**FUNDAÇÃO UNIVERSIDADE FEDERAL DO RIO GRANDE**
**CURSO DE PÓS GRADUAÇÃO EM ENGENHARIA OCEÂNICA**

# O Uso de Realidade Virtual na Manufatura da Indústria de Construção Naval/*Offshore*

**Danúbia Bueno Espíndola**

Dissertação apresentada à Comissão de Curso de Pós-Graduação em Engenharia Oceânica da Fundação Universidade do Rio Grande, como requisito parcial à obtenção do título de Mestre em Engenharia Oceânica

Orientadora: Sílvia Silva da Costa Botelho, Dra.

Co-Orientador: Nelson Duarte Filho, Dr.

Rio Grande, julho 2007.

*A minha mãe Ângela e meu pai João Carlos, pelo apoio durante toda a minha vida*

**AGRADECIMENTOS**

À banca examinadora.

Ao CNPQ, pelo apoio financeiro.

À minha orientadora e ao meu co-orientador, professores Sílvia e Nelson, sou grato pela orientação.

O meu especial obrigado a Sílvia que mesmo estando constantemente cheia de trabalho sempre arranjava um tempo para me atender, uma pessoa a quem admiro pelo profissionalismo e me inspira, anonimamente, a continuar neste caminho....e que por sinal... não entendo como ainda não enlouqueceu, ;-) Obrigado pela confiança.

Aos colegas a quem sempre recorro, Nelson, Bicho, Vinícius, Ivete, Eder... Ao Jônata sempre disposto, atento e pronto a ajudar. Ao pessoal da Coppe-RJ, em especial ao Marcello pelos incansáveis e-mails respondendo as minhas dúvidas...

Às minhas "amiugas", de tempos e de agora. A Cacá, a quem admiro pela alegria e transparência e sempre me aguenta... A Polaca, Thayane, Gordo e Angela que adoro.... A Karina, Gi, Gre que iniciaram comigo esta caminhada....A turma do museu Carminha, Landa, Simone, Vilmar e Vika, nos divertimos muito... A Mi e Nega sempre carinhosas e se fazendo presentes, a loquinha da Mi com quem amo entrar em longas discussões filosóficas e a Negrucha a quem possuo um grande carinho quase que maternal, rimos e choramos muito nestes últimos tempos. A recente e atenciosa Nadica. Nilza, Meuri e Jane como poderia esquecer o trio barulho....A Rê e Mélia que mesmo distantes grandes companheiras de guerra. A todos mesmo os que esqueci...

Às minhas tias Graça, Ciça, Lena... e meus avós sempre presentes...

À Neida pela atenção, conversas, lanchinhos, passeios, risadas e travessuras. A Fabi pela amizade.

Ao meu namorado, Henrique, pelo carinho, paciência, incentivo, apoio, preocupação e amor em todos os momentos...Te amo.

À minha mãe Angela pelo apoio incondicional, a suas preocupações e indagações sobre o que falta, se consegui isso.... se fiz aquilo...se...se....pelo amor e eterna dedicação a mim e minha família. Agradeço ao meu pai João e minhas irmãs Danusa e Dandara que são os alicerces de minha força.

Enfim à Deus por me proporcionar está vida maravilhosa onde através da luta sempre conquisto o que desejo, dando maior sabor e valor às vitórias, obrigado!!!

# RESUMO

Este trabalho propõe uma metodologia para visualização distribuída de plantas de manufatura. A metodologia considera o uso de um conjunto de ferramentas capazes de propiciar a visualização imersiva de cenários virtuais. Projetos CAD, modelos e simulações são obtidos através de ferramentas de manufatura digital utilizadas na indústria e podem ser visualizados neste experimento em um dispositivo tipo CAVE. A proposta é validada em um estudo de caso associado à Indústria Naval/*Offshore*. Inicialmente o estudo apresenta os conceitos sobre os Sistemas de Construção Naval e *Offshore* no cenário mundial e os Sistemas de Manufatura Digital utilizados para esses fins. Após, o trabalho focaliza a utilização da Realidade Virtual em processos industriais identificando os objetivos e desafios da implantação de uma metodologia para visualização imersiva distribuída. Por fim, são apresentados testes e resultados obtidos com a utilização da metodologia de solução proposta, validados através de um estudo de caso realizado na Indústria Naval/Offshore.

Palavras-chave: Manufatura Digital, Realidade Virtual, Visualização Distribuída, Indústria Naval/Offshore, CAVE.

# ABSTRACT

This study proposes a methodology for immersive distributed visualization of Manufacturing Plants. A set of tools that make possible the immersive visualization of virtual sceneries for plant simulation are applied. CAD projects, models and simulations used in industry can be viewed in this experiment in device CAVE. The proposed ideas are then validated on a case study associated with Shipbuilding and Offshore Industries. Initially the study present the concepts about Shipbuilding and Offshore Systems in the world scenery and *Digital Manufacturing Systems* that can be used. After this discussion, this study focus on industrial application of virtual reality for then to identify the objectives and challenges of the implantation of one methodology for immersive distributed visualization. Finally *Tests and Results* are presented for the proposed methodology solution that will be validated through the study of a real case associated with Shipbuilding and Offshore Industries.



Keywords: Digital Manufacturing, Virtual Reality, Distributed Visualization, Shipbuilding and Offshore Industries, CAVE.

RESUMO ..... 4
ABSTRACT ..... 5
LISTA DE FIGURAS ..... 7
LISTA DE TABELAS ..... 8
1 INTRODUÇÃO ..... 9
1.1 OBJETIVOS ..... 11
1.2 ORGANIZAÇÃO DA DISSERTAÇÃO ..... 11
2 SISTEMAS DE CONSTRUÇÃO NAVAL E OFFSHORE ..... 13
2.1 UMA VISÃO PANORÂMICA DA INDÚSTRIA NAVAL/OFFSHORE ..... 13
2.2 O PROCESSO DE CONSTRUÇÃO NAVAL/OFFSHORE ..... 15
2.3 MANUFATURA DIGITAL ..... 20
2.4 FERRAMENTAS DE MANUFATURA DIGITAL PARA CONSTRUÇÃO NAVAL/OFFHORE ..... 24
3 A RV NA CONSTRUÇÃO NAVAL E OFFSHORE ..... 31
3.1 CONTEXTUALIZAÇÃO DA REALIDADE VIRTUAL ..... 31
3.2 SISTEMAS COMPUTACIONAIS PARA A RV ..... 38
3.3 APLICAÇÕES DA RV NA INDÚSTRIA ..... 39
3.4 RV APLICADA A CONSTRUÇÃO NAVAL/OFFSHORE ..... 41
4 DA MODELAGEM À VISUALIZAÇÃO DISTRIBUÍDA ..... 50
4.1 REQUISITOS NECESSÁRIOS ..... 50
4.2 ETAPA DE MODELAGEM ..... 51
4.3 ETAPA DE CONVERSÃO ..... 52
4.4 ETAPAS DE VISUALIZAÇÃO E INTERATIVIDADE ..... 54
5 TESTES E RESULTADOS ..... 61
5.1 ETAPA DE MODELAGEM ..... 61
5.2 ETAPA DE CONVERSÃO ..... 63
5.3 ETAPA DE VISUALIZAÇÃO ..... 66
6 CONCLUSÕES E PERSPECTIVAS FUTURAS ..... 73
APÊNDICES ..... 75
REFERÊNCIAS ..... 96

**LISTA DE FIGURAS**

Figura 2.1. Etapas do processo de construção na Indústria Naval/Offshore ..... 17
Figura 2.2. Esquema de montagem do protótipo virtual em sistemas digitais. ..... 18
Figura 2.3. Bloco pré-montado. ..... 19
Figura 2.4. Estação de corte das linhas de painéis. ..... 19
Figura 2.5. Módulo funcional pré-montado. ..... 19
Figura 2.6. Oficina de soldagem de canos. ..... 19
Figura 2.7. Modelo CAD da P43. ..... 22
Figura 2.8. Editor gráfico do PROMODEL. ..... 25
Figura 2.9. Saída do PROMODEL. ..... 25
Figura 2.10. Um dos módulos da ferramenta DELMIA. ..... 29
Figura 3.1. Exemplos de CAVES. ..... 34
Figura 3.2 Aplicações industriais da Realidade Virtual (*a, b, c, d, e, f*). ..... 41
Figura 3.3 Walkinside. ..... 43
Figura 3.4 Division Reality. ..... 43
Figura 3.5. Tela Environ. ..... 44
Figura 4.1. Metodologia da solução proposta. ..... 52
Figura 4.2. Estrutura modular da Game Engine. ..... 55
Figura 4.3. Ambiente trabalho do UnrealED. ..... 59
Figura 5.1. Oficina real do estudo de caso. ..... 62
Figura 5.2. Oficina virtual modelada. ..... 62
Figura 5.3. Estaleiro 1(a), estaleiro 2(b) e oficina(c). ..... 62
Figura 5.4. Conversão do estaleiro 1 (a), (b), (c), (d) e (e). ..... 64
Figura 5.5. Layout do sistema de projeção utilizado no experimento (Cave). ..... 68
Figura 5.6. Visualização distribuída de cenários navais na CAVE (*a, b, c, d, e, f, g, h*). ..... 71
Figura A.1. Redução de polígonos no VizUp. ..... 76
Figura A.2. Importar VRML. ..... 77
Figura A.3. Selecionando VRML. ..... 77
Figura A.4. Opções de importação VRML. ..... 77
Figura A.5. Exportar ASE. ..... 77
Figura A.6. Opções de exportação. ..... 77
Figura A.7. Utilizando plugin ActorX. ..... 78
Figura A.8. Diretório de exportação psa e psk. ..... 79
Figura A.9. Exportando psa. ..... 79
Figura A.10. Gerando o arquivo psa de animação. ..... 80
Figura A.11. Componentes utilizados da barra de ferramentas do UnrealED ..... 80
Figura A.12. Criando os limites do cenário virtual. ..... 81
Figura A.13. Aplicando texturas. ..... 81
Figura A.14. Editando pacote de importação das malhas estáticas. ..... 82
Figura A.15. Importando as malhas estáticas. ..... 82
Figura A.16. Adicionando no cenário componentes estáticos. ..... 83
Figura A.17. Aumentando o tamanho da malha. ..... 84
Figura A.18. Definindo atores e suas ações. ..... 85
Figura A.19. Importando animação. ..... 85
Figura A.20. Ligando a animação a malha estática. ..... 86
Figura A.21. Propriedades do ator. ..... 86
Figura A.22. Ações do ator. ..... 87
Figura A.23. Propriedades das ações do ator. ..... 87
Figura A.24. Executando a animação. ..... 88

**LISTA DE TABELAS**

Tabela 3.1. Ferramentas gratuitas e comerciais para aplicações de RV..... 38
Tabela 3.2. Desenvolvedores e seus projetos que utilizam Game Engines..... 46
Tabela 5.1. Configurações originais dos modelos utilizados. ..... 63
Tabela 5.2. Reduções do modelo 1 (estaleiro 1). ..... 65
Tabela 5.3. Reduções dos modelos 2 e 3 (estaleiro 2 e oficina) ..... 65
Tabela 5.4. Resultados quanto a escalabilidade. ..... 69

# 1 INTRODUÇÃO

Atualmente, a computação está cada vez mais presente nos mais variados setores produtivos. Com os avanços tecnológicos em *software* e *hardware* havidos ao longo do tempo, os recursos computacionais agora disponíveis vêm sendo utilizados em diferentes processos industriais, auxiliando a realização das diversas etapas de desenvolvimento e construção dos produtos.

Novas tecnologias de construção, abrangendo a definição do produto, o planejamento, o detalhamento e a validação da produção; o processo de construção; e a logística de distribuição, podem ser atualmente fornecidas sob a forma de ferramentas computacionais que possuem recursos para modelagem e simulação.

Integrado à ferramentas CAD/CAE/CAM (*Computer-Aided Design/Computer-Aided Engineering/Computer-Aided Manufacturing*) as recentes soluções DMU (*Digital Mock-Up* - construção de maquetes digitais) têm proporcionado a representação digital das diferentes etapas de manufatura de um produto [Kim 2002]. Tais soluções congregam tecnologias de comunicação e computação gráfica, fornecendo inovadoras possibilidades na realização de tarefas e processos de engenharia [Brunetti 1999].

Associada à computação gráfica, a Realidade Virtual (RV) surge como uma ferramenta capaz de acrescer novas técnicas a sistemas utilizados nas diferentes etapas de desenvolvimento de um produto, possibilitando uma interface realista de interação e visualização na área da manufatura.

O termo RV abrange tecnologias de interatividade e projeção tridimensional, imersivas e não imersivas, centralizadas ou distribuídas. Ambientes imersivos de multiprojeção tridimensionais são conhecidos como CAVEs (*Cave Automatic Virtual Environment*). Em ambientes de multiprojeção CAVE, a imagem final apresentada ao observador é formada pela renderização e composição de imagens projetadas por múltiplos dispositivos. A utilização de tal configuração possibilita, entre outros aspectos, o aumento do grau de imersão e interação do usuário do sistema com o ambiente virtual apresentado [Ferreira 1999].

Grandes conglomerados mundiais associados às indústrias automotiva, aeroespacial e naval, para citar alguns, vêm investindo crescentemente nessas tecnologias, de forma a responder mais eficientemente às demandas do mercado e dos consumidores reduzindo custos e tempo de desenvolvimento [Kim 2003] [Dassault 2005]. De outra parte, considerando o plano de reabilitação da indústria nacional de construção naval *offshore* recentemente lançado pelo governo federal, há que se qualificar mão de obra e encontrar soluções capazes de tornar competitivo o parque industrial naval a ser (re)instalado, o que passa necessariamente pelo domínio de tecnologias associadas à manufatura digital.

Nessa perspectiva, e devido à natureza distribuída e cooperativa dos atuais processos de construção e comissionamento de grandes estruturas navais/*offshore*, a possibilidade de reprodução remota, virtual e interativa das diferentes etapas de manufatura de navios e plataformas torna-se um aspecto importante, inclusive capaz de determinar o (in)sucesso dos consórcios constituídos para atender as encomendas estabelecidas.

A utilização de técnicas de Realidade Virtual (RV) surge como um novo paradigma capaz de integrar diferentes níveis da cadeia de produção naval/*offshore*, distribuídos geográfica e temporalmente, com o objetivo de enfrentar redução de custos e prazos de entrega. Nesse contexto surgem novos desafios associados ao desenvolvimento de ferramentas capazes de visualizar as diferentes etapas do processo de construção naval/*offshore*.

A obtenção e o tratamento de cenários, em tempo real, representando em detalhes a modelagem de projetos e a dinâmica da planta de construção; a necessidade do tratamento de situações realísticas de interação, tais como colisões de componentes, ferramentas e obstáculos; o acionamento de dispositivos; e a percepção de situações de risco, são algumas das questões a serem investigadas. Neste sentido ainda são escassas as soluções capazes de a partir das ferramentas digitais atualmente disponíveis, criarem ambientes imersivos de visualização distribuída onde todas essas características estejam presentes.

## 1.1 OBJETIVOS

Em sintonia com a política de retomada do funcionamento da indústria da construção naval/*offshore* brasileira, busca-se neste trabalho a identificação dos desafios associados à utilização de tecnologias de Realidade Virtual ao longo do processo de manufatura em questão e à integração de recursos de visualização imersiva às diferentes ferramentas DMU adotadas nos processos industriais correspondentes. De forma mais precisa, se propõe uma metodologia para a visualização imersiva distribuída, a ser utilizada em ambientes de multiprojeção para a representação de cenários associados ao processo de construção naval/*offshore*.

Consideram-se cenários que representam etapas do processo de manufatura, e podem ser fornecidos por diferentes ferramentas de DMU. A metodologia proposta é validada em uma implementação de baixo custo, a partir de modelos e simulações desenvolvidos na ferramenta DELMIA [Delmia 2002].

## 1.2 ORGANIZAÇÃO DA DISSERTAÇÃO

A presente dissertação está organizada em seis capítulos. O presente capítulo introduz o que será abordado, o **Capítulo 2 - Sistemas de Construção Naval e *Offshore*** - contextualiza a indústria da construção naval e *offshore* no cenário mundial, apresenta os aspectos relacionados ao processo de construção de navios e plataformas e finaliza com uma descrição dos sistemas de manufatura digital mais freqüentemente utilizados. Visa contextualizar o leitor para um melhor entendimento do trabalho enfocado.

O **Capítulo 3 - A RV na Construção Naval e *Offshore*** - apresenta uma breve contextualização sobre Realidade Virtual, introduzindo conceitos fundamentais a ela relacionados, e mostra algumas ferramentas de RV. Também são apresentadas aplicações da RV na indústria em geral e as suas utilizações na indústria naval/*offshore* em particular. O capítulo é finalizado com a descrição dos principais desafios e ferramentas associados à visualização distribuída de cenários navais/*offshore*.

