FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”
CENTRO UNIVERSITÁRIO “EURÍPIDES DE MARÍLIA” - UNIVEM
PROGRAMA DE MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

CESAR AUGUSTO CUSIN

UM SISTEMA MULTIMODAL COM MOVIMENTOS INTERPRETADOS EM LINGUAGEM NATURAL CONTROLADA

MARÍLIA
2005

CESAR AUGUSTO CUSIN

UM SISTEMA MULTIMODAL COM MOVIMENTOS INTERPRETADOS EM LINGUAGEM NATURAL CONTROLADA

Dissertação apresentada ao Programa de Mestrado do Centro Universitário Eurípides de Marília, mantido pela Fundação de Ensino Eurípides Soares da Rocha, para obtenção do Título de Mestre em Ciência da Computação. (Área de Concentração: Realidade Virtual)

Orientador:
Prof. Dr. José Remo Ferreira Brega

MARÍLIA
2005

CUSIN, Cesar Augusto

Um Sistema Multimodal com Movimentos Interpretados em Linguagem Natural Controlada / Cesar Augusto Cusin; orientador: José Remo Ferreira Brega. Marília, SP: [s.n.], 2005.

98 f.

Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília - Fundação de Ensino Eurípides Soares da Rocha.

1. Multimodal 2. Realidade Virtual 3. Interação Humano-Computador 4. Linguagem Natural Controlada

CDD: 006

CESAR AUGUSTO CUSIN

UM SISTEMA MULTIMODAL COM MOVIMENTOS INTERPRETADOS EM LINGUAGEM NATURAL CONTROLADA

Banca examinadora da dissertação apresentada ao Programa de Mestrado da UNIVEM,/F.E.E.S.R., para obtenção do Título de Mestre em Ciência da Computação.

Resultado: _____

ORIENTADOR: Prof. Dr. José Remo Ferreira Brega

1º EXAMINADOR: _____

2º EXAMINADOR: _____

Marília, _____ de _____ de 2005.

Dedico este trabalho

aos meus pais que tanto

me incentivaram e deram

apoio para a conclusão do mesmo.

AGRADECIMENTOS

A realização deste trabalho não poderia ter acontecido de forma satisfatória se não fosse o empenho e suporte irrestrito prestado pelo professor Dr. José Remo Ferreira Brega, que sempre se mostrou atencioso no decorrer deste trabalho, apontando sempre uma direção a seguir. A ele, meu muito obrigado.

Agradeço a Deus por me dar saúde e inteligência suficiente para concluir o presente trabalho; por me proteger em minhas viagens e cuidar de minha família quando estive ausente.

Agradeço aos professores que de forma simples e direta além de transmitir conhecimentos dos livros, transmitiram a essência de ser um pesquisador.

Agradeço à minha família, minha esposa Juliana, meus filhos Bruno e Guilherme, por compreender esse momento especial da minha vida, por entender a minha ausência, por dar o apoio necessário nas horas difíceis.

Agradeço ao amigo Fábio Alexandre Caravieri Modesto (Fabião) que esteve presente nos momentos bons e difíceis durante todo o Mestrado.

Agradeço aos demais colegas do Mestrado pela amizade durante esse tempo e pelo apoio nas horas necessárias. Agradeço aos colegas na pessoa dos amigos Ricardo Veronesi e André Gobbi.

A todos vocês que tanto me ensinaram, ajudaram e apoiaram em cada momento; o meu sincero agradecimento.

Agradeço aos membros da banca por disporem de seu precioso tempo na leitura deste trabalho.

CUSIN, Cesar Augusto. **Um Sistema Multimodal com Movimentos Interpretados em Linguagem Natural Controlada**. 2005. 98 f. Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2005.

RESUMO

Esta dissertação é o resultado do desenvolvimento de uma ferramenta Multimodal para Realidade Virtual que trata da Interação Humano-Computador. A Realidade Virtual é a forma mais avançada de interface do Usuário com o Computador por proporcionar a imersão, interação e envolvimento. Em se tratando de uma ferramenta para Realidade Virtual; para a entrada de dados; foi escolhida a forma Multimodal. Sabendo das dificuldades impostas pelos dispositivos de entrada (convencionais ou não); devido a estes enviarem sinais diferentes ao computador; o sistema proposto interpreta a entrada de dados, converte para uma linguagem própria (Linguagem Natural Controlada), e envia os comandos que irão alterar o Mundo Virtual. Por isso a necessidade do desenvolvimento de um sistema que trate dessa interação de uma forma Multimodal. Ao final é apresentado o sistema que justifica a abordagem proposta.

Palavras-chave: Multimodal. Realidade Virtual. Interação Humano-Computador. Linguagem Natural Controlada.

CUSIN, Cesar Augusto. **Um Sistema Multimodal com Movimentos Interpretados em Linguagem Natural Controlada**. 2005. 98 f. Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2005.

ABSTRACT

This dissertation is the result of the development of a Multimodal tool for Virtual Reality that deals with the Human-Computer Interaction. The Virtual Reality is the most advanced form of interface between the user and the computer by providing the immersion, interaction and involvement. Considering a tool for Virtual Reality; for the data entry; the Multimodal form was chosen. Knowing the difficulties imposed by the entrance devices (conventional or not); due to the sending of different signals to the computer by them; the proposed applicative interprets the data entry, converts to a proper language (Controlled Natural Language), and sends the commands that will modify the Virtual World. Therefore the necessity to develop a tool which deals with this interaction of a Multimodal form. In the end a prototype is presented which justifies the proposed approach.

Keywords: Multimodal. Virtual Reality. Human-Computer Interaction. Controlled Natural Language.

CUSIN, Cesar Augusto. **Um Sistema Multimodal com Movimentos Interpretados em Linguagem Natural Controlada**. 2005. 98 f. Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2005.

RESUMEN

Esta disetación es el resultado del desarrollo de una herramienta Multimodal para Realidad Virtual que versa sobre la Interacción Humano-Computador. La Realidad Virtual es la forma más avanzada de interface del Usuario con el Computador por proporcionar la inmersión, interacción y envolvimiento. Por se tratar de una herramienta para Realidad Virtual, para la entrada de datos, se ha elegido la forma Multimodal. Teniendo conocimiento de las dificultades impuestas por los dispositivos de entrada (convencionales o no) en razón de que estes envian señales diferentes al computador, el aplicativo propuesto interpreta la entrada de datos, convierte a una lenguage propia (Lenguage Natural Controlada) y envia los comandos que irán cambiar el Mundo Virtual. Por eso la necesidad del desarrollo de una herramienta que trata de esa interacción de una forma Multimodal. Al final se presenta un prototipo que justifica la versión propuesta.

Palabras clave: Multimodal. Realidad Virtual. Interacción. Humano-Computador. Lenguage Natural Controlada.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

2D: Duas Dimensões

3D: Tridimensional

API: *Application Programing Interface*

AV: Ambiente Virtual

AVI: Ambiente Virtual Interativo

CML: Chemical *Markup Language*

CSS: *Cascading Style Sheets*

DoF: *Degree-of-Freedom*

DOM: *Document Object Model*

EBNF: *Extended Backus-Naur Form*

GB: *GigaBytes*

GHZ: GigaHertz

GUI: *Graphical User Interface*

HD: *Hard Disk*

HMD: *Head Mounted Display*

HTML: *HyperText Markup Language*

IMSM: Interpretador de Movimentos para um Sistema Multimodal

JAXP: *Java API for XML Processing*

LN: Linguagem Natural

LNC: Linguagem Natural Controlada

LRC: Linguagem de Representação de Conhecimento

MB: *MegaBytes*

PC: *Personal Computer*

PDA: *Personal Digital Assistant*

RAM: *Random Access Memory*

RV: Realidade Virtual

RVI: Realidade Virtual Imersiva

SAX: *Simple API for XML*

SGML: *Standart Generalized Markup Language*

SMIL: *Synchronous Multimedia Integration Language*

SO: Sistema Operacional

TCP/IP: *Transmission Control Protocol*/*Internet Protocol*

USB: *Universal Serial Bus*

VRML: *Virtual Reality Modeling Language*

W3C: *World Wide Web Consortium*

XBRL: *Extensible Business Reporting Language*

XHTML: *Extensible HyperText Markup Language*

XML: *Extensible Markup Language*

XSL: *Extensible StyleSheet Language*

SUMÁRIO

INTRODUÇÃO

Para Hancock (1995), "...A Realidade Virtual (RV) pode ser definida, de uma maneira simplificada, como sendo a forma mais avançada de interface do usuário com o computador até agora disponível...", "...e que faz a junção de três idéias básicas: imersão, interação e envolvimento em um ambiente virtual sintético...", segundo Brega *et al.* (2003) (*apud* Pinho e Kirner, 1997).

Para obter-se a junção dessas três idéias básicas, é interessante o uso de dispositivos de entrada, sendo eles convencionais (Teclado, *Mouse*) ou não (Luvas, *Mouse* 3D).

Estes dispositivos de entrada, por não serem utilizados de forma comum, enviam "sinais" ao computador e não existe um *software* padrão que se responsabilize por tratar todos eles na mesma aplicação.

O objetivo principal deste trabalho é através da entrada de dados feita pelo Usuário, independente de dispositivo, o sistema que será proposto interpreta a entrada de dados, analisa através de uma gramática própria e altera o Mundo Virtual.

No sistema, o Usuário irá se deparar com um Ambiente Virtual (AV) desenvolvido em Java 3D, no qual ele poderá fazer alterações, tais como: anexar objetos, alterar geometricamente o objeto (posição) e liberar os objetos anexados. Para isso ele poderá optar como forma de entrada de dados no computador por diferentes dispositivos (convencionais ou não), ou seja, uma entrada de dados Multimodal.

Para executar essas alterações, foi necessário "traduzir" todos esses "sinais" dos diferentes dispositivos de entrada para uma única linguagem, a Linguagem Natural Controlada (LNC) que foi desenvolvida na linguagem XML (*Extensible Markup Language*). O sistema é baseado na linguagem Java e, para o Java compreender a gramática que está em XML, foi

necessário usar o JDOM, que é uma API JAVA, permitindo assim, ao Java interagir com a XML.

Para alcançar este objetivo, o trabalho está dividido nos seguintes Capítulos:

O Capítulo 1 trata da Interação Humano-Computador, sua definição, como ela pode executar, simplificar ou apoiar as tarefas do Usuário. Trata também dos modelos de interação e suas dificuldades; os tipos de interação mais comuns, mostra os dispositivos não-convencionais e apresenta uma introdução a Realidade Virtual com exemplos de uso, suas necessidades e restrições.

No Capítulo 2, trata-se da Manipulação de Objetos. Nele aborda-se o tema da navegação interativa e exploração em Ambientes Virtuais. Discute-se sobre o melhor tipo de Interação; dos ambientes intuitivos; a dificuldade da manipulação precisa de objetos virtuais. Este Capítulo mostra as etapas de uma técnica interativa de manipulação, das aplicações Multimodais, das suas vantagens; e mostra também as Tecnologias utilizadas e os Ambientes.

A Linguagem Natural Controlada é tratada no Capítulo 3, iniciando com a definição de Sintaxe e Semântica; a explicação de Linguagem Formal; a Linguagem Regular e suas limitações; as vantagens da Linguagem Livre de Contexto; a Gramática com seu conceito; a Gramática Livre de Contexto, seus usos mais comuns e as partes de um Compilador; por fim; a Linguagem Natural Controlada; sua definição; o uso da Linguagem de Representação do Conhecimento, suas desvantagens; o uso da Linguagem Natural, suas ambigüidades, e a proposta de uma Linguagem Natural Controlada, como ela é apresentada sendo uma saída para esse problema. São apresentados também casos de sucesso com o uso da Linguagem Natural Controlada.

Já o Capítulo 4 aborda o XML; seu histórico; sua função e vantagens. As regras do DTD e dos Documentos de Esquema com suas vantagens, também são vistos nesse Capítulo. O Analisador Sintático DOM e sua estrutura de árvore são vistos com exemplo, o outro Analisador Sintático SAX e seus eventos, vantagens e desvantagens. O JDOM é apresentado co-

mo a melhor saída para a uma aplicação que envolve Java com XML, exemplos comprovam a afirmação.

A Interação é abordada no Capítulo 5, mostrando inicialmente como se faz a geração de Ambientes Virtuais por computador e sua Interação com o Usuário, trata posteriormente da Interação do Usuário com o Sistema Operacional, as definições de Sistema Operacional e como é esse tipo de Interação e suas interfaces. A Interação do Usuário com a Internet, suas vantagens e como navegar em Mundos Virtuais na Internet. O desafio de alterar o Mundo Virtual independente do dispositivo de entrada. Este Capítulo aborda também a Interação do Usuário com o Aplicativo Proposto.

O Capítulo 6 mostra o protótipo, a justificativa pela escolha das linguagens e do tipo de universo escolhido, a descrição detalhada da implementação, como é feita a captura dos movimentos dos diferentes dispositivos de entrada, programação dos botões do *joystick* e das teclas do teclado, a definição da Gramática da aplicação, a conversão dos valores para a LNC, os testes de colisão e anexação e os resultados obtidos.

Posteriormente tem-se a Conclusão, Referências e um Anexo que mostra uma proposta com o uso do VRML combinado com o JavaScript, suas limitações e discussões sobre o assunto.

1. INTERAÇÃO HUMANO-COMPUTADOR

Segundo Dix *et al.* (1998), os modelos de interação nos ajudam a entender o que está "entrando" na relação entre o usuário e o sistema. Eles enviam as traduções entre o que o usuário quer e o que o sistema faz.

Rodello *et al.* (2003) afirma que "...A idéia de interação é ligada com a capacidade do computador detectar as entradas do usuário e modificar instantaneamente o Ambiente Virtual (AV) e as ações sobre ele...".

O diálogo entre usuário e sistema é influenciado pelo estilo da interface. A interação acontece dentro de um contexto social e organizacional que afeta usuário e o sistema.

A Interação Humano-Computador está preocupada em como o usuário usa o computador como uma ferramenta para executar, simplificar ou apoiar uma tarefa. Para fazer isto o usuário tem que comunicar suas exigências à máquina.

Há vários modos nos quais o usuário pode interagir com o sistema. Em um extremo, é a contribuição na qual o usuário provê imediatamente à informação ao computador para executar a tarefa. Esta aproximação efetua uma interação entre o usuário e computador, mas não se apóia bem em muitas tarefas. No outro extremo, são usados dispositivos de contribuição altamente interativos e paradigmas, como manipulação direta e as aplicações de Realidade Virtual (RV). Neste extremo o usuário está constantemente provendo instruções que estão sendo avaliadas pelo sistema

Considera-se a comunicação entre o usuário e sistema como interação. Ao verificar alguns modelos de interação, nos é permitido identificar e avaliar componentes da interação, como também assuntos físicos, sociais e organizacionais que provêem o contexto para isto. Também se inspeciona alguns dos estilos diferentes de interação que são usados e pode-se considerar se são ou não efetivos no apoio ao usuário.

Nos modelos de interação, verifica-se que a interação envolve dois participantes pelo menos: o usuário e o sistema. Ambos são complexos e muito diferentes um do outro, do modo que se comunicam e "vêem" a tarefa. A interface tem então de funcionar efetivamente como um tradutor entre eles de modo a permitir a interação. Esta tradução pode falhar em vários pontos e por vários razões.

O uso de modelos de interação pode nos ajudar a entender o que está "entrando" na interação e identificar as dificuldades em transformar as vontades do Usuário em ações no sistema. Eles também nos proporcionam um embasamento para comparar interações diferentes e considerar problemas de interação. (DIX *et al.*, 1998)

A manipulação direta é, talvez, o modelo de interação mais influente na Interação Humano-Computador, possivelmente por causa de sua proximidade para nosso entender intuitivo da interação entre o usuário e computador pelo fato de poder apontar para os objetos tridimensionais. Já na manipulação indireta, o Usuário interage usando uma interface bidimensional conhecida como *widget*, e os valores obtidos destes objetos são usados para a manipulação dos objetos tridimensionais como na Figura 1 usando o *slider*.

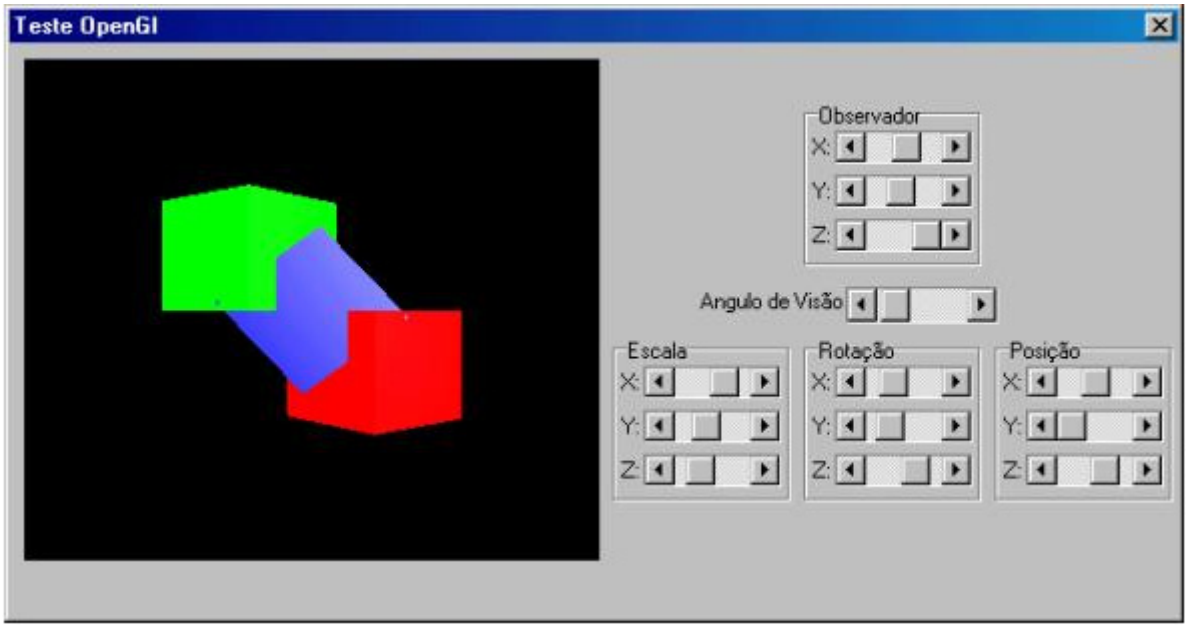


Figura 1 – Exemplo de manipulação indireta com o uso de *sliders* (PINHO, 2000).

Na manipulação direta, o usuário formula um plano de ação que é executado então à interface de computador. Quando o plano ou parte dele foi executado, o usuário observa a interface de computador para avaliar o resultado do plano executado, e determinar ações adicionais.

O ciclo interativo pode ser dividido em duas fases principais: execução e avaliação. Estas duas fases (execução e avaliação) podem ser subdivididas então em sete etapas adicionais. As fases no modelo de interação direta são como segue sendo que as quatro primeiras partes correspondem à fase de execução e as outras três correspondem à fase de avaliação:

* estabelecendo a meta;

* formando a intenção;

* especificando a sucessão de ação;

* executando a ação;

* percebendo o estado de sistema;

* interpretando o estado de sistema e

* avaliando o estado de sistema com respeito às metas e intenções.

Cada uma das fases é uma atividade do usuário. Durante a primeira fase o usuário forma uma meta. Esta é a necessidade do usuário do que precisa ser feito e é moldada em termos do domínio do usuário, no idioma de tarefa. Para não haver imprecisão na execução é necessário então ser traduzida a intenção mais específica e, definidas as ações atuais que alcançarão a meta, antes das ações serem executadas pelo usuário. O usuário percebe o novo estado do sistema depois da execução da sucessão de ações e interpreta isto em termos das suas expectativas. Se o estado do sistema reflete a meta do usuário, então o computador fez o que o usuário quis e a interação teve êxito; caso contrário o usuário tem de formular uma nova meta e repetir o ciclo.

