundidas a todos os outros clientes. Os filtros são em geral espaciais: a mensagem de um cliente é enviada a todos os clientes cujo o avatar (entidade gráfica que representa o usuário) está próximo no mundo virtual.

No caso de mundos virtuais de grande porte e dinâmicos, o servidor pode rapidamente se tornar um ponto de estrangulamento do sistema.

Arquitetura Cliente-Servidor Distribuído com vários Servidores

Para evitar o problema de estrangulamento da rede com um único servidor, foi realizada uma modificação na arquitetura anterior, utilizando agora vários servidores, organizados de diversas formas.

A mais frequentemente encontrada se trata de uma organização funcional: cada servidor se ocupando de um tipo de comunicação. Assim é possível ter, por exemplo, um servidor para gerenciar as colisões e outro para gerenciar as interações. Neste caso, continua a existir um ponto de estrangulamento do sistema, caso várias entidades interajam simultâneamente.

Para evitar este problema, é possível hierarquisar servidores se ocupando de um mesmo tipo de comunicação. Cada cliente se comunica diretamente com o servidor mais próximo (em termos de distância na rede), que envia as mensagens de atualização aos outros servidores

Peer-to-Peer

Neste tipo de arquitetura, todos os computadores desempenham o mesmo papel. A base de dados representando o mundo virtual é replicada nos diferentes sítios e todas as modificações locais são comunicadas a todas as outras máquinas.

O principal problema nesta arquitetura, do ponto de vista da escalabilidade, é que o número de mensagens de atualização aumenta muito com o número de usuários. Se cada usuário enviar uma mensagem de atualização, a cada ciclo de simulação, n*(n-1) mensagens serão enviadas através da rede. Esse problema pode ser minimizado utilizando-se a técnica do *dead-reckoning* [25]. Outro problema com a arquitetura *Peer-to-Peer* é manter a coerência das cópias da base de dados representando o

mundo virtual. Para evitar que a mesma parte da base de dados seja modificada simultâneamente por dois ou mais usuários, um mecanismo de proteção deve ser implementado.

Um mecanismo simples para gerenciar o acesso ao objeto distribuído é criar um *Token* (mensagem), que transita entre as máquinas clientes, e a máquina que receber o *Token* tem o direito de modificar a base de dados distribuída. Outro mecanismo mais complexo, com o mesmo objetivo, prevê que todos os clientes façam as modificações desejadas, registrando a data da modificação. Assim, caso haja algum conflito, a última modificação é considerada e as outras são anuladas.

A seguir é apresentada uma série de AVCs descritos na literatura.

2.3 Trabalhos Relacionados

Ambientes Virtuais Colaborativos (AVCs) têm sido usados em diversos tipos de aplicação, tais como: entretenimento, *e-learning*, treinamento, simulações de guerra, etc. São encontrados também na literatura Ambientes Virtuais baseados no paradigma de Agentes. Tais ambientes são comumente chamados de Ambientes Virtuais Inteligentes (*AVI*)[26]. Um AVI é um ambiente virtual habitado por entidades inteligentes autônomas exibindo uma variedade de comportamentos [27]. Os ambientes virtuais inteligentes encontram uma ampla gama de aplicações: simulação, treinamento, entretenimento, jogos, educação, etc.

Esta seção apresenta uma revisão no que diz respeito à pesquisa em ambientes virtuais colaborativos. Os critérios utilizados para comparação entre os AVCs são os seguintes:

- capacidade de representar a dinâmica do trabalho colaborativo;

- capacidade de gerenciar as aplicações colaborativas;
- capacidade de representar os direitos de acesso dos usuários;
- gerenciamento de papéis (*roles*);
- integração aberta do AVC a outras ferramentas colaborativas;
- tecnologia utilizada.

A seguir uma breve descrição dos AVCs estudados, categorizados em ambientes não-baseados no paradigma de agentes e baseados em agentes.

2.3.1 Ambientes Virtuais não-baseados em Agentes

DIVE

DIVE [28] é um sistema de realidade virtual desktop, onde os usuários representados por seus avatares podem navegar nos mundos virtuais, dinamicamente programar comportamentos para os objetos e avatares. *DIVE* oferece ainda uma integração restrita com outras aplicações colaborativas.

São permitidas às aplicações externas, executando como uma aplicação *standa-lone*, se conectarem através da interface cliente de *DIVE*. Neste caso, a aplicação *DIVE* se comporta como um servidor e a aplicação externa como um cliente padrão TCP. A aplicação externa será representada no ambiente por uma entidade de um tipo específico, e esta desempenhará seu comportamento no ambiente. A linguagem utilizada é o *DIVE/TCL*.

DIVE suporta também transmissão de áudio e vídeo utilizando os protocolos do *MBONE: VAT e VIC* [29]. A Figura 2.1 ilustra exemplos de mundos virtuais desenvolvidos no sistema *DIVE*.

Figura 2.1: Exemplo de cena virtual no sistema DIVE.

EVE

EVE [30] é um ambiente virtual colaborativo voltado para Ensino a Distância e baseado em *VRML*. A Figura 2.2 mostra uma sala de aula virtual em *EVE*. Professor e alunos, dois perfis diferentes, são representados por avatares capazes de realizar gestos expressando opiniões (concordando ou discordando), assim como sentimentos (alegria ou tristeza). *EVE* oferece ainda comunicação por *chat* ou áudio.

INVITE

INVITE[2] foi um projeto Europeu da IST (*Information Society Technology*), programa da (*European Comission*), iniciado em Fevereiro de 2000, com duração de quase três anos, com a participação de nove instituições de quatro países Europeus. O principal objetivo deste projeto foi construir uma plataforma para teleaprendizado síncrono. [31].

INVITE oferece entre outros recursos:

[2]http://invite.fh-joanneum.at

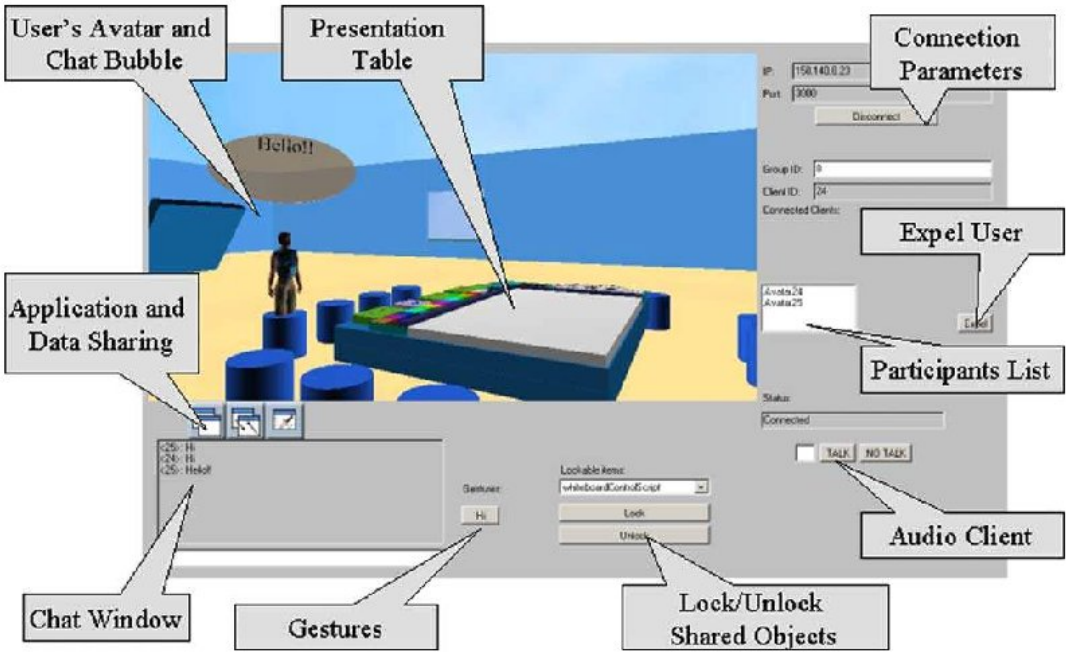

Figura 2.2: Sala de reunião em EVE.

- representação dos usuários por avatares com aparência realística, chamados de avatares foto-realísticos. Avatares foto-realísticos são mais efetivos quando usados em um ambiente colaborativo oferecendo interação multi-modal entre os usuários por meio de gestos, assim como movimentos em tempo-real [32];
- integração do ambiente 3D com vídeos;
- fornece comunicação multi-modal por meio de áudio e texto.

Entretanto, *INVITE* não considera os direitos de acesso dos usuários.

Programa de Ottawa

O *Programa de Ottawa* [33] é um ambiente virtual voltado para treinamento industrial. Neste ambiente, os usuários, representados por seus respectivos avatares,

aprendem a substituir uma placa defeituosa em um *Switch ATM*, como ilustrado na Figura 2.3. As ações executadas pelo avatar são exatamente as mesmas que seriam executadas no mundo real.

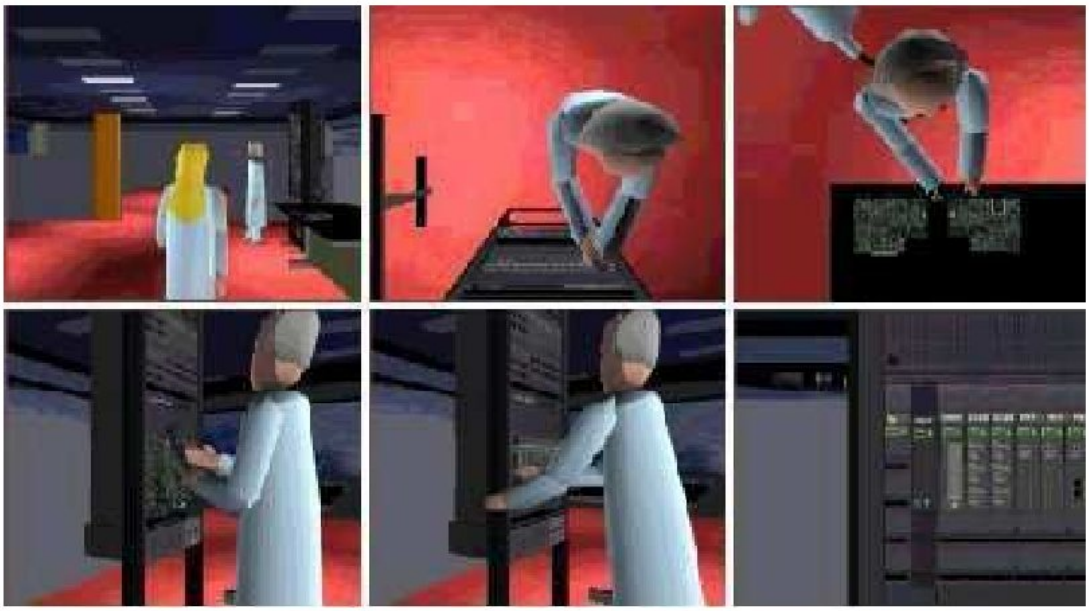

Figura 2.3: Exemplo de cena virtual no Programa de Ottawa.

ADVICE

O sistema *ADVICE* [34], desenvolvido a partir do *Programa de Ottawa* [33], é um ambiente virtual colaborativo voltado para Ensino a Distância. *ADVICE* oferece uma sala de aula 3D, onde o professor e os alunos, representados por seus avatares sentados em suas respectivas cadeiras, conversam por meio de um chat, como ilustrado na Figura 2.4.

Figura 2.4: Exemplo de cena virtual no sistema ADVICE.

NETICE

NETICE [35] permite a usuários compartilhar uma aplicação de *white-board* a partir de uma interface 3D e oferece usuários com um conjunto de expressões corporais e faciais. Mas, as ações no ambiente são realizadas a partir de um menu localizado na interface, independente das ações realizadas no *whiteboard* e não são realizadas pelos avatares, o que diminue consideravelmente a percepção dos usuários sobre quem está realizando as ações.

Soares et al. [36], propõe uma interface híbrida 2D-3D. As ações dos usuários em aplicações 2D são representadas em um ambiente virtual 3D, como ilustrado na Figura 2.5.

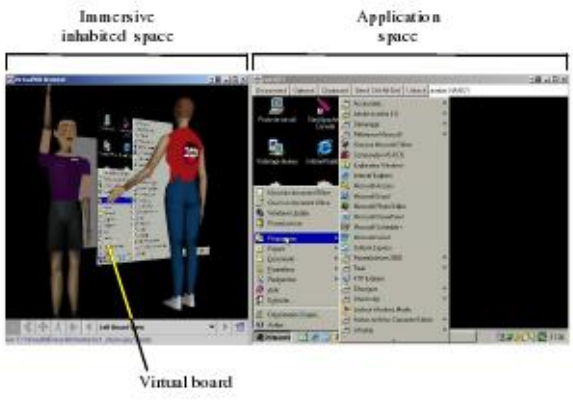

Figura 2.5: Interface 2D representada em um ambiente 3D.

VREng

VREng [37] é um sistema de realidade virtual distribuída que permite aos seus usuários interagir e navegar em mundos virtuais 3D. O sistema é dividido em três módulos principais: interface de usuário, representação 3D e um terceiro módulo que trata da gerência do mundo virtual. O primeiro módulo é o mais importante deles, pois além de ser responsável por oferecer ao usuário uma interface que lhe permita se movimentar dentro do ambiente virtual, enviar mensagens aos outros usuários e interagir com o ambiente virtual de maneira intuitiva, ele se encarrega da comunicação entre os módulos existentes. O segundo módulo determina como os objetos serão mostrados em três dimensões, segundo sua forma, cor, textura, etc. Por fim, o módulo responsável pela gerência do mundo virtual define os objetos que irão compor uma determinada cena virtual, gerencia animações e as interações entre os objetos. Uma lista não exaustiva de objetos é oferecida sobre os quais é possível a execução de ações adequadas pelos usuários. O conceito de *role* e nenhum esquema de gerenciamento dos direitos dos usuários é oferecido. *VREng* oferece uma integração ad-hoc do seu ambiente virtual 3D com outras aplicações colaborativas. Além disso, as ações dos usuários não são representadas por ações dos avatares,

o que prejudica a percepção dos demais usuários sobre quem está realizando uma determinada ação.

There e Second Life

There [38] oferece aos seus usuários avatares bastante ricos em detalhes, avançados e extremamente customizáveis. Usuários podem escolher entre muitos ambientes 3D para navegar e interagir com outros usuários por meio de um sistema de chat.

Second Life[39] surge como um novo conceito de ambientes virtuais, onde os usuários podem, além de navegar e interagir com outros usuários: trabalhar, comprar e vender, construir, etc. As transações comerciais realizadas nos mundos virtuais em Second Life utilizam uma moeda própria, denominada *Linden Dolar*, que podem ser convertidos em dólares verdadeiros.

No entanto, as possibilidades de interação entre avatares e objetos são limitadas a um mesmo conjunto de ações predefinidas.

2.3.2 Ambientes Virtuais baseados em Agentes

STEVE

Em [40] foi proposto um ambiente virtual voltado para o treinamento de procedimentos em operações navais. Neste ambiente um agente chamado *STEVE* explica aos usuários o funcionamento dos equipamentos, além de demonstrar as tarefas a serem realizadas e explicar suas ações, como ilustrado na Figura 2.6.

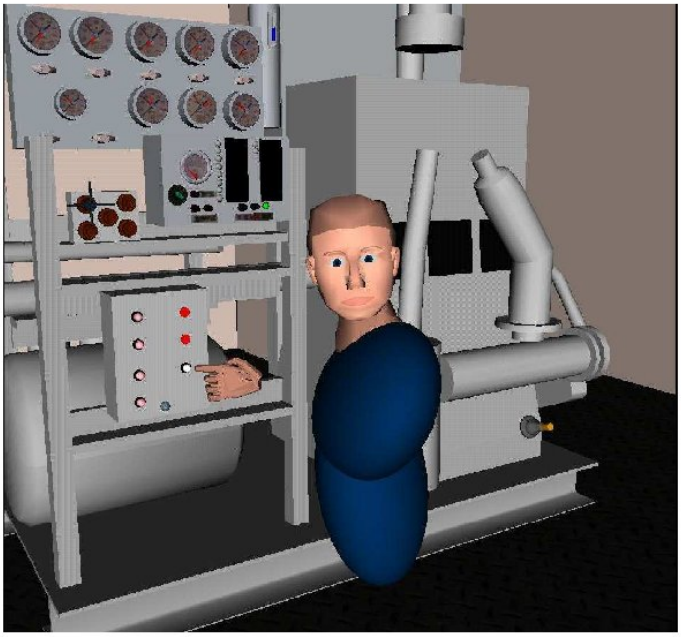

Figura 2.6: O Agente STEVE explica aos usuários o funcionamento de um painel de controle.

ACTIVE WORLDS

Active Worlds [41] consiste em um conjunto de ambientes 3D, onde o usuário, representado por seu avatar pode navegar e interagir com outros usuários. As Figuras 2.7 e 2.8 mostram exemplos de mundos virtuais no *Active Worlds*.

Figura 2.7: Exemplo de cena no virtual no Active Worlds 1

Figura 2.8: Exemplo de cena no virtual no Active Worlds 2

VITAL

Em [27], *Anastassakis et al.* apresentam VITAL, um sistema multiagentes inteligente capaz de suportar ambientes virtuais inteligentes de propósito geral. Um exemplo da aplicação de VITAL é mostrado nas Figuras 2.9 e 2.10.

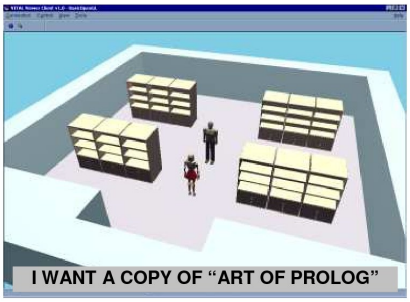

Figura 2.9: O agente cliente requisita uma cópia de "Art of PROLOG" de um agente bibliotecário.

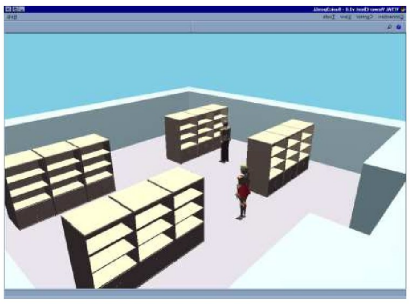

Figura 2.10: O agente bibliotecário está buscando o exemplar requisitado enquanto o cliente espera.

RIZZO ET AL.

Em [42] e [43], *Rizzo et al.* apresentam ambientes virtuais onde as crianças e o professor representados por avatares navegam pelo ambiente. A interação entre as crianças e o ambiente é monitorada e os dados coletados são utilizados para traçar um perfil comportamental de cada criança, servindo como auxílio no tratamento de crianças com hiperatividade e défict de atenção.

2.4 Posicionamento deste trabalho de Tese

A partir da comparação dos AVCs relacionados, concluímos que existem diversos tipos de ambientes virtuais, englobando diferentes tecnologias e capazes de repre-

Figura 2.11: Crianças em uma sala de aula virtual.

Figura 2.12: Crianças em uma sala de aula virtual.

sentar uma ampla gama de aplicações. No entanto, nenhum dos ambientes virtuais encontrados na literatura oferece aos usuários mecanismos para representar qualquer tipo de atividade colaborativa, incluindo seus protocolos sociais.

A Tabela 2.1 retrata uma comparação entre os ambientes virtuais estudados. No que diz respeito ao nível de representação das ações dos usuários dos ambientes descritos na literatura, eles foram classificados da seguinte forma:

- alta capacidade de representação: as ações dos usuários são representadas por ações dos avatares respeitando a funcionalidade dos objetos presentes na cena virtual;
- média capacidade de representação: apenas um subconjunto das ações dos usuários são representadas por ações dos avatares;
- baixa capacidade de representação: as ações dos usuários não são representadas por ações dos avatares.

A maioria dos ambientes estudados oferece algum nível de integração com aplicações externas, no entanto, essa integração é fechada, limitando assim a colaboração àquelas ferramentas já integradas.

Nenhuma das ferramentas estudadas oferece mecanismos para gerenciar as aplicações externas, deixando assim de explorar um importante potencial dos ambientes virtuais no controle das aplicações externas.

Outro aspecto a ser observado, com base na Tabela comparativa 2.1, é que não houve, por parte dos projetistas de ambientes virtuais, uma preocupação em representar na cena virtual os direitos de acesso dos usuários enquanto participantes de uma sessão colaborativa.

Esta Tese apresenta *RECOLLVE*, um ambiente virtual voltado para a representação de atividades colaborativas, fornecendo todos os mecanismos necessários para representar na cena virtual uma sessão colaborativa do mundo real.

O próximo Capítulo apresenta as tecnologias utilizadas na concepção, modelagem e implementação de RECOLLVE.

Tabela 2.1: Quadro comparativo entre os ambientes virtuais estudados.

	Integração	Representação	Direitos de Acesso	Gerenciamento	Baseado em Agentes	Role
INVITE	*Fechada*	*Alta*	*Nao*	*Nao*	*Nao*	*Sim*
DIVE	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
VREng	*Fechada*	*Baixa*	*Nao*	*Nao*	*Nao*	*Nao*
Netice	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
EVE	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
Programa de Ottawa	*Fechada*	*Alta*	*Nao*	*Nao*	*Nao*	*Sim*
There	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
Second Life	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
Soares et al	*Fechada*	*Alta*	*Nao*	*Nao*	*Nao*	*Nao*
STEVE	*Fechada*	*Media*	*Nao*	*Nao*	*Sim*	*Sim*
VITAL	*Fechada*	*Media*	*Nao*	*Nao*	*Sim*	*Sim*
Rizzo et al.	*Fechada*	*Baixa*	*Nao*	*Nao*	*Sim*	*Sim*
Active Worlds	*Fechada*	*Alta*	*Nao*	*Nao*	*Sim*	*Nao*

Capítulo 3

Infraestrutura utilizada na concepção, modelagem e implementação de RECOLLVE

Sistemas de agentes inteligentes têm sido um tema sujeito a intensas pesquisas ao longo dos anos. Esses sistemas compreendem um dos campos de pesquisa mais promissores da computação, pois são capazes de tratar aspectos que demandam uma modelagem abstrata e um alto nível de raciocínio. Ambientes virtuais por sua vez, oferecem um meio ideal para produzir simulações do mundo real para propósito de entretenimento, educação, entre outros. A mistura desses dois campos de pesquisa tem muito a oferecer à pesquisa e ao desenvolvimento de aplicações.

Este Capítulo descreve a infraestrutura e as tecnologias utilizadas na concepção, modelagem e implementação do sistema RECOLLVE: um sistema de realidade virtual desktop baseado no paradigma de agentes inteligentes.

3.1 Sistemas Multiagentes

Russel e Norvig [44] definem um agente como um sistema capaz de perceber as informações do ambiente onde está inserido, por meio dos sensores, e de reagir por meio de atuadores.

Os Sistemas Multiagentes (*SMA*) caracterizam-se pela existência de dois ou mais agentes que se comunicam e interagem entre sí de forma autônoma para resolver problemas, executar uma tarefa, ou cumprir um objetivo comum.

O principal mecanismo de comunicação entre agentes é por meio de troca de mensagens. Um agente informa suas intenções e necessidades a outros agentes por meio de troca de mensagens assíncronas. A *FIPA* [45] com o objetivo de padronizar os protocolos de comunicação entre agentes, definiu um conjunto de primitivas de comunicação para a linguagem de comunicação *ACL* [46].

A plataforma *JADE* [47], utilizada na simulação da arquitetura RECOLLVE, oferece facilidades para o desenvolvimento de sistemas multiagentes que auxiliam o desenvolvedor na implementação de uma plataforma eficiente de agentes. Com o uso de JADE, a comunicação, a troca de mensagens e outros atributos que um sistema multiagentes necessita são gerados automaticamente. Além disso, JADE oferece uma variedade de ferramentas de monitoração, gerenciamento e depuração que ajudam tanto no desenvolvimento, quanto na manutenção e suporte a sistemas multiagentes. JADE mantém os padrões especificados pela FIPA, aumentando o grau de integração e escalabilidade do ambiente em relação a outros SMA.

3.2 Representação 3D da Cena Virtual

3.2.1 VRML, VIP, X3D e H-Anim

VRML

VRML é uma linguagem de descrição de conteúdo 3D e foi o primeiro padrão adotado para a descrição de cenas virtuais.

Uma cena *VRML* é organizada sob a forma de um grafo de cena hierárquico direto acíclico. Os nós são os componentes básicos deste grafo de cena. Um nó é um conjunto de especificações que determinam as características dos objetos contidos na cena virtual, definindo a hierarquia e as características individuais de cada objeto.

Cada nó é capaz de emitir e de receber eventos através dos campos *eventOut* e *eventIn*, respectivamente. *VRML* oferece nós do tipo sensores que permitem capturar alguns tipos de eventos ocorridos na cena. Um nó sensor do tipo *TouchSensor* poderá, por exemplo, emitir um evento do tipo *SFTime* quando receber um clique de mouse. Este evento traduz o instante preciso do clique. Um outro nó, ao receber este evento pode disparar uma ação qualquer.