O **Capítulo 4 - Da Modelagem à Visualização Distribuída** - descreve a metodologia para visualização de cenários associados ao processo de manufatura naval/*offshore* proposta neste trabalho. Para esse propósito são mostrados aspectos de modelagem, de conversão de

formatos de representação, e de visualização e interatividade de cenários dinâmicos obtidos com ferramentas DMU.

O **Capítulo 5 - Testes e resultados** - apresenta um estudo de caso associado à utilização da ferramenta DELMIA na modelagem de cenários em processos de construção da indústria naval. São descritos aspectos utilizados para a validação da metodologia e indicadas as suas vantagens e limitações.

Finalmente, o **Capítulo 6 - Conclusões e Perspectivas Futuras** - indica as contribuições do trabalho, apresenta algumas restrições para a sua plena utilização e sugere implementações capazes de dar continuidade ao estudo, com o propósito de aperfeiçoar a metodologia proposta.

# 2 SISTEMAS DE CONSTRUÇÃO NAVAL E OFFSHORE

*Ambientar sobre a posição e a importância da Indústria Naval/Offshore e conceituar processos de construção nessa indústria, além de apresentar ferramentas computacionais nela utilizadas.*

## 2.1 UMA VISÃO PANORÂMICA DA INDÚSTRIA NAVAL/OFFSHORE

No cenário mundial, mais de 90% da Indústria de Construção Naval está localizada em quatro regiões: Coréia, Japão, China e União Européia. A União Européia abrange 15,7% da indústria naval do mundo ficando em terceiro lugar; logo atrás vem a China com 14,4% após uma crescente evolução a partir da década de noventa; em segundo lugar aparece o Japão com 27,1% da produção mundial; e em primeiro lugar aparece a Coréia do Sul com 38,0%. [Pires 2005].

Além de ser a terceira maior fabricante mundial de aço, no período entre 1994 e 1996, a Coréia triplicou a capacidade de seus estaleiros, reduzindo seus custos de forma a conquistar novos mercados [Kosa & Association 2005]. As principais empresas atualmente atuantes na Coréia são: *Hiunday Heavy Industries* (elevado nível de automação), *Daewoo Shipbuilding and Heavy Machinery* (um dos mais modernos estaleiros do mundo), *Samsung Heavy Industries* e *Hanjin Heavy Industries*.

Japão, como segundo maior construtor naval/*offshore*, caracteriza-se pelo massivo uso de ferramental tecnológico, alcançando altos níveis de qualidade no desenvolvimento de seus produtos. Uma estratégia japonesa é o estabelecimento de parcerias entre diferentes grupos, como, por exemplo, a *Mitsubishi* e a *Hitachi*, na área siderúrgica. Atualmente, dezenove empresas japonesas detêm 95% da construção naval no país, sendo doze de médio porte e sete de grande porte. Dessas, podem-se citar: *Mitsubishi Heavy Industries*, *Ishikawagima Harima Heavy Industries*, *Hitachi Zosen Corporation*, *Kawasaki Heavy Industries*, *Sumitomo Heavy Industries*, *Mitsui Engineering and Shipbuilding* e NKK Co[Barboza 2005].

Mesmo que inicialmente desprovida de tecnologias sofisticadas na produção de navios graneleiros de baixo custo, a China ocupa hoje uma posição de destaque no mercado mundial. Apesar das dificuldades associadas à importação de peças e equipamentos, e ao pagamento de

licenças para utilização de projetos e processos, muitos deles desenvolvidos pelos japoneses, a China aposta firme no segmento naval, tendo como meta se tornar líder mundial na construção de navios. A China possui dois principais grupos empresariais: o *North Shipbuilding Group* e o *South Shipbulding Group*. Cabe salientar que enquanto estaleiros chineses possuem em média de nove a doze mil operários, estaleiros japoneses com 900 à 1500 trabalhadores possuem a mesma capacidade de produção, característica que evidencia a alta automatização da indústria naval japonesa.

Por outro lado, a Indústria de Construção Naval européia vem se organizando a fim de voltar a competir com a Ásia na produção de estruturas navais. Recentemente criada a *Association of Western European Constructors* (AWES) tem como objetivo a formação de nichos de mercado associada a navios de passageiros e a estruturas offshore. Entretanto a falta de espaço físico para a construção de grandes petroleiros é ainda um desafio na Europa. Os principais grupos empresariais europeus na construção naval civil são: AKER, KVAERNER, DAMEN, HDW, ODENSE/APMOLLER, THYSSENKRUPP, GEC, IZAR e FINCANTIERI [Barboza 2005].

O Brasil, na década de setenta, alcançou o apogeu de seu crescimento no setor de construção naval. Na época, a crescente demanda de importação de petróleo fez com que ocorresse uma expansão dos estaleiros para produção de navios petroleiros e minipetroleiros. A maior parte dos navios produzidos era direcionada ao mercado interno, tendo como maior contratante a Petrobrás. Durante a década de oitenta ocorreu uma crise no setor, devido à queda no número de encomendas dos armadores nacionais. Em meados de 2000, em decorrência de políticas governamentais, iniciou-se a retomada do número de encomendas pela Petrobrás. Tais encomendas estão associadas à necessidade de substituição da frota de navios de apoio *offshore* e a demanda pela construção de novas plataformas. Estima-se que o Brasil deixe de ganhar, anualmente, cerca de sete bilhões de dólares, devido aos fretes realizados por navios estrangeiros [Barboza 2005].

No Brasil existem hoje vinte e sete estaleiros que geram cerca de trinta mil empregos diretos. O desafio atual é implantar um parque industrial competitivo, visando atender a eminente demanda de embarcações, principalmente pela Petrobrás, bem como tornar o país um pólo exportador, em médio prazo. Para tal, se fazem necessárias a modernização dos estaleiros e a implantação de novas tecnologias de construção que garantam preço, qualidade e prazo, capazes de competir com os principais fabricantes mundiais.

## 2.2 O PROCESSO DE CONSTRUÇÃO NAVAL/OFFSHORE

Apresentam-se a seguir os principais conceitos e componentes associados à cadeia produtiva da construção naval/*offshore*, a fim de auxiliar a compreensão do impacto produzido pelo uso de tecnologias inovadoras nos processos de manufatura dessa indústria.

Os principais integrantes do processo de construção de navios e plataformas são os armadores, os clientes, os agentes financeiros, os estaleiros e a indústria de navipeças. Tal composição conduz a um segmento de mercado que pode ser dividido em duas grandes áreas: i. a construção naval militar; e ii. a construção naval civil. Este último pode ser categorizado no setor de transportes marítimos e no *offshore*. Por outro lado, os navios de transporte podem ser caracterizados, em função da carga transportada, como graneleiros, químicos e sólidos.

A indústria naval civil de transporte exerce papel fundamental no comércio mundial. Segundo a OMC (Organização Mundial do Comércio) estima-se que 80% das mercadorias são transportadas por via marítima. Devido a essa importância, a maior parte dos construtores navais focaliza sua produção na área naval civil.

No que se refere à área *offshore* é importante salientar que grandes campos de petróleo vêm sendo descobertos graças aos avanços tecnológicos de exploração em águas ultraprofundas (mais de mil metros). Em locais como África, China, Golfo do México e Brasil estão sendo realizadas pesquisas e mapeamentos de áreas de exploração futura, porém não existe ainda estoque de equipamentos, navios e plataformas condizentes com as exigências dessas novas prospecções.

Devido à diversificação dos tipos de estruturas navais e à dinâmica da demanda associada, pôde-se identificar um conjunto de características específicas, próprias do setor, e enumeradas a seguir:

- dificuldade de padronização dos processos devido a diversidade de produtos e equipamentos;
- utilização de equipamentos e materiais de grande dimensão e peso;
- processo de fabricação envolvendo complexas estruturas e necessidade de alta precisão;
- início da definição e compra de suprimentos, equipamentos e materiais, bem como a fabricação de peças antes da finalização do projeto do produto;

- grande quantidade de informação sendo processada devido aos diversos tipos de navios com especificações diferentes construídas em um mesmo estaleiro, ou em diferentes componentes de consórcios.

Os principais estaleiros mundiais devem ser flexíveis, se adequando a tais necessidades e cenários, de forma a manter as exigências de qualidade, preço e prazo. Tal capacidade de adaptação deve ainda tratar as diferentes características relacionadas à robustez e à resistência das estruturas a serem construídas, que dependerão do tipo de carga, mar em que serão utilizadas e duração de rota de cada projeto encomendado. Consórcios envolvendo diferentes grupos empresariais tornam-se uma realidade, objetivando a integrar as diferentes competências necessárias a cada nova encomenda.

A utilização de novas tecnologias digitais associadas às diferentes etapas do processo de construção de navios e plataformas permite a agilidade e adaptabilidade necessária ao segmento, além de propiciar um meio de integração e padronização entre os diferentes componentes dos consórcios envolvidos.

**Etapas do Processo de Construção Naval/*Offshore***

Atualmente, com base nas peculiaridades do setor, os sistemas digitais na indústria naval buscam automatização, integração e padronização. Em estaleiros de alta performance 90\% dos operários atuam na etapa de acabamento (pintura e soldagem), enquanto os demais processos são em grande parte automatizados.

A automatização passa pela utilização de diferentes sistemas de manufatura, compartilhados com integração de ferramentas CAD/CAE/CAM. Tecnologias de hardware para gestão de redes de alta velocidade e suporte à visualização de alto desempenho vêm sendo aplicadas, servindo como estrutura de interligação e padronização entre os diferentes sistemas existentes.

Basicamente a utilização dos sistemas digitais na Indústria Naval/*Offshore* ocorre em dois momentos distintos (ver Figura 2.1): etapa de projeto e etapa de manufatura. A etapa de projeto tem início a partir da requisição do cliente e vai até o refinamento de detalhes, fornecendo um modelo funcional e estrutural do produto para o início da manufatura. A etapa de manufatura compreende complexos processos de fabricação, montagem e erguimento de blocos, até a validação e teste dos mesmos, resultando no modelo de processos do produto digital [Kim 2002].

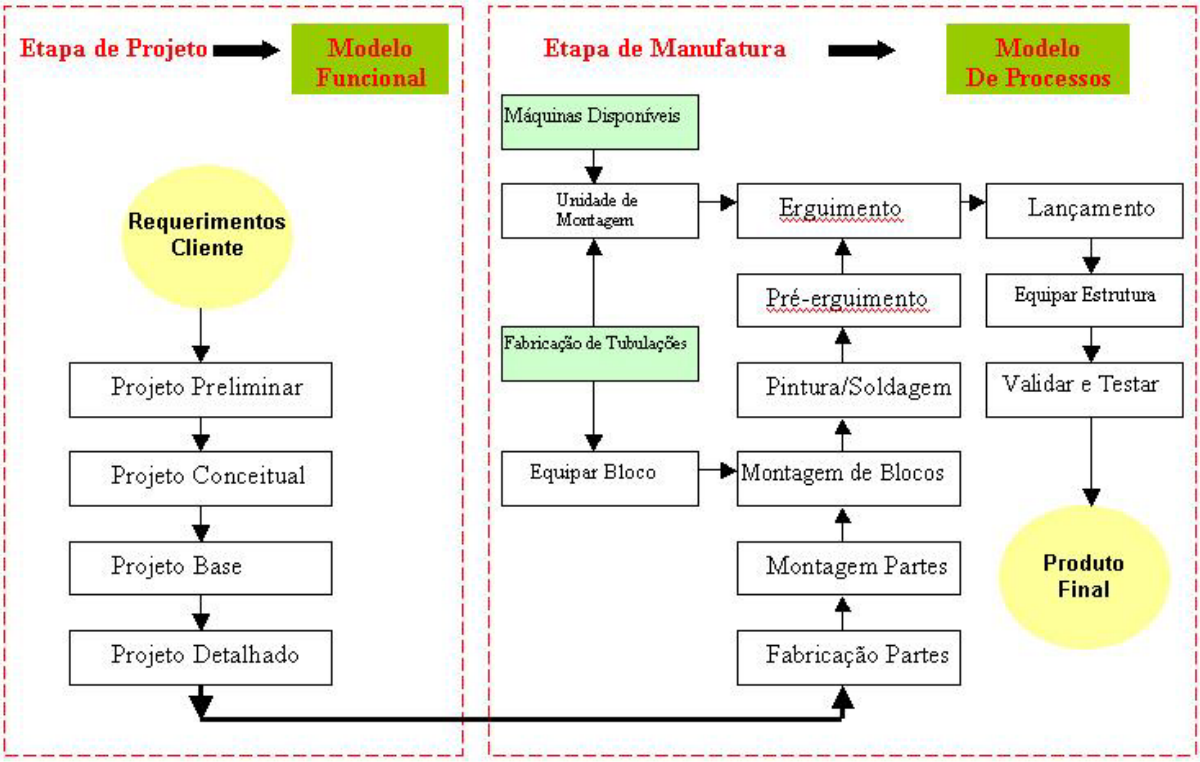



Figura 2.1. Etapas do processo de construção na Indústria Naval/Offshore

Ambas as etapas devem compartilhar informações fornecidas por projetistas, engenheiros e operários. Para tal fazem-se necessários mecanismos de padronização da informação armazenada em diferentes bancos de dados.

A etapa de projeto pode ser dividida em: projeto preliminar, conceitual, base e detalhado. O projeto preliminar é o firmado entre o contratante e o estaleiro. Ele engloba as requisições do cliente tais como capacidade, flutuação e movimento. Em seguida são definidos os materiais que serão utilizados para a produção e as suas restrições, quantidades e preços, resultando no projeto conceitual da encomenda. Com base na extração das informações do requisitante, ferramentas computacionais de projeto são utilizadas para o desenvolvimento do projeto base. Ao longo do processo o projeto base é detalhado resultando no Modelo Funcional final da embarcação.

A etapa de manufatura da produção de navios e plataformas inclui processos tais 6como: fabricação, montagem e erguimento de blocos. Tais processos representam diferentes módulos de produção, que ainda dependem de sub-processos: disponibilidade de máquinas, pintura, soldagem e pré-erguimento, formando assim o Modelo de Processos. Estes módulos acontecem em seqüências complexas de execução, através de um longo período de tempo.

Um protótipo virtual do navio ou estrutura naval, bem como o ambiente de construção, podem ser gerados a partir das informações advindas das etapas de projeto e manufatura. Tais informações, armazenadas em banco de dados, são utilizadas para diferentes tipos de simulações, testes e experimentos (ver Figura 2.2) [Kim 2003].

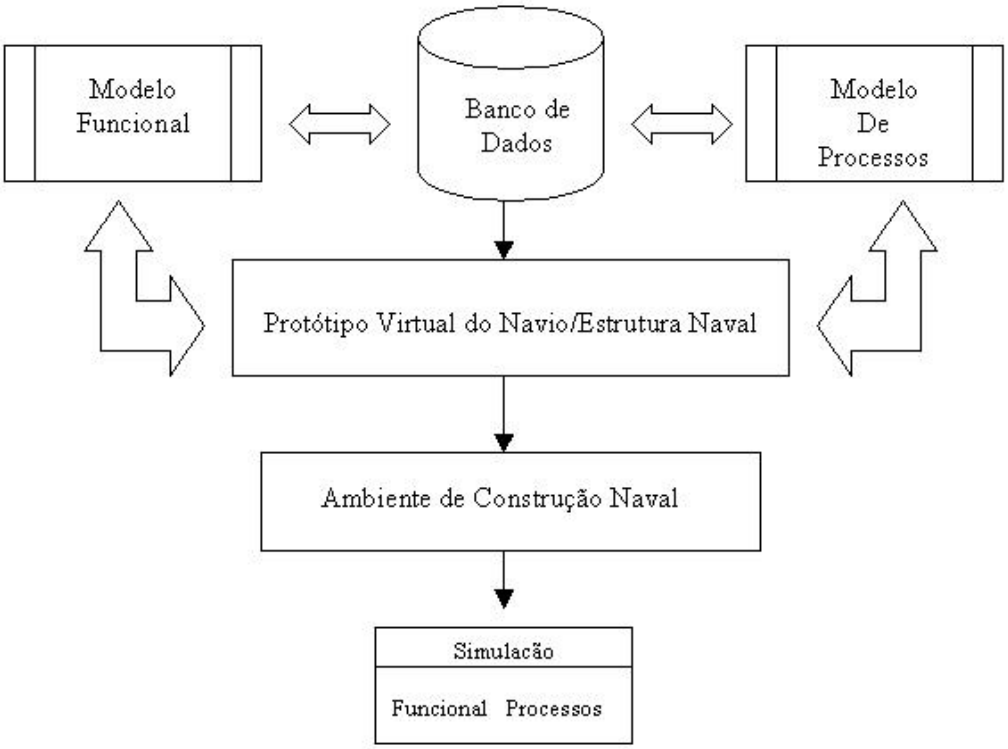



Figura 2.2. Esquema de montagem do protótipo virtual em sistemas digitais.

Basicamente, as simulações realizadas dividem-se em duas principais categorias: Simulações de Processo e Simulação Funcional.