A Interação pode ser vista como um diálogo entre o computador e o usuário. A escolha de estilo de interface pode ter um efeito profundo na natureza deste diálogo. Há várias interfaces e estilos, incluindo os comuns abaixo:

* Interface de Linha de Comando e Menus;

* Linguagem Natural (LN);

* Apontar-e-Clicar e

* Interfaces Tridimensionais

1.1. Interface de Linha de Comando e Menus

A interface de Linha de Comando foi o primeiro estilo de interface de diálogo interativo a ser usado e, apesar da disponibilidade de interfaces de Menus, é usada ainda amplamente. Propicia meios de expressar instruções diretamente ao computador (Figura 2), usando funções de tecla, caracteres únicos, abreviações ou comandos de palavras inteiras. Em alguns sistemas, a Linha de Comando é o único modo de se comunicação, especialmente para acesso remoto (Telnet). É utilizada mais comumente hoje a interface baseada em Menus, que permite acesso rápido à funcionalidade do sistema.

```
sable.soc.staffs.ac.uk> javac HelloWorldApp
javac: invalid argument: HelloWorldApp
use: javac [-g][-O][-classpath path][-d dir] file.java...
sable.soc.staffs.ac.uk> javac HelloWorldApp.java
sable.soc.staffs.ac.uk> java HelloWorldApp
Hello World!!
sable.soc.staffs.ac.uk>
```

Figura 2 - Interface de Linha de Comando (DIX *et al.*, 1998).

Interfaces de Linha de Comando são poderosas pois oferecem acesso direto à funcionalidade de sistema (ao invés da natureza hierárquica de Menus) e podem ser combinadas para aplicar várias ferramentas aos mesmos dados. Elas também são flexíveis: o comando tem freqüentemente várias opções ou parâmetros que poderão variar de algum modo e pode ser aplicado imediatamente a muitos objetos, sendo útil para tarefas repetitivas. Porém, esta flexibilidade traz dificuldade de aprendizado. Isto ocorre devido à necessidade de memorizar os comandos, já que nenhuma sugestão é disponibilizada na Linha de Comando para indicar qual comando é necessário. As Interfaces de Linha de Comando são então melhores para usuários especialistas que para novatos. Porém, este problema da quantidade de comandos pode ser reduzido usando comandos consistentes, significantes e abreviações. Os comandos usados devem ser condições dentro do vocabulário do usuário em lugar do técnico. Infelizmente, comandos são freqüentemente obscuros e podem variar entre os diferentes sistemas, o que poderá causar confusão ao usuário e com isso, dificultando a aprendizagem.

Na interface via Menu, as opções disponíveis ao usuário são exibidas ou ocultadas com o uso do *Mouse*, chaves numéricas ou alfabéticas (Figura 3). Porém, opções de Menu ainda precisam ser significantes e agrupadas logicamente para ajudar o reconhecimento. Freqüentemente Menus são ordenados hierarquicamente e a opção requerida pode não estar disponível na camada de topo da hierarquia. O agrupamento e a nomeação de opções de Menu permite uma sugestão para o usuário achar a opção desejada.

```
      PAYMENT                                                DETAILS
p3-7

      please select payment method:
      1.  cash
      2.  cheque
      3.  credit card
      4.  invoice

        9. abort transaction
```

Figura 3 - Interface de *Menu* (DIX *et al.*, 1998).

1.2. Linguagem Natural

Possivelmente, a Linguagem Natural (LN) seja o meio mais atraente de se comunicar com computadores, pelo menos à primeira vista. Usuários, incapazes de se lembrar de um comando ou em uma hierarquia de Menu mal organizada, podem almejar que o computador entenda instruções em palavras cotidianas. Entender a LN na contribuição da fala e da escrita é assunto de muito interesse e pesquisa. Por outro lado, a ambigüidade da LN dificulta o processamento para uma máquina entender a sintaxe, ou a estrutura de uma frase pode não estar clara. Veja o exemplo da Figura 4.

```
O homem bateu no menino com a vara
```

Figura 4 - Exemplo de Linguagem Natural (DIX *et al.*, 1998).

Não se pode estar seguro se a vara é o instrumento com que bateram no menino, ou se a vara é do menino (Figura 5).

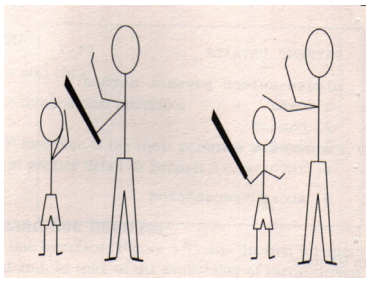


Figura 5 - Um homem com uma vara ou um menino com uma vara? (DIX *et al.*, 1998).

Estas interfaces, que são mais fáceis de aprender e usar mas estão limitadas em funcionalidade e poder, são apropriados para domínios restritos (particularmente sistemas de informação) e para iniciantes ou usuários casuais.

São usadas linguagens restritas para recuperar informação de um banco de dados. O Banco de dados usa frases da LN, mas na realidade requer sintaxe específica, como também conhecimento da estrutura de banco de dados.

1.3. Apontar-e-Clicar

Na Interface no estilo Apontar-e-Clicar percebe-se que a maioria dos sistemas multimídia e em *Browsers* virtuais, em todas as ações usam um único clique do *Mouse*. O usuário pode apontar em uma cidade em um mapa e quando se clica, uma janela abre, mostrando informações turísticas sobre a cidade. Pode-se apontar para uma palavra em algum texto e, quando clicado, vê-se uma definição da palavra. Aponta-se para um ícone e, quando clicado, alguma ação é executada.

Porém, a Interface no estilo Apontar-e-Clicar é mais simples e mais próxima das idéias de hipertexto. O estilo de Apontar-e-Clicar foi popularizado por páginas da *World Wide Web* que incorporam todos os tipos anteriores de navegação.

1.4. Interfaces Tridimensionais

No tocante a Interfaces 3D (tridimensionais), há um uso crescente de efeitos 3D em interfaces de usuário. O exemplo mais óbvio é RV; mas RV é só parte de uma gama de técnicas 3D disponível para o projetista de interface. A técnica mais simples é onde elementos podem ser áreas ativas.

Finalmente, há na RV a possibilidade do usuário poder interagir dentro de um mundo 3D simulado. Estes mecanismos se sobrepõem aos outros estilos de interação relacionados nos itens 1.1, 1.2 e 1.3.

O sistema a ser escolhido para fazer a Interação Humano-Computador tem de ser muito bem definido de acordo com algumas necessidades. Por exemplo, a presença de outras pessoas em um ambiente de trabalho afeta o desempenho do trabalhador em qualquer tarefa. A competição aumenta o desempenho, pelo menos em tarefas conhecidas. Semelhantemente, o desejo de impressionar a administração e os superiores melhora o desempenho nestas tarefas. Porém, quando vier à aquisição de novas habilidades, a presença destes grupos pode inibir desempenho, devido ao medo de fracasso. Por conseguinte, é importante permitir a privacidade para os usuários terem a oportunidade para experimentar.

Para executar bem a interação, os usuários devem ser motivados. Há várias fontes possíveis de motivação, como também essas já mencionadas, inclusive medo, submissão, ambição e presunção. O último destes é influenciado pela percepção do usuário da qualidade do trabalho feito que conduz a satisfação no cargo. Se um sistema torna difícil para o usuário executar tarefas necessárias, ele se frustrará ao usar, e, por conseguinte, o desempenho será reduzido.

O usuário também pode perder a motivação se um sistema é introduzido e não atende as exigências atuais do trabalho ser feito. Freqüentemente sistemas são escolhidos e introduzidos por gerentes em lugar de os próprios usuários. Em alguns casos a percepção do gerente do trabalho pode ser fundada na observação de resultados e não em atividades atuais. O sistema introduzido pode impor então um modo de trabalhar e isso se torna insatisfatório aos usuários. Se isto acontece pode haver três resultados: o sistema será rejeitado, os usuários ficarão desmotivados, ou o usuário adaptará a interação planejada às próprias exigências dele. Isto mostra a importância de envolver os usuários atuais no processo de escolha da interação.

A RV recorre à simulação de um mundo gerado por computador, ou um subconjunto disto no qual o usuário tem a sensação de imersão, representando assim o estado da arte em sistemas multimídia, mas concentra-se nos sensos visuais.

O termo Realidade Virtual leva a imaginar a imagem de um usuário com um Capacete ou Óculos (Figura 6)- ambos dispositivos não-convencionais -, aparentemente cegamente, tentando tocar ou pegar algo com o uso de um *Mouse* 3D (Figura 7) ou *Dataglove* (Figura 8) (luva) em um espaço vazio.

Figura 6 - Exemplos de Capacetes e Óculos 3D (ABS-TECH, 2005).

O usuário inserido e isolado dentro de um Ambiente Virtual (AV), se move por uma paisagem simulada, enquanto apanha objetos no mundo. Este é o cenário da Realidade Virtual Imersiva (RVI). Porém, essa é só uma parte da RV que também inclui comandos e situações de controle, aumentando a realidade, onde o virtual e o real se encontram.

Figura 7 - Exemplos de *Mouse* 3D (ABS-TECH, 2005).

Apesar da elevação exponencial da velocidade de processamento dos processadores, os sistemas de RVI ainda não estão acessíveis devido ao seu custo, e muitos destes sistemas são, principalmente, projetos de pesquisa.

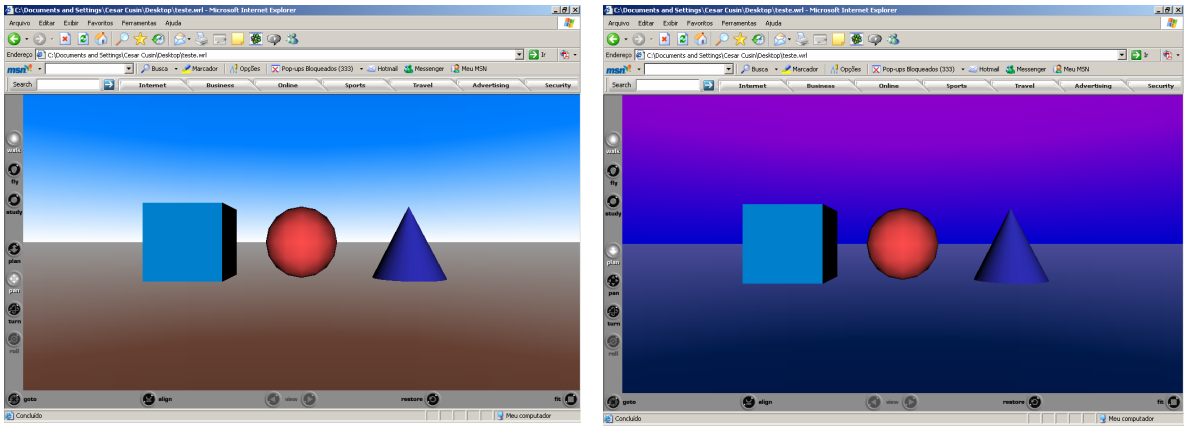
5DT Data Glove ***CyberGlove*** ***CyberGrasp*** ***Pinch Glove***

Figura 8 - Exemplos de *Datagloves* (ABS-TECH, 2005).

Rodello *et al.* (2003) lembra que "...o desenvolvimento de AVs com suporte a tais dispositivos não é trivial, existindo vários fatores a serem considerados quando da sua utilização...".

Sistema de RV não-imersivo é uma alternativa de abaixar o custo. Nestes sistemas, são apresentadas imagens 3D em uma tela de computador normal e a manipulação é feita com o uso do *Mouse* e do Teclado, em lugar de usar Óculos ou Capacetes para a visualização e *Dataglove*s (luvas) para a manipulação.

Esta forma de trabalhar com RV ficou acessível para muitos usuários com a ajuda do VRML que permite montar mundos virtuais para a *Web* integrados com outros materiais baseados também na *Web*. Mundos VRML podem incluir objetos 3D estáticos onde o usuário pode navegar ao redor e olhar para aspectos diferentes e objetos dinâmicos que se mudam e reagem quando colidem com o cursor do *Mouse* (Figura 9). Além disso, VRML é integrado com o resto da *Web* através de objetos de ligação que quando clicados os levem para outra página da *Web* ou outro mundo VRML.



Mundo Original ***Mundo Alterado depois que o cursor no Mouse "tocou" o cone***

Figura 9 - Exemplo de um Mundo VRML Original e Alterado posteriormente (AMES *et al.*, 1997).

1.5. Realidade Virtual

A RV, de uma maneira simplificada, é a forma mais avançada de interface do usuário com o computador até agora disponível (HANCOCK, 1995). Caracteriza-se também pela co-existência integrada de três idéias básicas: imersão, interação e envolvimento.

Existem requisitos para um sistema de RV. Dentre estes requisitos, destaca-se o de Interface de Alta Qualidade, por imitar o que acontece na interação do usuário com o mundo real. A Alta Interatividade, onde o ambiente deve reagir de maneira adequada às ações do usuário e permitir o maior número de ações de interação e a Imersão, permitindo que o usuário se sinta "dentro" do mundo virtual (Figura 10), seja com o seu corpo físico ou com uma representação qualquer (avatar).

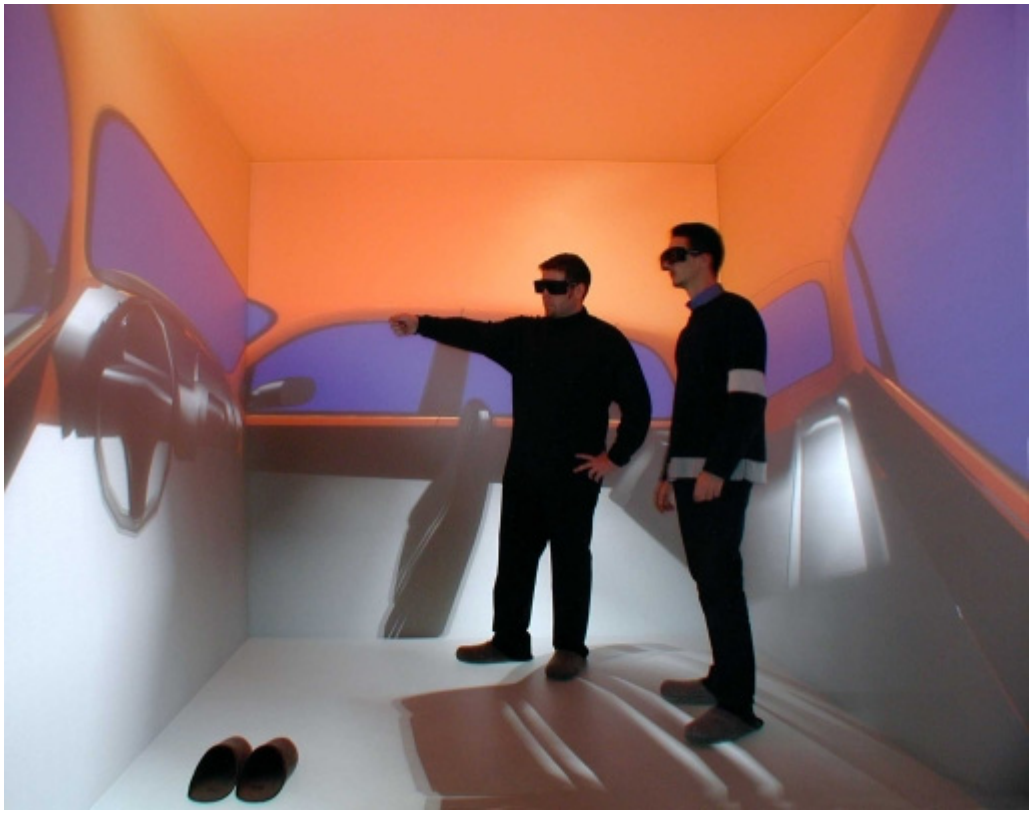


Figura 10 - Exemplo de usuário “dentro” do mundo virtual (TU, 2005).

É importante que, de alguma forma, o usuário seja “envolvido” pelo ambiente. A I-mersão não está inteiramente ligado a RVI; e por último, o Envolvimento, em que o sistema envolva o usuário com o ambiente e, assim, proporcionar novas formas de interação (NETTO *et al*., 2002).

As aplicações em RV são muitas e dentre elas destaca-se o seu uso na Medicina (Figura 11 e Figura 12). Neste caso, simulações de cirurgias laparoscópicas, em treinamentos com pacientes virtuais, no ensino da anatomia.

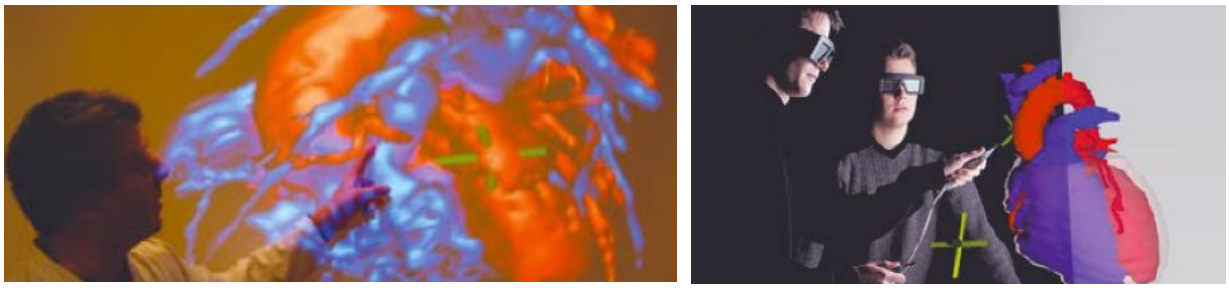


Figura 11 - Realidade Virtual na Medicina (RPTT, 2004).

Figura 12 - Visualizações Médicas (UNT, 2001).

Outra área em que a RV destaca-se é na educação, para modelar laboratório de física e química, mostrando o comportamento dos materiais. Ou ainda em entretenimento, jogos, simuladores de vôos, demonstrações (Figura 13) e treinamentos (Figura 14).

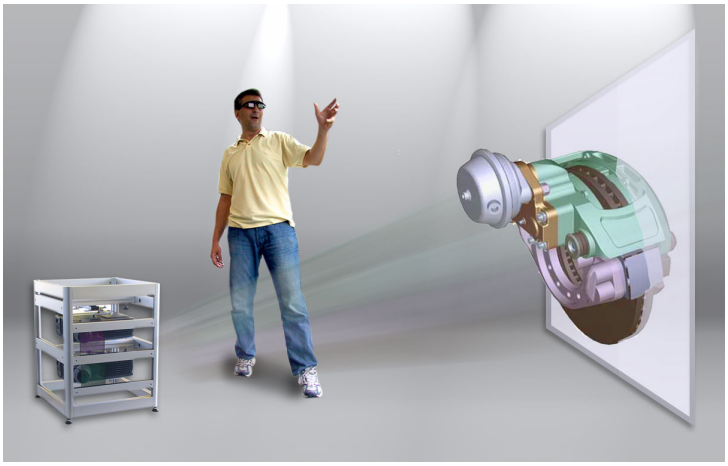


Figura 13 - Demonstração com o uso da Realidade Virtual (EMEDIAWIRE, 2005).

Figura 14 - Treinamento com o uso da Realidade Virtual (NIST, 2005).

Usa-se também a RV para visualização científica (Figura 15), de negócios (Figura 16), auditórios virtuais para promoções corporativas, teatro (Figura 17), tele-presença e sistemas de manutenção.