A transmissão de um evento de um nó a outro, se dá através do mecanismo de rotas. O mecanismo de rotas permite definir a transmissão de um evento de um campo *eventOut* de um nó em direção a um campo *eventIn* de um outro nó. Uma rota é definida da seguinte maneira:

```
ROUTE nó1.eventOut1 TO nó2.eventIn2.
```

Estas rotas são a base do sistema de eventos de VRML. *VMRL* oferece um mecanismo que permite que o conjunto original de tipos de nós seja estendido. A instrução *PROTO* permite criar seus próprios nós, definindo um protótipo de objetos

parametrizáveis. Estes nós podem conter formas geométricas, sensores, scripts, etc. Os nós de script permitem obter formas mais complexas de interação. Os nós de scripts são incluídos no grafo de cena e podem conter código compilado Java ou linguagens de script como Javascript ou ECMAScript [48], que interagem com o conteúdo *VRML* através da BSI [49]. Esta interface permite aos scripts, entre outras coisas: localizar nós definidos explicitamente, modificar rotas existentes, ou ainda, modificar o conteúdo do navegador *VRML*, adicionando ou removendo nós-filhos de um nó de agrupamento, por exemplo.

Para a visualização de cenas *VRML*, normalmente utiliza-se browsers como o *Internet Explorer* ou *Mozilla*, enriquecidos com plugins, tais como o *Cortona*[1] ou *Blaxxun Contact*[2] que fazem a renderização da cena virtual.

O usuário interage com a cena virtual através da *EAI*. A *EAI* foi desenvolvida pelo grupo de trabalho de mesmo nome do *Web3D Consortium* [50] e funciona como interface entre a cena virtual (*VRML*) e o mundo exterior.

Graças à EAI, é possível se comunicar com a cena *VRML* exatamente da mesma maneira que um nó de script declarado no interior do mesmo grafo de cena. Assim, é possível recuperar eventos produzidos por um nó, e de enviar este evento a um outro nó qualquer.

A *EAI* se apresenta sob a forma de uma API Java. Ela necessita da presença de um applet na mesma página do navegador *VRML*. O applet se conecta ao browser *VRML*, e assim pode-se recuperar uma referência de qualquer nó do grafo de cena, à condição de que ele tenha sido definido. Definir um nó em *VRML* significa lhe dar um nome único graças ao comando "DEF".

[1] http://www.parallelgraphics.com

[2] http://www.blaxxun.com

A interface *Browser* oferece um conjunto de métodos que permite comandar totalmente o grafo de cena do navegador:

- métodos para manipular o grafo de cena, como: adicionar e retirar nós, carregar outra cena, etc;
- métodos para acessar os nós contidos no grafo de cena: o programador pode obter valores do campo *eventOut* e escrever valores no campo *eventIn*

Mundos VRML Multiusuários

Do ponto de vista técnico, *VRML2.0*[51] é o padrão tecnológico para a criação de ambientes virtuais. *VRML* é suportada por vários editores e ferramentas de edição 3D como *VRMLPad*[3] e *X3DEdit*[4]. No entanto, *VRML* não fornece suporte a mundos virtuais multiusuários. O modelo *VRML* foi definido para satisfazer apenas um único grafo de cena. O que para a implementação de RECOLLVE representa um problema, pois no nosso caso, cada usuário tem seu próprio grafo de cena não compartilhado, executando em um plug-in *VRML* em diferentes máquinas.

Como já descrito na Seção 3.2.1, uma cena VRML é organizada como um grafo direto acíclico, onde cada nó é capaz de produzir e receber eventos. Por exemplo, um nó do tipo *TouchSensor* gera um evento do tipo *SFTime* sempre que um usuário usa o mouse para clicar sobre um objeto. Este evento captura o exato instante do clic do mouse. Outro nó pode receber este evento como um *EventIn* e iniciar uma ação.

A fim de apontar as incoerências da propagação de eventos em mundos VRML multiusuários, será usado como exemplo um mundo VRML multiusuário contendo

[3]http://www.parallelgraphics.com/products/vrmlpad/

[4]http://www.web3d.org/x3d/content/README.X3D-Edit.html

dois avatares, representando dois usuários, e um objeto representando uma cadeira. Quando um usuário seleciona a cadeira com o mouse, o nó *TouchSensor* da cadeira produzirá um *eventOut*, que será roteado para o *eventIn* do avatar. Assim, a animação correspondente deveria iniciar. O cenário desejado é ilustrado na Figura 3.1. No entanto, a situação real é ilustrada na Figura 3.2, pois o mecanismo de roteamento de eventos não é capaz fazer diferença entre os dois avatares A e B, e um evento aleatório ocorrerá: ou os dois avatares ou nenhum avatar tentará se sentar na cadeira.

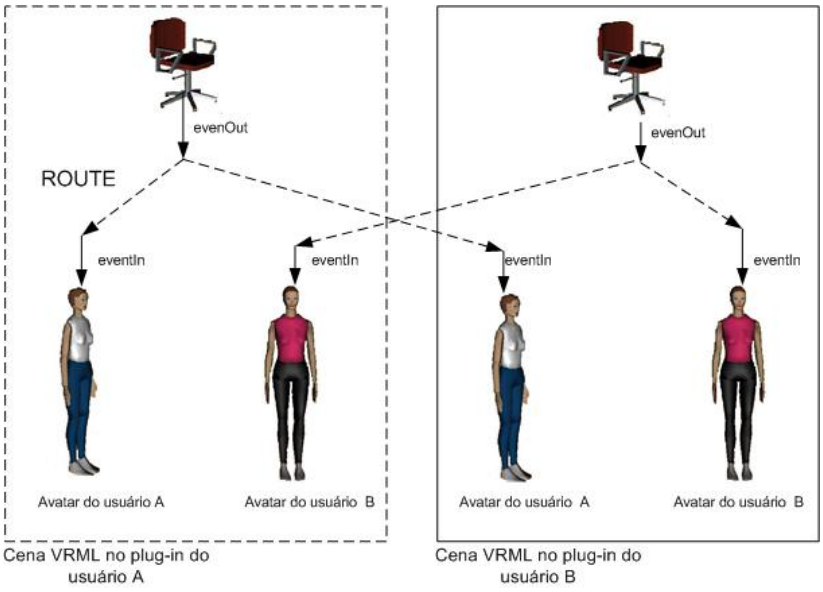

Figura 3.1: Sincronização de eventos - situação desejada.

Assim, conclui-se que o mecanismo de roteamento de VRML não pode ser utilizado para o caso multiusuário. Existe uma necessidade, portanto, de um mecanismo para tornar possível a utilização de VRML em mundos virtuais multiusuários. Para esse fim, várias abordagens, descritas a seguir, foram utilizadas.

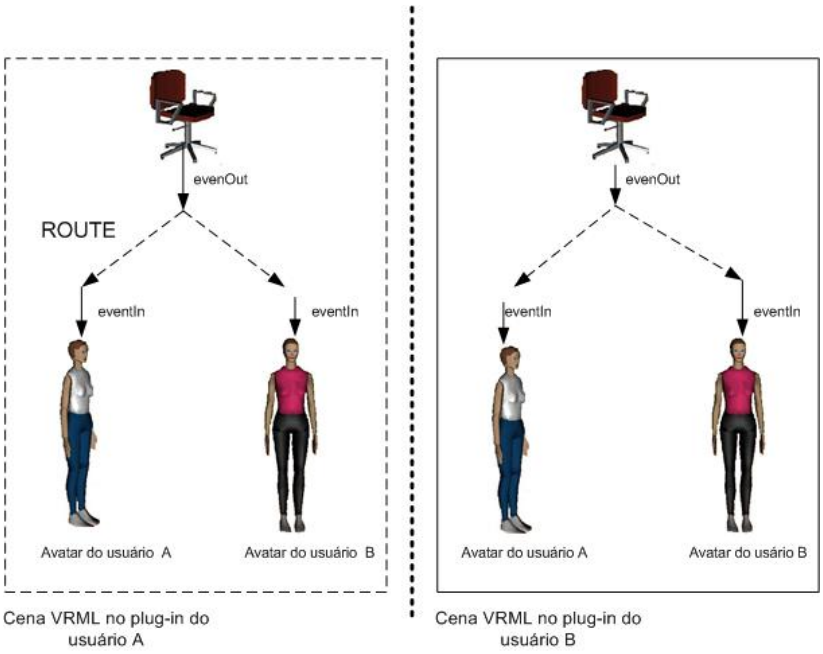

Figura 3.2: Sincronização de eventos - situação real.

Ambientes Virtuais Multiusuários baseados em VRML

Como já foi dito anteriormente, o padrão VRML não é apropriado para a construção de ambientes virtuais multiusuários por conta da limitação no seu mecanismo de tratamento de eventos. Na busca por um padrão para ambientes virtuais multiusuários baseados em *VRML97*, o grupo de trabalho do *Web3D Consortium*, LW [52] fez a primeira tentativa de padronização. *LW* define uma infraestrutura propondo um conjunto de novos nós permitindo a descrição e o suporte de mundos virtuais multiusuários utilizando o padrão *VRML97*. Ele utiliza os diferentes mecanismos propostos pelo padrão *VRML97*, como protótipos, scripts, applets JAVA e a *EAI* [53], sendo, portanto, inteiramente compatível com os browsers que respeitam o padrão *VRML97*.

Um mundo multiusuário *LW* é subdividido em uma ou mais *Zones*. Uma *Zone* permite reagrupar os diferentes elementos da cena *VRML97* que devem ser compartilhados: formas geométricas, dados, etc. *Zones* não precisam necessariamente

serem associadas a espaços dentro do mundo virtual, mas esta é uma boa forma de utilizá-las. Uma *Zone* contém um ou vários *SharedObjects*. Cada *SharedObject* representa uma entidade cujo estado deve ser compartilhado entre os diferentes usuários representados no mundo virtual.

Todo *SharedObject* pode ter dois estados: *Pilot* ou *Drone*. Em um dado instante e para um dado *SharedObject*, existe somente um *Pilot* e vários *Drones*; o *Pilot*, cópia do *SharedObject* que sofre modificação, deve comunicar toda mudança de estado aos *Drones*. Um *SharedObject* contém vários valores de estado que devem ser sincronizados entre os usuários representados no mundo virtual.

Na infraestrutura *LW* cada estado compartilhado é chamado *NetworkedState*: para cada tipo de base *VRML97* (*SFBool, SFColor, etc*), *LW* define um nó equivalente compartilhado (*NetworkSFBool, NetworkSFColor, etc*).

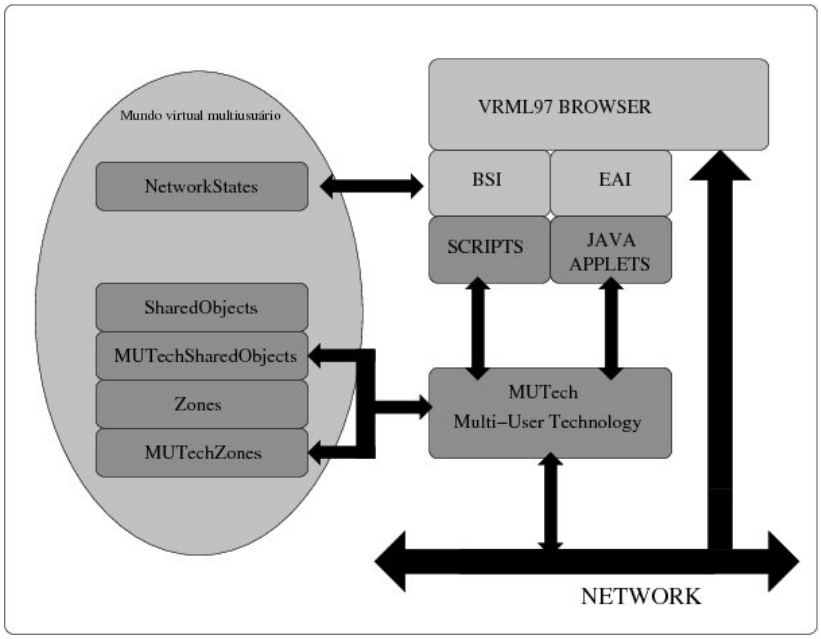

Figura 3.3: Estrutura Living Worlds.

A Figura 3.3 mostra a arquitetura de *LW*. A tecnologia multiusuários *MUTech* é o componente da arquitetura *LW* que implementa o comportamento dos elementos:

Zones, SharedObjects e NetworkStates. *MUTech* também é encarregado de gerenciar a distribuição dos eventos entre os clientes. *MUTechZone* e *MUTechSharedObject* representam os componentes de uma implementação *LW* específica.

O padrão *VRML97* não contempla a sua utilização em ambientes virtuais compartilhados por vários usuários conectados através de uma rede de computadores. Isto porque seu sistema de tratamento de eventos admite o roteamento de eventos apenas dentro de um único grafo de cena. Para resolver este problema, *LW* utiliza o paradigma da inserção de nós para transmitir os eventos à distância: novos nós, encarregados de propagar as modificações locais, são introduzidos no grafo de cena existente. Para isto, é necessário modificar as rotas existentes a fim de levar em conta os novos nós introduzidos, ditos "compartilhados", como mostrado na Figura 3.4.

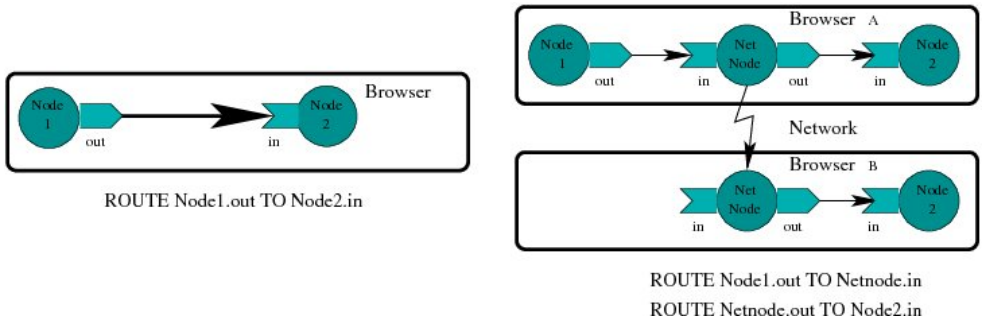

Figura 3.4: Mecanismo de inserção de nós no grafo de cena: as rotas existentes devem ser modificadas.

Como *LW* se limitou apenas a ser uma técnica de descrição de mundos multiusuários em VRML97 e não definiu nenhum protocolo de rede no nível do *MUTECH*, vários blocos *MUTECHs* não interoperáveis entre eles foram desenvolvidos. A difícil sintaxe de descrição da cena virtual utilizada por LW foi alvo de críticas e levou ao surgimento de várias propostas alternativas [54], dentre elas, a proposta

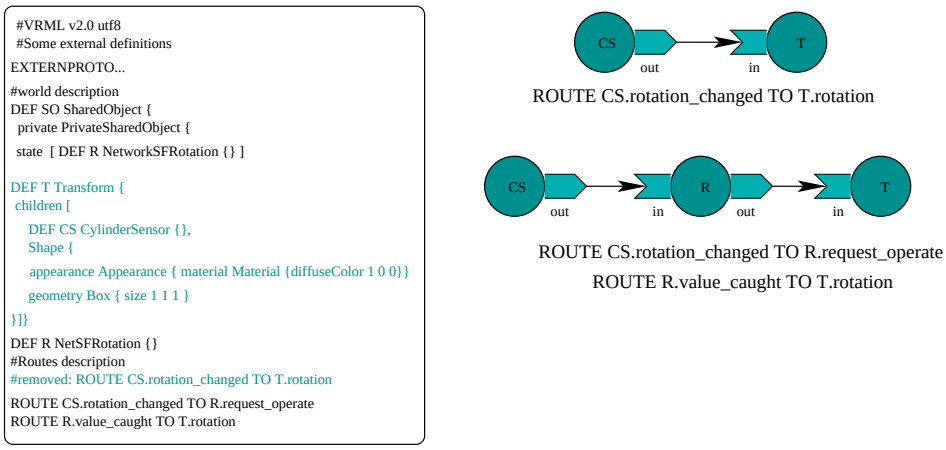

Figura 3.5: Exemplo de conteúdo multiusuário concebido de acordo com a estrutura Living Worlds.

Core Living Worlds [55]. *Core Living Worlds* retomou os conceito de LW como: *Zones*, *SharedObjects* e *networkStates*, mas suas interfaces foram simplicadas. A Sony propôs uma implementação desta proposta por meio do seu browser *Community Place* [56].

Inspiraram-se também na proposta de *Core Living Worlds*, DeepMatrix [57] e VNet [58]. VNet desenvolveu seu próprio protocolo (*VIP*), utilizado também por *DeepMatrix*, para trocar dados entre o servidor e os clientes. Outra implementação da proposição *Core Living Worlds* é *Living Space* [59] que utiliza um sistema de notificação chamado *Keryx notification System*.

Blaxxun[5] retoma em linhas gerais a linha introduzida por LW. Encontra-se em *Blaxxun* os mesmos conceitos de LW com nomes diferentes: *Blaxxun* Zone ao invés de Zone, *SharedEvent* ao invés de *NetworkStates*. A proposição multiusuários de *Blaxxun* também se apóia sobre o paradigma de inserção de nós. No entanto, contrariamente ao *core* LW, que para cada tipo base de VRML97 definia um nó equivalente embutindo o comportamento da rede, *Blaxxun* propõe somente um nó *SharedEvents* para todos os tipos VRML97 de base e define vários campos, um para cada tipo

[5]www.blaxxun.com

básico VRML97. *Blaxxun* propõe também diferentes mecanismos para a gestão de eventos compartilhados (*SharedEvents*), como a gestão de acesso concorrente e a distribuição dos eventos compartilhados apenas à um grupo de usuários e não a todos os usuários representados no mundo virtual.

Outra proposta de ambientes multiusuários que se apoiou na crítica da complexidade sintática de LW foi *VSPLUS* [60]. Este sistema propõe uma alternativa à utilização de LW, simplificando, mesmo para não-experts em programação e nem em VRML, o processo de descrição de mundos virtuais multiusuários. Assim como LW, *VSPLUS* utiliza o paradigma de inserção de nós: os chamados *NetNodes* em *VSPLUS* são adicionados ao grafo de cena monousuário. Cada *NetNode* é encarregado de transmitir os eventos que eles recebem na entrada (*eventIns*) para as cópias distantes, de maneira que essas cópias gerem os eventos de saída (*eventOuts*).

Os *NetNodes* podem ser comparados aos *NetworkedStates* propostos por LW: para cada tipo de base de VRML97 (*SFBool, SFColor, etc*), *VSPLUS* define um nó embutindo o comportamento de rede (*NetSFBool, NetSFColor, etc*).

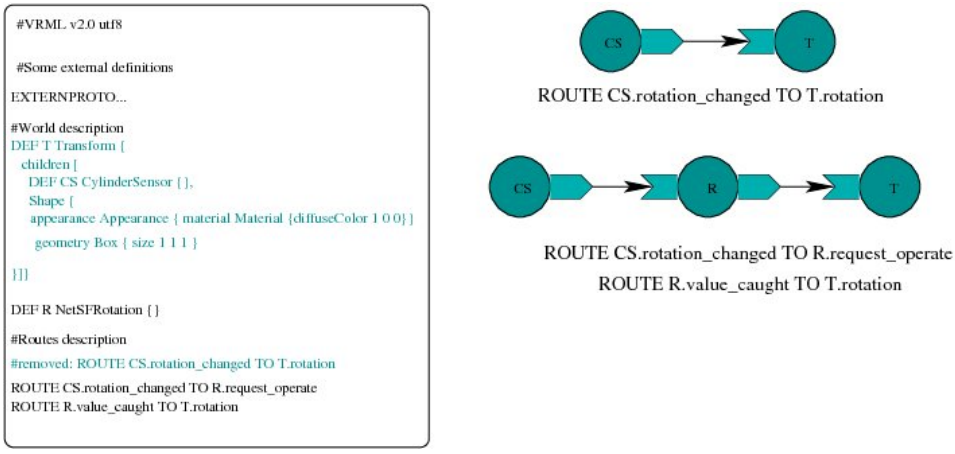

Figura 3.6: Um exemplo de conteúdo multiusuário concebido de acordo com a estrutura VSPLUS.

O script mostrado na Figura 3.6 mostra o arquivo VRML97 original praticamente intacto, sofrendo apenas a adição de um *netnode* no grafo de cena original e a consequente substituição das rotas.

VSPLUS utiliza o navegador *VRML97 Community Place* da *Sony* para mostrar a cena 3D e sua plataforma de comunicação é do tipo cliente/servidor. Foi definido também na arquitetura *VSPLUS*, o *Consistency Manager*, encarregado de garantir a consistência dos netnodes em todos os clientes. O *consistency manager* é um componente conceitual e dependente da implementação. No *Community Place*, por exemplo, o *Consistency Manager* de um *Netnode* para um avatar é um processo sendo executado em um browser. Browsers e *Consistency Manager* comunicam-se através da troca de mensagens via rede de computadores. Existe somente um *Consistency Manager* para cada *Netnode*. Na Figura 3.7 pode-se verificar o comportamento dos *Netnodes* e do *Consistency Manager* através de três processos : Pa, Pb e Pc

- Pa - quando um evento E é enviado do *node*0 para o *nodeN* no browser A, o *nodeN* gera uma mensagem para compartilhar o evento E e a envia ao *consistency manager*.

- Pb - o *consistency manager* processa a mensagem e envia o resultado para todos os browsers via rede de computadores.

- Pc - Em cada browser, o netnode *nodeN* correspondente recebe a mensagem e gera o evento E.

Além de fazer a sincronização dos eventos entre os browsers, o *Consistency Manager* fornece aos usuários recém-conectados o estado atual do mundo virtual com-

partilhado.

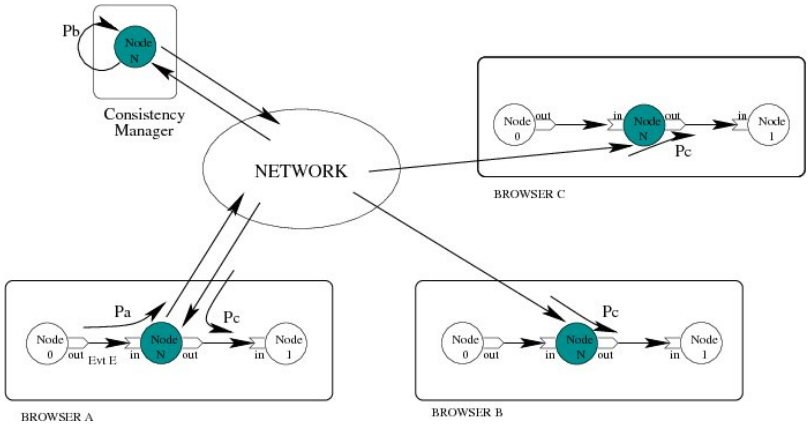

Figura 3.7: Funcionamento dos *Netnodes*

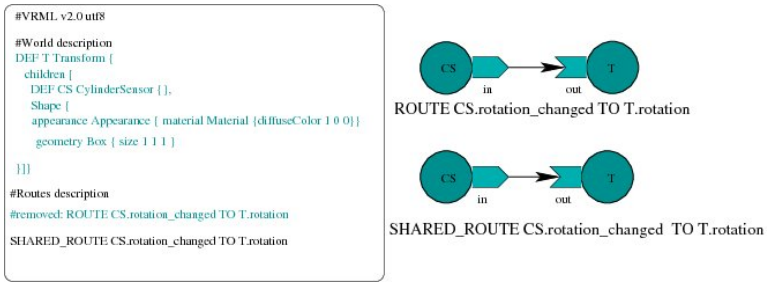

Figura 3.8: Descrição de um ambiente virtual compartilhado com a proposição de Carson

Uma abordagem para a concepção de mundos virtuais multiusuários diferente das apresentadas até aqui, todas baseadas no paradigma de inserção de nós, foi proposta por Carson em [61]. Carson propõe o conceito de rotas compartilhadas, adicionando à linguagem *VRML97* as palavras chaves *SHARED_ROUTE* e *LOCKED_ROUTE*. Estas novas palavras-chave mantêm a sintaxe da palavra-chave *ROUTE* do padrão *VRML97*, no entanto suas semânticas são diferentes. Um evento transmitido ao longo de uma *SHARED_ROUTE* é propagado para todas as outras cópias do mundo virtual. Um exemplo do uso da diretiva *SHARED_ROUTE* é mostrado na Figura 3.8. O arquivo original *VRML97* permanece praticamente o mesmo, exceto pela

substituição da diretiva *ROUTE* pela diretiva *SHARED_ROUTE*.

Enquanto uma *SHARED_ROUTE* propaga os diferentes eventos sem efetuar nenhum controle de acesso, uma *LOCKED_ROUTE* efetua uma verificação do *LOCK* antes de efetuar a propagação do evento. Para a gestão dos *LOCKs*, um serviço centralizado *Gatekeeper* é introduzido na arquitetura. O *Gatekeeper* é encarregado também de fornecer o estado atual do mundo virtual compartilhado aos usuários recém-conectados. Ao invés de optar pelo desenvolvimento de um navegador *VRML97* estendido, levando em conta as extensões da linguagem, Carson preferiu utilizar um navegador *VRML* clássico para visualizar o mundo compartilhado. Como mostra a Figura 3.9, o arquivo *VRML97* estendido com as diretivas *SHARED_ROUTE* e *LOCKED_ROUTE* são pré-processados por um compilador, a fim de que o arquivo com os dados 3D seja compatível com um navegador *VRML97* clássico.