A Simulação Funcional abrange aspectos relacionados às requisições da etapa de projeto como resistência da embarcação, comportamento hidrodinâmico, fluxo de passageiros, simulação de ancoragem, simulação de carga e descarga. Complexos modelos matemáticos são desenvolvidos e implementados em ferramentas de modelagem e simulação de alto desempenho.

A Simulação de Processos compreende aspectos de montagem e considera a área de pátio disponível para movimentação de guindastes, ordem em que os blocos serão erguidos, simulação de fabricação de peças, local onde o usuário possa interagir manipulando máquinas de corte e robôs soldadores, através de técnicas de realidade aumentada, visualização gráfica

para avaliação da manufatura, entre outros. Nessa categoria são realizadas simulações de eventos discretos, em diferentes seqüências de montagem, resultando em gráficos para análise de tempo e processos.

Na Manufatura Digital os engenheiros de produção colaboram com os grupos de projetistas de navios e plataformas para desenvolverem as seqüências de montagem do produto. Seqüências de montagem complexas podem ser quebradas em módulos menores. Dessa forma, utilizando uma estratégia baseada na hierarquia das partes e seqüências, as diferentes camadas de produção podem ser visualizadas em gráficos, verificando-se a ordem e relacionamento das partes na montagem final. Análises de tempo e custos são realizadas, de forma a cumprir as metas estabelecidas pelos investidores.

As Figuras 2.3, 2.4, 2.5 e 2.6 apresentam exemplos de oficinas de estaleiros que são simulados e modelados funcionalmente em Sistemas Digitais da Indústria Naval/*Offshore*.

Figura 2.3. Bloco pré-montado.

Figura 2.4. Estação de corte das linhas de painéis.

Figura 2.5. Módulo funcional pré-montado.

Figura 2.6. Oficina de soldagem de canos.

Também são desenvolvidas simulações envolvendo aspectos humanos e robóticos para caracterização dos fatores ergonômicos (posição dos operários, altura média dos funcionários entre outros) e detalhamentos funcionais de máquinas. As simulações irão determinar a eficácia do planejamento e da execução do projeto, além de garantir a segurança nas tarefas de treinamento e manutenção de equipamentos das plantas de manufatura. É possível também testar a habilidade humana na realização de ações que possuam pouquíssimo espaço para execução, como, por exemplo, soldagem em locais de difícil acesso.

Após detalhar e validar os processos, com o auxílio de simulações e visualizações da dinâmica do sistema de construção, as informações estarão corretas e prontas para serem utilizadas no processo de produção. Os sistemas digitais devem proporcionar uma maneira simples de extrair informações de manufatura e instruções de trabalho. A heterogeneidade de soluções existentes faz com que os desenvolvedores acabem utilizando mais de uma ferramenta para diferentes propósitos, sendo um desafio a integração de tais sistemas.

## 2.3 MANUFATURA DIGITAL

Nas últimas décadas uma série de conceitos, técnicas e ferramentas computacionais, aqui denominadas Manufatura Digital, vêm sendo utilizadas ao longo de toda a cadeia de produção industrial. Entre as ferramentas e técnicas utilizadas podem-se citar os sistemas CAD/CAM/CAE como precursores desse conceito, evoluindo para ferramentas de Engenharia Virtual, conhecidas no setor automobilístico como soluções DMU (*Digital Mock-Up*) e agregam técnicas de computação gráfica e tecnologias de comunicação [Binotto 2006].

A Manufatura Digital é um conceito amplo que abrange as etapas de planejamento e projeto, os processos de detalhamento e validação, o desenvolvimento de recursos de modelagem e simulação e, por fim, a extração dos dados de manufatura e instruções de trabalho. Neste sentido, um conjunto de ferramentas e técnicas é utilizado para alcançar estes objetivos.

Ferramentas CAD (*Computer-Aided Design* - Projeto Assistido por Computador) permitem aos *designers* e engenheiros conceber, testar, avaliar e definir um produto para

fabricação, contendo essencialmente características geométricas e físicas, concentrando-se basicamente na geração de desenhos 2D e 3D.

Sendo assim, sistemas CAD integram informações como geometria, massa e material de um produto, em uma base de dados que possibilita gerenciar todo o projeto. Essa integração reduz o ciclo produtivo e garante uma boa qualidade final do produto. De acordo com [Bessant 1986], o projeto engloba vários procedimentos, desde a criação inicial de idéias, a geração de formas geométricas, cálculos de desempenho, até o processo de fabricação.

Um dos programas mais utilizados para projetos CAD é o AutoCAD. Esse programa, desenvolvido pela Autodesk [AutoDesk 1985], está para o mercado de CAD como o Windows está para os PCs. Mais antigo que o Windows, lançado em 1985, a primeira versão do AutoCAD foi elaborada para o ambiente DOS. Tal foi a popularização desse software que virou sinônimo de CAD.

Os programas CAD, como a própria sigla sugere, permitem criar e armazenar projetos para posteriores modificações no computador. A possibilidade de modificação agiliza a produtividade dos projetistas, diminuindo os erros ocorridos, além de reduzir a quantidade necessária de projetistas para realizar determinados projetos [Castelltort 1988].

A criação de projetos 2D e 3D, no setor mecânico, contribuem para a prototipagem virtual, possibilitando o estudo das peças e das montagens. A visualização é facilitada com resultados próximos da realidade. Além disso, os programas permitem aplicações paramétricas, ou seja, qualquer modificação feita em uma peça ou desenho será refletida em todas as demais do conjunto.

Os sistemas CAD podem ser entendidos, em termos gerais, como uma aplicação da informática que tem a finalidade de facilitar e agilizar a construção de desenhos ou projetos em diversas áreas, como a indústria mecânica, automobilística, aeronáutica, eletrônica, entre outras [Zerbone 1995]. Atualmente, a maioria dos sistemas CAD modela objetos tridimensionais, por meio do qual se obtém projeções bidimensionais e volumétricas. Um exemplo de modelo CAD pode ser observado na Figura 2.7, extraída do trabalho de Corseuil [Corseuil 2003].

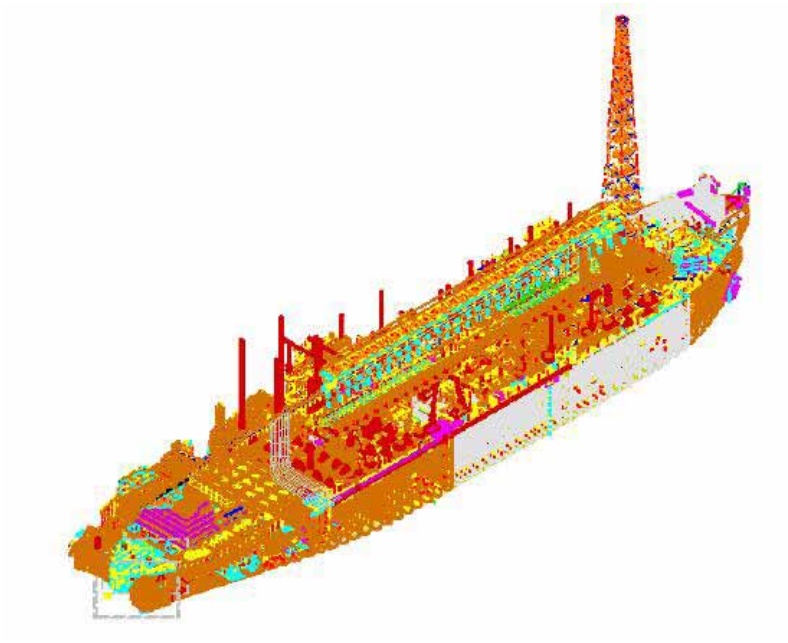


Figura 2.7. Modelo CAD da P43.

A Engenharia Assistida por Computador ou CAE (*Computer-Aided Engineering*}), ou os *softwares* CAE, utilizam modelos 3D digitais, geralmente obtidos a partir de ferramentas CAD, para simular fenômenos físicos reais como esforços, deformações, transferências de calor, tensões, escoamentos de fluídos. Outros exemplos de análise em sistemas CAE são durabilidade, acústica, injeção de plásticos, conformação mecânica e colisões.

Esses softwares permitem a realização de uma grande quantidade de cálculos em tempo reduzido, e utilizam métodos de elementos finitos e simulação de programas CNC (Controle Numérico Computadorizado).

Com a evolução do cálculo estrutural, voltado para o dimensionamento de estruturas e componentes mecânicos, a simulação computacional se tornou possível, permitindo a avaliação de um maior número de variações do projeto através de simulações baseadas em métodos de elementos finitos nos sistemas CAE.

Segundo [Zerbone 1995] a tecnologia CAE permite uma abordagem de engenharia preditiva, que envolve a construção e teste de protótipos em nível de *software*, o que diminui custos e tempo de projeto, aprimorando a qualidade do produto.

As simulações obtidas por meio de um *software* CAE auxiliam em diversas áreas de engenharia, como estruturas, análise de circuitos elétricos, projeto mecânico entre outras.

Atualmente, os computadores destinados à execução de programas CAE possuem alta capacidade computacional, o que diminui a necessidade de simplificações geométricas, sendo

as simulações realizadas mais próximas da realidade. Uma vez que o produto tenha sido testado e analisado, com o uso de ferramentas CAE, é possível utilizar o modelo digital para a criação do protótipo real, a fim de concluir os testes finais antes da fabricação, adotando uma tecnologia CAM.

Os softwares CAM (*Computer-Aided Manufacturing*) – Manufatura Assistida por Computador - são desenvolvidos para controlar o processo de fabricação de um determinado produto. Estes *softwares* são utilizados para transformar o modelo virtual em um produto real. De posse do arquivo CAD já analisado em ferramentas CAE, o CAM pode preparar algoritmos e programas que controlam máquinas para a fabricação de modelos, peças e produtos.

Assim, o sistema CAM, integrado a uma máquina operatriz, pode determinar os melhores procedimentos industriais para o processo de manufatura executado por essa máquina [Pessoa 2006].

Na década 80 já existiam sistemas que integravam todo o processo CAD/CAE/CAM. O projeto iniciava com a construção da geometria básica e de vistas do produto, em seguida era possível simulá-lo através da análise de elementos finitos. Nessa etapa o produto era definido pela interpretação de sua geometria e simulação de seu comportamento em situações reais geradas pelo computador. Posteriormente, havia a geração de programas para a serem instalados nas máquinas que finalizavam o processo de manufatura [Castelltort 1988].

Resumidamente, CAM é uma ferramenta computacional de auxílio à preparação da fabricação, reproduzindo as tecnologias usadas no chão de fábrica. Muitas vezes o termo CAM é sinônimo da programação CNC, concepção que ficou conhecida com a sigla CAD/CAM, que significa módulos de programação CNC em sistemas CAM [Ribeiro 1999].

O controle numérico computadorizado é uma técnica de controle dos movimentos das máquinas pela interpretação direta dos programas que representam instruções codificadas na forma de letras e números. Os dados são interpretados pelo sistema e um sinal de saída é gerado para controlar os componentes da máquina [Marcicano 2003].

As informações geométricas e dimensionais de um objeto, representadas pelos desenhos e cotas, são processadas por uma máquina que propicia a automação da operação. A máquina consiste em um equipamento eletrônico que recebe os dados da forma de operação, por intermédio de uma linguagem própria denominada programa CNC. As informações são processadas e devolvidas ao sistema por meio de impulsos elétricos, que são responsáveis pelo acionamento dos motores que conduzirão à máquina os movimentos com os padrões de

usinagem, realizando a operação na seqüência programada sem a intervenção do operador [Mota 2002].

O termo DMU, advindo da indústria automotiva, baseia-se em protótipos virtuais de produtos com grande utilização de técnicas e dispositivos de Realidade Virtual. Estas tecnologias estão sendo aplicadas e consideradas como a nova geração de ferramentas de apoio aos sistemas de Manufatura Digital [Kim 2002]. Essas ferramentas são utilizadas ao longo de todo o processo de desenvolvimento do produto, integrando técnicas de modelagem com computação gráfica, recursos de comunicação e acesso a banco de dados para a simulação de protótipos virtuais [Brunetti 1999].

Segundo [Daí 1996], *Digital Mock-Up* (DMU) é uma simulação computacional realística de um produto com a capacidade de desempenhar todas as funcionalidades exigidas pelo projeto, engenharia, manufatura, manutenção, e reciclagem de produto. DMUs são utilizadas como plataformas para produção e desenvolvimento de processos para comunicação e tomada de decisões de *layouts* conceituais. Isto inclui todos os tipos de simulações geométricas, ergonômicas e funcionais, envolvendo ou não os humanos.

Além disso, ferramentas de *Digital Mock-Ups* são integradas com ferramentas básicas de gerenciamento e projeto de produtos e permitem etapas de revisão, criação e pré-montagem colaborativas, utilizando ferramentas de visualização, navegação e comunicação. Atualmente algumas ferramentas DMU começam a apresentar também alguns recursos de visualização mais avançados, porém ainda com alto custo associado e com restrito nível de qualidade e desempenho.

Enfim, a Manufatura Digital se mostra como uma promessa para elevar a produtividade, melhorar a qualidade de produção, encurtar o tempo de desenvolvimento do produto, reduzir riscos de produção e, conseqüentemente, reduzir custos de desenvolvimento de projetos, passando a ser uma tendência na implementação personalizada de produção em massa e uma resposta à demanda dos clientes e ao ganho da competição global.

## 2.4 FERRAMENTAS DE MANUFATURA DIGITAL PARA CONSTRUÇÃO NAVAL/OFFHORE

Entre as ferramentas de Manufatura Digital utilizadas na Indústria Naval/*Offshore*, podem ser citadas: Promodel, Arena e ProEngineer, mais especificamente utilizadas na área naval; e o

conjunto de soluções PLM da Dassault Systems que serão utilizadas na validação deste trabalho.

**PROMODEL**[Benson 1997]

O Promodel Optimization Suite é uma ferramenta para modelagem de diversos tipos de sistemas de manufatura. Desenvolvido para a plataforma Windows, o sistema possui uma interface gráfica intuitiva e modelagem orientada a objeto, que reduzem a necessidade de programação.

O Promodel pode ser considerado um pacote de alto nível para simulações de eventos discretos, permitindo a modelagem de processos de atrasos, filas, ramificações, divisões, fundações e recursos do processo de manufatura.

Para situações especiais de modelagem, onde se necessite alguma programação extra, as características da linguagem de programação do Promodel permitem uma lógica simples de programação, através de comandos como "if-then-else", expressões booleanas, variáveis, conjuntos e até mesmo acesso a planilhas eletrônicas externas. Há também a possibilidade de se importar rotinas com lógicas mais complexas nas linguagens C, Pascal e Basic.

Como vantagens deste sistema podem-se citar o baixo custo de aquisição e a facilidade de utilização, devido à sua interface intuitiva (ver Figura 2.8 e 2.9). A saída do sistema se dá através de gráficos, documentos textos e planilhas eletrônicas, facilitando o acesso à informação e simplificando a visualização dos resultados.

Entretanto, comparado ao Arena, apresentado a seguir, a abordagem em camadas para o desenvolvimento de modelos torna a tarefa mais trabalhosa e menos intuitiva [Silva 2006].

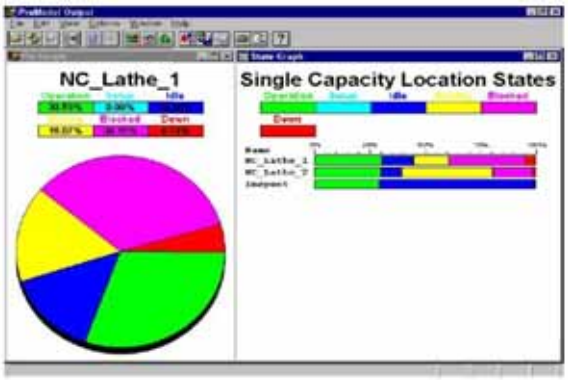



Figura 2.8. Editor gráfico do PROMODEL.

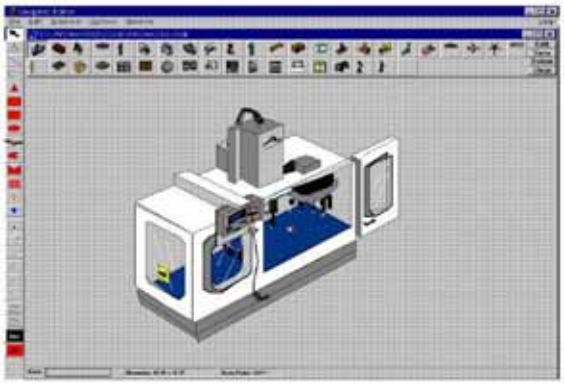


Figura 2.9. Saída do PROMODEL.

**ARENA** [Hammann 1995]

Arena é um *software* para simulação de eventos discretos que fornece um *framework* integrado para construção de modelos de simulações para uma ampla variedade de aplicações. O Arena possui ainda um conjunto de funções integradas para execução de simulações incluindo análise dos dados de entrada associada às características geométricas e funcionais do produto, construção do modelo, execução interativa, animação, verificação do modelo e análise de saída da simulação do produto.