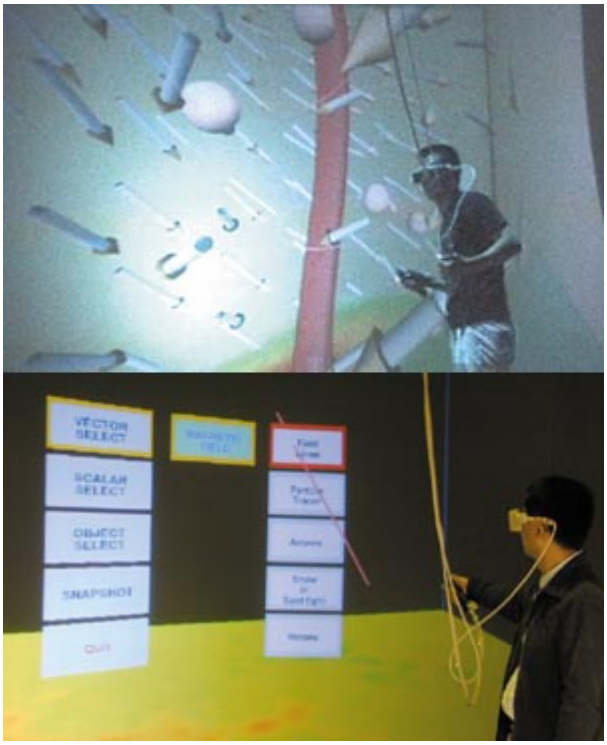


Figura 15 - Visualização Científica (NIFS, 2005).

Figura 16 - Apresentação de Negócios (QMI SOLUTIONS, 2005).

Figura 17 - Teatro em Realidade Virtual (STRAYLIGHT, 1994).

A maior restrição para as aplicações de RV ainda é o custo dos dispositivos (não-convencionais) de entrada e saída e o alto grau de computação envolvido (máquinas robustas).

2. MANIPULAÇÃO

A navegação na Internet é hoje em dia conceitualmente baseada no paradigma de hipertexto e teve início com o uso do *HyperText Markup Language* (HTML), hoje *Extensible HyperText Markup Language* (XHTML). A evolução de *hardwares* gráficos e linguagens como VRML permitem a montagem de mundos 3D para a apresentação de conteúdo em vários formatos, e é acessível para todos através da Internet.

A possibilidade representação de informação em espaço tridimensional é extremamente atraente. Aplicações que usam navegação de espaço poderiam apresentar informação dentro um modo muito mais natural. Vários exemplos podem ser citados como uma loja de departamentos, planejamento urbano, treinamento, visualização científica, medicina, tratamento de fobia, etc.

Um usuário de aplicação 3D exige um sistema fácil de usar e com boa qualidade visual. É desejável que este sistema trabalhe em uma variedade de plataformas, desde *desktops* até a Internet. Os usuários também esperam "guias" para ajudar a navegar. Personalização de acordo com o interesse de usuário e a ajuda de navegação de acordo com o conteúdo desejado tornando o mundo virtual mais realístico e agradável é a proposta para envolver usuários (FRERY *et al.*, 2002). Melhor seria um sistema que tenta descobrir em que o usuário poderia estar interessado para enfatizar o conteúdo mais pertinente (KELNER *et al.*, 1999).

Obviamente, um autor de aplicações 3D quer fazer um mínimo de trabalho com um máximo de eficiência. Autores atuais disponibilizam o famoso "arraste-e-solte" do *Mouse* para criar interfaces de usuário fora de componentes de *Graphical User Interface* (GUI) – Interface Gráfica do Usuário – pré-definidos (KELNER *et al.*, 1999).

Um dos assuntos cruciais é a navegação interativa e exploração destes ambientes 3D. Interfaces de RV são um dos tipos mais avançados de interface disponível, oferecendo muita

interação e possibilidades de exploração. Porém, é importante assegurar que o desenvolvimento de Ambientes Virtuais deve ser cuidadosamente feitos para permitir ao usuário ferramentas amigáveis e com a possibilidade de desfrutar as vantagens dessas interfaces.

Para resolver os problemas de navegação, exploração e recuperação de informação em Ambientes Virtuais 3D é necessário tornar o ambiente intuitivo de acordo com o interesse do usuário e oferecer ajuda na navegação de acordo com o conteúdo que o usuário quer explorar pela construção de rotas de navegação. Isso torna o Ambiente Virtual mais realístico e agradável, orienta o usuário dentro do mundo e o faz sentir mais envolvido (FRERY *et al.*, 2002).

As ferramentas 3D devem proporcionar uma aproximação comparável ao padrão predefinido dos objetos (modelos) por estruturar o espaço 3D. Além disso, as ferramentas devem ser utilizáveis para definir todos os aspectos de conteúdo, quer dizer, comportamento, conhecimento e representação dos objetos (KELNER *et al.*, 1999).

Pessoas exibem comportamentos diferentes quando expostas à mesma situação e, também, têm desejos diferentes que precisam ser resolvidos. Então, modelar um Ambiente Virtual 3D de tal modo que um usuário se sinta satisfeito é uma tarefa difícil. A adaptação do ambiente pela personalização do conteúdo e o oferecimento de interatividade provou que é uma boa solução para a exploração de mundos desconhecidos. Neste caso, o usuário passa a explorar mais intuitivamente as informações apresentadas sem ajuda adicional (FRERY *et al.*, 2002).

2.1. Manipulação de Objetos

De acordo com Mine *et al.* (1997), um Ambiente Virtual Interativo (AVI) é uma das promessas de um ambiente tridimensional no qual um usuário fará a manipulação de objetos virtuais 3D. O que motiva a pesquisa na RV é que isto conduzirá a uma Interação Humano-Computador com interfaces mais naturais. Resultados promissores obtidos em várias aplicações foram fundamentais como: Terapia de fobia, Entretenimento, Cirurgia, Exército, Manutenção, Arquitetura e Modelagem 3D dentre outras.

A manipulação de um objeto em um AVI consiste na mudança de algum parâmetro ou estado de um objeto previamente selecionado. Esta mudança de estado inclui orientação, posição, tamanho ou outro parâmetro qualquer, seja geométrico (forma ou posição), visual (cor ou textura) ou comportamental (iniciar movimento ou parar, por exemplo) (PINHO, 2002).

O número de aplicações para AV é crescente, ainda pequeno e com poucas aplicações, porém, já vão além dos laboratórios de pesquisa. Muitos destas aplicações com sucessos caem dentro do contexto do espaço da visualização. As aplicações exploram a visão intuitiva oferecida através de sistemas de RV, mas faz pouco uso de manipulação direta do objeto virtual. O pouco uso se explica pelas limitações tecnológicas, altos preços envolvidos e vários outros fatores menos óbvios que impedem o desenvolvimento de aplicações de AV.

A manipulação precisa de objetos virtuais é difícil (MINE *et al.*, 1997). As técnicas de manipulação mais comuns em Ambientes Virtuais são aquelas classificadas como de interação direta. Estas técnicas permitem ao usuário tocar virtualmente o objeto e utilizar os movimentos de seu corpo (mãos, braços, cabeça etc) para provocar mudanças na posição e/ou orientação deste objeto. Estas técnicas são classificadas como técnicas individuais (que será

vista no item 2.2), pois se referem à manipulação que um único usuário realiza sobre um objeto (PINHO, 2002).

Embora havendo imersão, especificação de visão e seis graus-de-liberdade (*Degree-of-Freedom* – DoF) que facilitam a manipulação de objetos virtuais, no mundo real faz-se isto e muito mais.

Na Tabela 1 pode-se observar uma comparação entre tarefas do mundo real e as implicações de realizá-las num AV. Pela análise desta tabela, nota-se claramente que a utilização de sistemas de RV, de forma geral, ainda depende de um grande desenvolvimento na área das técnicas de manipulação.

Mine *et al.* (1997) oferece uma solução para aumentar o número de aplicações de RV: use o que você tem. Sem toque, o usuário não pode sentir o ambiente e nem usa a colisão para manipular objetos. É imperativo, então, tirar proveito de uma coisa fazendo com que o usuário ainda sinta no mundo virtual o seu corpo. O senso de uma pessoa, da posição e orientação do corpo dele e suas várias partes é a idéia de Mine *et al.* (1997). Um usuário pode tirar proveito durante esse tipo de interação de pelo menos três modos:

* Se um objeto virtual fica diretamente situado à posição da mão do usuário, o usuário tem um bom senso de posição do objeto podendo fazer a manipulação direta;

* Usuários podem armazenar objetos virtuais em Menus. Se um usuário já não pode sentir o mundo ao seu redor, ele pode selecionar e usar controles virtuais;

* Gestos podem ser usados para executar ações invocando comandos ou comunicar informação.

Tabela 1 - Características de tarefas do Mundo Real no Mundo Virtual (PINHO, 2002).

Tarefa	Mundo Real	Ambientes Virtuais
Manipulação de objetos	A manipulação de objetos é usualmente feita com ferramentas ou com a mão	A seleção de ferramentas é complicada
Comunicação e comandos através de voz	A possibilidade de comunicação com outros usuários através de voz é de fundamental importância no processo interativo entre usuários	A tecnologia de reconhecimento de voz ainda é precária
Medição de objetos	A possibilidade de medir objetos do ambiente é bastante natural para aplicações reais	Ainda é difícil e pouco precisa a possibilidade de medir objetos em Ambientes Virtuais
Anotação de informações sobre os objetos do ambiente	A anotação de informações textual e gráfica sobre papel ou quadros de aviso é extremamente simples e útil no processo de interação em ambientes reais	A entrada de textos e números é pouco desenvolvida em Ambientes Virtuais

Navegar é a dificuldade mais comum que os usuários enfrentaram em ambientes 3D (FRERY *et al.*, 2002). Esta dificuldade ocorre, possivelmente, devido ao fato dos usuários não terem experiência com o uso de ferramentas de navegação oferecidas por *plug-ins* e *Browsers* para a visualização das cenas. Mas, na maioria dos casos, o problema está no fato dos ambientes 3D ficarem grandes e não oferecerem a própria ajuda de navegação. Com isso, o usuário, perdido, não recebe auxílio do ambiente. Este aspecto deveria ser levado seriamente em consideração durante a construção de mundos 3D.

Muitas pesquisas foram feitas propondo soluções para a navegação e problemas de desorientação. A maioria das soluções aponta para mundos virtuais que ofereçam estratégias de navegação de espaço, ajudando o usuário a explorar e realizar eficazmente seus objetivos, até mesmo se o usuário não tiver conhecimento prévio de como se mover e localizar dentro do ambiente.

Uma vantagem de mundos 3D é que eles permitem visualização de diferentes pontos de vista e, pela definição de pontos de vista pertinentes, o usuário pode ter uma idéia geral do mundo. Algumas estratégias para navegação sugerem o uso de rotas para a visualização do mundo inteiro. Rotas de navegação são um método simples, mas eficiente, para criação de animação. Elas estão compostas por uma coleção de pontos que indicam por quais posições

um objeto passará durante o processo de navegação, enquanto também estabelecem a orientação dentro do qual este objeto visualizará o ambiente durante a animação.

Outro aspecto relaciona a construção de rotas onde estas têm de mostrar o ambiente de perspectivas diferentes. Isto insinua que é interessante que o usuário visualize o ambiente sob a perspectiva que eles teriam dentro do mundo real, de acordo com sua altura ou idade. Esta estratégia permite uma melhor visualização do ambiente e seu conteúdo, e pode trazer maior empatia com o mundo virtual. Rotas que permitam aproximação dos objetos selecionados durante a navegação para visualização mais detalhada de partes específicas do conteúdo, como para um quadro em uma sala de museu, também deveriam ser oferecidas pelo sistema de apoio à navegação (FRERY *et al.*, 2002).

2.2. Metáforas Interativas Individuais

As técnicas interativas individuais têm uma especificação que define quais ações devem ser executadas por um usuário para controlar o objeto e quais os efeitos destas ações sobre o estado corrente deste objeto. Esta especificação recebe o nome de metáfora interativa ou metáfora de interação individual. Em outras palavras, a metáfora interativa define um mapeamento entre as ações do usuário e seus efeitos sobre o objeto.

É fundamental mencionar que uma metáfora interativa define tanto o mapeamento entre os movimentos do usuário e do objeto, como o que deve ser feito, por exemplo, nos momentos em que o usuário “segura” ou “solta” um objeto. A modelagem das técnicas interativas é dividida em quatro etapas distintas: seleção, anexação, posicionamento e liberação. Na Tabela 2 apresenta-se o significado de cada etapa.

Tabela 2 - Etapas de uma técnica interativa de manipulação (PINHO, 2002).

Etapa	Descrição
Seleção	Especifica o método usado para indicar um objeto a ser manipulado
Anexação	Especifica o que acontece no instante em que o objeto é apreendido pelo usuário (ou preso a seu apontador)
Posicionamento	Especifica como os movimentos do usuário e de seu apontador afetam o movimento do objeto
Liberação	Especifica o que acontece no instante em que o objeto é solto pelo usuário

2.3. Sistemas Multimodais

Uma aplicação Multimodal propicia uma interface interativa na qual o usuário pode manipular, escrever, falar ou clicar (por exemplo, botões, barras de rolagem, etc) (CHEYER; JULIA, 1999).

Sistemas Multimodais são particularmente apropriados para aplicações nas quais os usuários interagem com o Ambiente Virtual. Nas aplicações citadas em Kehler *et al.* (1998), os sistemas usam escrita, reconhecimento de gestos, interpretação de LN, botões, barras de rolagem, reconhecimento de voz e em Cheyer *et al.* (1998) usam também caixas de texto e Menus.

Segundo Martin *et al.* (1998), não há nenhum acordo real entre as definições de "mídia" e "modalidade". Uma "mídia" é um dispositivo físico que habilita a troca de informação entre o usuário e o computador. Exemplos de mídia são Teclados, *Mouse*, Microfone, Tela e Alto-falante. Uma "modalidade" é o modo para explorar mídias específicas. São exemplos de modalidades: Linhas de Comando, Linguagem Natural Falada e *Mouse* 2D.

São transmitidos "pedaços" de informação por várias modalidades do usuário para o computador durante as interações, e vice-versa. Para "Multimodal", é proposta a seguinte definição: "a cooperação entre várias modalidades em ordem para melhorar a interação".

Como uma nova geração de Sistemas Multimodais começa a se definir, um tema que freqüentemente emerge é a integração e exigências de sincronização para combinar modos diferentes em um sistema inteiro (OVIATT *et al.*, 1997). De uma perspectiva lingüística, o uso em comum de modos naturais, como fala e gestos, foi descrito durante a interação humano-humano, o papel do gesto em discurso e em pensamento. O gesto foi visto como uma ajuda cognitiva dentro da realização de pensar e, também, como portador de conteúdo semântico diferente que fala.

Fala e gestos são portadores de materiais diferentes e não são redundantes, mas estão relacionados e assim, a tensão necessária entre eles leva o pensamento adiante.

Na aplicação feita por Oviatt *et al.* (1997), os usuários tiveram uma forte preferência a interagir usando o Sistema Multimodal durante as tarefas, ao invés de Unimodal. Todos eles (100%) usaram tanto a fala quando a escrita durante cada tarefa. Durante as entrevistas, 95% dos usuários informaram também uma preferência para a Interação Multimodal.

Petrelli *et al.* (1997) aborda que em usuários experientes, apenas 84% usaram o Sistema Multimodal composto de Linguagem Natural Escrita e *Mouse*. Por outro lado, dos usuários inexperientes, somente 30% usaram o Sistema Multimodal.

A comunicação Multimodal é freqüentemente e eficazmente usada pelos humanos para identificar objetos em um espaço físico. Isto ocorre, combinando modalidades diferentes, por exemplo, a fala e os gestos. Referências Multimodais agem como ferramentas eficientes ao interagir com a complexidade do espaço físico. Estudos em humanos com o uso de idioma natural para a interação com computador confirmam peculiaridades gerais. Falando para um computador, os humanos mantêm uma comunicação sociável, mas tendem a simplificar a estrutura sintática e reduzir o comprimento de expressões vocais. Em outras palavras, os usuários selecionam um registro simples para interagir com computadores de forma que ele se iguale (simule) às capacidades humanas.

Questiona-se se a Interação Multimodal instintiva realmente faz com que usuários sem experiência executem tarefas como usuários experientes.

De acordo ainda com Petrelli *et al.* (1997), em geral, Sistemas Multimodais parecem ajudar quando permite aos humanos se comunicar de uma forma mais espontânea. Então, a pessoa poderia deduzir que a comunicação Multimodal é o melhor estilo de interação para usuários sem experiência. Porém, alguns autores sugerem que linguagem baseada na interação é mais satisfatória para usuários experientes.

Para Oviatt (1999) o sucesso dos Sistemas Multimodais dependerá de conhecimento dos padrões de integração naturais que tipificam pessoas e do uso combinado de modos de contribuição diferentes. Alertando para o fato de os usuários preferirem interagir de forma Multimodal não é nenhuma garantia que eles emitirão todos os comandos a um sistema na forma Multimodal. Ao contrário, eles sempre intercalam entre a forma Unimodal e Multimodal. Predizer se um usuário expressará um comando usando a forma Multimodal também depende do tipo de ação que eles estão executando.

Para os que afirmam que um Sistema não é Multimodal sem a contribuição da fala, Oviatt (1999) afirma que a fala não é a portadora exclusiva de conteúdo importante, nem tem precedência temporal em cima de outros modos de contribuição.

Mais uma vez Oviatt (1999) lembra que freqüentemente dizem que o conteúdo carregado por modos diferentes durante a comunicação Multimodal contém um alto grau de redundância. Porém, o tema dominante entre os usuários Multimodais é que os modos diferentes levam à complementaridade de conteúdo, não redundância. Durante comunicação humana, os lingüistas defendem que a fala espontânea e gestos não envolvem informação duplicada. Os dados atuais realçam a importância de complementaridade como um tema organizacional principalmente durante a comunicação Multimodal (OVIATT, 1999).

Na verdade, a dúvida reside sobre a reconhecida propensão ao erro com o uso de tecnologias individuais, como por exemplo, caneta e voz sendo combinadas em um Sistema Mul-

timodal, se causam ou não ainda mais erros. Em uma Interface Multimodal, as pessoas evitarão usar um modo de contribuição que eles acreditem que o conteúdo com certeza será propenso ao erro. A linguagem também é mais simples, minimizando assim os erros. Quando um erro de reconhecimento acontecer, os usuários alternam modos de contribuição e de certo modo isso tende a solucionar o problema efetivamente.

Outro resultado relevante apontado na pesquisa de Oviatt (1999) é que todos os usuários de Sistema Multimodal usaram comandos integrados de um modo uniforme. O padrão de integração dominante de cada usuário era identificado quando eles começaram a interagir com o sistema, e então persistiu ao longo da sessão. Quer dizer, o padrão de integração de cada usuário era estabelecido logo no início e permaneceu constante. Estes resultados permitem aos desenvolvedores de Sistemas Multimodais descobrir e adaptar à integração dominante de um usuário padrão podendo assim conduzir consideravelmente para a melhora nas taxas de reconhecimento.

Essa pesquisa constatou também que modos de contribuição diferentes são capazes de comparar o conteúdo transmitido. Modos diferentes variam basicamente dentro do grau para o qual eles são capazes de transmitir informação semelhante, com alguns modos relativamente mais comparáveis (fala e escrita) e outros menos (fala e olhar). Embora fala e escrita possam carregar muitos conceitos semelhantes, eles ainda diferem na gama e precisão da sua expressividade.

É sabido que o aumento da velocidade e eficiência pela contribuição paralela é primeira vantagem de um Sistema Multimodal se comparado com um sistema Unimodal ou de interface gráfica. Por exemplo, durante Interação Multimodal usando voz e caneta em um domínio de espaço, um aumento de velocidade acima de 10% foi documentado em comparação com uma interface somente de fala. Um terço da vantagem mais significante é a flexibilidade que Sistemas Multimodais permitem aos usuários selecionando e alternando entre os

modos de contribuição. Também permite uma diminuição de erro significativa e recuperação mais fácil, como previamente discutido.