O pré-processador produz os seguintes arquivos de saída:

- arquivos de dados 3D compatíveis *VRML97*, eles são carregados pelo navegador *VRML97* clássico;
- arquivos fontes Java implementando as diferentes rotas compartilhadas. Uma vez que estas fontes são compiladas em *applets*, elas são carregadas pelo navegador *HTML* durante o carregamento do mundo compartilhado e se comunicam com o navegador *VRML97* utilizando a EAI;
- arquivos específicos permitindo ao *Gatekeeper* de conhecer as diferentes variáveis que armazenam o estado atual do mundo compartilhado.

O fato de utilizar um pré-processador impede qualquer compartilhamento

dinâmico durante a execução da sessão virtual, pois o compartilhamento é gerado no momento da compilação e da geração dos diferentes arquivos (applets Java e fontes *VRML97*).

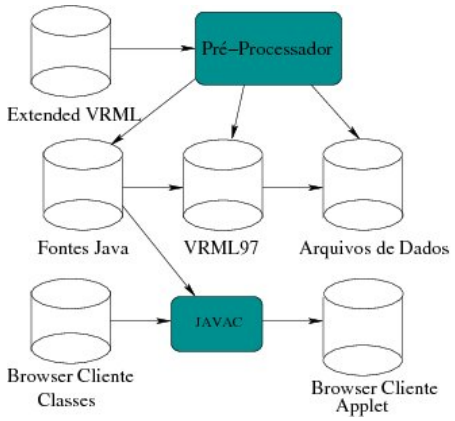

Figura 3.9: Processo de criação de um ambiente virtual compartilhado utilizando a proposição de Carson.

O projeto *SPIN-3D* [62] [54] propõe introduzir uma descrição compartilhada para cada objeto *VRML97* monousuário a fim de criar um objeto multiusuário. *SPIN-3D* propõe o conceito de *SharedNode* e de *SharedSet*. *SharedNode* é o nome genérico dado aos nós de descrição de compartilhamento; um nó deste tipo permite ao criador do mundo virtual de especificar um campo do objeto *VRML97* que será compartilhado, ou seja, que terá seu valor sincronizado entre os diferentes usuários representados no mundo virtual. A substituição de campos *VRML97* por campos específicos torna *SPIN-3D* incompatível com os browsers *VRML97* padrões. Os *SharedNodes* são equivalentes aos *NetworkStates* de *Living Worlds*, dos *NetNodes* de *VSPLUS* e dos *SharedEvents* da tecnologia *Blaxxun*. Assim, para cada tipo base de *VRML97* (*SFBool, SFColor, etc*) existe um *SharedNode* associado (*SharedSFBool, SharedSFColor, etc*).

Na Figura 3.10, o campo *alias* permite designar o nome do campo a ser compar-

```
EXTERNPROTO  Sharedxxx[
field               SFString alias ""
exposedField    SFSTring mode "public"
field               SFString  how  "action"
field               SFString lockName ""
]  [:proto/sharedxxx
```

```
EXTERNPROTO  SharedSFBool [
field               SFString alias ""
exposedField    SFSTring mode "public"
field               SFString  how  "action"
field               SFString lockName ""
]  [:proto/sharedxxx
```

Figura 3.10: Protótipo de um SharedNode.

tilhado, isto é, o campo de um nó contido na geometria 3D. Já a noção de *SharedSet*, do ponto de vista semântico, pode ser associada a um objeto 3D compartilhado. Os filhos de um *SharedSet* são nós de compartilhamento (*SharedNode*).

```
EXTERNPROTO  SharedSet [
exposedField     SFString mode "public"
exposedField      MFNode children [ ]
eventIn              MFNode  addChildren
eventIn          MFNode  removeChildren
]  [:externproto/sharedset
```

Figura 3.11: Protótipo do nó SharedSet.

```
#VRML v2.0 utf8
#Some external definitions
EXTERNPROTO...
#Sharing description
SharedSet {
 children [
    SharedSFRotation {
    alias "S.rotation_changed"
    lockName "rotation"
} ] }
#World description
DEF T Transform {
 children [
   DEF S Sensor 3D{ lockName "rotation" },
   Shape {
   appearance Appearance { material Material {diffuseColor 1 0 0}}
    geometry Box { size 1 1 1 }
}]}
#Routes description
ROUTE S.rotation_changed TO T.rotation
```

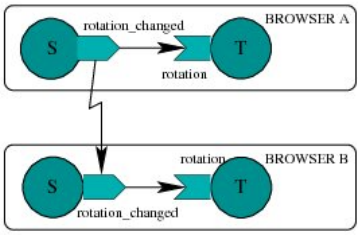

Figura 3.12: Exemplo de conteúdo VRML97 multiusuário concebido com a infraestrutura SPIN-3D.

O sistema *EVE* [30] introduz os seguintes conceitos no que diz respeito ao compartilhamento de dados *VRML*: um arquivo SVE, onde é feita a descrição de compartilhamento; *SharedNodes*, contendo dois campos *SharedEventIn* e *SharedEventOut*; e os eventos *SFCreate* e *SFDelete*. O criador do mundo virtual deve executar os seguintes passos:

1. Inserir, no arquivo *VRML*, nós de colisão e um nó sensor de proximidade (que fornece a posição e a orientação dos avatares no mundo virtual);

2. Remover todas as rotas compartilhadas do arquivo *VRML* original;

3. Criar o arquivo *SVE* correspondente. O arquivo *SVE* contém a definição dos nós que se quer compartilhar e todas as rotas que foram removidas do arquivo *VRML* original.

```
#VRML v2.0 utf8
# Step 1: Adicionar um Avatar Root e um sensor de proximidade

Collision [
  children  DEF  AvatarRoot  Group  [
    children [ ] } }

DEF  ProxSensor  ProximitySensor  {
     size  1000  1000  1000  }
Group [ children [
   DEF  box  Transform  {
        children Shape  { geometry Box  { }
]}
    DEF sensor  SphereSensor  {
    } ] ]
# Step 2 : Removal of the shared ROUTEs
# ROUTE  sensor.rotation_changed  TO  box.set_rotation
```

Figura 3.13: Descrição de um mundo virtual compartilhado utilizando a proposição EVE.

```
node box
[
        eventIn SFRotation set_rotation;
]
node sensor
[
      eventOut  SFRotation  rotation_changed;
]
ROUTE sensor.rotation_changed  TO box.set_rotation;
```

Figura 3.14: Definição de um SharedNode em EVE.

EVE também previu uma maneira de compartilhar os objetos que não faziam parte da configuração inicial do mundo virtual e que foram inseridos dinamicamente. Para isso foi definido um nó especial chamado *routeCreator* que permite a comunicação entre o applet Java e o mundo virtual multiusuários através da EAI.

Até a presente data, nenhuma das abordagens para mundos VRML multiusuários descritas nesta seção foi efetivada como um padrão pelo *Web3D Consortium*. A seguir, é descrito o protocolo VIP para troca, entre usuários, de mensagens com conteúdo VRML.

VIP - *VRML* Interchange Protocol

RECOLLVE se apóia sobre o protocolo VIP[58] para conceber um modelo único de mensagens capaz de atender a todas as necessidades de comunicação do sistema. O termo protocolo não deve ser tomado aqui estritamente no seu significado de redes de computadores. Trata-se, de fato, apenas de uma estrutura de mensagens.

Para cada tipo base de *VRML*, *VIP* definiu um tipo equivalente que se apresenta sob a forma de um *VField*, e cujo a correspondência é dada na Tabela 3.1.

As mensagens em RECOLLVE podem ser de dois tipos:

- O primeiro tipo de mensagem refere-se às mensagens ligadas diretamente ao estado da base de dados das entidades. Elas contêm dados primários, imagens dos dados extraídas e colocadas na cena virtual graças à SAI [63].

- O segundo tipo de mensagem refere-se às mensagens de caráter administrativo.

Uma mensagem VIP conforme ilustrada na Tabela 3.2, possui três campos que são descritos em seguida.

A seguir são descritos os diferentes campos desta mensagem:

- *Id*: este campo indica a entidade referida pela mensagem;

- *Field*: este campo indica a natureza da mensagem;

Tabela 3.1: Correspondência entre tipos *VRML* e VFields em RECOLLVE.

Campos *VRML*	Definição	Valor	VField
SFBool	Booleano	Bool	VFSBool
SFString MFString	Cadeia de caracteres Cadeia múltipla	String String[]	VSFString VMFString
(S/M)FColor	Cor	Float[3]	V(S/M)FColor
SFImage	Imagem	Byte[n]	VSFImage
(S/M)FInt32	Inteiro	Int	V(S/M)FInt32
(S/M)FNode	Nó	n/a	n/a
(S/M)FRotation	Rotação	Float[4]	V(S/M)FRotation
(S/M)FTime	Tempo	Double	V(S/M)FTime
(S/M)FVec2f	Vetor plano	Float[3]	V(S/M)FVec2f
(S/M)FVec3f	Vetor posição	Float[2]	V(S/M)FVec3f

Tabela 3.2: Forma da mensagem VIP.

id	Field	Value
Int	Short	Byte Data

- *Value*: este campo representa o *VField* associado à mensagem; o primeiro sub-campo indica o tipo de *VField*.

O campo *Field* determina a natureza, e portanto, o papel da mensagem. Os valores possíveis deste campo são separados em duas partes:

- A primeira parte define as mensagens de caratér administrativo. Todos os

campos tem valor negativo, como ilustrado na Tabela 3.3;

Tabela 3.3: Mensagens de caráter administrativo.

Valor	Nome VIP	Descrição da Ação
-1	$FIRE\_ACTION$	Dispara uma ação composta
-2	$ROUTER\_COMMAND$	Modifica a tabela de roteamento
-3	$USER\_INFO$	Declara um novo usuário
-4	$QUIT$	Notifica a saída de um usuário
-5	ADD_OBJECT	Adiciona uma entidade à cena
-6	$REMOVE\_OBJECT$	Retira uma entidade da cena
-7	$MESSAGE$	Envio de mensagem
-8	$PRIVATE\_MESSAGE$	Envio de mensagem privada
-9	SET_RIGHT	Concede um direito
-10	$UNSET\_RIGHT$	Retira um direito
-11	$TAKE\_START\_OBJECT$	Notifica a tomada de posse uma entidade por outra
-12	$TAKE\_STOP\_OBJECT$	Notifica a liberação de uma entidade
-13	$MOVE\_START\_OBJECT$	Notifica o deslocamento de uma entidade por outra
-14	$MOVE\_STOP\_OBJECT$	Notifica o fim do deslocamento

- A segunda parte refere-se às mensagens de sincronização: quando um campo tem um valor positivo ou nulo, isto significa que a mensagem corresponde a uma informação de estado. O valor indica o indíce do estado referido. Os três primeiros valores são comuns a todas as entidades, como ilustrado na Tabela 3.4..

Tabela 3.4: Mensagens de estado.

Valor	Nome VIP	Descrição da Ação
0	NAME	Transmite o nome de uma entidade
1	ORIENTATION	Transmite a orientação de uma entidade
2	POSITION	Transmite a posição de uma entidade
> 2	N/A	Transmite um campo qualquer

X3D

X3D é um padrão também criado pelo *Web3D Consortium* para suceder VRML. Dentre as melhorias introduzidas pela especificação *X3D* está a expansão das capacidades do grafo de cena para incorporar os avanços nos dispositivos gráficos comerciais, via introdução de novos nós e campos de dados; e o suporte a múltiplos formatos de codificação, incluindo o formato *XML*. O padrão *X3D* oferece ainda a *SAI* [63], que é a evolução da *EAI*.

H-Anim

H-Anim [64] é um padrão utilizado para a descrição de humanóides animados. Um corpo *H-Anim* é um conjunto hierárquico de nós de articulação, aos quais outros nós podem ser associados. Assim, para deslocar uma parte de um corpo *H-Anim* é preciso atribuir novos valores de localização para os nós de articulação envolvidos. Especificar uma animação seguindo o padrão *H-Anim* é uma tarefa complicada e nada intuitiva por conta da distância entre *H-Anim*, uma linguagem de animação, e a linguagem natural.

3.2.2 Xj3D

Xj3D [65] é um projeto em desenvolvimento, liderado pelo *Web3D Comsortium*, e tem como objetivo fornecer um browser de código aberto para manipulação de conteúdos VRML e X3D. O browser Xj3D é totalmente desenvolvido usando as APIs Java e Java3D.

Xj3D oferece suporte a *SAI* [63], que é a API usada na comunicação entre os aplicativos desenvolvidos em Java e o conteúdo da cena descrito em X3D. A *SAI* oferece dois tipos de acesso à cena virtual: acesso interno e acesso externo. No acesso interno, um nó de script dentro do arquivo X3D que descreve a cena, ao ser carregado pelo browser (Xj3D), chama a classe Java onde está localizado o código que irá realizar as alterações pretendidas na cena virtual. No acesso externo, a classe Java carrega o arquivo X3D e realiza as alterações na cena correspondente. Neste tipo de acesso, a classe chama uma instância do browser, e utiliza a API para receber informações oriundas do browser, ou para realizar uma ação dentro da cena virtual.

3.3 Plataforma de Integração

A problemática associada à integração de aplicações não se restringe ao domínio do TCSC. Em numerosos domínios e, principalmente no domínio do desenvolvimento de aplicações distribuídas, um dos principais eixos de pesquisa diz respeito à possibilidade de integrar diferentes sistemas. Neste contexto, foi utilizado como plataforma de integração entre RECOLLVE e outras aplicações colaborativas o sistema LEICA [4], descrito a seguir.

3.3.1 LEICA

LEICA [1] representa um ambiente de integração que adota uma abordagem fracamente acoplada, o que permite a abstração dos detalhes internos de cada aplicação, facilitando assim o processo de integração. O papel principal de *LEICA* é tornar possível a utilização das diferentes ferramentas de comunicação e de colaboração no contexto de uma mesma sessão colaborativa. Uma aplicação colaborativa, concebida independentemente de *LEICA*, uma vez integrada a este ambiente, será capaz de interagir com o ambiente e com as outras aplicações integradas, preservando sua autonomia de execução. O grau de interação entre as aplicações integradas depende exclusivamente da API de cada aplicação, sendo que esta API caracteriza a natureza dos eventos que podem ser observados e as ações que cada aplicação é capaz de executar. O objetivo desta integração é poder coordenar de maneira controlada diferentes funcionalidades das aplicações, indo além de uma simples utilização de várias aplicações simultâneamente.

Para realizar uma atividade colaborativa na plataforma *LEICA*, é preciso definir e configurar o que se chama de *SuperSessão*. Uma *SuperSessão* representa uma sessão de trabalho global, compreendendo todos os atores de uma atividade colaborativa (as aplicações colaborativas, os usuários e seus papéis, etc.). Dentro do contexto de uma *SuperSessão* podem existir diferentes Sessões Específicas. Cada Sessão Específica é na verdade uma sessão colaborativa convencional definida por uma aplicação colaborativa (por exemplo, uma sessão de áudioconferência ou de co-navegação web) [66].

Para cumprir o objetivo de adaptar as aplicações e integrá-las à *LEICA*, um *Wrapper*, ou *Empacotador*, é utilizado. O *wrapping* é uma técnica tradicional, onde

uma aplicação é embutida em uma nova interface para torná-la executável em um contexto diferente.

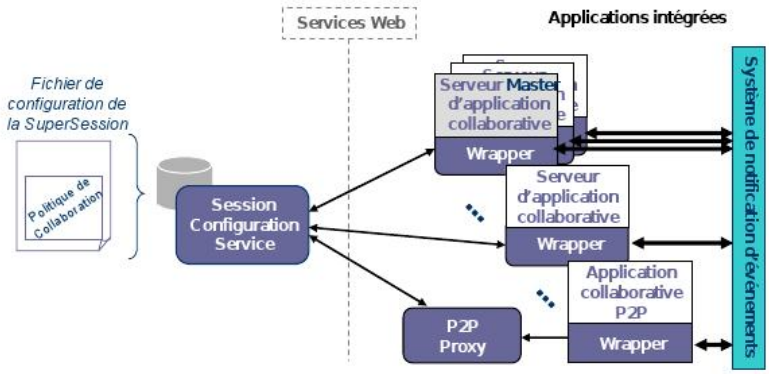

Figura 3.15: Estrutura de integração de LEICA.

Como ilustrado na Figura 3.15, os *Wrappers* são integrados aos servidores das aplicações cliente/servidor ou multiservidor, e aos pares das aplicações *P2P* (Peer-to-Peer). No caso dos servidores, cada *Wrapper* compreende uma interface de Serviço Web permitindo a aplicação a interagir com *LEICA* A aplicação se registra em *LEICA* como aplicação integrada, e através de sua interface de Serviço Web pode interagir com o *SCS - Serviço de Configuração de Sessão.*

Durante a configuração de uma *SuperSession*, o SCS contacta dinamicamente cada aplicação integrada para requisitar informações específicas sobre sua API. Tais informações serão usadas na criação de Sessões Específicas e especificação da Política de Colaboração de uma *SuperSessão.*

Enquanto uma *SuperSessão* é executada e as atividades colaborativas evoluem, as aplicações colaborativas envolvidas notificam a ocorrência de eventos entre elas (através do Sistema de Notificação de Eventos, baseado no paradigma *Publish/Subscribe*). Paralelamente a essas notificações, os *Wrappers* são encarregados de gerenciar a política de colaboração da *SuperSession.* Esta política é constituída

de um conjunto de regras determinando como as aplicações devem reagir às notificações de evento. A política de Colaboração permite, portanto, que seja definida para cada *SuperSession* uma semântica de integração específica (i.e. como coordenar as aplicações integradas) de acordo com diferentes requisitos dos usuários.

3.4 Considerações Finais

Este capítulo apresentou os conceitos de agentes inteligentes e sistemas multiagentes que nortearam a concepção da arquitetura RECOLLVE. Apresentou também a tecnologia utilizada na descrição e animação dos ambientes virtuais, assim como as propostas de padronização para ambientes virtuais multiusuários em VRML. Este capítulo apresentou ainda a plataforma LEICA, ambiente de integração necessário à integração de RECOLLVE com as aplicações externas. No capítulo seguinte é apresentada e detalhada a arquitetura multiagentes do sistema RECOLLVE.

Capítulo 4

Arquitetura do Sistema RECOLLVE

RECOLLVE é um sistema de realidade virtual desktop que foi concebido com o objetivo de reforçar a percepção dos usuários na realização de trabalho colaborativo. Este sistema é capaz de representar (simular) a dinâmica das atividades colaborativas realizadas pelos usuários, e ao mesmo tempo, de gerenciar as aplicações colaborativas integradas a ele. Seus principais beneficiários são todos os usuários que fazem uso de alguma aplicação colaborativa.

RECOLLVE pode ser usado como uma plataforma virtual para representar a dinâmica do trabalho colaborativo em diversas aplicações, tais como, Ensino a Distância (EaD), Reuniões Virtuais, Treinamento, e em toda aplicação que exija um esforço de coordenação dos usuários que estão participando de uma sessão colaborativa. Neste contexto, é extremamente importante gerenciar os diferentes papéis desempenhados pelos usuários, assim como os direitos de acesso associados a eles. O

objetivo de RECOLLVE não é definir políticas de acesso para os usuários, mas sim de representá-las no interior da cena virtual. Este aspecto é importante pois tenta transcrever para a cena virtual um aspecto natural dos sistemas reais: o protocolo social controlando o trabalho colaborativo real.

Antes de detalharmos a arquitetura do sistema RECOLLVE, é apresentada a seguir a definição de uma sessão virtual colaborativa em RECOLLVE.

4.1 Sessão Virtual

Um Ambiente Virtual Colaborativo deve suportar diferentes sessões de acordo com o propósito da simulação. Formalmente, em RECOLLVE, uma sessão virtual colaborativa é descrita como uma 4-tupla: ($cenaV$, $Papeis_l$,$Usuarios_l$, Atr_l). Onde $CenaV$ é a cena virtual. Ela é definida como sendo um conjunto finito de espaços virtuais conectados através de portas virtuais;

$Papeis_l$ é um conjunto finito de papéis gerais; $Usuarios_l$ é o conjunto finito de usuários conectados em uma mesma sessão virtual colaborativa; e Atr_l é a lista de atributos caracterizando a sessão virtual colaborativa. Estes atributos descrevem informações sobre o contexto da sessão (nome, propósito) e o número máximo de usuários permitido.

4.1.1 Cena Virtual

A cena virtual é definida como uma 4-tupla: ($SalaV$, $Avatares_l$, $Objetos_l$, API). $SalaV$ é uma representação 3D de um ambiente virtual especificado segundo os padrões VRML/X3D [51] [67]. Uma sala virtual pode representar uma sala de aula, uma biblioteca ou a linha de produção em uma fábrica. Os espaços virtuais podem

ser descritos utilizando-se das diversas ferramentas disponíveis tais como vrmlpad[1] e x3dedit[2].

Avatares

Os avatares são utilizados para representar o usuário na cena virtual. Cada usuário é representado por um agente com uma representação 3D, i.e., um avatar na forma humanóide e um Agente Usuário. O comportamento do Agente Usuário é descrito na Seção 4.2.1. Nosso modelo de avatar segue a especificação *H-Anim* (Humanoid Animation Working Group)[3], descrita no Capítulo 3.

A presença de avatares reforça a noção de presença dos outros usuários com os quais é possível colaborar e, à esta noção de presença do outro usuário, dá-se o nome de *awareness* de pessoas. As ações realizadas pelos usuários nas aplicações colaborativas são representadas na cena cena virtual por ações dos avatares.

Dessa forma, como os avatares podem executar ações que são adequadas aos objetos, respeitando sua funcionalidade, reforça para um usuário a consciência de quais ações estão sendo executadas pelos outros usuários com os quais ele compartilha a cena virtual. As atividades de colaboração se tornam mais claras quando expressas por meio de ações dos avatares, e quanto maior for a gama de ações disponíveis para os avatares, maior será a capacidade dos usuários de expressarem suas pretensões de colaboração.

Para que os avatares fossem capazes de representar as ações dos usuários, uma biblioteca de ações foi construída e categorizada da seguinte forma:

[1]http://www.parallelgraphics.com/products/vrmlpad/

[2]http://www.web3d.org/x3d/content/README.X3D-Edit.html

[3]http://www.h-anim.org

- *Ações de Navegação*: estas ações são usadas para navegação dos avatares na cena virtual, como andar, por exemplo;

- *Ações de Controle*: este tipo de ação tem uma forte carga semântica, pois elas indicam, por exemplo, quem protagoniza a ação em um dado momento; são elas: pegar, entregar, liberar;

- *Ações Apropriadas*: este tipo de ação é executado sobre os objetos virtuais considerando sua funcionalidade, por exemplo, sentar, ler, telefonar, etc;

- *Expressões Gestuais*: estas ações são usadas para indicar opiniões e desejos.

Objetos Virtuais

Em RECOLLVE, Objetos são representados por agentes com uma representação 3D. Cada objeto é composto por uma descrição 3D no padrão *VRML* e por um Agente Objeto. O Agente Objeto mantém as regras de acessibilidade associadas ao mesmo. A descrição 3D do objeto, por sua vez, contém também uma interface, como ilustrado na Figura 4.1.

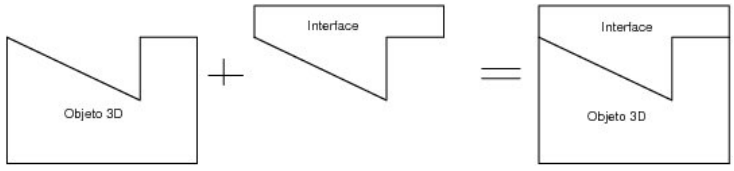

Figura 4.1: Composição do objeto.

A seguir, um exemplo de interface para o objeto *cadeira*. A interface contém também as informações necessárias para a correta manipulação do objeto. O conteúdo desta interface é usado para, ao clic do mouse, o instante exato do clic

e a posição na cena 3D da cadeira serem roteados para o avatar do usuário que selecionou o objeto.

```
#VRML V2.0 utf8
####################
# Do not edit this part. #
####################
### StartOfDeclaration
### instanciationParameters
### objectType
# chair
### eventInList
### enventOutList
# SFTime sit_start_time
# SFVec3f sit_start_position
# SFRotation sit_start_orientation
### routeList
# chair sit_start_time avatarSelf sit_start_time
# chair sit_start_position avatarSelf sit_start_position
### EndOfDeclaration
```

API

Em RECOLLVE foi definida uma API permitindo sua integração com outras aplicações colaborativas. Desta forma, eventos externos têm consequências na cena virtual. Da mesma forma, eventos ocorridos na cena virtual têm consequências em

aplicações externas. Por exemplo, a seleção do avatar de outro usuário poderia iniciar uma sessão de áudioconferência, e um objeto 3D representando um microfone seria inserido na cena virtual; ou um objeto 3D representando um telefone desempenharia o papel de um *link* para uma aplicação de Voz sobre IP, como o sistema Skype. Assim, uma sessão de conversação utilizando o aplicativo Skype poderia ser representada na cena virtual.

A seguir, um extrato da API implementada em RECOLLVE:

- *add_object* (Entity object): adiciona o objeto *object*;
- *remove_object* (Entity object, int id): remove o objeto *object*;
- action (Entity avatar, String action, Entity object): avatar realiza a ação *action* sobre o objeto *object*.