O Arena faz uso de simulações para ampliar suas saídas de dados. Suas aplicações focam o projeto, as necessidades da manufatura, bem como suporte a decisões para outras áreas como: processos de negócios, reengenharia, sistemas médicos, transporte, logística e comunicação de dados.

Além disso, o Arena fornece ao usuário a capacidade para modelar o sistema para aplicações personalizadas de forma simples, sem limitar o usuário à construção de soluções restritas ao domínio, podendo este compartilhar parte das soluções entre uma ou mais aplicações. O sistema é hierárquico e orientado a objetos. Uma desvantagem do Arena em relação ao Promodel é que o Arena possui maior dificuldade para modelagem de funções de leitura e escrita. Um estudo comparativo entre as ferramentas Promodel e Arena pode ser observado no trabalho intitulado "Método para avaliação e seleção de softwares de simulação de eventos discretos aplicados à análise logística" [Silva 2006]. Embora o Promodel e o Arena sejam softwares voltados a simulação de eventos discretos ambos foram muito utilizados em parte do processo de manufatura de navios e plataformas podendo ser considerados como ferramentas de apoio ao processo de manufatura digital.

**PROENGINEER**

ProEngineer é um *software* comercial utilizado em empresas para o desenvolvimento de produtos. Esse sistema integra de forma associativa diversos módulos entre estes CAD/CAE/CAM. Além disso, permite colaboração entre dois ou mais usuários pela internet, elaboração de processos de montagem e manufatura de estruturas metálicas, tubulações, cabeamentos elétricos para citar alguns. É voltado principalmente para engenharia mecânica.

O ProEngineer, lançado em 1987, foi um dos primeiro *softwares* paramétricos comerciais e destacava-se pela facilidade de uso e velocidade na modelagem de sólidos. Além

disso, o ProEngineer permitia editar as propriedades dos modelos 3D através de parâmetros. Suas funcionalidades levaram os concorrentes a desenvolver recursos semelhantes e influenciaram de forma significativa na evolução dos softwares paramétricos.

Atualmente o ProEngineer baseia-se na utilização de protótipos digitais 3D para simulação de situações reais a fim de analisar o comportamento e prever falhas. O Pro-Engineer é considerado uma ferramenta DMU. Além de diminuir ou eliminar a necessidade de criação de protótipos reais de alto custo, através de recursos CAD/CAE/CAM integrados, o ProEngineer auxilia desde a etapa do projeto conceitual até a manufatura propriamente dita. Ou seja, atua nas etapas de análise, detalhamento e projeto, reduzindo tempo e custos de execução do projeto final.

Entre os módulos de desenvolvimento do ProEngineer têm-se as seguintes aplicações na etapa de projeto: superfícies complexas, chaparia, engenharia reversa, montagens. Na etapa de simulação tem-se simulação dinâmica, cinemática, análise térmica e estrutural, otimizações e análise de tolerâncias. Possui módulos para projetos de cabeamento e tubulações e nas etapas de produção permite usinagem prismática, multi-eixos, torneamento entre outros.

O ProEngineer permite ainda a exportação e a importação de dados nos formatos STEP/IGES/DXF/DWG/STL/VRML/ECAD, além de outros formatos para diversos *softwares* CADs. Basicamente o ProEngineer possui um desenvolvimento voltado à manufatura, com uma arquitetura aberta para interação com outros sistemas, engenharia digital, comunicação colaborativa e dados de custos intrínsecos ao projeto. Uma característica importante é versão *Wildfire* para Linux, sendo o primeiro sistema CAD para esse sistema operacional [Division Reality 2003].

Posterior e paralelamente ao ProEngineer surgem diversos softwares DMU voltados ao desenvolvimento de produtos e equipamentos industriais. Entretanto, seus custos ainda são elevados, o que dificulta a possibilidade de contribuições através de aplicações acadêmicas e estudos científicos, ocorrendo geralmente com o financiamento de instituições privadas que possuem interesses particulares.

## DELMIA, CATIA, ENOVIA

A Dassault Systèmes, uma das maiores empresas de desenvolvimento de sistemas de manufatura, surge com um pacote de ferramentas para atender as pressões pela diminuição de custo e do *time to market*. Essas questões ganham ainda mais força com o sucesso da *Boeing*

em reduzir o tempo de desenvolvimento do avião 777 através da utilização do CATIA, software CAD adotado pela empresa. Atualmente, diversas indústrias das áreas automotiva, aérea e naval utilizam essas ferramentas nas operações de seus processos. Embraer, GM, Ficantieri *Shipyard S.p.A, Samsung Heavy Industries* são alguns exemplos [Dassault 2005].

A Dassault Systèmes desenvolveu um conjunto de soluções que utilizam sistemas de dados baseados em modelos PPR (*Product, Process, Resource)* - Produto, Processo e Recurso). CATIA, DELMIA e ENOVIA são algumas das soluções desenvolvidas pela Dassault Systèmes que atuam de forma específica, podendo ser integradas para compartilhamento de informações.

O sistema CATIA (*Computer-Aided Three-dimensional Interactive Application)* - Aplicação Interativa Tridimensional Auxiliada por Computador) é uma ferramenta desenvolvida para definição e simulação do produto digital. Com esta ferramenta é possível simular a escala inteira de processos do projeto industrial desde o *marketing* até a análise e manutenção do produto final. Essa ferramenta contempla tecnologias CAD/CAM e permite a integração de profissionais de projeto, *marketing*, vendas, manufatura, fornecedores e clientes através de um banco de dados compartilhado e possui fácil integração com ENOVIA e DELMIA. O CATIA permite exportar modelos em formatos: WRL/STP/IGS/3DXML/CGR/EXE/HSF/STBOM/TXT. Seu formato nativo é CATProduct.

O DELMIA (*Digital Enterprise Lean Manufacturing Interactive Application* - Aplicação Interativa da Manufatura para Empresa Digital) permite aos engenheiros de controle definir, controlar e monitorar digitalmente sistemas automatizados em um ambiente virtual 3D. O sistema DELMIA valida e melhora o comportamento interno dos sistemas de controle.

O DELMIA pode ser utilizado também como ambiente de programação dedicado para PLCs (Controladores Lógicos Programáveis) que utilizam linguagens definidas no padrão IEC61131-3 ou implantadas como solução de automação completa, integrando dados de projeto mecânico, elétrico e de controle em um ambiente 3D. O DELMIA oferece as ferramentas necessárias para criar, editar, depurar e validar um programa PLC e permite aos usuários criar e reutilizar blocos lógicos de controle predefinidos.

Os *plugins* PLC para DELMIA *Automation* são desenvolvidos em estreita colaboração com os fornecedores de PLC como SIEMENS AD *Automation Systems*, *Schneider Electric* e OMRON *Corporation*.

ENOVIA (*Enterprise InNOvation VIA)* é uma ferramenta utilizada para a sustentação de decisão e a gerência do ciclo de vida do produto. Integrado com CATIA e DELMIA, o

ENOVIA permite aos usuários capturar, controlar, e explorar a propriedade intelectual criada com o desenvolvimento do produto. Fornece os espaços de trabalho 3D colaborativos que conectam fornecedores e clientes através da Internet.

Em suma, o CATIA permite desenvolver o projeto do produto, o DELMIA oferece soluções na automação e engenharia da manufatura e o ENOVIA fornece suporte a ferramentas de decisão através da extração de informações com a integração dos demais sistemas. Logo, estas ferramentas contemplam características de integração e automatização essenciais aos processos de manufatura. A Figura 2.10 mostra o ambiente de trabalho do DELMIA.

Figura 2.10. Um dos módulos da ferramenta DELMIA.

Diversos sistemas de Manufatura Digital, entre esses os desenvolvidos pela Dassault Systemes, dão suporte a dispositivos de Realidade Virtual como óculos e luvas por exemplo [Delmia 2002]. Entretanto ainda não é possível a partir desses programas gerar automaticamente multiprojeções destes ambientes de forma que o usuário sinta-se imerso na planta de produção.

O próximo capítulo irá apresentar alguns conceitos e aplicações da Realidade Virtual, suas vantagens, desvantagens, e a justificativa de sua utilização como ferramenta de apoio aos sistemas de Manufatura Digital tais como o DELMIA, entre outros.

# 3 A RV NA CONSTRUÇÃO NAVAL E OFFSHORE

*Conceituar a Realidade Virtual bem como suas aplicações e ferramentas na Indústria Naval/Offshore.*

Recentemente, diferentes tecnologias associadas aos processos produtivos começam a fazer uso de técnicas de Realidade Virtual, integradas às funcionalidades dos sistemas de manufatura digital. Esses sistemas passam a utilizar dispositivos de RV para interação de usuários e máquinas durante o treinamento de pessoal, manutenção de equipamentos e previsibilidade de situações. Também simulações tridimensionais em ambientes não-imersivos tornam-se comuns em sistemas de manufatura digital. Para melhor compreensão do significado do termo Realidade Virtual - RV, realiza-se neste capítulo uma contextualização histórica e conceitual sobre o assunto.

## 3.1 CONTEXTUALIZAÇÃO DA REALIDADE VIRTUAL

O termo Realidade Virtual (RV) surgiu nos anos 80 através do fundador da VPL. *Research Inc.*, Jaron Lanier, para diferenciar simulações tradicionais feitas por computador, de simulações em ambientes compartilhados por múltiplos usuários [Araújo 1996]. Desde então, surgiram diversas definições para o termo. Segundo Aukstakalnis e Blatner, a RV é uma forma das pessoas visualizarem, manipularem e interagirem com computadores e dados extremamente complexos [Aukstakalnis 1992]. Por outro lado, [Latta 1994] citam Realidade Virtual como uma avançada interface homem-máquina que simula um ambiente realístico e permite que participantes interajam com ele "*Virtual Reality involves the creation and experience of environments*".

Já [Machover 1994] afirmam que a qualidade da experiência em RV é crucial, pois deve estimular o usuário ao máximo e de forma criativa e produtiva - "...a realidade precisa reagir de forma coerente aos movimentos do participante, tornando a experiência consistente".

[Pimentel 1995] define RV como o uso da alta tecnologia para convencer o usuário de que ele está em outra realidade - "...um novo meio de estar e tocar em informações", "*Virtual Reality is the place where humans and computers make contact*".

Apesar da diversidade de conceitos sobre Realidade Virtual, a maioria recai em algumas das características de sua essência básica: imersão, interatividade e envolvimento. A imersão pode ser definida como o sentimento de participar do ambiente tridimensional, estando incluso no mesmo. A sensação de imersão completa pode ser obtida com a utilização de dispositivos tais como: HMD (*Head Mounted Display* - capacetes), óculos tridimensionais e CAVEs (*Cave Automatic Virtual Environment*) ou salas de projeção [Kirner 2004].

A interação é a capacidade de o usuário receber a resposta do sistema após realizar alguma ação sobre o ambiente, ou seja, é a resposta do sistema à ação do usuário (capacidade reativa do ambiente). Sensações táteis, de olfato e som aumentam a sensação de imersividade e interatividade [Kirner 2004].

O envolvimento é medido pelo nível de motivação do usuário em realizar alguma tarefa. Tal fato depende muitas vezes do grau de entendimento em relação ao sistema ao qual a pessoa está operando e da facilidade de navegar e interagir no mesmo.

A tridimensionalidade do ambiente fornece uma visão mais realista de profundidade e volume. Tais sensações visuais são obtidas com a utilização de técnicas de estereoscopia por computador, que por sua vez baseiam-se na estereoscopia visual humana.

A estereoscopia visual advém do fato de que cada um dos olhos humanos enxerga uma imagem diferente da outra. Essas diferentes imagens são processadas pelo cérebro, fornecendo a idéia de profundidade e consequentemente a sensação de imersão em ambientes, com elementos em diversas posições.

Diferentemente da Computação Gráfica em que as imagens são mono (geradas a partir de uma única câmera virtual - volume de visualização definido a partir de um único ponto), em Realidade Virtual é comum a utilização de imagem estéreo. A imagem estéreo é formada a partir de duas câmeras virtuais separadas por uma distância determinada e sua qualidade depende diretamente dos equipamentos de visualização utilizados. Logo, exibir imagens separadas para o olho esquerdo e direito exige do *hardware* o dobro de poder de processamento de imagens. Essas imagens podem ser visualizadas em monitores ou projetadas em *n* planos.

Os equipamentos utilizados para estereoscopia são classificados como estéreo ativo e estéreo passivo. Na maioria das aplicações utilizam-se óculos para estereoscopia passiva. Tais

óculos funcionam como filtros e as imagens são mostradas simultaneamente. Um exemplo deste tipo de aplicação estéreo é o anáglifo. Nele, a filtragem é feita por cores, geralmente vermelho e verde, sendo que a qualidade da imagem é prejudicada devido à coloração. Entretanto, esse tipo de estéreo funciona com apenas um monitor e o custo dos óculos necessários é baixo.

A tecnologia atual torna possível a visualização de imagens estereoscópicas sem o uso de óculos, através da utilização de displays de cristal líquido, como, por exemplo, o DTI (*Dimension Technologies* [Dimension 2004]. Porém essas soluções ainda possuem a característica de serem monousuárias.

No estéreo ativo os óculos são sincronizados com um monitor de alta freqüência e são formados por duas lentes capazes de impedir de forma alternada a visão de um dos olhos. Ou seja, quando a visão de um dos olhos é aberta a outra é bloqueada, sincronamente. Uma desvantagem desses sistemas é que, sendo 60Hz a freqüência mínima para exibição de imagens, e devendo-se projetar simultaneamente as duas imagens, torna-se necessária a utilização de projetores de no mínimo 120Hz. Esses projetores ainda apresentam custo elevado, porém oferecem suporte a multiusuários e alta qualidade de projeção [McAllister 2002].

Outros dispositivos de Realidade Virtual importantes são os que possibilitam manipular objetos e navegar pelo ambiente virtual. Esses dispositivos podem ser classificados em duas categorias principais: dispositivos de entrada de dados e dispositivos de saída de dados. Conforme [Pimentel 1995] os dispositivos de entrada de dados podem ser classificados em dois tipos: dispositivos de interação que dão suporte ao movimento e manipulação de objetos e dispositivos de trajetória que servem para monitorar os movimentos do usuário.

Os dispositivos de entrada de dados mais comuns em sistemas de Realidade Virtual são: *mouses*, teclados e *joysticks*. Esses dispositivos possuem dois graus de liberdade (2DOF - *Degree Of Freedom)* e agem de forma simples e direta no ambiente virtual, além possuírem a vantagem da fácil utilização e baixo custo de implementação. Luvas, rastreadores, dispositivos de trajetória, sensores de comando de voz também são considerados dispositivos de entrada de dados.

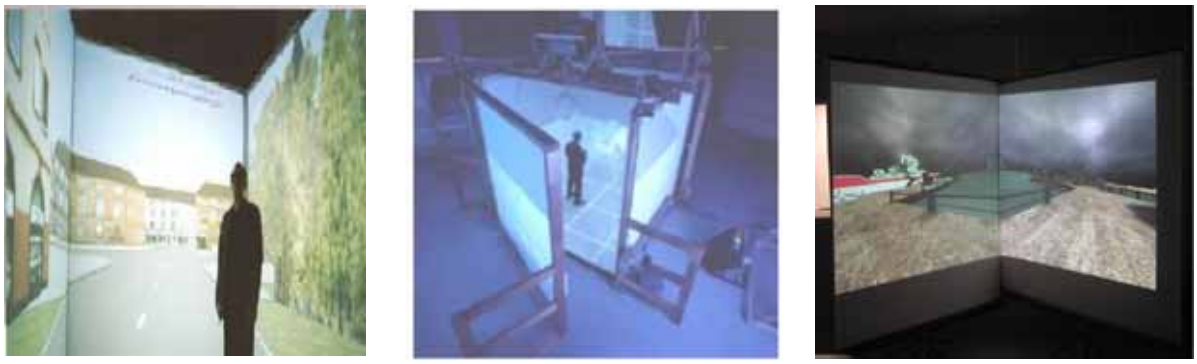
Os dispositivos de saída de dados podem ser classificados de acordo com o grau de imersão fornecida ao usuário, sendo classificados como não imersivos e imersivos. Sistemas baseados em monitores comuns (*desktops*) são considerados não imersivos por limitarem a visualização do usuário ao tamanho da tela de projeção, podendo ou não utilizar óculos de estereoscopia.

Sistemas considerados imersivos utilizam dispositivos tipo óculos, capacetes e cavernas virtuais (CAVEs). Este último, por propiciar alto grau de imersividade e capacidade de suportar os demais equipamentos de interatividade, passa a ser enfocado ao longo deste trabalho, sendo descrito de forma mais precisa a seguir.