2.4. Evoluções

Nos últimos anos, *hardware* e *software* evoluíram a um tal ponto onde a maioria dos computadores são capazes de apresentar conteúdos 3D de alta qualidade, principalmente por causa do *hardware* especializado (aceleradores 3D).

Vários formatos de mídia disponíveis e ferramentas associadas permitem que os autores criem tais conteúdos (VRML, Java3D, X3D etc.) (KELNER *et al.*, 1999).

2.5. Ambientes

Atualmente, há vários Ambientes Virtuais baseados em texto. Porém, estes sistemas não oferecem para os usuários as habilidades para interagir com o ambiente (modificação do ambiente etc.).

Se não são pretendidos Ambientes Virtuais para modelar a realidade, eles normalmente não têm nenhuma noção de tempo ou gravidade, o que pode confundir os usuários. Além disso, Ambientes Virtuais tendem a ficar bastante complexos. Então, são necessários "guias" ou interfaces dedutíveis que auxiliem na navegação e, no final das contas, ajude no processo de recuperação de informação (KELNER *et al.*, 1999).

3. LINGUAGEM NATURAL CONTROLADA

3.1. Sintaxe e Semântica

A Sintaxe trata das propriedades livres da linguagem como, por exemplo, a verificação gramatical de programas. A Sintaxe basicamente manipula símbolos sem considerar os seus correspondentes significados.

A Semântica objetiva dar uma interpretação para a linguagem como, por exemplo, um significado ou valor para um determinado programa. Para resolver qualquer problema real, é necessário dar uma interpretação semântica aos símbolos como, por exemplo, "estes símbolos representam os inteiros" (MENEZES, 2000).

Segundo Gries (1975), Sintaxe é a estrutura e Semântica é o significado.

3.2. Alfabeto e Palavra

O Dicionário Aurélio (FERREIRA, 1999) define Linguagem como "o uso da palavra articulada ou escrita como meio de expressão e de comunicação entre pessoas".

Um Alfabeto é um conjunto de Símbolos. Um Símbolo (ou caractere) é uma entidade abstrata básica a qual não é definida formalmente. Letras e dígitos são exemplos de símbolos freqüentemente usados.

Uma Palavra, Cadeia de Caracteres ou Sentença sobre um alfabeto é uma seqüência finita de Símbolos (do Alfabeto) justapostos (MENEZES, 2000).

3.3. Linguagem Formal

Uma Linguagem Formal é um conjunto de palavras sobre o Alfabeto (MENEZES, 2000).

3.4. Linguagens Regulares

Os formalismos que representam Linguagens Regulares são de pouca complexidade, grande eficiência e fácil implementação. Entretanto a Classe das Linguagens Regulares é muito restrita e limitada, sendo fácil definir linguagens que não são regulares (MENEZES, 2000).

3.5. Linguagem Livre do Contexto

Uma linguagem é dita Linguagem Livre do Contexto se for gerada por uma Gramática Livre do Contexto.

O nome “Livre do Contexto” se deve ao fato de representar a mais geral classe de linguagens cuja produção é da forma $A \rightarrow \alpha$. Ou seja, em uma derivação, a variável A deriva α sem depender (“Livre”) de qualquer análise dos símbolos que antecedem ou sucedem A (“Contexto”) na palavra que está sendo derivada (MENEZES, 2000).

3.6. Gramática

Ferreira (1999) define gramática como “...a arte de falar e de escrever bem em uma língua...”.

Já Borba (1971) diz que a gramática em sentido amplo diz respeito à organização da língua, isto é, é o conjunto de elementos estruturalmente dispostos que compõem a língua. Em sentido restrito, a gramática é o estudo das formas e suas combinações encaradas de um ponto de vista exclusivamente lingüístico. Abrange assim a morfologia e a sintaxe de um determinado estado de língua.

O papel da gramática é explicar a constituição da língua, ou seja, identificar, caracterizar, relacionar e classificar as formas lingüísticas apresentando as oposições do sistema realizadas ou apenas possíveis. A gramática mostra o que realmente se usa ou o que se espera que se use em determinadas situações (BORBA, 1971).

3.7. Gramática Livre do Contexto

A Classe das Linguagens Livre do Contexto contém propriamente a Classe das Linguagens Regulares. Seu estudo é de fundamental importância na informática, pois compreende um universo mais amplo de linguagens (comparativamente com as regulares) tratando, adequadamente, questões como parênteses balanceados, construções bloco-estruturadas, entre outras, típicas de linguagens de programação como Pascal, C. Algol, etc. Os algoritmos reconhecedores e geradores que implementam as Linguagens Livres do Contexto são relativamente simples e possuem uma boa eficiência. Exemplos típicos de aplicações dos conceitos e resultados referentes às Linguagens Livres do Contexto são analisadores sintáticos, tradutores de linguagens e processadores de texto em geral.

Em resumo; a Gramática Livre do Contexto é uma gramática onde as regras de produção são definidas de forma mais livre que na Gramática Regular (MENEZES, 2000).

3.8. Linguagem Natural Controlada

Atualmente as fases de aquisição e refinamento de conhecimento que compõem o processo de construção de uma base de conhecimento ocorrem de forma indireta, sendo, geralmente, realizadas de duas formas. A primeira, e mais comum, envolve um grupo de usuários com o domínio da linguagem e um grupo de engenheiros de conhecimento que conhecem a Linguagem de Representação de Conhecimento (LRC) formal empregada no sistema que gerenciará a base. A segunda, e mais recente, envolve a utilização de algoritmos de aprendi-

zado de máquina que extraem o conhecimento de uma massa de dados provenientes do domínio da linguagem.

Estes processos, em geral, são muito demorados e custosos e, além disso, apresenta um grande risco de geração de inconsistências na base (DA SILVA; PINHEIRO, 2003). Estas inconsistências são mais freqüentes na primeira forma citada e são inerentes ao fato de que em um processo de comunicação, como o que ocorre entre os conhecedores da linguagem e os engenheiros de conhecimento, a interpretação dada para um mesmo elemento do domínio pelos diferentes integrantes do processo é sempre diferente. Na segunda forma, o maior problema está na escolha de um algoritmo adequado para o domínio em questão e na adequação do conjunto de dados às limitações impostas por este algoritmo.

Uma maneira de evitar o problema de interpretação é possibilitar aos profissionais ou usuários que dominam a linguagem fazerem a aquisição de conhecimento de forma direta. Para tal, é necessário que a LRC utilizada nesta tarefa seja de fácil compreensão e uso para ser entendida e usada por estes usuários. No entanto, as LRC atuais mais comumente usadas, geralmente, são muito complexas em sua estrutura sintática e semântica para que estes profissionais a utilizem de forma direta e natural. De fato, a maioria destas linguagens foi criada tendo, basicamente, a eficiência computacional como prioridade e acabaram deixando de lado aspectos comunicativos que qualquer LRC deve apresentar.

O objetivo é que o uso de uma linguagem, que valorize os aspectos comunicativos, possa permitir trazer os usuários com o domínio da linguagem para o primeiro plano nas fases de aquisição e refinamento do conhecimento, reduzindo, assim, seus custos e riscos. Mais especificamente, o uso de uma Linguagem Natural Controlada (LNC) – um subconjunto da LN cujo léxico, sintaxe e semântica são restringidos para um domínio especifico – como um novo nível de interação com a base. Recentemente, as LNCs são propostas e usadas com relativo sucesso em várias tarefas, mostrando que, dentro de um domínio específico, esta pode ser

uma boa alternativa para possibilitar que os peritos no domínio se comuniquem diretamente com o sistema de gerenciamento da base de conhecimento.

Uma LRC é uma linguagem formal empregada no processo de representação de conhecimento. Em geral, sua estrutura está intimamente ligada ao tipo de conhecimento a se representar e ao tipo de inferência que se deseja realizar sobre este conhecimento.

O maior problema com estas linguagens é que elas não fazem parte do dia-a-dia dos usuários finais – os peritos no domínio –, sendo, portanto, necessário um árduo processo de aprendizado, caso estes as queiram utilizar.

Para tornar o processo de aquisição de conhecimento e refinamento de uma base de conhecimento um processo acessível aos *experts* no domínio é necessário disponibilizar uma LRC que seja de fácil aprendizado e uso. Para isto, é necessário que ela contemple mecanismos comunicativos que sejam conhecidos destes usuários.

Olhando por este aspecto, a candidata mais apropriada seria o uso da LN, visto que os usuários já a conhecerem. Entretanto, além da ambigüidade inerente a tal linguagem, o processamento de sua forma irrestrita ainda não é viável, apesar dos enormes avanços da área de processamento de LN.

Contudo, o uso de uma LNC (um subconjunto da LN específico para comunicação dentro de um domínio) tem sido explorado em tarefas como a especificação de requisitos e a representação de conhecimento entre outras, com relativo sucesso. A grande vantagem de se usar uma LNC está no fato de que todas as sentenças usadas nesta linguagem estão corretas na LN da qual ela faz parte, mas nem todas as sentenças desta LN são permitidas na linguagem controlada, o que possibilita a eliminação das sentenças ambíguas. Assim, um *expert* no domínio terá somente de aprender quais os tipos de sentenças que ele pode e quais ele não pode usar na representação do conhecimento, o que se torna uma tarefa bem mais simples do que aprender uma linguagem inteiramente nova (DA SILVA; PINHEIRO, 2003).

LNC é um subconjunto da LN que pode ser processada com precisão e eficazmente por um computador, mas é expressiva bastante para permitir uso natural por qualquer pessoa (FUCHS; SCHWITTER, 1995). Ele ainda completa: "...Linguagem Natural Controlada – um subconjunto de Linguagem Natural caracterizado por uma gramática restringida e um vocabulário de aplicação específica..."

Zaïane *et al.* (1997) salienta o problema dos sites de pesquisa que indexam documentos *on-line* de uma forma simples, sem muitas regras. Isto era aceitável para uma rede de informação global relativamente pequena ou um banco de dados de documento local. Porém, com o avanço da Internet, menos satisfatórias as respostas das pesquisas nos *sites* de busca se tornarão. Alguns sites de pesquisa agregaram características novas, como dados do documento, domínio de Internet de origem (local) e, até mesmo, o formato do documento estreita a procura.

Um sistema de recuperação de documento ideal deveria permitir o uso da LN (ou uma pseudolinguagem natural); uma LNC; e deveria indexar documentos baseado nos conceitos presente neles (ZAÏANE *et al.*, 1997).

No entanto, é importante salientar que na definição de uma LNC não basta introduzir ricos mecanismos de comunicação a uma LRC. É necessário que estes mecanismos sejam realmente empregados pelos *experts* no domínio na qual a LNC será usada. Além disso, é extremamente importante que o conjunto de mecanismos comunicativos introduzidos permita manter o processamento computacional da LNC dentro de uma faixa de eficiência razoável, de forma a não inviabilizar sua implementação.

O objetivo é definir uma LNC que possa funcionar como um nível superior de interação para as tarefas de aquisição e refinamento de conhecimento, atuando como uma camada acima da LRC empregada no gerenciamento da base. O intuído final desta camada é prover mecanismos comunicativos que habilitem a LRC a ser usada por usuários finais (DA SILVA; PINHEIRO, 2003).

Uma especificação de linguagem chamada "*Attempto*" foi proposta por Schwitter e Fuchs (1996). A idéia geral está nas associações entre a familiaridade da LN com o rigor das linguagens formais. "*Attempto*" força padrões de escrita que restringem a gramática e o vocabulário, enquanto conduzindo assim para documentos que contêm linguagem mais previsível e menos ambígua.

Controlar ou simplificar a linguagem não é uma idéia nova. Porém, poucos pesquisadores tentaram empregar a LNC para especificações mais exigentes já que isto leva a restrições sintáticas e semânticas adicionais para o idioma (FUCHS; SCHWITTER, 1996).

Uma especificação de sistema segundo Fuchs e Schwitter (1995); é uma declaração dos serviços que um *software* propicia a seus usuários. Deveria ser escrito de um modo conciso que seja compreensível por todos os potenciais usuários do sistema. Porém, esta meta é difícil de se alcançar quando são expressas especificações em LN. O grande desafio é melhorar a qualidade de especificações sem se desfazer da "*readability*". A proposta de Fuchs e Schwitter (1995) é restringir o uso de LN em especificações para um subconjunto controlado com uma sintaxe e semântica bem definida. Por um lado este subconjunto deveria ser expressivo o bastante para permitir uso natural por usuários sem experiência, e por outro lado, a linguagem deveria ser processada eficazmente e com precisão pelo computador.

Osborne e Macnish (1996) também usam LNC na especificação de sistemas. A linguagem controlada foi usada em uma variedade de aplicações. Por exemplo, uma indústria de aeronaves desenvolveu uma linguagem controlada projetada para uso do pessoal de manutenção que não são ingleses nativos. Este padrão foi usado pela Boeing desde 1990. A "*Caterpillar Tractor Company*" tem projetado e adaptou a linguagem controlada durante os últimos 15 anos. Realmente, pesquisas da Universidade de Washington e a Boeing levaram a cabo experiências que sugerem que documentos escritos em linguagem controlada são mais compreensíveis que documentos escritos em LN.

Schwitter e Fuchs (1996) entendem a LN como um meio de comunicação familiar que tem uma longa tradição em criar exigências. Porém, o uso descontrolado da LN conduz para a ambigüidade, imprecisão e especificações obscuras. Além disso, exigências mudam e exigências novas surgem de forma que a especificação está sujeita a mudanças freqüentes. Algumas pessoas defendem o uso da linguagem formal para eliminar alguns problemas associados com a LN. Porém, por causa da necessidade de compreensão, não se pode substituir documentos escritos dentro da LN por especificações formais em todos os casos. Daí então a idéia de usar a LNC.

A proposta é reduzir os custo e riscos dos processos aquisição de conhecimento e refinamento de bases de conhecimento trazendo os técnicos no domínio para o primeiro plano destas tarefas. Para tal, o uso de uma LNC como uma nova camada de interação com a LRC da base.

A vantagem direta de se usar uma LNC na aquisição de conhecimento advém do fato do usuário não ter que aprender uma nova linguagem. Em vez disso, ele terá apenas que se acostumar com os tipos de sentenças que são permitidas na LNC proposta. Uma outra vantagem relevante é o fato de se permitir que o usuário use recursos naturais de comunicação da LN, como as figuras de linguagem. Estes mecanismos aumentam a coesão textual reduzindo a quantidade de texto a escrever e interpretar, tornando a tarefa de aquisição e refinamento de conhecimento um processo muito mais natural para os peritos no domínio.

É necessário dizer que a simples definição de uma LNC não resolve o problema de aquisição de conhecimento. É preciso também que a interface do ambiente de atualização auxilie o usuário nesta tarefa, orientando seus passos durante o processo, como se o usuário estivesse seguindo um roteiro (DA SILVA; PINHEIRO, 2003).

3.9. Compiladores

De forma simples, um compilador é um programa que lê um programa escrito numa linguagem – a linguagem fonte – e o traduz num programa equivalente numa outra linguagem – a linguagem alvo (Figura 18). Como importante parte desse processo de tradução, o compilador relata a seu usuário a presença erros no programa fonte.

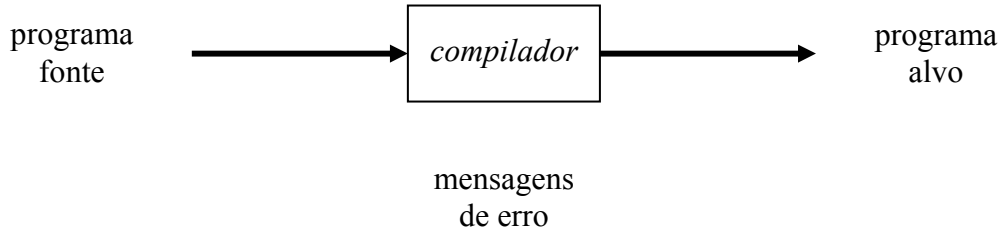


Figura 18 - Um compilador (AHO *et al*., 1995).

Existem duas partes na compilação: a análise e a síntese. A parte de análise divide o programa fonte nas partes constituintes e cria uma representação intermediária do mesmo. A de síntese constrói o programa alvo desejado, a partir da representação intermediária (AHO *et al*., 1995).

4. XML (*EXTENSIBLE MARKUP LANGUAGE*)

4.1. Extensible Markup Language (XML)

O *XML Working Group* da *World Wide Web Consortium* (W3C) desenvolveu a XML (*Extensible Markup Language*), que está relacionada a SGML (*Standart Generalized Markup Language*), em 1996. A XML é uma tecnologia aberta, amplamente suportada (isto é, tecnologia não-privada) para troca de dados (BIGGS; EVANS, 2001).

Os documentos XML só contêm dados, sem instruções formatadas. Desse modo, os aplicativos que processam os documentos XML devem decidir como exibir os dados do documento. Por exemplo, o PDA (*Personal Digital Assistant*) pode exibir um documento XML de maneira diferente da que um celular ou um computador *desktop* exibiriam esse mesmo documento.

A XML permite criar marcação para quase todos os tipos de informações. Essa extensibilidade permite criar linguagens de marcação inteiramente novas para se descrever tipos de dados específicos, incluindo fórmulas matemáticas, estruturas moleculares, músicas, receitas, etc. Entre as linguagens de marcação baseadas em XML, incluem-se XHTML, MathML (para matemática), VOICEXML™ (para reconhecimento de voz), SMIL™ (a *Synchronous Multimedia Integration Language* – para representações de multimídia), CML (*Chemical Markup Language* – para química) e a XBRL (*Extensible Business Reporting Language* – para troca de dados financeiros) (DEITEL *et al*., 2003).

Os elementos de XML descrevem os dados contidos nesses elementos. Assim, os programas de processamento de XML podem pesquisar, classificar, manipular e exibir os

documentos XML utilizando tecnologias como XSL (*Extensible StyleSheet Language*) ou CSS (*Cascading Style Sheets*).

Os documentos XML são altamente portáveis. Visualizar ou modificar um documento XML não requer nenhum *software* especial. Qualquer editor de textos que suporta caracteres ASCII/Unicode pode abrir documentos XML para visualização e edição. Uma característica importante da XML é que ela é legível tanto pelo olho humano como por máquina.

Processar um documento XML exige um programa de *software* denominado analisador sintático de XML (XML *Parser*, ou um processador de XML). A maioria dos analisadores sintáticos de XML está disponível gratuitamente para uma variedade de linguagens de programação (como Java, Pyton, C++, etc.). Os analisadores sintáticos verificam a sintaxe de um documento XML e permitem aos *softwares* processar os dados marcados. Os analisadores sintáticos de XML podem suportar o DOM (*Document Object Model* – Modelo de Objeto de Documento) ou a SAX (*Simple API for XML*).

Os analisadores sintáticos baseados em DOM constroem uma estrutura de árvore que contém dados de um documento XML na memória. Eles permitem que os *softwares* manipulem dados em um documento XML. Os analisadores baseados em SAX processam documentos XML e geram eventos quando o analisador encontra marcas, texto, comentários, etc. Esses eventos contêm dados do documento XML. Os *softwares* podem "ouvir" esses eventos para obter dados a partir do documento XML.

Esses analisadores sintáticos; DOM e SAX; são vistos com mais detalhes nas seções 4.4 e 4.5, respectivamente.

Um documento XML pode opcionalmente fazer referência a um documento que define a estrutura desse documento XML. Esse documento é uma DTD (Definição de Tipo de Documento) ou um Esquema. Quando um documento XML fizer referência a uma DTD ou a um Esquema, alguns analisadores sintáticos (chamados de analisadores sintáticos de validação – *validating parsers*) podem ler a DTD e o Esquema, verificando se o documento segue a

estrutura que a DTD ou o Esquema definem. Se o documento obedecer a DTD ou ao Esquema (isto é, o documento tiver a estrutura apropriada), o documento de XML é válido. Os analisadores sintáticos que não podem verificar a conformidade do documento em relação as DTDs e aos Esquemas são analisadores não-validadores. Se um analisador sintático de XML (validador ou não-validador) puder processar um documento XML com sucesso, esse documento está bem-formatado (isto é, sintaticamente correto). Por definição, o documento XML válido também é bem-formatado.