4.1.2 Papéis

Em uma sessão virtual colaborativa, cada usuário conectado é associado a um papel. O conceito de papel se refere a um mesmo conjunto de responsabilidades e privilégios em um AVC. Um papel é definido como sendo uma tupla: $Rl = (Rl_{id}, Rl_{name}, Rl_{at})$. Onde Rl_{id} é o único identificador do papel; Rl_{name} é o nome do papel; Rl_{at} é a lista de atributos que caracteriza este papel.

4.1.3 Usuários Conectados

Usuários conectados corresponde a todos os usuários que juntaram-se a uma mesma sessão virtual colaborativa. Como descrito antes, cada usuário é associado a um papel geral e ele é representado na cena virtual por um avatar. Formalmente, em

RECOLLVE, um usuário conectado é definido como uma tupla: $U_l = (U_{id}, U_{profile}, U_m, U_{at})$. Onde U_{id} é o único identificador do usuário; $U_{profile}$ é o identificador do perfil ao qual o usuário está associado; o perfil é definido como sendo uma tupla: $U_{profile} = (Rl, SalaV, OwnObj)$, onde $OwnObj$ é a lista dos objetos de posse do usuário. U_{at} é a lista de atributos caracterizando o usuário conectado. Atributos fornecem informações pessoais sobre o usuário (nome, e-mail), informações sobre o acesso do usuário ao AVC, como o endereço IP e o tipo de dispositivo que está sendo utilizado (Computador Pessoal, PDA, etc).

4.2 Arquitetura multiagentes de RECOLLVE

Considerando os objetivos de RECOLLVE, algumas premissas foram definidas para a configuração da sua arquitetura. O sistema deve:

1. ser capaz de gerenciar diferentes perfis dos usuários;
2. gerenciar diferentes níveis de acesso para os usuários;
3. ter flexibilidade para representar na cena virtual os diferentes estados de uma atividade colaborativa;
4. ter flexibilidade para representar na cena virtual as diferentes funcionalidades presentes nas APIs das aplicações;
5. ter flexibilidade para gerenciar, a partir da cena virtual, as aplicações colaborativas integradas ao sistema.

Tomando como base as premissas definidas para o sistema RECOLLVE, optou-se por utilizar o paradigma de Agentes para modelar a sua arquitetura. A utilização

de agentes na modelagem do sistema RECOLLVE confere, além de elegância ao modelo, uma grande vantagem nas tomadas de decisões, pois os agentes podem, a partir da interação de um usuário com seu ambiente, obter dados por meio de sensores e tomar decisões baseadas nos dados coletados.

A Arquitetura do sistema RECOLLVE é composta por um sistema multiagentes, ilustrada na Figura 4.2. Em RECOLLVE, todas as entidades são representadas por agentes que se comunicam e interagem entre sí.

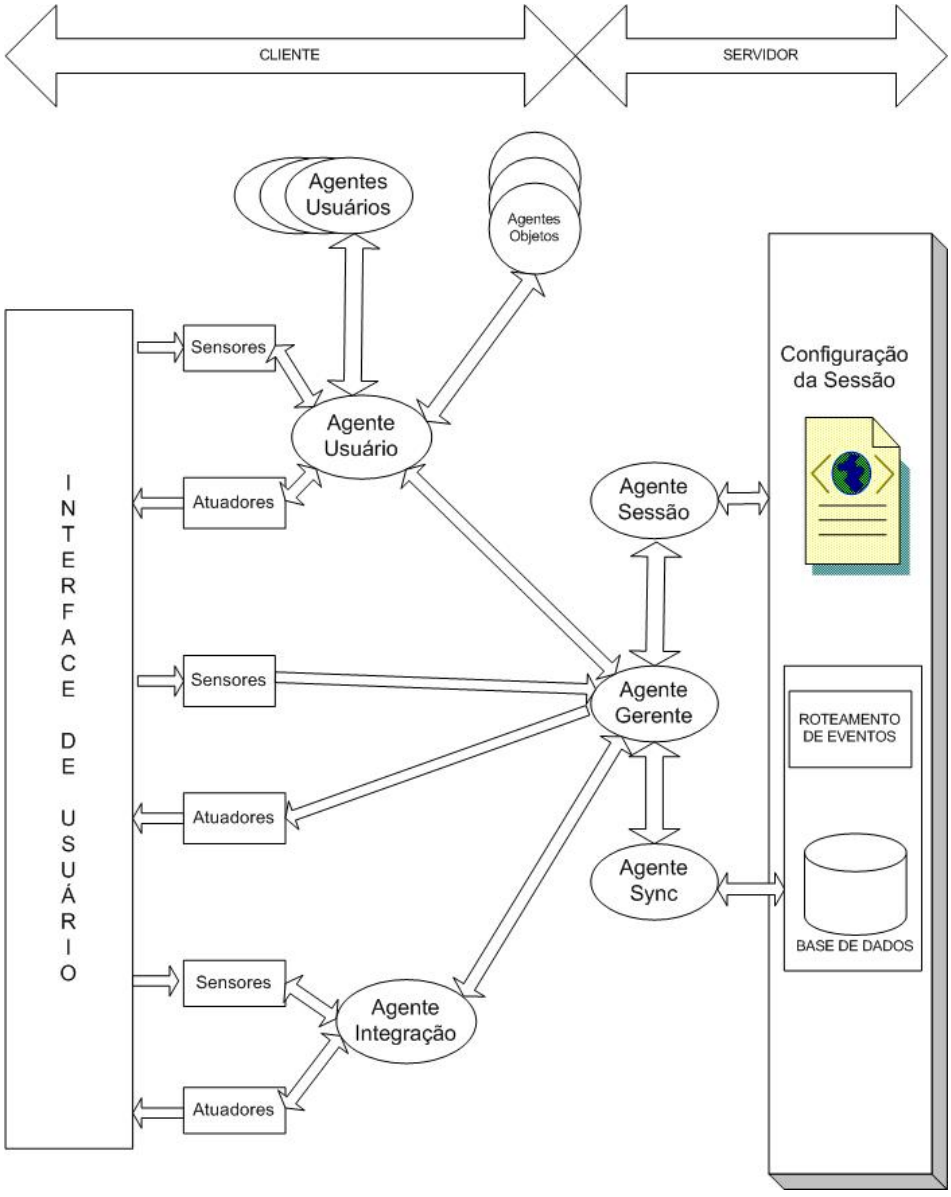

Figura 4.2: Arquitetura multiagentes de RECOLLVE.

Foram definidos seis tipos de Agentes:

1. Agente Usuário;
2. Agente Objeto;
3. Agente Gerente;
4. Agente Integração;
5. Agente Sync;
6. Agente Sessão.

O Agente Usuário é responsável por gerenciar toda interação do usuário com a cena virtual. O Agente Objeto gerencia o acesso dos usuários aos objetos. O Agente Gerente coordena e controla as interações entre os agentes e o Agente Integração, por sua vez, gerencia as interações de RECOLLVE com o ambiente externo.

O Agente Sessão é responsável por adicionar novos usuários para a sessão virtual e por remover aqueles que desejam deixar a sessão virtual. Além disso, ele também é responsável por tratar as requisições de conexão e validar login e senha de usuários.

O Agente Sync é responsável por manter a cena virtual sincronizada em todos os clientes. Ele é responsável pela inicialização de cada novo cliente no mundo virtual multiusuário. Ele também é responsável por transmitir o estado corrente da cena 3D para clientes recém conectados, assim como por enviar mensagens de atualização com respeito a posição e orientação do avatar no mundo virtual multiusuário.

O middleware JADE [47] foi utilizado como plataforma para a simulação da arquitetura multiagentes de RECOLLVE. Objetos e Avatares são tratados da mesma forma devido às facilidades de comunicação entre os agentes fornecidos pela linguagem padrão ACL[46].

4.2.1 Agente Usuário

No sistema RECOLLVE, os usuários são representados na cena virtual por agentes com uma representação 3D, ou seja, por um avatar e seu respectivo Agente Usuário. A Figura 4.3 mostra a arquitetura interna do Agente Usuário. Um Agente Usuário é composto basicamente por:

- unidade central do usuário - responsável por tratar da interação com o usuário e selecionar a ação escolhida pelo usuário para ser executada pelo agente;
- módulo de mensagens - responsável por manipular as mensagens oriundas do Agente Objeto e de outro Agente Usuário e enviar mensagens com destino ao Agente Objeto e Agente Usuário;
- biblioteca de ações - conjunto de ações disponíveis para todos os Agentes Usuários.

O Agente Usuário segue um modelo semiautônomo bastante simples de agente: quando um usuário seleciona um objeto ou outro avatar, o Agente Usuário que representa aquele usuário cria e envia uma mensagem para o Agente Objeto ou outro Agente Usuário informando seu perfil. Quando o Agente Usuário recebe uma mensagem do Agente Objeto ou de outro Agente Usuário, ele apresenta o conteúdo dessa mensagem ao usuário. Seu comportamento pode ser ilustrado pela Rede de Petri mostrada na Figura 4.4.

4.2.2 Agente Objeto

Os Objetos, assim como os usuários, são representados na cena virtual por meio de agentes com representação 3D, ou seja, uma descrição 3D de um objeto e do seu

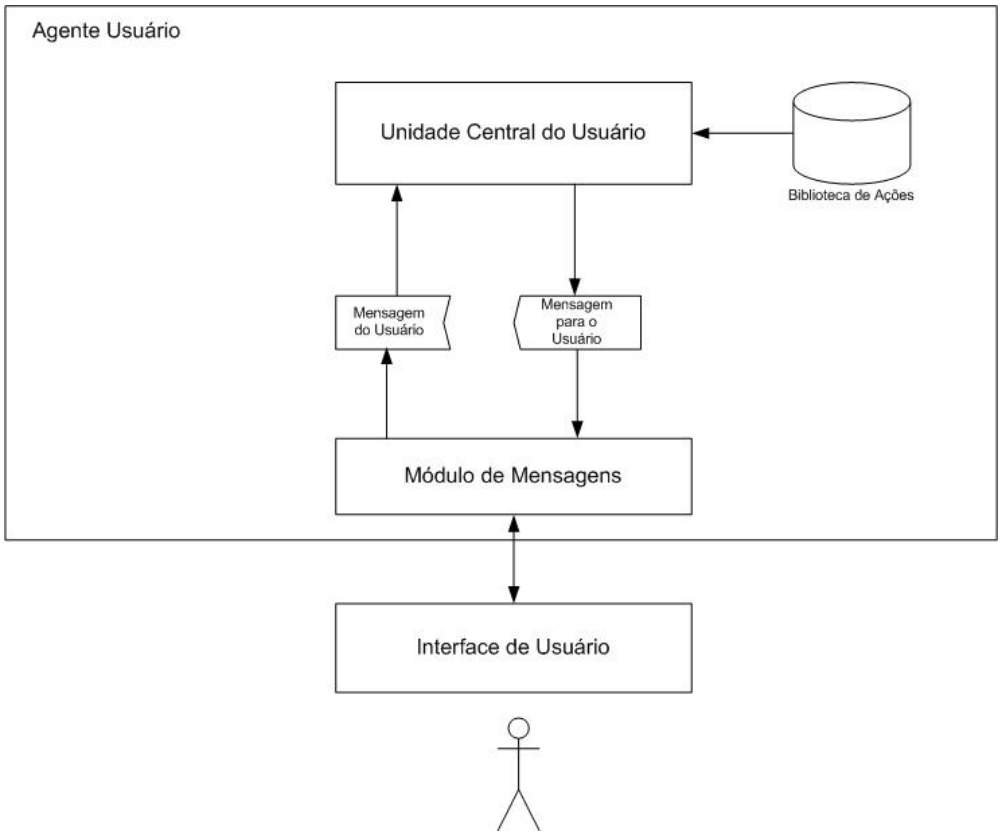

Figura 4.3: Arquitetura do Agente Usuário.

respectivo Agente Objeto. Os Agentes Objetos são reativos e definem quais as ações um avatar, representando um usuário, pode realizar sobre ele, baseado em regras previamente definidas. A Figura 4.5 mostra a arquitetura interna do Agente Objeto.

Um Agente Objeto é composto por:

- unidade central do objeto - responsável pelo processamento das regras. As regras, por ora, são definidas com base em três parâmetros: i) papel do usuário; ii) sala virtual em que o usuário se encontra; iii) posse de um determinado objeto. Após o processamento das regras executadas pelo Agente Objeto, uma mensagem é enviada ao Agente Usuário informando-lhe quais são as ações que estão disponíveis para ele naquele momento;

- módulo de mensagens - responsável por manipular as mensagens oriundas do

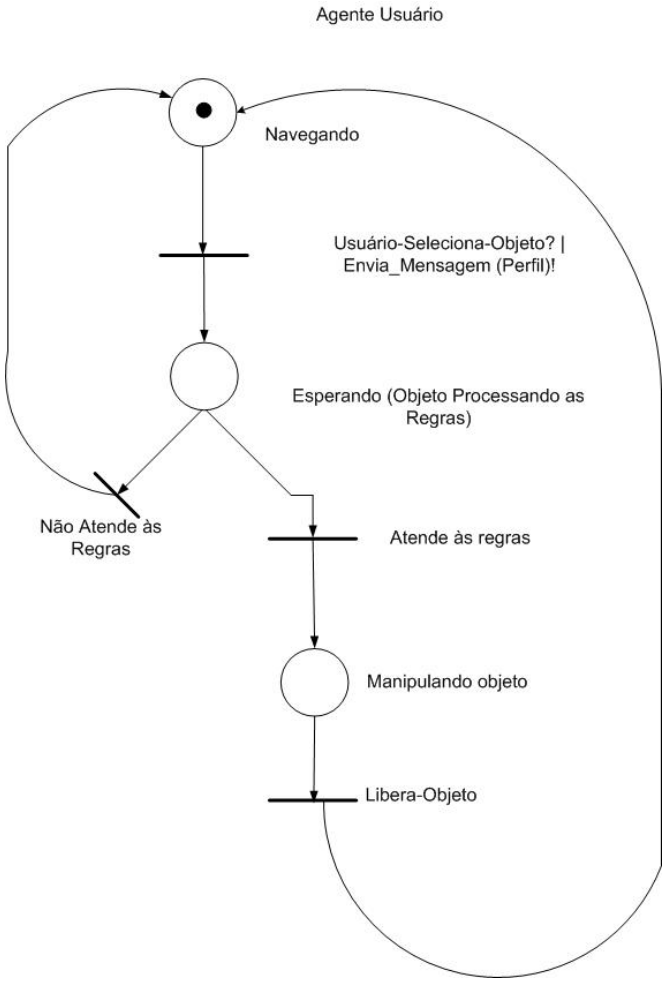

Figura 4.4: Rede de Petri que representa o Agente Usuário.

Agente Usuário e enviar mensagens com destino ao Agente Usuário;

- comportamentos - o Agente Objeto apresenta dois estados: livre e bloqueado. O estado livre indica que aquele objeto pode ser manipulado. O estado bloqueado indica que aquele objeto está sendo manipulado. Esse esquema permite fazer um controle de acesso, evitando que dois ou mais usuários consigam manipular o mesmo objeto simultâneamente. O comportamento do Agente Objeto é descrito pela Rede de Petri apresentada na Figura 4.6;

A Figura 4.7 mostra a interação entre um avatar e um Agente Objeto. O avatar envia uma mensagem para o Agente Objeto que ele quer manipular informando seu

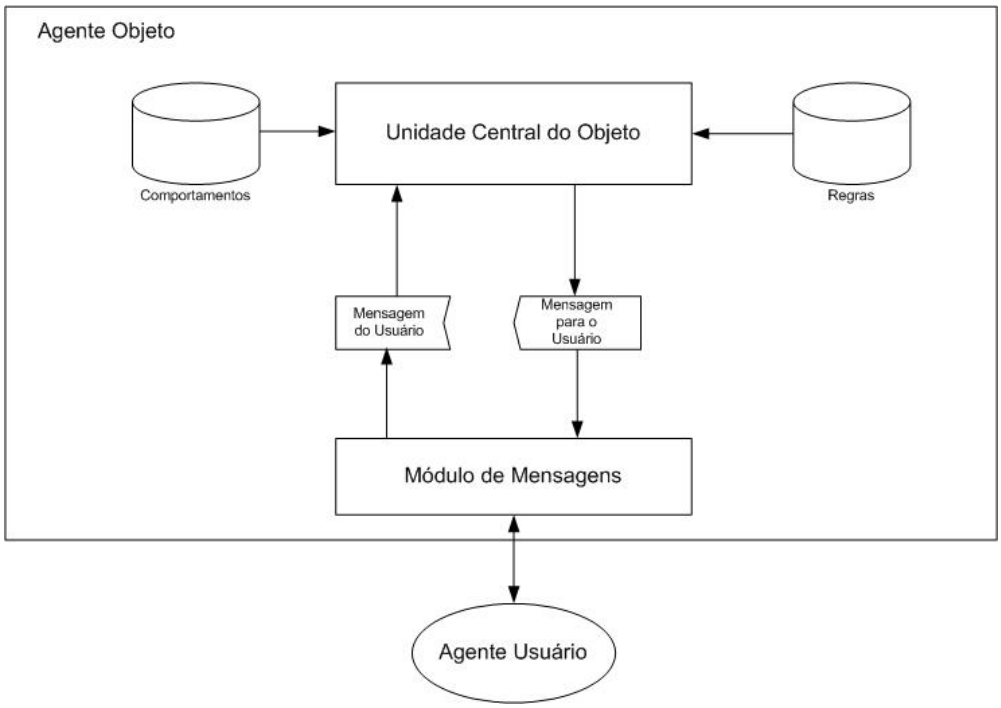

Figura 4.5: Arquitetura do Agente Objeto.

perfil. O Agente Objeto, baseado em regras previamente estabelecidas, avalia que ações aquele avatar pode realizar sobre ele. O Agente Objeto então, envia de volta para o avatar uma mensagem contendo o conjunto de ações disponíveis. O Agente Usuário ao receber a mensagem do Agente Objeto, apresenta as ações disponíveis para o usuário, que faz a sua escolha. Dessa forma, é possível transcrever para a cena virtual, os direitos de acesso dos usuários, de acordo com seu perfil.

Quando um objeto é manipulado por um avatar, ele permanece bloqueado para outros avatares.

Em nosso modelo de Agente, cada Agente tem uma *mailbox* onde as mensagens dos usuários são armazenadas e o agente é notificado. O Agente Objeto então retira a mensagem da fila de mensagens para processá-la. Assim, dois ou mais Agentes Usuários não podem manipular o mesmo objeto simultâneamente.

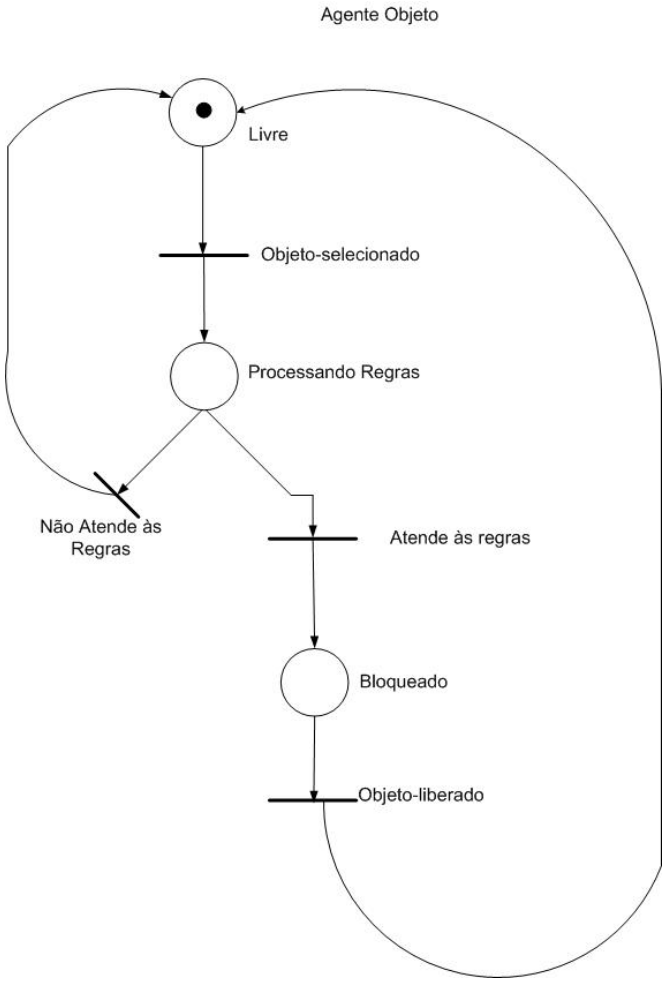

Figura 4.6: Rede de Petri que representa o Agente Objeto.

4.2.3 Agente Gerente

O Agente Gerente coordena e monitora as ações dos usuários e dos outros agentes.

A Figura 4.8 mostra a arquitetura interna do Agente Gerente.

Ele é composto basicamente por:

- uma unidade de gerenciamento;
- uma base de comportamentos;
- módulo de mensagens;
- um conjunto de regras.

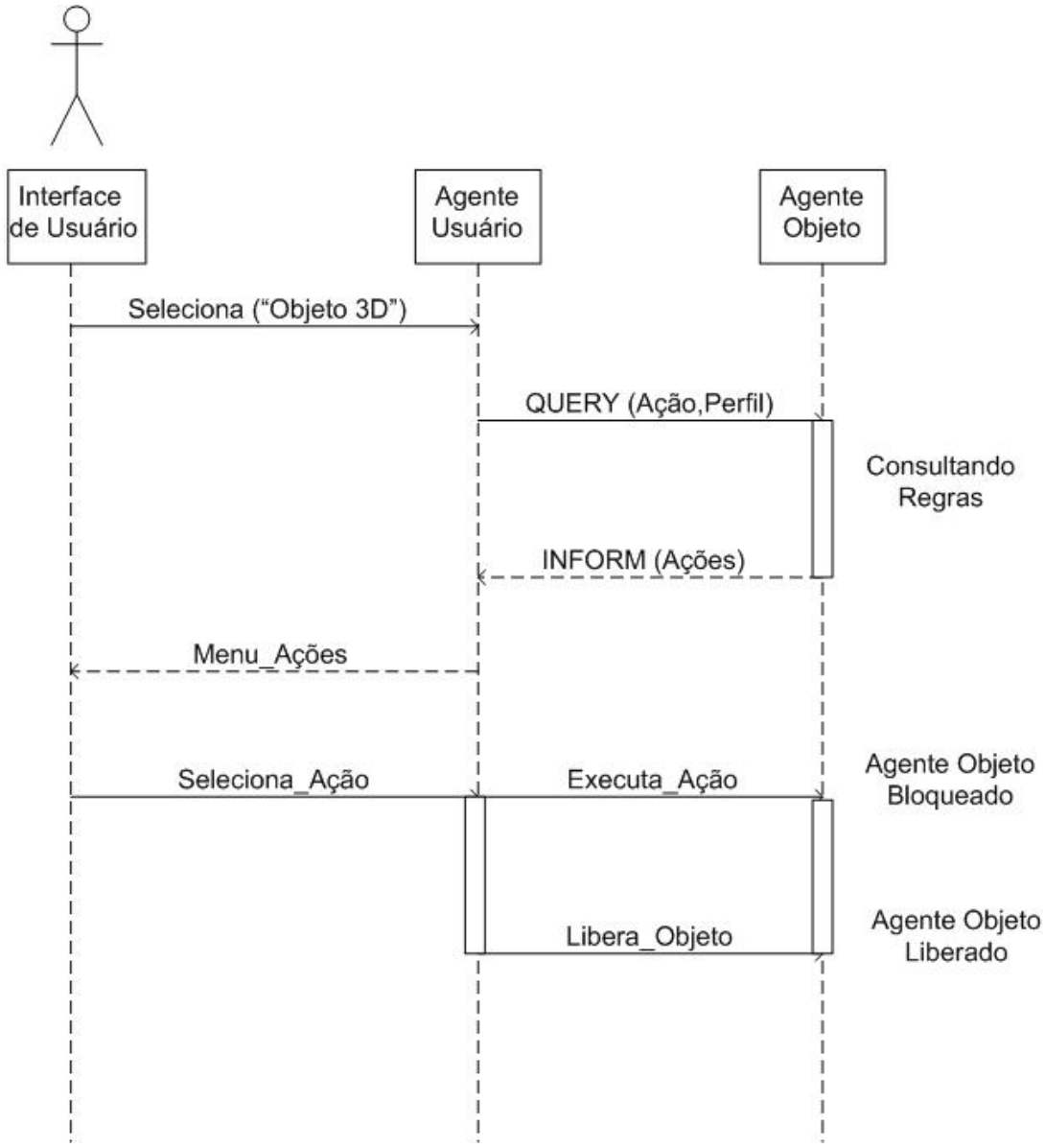

Figura 4.7: Diagrama de Sequência: interação entre Agente Usuário e Agente Objeto.

O Agente Gerente é reativo. Ele recebe mensagens dos outros agentes por meio de sensores presentes na cena virtual. Tais mensagens possuem no seu conteúdo as ações realizadas pelo usuário. O Agente Gerente identifica o tipo de ação que o usuário realizou, em seguida, o módulo Gerenciador de Comportamentos, baseado em regras, seleciona o comportamento adequado.