## DISPOSITIVOS DE MULTIPROJEÇÃO

O termo CAVE (*Cave Automatic Virtual Environment*) embora tenha sido originado de um trabalho particular [CruzNeira 1992] popularizou-se de tal forma que desde então a grande maioria dos dispositivos de multiprojeção ou caverna digital passaram a ser nomeados como CAVE. CAVEs são salas ou paredes de projeção que possuem a grande vantagem de fornecer ao usuário a sensação de imersão sem a necessidade de utilização de qualquer outro dispositivo, bastando para isso apenas estar dentro ou perante a projeção.

Em conjunto com a estrutura física da CAVE podem ser utilizados outros equipamentos e sensores, com vista a melhorar o resultado final da percepção, no que tange ao sentimento de imersão. Dispositivos que propiciam uma maior interatividade com o usuário, tais como óculos de estereoscopia e capacetes, são exemplos desse tipo de equipamentos. Algumas configurações de CAVEs podem ser vistas na Figura 3.1



a) CAVE USP b) Estrutura física de CAVE. c) CAVE FURG.

Figura 3.1. Exemplos de CAVES.

Entre as vantagens deste tipo de dispositivo podem-se citar a possível alta resolução e a ampla cobertura do campo de visão. Entretanto, os custos de implantação tradicionais ainda são altos.

Sob o ponto de vista do seu funcionamento, são necessários cuidados em relação à distância entre os projetores e as telas, que devem ser iguais. Também a altura dos projetores em relação ao piso, e os ângulos de projeção, devem ser cuidadosamente analisados. Afora isso, há que se perceber que a sensação de profundidade nas CAVEs é proporcionada por dispositivos que utilizem estereoscopia ativa, tais como óculos estereoscópicos.

A calibração eletrônica dos projetores é outra tarefa que demanda esforço e tempo, pois existe uma grande possibilidade de ajustes para que as imagens possuam as dimensões exatas das faces de projeção, estejam totalmente alinhadas e apresentem cores compatíveis.

Segundo [Zuffo 2001] dois focos de pesquisa devem ser considerados na direção do uso mais disseminado deste tipo de tecnologia: o desenvolvimento de sistemas de autoestereoscopia, eliminando-se assim o único elemento invasivo ao usuário que são os óculos; e a minimização dos custos associados.

A partir do grau de imersão, interatividade e envolvimento, bem como da natureza dos ambientes tridimensionais a serem visualizados, pode-se classificar diferentes categorias de Sistemas de Realidade Virtual, descritos a seguir.

**Realidade Virtual de Simulação**

A Realidade Virtual de Simulação foi o primeiro tipo de sistema de RV, surgindo em meados da década de 60 em simuladores de vôo. Desde então as demais formas de Realidade Virtual podem ser vistas como um aperfeiçoamento da RV de simulação. Este tipo de sistema consiste em um usuário, uma interface homem-máquina e um computador. O usuário utiliza esta interface para interagir com o ambiente virtual através de sinais de controle e percepção.

**Realidade Virtual de Projeção**

A Realidade Virtual de Projeção não necessita que o usuário utilize equipamentos e dispositivos de entrada de dados. Nesse tipo de sistema o usuário visualiza o ambiente virtual através de monitores ou paredes de projeção. A partir de uma disposição adequada, quanto maior o número de paredes de projeção maior a sensação de imersão para o usuário. CAVEs são exemplos de tais dispositivos. Em sistemas de projeção a navegação pode ser controlada ou não pelo usuário.

**Realidade Virtual de Telepresença**

Em sistemas de telepresença, o usuário, localizado em um ambiente virtual remoto, tem a sensação de estar fisicamente presente no ambiente real. A telepresença é implementada por mecanismos de teleoperação que irão atuar sobre um robô comandado de forma remota pelo usuário, denominado telerobô. A diferença entre sistemas de telepresença e de simulação se dá no modo de atuação sobre o ambiente. Em um ambiente de simulação a interface atua sobre o computador, em um sistema de telepresença a interface atua sobre o dispositivo que irá atuar sobre o mundo real. Alguns dispositivos utilizados neste tipo de sistemas são: robôs, câmeras de vídeo e luvas. Com esse tipo de solução o usuário pode realizar a manutenção de uma máquina manipulando a chave de ajuste à distância, por exemplo.

**Realidade Aumentada (AR)**

Um sistema de Realidade Aumentada (AR) - *Augmented Reality* combina imagens geradas no mundo virtual com imagens do mundo real. Estes sistemas buscam integrar as informações obtidas no mundo real com as informações do mundo virtual geradas pelo computador, dando a impressão de que o mundo real e o virtual ocupam o mesmo espaço tridimensional. Nos ambientes de Realidade Aumentada predominam situações em que objetos reais são exibidos com alguns objetos virtuais

Afirma-se que a AR é uma combinação dos sistemas de Realidade Virtual de simulação com telepresença. A AR utiliza *displays* transparentes, denominados *heads-up-displays* (HUDs), fixos à cabeça, onde gráficos 3D, animações e diagramas são projetados, a fim de sobrepor as informações do mundo real que estão sendo visualizadas juntamente com as informações geradas pelo computador. É possível, por exemplo, obter a imagem através de uma câmera de vídeo e sobrepor esta com a imagem virtual gerada pelo computador.

Um dos desafios deste tipo de sistema é a superposição exata do mundo real com o virtual. Como o nome sugere, AR aumenta o nível de percepção do usuário e a sua interação com o mundo real, provendo informações que este por si só não é capaz de perceber apenas com seus próprios sentidos [Bastos 2005].

**Realidade Mista**

A Realidade Mista (*Mixed Reality*) é um conceito mais amplo, que abrange a Realidade Aumentada (AR) e Virtualidade Aumentada. A Virtualidade Aumentada refere-se a ambientes formados por objetos virtuais sendo estes exibidos juntamente com alguns objetos reais onde quase todo o ambiente é virtual.

A Realidade Mista é também definida como sendo uma tecnologia que mistura o mundo virtual como, por exemplo, gráficos e/ou imagens geradas por computador, e o mundo real tão naturalmente quanto possível, em tempo real [Bastos 2005]. Em suma, na Realidade Aumentada é predominante a ocorrência de objetos reais com alguns objetos virtuais. Na Virtualidade Aumentada é o contrário: predominam objetos virtuais com alguns objetos reais inseridos. Como exemplo de aplicação pode-se citar a utilização desta tecnologia para a manutenção de equipamentos. Neste, o operador recebe informações e esquemas de montagem diretamente projetados sobre o equipamento a ser reparado [Sherman 2003].

**Realidade Melhorada**

A Realidade Melhorada utiliza técnicas de processamento digital de imagens (PDI) e de Computação Gráfica para melhorar a imagem em vídeo. Nessa técnica são geradas informações adicionais que serão sobrepostas à imagem real. A Realidade Melhorada pode ser considerada uma variação da Realidade Aumentada, onde técnicas de processamento de imagens são aplicadas, a fim de melhorar a qualidade da imagem ou até mesmo modificar posições espaciais dos objetos, aplicando transformações e anotações sobre a imagem [Bowskill 1995].

## 3.2 SISTEMAS COMPUTACIONAIS PARA A RV

Existem diversos tipos de sistemas computacionais que implementam soluções em RV. Essas ferramentas permitem o interfaceamento com diferentes dispositivos de Realidade Virtual, tais como luvas, óculos e multiprojeções. Na Tabela 3.1 são apresentadas, em resumo, algumas destas ferramentas.

Tabela 3.1. Ferramentas gratuitas e comerciais para aplicações de RV.

| Ferramenta | Empresa | Plataforma | Custo | Características |
|---|---|---|---|---|
| REND386 | | Windows | Gratuito | Software de modelagem e visualização que possui a capacidade de definir cenários, atribuir cores, suportando navegação em tempo real. Entretanto necessita de programadores para operar o sistema. Apresenta altas taxas de frames por segundo devido à utilização de linguagens de baixo nível (Assembler) em grande parte do seu código, reduzindo sua portabilidade. |
| VR386 | | DOS | Gratuito | Uma das primeiras ferramentas de aplicação de RV, possuindo também parte de seu código escrito em linguagem assembler. |
| AVRIL | | Windows | Gratuito | Rápido, portátil e fácil usar. Escrito em ANSI C com algumas rotinas em linguagem Assembler para a versão PC. Foi projetado para ser compatível com a engine de renderização REND386. |
| MRToolKit | | Windows | Gratuito | MRToolKit [Toolkit, 2006] é um conjunto de ferramentas de software para a produção de sistemas de Realidade Virtual e interfaces tridimensionais. É composto por um conjunto de bibliotecas, acionadores de dispositivos, programas de apoio e uma linguagem de descrição de geometria e comportamento. |
| CDK2 | Autodesk | DOS/Windows | Não gratuito | *Cyberspace Development Kit é* um conjunto de bibliotecas para programadores C++ que permite *interface* com os dispositivos mais comuns de RV e geração de imagens em estéreo. Reconhece arquivos com extensão DXF (modelados em CAD) |
| dVS/DVISE | Division | SGI,Windows | Não gratuito | Conjunto de ferramentas 3D, multiusuário, com suporte à textura em ambiente colaborativo. |
| WorldToolKit | Sense8 | SGI, Windows, Sun, HP, DEC | Não gratuito | Biblioteca gráfica em C que oferece suporte a textura, variedade de hardware, vídeo em tempo real e som interativo. Porém, exige sólidos conhecimentos de programação. |

| | | | | |
|---|---|---|---|---|
| Superspace | Superspace | DOS | Não gratuito | Multiusuário. Suporta capacete, joysticks e textura. |
| VRE4AM | VREAM | DOS | Não gratuito | Suporta capacete, joysticks, textura, rede, multiusuário. Não necessita de programadores para operar o sistema. Possui capacidade de criar, mudar e apagar objetos. Apresenta suporte a som interativo e imagens estéreo. Permite navegação em tempo real. |
| GVS | Gemini Tech Corp. | UNIX | Não gratuito | Suporta capacete, textura, interação, rede, e é multiusuário. |
| 3D Interaction Acelerator | IBM Corp | UNIX | Não gratuito | Suporta capacete, textura e é monousuário. |
| Veja VR | Paradigm Simulation Inc | SGI | Não gratuito | Suporta capacete, textura e é multiusuário. |
| Performer | Silicon Graphics | SGI | Não gratuito | Textura, simulação, multiusuário. |

Apesar da gama de ferramentas existentes, apresentadas na Tabela 3.1., não existe uma padronização da funcionalidade das mesmas. Ou seja, algumas possuem suporte a determinadas funções e outras não, necessitando que adequações sejam realizadas a cada caso de aplicação ou mais de uma ferramenta seja utilizada para um determinado fim.

Logo, os diferentes recursos oferecidos por diferentes sistemas implicam que o desenvolvimento de uma solução que possua recursos de Realidade Virtual passe algumas vezes pela utilização de um conjunto de ferramentas ou até mesmo pela programação em baixo nível de códigos que contemplem as características necessárias à aplicação. É o que acontece com muitos dos sistemas de manufatura que utilizam RV em seus processos, como será observado na Sessão 3.4.

## 3.3 APLICAÇÕES DA RV NA INDÚSTRIA

Atualmente, com os avanços das tecnologias de *hardware* e *software*, cresce o número de investimentos em projetos de sistemas que utilizam RV, principalmente na indústria. Ao longo de todo o processo de manufatura podem-se citar diferentes situações onde a RV pode ser aplicada, englobando desde o auxilio na avaliação de fatores ergonométricos, passando

pelo estudo de técnicas de engenharia, planejamento, na simulação de montagens, estruturas e processos, até no treinamento e na manutenção.

Com a utilização de visualizações realísticas possíveis problemas de montagem e de manutenção podem ser previamente revelados durante a simulação, na etapa de projeto e de modelagem dos processos.

A Realidade Aumentada e os protótipos digitais podem ainda reduzir o tempo de execução de tarefas de treinamento e ainda aumentar a produtividade do processo de construção.

Outra aplicação da RV na indústria é o auxílio à coordenação e cooperação entre diferentes equipes de engenharia que atuam em diferentes etapas do processo, servindo como uma ferramenta de padronização e comunicação entre as partes. A Figura 3.2 (*a*, *b*, *c*, *d*, *e* e *f*) ilustra algumas aplicações industriais da Realidade Virtual.

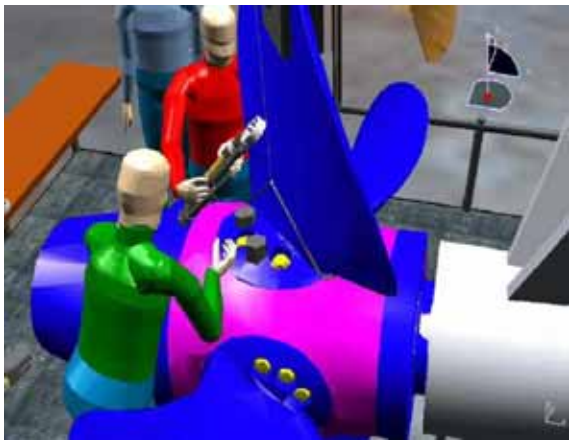
a) AR em projeto.

b) Sistema colaborativo para conferência

c) Revisão interativa de projeto.

d) Simulação ergonômica

e) Revisão e interação com protótipo digital.



f) Simulação de treinamento para manutenção de equipamento.

Figura 3.2 Aplicações industriais da Realidade Virtual (*a, b, c, d, e, f*).

Enfim, técnicas de RV começam a ser aplicadas em grande parte das etapas do processo de produção industrial, para melhorar a qualidade, preço e prazo de entrega do produto final.

## 3.4 RV APLICADA A CONSTRUÇÃO NAVAL/OFFSHORE

Uma aplicação interessante da RV na indústria naval e *offshore* é a utilização de ambientes de multiprojeção do tipo CAVEs, para visualização distribuída ou seja visualização em *n* planos das plantas de manufatura. Entretanto, o processo de visualização distribuída de modelos e simulações de plantas na manufatura naval e *offshore* abrange um conjunto de desafios e questões a serem tratados. Da concepção do modelo virtual da planta e sua simulação, até o efetivo uso da visualização distribuída imersiva, identificam-se, neste trabalho, as seguintes etapas associadas:

i. **Modelagem:** Nessa etapa é obtido o cenário ou modelo a ser visualizado. Atualmente, ferramentas DMU, como descrito na *Sessão 2.4.*, estão sendo utilizadas para modelagem, integrando tecnologias CAD/CAE/CAM. Entretanto, tais ferramentas focalizam-se na modelagem geométrica, necessitando que os cenários por estas, gerados, sejam tratados para que suas visualizações possam ser realizadas de forma imersiva e interativa.

ii. **Conversão:** Uma etapa de conversão se faz necessária, de modo a possibilitar a transformação do cenário adquirido através da modelagem fornecida pela ferramenta de Manufatura Digital, aqui denominado ***Modelo CAD***, em um modelo capaz de propiciar uma visualização imersiva e interativa, denominado ***Modelo-RV Imersivo***. Esse novo termo refere-se a um conjunto de atributos, entidades e comportamentos, que irão compor o cenário dinâmico a ser visualizado.

iii. **Visualização:** Com o objetivo de propiciar uma visualização mais realística, em termos de imersão, se estabelece a utilização de visualização distribuída em *n* planos. Assim, nesta etapa são encontrados basicamente dois desafios: a renderização e a multiprojeção em tempo real.

iv. **Interatividade:** Os sistemas de visualização distribuída com dispositivos de RV devem fornecer suporte a diferentes graus de interatividade. No caso de cenários associados a plantas de manufatura naval e offshore, o tratamento de colisões, o uso de dispositivos de RV, a percepção das características do ambiente através de texturas e sons, a possibilidade de visualização multi-usuária, são fatores importantes e devem estar previstos.

O estudo e desenvolvimento de técnicas de modelagem e simulação de processos de produção vêm sendo alvo de pesquisa em diferentes áreas associadas à engenharia. No Capítulo 2 foram apresentadas diferentes ferramentas DMU que implementam recursos de Computação Gráfica para a modelagem e simulação de produtos e processos. Tais sistemas apresentam diferentes níveis de qualidade em termos de interface de visualização.

Entretanto, as quatro etapas descritas acima não são tratadas por nenhuma ferramenta única, sendo escassas soluções que, a partir da modelagem, forneçam o sistema visual imersivo final distribuído. Por outro lado, muitos são os estudos e técnicas associadas a cada questão de forma individual, resultando em um conjunto de ferramentas, algumas das quais descritas a seguir.

**Ferramentas**

Mesmo voltadas para cenários estáticos, ou seja, sem componentes animados, algumas ferramentas especializaram-se na conversão de modelos CAD para cenários a serem visualizados por ferramentas de RV. WalkInside [Walkinside 2006] e Division Reality

[Division Reality 2003] são algumas destas soluções comerciais. WalkInside oferece suporte à detecção de colisões e simulações de gravidade, possuindo melhores taxas de *frames* por segundo (fps) do que o Division Reality. Este último pode ser utilizado para ambientes imersivos tipo CAVE, suportando a manipulação direta de objetos. As Figuras 3.3 e 3.4 ilustram a utilização dessas ferramentas na indústria naval.