4.2. Definições de Tipo de documento (DTDs)

A DTD permite que um analisador sintático de XML verifique se um documento XML é válido (isto é, seus elementos contêm os atributos adequados e estão na seqüência adequada, etc.). A DTD expressa o conjunto de regras para estrutura de documentos utilizando uma gramática EBNF (*Extended Backus-Naur Form*) – não uma sintaxe de XML (BIGGS; EVANS, 2001).

Se o documento fizer referência a uma DTD e esse documento contiver algum elemento ou atributo que a DTD não define, o documento é inválido.

4.3. Documentos de Esquema XML do W3C

Muitos desenvolvedores na comunidade de XML acreditam que as DTDs não são suficientemente flexíveis para atender às necessidades de programação da atualidade. Por e-

xemplo, os programas não podem manipular as DTDs da mesma maneira que os documentos XML, porque as DTDs não são documentos XML em si mesmas. Essas e outras limitações têm levado ao desenvolvimento dos esquemas (BIGGS; EVANS, 2001).

Diferentemente da DTD, os esquemas não utilizam a gramática de EBNF. Em vez disso, utilizam a sintaxe de XML e são realmente documentos XML que os programas podem manipular como outros documentos XML. Como as DTDs, os esquemas exigem analisadores sintáticos de validação. No futuro próximo, os esquemas provavelmente substituirão as DTDs como o principal meio de descrever a estrutura de documento XML.

4.4. *Document Object Model* (DOM – Modelo de Objeto de Documento)

Embora um documento XML seja um arquivo de texto, recuperar dados do documento com técnicas tradicionais de acesso de arquivo seqüencial não é algo prático nem eficiente, especialmente para se adicionar e remover elementos dinamicamente. Quando um analisador sintático de DOM analisa sintaticamente um documento XML com sucesso, o analisador sintático cria uma estrutura de árvore na memória que contém os dados do documento. A Figura 19 mostra um arquivo XML em que se tem uma matéria marcada com XML (BIGGS; EVANS, 2001).

```
<artigo>
     <titulo>XML Simples</titulo>
     <data>21 de Junho de 2005</data>
     <autor>
          <nome>Cesar Augusto</nome>
          <sobrenome>Cusin</sobrenome>
     </autor>
     <resumo>XML é bem fácil.</resumo>
     <conteudo>XML é muito simples e fácil.
          Você deve lembrar-se que XML não é para
          exibir informações e sim para administrá-la.
     </conteudo>
</artigo>
```

Figura 19 - Matéria marcada com XML.

A Figura 20 mostra a estrutura de árvore para o arquivo código da Figura 19. Essa estrutura de árvore hierárquica é uma árvore DOM. Cada nome (por exemplo, **`artigo`**, **`titulo`**, **`data`**, etc.) representa um nó. O nó que contêm outros nós (chamados nós filhos) chama-se nó pai (por exemplo, **`autor`**). O nó pai pode ter muitos filhos, mas o nó filho pode ter apenas um nó pai. Os nós que são do mesmo nível (por exemplo, **`nome`** e **`sobrenome`**), chamam-se nós irmãos. Os nós descendentes de um nó incluem os deste nó, os filhos de seus filhos, e assim por diante. Os nós antepassados de um nó incluem o pai desse nó, o pai do seu pai, e assim por diante.

O DOM tem um único nó raiz, que contêm todos os outros nós do documento. Por exemplo, o nó raiz para a Figura 19 é **`artigo`**.

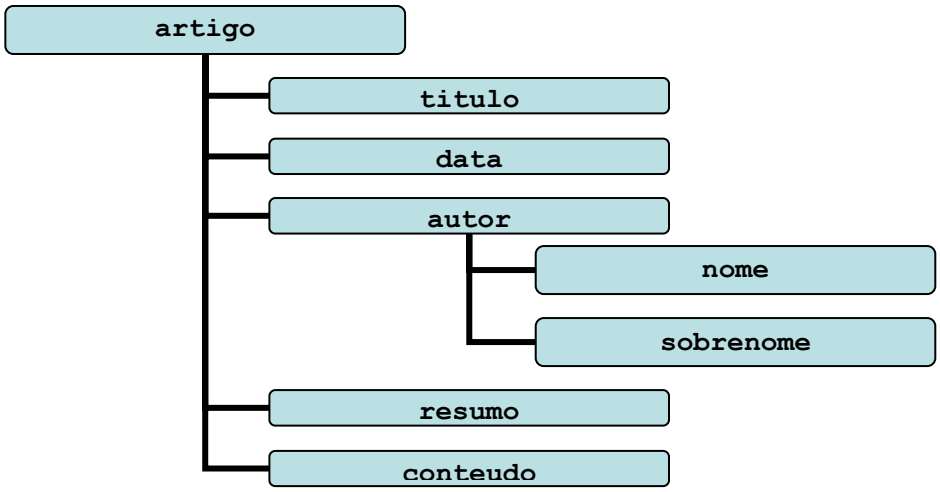


Figura 20 - Estrutura de árvore para a Figura 18.

Cada nó é um objeto que tem propriedades, métodos e eventos. As propriedades associadas a um nó incluem nomes, valores, nós filhos etc. Os métodos permitem criar, excluir e acrescentar nós, carregar documentos XML etc. O analisador sintático de XML expõe esses métodos como uma biblioteca de programação – chamada de API (*Application Programing Interface*).

4.5. Simple API for XML (SAX)

Os membros da lista de mala direta XML-DEV desenvolveram a SAX (*Simple API for XML*), que foi lançada em maio de 1998. A SAX é um método alternativo para analisar sintaticamente documentos XML que utiliza um modelo baseado em evento – os analisadores sintáticos baseados em SAX geram notificações chamadas de eventos enquanto o analisador examina o documento. Os *softwares* podem "ouvir" esses eventos para recuperar dados particulares do documento. Por exemplo, o programa que constrói uma lista de mala direta poderia ler informações como nomes e endereços de um documento XML que contém muito mais que apenas informações de endereço (por exemplo, aniversários, números de telefone, endereços

de correio eletrônico etc.). Esse programa poderia utilizar um analisador sintático SAX para analisar sintaticamente o documento e poderia "ouvir" somente os eventos que contivessem informações de nome e endereço. Se esse programa utilizasse um analisador sintático DOM, o analisador carregaria cada elementos e cada atributo na memória e o programa teria de percorrer a árvore DOM para localizar as informações relevantes de endereço (BIGGS; EVANS, 2001).

SAX e DOM fornecem APIs significamente diferentes para acessar informações de documentos XML; cada API tem vantagens e desvantagens. O DOM é um modelo baseado em árvore que armazena os dados do documento em uma memória hierárquica de nós. Os programas podem acessar dados rapidamente, porque todos os dados do documento estão na memória. O DOM também oferece funções para adicionar ou remover nós, o que permite modificar facilmente os documentos XML.

Os analisadores SAX invocam métodos ouvintes quando o analisador sintático encontra a marcação. Com esse modelo baseado em evento, o analisador sintático baseado em SAX não cria uma estrutura de árvore para armazenar os dados do documento XML – em vez disso, o analisador passa os dados do documento XML para o aplicativo quando localiza esses dados. Isso resulta em maior desempenho e menos *overhead* de memória do que com analisadores sintáticos baseados em DOM. Na realidade, muitos analisadores DOM utilizam os analisadores SAX "às escondidas" para recuperar os dados de um documento para construir a árvore DOM na memória. Muitos programadores acham mais fácil percorrer e manipular o documento XML com a estrutura de árvore DOM. Como resultado, os programas em geral utilizam analisadores SAX para ler os documentos XML que o programa não modificará. Os analisadores baseados em SAX estão disponíveis para uma variedade de linguagens de programação, como Java, Python, C++ etc.

4.6. JDOM

O Projeto JDOM foi desenvolvido em Junho de 2000 por Brett McLaughlin e Jason Hunter, pois o DOM e o SAX não eram suficientes (HUNTER, 2002).

JDOM é uma API Java para processar documentos XML. JDOM foi especificamente projetado para a linguagem Java ler e escrever (manipular) facilmente e eficientemente documentos XML. JDOM uma alternativa a DOM e SAX, embora interaja bem com eles.

Uma das características interessantes de JDOM é sua interoperabilidade com outras APIs. Usando JDOM não somente cria-se documento, mas interage também com SAX ou DOM. Esta flexibilidade permite a JDOM ser usado em um ambiente heterogêneo ou ser acrescentado a sistemas que já usam outro método que controlem XML. Outro uso de JDOM explora a habilidade para ler e manipular dados de XML que já existem. Ler um arquivo XML bem-formado é possível usando um das classes org.jdom.input.

Tem-se algumas definições para JDOM. A filosofia de JDOM é a de ser feito para programadores Java. Deve modificar um documento XML de forma fácil e eficiente, deve integrar com DOM e SAX, ser leve e rápido, deve resolver 80% (ou mais) de problemas de Java/XML com 20% (ou menos) de esforço. Permite ao Java ter acesso, manipular, e exibir dados XML em código Java (JDOM.ORG, 2005). JDOM é uma biblioteca *open source* para manipular dados XML (HUNTER, 2002). Esta API *open source* faz um documento XML ser manipulado facilmente por desenvolvedores Java. É uma ferramenta completa por criar, manipular, transformar e analisar gramaticalmente documentos XML (BIGGS; EVANS, 2001).

Atualmente não é possível executar a validação de um arquivo XML na memória em razão do desempenho, mas o Projeto JDOM já tem um voluntário trabalhando nisso.

Na Tabela 3 vê-se que o JDOM, em sua versão 1.0, inclui os seguintes Pacotes (*Packages*):

Tabela 3 - Pacotes (*Packages*) (API JDOM, 2005).

Pacote	Descrição
org.jdom	Classes para representar os componentes de um documento XML.
org.jdom .adapters	Classes para conectar com várias implementações DOM.
org.jdom.filter	Classes para programar nós filtro a um documento baseado em tipo, nome, valor, ou outros aspectos e para regras boleanas and/or/negate.
org.jdom.input	Classes para construir documentos JDOM de várias fontes.
org.jdom.output	Classes para produção de documentos JDOM a vários destinos.
org.jdom.transform	Classes para ajudar com transformações, baseado em classes JAXP TrAX.
org.jdom.xpath	Suporte para XPath dentro de JDOM.

4.7. JDOM versus DOM

A Figura 21 mostra como criar um simples documento em JDOM (HUNTER, 2002):

```
Document doc = new Document();
Element e = new Element("root");
e.setText("This is the root");
doc.addContent(e);
```

Figura 21 - Documento JDOM.

Já a Figura 22 mostra o mesmo documento feito por usuários experientes:

```
Document doc = new Document(
     new Element("root").setText("This is the root"));
```

Figura 22 - Documento JDOM para usuários experientes.

A mesma tarefa em JAXP/DOM (JAXP – *Java API for XML Processing*) requer um código mais complexo (Figura 23):

```
DocumentBuilderFactory factory =
     DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document doc = builder.newDocument();
Element root = doc.createElement("root");
Text text = doc.createText("This is the root");
root.appendChild(text);
doc.appendChild(root);
```

Figura 23 - Construindo com JAXP/DOM.

Nota-se na Figura 24 como construir um documento com XML e o mesmo documento usando JDOM com as Figuras, 25, 26 e 27.

```
<?xml version="1.0" encoding="UTF-8"?>
<car vin="123fhg5869705iop90">
     <!--Description of a car-->
     <make>Toyota</make>
     <model>Celica</model>
     <year>1997</year>
     <color>green</color>
     <license state="CA">1ABC234</license>
</car>
```

Figura 24 - Como construir um documento XML (BIGGS; EVANS, 2001).

Para construir o mesmo documento usando JDOM, cria-se o elemento **`root`** (raiz) e acrescenta ao documento (Figura 25):

```
Element carElement = new Element("car");
Document myDocument = new Document(carElement);
```

Figura 25 - Criando o documento.

Este código (Figura 25) cria um org.jdom.Element novo e faz dele o elemento raiz de **`myDocument`** de org.jdom.Document. (não se pode esquecer de importar org.jdom.*). Depois, adiciona-se o atributo **`vin`** (Figura 26):

```
carElement.addAttribute(new Attribute("vin",
     "123fhg5869705iop90"));
```

Figura 26 - Adicionando um atributo.

Na Figura 27 adiciona-se o elemento **`make`**:

```
Element make = new Element("make");
make.addContent("Toyota");
carElement.addContent(make);
```

Figura 27 - Elemento e conteúdo do elemento.

O elemento e o conteúdo também podem ser escritos da seguinte forma (Figura 28):

```
carElement.addContent(new Element("make").addContent("Toyota"));
```

Figura 28 - Adicionando elemento de uma forma concisa.

As duas formas (Figuras 27 e 28) realizam a mesma coisa. Diz-se que o primeiro exemplo (Figura 27) é mais legível, mas o segundo exemplo (Figura 28) fica mais claro se você estiver construindo muitos elementos.

Nota-se, que para o elemento **`licence`** (Figura 29) adicionou-se não só o conteúdo do elemento, como também um atributo, especificando o estado no qual a licença foi emitida.

```
carElement.addContent(new Element("model").addContent("Celica"));
carElement.addContent(new Element("year").addContent("1997"));
carElement.addContent(new Element("color").addContent("green"));
carElement.addContent(new Element("license")
     .addContent("1ABC234").addAttribute("state", "CA"));
```

Figura 29 - Adicionando os elementos restantes.

A Figura 30 mostra como adicionar um comentário.

```
carElement.addContent(new Comment("Description of a car"));
```

Figura 30 - Adicionando um comentário.

A manipulação do documento acontece em uma forma semelhante. Por exemplo, para obter uma referência ao elemento **`year`**, usa-se o método **`getChild`** de **`Element`**, como mostra a Figura 31:

```
Element yearElement = carElement.getChild("year");
```

Figura 31 - Acessando elementos filho.

Esta declaração na verdade devolverá o primeiro elemento filho com o nome **`year`**. Se não houver nenhum elemento **`year`**, então a chamada retornará nula.

A chamada da Figura 32 removerá somente o elemento **`year`**; o resto do documento permanece inalterado.

```
boolean removed = carElement.removeChild("year");
```

Figura 32 - Removendo elementos filho.

Para finalizar usa-se a classe **`XMLOutputter`** de JDOM (Figura 33):

```
try {
     XMLOutputter outputter = new XMLOutputter(" ", true);
     outputter.output(myDocument, System.out);
} catch (java.io.IOException e) {
     e.printStackTrace();
}
```

Figura 33 - Transformando JDOM em texto XML.

`XMLOutputter` tem algumas opções de formatação. Na Figura 33 especifica-se que os elementos filho recuem dois espaços do elemento de pai, e que sejam inseridas linhas novas entre elementos. **`XMLOutputter`** pode exibir ou criar um arquivo. Para criar um arquivo muda-se a linha de produção como mostra a Figura 34:

```
FileWriter writer = new FileWriter("/some/directory/myFile.xml");
outputter.output(myDocument, writer);
writer.close()
```

Figura 34 - Criando um arquivo em XML.

Com esses exemplos mostrou-se a vantagem de se usar JDOM ao invés de DOM.

5. INTERAÇÃO

Neste Capítulo mostra-se os tipos de Interação do Usuário com o Sistema Operacional, Interação do Usuário com a Internet e a Interação do Usuário com o Protótipo Proposto.

Para tanto, se faz necessário uma demonstração de como é feito a geração de Ambientes Virtuais por computador. Para isso, mostra-se uma introdução sobre uma Estrutura Detalhada de um Sistema de RV para posteriormente entender as demais Interações e suas respectivas funções no Protótipo Proposto (Figura 35).

O componente de Rede de um Sistema de RV não é necessário e, sim, possível de ser implementado.

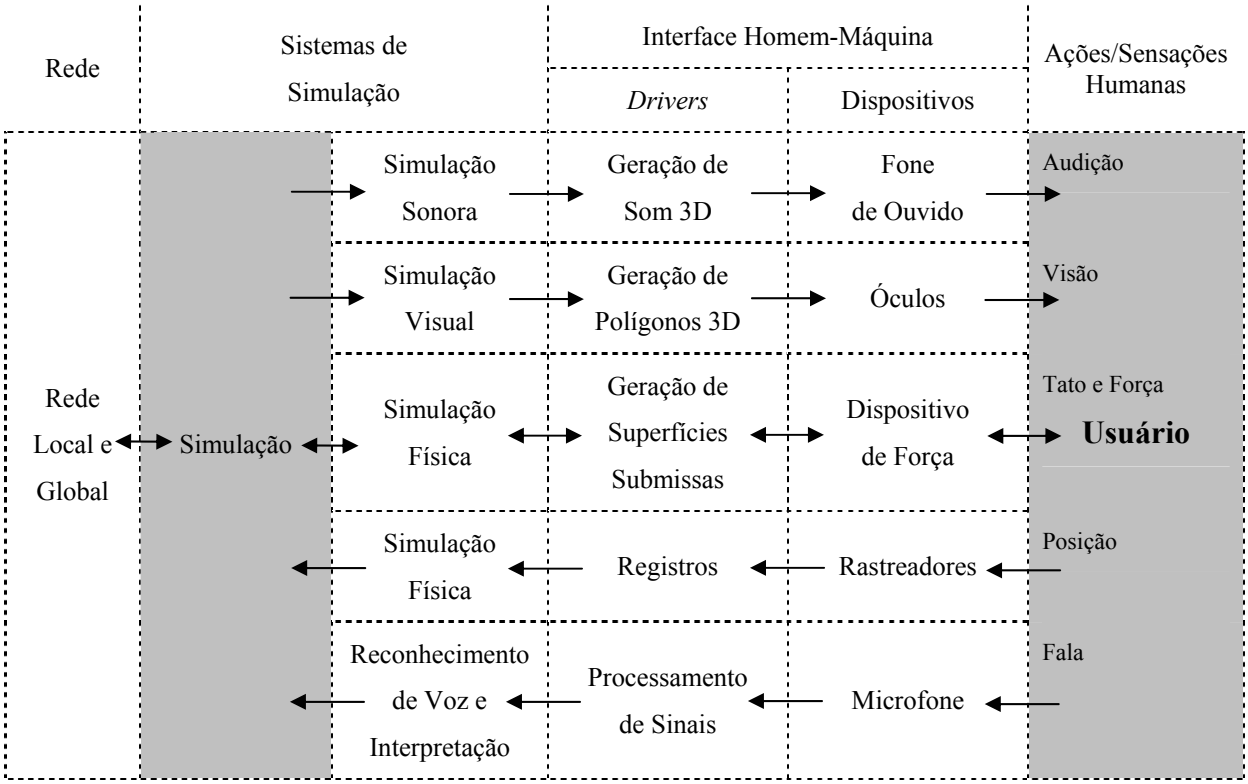


Figura 35 - Estrutura Detalhada de um Sistema de RV.

5.1. Interação do Usuário com o Sistema Operacional

Um Sistema Operacional (SO) é um programa, ou conjunto de programas, inter-relacionados cuja finalidade é agir como intermediário entre o usuário e o *hardware*.

Tanenbaum (1999) diz que "...um sistema operacional é um programa que, do ponto de vista do programador, adiciona um conjunto de novas instruções e de funcionalidades..." e que "...em geral o sistema operacional é implementado por *software*, mas não há motivo que impeça de ele ser implementado em *hardware*..." (Figura 36).