As principais ações do Agente Gerente que determinam seu comportamento são:

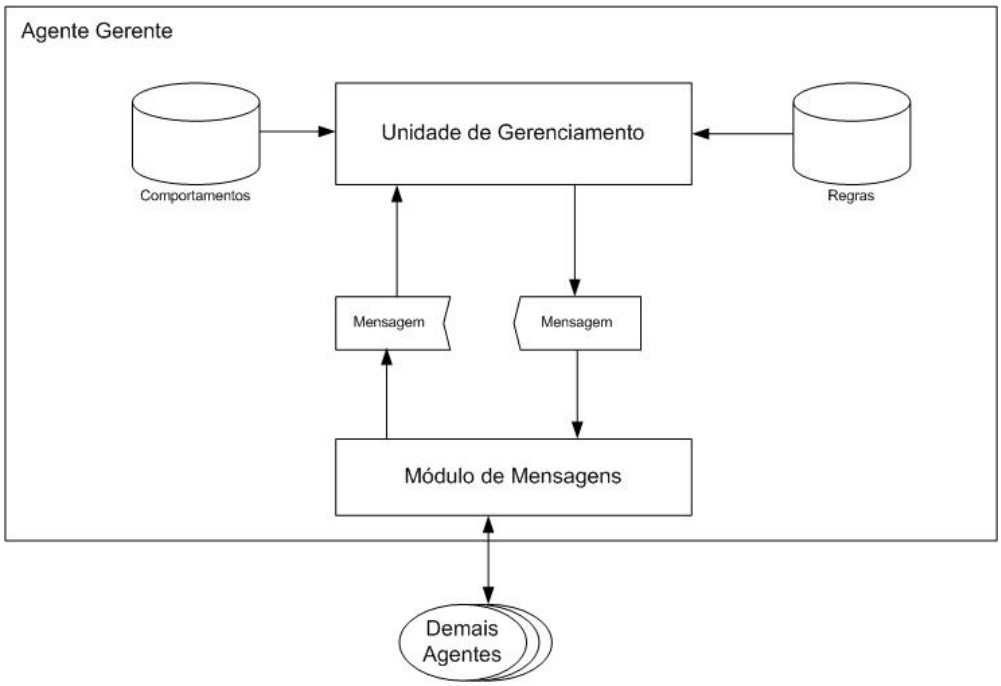

Figura 4.8: O Agente Gerente.

- acompanhar a navegação do usuário;
- solicitar ao Agente Sync a sincronização em todos os clientes sempre que ocorrer alguma modificação na cena virtual;
- solicitar ao Agente Sync a atualização do ambiente inserindo ou removendo um objeto da cena virtual;
- solicitar ao Agente Sessão a inclusão ou exclusão de um usuário que deseje se juntar ou deixar a sessão virtual respectivamente.

As mensagens trocadas entre o Agente Gerente e a Interface de Usuário, ilustradas na Figura 4.9, são:

- receber ações dos usuários;
- enviar ambiente para o usuário;

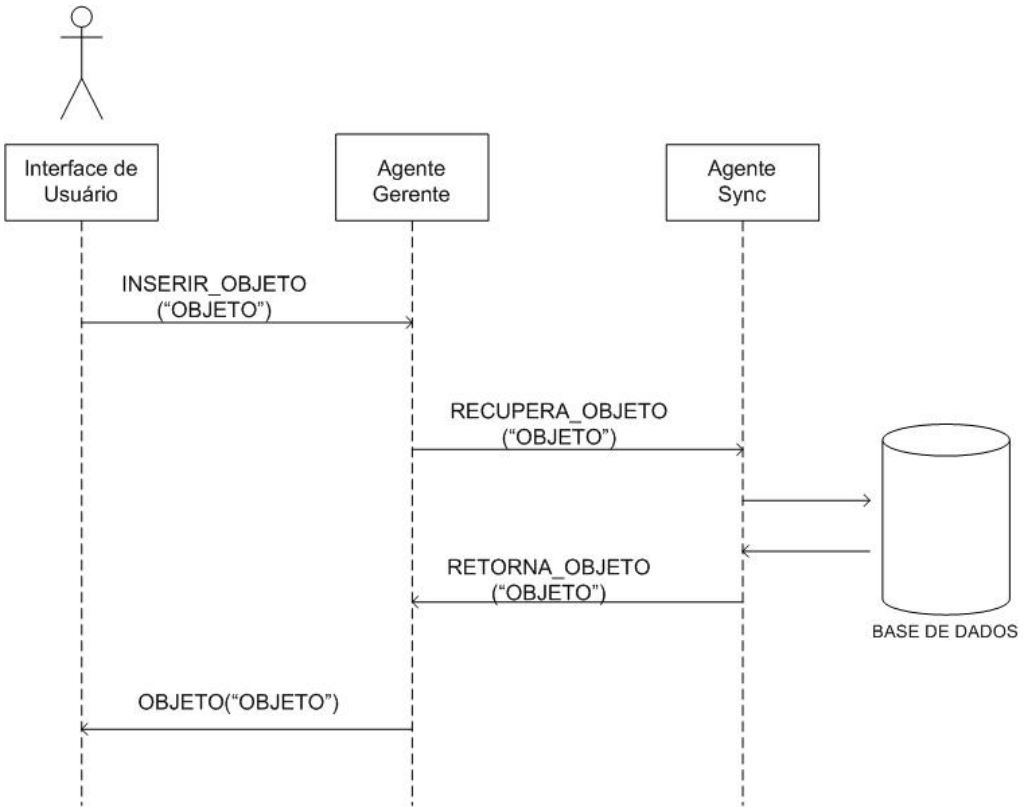

Figura 4.9: Diagrama de Sequência: interação entre Agente Gerente e Interface de Usuário.

- enviar objetos 3D para atualização do ambiente.

As mensagens trocadas entre o Agente Gerente e o Agente Sessão, ilustradas na Figura 4.10, são:

- verificar login e senha;
- armazenar cadastro do novo usuário.

As mensagens trocadas entre o Agente Gerente e o Agente Sync, já ilustradas na Figura 4.9, são:

- gerar ambiente para o usuário;
- recuperar objetos para atualização do ambiente.

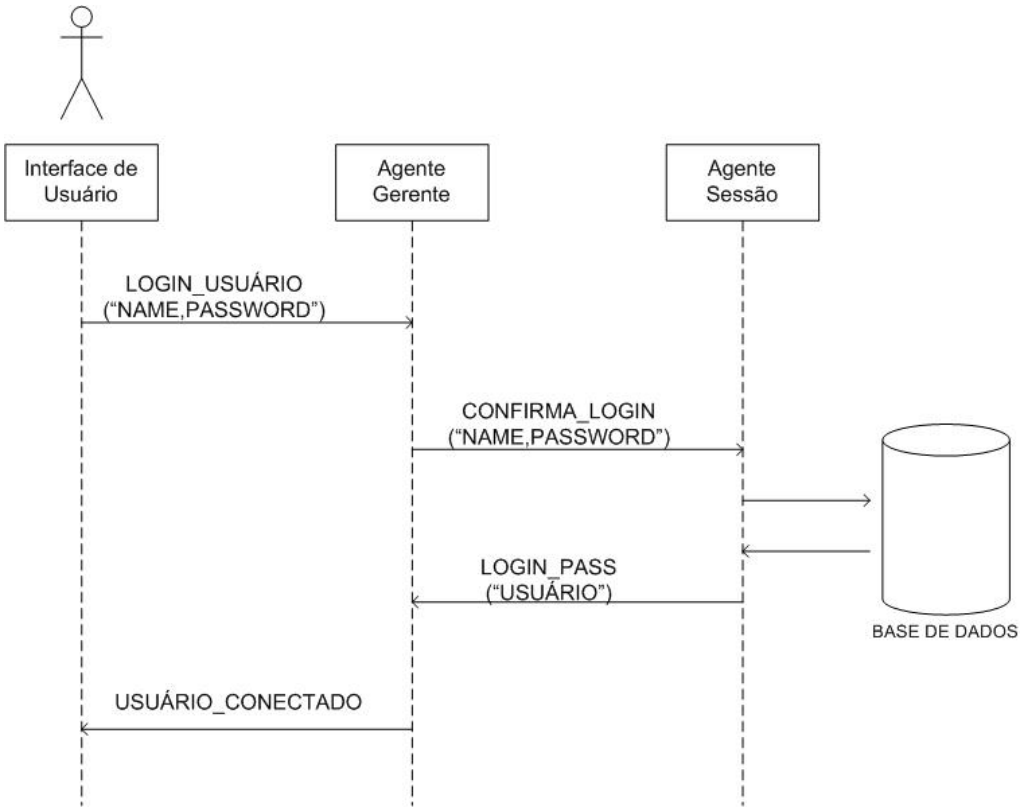

Figura 4.10: Diagrama de Sequência: interação entre Agente Gerente e Agente Sessão.

As mensagens trocadas entre o Agente Gerente e o Agente Integração, ilustradas na Figura 4.11, são:

- solicitar a realização de uma ação em aplicação externa;
- recuperar objetos para atualização do ambiente;
- remover objetos do ambiente.

4.2.4 Agente Integração

O Agente Integração é responsável por toda e qualquer interação com o ambiente externo. Ele atende requisições de aplicações externas e então solicita ao Agente Gerente que realize alguma ação. Após ocorrer algum evento em particular ou

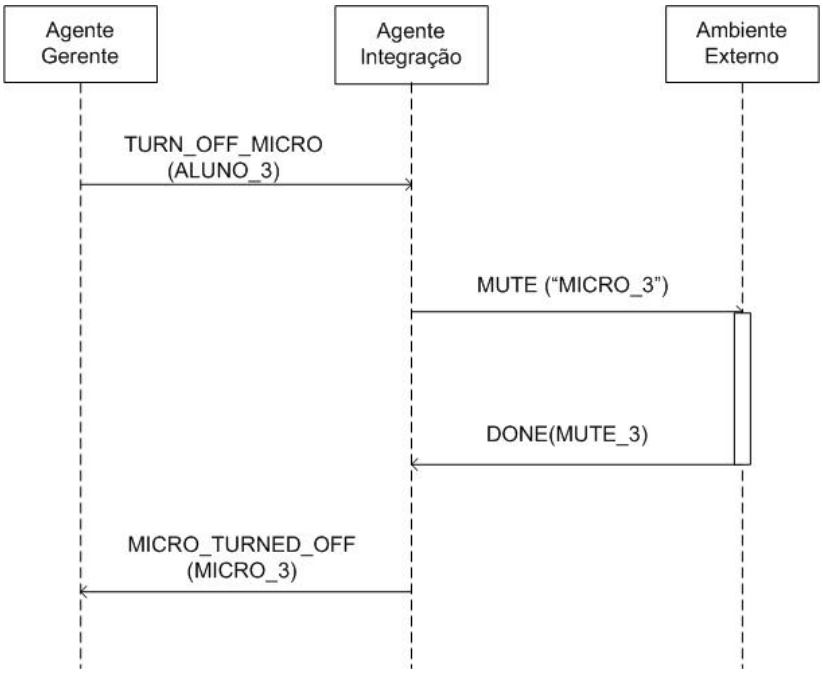

Figura 4.11: Diagrama de Sequência: interação entre Agente Gerente e o Agente Integração.

uma combinação de eventos na cena virtual, e de acordo com regras previamente definidas, o Agente Integração solicita às aplicações externas a realização de uma determinada ação.

Todas as funcionalidades disponibilizadas pela API (*Application Program Interface*) das aplicações externas devem ser acessadas por meio do Agente Integração.

O Agente Integração é próativo, pois ele é capaz de perceber a intenção dos usuários de realizar uma determinada ação em grupo, e se antecipa, por meio de regras, efetuando ações que darão suporte aos usuários para realizarem a ação desejada. Por exemplo, o posicionamento de dois ou mais avatares em torno de uma mesa, pode ser o suficiente para o Agente Integração solicitar ao Ambiente de Integração que seja iniciada uma sessão de áudioconferência entre aqueles usuários.

A Figura 4.12 ilustra a arquitetura interna do Agente Integração, que é composto por uma unidade de integração, uma base de comportamento e um conjunto de

regras. A unidade de integração processa as mensagens recebidas de acordo com as regras definidas e seleciona o comportamento mais apropriado para cada situação.

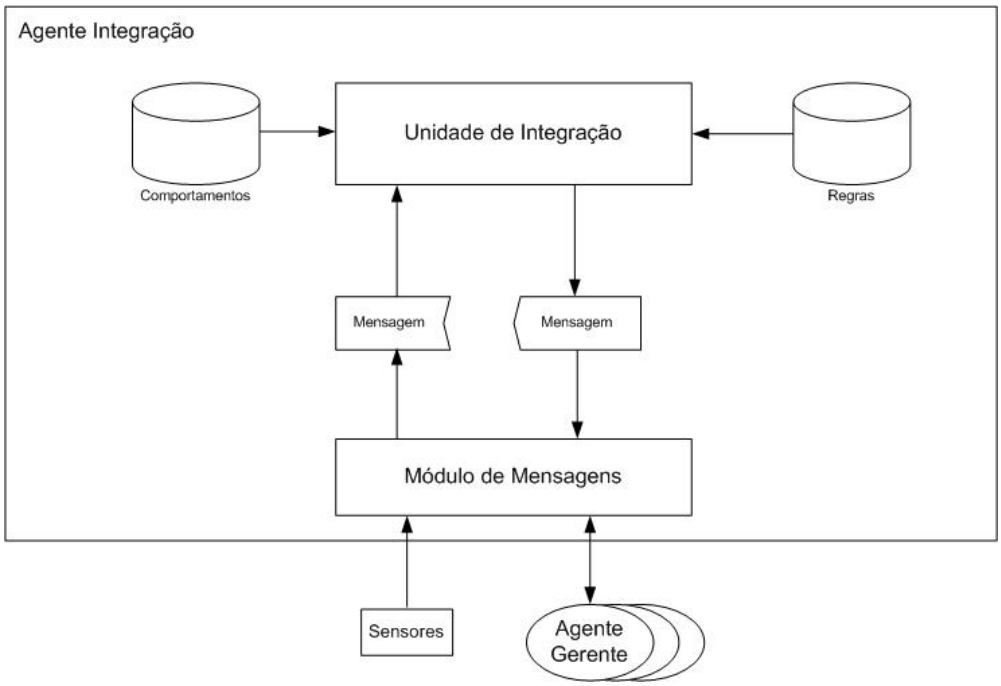

Figura 4.12: O Agente Integração.

O Agente Integração torna o sistema RECOLLVE bastante flexível no que diz respeito ao gerenciamento das aplicações colaborativas integradas ao sistema, pois é possível mapear diferentes configurações da cena virtual em funcionalidades presentes na API de uma aplicação.

O Agente Integração recebe mensagens dos sensores presentes na cena virtual, do Agente Usuário, do Agente Gerente e da Aplicação Externa. A Figura 4.11 ilustra a troca de mensagens ocorridas quando o Agente Gerente solicita ao Ambiente Externo que realize uma determinada ação.

A troca de mensagens ilustrada na Figura 4.11 poderia servir para representar o cenário onde um usuário participando de uma sessão de áudioconferência cede a palavra a outro usuário. Para representar tal estado na cena virtual, o Agente Ge-

rente solicita ao Agente Integração que o microfone de um determinado participante seja desligado. O Agente Integração, por sua vez, solicita ao Ambiente Externo que o aplicativo envolvido, Skype por exemplo, acione a função MUTE daquele usuário.

4.2.5 Agente Sync

O Agente Sync é um agente reativo, responsável por manter a cena virtual sincronizada para todos os clientes. Ele fornece o ambiente inicial para todos os usuários que acabam de se conectar, além de ser responsável por fazer as atualizações na cena, adicionando ou removendo objetos, em resposta a solicitação do Agente Gerente.

4.2.6 Agente Sessão

O Agente Sessão também é um agente reativo, responsável por gerenciar a sessão virtual colaborativa. Responsável por adicionar novos usuários à sessão virtual e por remover aqueles usuários que desejam deixá-la. Ele também é responsável por validar o login e senha dos usuários.

4.2.7 Interface de Comunicação Cliente/Servidor

A função da interface de comunicação é capturar as ações dos usuários e comunicar estas ações ao Agente Gerente. Do lado cliente, o usuário pode realizar funções como conectar-se a uma determinada sessão virtual colaborativa, inserir e remover objetos se seu perfil permitir.

Do lado servidor, quando uma ação do usuário é recebida, o Agente Gerente se encarrega de determinar quais devem ser as ações a executar. Estas ações são transformadas em mensagens e enviadas aos demais agentes: Agente Sessão, Agente

Sync e Agente Integração. O sistema multiagentes toma as decisões necessárias e retorna à interface, a atualiza ção da cena virtual, quando necessário ou solicita ao Agente Integração que uma determinada ação externa seja realizada.

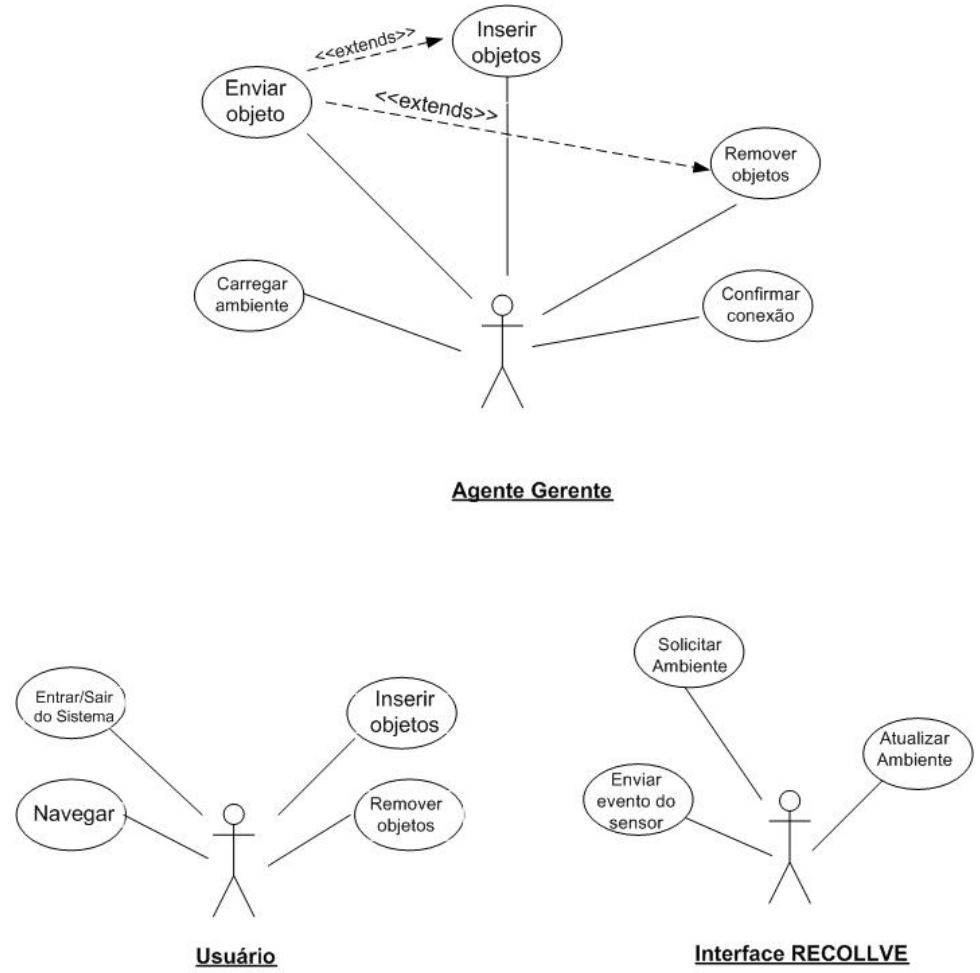

Figura 4.13: Diagramas dos Casos de Uso da comunicação com o usuário.

A comunicação do Agente Gerente com a Interface de usuário, ilustrada pelos casos de uso da Figura 4.13 é realizada pelas seguintes ações:

- confirmar login de usuário: responsável por gerar o ambiente para um usuário já cadastrado, após ter sido realizada a autenticação no sistema; para o sistema, este caso de uso fornece o ID do usuário e o seu *Perfil de Usuário*;
- inserir objeto: este caso de uso é responsável pela inserção de objetos na cena

virtual;

- remover objeto: este caso de uso é responsável por remover objetos da cena virtual;

4.3 Considerações Finais

Este Capítulo apresentou a arquitetura multiagentes de RECOLLVE, descrevendo cada um dos agentes participantes da sociedade, e como eles se inter-relacionam. A adoção do paradigma de agentes para modelar o sistema RECOLLVE, conferiu ao sistema uma grande flexibilidade na coordenação e representação das atividades colaborativas.

Capítulo 5

Implementação e Estudos de Caso

Este Capítulo apresenta detalhes da implementação do sistema RECOLLVE, assim como alguns estudos de caso. O sistema RECOLLVE foi concebido para ser uma ferramenta de código aberto, independente de plataforma e poder utilizar recursos computacionais de baixo custo para executar aplicações de Realidade Virtual. Segue o padrão VRML/X3D para a descrição de objetos 3D, facilitando assim sua utilização como plataforma de suporte para uma ampla gama de aplicações colaborativas.

A próxima seção trata dos aspectos de implementação de RECOLLVE.

5.1 Aspectos da Implementação de RECOLLVE

O protótipo desenvolvido apresenta uma arquitetura Cliente-Servidor, como ilustrado na Figura 5.1. A opção por uma arquitetura Cliente-Servidor se deve à sua facilidade de implementação, uma vez que o objetivo principal do desenvolvimento do protótipo é dispor de uma plataforma para realização dos estudos de caso no

ambiente RECOLLVE.

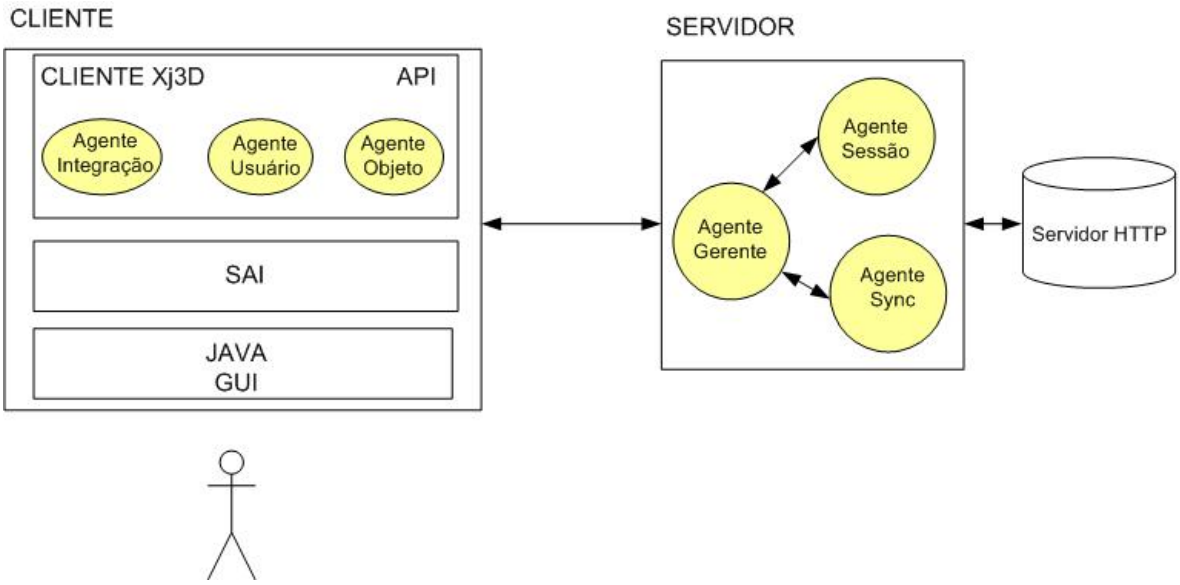

Figura 5.1: Implementação do Sistema RECOLLVE.

5.1.1 O Cliente

No lado cliente são executados o Agente Usuário, o Agente Objeto e o Agente Integração. O Cliente é composto de uma interface de usuário desenvolvida em Java, de um browser Xj3D[1], da SAI (*Scene Access Interface*) e da API definida no Capítulo 4.

O código do cliente está estruturado como ilustrado na Figura 5.2. Uma versão mais leve do cliente pode ser construída, visando sua utilização em um PDA, por exemplo. Outra característica presente no cliente é que outros meios de interação com a cena também são permitidos, como Luvas 3D. Bastando apenas redirecionar os dados da entrada externa para as ações internas do usuários.