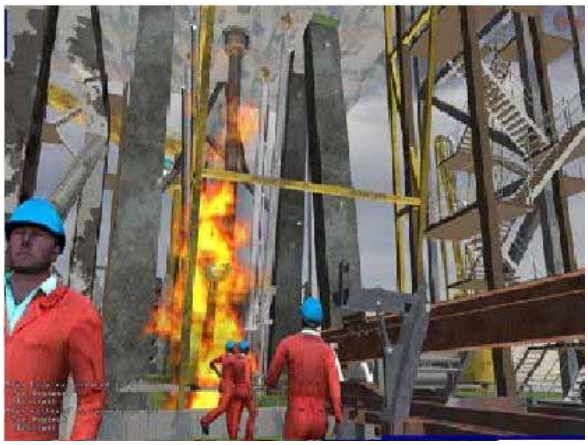


Figura 3.3 Walkinside.

Figura 3.4 Division Reality.

O GigaWalk [BaxterIII 2002], como solução acadêmica, possui técnicas de reorganização de objetos, descarte por oclusão e detalhamento de níveis hierárquicos. Entretanto, o modelo a ser tratado deve ser pré-processado, o que o torna inviável em termos de tempo de execução para plantas de alta complexidade.

Entre os sistemas desenvolvidos para visualização de modelos CAD pode-se citar também o ENVIRON [Corseuil 2004], mostrado na Figura 3.5, que associado a uma ferramenta para aplicações de Realidade Virtual permite a visualização de modelos CAD em ambientes de RV de projetos da Indústria *Offshore*.

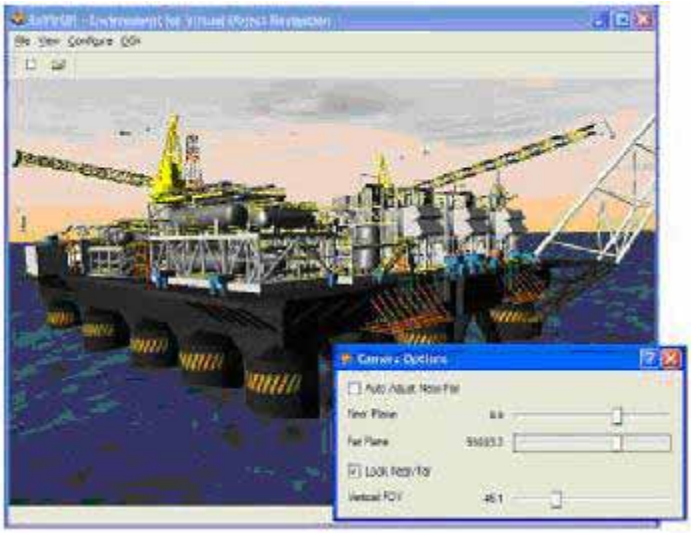


Figura 3.5. Tela Environ.

O ENVIRON é um sistema que combina ferramentas de RV com *plugins* desenvolvidos sobre soluções comercias de modelagem e simulação gráfica. As ferramentas utilizadas pelo ENVIRON são: VIRAL[Bastos et al. 2004] e OpenSceneGraph [Osfield 2003].

VRJuggler[Biernaum 2000] e VIRAL são exemplos de ferramentas desenvolvidas para aplicações de Realidade Virtual.

O VRJuggler possui código aberto, é multi-plataforma e multi-dispositivos. Entretanto, grande parte de sua configuração é feita de forma manual e através da edição de arquivos. Também tal ferramenta depende de muitas bibliotecas externas, dificultando sua compilação. Uma aplicação desenvolvida com o VRJuggler é completamente imersiva, ou seja, não é possível utilizar ao mesmo tempo o VRJuggler e um sistema de diálogos convencional (menus, listas, botões). A reconfiguração em tempo de execução é limitada.

O VIRAL é uma ferramenta desenvolvida em C++, projetada para facilitar as aplicações de Realidade Virtual, equivalente ao VRJuggler, com a diferença de possuir uma interface gráfica em QT [Silva 2004], o que facilita o desenvolvimento das aplicações. O VIRAL é totalmente orientado a objetos e é independente do dispositivo. Ou seja, a aplicação não precisa conhecer quantas janelas, usuários ou dispositivo serão utilizados.

Outra ferramenta de RV utilizada na Manufatura Naval é o OpenSceneGraph. Esse sistema fornece um controle de cena mais eficiente que o OpenGL, melhorando a *performance* de renderização, além de possuir a vantagem de possuir código aberto.

Além disso, como mencionado anteriormente, a obtenção de um ambiente virtual adequado passa pela definição dos atributos relacionados às suas entidades, bem como a redução de complexidade da sua representação. Entre as ferramentas conhecidas para essas finalidades pode-se citar World-Up [Sense8 1999] e 3DStudioMax [3DStudioMax 2004]. Ambas reconhecem diversos formatos de arquivos, têm suporte a periféricos 3D e incluem o paradigma da orientação a objeto.

Mesmo que ainda não utilizada na manufatura naval/*offshore*, outra possibilidade relacionada a sistemas para visualização de cenários virtuais é a utilização de *Game Engines*. Desses, podem-se citar o *Unreal Tournament* [Unreal 2004] e o Quake [Quake 1997] que além de possuírem código parcialmente aberto para modificações, fornecem cenários com alto nível de detalhamento, alta velocidade de processamento e robusta transmissão em rede, suportando visualização distribuída. Tais características conduziram a utilização de *Game Engines* em diferentes propostas e trabalhos acadêmicos e comerciais [Bylund 2002] [Piekarski 2002] [Laird 2001] [Carpin 2005] [Carpin 2006] [Kaminka 2002]. Alguns trabalhos encontram-se resumidos na Tabela 3.2. extraída do artigo "*Virtual Office Walkthrough Using a 3D Game Engine*}" [Shiratuddin 2002].

Tabela 3.2. Desenvolvedores e seus projetos que utilizam Game Engines.

| Ano | Projeto | Engine | Desenvolvedor | Descrições |
|---|---|---|---|---|
| **1998** | Virtual Florida Everglades National Park | **Unreal** | Project leader: Victor DeLeon | Um projeto educacional e para promover consciência ecológica |
| **1998** | Notre Dame Cathedral of France | **Unreal** | Digital Studio | Fundada pela UNESCO. Versão demonstrativa pode ser baixada em http://www.vrndproject.com. |
| **1999** | Long Island Technology Center | **Unreal** | Perillith Industrielle for Rudin Management | Versão demonstrativa pode ser baixada em http://www.unrealty.net. |
| **1999** | Heartland Business Center | **Unreal** | Perillith Industrielle | Um complexo escritório em New York. |
| **1999** | HypoVereins Bank | **Unreal** | Perillith Industrielle for Turbo D3 | Banco virtual na Alemanha. Versão demonstrativa pode ser baixada em http://www.unrealty.net. |
| **2000** | Virtual Graz of Austria | **Unreal** | Bongfish | Graz é a segunda maior cidade da Austria. Fundada pela UNESCO. |
| **2000** | Virtual International Space Station - VISS | **Unreal** | NASA Langley Research Center Spacecraft & Sensors Branch An International Virtual Space Station | Versão demonstrativa pode ser baixada em http://www.unrealty.net. |
| **2000** | Cambridge University and Microsoft Science and Technology site in West Cambridge | **Quake 2** | Martin Centre for Architectural and Urban Studies, Cambridge University | Parte do projeto utiliza comunicação eletrônica entre construtores e arquitetos. |
| **2000** | CAVE Quake 3 | **Quake 3** | Quake 3 Visualization and Virtual Environments Group, NCSA. | Um sistema CAVE baseado na engine Quake 3 Arena. Web-site: http://www.visbox.com/cq3a/. |
| **2001** | CAVE UT | **Unreal Tournament** | Medical Virtual Reality Center, Department of Otolaryngology, University of Pittsburgh | Um sistema CAVE baseado na engine Unreal Tournament. Web-site: http://www2.sis.pitt.edu/~jacobson/ut/CaveUT.html. |

## Desafios do uso de Visualização Imersiva na Construção Naval e *Offshore*

Como visto, diferentes ferramentas atuais relativas à Manufatura Digital podem modelar cenários associados à Construção Naval e *Offshore*. Tais modelos representam desde simples estruturas 2D até complexos cenários com características tridimensionais.

As plantas de manufatura Naval/*Offshore* podem incorporar desde detalhados arquivos CAD resultantes do projeto mecânico de seus produtos, até esquemas de funcionamento de

máquinas e animações de operários durante o processo de montagem de navios e plataformas. As simulações geradas podem representar em 3D a dinâmica associada aos diferentes componentes do processo de produção de navios e plataformas. Entretanto, mesmo que algumas destas ferramentas, utilizando tecnologias DMU, forneçam recursos para a geração de cenários 3D, principalmente associadas a projetos CAD, eles são utilizados para fins funcionais e de projeto, enquanto os modelos virtuais preocupam-se com aspectos de visualização, visando fornecer realismo, imersão e interatividade ao usuário.

As pesquisas para a transcrição direta de modelos CAD em modelos RV são temas contínuos de estudos. As soluções desenvolvidas são geralmente fechadas a aplicações comerciais e tratam modelos que são gerados em ferramentas específicas destas indústrias [Corseuil 2003]. Mesmo com diversas pesquisas na área, a falta de um padrão CAD-RV imersivo torna necessário o desenvolvimento de soluções parciais que nem sempre agem sobre os principais problemas inerentes ao tema.

Em muitas destas ferramentas, os componentes básicos de ambientes virtuais, como as entidades que os compõem, sua descrição geométrica, aparência e comportamento, não são claramente definidos. Muitos dos sistemas de manufatura permitem apenas a exportação de cenários em formatos simples de descrição de vídeo como AVI e MPEG, outros de forma mais elaborada, através de linguagens de modelagem de ambientes virtuais como o VRML (*Virtual Reality Modeling Language*) [Vrml 2000].

Como o intuito principal é a análise funcional dos produtos e processos de produção da indústria Naval/*Offshore*, na maioria dos casos os modelos exportados possuem informações desnecessárias para a utilização em RV. Por outro lado, carecem de recursos fundamentais para a visualização, como texturas e simplificações de superfície.

Frente ao extenso conjunto de formatos para exportação de modelos e simulações existentes, e as necessidades peculiares associadas à RV, a definição das melhores opções que propiciem efetivamente uma visualização imersiva distribuída de cenários de plantas Navais/*Offshore* ainda são um desafio.

Assim, busca-se na etapa de conversão a definição do conjunto de atributos associados ao modelo virtual e suas entidades a serem fornecidos a API (*Application Programming Interface*). Tal definição pode ser estática (*offline*) ou aplicada em tempo real de visualização. De forma mais precisa, nessa etapa é realizada a aplicação de texturas, bem como a obtenção de comportamentos característicos das entidades componentes do cenário, contribuindo para a obtenção das animações associadas à dinâmica destas entidades (operários, guindastes, veículos em movimento para citar alguns).

Também na etapa de conversão ocorre a redução de complexidade das informações que constituem o cenário virtual obtido na etapa de modelagem. Tal questão, já relevante no caso genérico de aplicações em RV, assume maior importância quando relacionada à manufatura digital de indústrias navais/*offshore*, tendo em vista que as ferramentas de manufatura digital utilizadas não são desenvolvidas com o objetivo de concepção de cenários virtuais e sua visualização imersiva. A diversidade de formatos exportados e o número de estruturas geométricas associadas às diferentes entidades do cenário Naval/*Offshore* podem tornar proibitiva a renderização das cenas a serem visualizadas.

No que tange a visualização, aspectos referentes à renderização em tempo real e multiprojeção em *n* planos devem ser tratados. A capacidade de renderização de cenários em tempo real depende, entre outros fatores, da utilização de *hardware* gráfico avançado.

A multiprojeção, aspecto característico de visualização imersiva distribuída, envolve a sincronização de *frames* e o compartilhamento de dados do modelo virtual entre os *hardwares* responsáveis pelas projeções nos diversos planos. O nível de paralelismo dos dados deve ser apurado de forma que a sincronização seja precisa, a fim de não haver perda da qualidade visual e prejuízo no realismo da simulação.

Abordagens tradicionais para o tratamento de multiprojeção fazem uso de sistemas especializados, que utilizam arquiteturas de *hardware* dedicado para controlar os múltiplos dispositivos de visualização. Sendo assim, algumas questões específicas de visualização distribuída devem ser averiguadas:

- A manutenção da consistência relacionada aos estados das entidades (componentes do modelo virtual), seus atributos e convergência do estado dinâmico corrente destes entre os *n* planos projetados;
- A sincronização dos diferentes relógios virtuais (processos temporizadores) associados aos diferentes componentes físicos do sistema, de forma a manter coerente a visualização dos cenários dinâmicos e sua relação com a taxa de amostragem dos quadros;
- O desempenho do sistema frente à relação número de entidades e número de planos de projeção.

A utilização de dispositivos de interatividade também são aspectos a serem considerados. Dispositivos como: luvas, óculos e capacetes podem ser utilizados para

interação no ambiente da planta naval/*offshore*, bem como o estudo das funcionalidades dos sistemas de manufatura frente à disponibilidade de integração com dispositivos de RV.

Como visto os desafios são muitos, entretanto a visualização distribuída imersiva, ou seja, visualização em *n* planos simulando os produtos e processos envolvidos na manufatura naval/*offshore* apresenta-se como um importante recurso às ferramentas de Manufatura Digital. O próximo capítulo descreve uma metodologia como solução para tais necessidades.

# 4 DA MODELAGEM À VISUALIZAÇÃO DISTRIBUÍDA

*Apresentar os desafios da utilização de RV em sistemas de manufatura da Indústria Naval/Offshore. Descrever a metodologia para solução destes desafios.*

Com base nas políticas públicas visando o restabelecimento da indústria de construção naval/*offshore* e frente ao panorama tecnológico associado a tal setor, conforme descrito nos capítulos anteriores, urge a necessidade de investimentos em pesquisa e desenvolvimento, associados às diferentes etapas do processo de manufatura naval/*offshore*. Novas facilidades associadas a sistemas de Manufatura Digital, como, por exemplo, ferramentas DMU integrando recursos de Realidade Virtual, surgem como áreas importantes a serem exploradas, necessitando de soluções que integrem diferentes ferramentas e sistemas hoje utilizados.

Nesse contexto, o presente capítulo apresenta uma metodologia para aplicação de recursos de Realidade Virtual aos sistemas de Manufatura Digital. Mais precisamente, é descrita a utilização de técnicas de visualização distribuída imersiva, em dispositivos do tipo CAVE, de cenários associados à Indústria Naval/*Offshore*.

## 4.1 REQUISITOS NECESSÁRIOS

Com base nas questões levantadas na Sessão 3.4 apresenta-se uma metodologia para visualização distribuída de plantas digitais de manufatura, a partir de modelos e simulações da Indústria Naval/*Offshore*. Essa solução deve ser capaz de possibilitar a visualização distribuída de processos de manufatura, a partir de modelos e simulações obtidas através de ferramentas comerciais costumeiramente utilizadas no setor produtivo.

Mais precisamente, estabelece-se como caso de estudo empresas de projeto, construção e comissionamento de produtos de alto valor agregado, como navios e plataformas *offshore*. A proposta faz uso de uma série de técnicas e ferramentas existentes, que integradas devem buscar as seguintes características:

- heterogeneidade: a qualidade e consistência da visualização deve ser garantida, independentemente das diferentes plataformas de hardware utilizadas numa mesma sessão de imersão;
- escalabilidade: o desempenho do sistema não deve ser comprometido pelo número de planos utilizados na projeção distribuída;
- portabilidade: a metodologia deve poder ser aplicada a diferentes tipos de plataformas de hardware e software, bem como diferentes padrões e formatos de dados;
- característica colaborativa: o sistema de visualização deve dar suporte a visualização colaborativa entre diferentes sistemas de visualização geograficamente distribuídos.
- baixo custo de implementação.

O diagrama apresentado na Figura 4.1 esquematiza a metodologia proposta neste trabalho, explicitando o fluxo de técnicas e ferramentas utilizadas. Retoma-se aqui as etapas descritas na Sessão 3.4., adequando-as ao caso de estudo.

Em síntese, propõe-se que, a partir de plantas de projeto CAD e modelos de simulação de processo fornecidas por ferramentas DMU, sejam aplicadas técnicas de redução de complexidade e conversão CAD-RV Imersivo, visando à obtenção de cenários virtuais com seus componentes, comportamentos e atributos. Os cenários e seus componentes são renderizados e multiprojetados. Para tal, se sugere a utilização da *engine* do jogo *Unreal Tournament*, a qual permitirá a navegação e interação do usuário com o ambiente.

A seguir apresenta-se o detalhamento de cada uma das etapas da metodologia, bem como a justificativa que suporta a utilização das ferramentas.