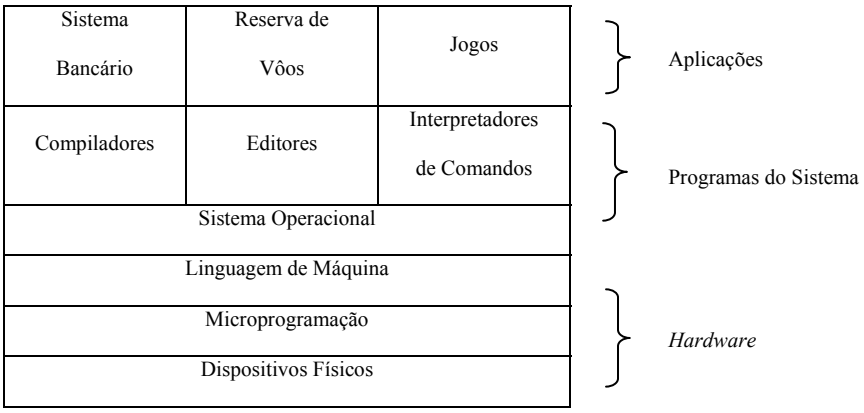


Figura 36 - Um sistema de computador consiste em *hardware*, programas do sistema e aplicações.

Silberschatz *et al.* (2000) afirma que "...um sistema operacional é um programa que atua como intermediário entre o usuário e o *hardware* de um computador. O propósito de um sistema operacional é fornecer um ambiente no qual o usuário possa executar programas. O principal objetivo de um sistema operacional é portanto tornar o uso do sistema de computação conveniente. Uma meta secundária é usar o *hardware* do computador de forma eficiente...".

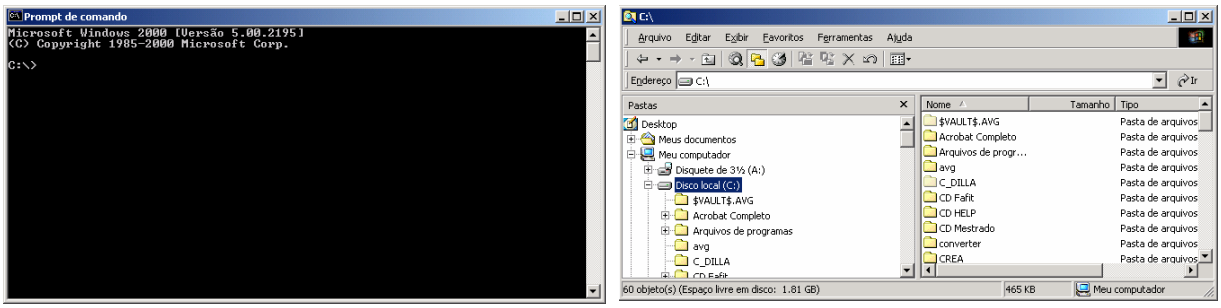
Já Shay (1996) argumenta que "...um sistema operacional é um programa que permite às pessoas usar o *hardware* do computador (CPU, memória e armazenamento secundário)...".

Para Torres (1996), "...o sistema operacional é um tradutor entre o microprocessador e o mundo externo a ele...".

O Usuário interage com o SO de maneira direta, por meio de comandos pertencentes a uma linguagem de comunicação especial, chamada "linguagem de comando". Esses Comandos podem ser dados de duas formas (Figura 37):

* Interface em Modo Texto (Linha de Comando);

* Interface Gráfica.



Interface em Modo Texto (Linha de Comando) ***Interface Gráfica***

Figura 37 - Exemplos de Interfaces de Comandos.

5.2. Interação do Usuário com a Internet

A Internet é uma Rede Pública de alcance mundial, baseada na arquitetura de comunicação TCP/IP (*Transmission Control Protocol/Internet Protocol*), que oferece uma série de serviços de comunicação padronizados e difundidos publicamente. A Figura 38 mostra uma possível estrutura para a Interação do Usuário com a Internet.

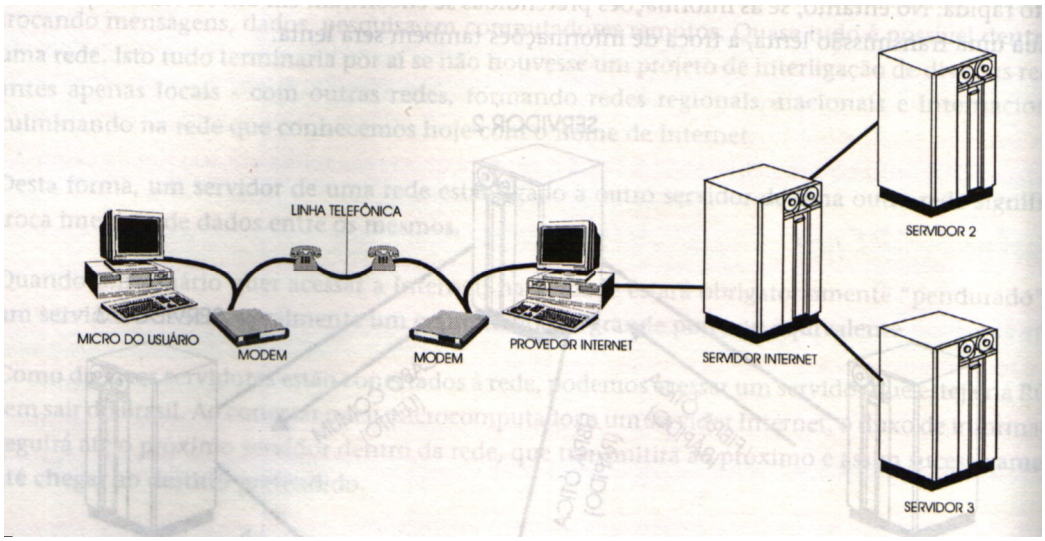
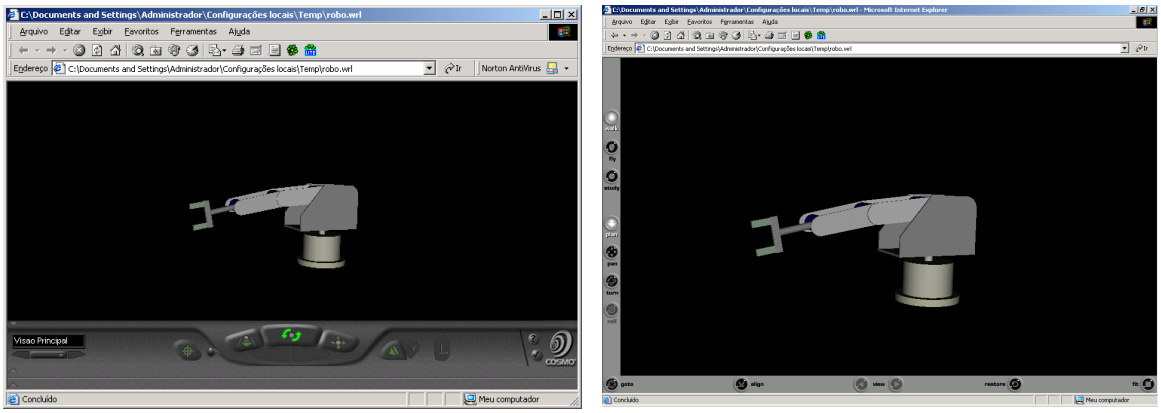


Figura 38 - Infra-estrutura de Comunicação (TORRES, 1996).

Para navegar na Internet usa-se uma variedade de aplicações, dentre elas, a mais importante é o *Browser* (navegador). Os *Browsers* mais conhecidos são: o *Internet Explorer* e o *Netscape Navigator*. Todos os *Browsers* mostram primariamente textos marcados em uma determinada linguagem (HTML ou XHML) e imagens embutidas no mesmo documento.

Quando se clica em um *link* de uma página, o *Browser* verifica o tipo do arquivo; e se contém texto, HTML, XHTML ou imagens. O *Browser* se encarrega de exibir o conteúdo. Quando o *Browser* verifica que são outros tipos de informações, tais como, sons, filmes, e mundos 3D, o ele passa a informação para o respectivo *plug-in* (programas que executam aplicações que são informações diferentes de HTML ou XHTML) que será executado na própria janela do *Browser* (AMES *et al.*, 1997).

Os *plug-ins* para a visualização de Mundos Virtuais mais conhecidos são: o *Cosmo-Player* e o *Cortona* (Figura 39).



CosmoPlayer ***Cortona***

Figura 39 - Exemplo de *Plug-in* no *Browser Internet Explorer*.

5.3. Interação do Usuário com o Protótipo Proposto

O Protótipo Proposto; Interpretador de Movimentos para um Sistema Multimodal (IMSM); em sua definição de arquitetura, exibe os recursos utilizados, tanto de *software* quanto de *hardware*, a fim de analisar a viabilidade e a qualidade do mesmo na alteração do Mundo Virtual usando diferentes dispositivos de entrada (convencionais ou não).

Para o desenvolvimento do IMSM, utilizou-se os seguintes *softwares* (Tabela 4):

Tabela 4 - *Softwares* utilizados no IMSM.

Programa	Qualificação	Desenvolvedor	Objetivo
Java 3D 1.3.1 (Open GL) SDK	API Gráfica	*Sun*	Renderização de Objetos 3D
XML	Linguagem de Marcação	*W3C*	Troca de dados
JDOM 1.0	API Java	*Sun*	Processar documentos XML
Java 2 SDK, SE v1.4.2_04	Gerador de Aplicações	*Sun*	Linguagem baseada em linguagem C e C++
Windows XP	Sistema Operacional	*Microsoft*	Sistema Operacional utilizado para desenvolvimento do protótipo

Com relação aos recursos de *hardware*, é o que se segue (Tabela 5):

Tabela 5 - *Hardware* utilizado no IMSM.

Equipamento	Descrição
Computador PC (*Personal Computer*)	Processador 2.0 GHZ, 256 MB RAM, HD 40 GB, 4 portas USB
Monitor	Monitor 17 polegadas, Marca LG, Modelo Flatron 775FT
Teclado	Teclado 110 teclas, Marca MTek, Modelo K279
Mouse	*Mouse* Marca Dr. Hank, Modedo MO-D35P-DD
Joystick Comand Fire Vibration	*Joy Pad Comand Fire Vibration*, 8 botões, USB, Marca CLONE

As justificativas em relação a escolha dos Softwares seguem-se na Seção 6.2.

A arquitetura geral do IMSM está representada na Figura 40:

Figura 40 - Arquitetura Geral do IMSM.

A Figura 41 mostra a interação do usuário com protótipo proposto.

Segundo o protótipo proposto, o usuário, independente do periférico de entrada (convencional ou não), faz alterações no Mundo Virtual, passando, para tanto, pelo Sistema Ope-

racional, pelo tradutor e pela gramática com suas regras e, aí sim, efetuam alterações no Mundo Virtual (Figura 42).

São diversas as soluções para interação e é apresentada em anexo uma solução baseada em JavaScript (ANEXO A).

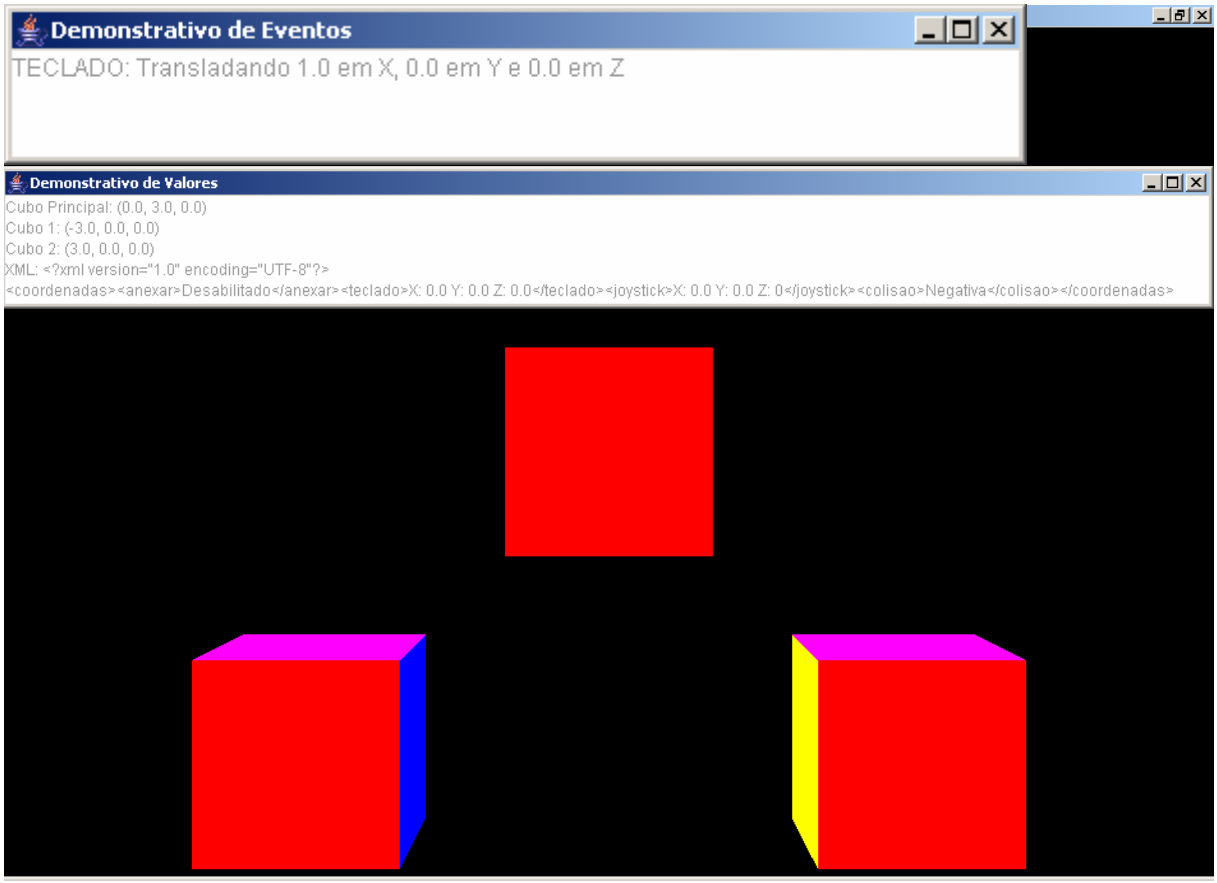


Figura 41 - Interação do Usuário com o IMSM.

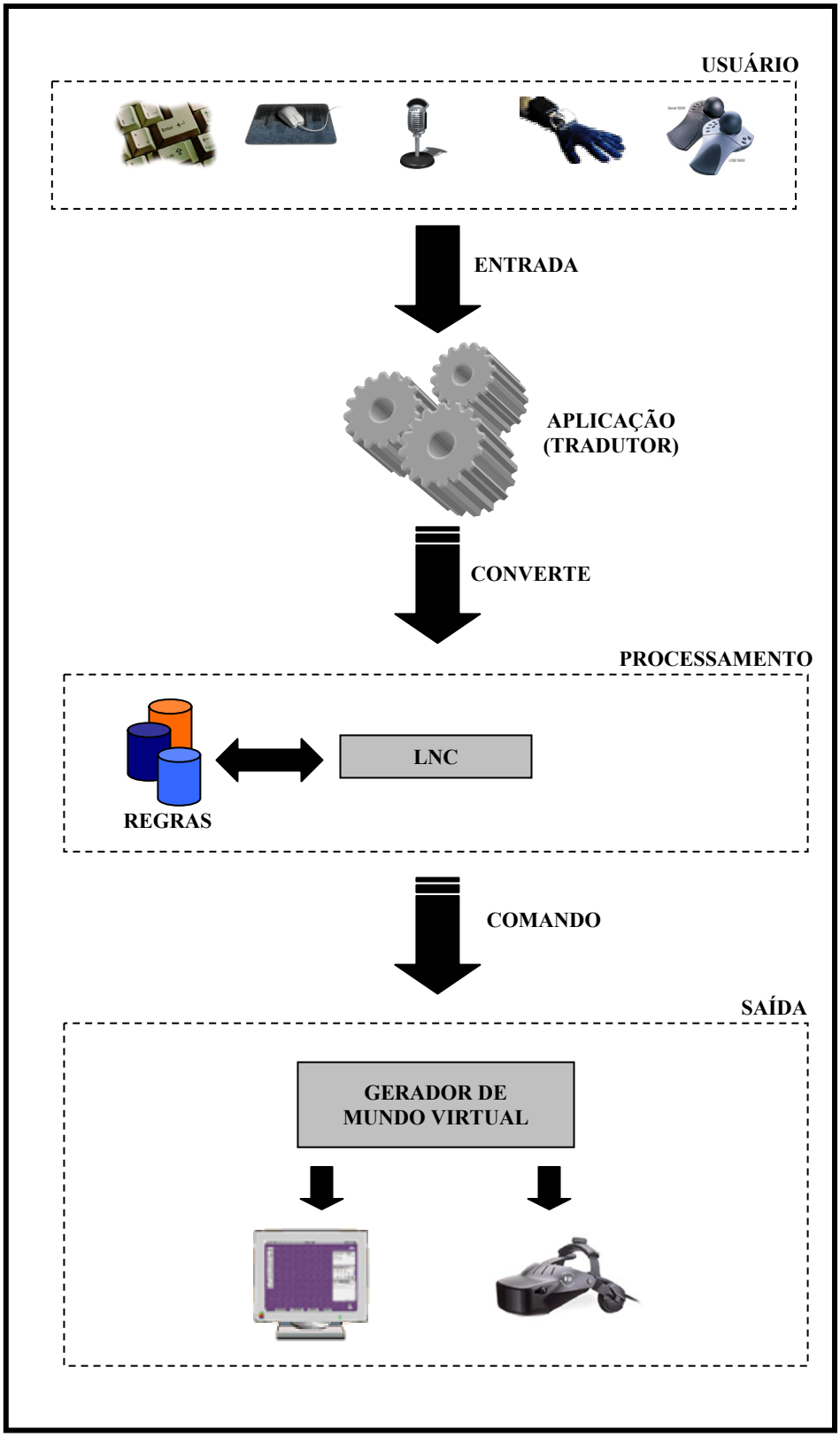


Figura 42 - Interação do Usuário com o Aplicativo Proposto.

6. DESCRIÇÃO DO PROTÓTIPO

6.1. Proposta

Neste Capítulo é apresentada uma ferramenta baseada em sistema de RV com a unificação dos meios de entrada, independente de ser ou não um dispositivo convencional, para uma única linguagem que efetue alterações no Mundo Virtual.

Devido a necessidade de conversão da linguagem de todos dispositivos para uma linguagem comum, pode-se classificar como um "Tradutor". Ele interpreta os mais variados tipos de entrada, converte em uma linguagem única (LNC) e efetua as devidas mudanças no Mundo Virtual.

6.2. Protótipo

O IMSM é um Sistema de Realidade Virtual Não-Imersivo. O Java é o gerador da aplicação devido à sua grande portabilidade. Para a geração do mundo e dos objetos contidos nele, usa-se a API Java 3D.

No caso do Java 3D, faz-se necessário um comentário à parte com relação ao tipo de universo utilizado. O *SimpleUniverse*; universo mais utilizado; permite aplicações básicas, pois, cria um ambiente mínimo necessário para um programa Java 3D. Além disso, cria os objetos necessários na parte *View* do grafo de cena. Especificamente, esta classe cria um *Lo-*

cale, um único *ViewingPlataform* e um objeto *Viewer*. Porém, para aplicações mais sofisticadas, precisa-se de mais controle para adquirir funcionalidade extra.

Sabendo-se disso, opta-se então pela classe *VirtualUniverse*, que permite mais objetos *Locale*, cada um dos quais com suas coordenadas de alta resolução dentro do universo virtual, o que permite a entrada de dispositivos não-convencionais, possuindo métodos para enumerar e, se necessário, remover um *Locale* no universo virtual.