A interface, ilustrada na Figura 5.3 é utilizada pelos usuários para se conectarem

[1]http://www.xj3d.org

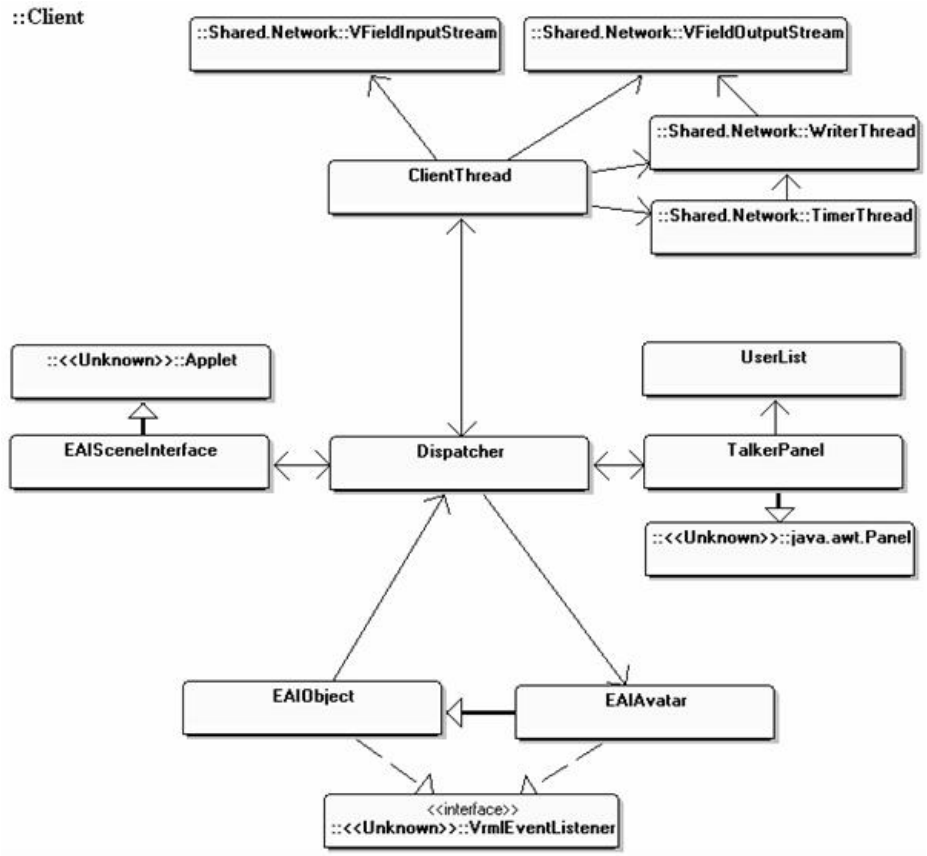

Figura 5.2: Diagrama de classes do Cliente RECOLLVE.

na sessão virtual.

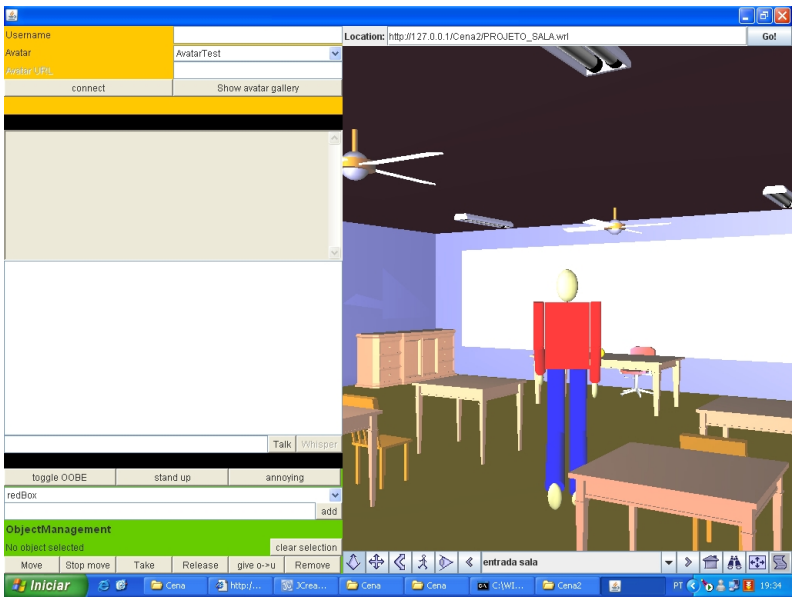

Figura 5.3: Interface do Cliente.

A partir da Interface, o usuário pode escolher seu avatar, adicionar e remover objetos se seu perfil permitir.

5.1.2 O Servidor

No lado Servidor são executados os Agentes Gerente, Sessão e Sync. O Servidor RECOLLVE é responsável por adicionar novos usuários para a sessão virtual e por remover aqueles que desejam deixar a sessão virtual. Além disso, ele também é responsável por tratar as requisições de conexão e validar login e senha de usuários. É responsável por manter a cena virtual sincronizada em todos os clientes e pela inicialização de cada novo cliente no mundo virtual multiusuário. Ele também é responsável por transmitir o estado corrente da cena 3D para clientes recém conectados, assim como por enviar mensagens de atualização com respeito a posição e orientação do avatar no mundo virtual multiusuário.

O código do servidor está estruturado como ilustrado na Figura 5.4.

Gerenciamento da base de dados persistente

Motivado pelo fato de que *VRML* utiliza um grafo de cena hieráquico composto de nós para estruturar os objetos. Em RECOLLVE, cada objeto do mundo virtual é representado por uma entidade. As entidades são agrupadas na forma de uma hierarquia similar àquela utilizada por *VRML*. Cada entidade é definida por um grupo de atributos. Uma parte desses atributos serve a identificar a entidade graça a um número único e a um nome. A outra parte serve a definir a posição da entidade na hierarquia corrente da cena virtual. A última parte dos atributos é composta de campos representando a fonte de dados definindo o objeto *VRML* e o estado dos

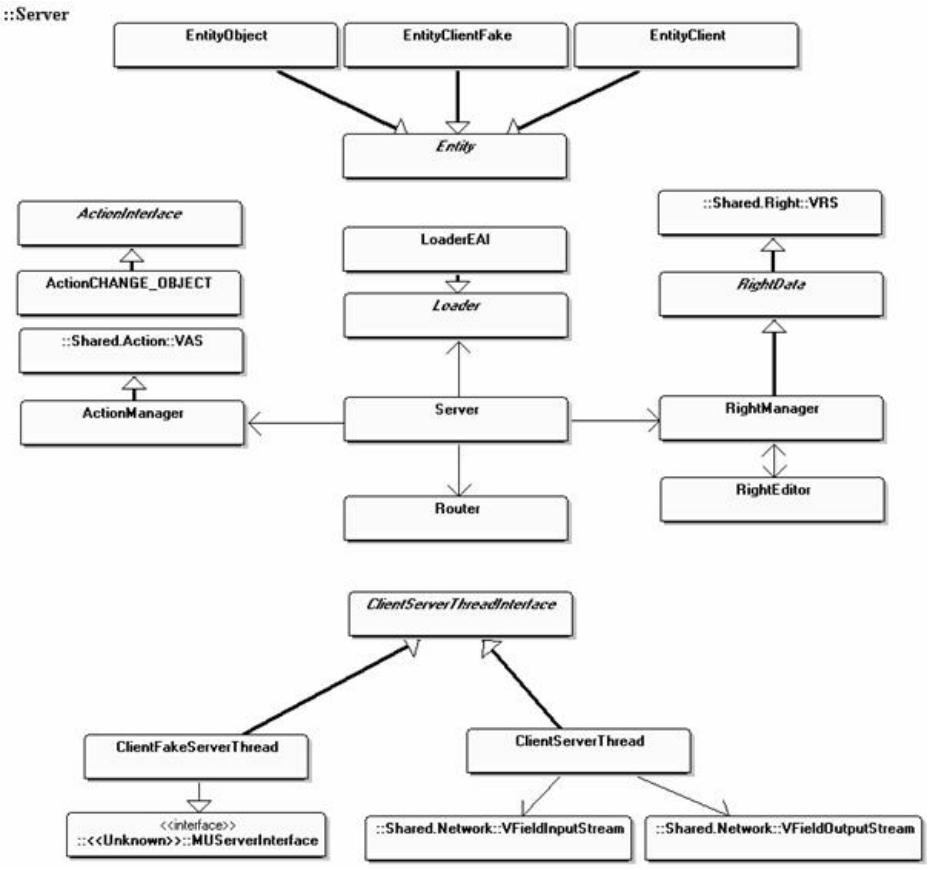

Figura 5.4: Diagrama de classes do Servidor RECOLLVE.

campos com os quais a SAI/EAI se comunica.

O reagrupamento dos elementos constituintes da cena virtual é a base de dados das entidades do mundo virtual. Um aspecto importante dessa base de dados é a sua perenidade. Qualquer mudança de uma entidade é atualizada na base de dados.

Gerenciamento dos clientes

A primeira tarefa do Servidor é gerenciar os clientes que se conectam a ele. Quando um cliente se conecta ao sistema, o Servidor atualiza a base de dados contendo as informações concernentes a todos os clientes conectados. Em seguida, o Servidor informa a todos os outros clientes a mudança ocorrida, a fim de que o estado da

sessão seja atualizado em cada um dos clientes conectados. Por fim, ele fornece ao novo cliente uma representação do mundo virtual neste dado instante.

A seguir, é apresentada de que maneira as mudanças na base de dados persistentes das entidades são sincronizadas.

Sincronização dos eventos

O modelo de sincronização adotado é simples. Quando uma mudança no estado de uma entidade ocorre, esta mudança é capturada graças à SAI/EAI e transmitida ao Servidor. O Servidor, por sua vez, atualiza a sua base de dados e transmite a atualização a todos os clientes conectados. A SAI/EAI, agora, permite a ação inversa modificando o estado de um nó do grafo de cena conforme a informação recebida.

Comunicação entre Clientes e o Servidor

Clientes e Servidor utilizam conexões TCP ponto-a-ponto para efetuar a comunicação entre sí. O Servidor dispõe de uma classe *ServerSocket* que espera um pedido de conexão de um cliente. Uma vez que a conexão é efetuada, um *Socket* é criado, e o Servidor cria uma *Thread*, o *ClientServerThread*, para tratar o *Socket* cliente. O *ClientServerThread* tem seu diagrama de sequência apresentado na Figura B.4 do apêndice. O Cliente também dispõe de uma *Thread* dedicada à escuta do *Socket*. Esses dois *Sockets* pertencem a um mesmo módulo agrupado no pacote *Shared.Network*. Este pacote é composto de classes que permitem a criação de mensagens e sua transmissão utilizando métodos das classes *DataInputStream* e *DataOutputStream*.

Um dos aspectos importantes do pacote *Shared.Network* é sua política de trata-

mento de mensagens, descrita a seguir.

Fila de Espera Adaptativa

Quando o servidor ou cliente querem enviar uma mensagem, esta é colocada em uma fila de espera, definida no pacote *Shared.Network*. A fila de espera desempenha o papel de buffer para a classe *DataOutputStream* do *Socket*. As mensagens, dependendo do seu tipo, recebem tratamentos diferentes:

- todas as mensagens cujo campo *Field* é negativo - as mensagens administrativas - são enviadas instantaneamente e em ordem. As outras mensagens - aquelas puramente de sincronização - são colocadas em uma fila de espera intermediária e são enviadas periodicamente graças a um temporizador;

- a fila de espera é importante no tratamento dado às mensagens de sincronização. Esse tipo de mensagem só é enviada uma vez a cada período do temporizador, dessa forma, a fila de espera adaptativa se propõe a diminuir a carga na rede. No caso em que uma entidade deseja emitir duas mensagens no mesmo período do temporizador, apenas a segunda mensagem tem efeito. Essa estratégia não afeta o resultado final na cena virtual dos demais clientes. Apenas a fluidez do deslocamento é afetada proporcionalmente ao período do temporizador.

Mundos VRML multiusuários em RECOLLVE

No Capítulo 3 foi apresentado um estudo sobre as abordagens utilizadas para viabilizar a implementação de mundos VRML multiusuários. Em RECOLLVE, foi utilizada uma solução *ad-hoc*. Eventos gerados por entidades na cena virtual são

capturadas por X3D/SAI [63] e encaminhados para uma tabela de roteamento localizada no Servidor, como mostrado na Figura 5.5. Esta tabela de roteamento permite definir quais *eventIn* de quais entidades receberão o evento produzido na cena virtual. Da perspectiva de VRML, este procedimento é transparente, mas da perspectiva do sistema, a tabela de roteamento armazena informações que permitem diferenciar as entidades segundo os clientes.

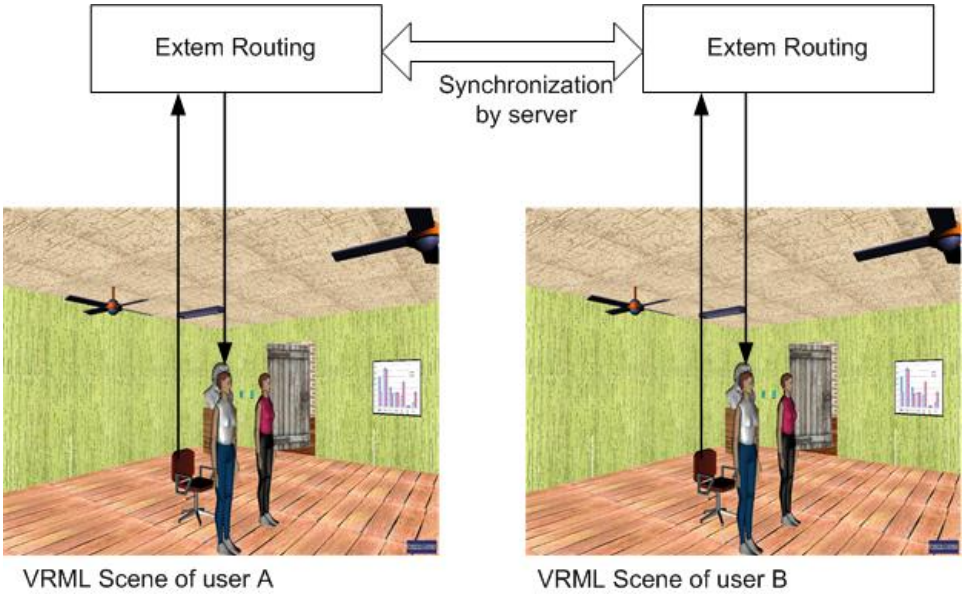

Figura 5.5: Sincronização dos eventos realizada pelo Servidor.

A Figura 5.6 mostra o caso do clic sobre a cadeira. O evento produzido pelo clic sobre a cadeira permite a Tabela de roteamento compor um par identificador, que é composto por um ID, único identificador de cada entidade e um index que identifica um *EventOut* para aquela entidade. Depois, a Tabela será inicializada com todos os pares possíveis. O valor inicial do evento será enviado para a entidade correta graças a esses pares. Neste caso, somente o avatar do usuário A que realizou o clic receberá o comando para iniciar a animação "sentar na cadeira", como mostrado na Figura 5.7.

A seguir são apresentados estudos de caso mostrando o potencial de RECOLLVE

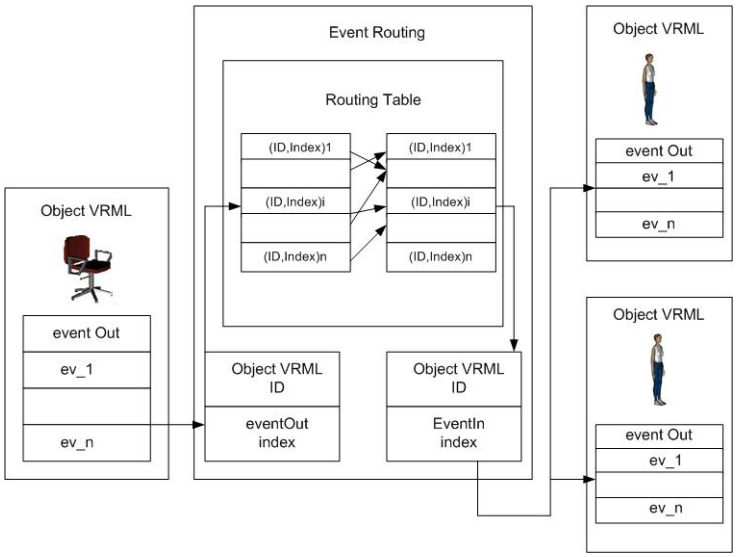

Figura 5.6: Tabela de roteamento.

Figura 5.7: Sincronização de eventos - situação desejada.

para suportar atividades colaborativas.

5.2 Estudos de Caso

Os estudos de caso apresentados nesta seção tem o objetivo de mostrar o potencial de RECOLLVE para suportar percepção em atividades colaborativas.

5.2.1 Primeira Experiência

A primeira experiência de integração do AVC com outra aplicação colaborativa, através de LEICA, é mostrada na Figura 5.8. Eventos ocorridos no interior da

cena virtual provocam consequências em COLAB [68], da mesma forma, eventos ocorridos em COLAB [68] têm consequências em RECOLLVE. Neste cenário, é realizada a representação de uma sessão de navegação colaborativa usando o sistema COLAB. COLAB é um sistema de navegação colaborativa onde usuários podem criar e destruir todas as relações de sincronismo entre os usuários de forma dinâmica e distribuída. Um cliente COLAB é um Applet JAVA que sincroniza e apresenta páginas Web. Desta forma, COLAB permite aos usuários: navegar conjuntamente na Web (um usuário controla a navegação) ou usuários navegarem separadamente na Web, mas tendo a percepção da navegação dos outros usuários.

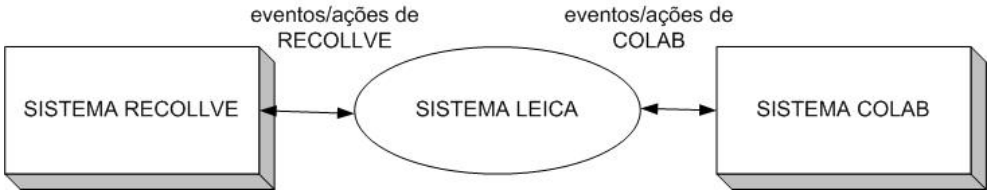

Figura 5.8: Integração de RECOLLVE e COLAB através de LEICA.

Para representar uma sessão colaborativa em RECOLLVE, o primeiro passo é escolher estados que devem ser representados, e as metáforas do mundo real que serão associadas a essas representações. Em COLAB, dois principais aspectos foram escolhidos:

- quando a navegação é síncrona, um usuário a controla (o líder) e os outros usuários o seguem. Este estado é representado por uma entidade (Objet 3D) que indica a URL atual. Quando a navegação é assíncrona, a entidade desaparece.

- *Floor* - este termo indica um único atributo associado ao usuário, que permite controlar a navegação colaborativa. Este atributo é representado por um

objeto 3D (um token). Regras de acessibilidade podem ser definidas para a manipulação do Token, limitando assim, o acesso dos usuários.

Quando COLAB alterna entre os modos síncrono e assíncrono, o Agente Gerente faz um objeto 3D representando o Token aparecer e desaparecer da cena virtual, respectivamente. Quando o usuário modifica a URL no seu browser, o texto indicador na cena virtual também é modificado.

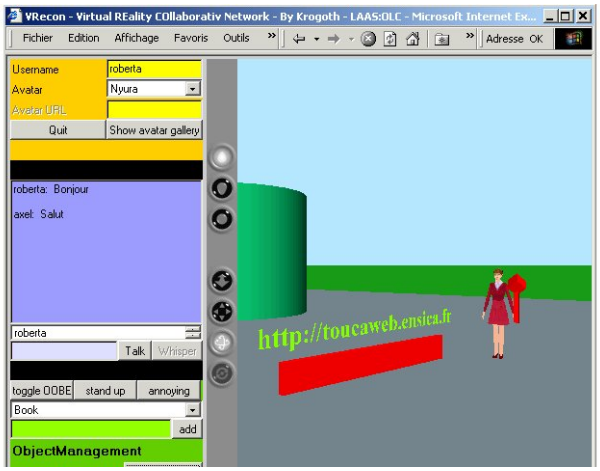

Figura 5.9: Representando uma sessão Colab na cena virtual.

A Figura 5.9 mostra uma interface COLAB-Página Web, onde o texto indicador é mostrado na cena virtual. Em COLAB, quando o usuário passa o *floor* para outro usuário, o Token 3D será passado também na cena virtual. Finalmente, se um usuário passa o Token 3D na cena virtual, o *floor* também será passado em COLAB.

Considerando um curso a distância como um cenário de aplicação, ilustrado na Figura 5.10, o professor quer mostrar uma página Web para os estudantes e explicar um conteúdo, ou ele quer mostrar uma experiência. O professor então pode navegar através da Web e os alunos recebem a mesma URL que é mostrada na sala virtual. O avatar representando o professor detém o Token 3D, simbolizando que ele tem o controle da navegação naquele momento.

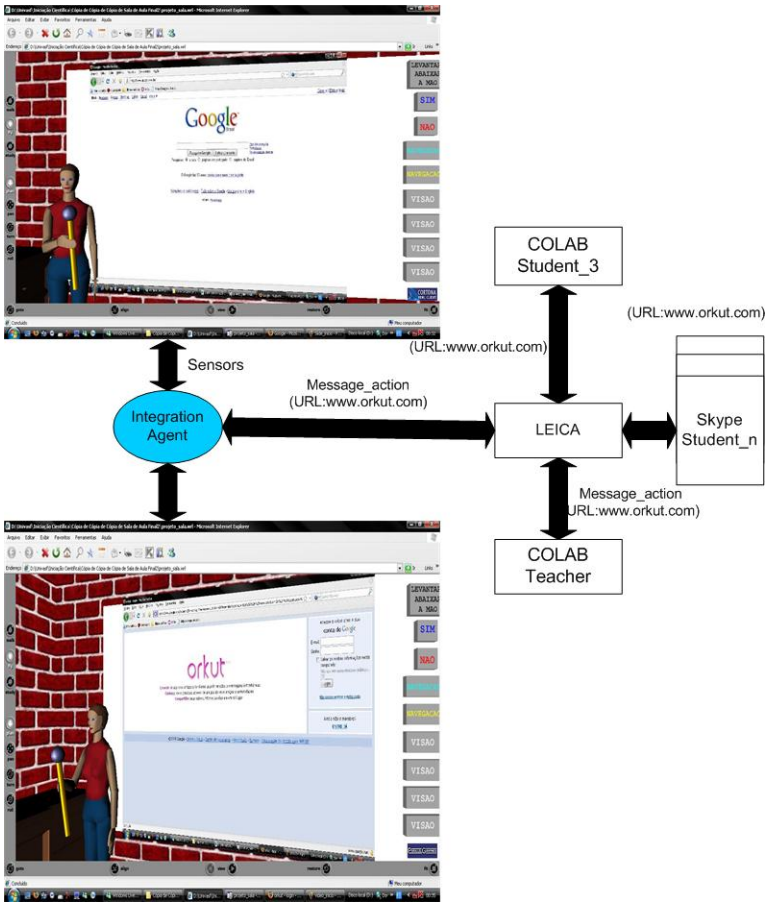

Figura 5.10: Representando uma sessão Colab na cena virtual.

5.2.2 Segunda Experiência

A segunda experiência, ilustrada na Figura 5.11, mostra a integração de três aplicações colaborativas. O protótipo de RECOLLVE, um sistema de Chat e um Editor Compartilhado são integrados pelo sistema LEICA [4].

Nos sistemas de Chat, usuários compartilham experiências e opiniões por troca de mensagens. Representar este tipo de aplicação em uma cena virtual, aumenta a percepção entre os usuários.

Editor de texto compartilhado é um tipo de ferramenta que permite a dois ou mais usuários editarem conjuntamente um mesmo documento. O principal problema no emprego deste tipo de aplicação é gerenciar tarefas concorrentes, quando dois ou

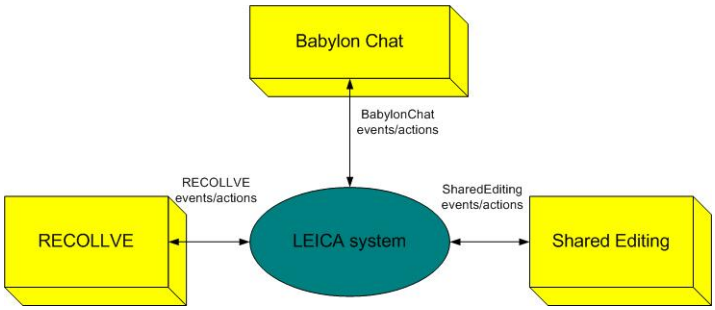

Figura 5.11: Um sistema de chat e um editor compartilhado integrados através de LEICA.

mais usuários simultâneamente editam o mesmo documento. A Figura 5.12 mostra, na primeira cena, um cenário onde três usuários, representados por seus avatares, manipulam três partes diferentes de um mesmo documento, representado por partes da lousa. Enquanto isso, outros dois usuários esperam sua vez. A segunda cena mostra que o usuário representado pelo avatar "amarelo" terminou de fazer a edição da parte dois do documento e ele espera agora a liberação da parte três. Os outros usuários podem ver que a parte dois agora está livre.

Este cenário ilustra o aumento de percepção de qual usuário está manipulando qual parte do editor de texto compartilhado. O número de partes (capítulo, página, etc.) que o editor de textor permite manipular depende da sua granularidade de bloqueio.

Quando o usuário representado pelo "avatar amarelo" termina de manipular a parte número dois do documento, o Agente Integração, segundo regra previamente definida, envia uma mensagem a LEICA solicitando a liberação do acesso para o usuário, representado pelo "avatar vermelho" que inicia a fila de espera para aquela parte do documento. A parte número dois permanecerá bloqueada para os outros usuários.