## 4.2 ETAPA DE MODELAGEM

Nessa etapa se supõe a utilização de alguma ferramenta DMU de modelagem e simulação de processos industriais. Tal ferramenta poderá importar arquivos em diferentes formatos CAD, oferecer bibliotecas para criação de entidades animadas, assim como outros recursos que facilitem o projeto mecânico de equipamentos, estudo do fluxo de produção e de aspectos ergonômicos associados aos operários. Em geral a modelagem é realizada por técnicos navais, e obtém uma descrição da planta industrial que permite a realização de simulações e análises.

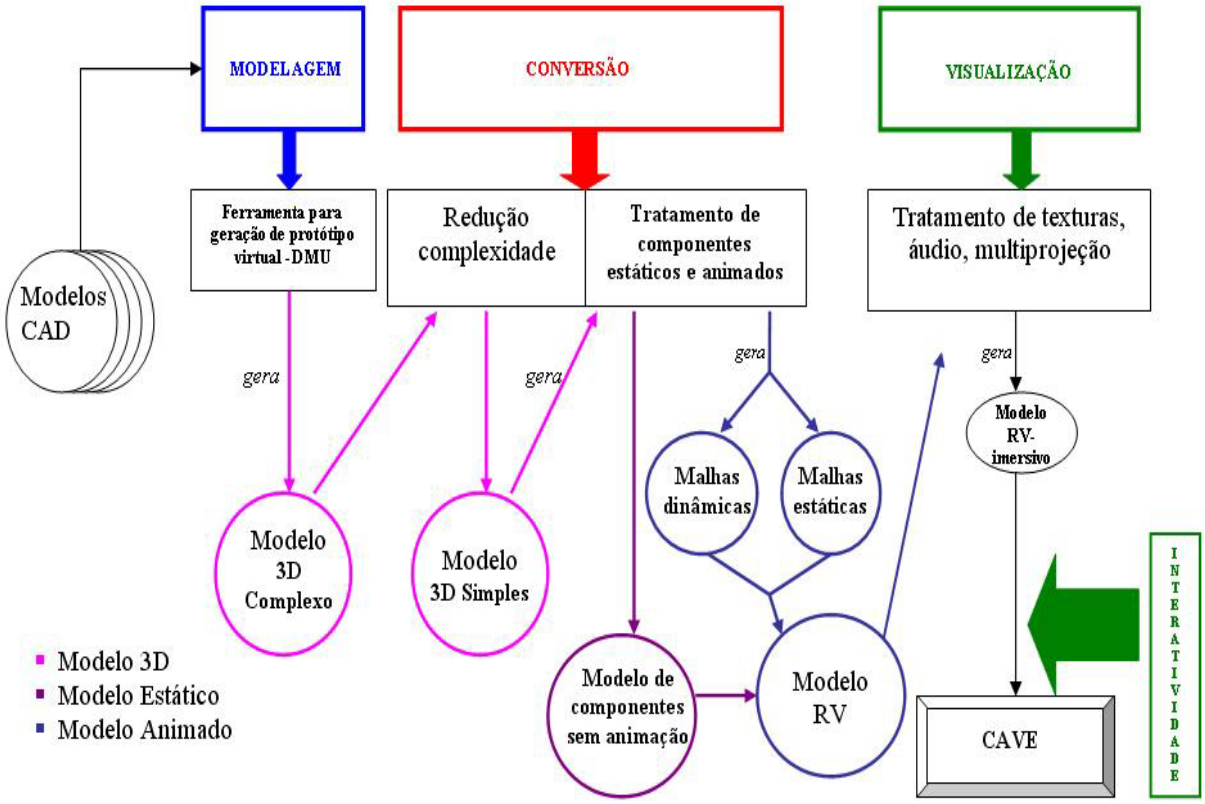



Figura 4.1. Metodologia da solução proposta.

Ferramentas DMU atuais permitem que tais cenários sejam descritos e exportados em diferentes formatos, tais como: CAD, VRML, ASE, DGN. Após a obtenção de um modelo da planta industrial torna-se necessário adequá-lo a um formato de visualização. Isso implica na introdução de uma etapa de conversão de dados, que irá tratar a transcrição de formatos e o tratamento dos componentes estáticos e animados do modelo.

## 4.3 ETAPA DE CONVERSÃO

Devido ao não comprometimento da etapa de modelagem e projeto com a possibilidade de visualização imersiva, os seguintes aspectos devem ser considerados:

- Os cenários fornecidos pelas ferramentas DMU podem apresentar excesso de determinadas informações, que não são utilizadas na etapa de visualização. Essas informações descrevem os componentes do modelo através de um número muito

grande de entidades geométricas, o que dificulta o processo de renderização a ser realizado online pela etapa de visualização.

- Certas características dos componentes do modelo, como texturas e comportamentos, não são consideradas para efeito de simulação na etapa de modelagem e, portanto, devem ser adicionadas aos cenários, para obtenção de um processo de visualização mais realista.

Assim, a utilização de técnicas/ferramentas capazes de efetuar a conversão CAD-RV Imersivo se faz necessária, visando à redução de complexidade das malhas componentes de cada elemento, bem como o tratamento de características dinâmicas e imersivas do modelo.

Após realizados uma série de testes com diferentes ferramentas de conversão de formatos e redução de complexidade, como: DeepExploration [Deep 1999]; 3DstudioMax [3DStudioMax 2004]; OkinoPoliTrans [Okino 1991]; e VizUp[Vizup 2006], optou-se pelo uso da ferramenta VizUp para redução de complexidade, e pelo 3DStudioMax para conversão de formatos.

O OkinoPoliTrans é uma ferramenta conversão de formatos VRML, entretanto o tempo para converter modelos de grande complexidade cujo número de polígonos seja excessivamente grande, como por exemplo mais que 600.000, torna a operação de conversão custosa em termos de tempo, levando às vezes dias para ser concluída.

Por outro lado o Deep Exploration faz a leitura e conversão de arquivos VRML para outros formatos de forma mais rápida que o Okino e possui uma ampla variedade de formatos suportados em importação e exportação, entretanto ambos não possuem suporte a tratamento de componentes animados e redução de complexidade.

A ferramenta VizUp permite a redução de complexidade que pode ser aplicada a diferentes tipos de arquivos, por exemplo VRML, reduzindo o número de polígonos existentes de acordo com a qualidade de visualização desejada.

O 3DStudioMax um dos softwares mais utilizados para desenvolvimento de aplicações gráficas possui suporte à importação e exportação de diversos formatos incluindo suporte à aplicação de texturas bem como tratamento de componentes animados através do uso de plugins de exportação de malhas animadas. Por este motivo optou-se na etapa de conversão pela utilização do software 3DStudioMax para tratamento de componentes animados e o VizUp para redução de complexidade, ambos versões demonstrativas utilizadas no estudo de caso do *Capítulo 5*.

Logo após a redução de complexidade, sugere-se a utilização do 3DStudioMax para conversão de formato e separação dos componentes estáticos e animados, obtendo-se a descrição estática e animada do Modelo RV-Imersivo. Supondo que o cenário obtido na etapa de modelagem esteja descrito em VRML, o 3DStudioMax o converte para formato ASE [ASE 2004], e então é utilizado o plugin ActorX [ActorX 2006] para a geração dos componentes do modelo de animação *psa* e do componente estático *psk*, de maneira a torná-los importáveis pelo jogo.

Como será visto na próxima sessão, após a redução e conversão do modelo obtido via ferramenta DMU, propõe-se o uso de uma *game engine* para a renderização e multiprojeção imersiva do cenário. Tal *engine* permite visualizar cenários estáticos e animados, e descrever diferentes comportamentos das entidades componentes. De modo mais preciso, os cenários devem ter seus componentes divididos em: esqueleto da malha (malhas estáticas); e comportamento deste esqueleto (malhas animadas).

## 4.4 ETAPAS DE VISUALIZAÇÃO E INTERATIVIDADE

Neste trabalho se propõe a utilização da *engine* do jogo *Unreal Tournament* (UT) para a etapa de visualização, associada à utilização de recursos de RV. Para tal, se realizou estudo sobre a ferramenta de forma a adequá-la às necessidades de renderização, visualização distribuída e interatividade. A seguir se detalha a estrutura modular da *game engine*.

A *Game Engine Unreal Tournament* constitui-se basicamente pelo: núcleo do jogo; código de visualização distribuída CaveUT [CaveUT 2004]; editor de cenários UnrealED [UnrealED 2000]; e de um servidor, conforme ilustrado na Figura 4.2.

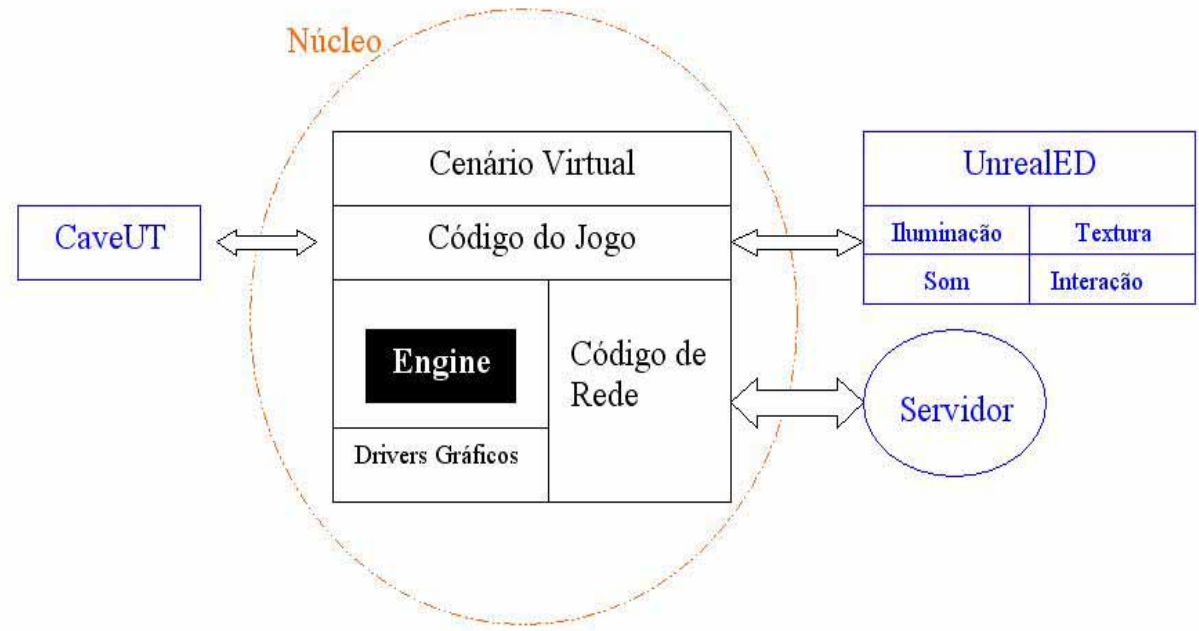



Figura 4.2. Estrutura modular da Game Engine.

**Núcleo do Jogo**

O Núcleo do jogo é dividido em módulos: Cenário Virtual, Código do Jogo, Código da Rede, *Drivers* Gráficos e a *Engine* propriamente dita.

O Cenário Virtual descreve o ambiente modelado onde os jogadores interagem. Nesse módulo são definidas as entidades, seus atributos de aparência e descrição geométrica. Ao instalar-se o jogo, diversos cenários virtuais são disponibilizados. Esses cenários são arquivos do tipo UT2004 Map, chamados genericamente de mapas. Como a proposta é utilizar ambientes navais renderizados pela *engine* do jogo, ao se criar ou obter o ambiente modelado, a partir de uma ferramenta DMU, é necessário transcrevê-lo para um arquivo do tipo UT2004 Map. Na sessão anterior descreveu-se a etapa de conversão onde os cenários descritos por ferramentas DMU podem ser convertidos em modelos de formatos passíveis de visualização na *engine*. Tais modelos após convertidos são considerados cenários ou mapas do jogo e devem ser inseridos em um diretório específico denominado *MAPS*, juntamente com os mapas padrão de instalação do jogo.

O módulo denominado Código do Jogo controla os mecanismos básicos do jogo, como: parâmetros de *display* (tipos de avatares e objetos, nível escolhido); parâmetros de rede; ações de jogadores; e todos os códigos referentes à implementação do jogo, exceto os

códigos de renderização. Os Códigos do Jogo são códigos abertos e podem ser alterados pelo usuário. Diversas comunidades de jogadores contribuem com alterações neste nível, porém é necessário um conhecimento sobre a linguagem de script utilizada.

De forma sucinta, se pode afirmar que esse módulo possui a descrição dos comportamentos associados aos diferentes componentes, através de uma linguagem de scripts orientada a objetos.

O Código de rede é construído sobre um protocolo similar ao DIS[1]/HLA[2], permitindo que diversos usuários localizados remotamente interajam em um mesmo ambiente virtual, tornando possível o trabalho colaborativo. Nesse protocolo um computador atua como servidor para o ambiente, enquanto os outros computadores comportam-se como clientes, dando suporte a usuários individualmente. Como o código de rede é robusto, este permite uma boa taxa de *frames* por segundo durante a renderização, mesmo em redes locais de alta latência.

A *Engine* propriamente dita é responsável pela renderização dos modelos tridimensionais, tratamento de obstáculos, e soluções referentes às simulações dinâmicas das leis físicas dos materiais. Esse módulo é o único com código proprietário, não permitindo qualquer tipo de modificação.

Por fim, os *Drivers* Gráficos traduzem os pedidos genéricos da *Engine* de renderização usando APIs gráficas como DirectX e OpenGL. Como os *drivers* possuem códigos abertos, o módulo CaveUT pode ser facilmente modificado para acomodar novos tipos de dispositivos de entrada/saída como CAVEs e HMDs.

**Servidor**

O servidor é um processo separado, geralmente executado em uma máquina distinta, que mantém a informação do mundo virtual coerente no tempo. O servidor deve manter as informações globais sobre o ambiente compartilhado, a interação dos jogadores, e realizar a sincronização das informações. Cada jogador possui uma janela distinta dentro do mesmo ambiente compartilhado, sendo essas janelas sincronizadas pelo servidor.

---

[1] Distributed Interactive Simulation
[2] High Level Arquitecture

**Código de visualização distribuída CaveUT**

O CaveUT, desenvolvido no *Pittsburg's Medical Virtual Reality Center* por Jeffrey Jacobson [Jacobson 2004] e fornecido sob a forma de código aberto, é um conjunto de modificações para o jogo *Unreal Tournament* que permite a visualização distribuída em *n* planos. A ferramenta CaveUT permite a modificação dos parâmetros de cada projeção, tais como perspectiva, rotação e deslocamento *offset*.

O CaveUT permite definir a configuração de cada projeção para simular os ângulos de visão que ficariam a esquerda, a direita, teto e chão de um observador, de acordo com a morfologia escolhida da CAVE. As modificações necessárias para produzir projeções fora do eixo são feitas em VRGL (*Virtual Reality Graphic Library)*, uma biblioteca modificada do OpenGL que torna possível a configuração dos parâmetros citados acima. O código do CaveUT pode ser encontrado dentro da pasta *SYSTEM* do jogo, no arquivo CaveUT.u, conforme Apêndice B.

A multiprojeção baseada no CaveUT requer um computador servidor conectado à uma rede local, com um número de computadores clientes conforme o número de projeções desejadas. O CaveUT permite 32 telas de visualização independentes para uma única aplicação. O operador deve iniciar o jogo *Unreal Tournament* instalado no servidor como *multiplayer*, e como *spectate* nas máquinas clientes. Maiores esclarecimentos sobre esses conceitos serão apresentados na Sessão 5.3.

Para a sincronização das renderizações o CaveUT adiciona um mecanismo *swaplock* que executa no computador servidor em paralelo com o jogo. O servidor envia para os computadores clientes um sinal de "pronto" e aguarda até receber de todos os clientes um sinal de "pronto" para enviar um sinal de "render". Os clientes aguardam até receber o sinal de "render" para executarem a renderização dos frames correntes. Isto assegura que todas as projeções serão renderizadas ao mesmo tempo na velocidade da máquina cliente mais lenta.

Quanto a interatividade, o CaveUT suporta dispositivos de rastreamento em tempo real utilizando por exemplo sistemas Intersense [Intersense 2004] ou outros dispositivos similares. Ligando sensores à cabeça do usuário e controladores em suas mãos, cria-se uma variedade de interações com o ambiente virtual. Os rastreadores ligados à cabeça do jogador permitem ao CaveUT gerar uma visualização estável enquanto o jogador movimenta-se dentro do ambiente. Ou seja, o servidor move o ponto de visão do avatar jogador no espaço virtual.

Além disso, outros dispositivos como *joysticks*, *gamepads* e botões de acionamento, complementam as possibilidades do CaveUT através do uso de um pacote *freeware* chamado VRPN (*Virtual Reality Peripheral Network*). Os controladores são ligados à máquina servidora e os dados dos periféricos são coletados pelo servidor VRPN, que também roda em paralelo na máquina servidora. Mais detalhes podem ser obtidos no projeto ALTERNE SASCubeTM [ALTERNE 2004].