Porém, sabendo das especificações dos universos acima explicados, para o IMSM, optou-se pelo *ConfiguredUniverse*, pois, ele cria todos os objetos *View* necessários ao grafo de cena. Especificamente, cria um *Locale*, um ou mais *ViewingPlatforms*, e pelo menos um objeto *Viewer* além suportar dispositivos não-convencionais.

O *ConfiguredUniverse* pode montar um ambiente baseado nos conteúdos de um arquivo texto de configuração, permitindo assim, uma aplicação rodar sem mudança por uma larga gama de configurações. Um arquivo de configuração pode criar o *InputDevice*, Sensores, *ViewPlatformBehavior* como também *Viewers* e *ViewingPlatforms*. Pelo menos um *Viewer* deve ser provido pela configuração. Se um *ViewingPlatform* não for provido, um padrão será criado e o *Viewer* será anexado a ele.

Quando cria-se um arquivo de configuração, informa-se o caminho do arquivo a um construtor do *ConfiguredUniverse*. Se um arquivo de configuração não for criado, o *ConfiguredUniverse* cria um ambiente *Viewing* padrão da mesmo forma como o *SimpleUniverse*. Todos os construtores disponíveis para o *SimpleUniverse* também estão disponíveis para o *ConfiguredUniverse*.

Verifica-se então, que a escolha pelo *ConfiguredUniverse* se mostrou a melhor opção, pois, permite mais opções que o *SimpleUniverse* e ainda se mostra mais simples de usar que o *VirtualUniverse*.

Faz-se o uso da linguagem XML que a exemplo do Java possui grande portabilidade, e por isso é muito útil pela necessidade de padronizar os dados de entrada uma vez que as

informações vêm de dispositivos diferentes. Para fazer a comunicação (manipulação) do XML com Java usa-se o JDOM.

Para o IMSM usa-se a técnica de Metáfora Interativa Individual, pois, faz-se o mapeamento entre as ações do usuário e seus efeitos sobre o objeto. Faz-se, também, o uso de uma LNC com Sintaxe e Semântica própria.

A grande vantagem de se usar uma LNC está no fato de que todas as sentenças usadas nesta linguagem estão corretas na LN da qual ela faz parte.

6.3. Implementação

O IMSM visa o desenvolvimento de uma ferramenta que trata da Interação Humano-Computador. Trabalha-se então com dispositivos de entrada (convencionais ou não); sabe-se que estes dispositivos enviam sinais diferentes ao computador. O aplicativo proposto desenvolvido em linguagem Java interpreta a entrada de dados (Figura 43).

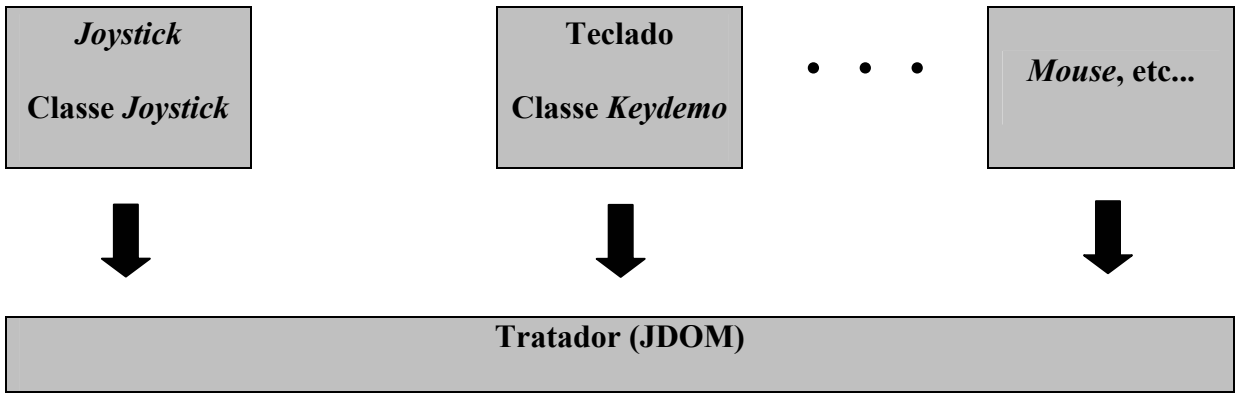


Figura 43 - Forma de trabalhar com os dispositivos de entrada no IMSM.

Para capturar os movimentos do *joystick*, a Aplicação principal instancia a classe *Joystick* que captura os movimentos do dispositivo (Figura 44).

```
public float getXPos() {
      return getXPos(joyID);
}

public float getYPos() {
      return getYPos(joyID);
}

public float getZPos() {
      return getZPos(joyID);
}
```

Figura 44 - Trecho da classe *Joystick*.

O *joystick* usado no IMSM foi um modelo 2D (Figura 45) e foi necessário programar seus botões (Figura 46) para que eles agissem de forma apropriada no IMSM.

Figura 45 - *Joystick* utilizado no IMSM (CLONE, 2005).

O *joystick* utilizado foi o *Joy Pad Comand Fire Vibration 8 Botões USB* da Marca Clone. Suas características são (CLONE, 2005):

* Compatível com IBM PC/Win 98;

* 8 Botões de disparo, 4 eixos;

* Botão de vibração;

* Botão Turbo;

* Controle direcional digital e

* Conector USB

```
if (Java3DBehavior.joy.getButtons()==1){
      _pos.z = _pos.z + 1;
}
else if(Java3DBehavior.joy.getButtons()==8){
      _pos.z = _pos.z - 1;
}
```

Figura 46 - Trecho da programação dos botões do *joystick*.

A forma de usar o *joystick* no IMSM é mostrada na Figura 47:

Figura 47 - Forma de usar o *Joystick* no IMSM.

A Figura 48 mostra o uso do *joystick* no IMSM.

Figura 48 - Translação com o *Joystick* no IMSM.

No caso do Teclado, a captura é feita pela classe *Keydemo* (Figura 49). Para o Teclado foi criado variáveis para transformar os códigos do mesmo em valores numéricos que tivessem significado na hora de alterar as coordenadas do IMSM.

A forma de usar o Teclado no IMSM é mostrada na Figura 50 e o exemplo do uso do mesmo no IMSM é mostrado na Figura 51.

```
if ( event.getKeyCode() == 38 ) { //Seta para cima (Up)
      tx = 0; ty = -1; tz = 0;
}
else if ( event.getKeyCode() == 40 ) { //Seta para baixo (Down)
      tx = 0; ty = 1; tz = 0;
}
```

Figura 49 - Trecho da classe *KeyDemo* (para o teclado).

← →	↑ ↓	Home	End	A
Movimentam em X	Movimentam em Y	Zoom In	Zoom Out	(Des)Anexa

Figura 50 - Forma de usar o Teclado no IMSM.

Figura 51 - Exemplo do Teclado no IMSM.

Com o problema da entrada de dados resolvido com o uso do Java, faz-se necessário definir a Gramática, bem como suas regras de produção.

Os termos para esta Gramática são compostos, como mostra a Tabela 6, por palavras que farão parte da LNC, apresentando sua sintaxe (estrutura), semântica (significado) e sua função no IMSM.

Tabela 6 - Definição da Gramática da Aplicação.

Palavra	Sintaxe	Semântica	Função na Aplicação
Transladar	Verbo no Infinitivo	Ato ou efeito de mudar algo de lugar	Transladar o objeto no Mundo Virtual
Anexar	Verbo no Infinitivo	Juntar (algo) a uma coisa considerada como principal	Anexar o Objeto Principal a outro Objeto no Mundo Virtual
Desanexar	Verbo no Infinitivo	Separar (aquilo que estava anexado); desligar, desmembrar	Desanexar o Objeto Principal do Objeto que está Anexo a ele
X	Letra do Alfabeto	Variável	Variável que leva o valor para a Posição X do Objeto Virtual
Y	Letra do Alfabeto	Variável	Variável que leva o valor para a Posição Y do Objeto Virtual
Z	Letra do Alfabeto	Variável	Variável que leva o valor para a Posição Z do Objeto Virtual

Definida a Gramática do IMSM, resta agora converter os valores das entradas do *joystick* e do teclado para XML, já que a Gramática da Aplicação é feita nesta linguagem. Este trabalho é realizado como mostra a Figura 52, gerando uma estrutura de árvore na memória (Figura 53).

```
Element coord = new Element("coordenadas");
Document doc = new Document(coord);
...
coord.addContent(new Element("anexar").addContent("Habilitado"));
...
coord.addContent(new Element("teclado")
   .addContent("X: " + KeyDemo.tx)
   .addContent(" Y: " + KeyDemo.ty)
   .addContent(" Z: " + KeyDemo.tz));
```

Figura 52 - Trecho da conversão dos valores de entrada para XML.

Figura 53 - Dados convertidos em XML em uma estrutura de árvore sendo mostrado na Janela de Demonstrativo de Valores do IMSM.

Para o Objeto Principal do IMSM anexar outro Objeto, é necessário haver a colisão entre eles. Porém, isso é inútil se o recurso de Anexar não está ativado. Por isso, é necessário que o IMSM faça essa verificação (Figura 54). O trecho que trata da colisão é o da Figura 55. Havendo a colisão, e o IMSM retornando positivamente a permissão para Anexar (Figura 56), os objetos, então anexados, podem ser movimentados juntos.

```
if (Java3DBehavior.anexar==1 && Java3DBehavior.colidir==1) {
...
}
```

Figura 54 - Trecho do teste de Anexação e Colisão.

```
CollisionDetector cd = new CollisionDetector(cubo2);
BoundingSphere bounds1 =
    new BoundingSphere(new Point3d(0.0,0.0,0.0), 100.0);
cd.setSchedulingBounds(bounds1);
bg1.addChild(cd);
```

Figura 55 - Trecho da colisão.

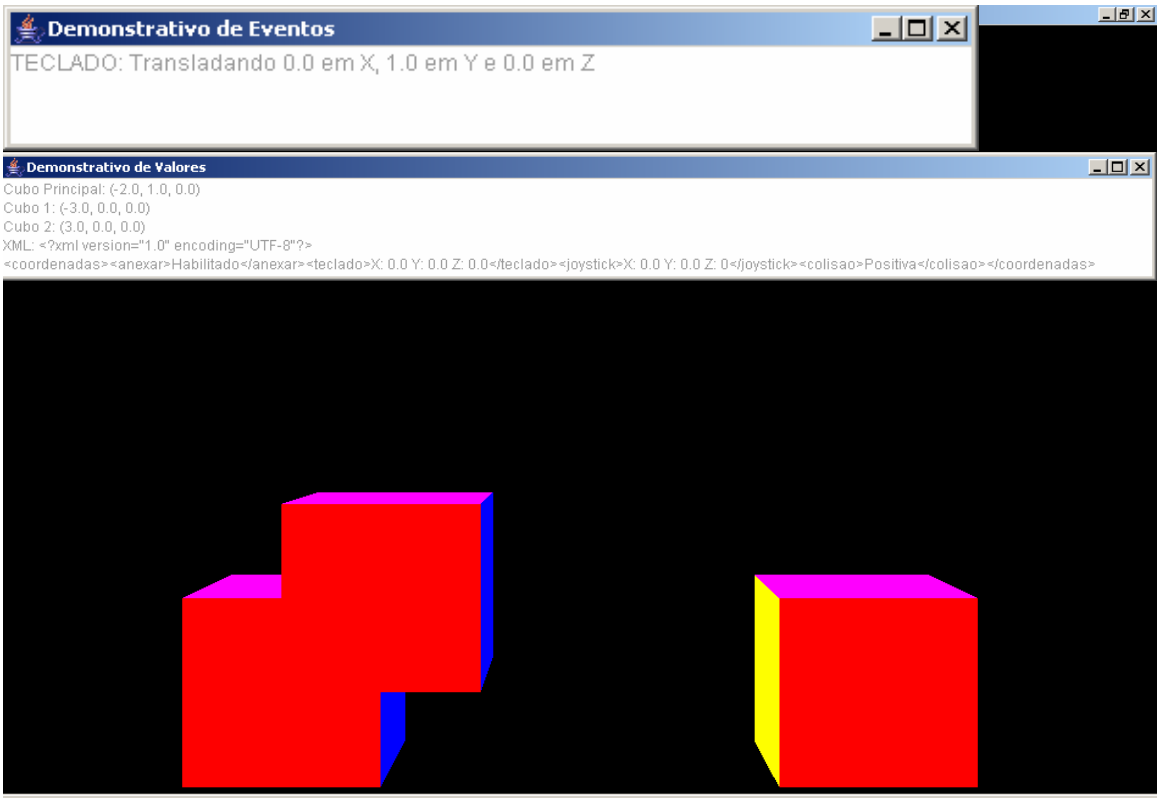


Figura 56 - Exemplo da Colisão e com Anexo ativado no IMSM.

6.4. Resultados

Os resultados obtidos foram muito satisfatórios, já que o objetivo é a unificação dos dispositivos de entrada em uma única linguagem para alterar o Mundo Virtual.

Os objetos responderam com precisão e perfeição às ordens enviadas simultaneamente pelos diferentes dispositivos de entrada (Figura 57). A conversão dos valores de entrada (embora o Java seja interpretado) não toma tempo da Aplicação, que responde em tempo real.

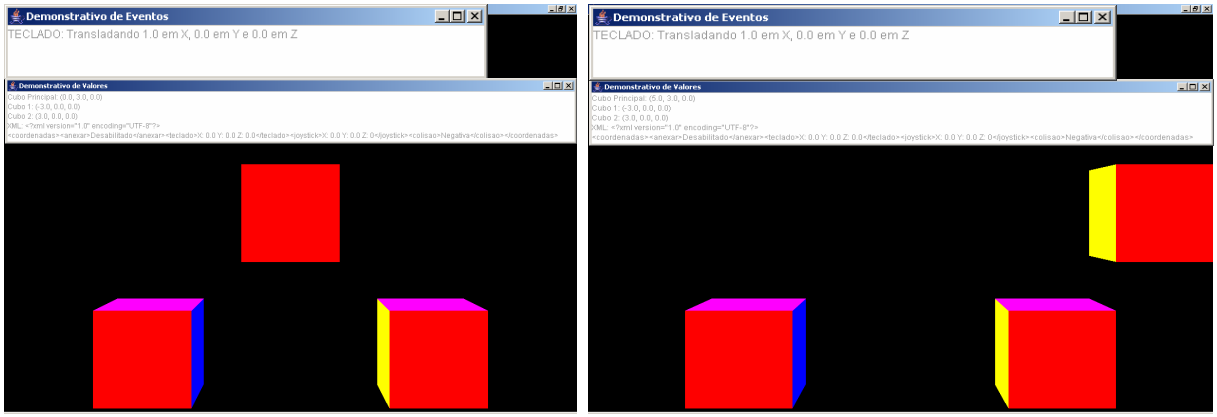


Figura 57 - Cubo Principal sendo movimentado no Mundo Virtual.

A conversão dos valores de entrada para a LNC é também eficiente; já que o JDOM trabalha e faz a conversão na memória; o que permite velocidade igual a do processador.

O resultado da visualização também é excelente, respondendo otimamente, o que é uma exigência das aplicações de RV (Figura 58).

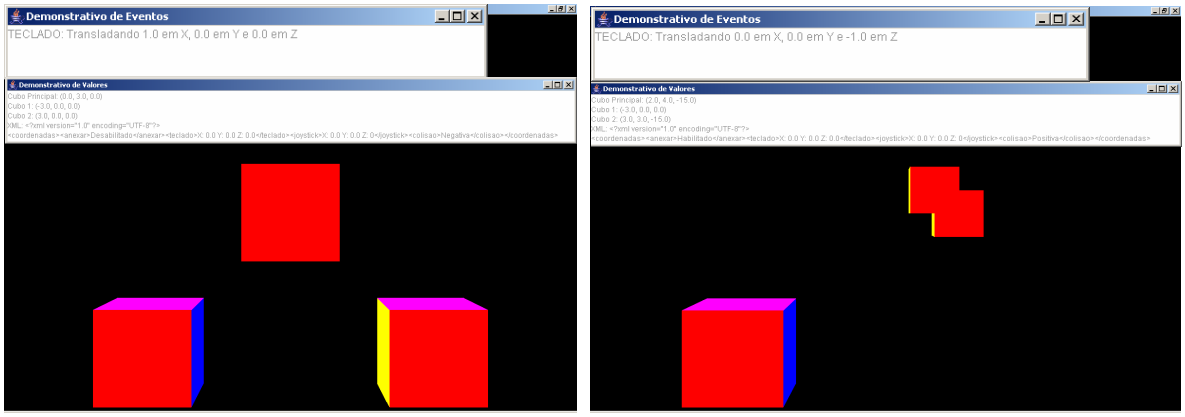


Figura 58 - Visualização do Cubo Principal Anexado ao Cubo 2 sendo movimentado no Mundo Virtual.

7. CONCLUSÃO

7.1. Considerações Finais

Este trabalho apresentou uma forma de entrada de dados feita pelo Usuário, independente de dispositivo. O aplicativo proposto interpreta a entrada de dados, analisa através de uma gramática própria e altera o Mundo Virtual.

O Aplicativo chamado IMSM unifica os meios de entrada, independente de ser ou não um dispositivo convencional, para uma única linguagem que efetue alterações no Mundo Virtual, trabalhando como um "Tradutor". Ele interpreta os mais variados tipos de entrada, converte em uma Gramática própria (LNC) e efetua as devidas mudanças no Mundo Virtual, gerando maior nível de interação e envolvimento.

Como contribuição, houve uma melhora na Interação Humano-Computador, simplificando ou apoiando as tarefas do Usuário através da técnica interativa de manipulação usando um sistema Multimodal.

O uso da Linguagem Natural Controlada se mostrou uma boa escolha, contribuindo com portabilidade e unificação de comandos. A linguagem XML é um caso a parte quando o assunto são dados portáveis.

O JDOM colaborou na união do XML com o Java, substituindo a contento as outras opções disponíveis (DOM e SAX), mas há ainda tarefas a serem feitas, como por exemplo, a possibilidade do funcionamento do analisador sintático quando se está trabalhando na memória.

7.2. Trabalhos Futuros

A título de trabalhos futuros, disponibilizar novos dispositivos de entrada, interação com som, aumentar a gramática do IMSM permitindo mais movimentos, trabalhar efetivamente com arquivos de dados; gravando e lendo arquivos, passando analisadores sintáticos, montando uma biblioteca de valores e movimentos para usos nas mais variadas aplicações. Aí nota-se a grande importância da portabilidade de XML.

Pode-se criar uma classe à parte, específica, para a transformação de valores dos sinais enviados pelos dispositivos de entrada e sua posterior conversão, gerando assim, uma biblioteca geral de entrada, funcionando como um "Tradutor" genérico para os mais variados tipos de entrada.

7.3. Conclusões

Os resultados obtidos são muito animadores, pois, o IMSM mostra uma possibilidade de trabalho com RV com custo operacional mínimo. As futuras aplicações de RV não ficarão mais presas a um único dispositivo (convencional ou não), não se prenderão a *Drivers* dos dispositivos, os portadores de deficiência podem optar por qual meio de entrada vão enviar os comando ao sistema, várias pessoas podem interagir ao mesmo tempo, utilizando meios de entrada diferentes na mesma aplicação.

REFERÊNCIAS

ABS-TECH. **Absolut Technologies: Soluções em Realidade Virtual**. 2005.
Disponível em <http://www.abs-tech.com>. Acesso em 07 de Setembro de 2005, 09:38.

AHO, A. V.; SETHI, R.; ULLMAN, J. D. **Compiladores – Princípios, Técnicas e Ferramentas**. LTC. Rio de Janeiro – RJ, 1995.

AMES, A. L.; NADEAU, D. R.; MORELAND, J. L. **VRML 2.0 Sourcebook**. Second Edition. John Wiley & Sons, Inc. EUA, 1997.