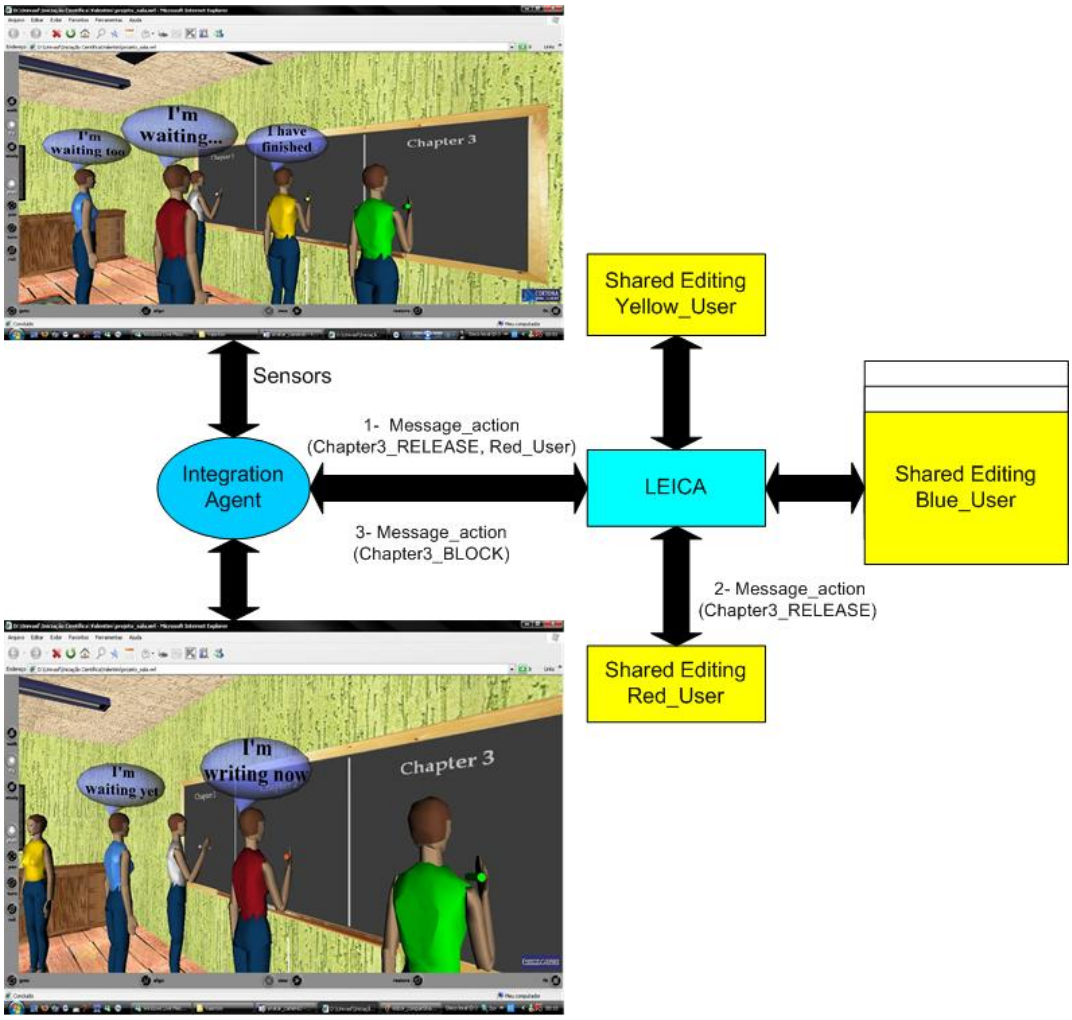

Figura 5.12: Representação na cena virtual de uma sessão de edição de texto compartilhado.

5.2.3 Terceira Experiência

Na terceira experiência, o objetivo é representar uma sessão de áudioconferência. No contexto de uma aplicação de *Ensino a Distância* (EaD), pode-se imaginar uma Universidade Virtual, onde estudantes podem compartilhar áreas de entretenimento, e onde cursos são ministrados em cada uma das salas virtuais.

Uma vez que o professor adentra em uma sala virtual, o Agente Integração solicita a LEICA [4] que inicie uma sessão de áudioconferência. Para simbolizar uma sessão de áudioconferência, o Agente Gerente, a pedido do Agente Integração, insere um

"Microfone 3D" na cena virtual.

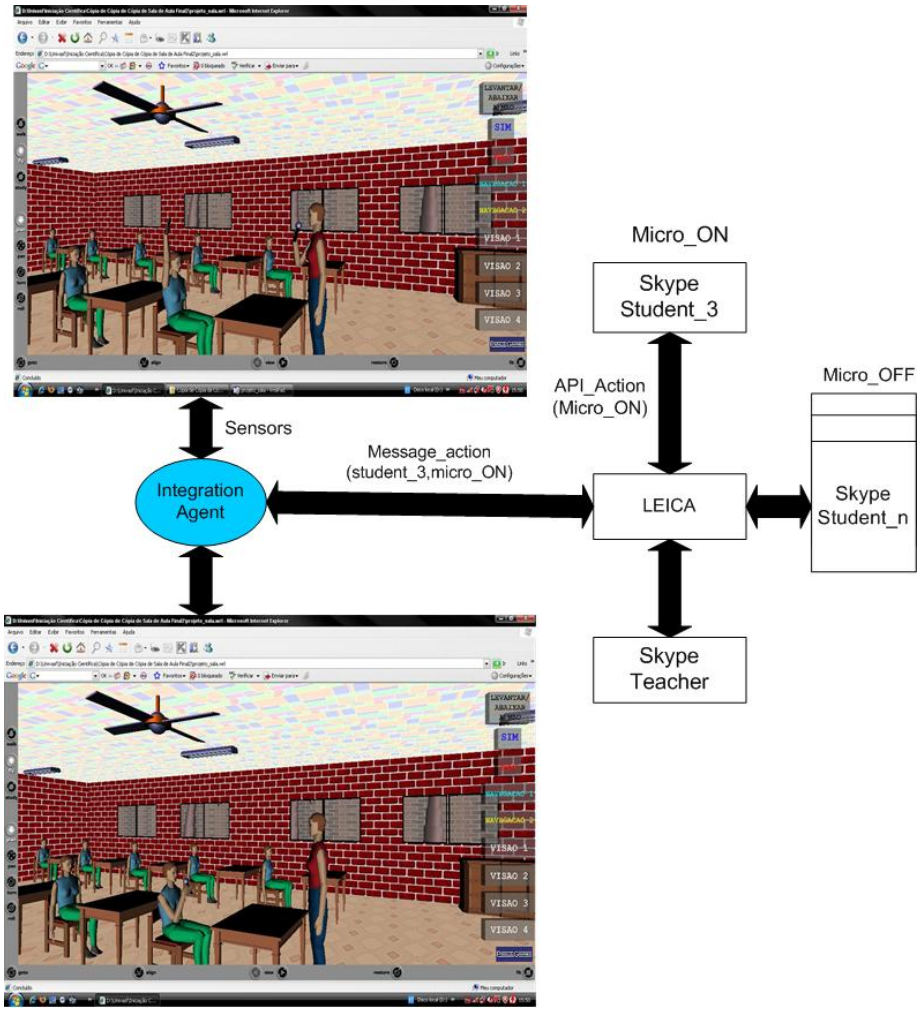

Figura 5.13: Representando na cena virtual uma sessão de áudioconferência.

O contexto refere-se a uma sala de aula virtual, ilustrada na Figura 5.13, onde os usuários desempenham os seguintes papéis: professor e estudante. Professor e estudantes são representados por seus respectivos avatares. Na primeira cena, o professor detém o "Microfone 3D" simbolizando que ele possui o Token para falar naquele momento. Nesta situação, somente o microfone da aplicação do professor está habilitado. Os microfones dos aplicativos dos alunos estão desabilitados (função MUTE). Dessa forma, assegura-se que nenhum estudante interromperá o professor enquanto este estiver falando. Quando um estudante quer falar, ele deve fazer com

que seu avatar erga o braço, indicando o seu desejo de falar. O professor então transfere o "Microfone 3D" para o avatar daquele estudante, transferindo-lhe o direito de falar. O Agente Integração, de acordo com regra previamente definida, envia uma mensagem a LEICA solicitando que o microfone do aplicativo deste estudante seja habilitado.

Este exemplo ilustra o protocolo social controlando e gerenciando o trabalho colaborativo. O professor pode pegar o "Microfone 3D" a qualquer momento. Os estudantes precisam pedir autorização para falar.

Outro exemplo, ainda na sala de aula virtual; se o professor resolver dividir a sala em dois ou mais grupos para fazer um trabalho específico; neste caso, quando os avatares dos usuários se posicionarem na "mesa 3D", o Agente Integração envia uma mensagem à LEICA para criar, no sistema Skype, dois ou mais grupos de áudioconferência. Esta situação é ilustrada na Figura 5.14. O Agente Integração torna RECOLLVE flexível o suficiente para explorar as funcionalidades das aplicações externas.

Figura 5.14: Representando dois grupos de áudioconferência na cena virtual.

No que diz respeito aos problemas relacionados à comunicação de áudio, estes são tratados pelo sistema Skype. RECOLLVE é responsável apenas pela sincronização

da cena: as posições do avatar, por exemplo.

5.2.4 Comparação de RECOLLVE com os Trabalhos Relacionados

Na Tabela 5.1, é mostrado um quadro comparativo de RECOLLVE com os demais trabalhos relacionados. RECOLLVE se destaca por, além de oferecer um alto grau de representação das atividades colaborativas, permite também o controle da aplicação externa a partir da cena virtual.

Além disso, RECOLLVE é o único que fornece um mecanismo para representar na cena virtual, os direitos de acesso dos usuários definidos na configuração de uma sessão colaborativa real. RECOLLVE também é o único dos AVCs encontrados na literatura que oferece uma integração aberta, potencializando a sua integração com qualquer tipo de aplicação colaborativa.

Tabela 5.1: Quadro comparativo entre os ambientes virtuais estudados.

	Integração	Representação	Direitos de Acesso	Gerenciamento	Baseado em Agentes	Papel
INVITE	*Fechada*	*Alta*	*Nao*	*Nao*	*Nao*	*Sim*
DIVE	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
VREng	*Fechada*	*Baixa*	*Nao*	*Nao*	*Nao*	*Nao*
Netice	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
EVE	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
Programa de Ottawa	*Fechada*	*Alta*	*Nao*	*Nao*	*Nao*	*Sim*
There	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
Second Life	*Fechada*	*Media*	*Nao*	*Nao*	*Nao*	*Nao*
Soares et al	*Fechada*	*Alta*	*Nao*	*Nao*	*Nao*	*Nao*
STEVE	*Fechada*	*Media*	*Nao*	*Nao*	*Sim*	*Sim*
VITAL	*Fechada*	*Media*	*Nao*	*Nao*	*Sim*	*Sim*
Rizzo et al.	*Fechada*	*Baixa*	*Nao*	*Nao*	*Sim*	*Sim*
Active Worlds	*Fechada*	*Alta*	*Nao*	*Nao*	*Sim*	*Nao*
RECOLLVE	*Aberta*	*Alta*	*Sim*	*Sim*	*Sim*	*Sim*

5.3 Comentários Finais

Este Capítulo apresentou detalhes da implementação de RECOLLVE como sua arquitetura Cliente-Servidor, o mecanismo utilizado que tornou possível a implementação de um mundo VRML multiusuários, assim como alguns estudos de caso que mostram o potencial de RECOLLVE como plataforma para suporte ao trabalho colaborativo.

Capítulo 6

Conclusões e Perspectivas para Trabalhos Futuros

Na presente Tese foi apresentado RECOLLVE, um sistema de realidade virtual desktop baseado em agentes, que provê mecanismos aos usuários para a representação de atividades colaborativas.

A arquitetura baseada em agentes foi validada por meio da plataforma JADE e um primeiro protótipo do sistema foi desenvolvido. Uma API também foi definida e implementada em RECOLLVE, permitindo assim que usuários colaborem além da cena virtual, o que foi comprovado por experiências de integração relatadas ao longo do texto.

6.1 Objetivos

6.1.1 Objetivo Geral

O objetivo da presente Tese é propor um ambiente virtual colaborativo que desempenhe o papel de interface virtual para a realização de trabalhos colaborativos, ora representando, ora gerenciando, no interior da cena virtual, as atividades de colaboração do mundo real.

6.1.2 Objetivos Específicos

Como objetivos específicos nesta Tese pretendeu-se:

- prover um mapeamento entre as ações dos usuários nas aplicações colaborativas e ações dos avatares na cena virtual;
- prover um mecanismo para representação dos direitos de acesso dos usuários no interior da cena virtual;
- prover um mecanismo que permita o controle da aplicação colaborativa a partir da cena virtual.

6.2 Trabalhos Futuros

Como perspectivas para trabalhos futuros pretende-se:

- desenvolver um conjunto de ferramentas de autoria para customização dos ambientes e configuração de uma sessão virtual colaborativa;
- propor e desenvolver uma arquitetura escalável que suporte uma grande quantidade de usuários;

- propor metodologias para avaliação de usabilidade desta interface virtual;
- desenvolver uma ontologia para o domínio de atividades colaborativas em ambientes virtuais.

Referências Bibliográficas

[1] GOMES, R. L., *LEICA: Un Environnement faiblement couplé pour l'intégration d'applications collaboratives*, Ph.D. Thesis, Université Paul Sabatier - Toulouse - France, 2006.

[2] BAUDIN, V., "Supporting distributed experts in e-meetings for synchronous collaboration". In: *Proc. IEEE International Conference on Systems, Man and Cybernetics, SMC'02, Tunisie*, 2002.

[3] HUMMES, J., MERIALDO, B., "Design of extensible component-based groupware", *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, v. 9, n. 1, pp. 53–74, 2000.

[4] GOMES, R. L., DE JESÚS HOYOS RIVERA, G., COURTIAT, J.-P., "LEICA: Loosely-Coupled Integration of CSCW Systems". In: *5th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 2005) LNCS Springer-Verlag, Athens, Greece*, June 2005.

[5] "Support for Distributed CSCW Applications", *Advances in Distributed Systems*, v. 1752/2000, pp. 295–326, 2000.

[6] ELLIS, C. A., GIBBS, S. J., REIN, G., "Groupware: some issues and experiences", *Communications of the ACM*, v. 34, n. 1, pp. 39–58, 1991.

[7] GRAHAM, T. C. N., GRUNDY, J., "External Requirements of Groupware Development Tools", *Engineering for Human-Computer Interaction*, pp. 363–376, 1999.

[8] LONCHAMP, J., *Le travail coopératif et ses technologies*. Hermès Lavoisier, 2003.

[9] DOURISH, P., BELLOTTI, V., "Awareness and coordination in shared workspaces", *ACM Conference on Computer Supported Cooperative Work (CSCW'92)*, pp. 107–114, 1992.

[10] GUTWIN, C., GREENBERG, S., "A Descriptive Framework of Workspace Awareness for Real-Time Groupware", *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, v. 11, n. 3-4, pp. 411–446, 2002.

[11] DIX, A., "Challenges for Cooperative Work on the Web: An analytical approach", *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, v. 6, n. 2-3, pp. 135–156, 1997.

[12] HAAKE, J. M., WIIL, U. K., NURNBERG, P. J., "Openness in shared hypermedia workspaces: The case for collaborative open hypermedia systems", *ACM SIGWEB Newsletter*, v. 8, n. 3, pp. 33–45, 1999.

[13] PINELLE, D., GUTWIN, C., GREENBERG, S., "Task Analysis for groupware usability evaluation: Modeling shared-workspaces tasks with the mechanisms of collaboration", *ACM Transactions on Computer-Human Interaction (TOCHI)*, v. 10, n. 4, pp. 281–311, 2003.

[14] DOMMEL, H. P., GARCIA-LUNA-ACEVES, J. J., "Efficacy of floor control protocols in distributed multimedia collaboration", *Cluster Computing Journal*, v. 2, n. 1, pp. 17–33, 1999.

[15] MUNSON, J., DEWAN, P., "A concurrency control framework for collaborative systems", *ACM conference on Computer supported cooperative work (CSCW'96)*, pp. 278–287, 1996.

[16] YANG, Y., LI, D., "Separating data and control: suport for adaptable consistency protocols in collaborative systems", *ACM conference on Computer supported cooperative work (CSCW'04)*, pp. 11–20, 2004.

[17] BERNARD, S., *Spécification d'un environnement d'ingénierie collaborative multisite - Application à l'industrie aéronautique européenne*, Ph.D. Thesis, École nationale supérieure d'arts et métiers - Aix en Provence - France, 2004.

[18] LAURILLAU, Y., *Conception et réalisation logicielles pour les collecticiels centrées sur l'activité de groupe: le modèle et la plate-forme Clover*, Ph.D. Thesis, Université Joseph Fourier - Grenoble I - France, 2002.

[19] MORA, M., FORGIONNE, G. A., GUPTA, J. N. D., "Decision making support systems: achievements, trends, and challenges for the new decade", *ACM conference on Computer supported cooperative work (CSCW'04)*, pp. 418, 2002.

[20] LORCY, S., *Infrastructure logicielle pour la gestion de la cohérence et de la qualité de service d'un environnement à objets réparti: application au*

telétravail coopératif, Ph.D. Thesis, Université de Rennes 1 - Rennes - France, 2000.

[21] KILGORE, R., CHIGNELL, M., SMITH, P., "Spatialized audioconferencing: what are the benefits ?" *Conference of the Centre for Advanced Studies on Collaborative research IBM Centre for Advanced Studies Conference*, pp. 135–144, 2003.

[22] NORMAND, V., BABSKI, C., BENFORD, S., et al., "COVEN Project: exploring applicative, technical, and usage dimensions of collaborative virtual environments". pp. 218–236, MIT Press, 1999.

[23] KILGORE, R., CHIGNELL, M., SMITH, P., "Spatialized audioconferencing: what are the benefits ?" *Conference of the Centre for Advanced Studies on Collaborative research IBM Centre for Advanced Studies Conference*, pp. 135–144, 2003.

[24] SINGHAL, S., ZYDA, M., "Networked Virtual Environments: design and implementation". ACM Press, 1999.

[25] TORGUET, P., *VIPER: Un modèle de calcul réparti pour la gestion d'environnements virtuels*, Ph.D. Thesis, l'UPS - l'Université Paul Sabatier - Toulouse - França, 1998.

[26] AQUINO, M. S., DE SOUZA, F. F., FRERY, A. C., "VEPersonal: an infrastructure of Virtual Reality components to generate web adaptive environments". In: *WebMedia '05: Proceedings of the 11th Brazilian Symposium on Multimedia and the web*, pp. 1–8, ACM Press: New York, NY, USA, 2005.

[27] ANASTASSAKIS, G., RITCHINGS, T., PANAYIOTOPOULOS, T., "Multi-agent Systems as Intelligent Virtual Environments", *LNAI*, , n. 2174, pp. 381–395, 2001.

[28] FRECON, E., STENIUS, M., "DIVE: A Scalable Network Architecture for Distributed Virtual Environments". In: *Distributed Systems Engineering Journal (special issue on Distributed Virtual Environments)*, v. 5, pp. 91–100, 1998.

[29] "http://www.ripe.net/ripe/wg/mbone/home.html". 2000.

[30] BOURAS, C., TSIATSOS, T., "Distributed virtual reality: bulding a multi-user layer for the EVE Platform". In: *Journal of Network and Computer Applications*, v. 27, pp. 91–111, Elsevier, 2004.

[31] KULJIS, J., LEES, D. Y., "Lessons from Industry in the Design of Virtual Collaborative Learning Environments". In: *24th Int. Conf. Information Technology Interfaces ITI 2002, Cavtat, Croácia*, 2002.

[32] THALLMAN, D., "The Role of Virtual Humans in Virtual Environment Technology and Interfaces". In: *Proceedings of Joint EC-NSF Advanced Research Workshop, Bonas, França*, 1999.

[33] "Programa de Ottawa", `http://www.mcrlab.uottawa.ca/`, 2009, [último acesso: 20/02/2009].

[34] "ADVICE: Um Ambiente VIrtual Colaborativo para o Ensino a Distância", June 2004.

[35] “”, http://amp.ece.cmu.edu/projects/NetICE/, 2009, [último acesso: 11/03/2009].

[36] HORAIN, P., SOARES, J. M., RAI, P. K., et al., “Virtually enhancing the perception of user actions”. In: *Proceedings of 15th International Conference on Artificial Reality and Telexistance (ICAT 2005)*, pp. 245–246, 2005.

[37] DAX, P., “Virtual Reality Engine - http://vreng.enst.fr/html/index.html”, .

[38] “There”, http://www.there.com, 2009, [último acesso: 10/01/2009].

[39] “Second Life: Your World, Your Imagination”, http://secondlife.com, 2009, [último acesso: 10/01/2009].

[40] JONHSON, W. L., “Pedagogical Agent Research at CARTE”, *AI Magazine*, v. 22, n. 4, pp. 85–94, 2001.

[41] “Active Worlds”, http://www.activeworlds.com/, 2009, [último acesso: 11/03/2009].

[42] RIZZO, A. A., BOWERLY, T., BUCKWALTER, J. G., et al., “Virtual Environments for the Assessment of Attention and Memory Processes: The Virtual Classroom and Office”. In: *Proceedings of the International Conference on Disability, Virtual Reality and Associated Technology 2002 (ICDVRAT2000)*, 2002.

[43] RIZZO, A. A., BOWERLY, T., BUCKWALKER, J. G., et al., “A Virtual Reality Scenario for All Seasons: The Virtual Classroom”, *CNS Spectrum*, v. 11, n. 1, pp. 35–44, 2006.

[44] RUSSEL, S., NORVIG, P., *Inteligência Artificial*. Editora Campus-Elsevier, 2004.

[45] "Foundation for Intelligent Physical Agents - FIPA", `http://www.fipa.org`, 2009, [último acesso: 20/01/2009].

[46] "FIPA Agent Communication Language Specifications", `http://www.fipa.org/repository/aclspecs.html`, 2009, [último acesso: 20/01/2009].

[47] "Java Agent DEvelopment Framework", `http://jade.tilab.com/`, 2009, [último acesso: 20/01/2009].

[48] "http://www.ecma-international.org/default.htm".

[49] HARTMAN, J., WERNECKE, J., *The VRML 2.0 Handbook: Bulding Moving Worlds on the Web*. Addison-Wesley, 1996.

[50] "Web3D Comsortium", `http://www.web3d.org/`, 2009, [último acesso: 15/03/2009].

[51] "VRML (ISO/IEC 14772-1), Virtual Reality Modeling Language", 1997.

[52] "Web3D Consortium - Living Worlds Working Groups". 1997.

[53] "VRML EAI (ISO/IEC FDIS 14772-2), The VRML External Authoring Interface Committee Final Draft", 2002.

[54] PICARD, S. L. D., *Plata-forme de communication distribuee pour les Environnements Virtuels Collaboratifs 3D a fort couplage d'activite synchrone*, Ph.D. Thesis, Nov. 2003.

[55] HONDA, Y., MITRA, A., "Core Living Worlds". 1998.

[56] LEA, R., HONDA, Y., MATSUDA, K. M. S., "Community Place: Architecture and Performance". In: *Proceedings of the VRML'97*, 1997.

[57] "http://http://www.geometrek.com/products/deepmatrix.html". 2000.

[58] CAPPS, M., MCGREGOR, D., BRUTZMAN, D., et al., "NPSNET-V: A New Beginning for Dynamically Extensible Virtual Environments". In: *IEEE Computer Graphics and Applications*, v. 20, pp. 12–15, 2000.

[59] HAWKES, R., WRAY, M., "LivingSpace: A Living Worlds Implementation using an Event-based Architecture". In: *Hewlett-Packard Technical Report*, 1999.

[60] ARAKI, Y., "VSPLUS: A High-level Multi-user Extension Library For Interactive VRML Worlds". In: *Proceedings of the VRML'98*, 1998.

[61] CARSON, J. A., CLARK, A. F., "Multicast Shared Virtual Worlds Using VRML97". In: *Proceedings de VRML'99*, Feb. 1999.

[62] STEPHANE LOUIS DIT PICARD, SAMUEL DEGRANDE, C. G., CHAILLOU, C., "VRMLData Sharing in the Spin-3D CVE". In: *WEB3D 2002 SYMPOSIUM : Proceedings 7th International Conference on 3D WEB SIGGRAPH, Tempe, Arizona, USA*, pp. 165–172, ACM Press, June 2002.

[63] "X3D SAI (ISO/IEC 19775-2:2004), The X3D Scene Access Interface", 2004.

[64] "H-Anim: Humanoid Animation Working Group", `http://www.h-anim.org/`, 2009, [último acesso: 11/03/2009].

[65] "Xj3D Project", `http://www.xj3d.org/`, 2009, [último acesso: 15/03/2009].

[66] LIMA, C. V., WILLRICH, R., DE J. H. RIVERA, G., et al., "Sistema de Co-Navegação com suporte a Áudioconferência". In: *Proceedings of IV Simpósio Brasileiro de Sistemas Colaborativos*, 2007.

[67] "X3D (ISO/IEC FDIS 19775:200x), X3D framework & SAI - Final Draft International Standard", 2004.

[68] DE JESÚS HOYOS RIVERA, G., *COLAB: Conception et mise en oeuvre d'un outil pour la navigation coopérative sur le WEB*, Ph.D. Thesis, Université Paul Sabatier - Toulouse - France, 2005.