API JDOM. **JDOM v1.0 API Specification**. 2005.
Disponível em <http://www.jdom.org/docs/apidocs/>. Acesso em 21 de Julho de 2005, 13:03.

BIGGS, W; EVANS, H. **Simplify XML programming with JDOM**. Ano 2001. Disponível em <http://www-128.ibm.com/developerworks/java/library/j-jdom/>. Acesso em 21 de Julho de 2005, 13:15.

BORBA, F. da S. **Pequeno Vocabulário de Lingüística Moderna**. Editora Nacional e Editora da USP, Iniciação Científica, Volume 31, 1971.

BREGA, J. R. F.; SEMENTILLE, A. C.; RODELLO, I. A. **Uma Interface de Reconhecimento de Voz para Movimentação de Agentes e Avatares Humanóides em Ambientes Virtuais**. Proccedings SVR 2003 – VI Symposium on Virtual Reality. 15-18 October. Ribeirão Preto, SP – Brasil.

CHEYER, A.; JULIA, L. **Designing, Developing & Evaluating Multimodal Applications**. CHI'99 (WS Pen/Voice Interfaces) : Pittsburgh, Pennsylvania, USA, May 15-20, 1999.
Disponível em <citeseer.nj.nec.com/69694.html>. Acesso em 15 de Fevereiro de 2004, 20:05.

CHEYER, A.; JULIA, L.; MARTIN, J. C. **A Unified Framework for Constructing Multimodal Experiments and Applications**. In Proc. of Cooperative Multimodal Communication: Second International Conference, CMC'98, pp. 63-69, Tilburg, The Netherlands, January 1998. Published in the Journal "Lecture Notes in Computer Science", vol. 2155, pág. 63-69, ano 2001.
Disponível em <citeseer.nj.nec.com/cheyer98unified.html>. Acesso em 15 de Fevereiro de 2004, 21:27.

CLONE. **Joy Pad Comand Fire Vibration 8 Botões USB**. 2005.
Disponível em
<http://www.clone.com.br/produtos_descricoes1024.asp?cod_produto=06080>. Acesso em 07 de Setembro de 2005, 14:52.

DA SILVA, S. R. P.; PINHEIRO, J. M. **O uso de Linguagem Natural Controlada para Representação de Conhecimento por Usuários Finais**. 1º Workshop em Tecnologia da Informação e da Linguagem Humana - TIL 2003. Evento Integrante do 16th Brazilian Symposium on Computer Graphics and Image Processing - SIBGRAPI 2003. ICMC-USP. São Carlos, 12 de Outubro de 2003.
Disponível em <http://www.nilc.icmc.usp.br/til2003/oral/daSilva_Melchiori_paper7.pdf>. Acesso em 20 de Fevereiro de 2004, 16:53.

DEITEL, H. M.; DEITEL, P. J. e NETO, T. R. **Internet & World Wide Web, como programar**. - 2ª Edição - Bookman. Porto Alegre, 2003.

DIX, A. J.; FINLAY, J. E.; ABOWD, G. D.; BEALE, R. **Human-Computer Interaction**. Second Edition. Prentice Hall, 1998.

EMEDIAWIRE. 2005.
Disponível em
<http://www.emediawire.com/prfiles/2004/08/26/152953/morph_mobile_man1205.jpg>. Acesso em 06 de Setembro de 2005, 13:54.

FERREIRA, A. B. H. **Novo Aurélio Século XXI: o Dicionário da Língua Portuguesa**. Nova Fronteira, 1999.

FRERY, A. C.; KELNER, J.; MOREIRA, J.; TEICHRIEB, V. **Satisfaction Through Empathy and Orientation in 3D Worlds**. Journal "CyberPsychology & Behavior", vol. 5, nº 5, pág. 451-459, ano 2002.
Disponível em <citeseer.nj.nec.com/frery02satisfaction.html>. Acesso em 15 de Fevereiro de 2004, 17:17.

FUCHS, N. E.; SCHWITTER, R. **Specifying Logic Programs in Controlled Natural Language**. CLNLP 95. Workshop on Computational Logic for Natural Language Processing, Edinburgh, 1995.
Disponível em <http://citeseer.nj.nec.com/167728.html>. Acesso em 22 de Fereveiro de 2004, 15:07.

FUCHS, N. E.; SCHWITTER, R. **Attempto Controlled English (ACE)**. in Proceedings of CLAW 96. The First International Workshop on Controlled Language Applications, Katholieke Universiteit Leuven, Belgium, (1996).
Disponível em <http://citeseer.nj.nec.com/312.html>. Acesso em 22 de Fevereiro de 2004, 14:13.

FURGERI, S. **Ensino didático da linguagem XML**. Érica. São Paulo, 2001.

GRIES, D. **Construccion de Compiladores**. Paraninfo. Magallanes, Madrid, 1975.

HANCOCK, D. **Viewpoint: Virtual Reality in Search of Middle Ground**, IEEE Spectrum, 32(1):68, January, 1995.

HUNTER, J. **JDOM Makes XML Easy**. JavaOne – Sun´s 2002 Worldwide Java Developer Conference. Ano 2002.
Disponível em <http://www.servlets.com/speaking/jdom-javaone.pdf>. Acesso em 21 de Julho de 2005, 13:13.

JDOM.ORG. 2005.
Disponível em <http://www.jdom.org>. Acesso em 21 de Julho de 2005, 13:01.

KEHLER, A.; MARTIN, J. C.; CHEYER, A.; JULIA, L.; HOBBS, J. R.; BEAR, J. **On Representing Salience and Reference in Multimodal Human-Computer Interaction**. American Association for Artificial Intelligence - AAAI'98 (Representations for Multi-Modal Human-Computer Interaction) : Madison (USA), pp 33-39.
Disponível em <citeseer.nj.nec.com/kehler98representing.html>. Acesso em 15 de Fevereiro de 2004, 20:34.

KELNER, J.; ORGAMBIDE, A. C. F.; TEICHRIEB, V. **Conceptual Navigation in Virtual Reality Applications ConVIRA**. Partners Federal University. Proc. 2nd Brazilian Workshop on Virtual Reality - WRV'99, Marilia, SP, Brazil, November, 1999.
Disponível em <citeseer.nj.nec.com/424613.html>. Acesso em 15 de Fevereiro de 2004, 16:29.

LIBERTY, J.; KRALEY, M. **Aprendendo a desenvolver documentos XML para Web**. Makron Books. São Paulo, 2001.

MARTIN, J. C.; JULIA, L.; CHEYER, A. **A Theoretical Framework for Multimodal User Studies**. Cooperative Multimodal Communication: Second International Conference, CMC98. 1998.
Disponível em <citeseer.nj.nec.com/martin98theoretical.html>. Acesso em 15 de Fevereiro de 2004, 21:55.

MENEZES, P. F. B. **Linguagens Formais e Autômatos**. Sagra Luzzatto. Porto Alegre: Instituto de Informática da UFRGS, 2000.

MINE, M. R.; BROOKS JR, F. P.; SÉQUIN, C. H. **Moving Objects in Space: Exploiting Proprioception in Virtual-Environment Interaction**. Proc. of SIGGRAPH '97, (1997), pp. 19-26. Published in the journal "Computer Graphics", vol. 31, number "Annual Conference Series", pág. 19-26, ano 1997.
Disponível em <http://citeseer.nj.nec.com/mine97moving.html>. Acesso em 20 de Fevereiro de 2004, 17:43.

NIFS. **VFIVE - Vector Field Interactive Visualization Environment**. NIFS – National Institute for Fusion Science, 2005.
Disponpível em <http://www.nifs.ac.jp/seika/ver2/vr-e.html>. Acesso em 07 de Setembro de 2005, 14:38.

NIST. **A Virtual Reality Machine Tool**. NIST – National Institute of Standards and Technology, 2005.
Disponível em <http://www.nist.gov/public_affairs/gallery/vrmanu.htm>. Acesso em 06 de Setembro de 2005, 14:03.

NETTO, A. V.; MACHADO, L. S.; OLIVEIRA, M. C. F. **Realidade Virtual – Fundamentos e Aplicações**. Visual Books, 2002.

OSBORNE, M.; MACNISH, C. K. **Processing Natural Language Software Requirement Specifications**. In Proc. ICRE'96: 2nd IEEE International Conference on Requirements Engineering, pages 229-236. IEEE Press, 1996.
Disponível em <http://citeseer.nj.nec.com/152389.html>. Acesso em 22 de Fevereiro de 2004, 16:37.

OVIATT, S. **Ten Myths of Multimodal Interaction**. Communications of the ACM, in press, vol. 42, pág. 74-81, ano 1999.
Disponível em <citeseer.nj.nec.com/oviatt99ten.html>. Acesso em 16 de Fevereiro de 2004, 22:01.

OVIATT, S.; DE ANGELI, A.; KUHN, K. **Integration and Synchronization of Input Modes during Multimodal Human-Computer Interaction**. Proceedings of the workshop "Referring Phenomena in a Multimedia Context and their Computational Treatment", The Association for Computational Linguistics/European Chapter of the Association for Computational Linguistics - ACL/EACL'97, July 11, 1997, Madrid. 1-13.
Disponível em <citeseer.nj.nec.com/oviatt97integration.html>. Acesso em 16 de Fevereiro de 2004, 18:02.

PETRELLI, D.; DE ANGELI, A.; GERBINO, W.; CASSANO, G. **Referring in Multimodal Systems: The Importance of User Expertise and System Features**. In The Association for Computational Linguistics - ACL (1997). 14-19.
Disponível em <citeseer.nj.nec.com/petrelli97referring.html>. Acesso em 16 de Fevereiro de 2004, 21:11.

PINHO, M. S. **Interação em Ambientes Tridimensionais**. 3rd SRV 2000 - Workshop on Virtual Reality. WRV Tutorial 1. October, 16-18, 2000. Gramado - RS - Brazil.
Disponível em <http://www.inf.pucrs.br/~pinho/3DInteraction/>. Acesso em 20 de Fevereiro de 2004, 19:35.

PINHO, M. S. **Manipulação Simultânea de Objetos em Ambientes Virtuais Imersivos**. Tese de Doutorado em Ciência da Computação. Porto Alegre, Dezembro de 2002.
Disponível em <http://www.inf.ufrgs.br/~pinho/Tese/PinhoTese.pdf>. Acesso em 20 de Fevereiro de 2004, 18:05.

PINHO, M. S., KIRNER, C. **Uma Introdução à Realidade Virtual**. SIBGRAPI'97. X Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens
14 a 17 de outubro de 1997, Campos do Jordão, SP.
Disponível em <http://mirror.impa.br/sibgrapi97/cursos/rvirtual/>. Acesso em 28 de Março de 2004, 18:48.

QMI SOLUTIONS. **Virtual Reality Presentations**. QMI Solutions' Technology Forum 19-20 May, Australia, 2005.
Disponível em <http://www.qmisolutions.com.au/article.asp?aid=155>. Acesso em 07 de Setembro de 2005, 14:37.

RODELLO, I. A.; BREGA, J. R. F.; SEMENTILLE, A. C. **Interação com dispositivos de entrada não convencionais em ambientes virtuais desenvolvidos com Java 3D**. Proceedings SVR 2003 – VI Symposium on Virtual Reality. 15-18 October. Ribeirão Preto, SP – Brasil.

RPTT. **Virtual Reality Helps to Treat Babie´s Hearts**. RPTT - Roland Piquepaille's Technology Trends, 20 de Outubro de 2004.
Disponível em <http://radio.weblogs.com/0105910/2004/08/20.html>. Acesso em 07 de Setembro de 2005, 09:24.

SCHWITTER, R.; FUCHS, N. E. **Attempto - From Specifications in Controlled Natural Language towards Executable Specifications**. In Proceedings of the GI EMISA Workshop, Tutzing, Germany, 1996.
Disponível em <citeseer.nj.nec.com/schwitter96attempto.html>. Acesso em 22 de Fevereiro de 2004, 16:17.

SHAY, W. A. **Sistemas Operacionais**. Makron Books. São Paulo – SP, 1996.

SILBERSCHATZ, A.; GALVIN, P.; GAGNE, G. **Sistemas Operacionais – Conceitos e Aplicações**. Campus. Rio de Janeiro – RJ, 2000.

STRAYLIGHT. **Straylight Installs 26 Seat Head Mount Based Theater**. Las Vegas, NV, May 3, 1994.
Disponível em <http://www.strayvr.com/CableTroni.html>. Acesso em 07 de Setembro de 2005, 14:39.

TANENBAUM, A. S. **Modern Operating Systems**. Pretice Hall, EngleWood Cliffs, NJ, 1992.

TANENBAUM, A. S. **Organização Estruturada de Computadores**. 4ª edição. Editora LTC. Rio de Janeiro – RJ, 1999.

TORRES, G. **Hardware Curso Completo**. Axcel Books. Rio de Janeiro – RJ, 1996.

TU. **Galeria de Fotos**. TU Berlin, 2005.
Disponível em
<http://www.tu-berlin.de/uebertu/fotogalerie/forschung_lehre/slides/VirtualReality.html>.
Acesso em 06 de Setembro de 2005, 14:33.

UNT. **REGO - Reinventing Government**. UNT – University of North Texas Libraries, 12 de Janeiro de 2001.
Disponível em <http://govinfo.library.unt.edu/npr/rego/update3/nasmed1.htm>. Acesso em 07 de Setembro de 2005, 14:36.

YNEMINE, S. T. **Conhecendo o JavaScript**. Visual Books. Florianópolis – SC, 2002.

ZAÏANE, O. R.; FALL, A.; ROCHEFORT, S.; DAHL, V.; TARAU, P. **Concept-Based Retrieval using Controlled Natural Language**. In NLDB: Workshop on Natural Language and Databases, Vancouver, June 1997.
Disponível em <citeseer.nj.nec.com/17303.html>. Acesso em 22 de Fevereiro de 2004, 13:50.

ANEXO A – Protótipo em JavaScript

Um arquivo VRML é uma descrição textual de um mundo virtual. O arquivo contém texto que pode ser criado com qualquer editor de texto.

O VRML dispõe de uma variedade de Nós para animações, incluindo sensores de tempo (*TimeSensor*), Nós interpoladores, Nós de sensores etc. As habilidades de animações dos sensores e interpoladores do VRML são estáticas, ou seja, são definidas e não podem ser alteradas pelo usuário. Para animações complexas, os sensores e interpoladores são limitados. Nesta hora é que se pode usar os Nós *Scripts* (AMES *et al.*, 1997).

O Nó *Script* permite agir como o estágio de partida para controlar a animação; processar entradas para agir como o estado de Lógica em uma animação; adicionar e remover objetos do mundo virtual; permite um controle matemático do comportamento.

O Nó *Script* associa VRML com uma linguagem de *script*, como JavaScript por exemplo.

Um Nó *Script* pode ser entendido como uma forma particular de controle ou de sensor. Como qualquer Nó de controle ou sensor, este Nó requer uma lista de campos (*field*), eventos de entrada (*eventIn*) e eventos de saída (*eventOut*). A descrição do Nó deve complementar a definição destes campos, permitindo uma dada finalidade. Um Nó *Script* deve ser entendido como um Nó que recebe informações de outros Nós através dos eventos de entrada, processa e envia estas informações na forma de eventos de saída.

Logicamente, não serão encontrados Nós *Script* independentes de descrição de rotas, que mapeiam a troca de informações entre os Nós externos e o Nó *Script*. Uma exigência importante é a necessidade de equivalência de tipos de elementos entre os Nós que trocam informações.

Ynemine (2002) explica que "...JavaScript é uma linguagem de *script* de programação. Não é compilada e sim interpretada, ou seja, para que ela seja executada, é necessário

que o navegador a interprete. Dessa maneira, é fundamental que o navegador seja compatível com a linguagem de *script*...".

É bom deixar claro que apesar do JavaScript ser derivado do Java, ambas são linguagens distintas.

O Java é uma linguagem que necessita de uma fase de compilação sendo necessário um compilador Java.

O JavaScript, por sua vez, é uma linguagem interpretada, além disso, contém bem menos objetos e comandos que o Java.

Existem duas maneiras de rodar um JavaScript em VRML: através de códigos embutidos diretamente no arquivo VRML ou através da inclusão de um arquivo com extensão ".js" no arquivo VRML.

A Figura 59 mostra a um arquivo VRML que permite a Interação do Usuário com o Mundo VRML com o uso do JavaScript (Figura 60).

O resultado pode ser visto na Figura 61.

Nota-se como resultado, as limitações com o uso do JavaScript, o usuário pode alterar o Mundo Virtual, fazer animações mais complexas, porém, não contempla o uso de dispositivos não-convencionais. Encontra-se então mais uma limitação.

Para que se possa fazer uso do poder máximo do VRML é necessário usar a linguagem Java, através dos *Applets* Java.

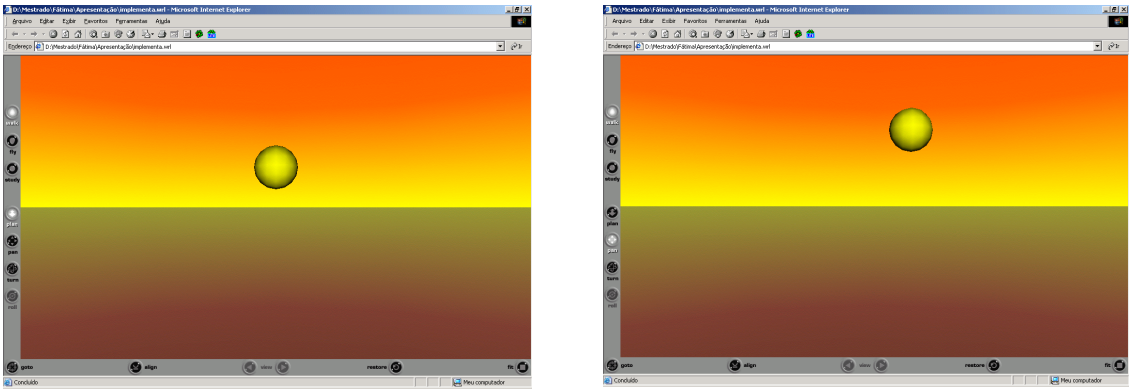
Através dos *Applets* Java é possível fazer a entrada de dados com o dos dispositivos convencionais ou não.

```
#VRML V2.0 utf8
Group {
      children [
         DEF Botao TouchSensor {}
         Background {
           skyColor [ 1.0 0.0 0.0,  1.0 0.4 0.0,  1.0 1.0 0.0 ]
           skyAngle [ 1.309, 1.571 ]
           groundColor [ 0.1 0.1 0.0,  0.5 0.25 0.2,  0.6 0.6 0.2 ]
           groundAngle [ 1.309, 1.571 ]
         },
         Transform {
             translation 0.0 1.1 0.0
             children DEF Esfera Transform {
                 children Shape {
                     appearance Appearance {
                         material Material {
                                      diffuseColor 1.0 1.0 0.0
                         }
                     }
                     geometry Sphere { radius 0.6 }
                 }
             }
         },
         DEF Tempo TimeSensor {
             cycleInterval 6.0
         },
         DEF Mover Script {
             url "movimenta.js"
             eventIn  SFFloat movimentar
             eventOut SFVec3f value_changed
         }
     ]
}
ROUTE Botao.touchTime TO Tempo.startTime
ROUTE Tempo.fraction_changed TO Mover.movimentar
ROUTE Mover.value_changed TO Esfera.set_translation
```

Figura 59 - Arquivo VRML (implementa.wrl) que chama um arquivo JavaScript (movimenta.js).

```
function movimentar( fraction, eventTime ) {
      value_changed[0] = fraction;
      value_changed[1] = fraction;
      value_changed[2] = 0.0;
}
```

Figura 60 - Arquivo JavaScript (movimenta.js).



Mundo original ***Mundo alterado pela ação do usuário***

Figura 61 - Exemplo do uso do VRML com JavaScript.