Apêndice A

Diagramas de sequência do módulo de comunicação

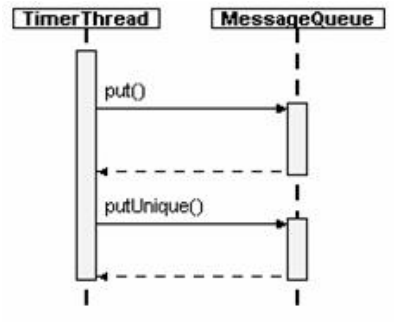

Figura A.1: Diagrama de escrita na fila de espera.

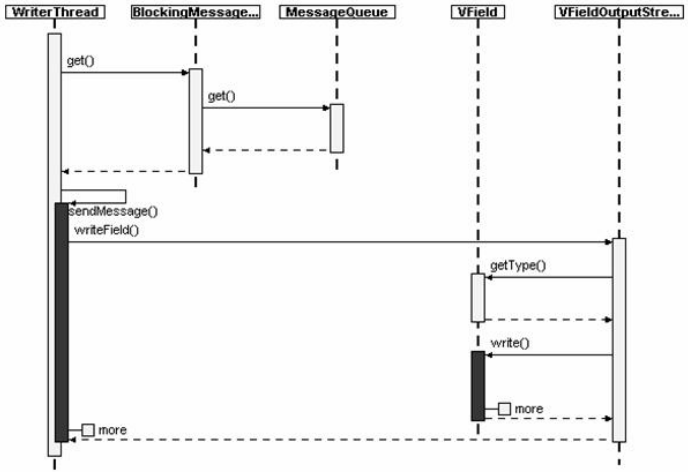

Figura A.2: Diagrama de envio de mensagens pelo Cliente ou Servidor.

Apêndice B

Diagramas de sequência do código do Servidor

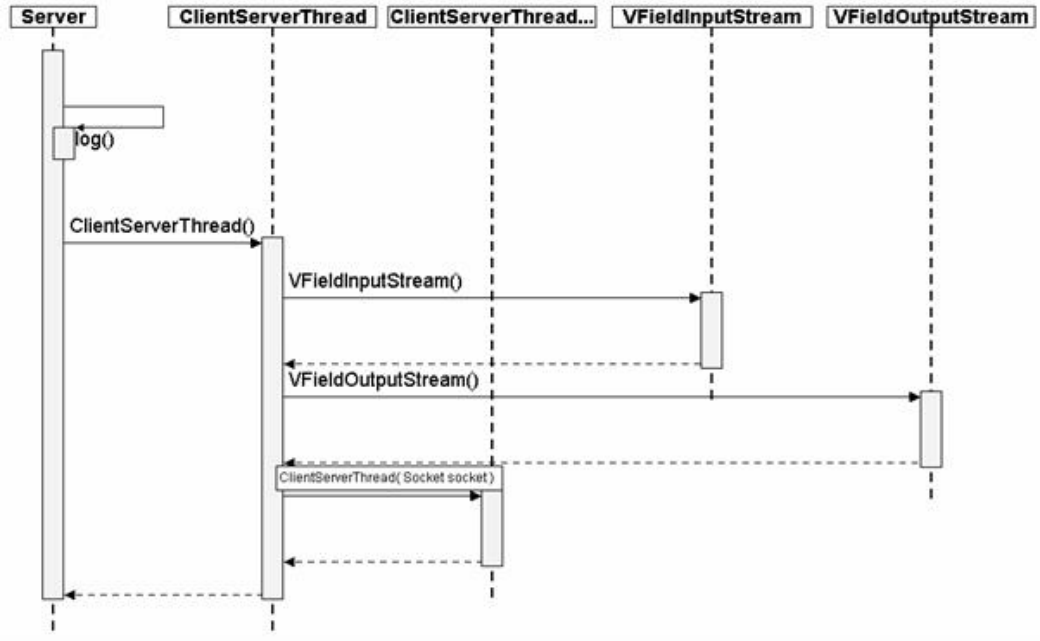

Figura B.1: Recepção do Cliente pelo Servidor.

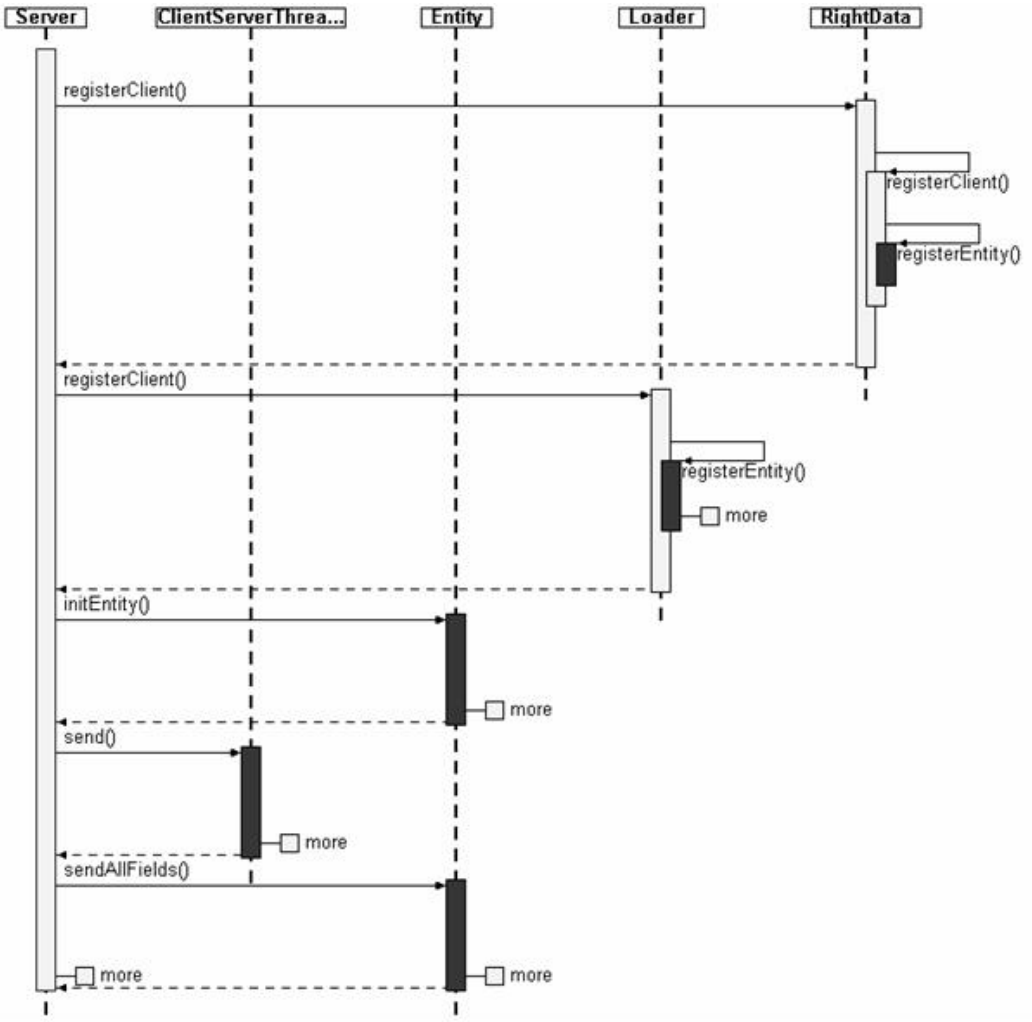

Figura B.2: Registro do Cliente no Sistema.

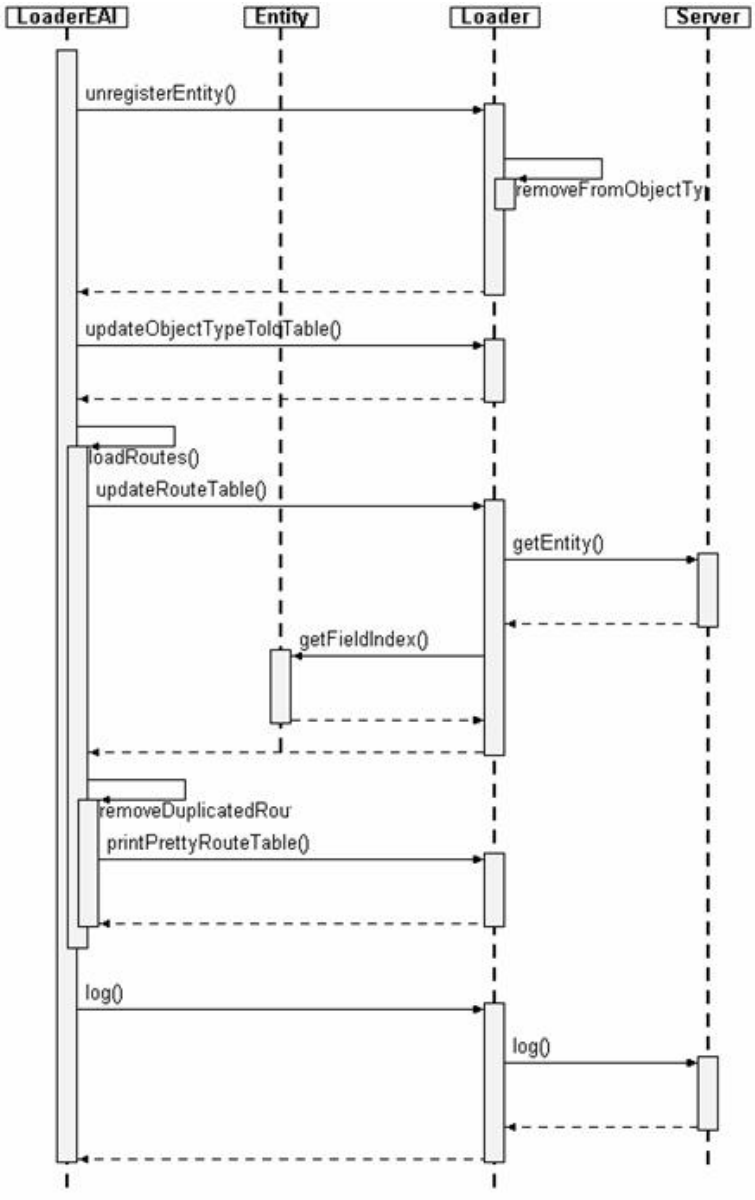

Figura B.3: Criação e configuração de uma nova entidade.

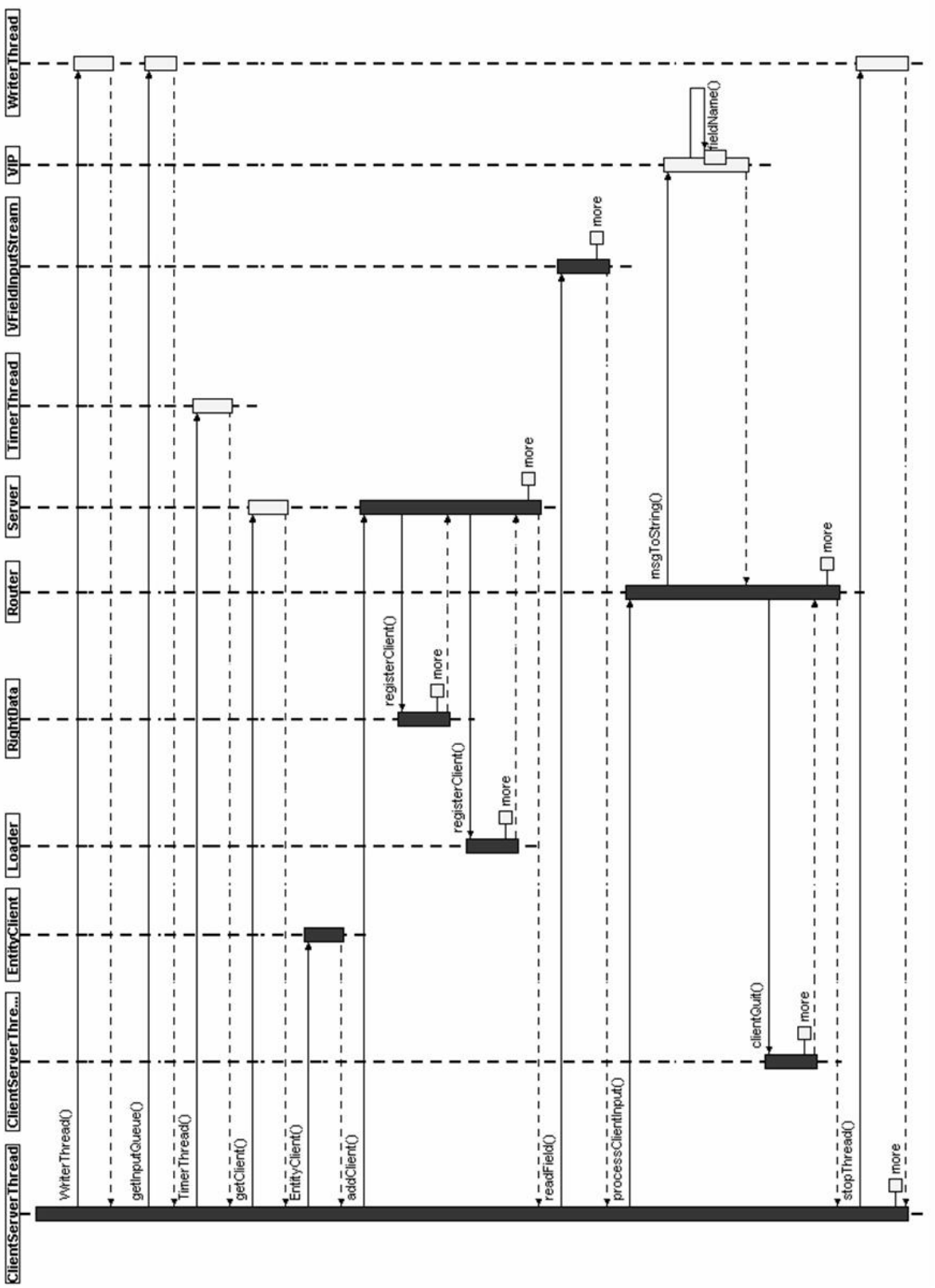

Figura B.4: Rotina do Thread Cliente-Servidor.

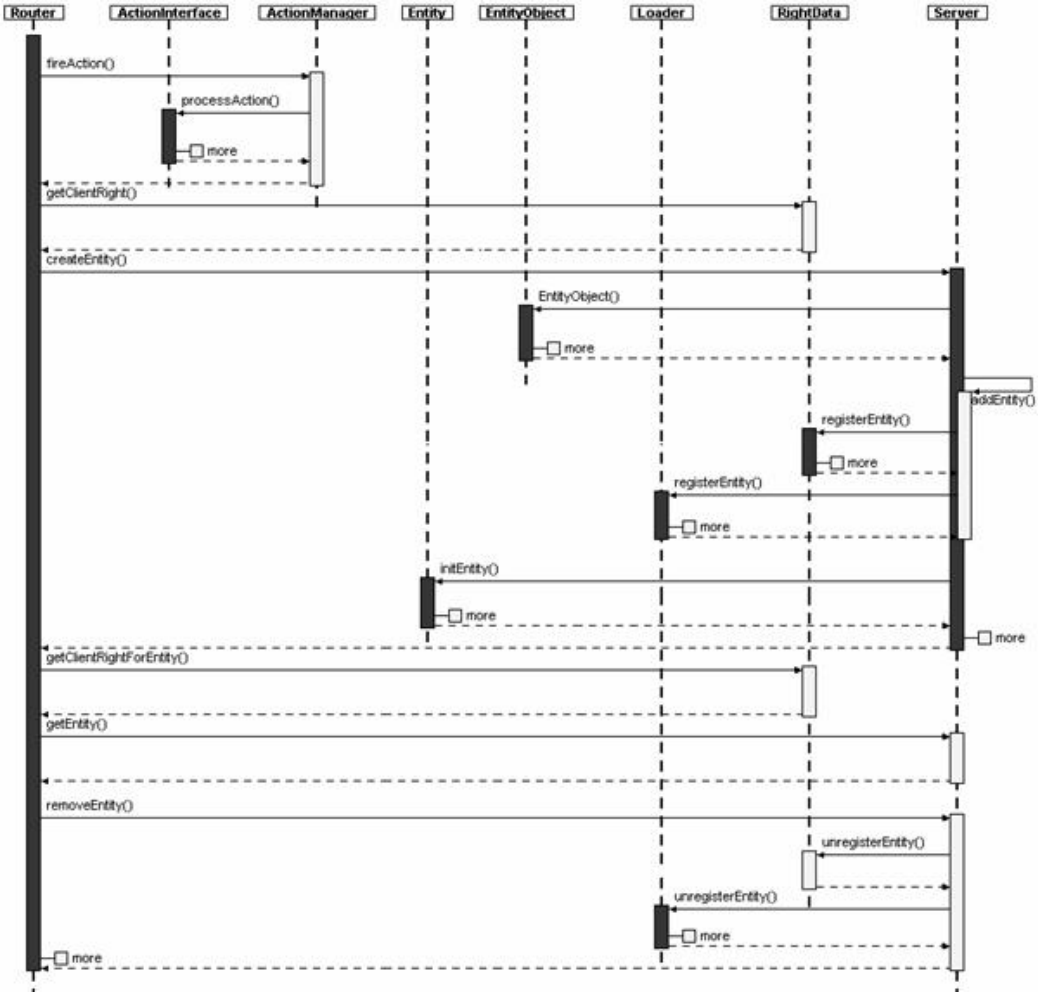

Figura B.5: Roteamento dos eventos.

Apêndice C

Diagramas de sequência do código do Cliente

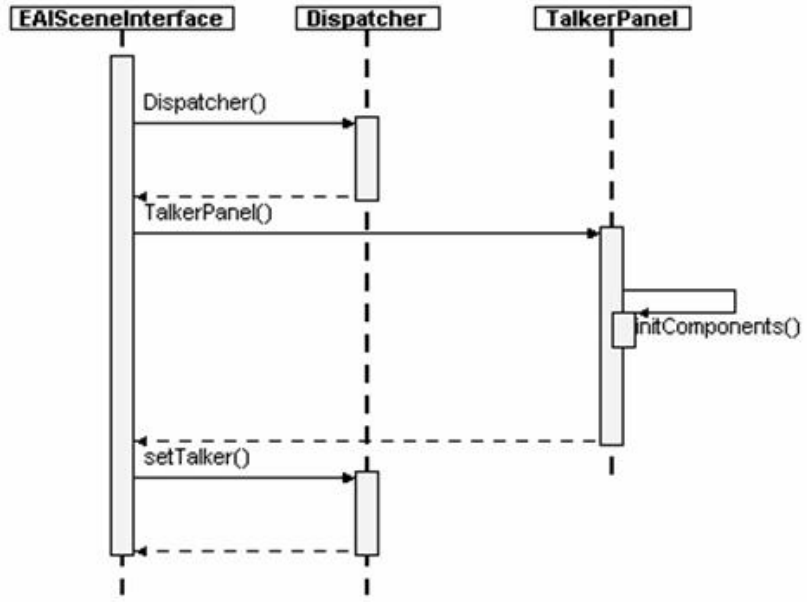

Figura C.1: Inicialização do Cliente.

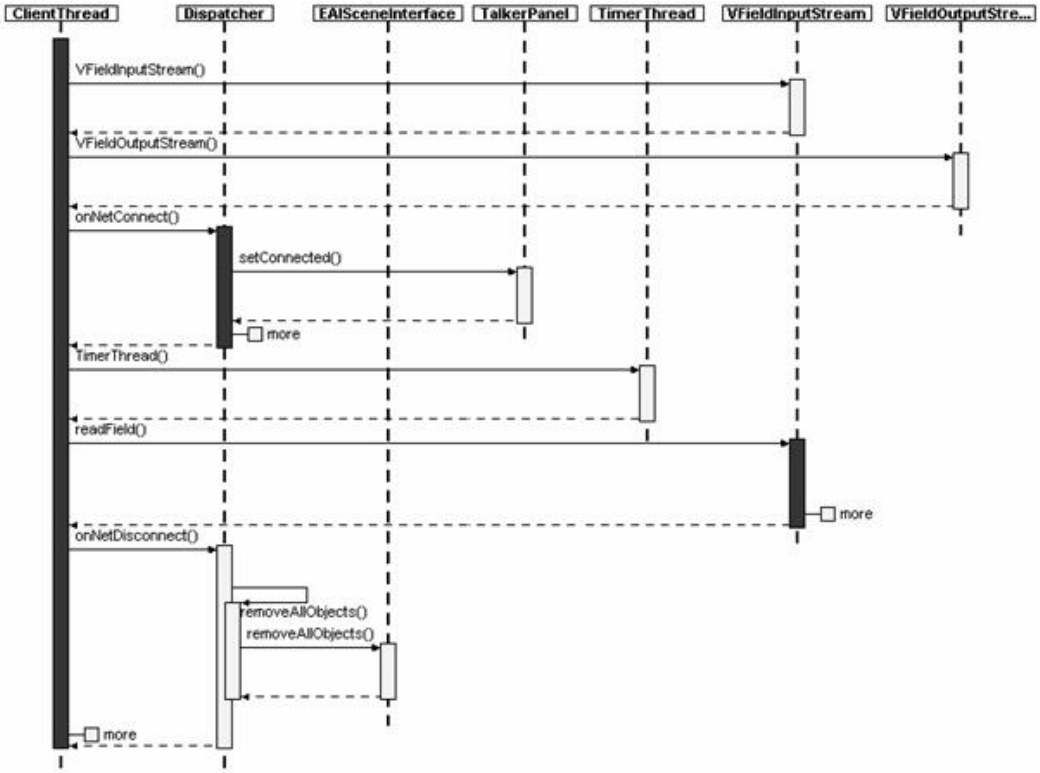

Figura C.2: Rotina do Thread Cliente.

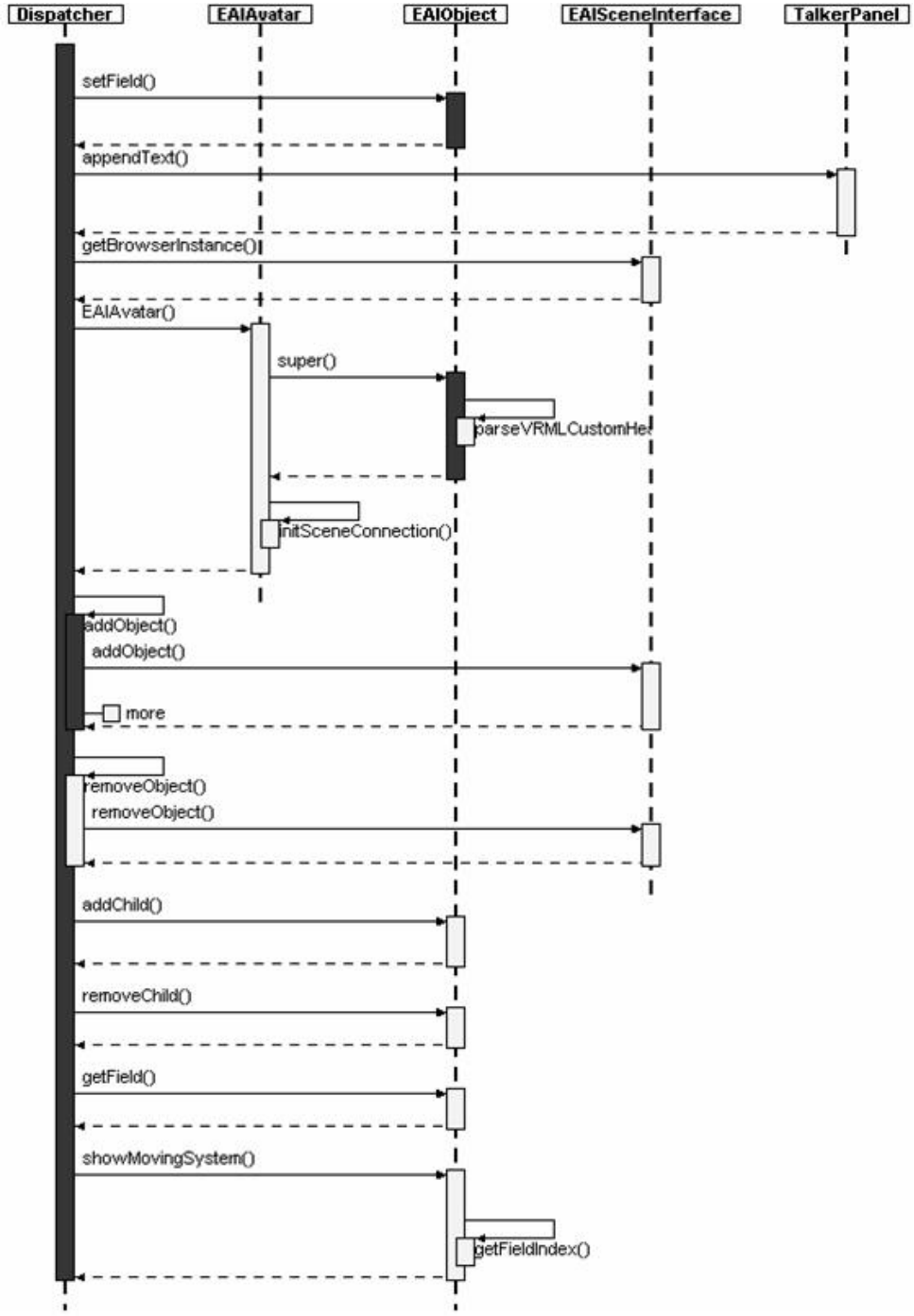

Figura C.3: Rotina do *Dispatcher*.