A estereoscopia pode ser implementada no CaveUT através do uso de dois computadores para cada tela de projeção, um para renderizar a visualização do olho esquerdo e outro para renderizar a visualização do olho direito. Para isso é necessário que o *offset* de visualização da câmera seja ajustado de acordo com os parâmetros desejados [ALTERNE 2004].

Existem outras formas de implementação de estereoscopia através do CaveUT. Entre elas pode-se citar o uso de projetores estereoscópicos que alternam entre as visualizações do olho esquerdo e direito em um taxa de 120 *frames* por segundo. Nesse caso, o jogador utiliza óculos estereoscópicos onde cada lente alterna entre aberto e fechado a uma taxa de 120 *frames* por segundo, sincronizadas com o projetor.

**O Editor UnrealED**

O UnrealED é o editor da linguagem de scripts do UT que permite o desenvolvimento de cenários interativos, bem como a aplicação e modificação das características da *engine* do *Unreal*, tais como iluminação, textura, som e interatividade dos componentes. Esse editor permite importar arquivos em alguns formatos tais como: ASE e DXF para modelos estáticos, e *psa* e *psk* para modelos animados.

A partir dos arquivos descrevendo a malha do esqueleto das entidades (*psk*) e o seu comportamento (*psa*), obtidos na etapa de conversão, utiliza-se o editor para aplicação de textura, áudio e iluminação, entre outras características próprias a ambientes de visualização imersiva. Todo componente animado do cenário é considerado um ator, pois possui movimento. Animação é um comportamento que o ator possui. Sendo assim, carros, navios, pessoas, helicópteros que possuam movimento são classificados pela linguagem de scripts como *actors* (atores). Componentes estáticos que não possuem movimentos são descritos pela linguagem de script como *static meshes* ou malhas estáticas.

O ambiente de trabalho do UnrealED é ilustrado na Figura 4.3. A interface do editor não é de utilização trivial quanto aos principais recursos relacionados à importação e manipulação de cenários animados, bem como na bibliografia são escassos os documentos de auxílio a tais procedimentos. Assim, descreve-se a seguir um conjunto de etapas associadas à importação e geração dos cenários a serem visualizados, e mais adiante, no Apêndice A, apresenta-se, como material de auxílio, um manual descritivo desses procedimentos.

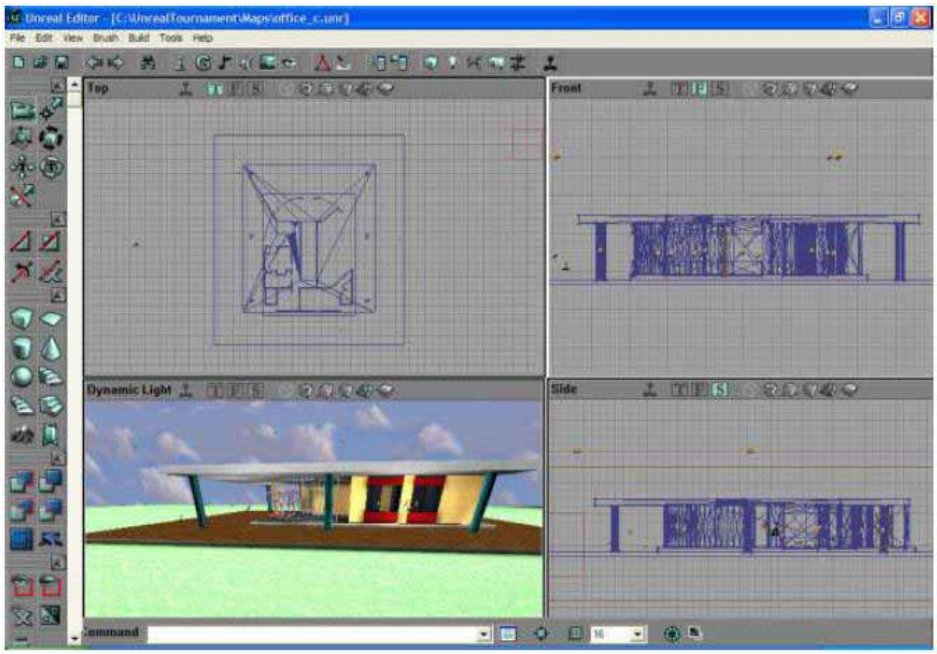



Figura 4.3. Ambiente trabalho do UnrealED.

### Aplicação de Texturas e Inclusão de Entidades Estáticas

A aplicação de texturas bem como a inclusão de entidades estáticas é simples bastando escolher entre as texturas disponíveis no UnrealED e para inserção de malhas estáticas posiciona-se com o mouse o local onde deseja-se inserir os componentes no cenário.

### Inserção de Entidades Animadas

O processo de criação de animações no UnrealED é um pouco mais complexo. O sistema utiliza o conceito de atores. Estes possuem comportamentos básicos definidos a partir de scripts de animação. Inicialmente, para um determinado ator, obtido da etapa de modelagem, o seu esqueleto deve ser importado via utilização de recursos específicos conforme Apêndice

A. Após, deve ser realizada a importação das possíveis animações associadas ao ator. Tais procedimentos compõem um ator animado, constituído de um conjunto de comportamentos.

O ator animado pode apresentar diferentes comportamentos ao longo da visualização do cenário. Para um determinado cenário, o comportamento do ator é definido em um script. Este define o conjunto de propriedades do ator animado, tais como sua posição inicial, o seqüenciamento de suas ações, a duração de cada comportamento, dentre outras características.

**Características de Visualização Imersivas**

É possível ainda atribuir propriedades aos cenários e atores que contribuíram para a obtenção de ambientes mais realísticos de visualização imersiva. Diferentes tipos de iluminação podem ser descritos, bem como se podem associar efeitos sonoros aos atores e componentes do cenário, dentre outros.

Recapitulando, a metodologia aqui proposta, partindo da modelagem de plantas de manufatura naval/*offshore* passa pela conversão dos modelos para cenários virtuais, finalmente possibilitando a visualização imersiva de cenários em dispositivo tipo CAVE, através da utilização da *game engine* UT. No próximo capítulo apresenta-se uma aplicação da metodologia e se descreve um conjunto de testes, conduzindo a uma análise da adequação da sua utilização.

# 5 TESTES E RESULTADOS

*Descrever um conjunto de testes realizados apontando resultados para as diversas etapas de solução da metodologia.*

A metodologia proposta no Capítulo 4 é validada em um estudo de caso relacionado com uma necessidade real de visualização distribuída e colaborativa associada à Indústria de Construção Naval e *Offshore*. Como outros grandes conglomerados industriais esse segmento utiliza cada vez mais tecnologias DMU na manufatura digital. Neste estudo de caso utiliza-se a ferramenta DELMIA/QUEST na etapa de modelagem devido ao seu crescente uso pelo setor, decorrente da diversidade de recursos para a modelagem e simulação que ela apresenta.

O QUEST possui recursos que vão desde a análise e simulação do fluxo de materiais e processos, passando pela análise do *layout*, identificação de gargalos na produção, determinação de máquinas e equipamentos necessários bem como verificação de impacto, em todo o sistema, de alterações no fluxo de processos, materiais e recursos.

O DELMIA, conforme visto anteriormente, permite a otimização do *layout* de fábrica assim como a determinação e validação das seqüências de montagem e análise ergonômica, possibilitando a análise global, simulação 3D e controle de automatização de máquinas e processos. Entre os diversos recursos disponíveis existe a possibilidade de criação de cenários 3D. Entretanto, tais cenários não podem ser diretamente visualizados de forma distribuída.

Assim, partindo-se de plantas CAD de projetos de navios e plataformas, bem como de modelos do processo de produção de um estaleiro, integrados em diversos cenários, fornecidos pelo DELMIA/QUEST, aplicam-se as etapas propostas na metodologia.

## 5.1 ETAPA DE MODELAGEM

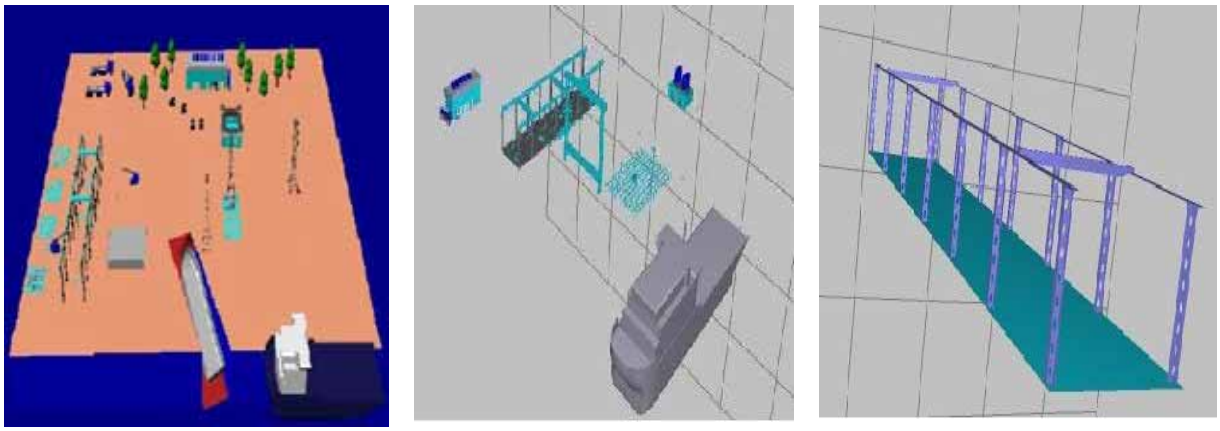
Na etapa de modelagem, os diferentes projetos em CAD são importados e integrados em um cenário virtual 3D através do Delmia [Delmia 2002]. Essa ferramenta importa e exporta arquivos representando cenários em diversos formatos tais como: IGES, DXF, DWG, VDA, DES, STL, VRML e STEP.A modelagem de um cenário real é ilustrada nas Figuras 5.1 e 5.2.

A Figura 5.1 apresenta uma das oficinas do cenário real e a Figura 5.2 representa a oficina virtual modelada utilizando as ferramentas QUEST e DELMIA.

Figura 5.1. Oficina real do estudo de caso.

Figura 5.2. Oficina virtual modelada.

Para o propósito da validação aqui descrita foram utilizados três modelos de cenários distintos, desenvolvidos com o uso das ferramentas QUEST e DELMIA. Os modelos possuem diferentes complexidades e foram escolhidos com o propósito de verificar o rendimento da etapa de visualização. Nas sessões seguintes eles serão referenciados como Estaleiro 1, Estaleiro 2 e Oficina. A Figuras 5.3 apresenta esses modelos.



(a) Estaleiro 1. (b) Estaleiro 2. (c) Oficina.

Figura 5.3. Estaleiro 1(a), estaleiro 2(b) e oficina(c).

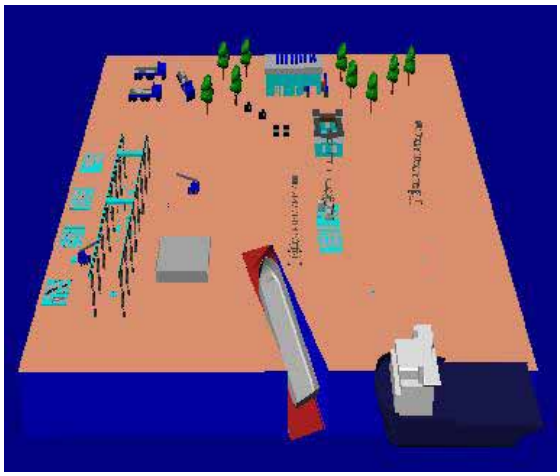
O Estaleiro 1, Figura 5.3 (a), apresenta um conjunto de entidades representando área do estaleiro, partes de um navio, armários, guindastes fixos e móveis, sucatas, máquinas de solda, empilhadeiras, chapas, colunas, blocos de construção, ponte rolante, oficina e prédios. O Estaleiro 2, Figura 5.3 (b) representa um estaleiro simples com poucos detalhes, e o modelo da oficina, Figura 5.3 (c), representa apenas a estrutura física (vigas) de uma oficina. Os cenários dos modelos foram exportados em formato VRML complexo (com grande número de polígonos - sem redução de complexidade), com as características apresentadas na Tabela 5.1.

Tabela 5.1. Configurações originais dos modelos utilizados.

| | **Vértices** | **Triângulos** | **Tamanho (KB)** |
|---|---|---|---|
| *Estaleiro 1* | 806.476 | 385.854 | 66.154 |
| *Estaleiro 2* | 202.731 | 224.522 | 30.085 |
| *Oficina* | 37.960 | 35.688 | 3.323 |

## 5.2 ETAPA DE CONVERSÃO

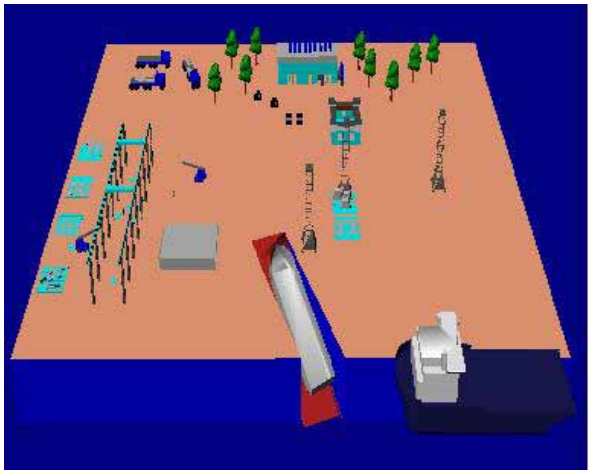
De posse do cenário exportado em VRML realiza-se a etapa de redução de complexidade dos modelos, de forma a atingir uma boa qualidade visual com um número de polígonos admitido pelo editor UnrealEd: 61.000 polígonos. Para isso utiliza-se a ferramenta VizUp que reduz de forma significativa à complexidade do modelo apresentado. A Figura 5.4 apresenta as etapas de redução do Estaleiro 1 até atingir as configurações desejadas.



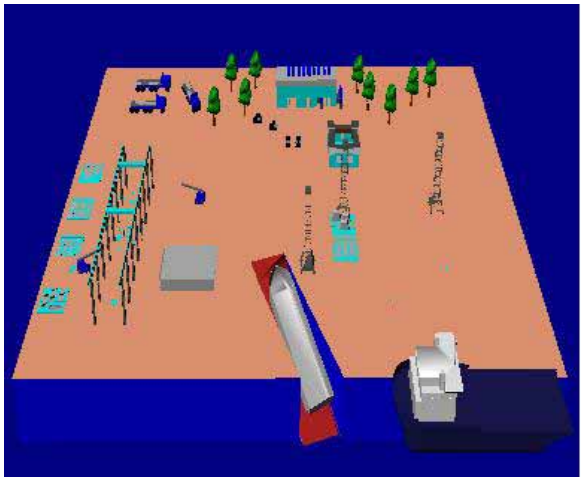
(a) Estaleiro1: Original.

(b). Estaleiro 1: Redução 90%.

(c). Estaleiro 1: Redução 88%.



(d). Estaleiro 1: Redução 86%.



(e). Estaleiro 1: Redução 84%.

Figura 5.4. Conversão do estaleiro 1 (a), (b), (c), (d) e (e).

Os modelos reduzidos do estaleiro 1 apresentaram as características descritas na *Tabela 5.2*.

Tabela 5.2. Reduções do modelo 1 (estaleiro 1).

| | **Taxa Redução** | **Vértices** | **Triângulos** | **Tamanho (kb)** |
|---|---|---|---|---|
| *Estaleiro 1* | - | 806.476 | 385.854 | 66.154 |
| *Estaleiro 1* | 90% | 58.415 | 35.443 | 2.685 |
| *Estaleiro 1* | 88% | 68.153 | 46.301 | 3.197 |
| *Estaleiro 1* | 86% | 79.114 | 54.019 | 3.790 |
| *Estaleiro 1* | 84% | 90.830 | 61.736 | 4.446 |

Iniciaram-se os testes com uma taxa máxima de redução de 99%, entretanto observou-se que no intervalo entre 99% e 91% as reduções implicavam em uma desfiguração completa do modelo original, tornando-o inutilizável. A partir de uma taxa de 90% é possível identificar o modelo, porém com perda de detalhes. Diminuíram-se então as taxas de redução até alcançar uma configuração com uma boa qualidade visual. A maior taxa considerada adequada foi de 84%.

O estaleiro 2 e a oficina, em formato VRML, após as reduções, apresentaram as características descritas na *Tabela 5.3.* Na *Figura 5.5* ilustram-se os aspectos visuais desses modelos.

Tabela 5.3. Reduções dos modelos 2 e 3 (estaleiro 2 e oficina)

| | **Taxa Redução** | **Vértices** | **Triângulos** | **Tamanho (kb)** |
|---|---|---|---|---|
| *Estaleiro 2* | - | 202.731 | 224.522 | 30.085 |
| *Estaleiro 2* | 73% | 89.087 | 61.919 | 17.692 |
| *Oficina* | - | 37.960 | 35.688 | 3.323 |
| *Oficina* | 46% | 22.087 | 19.271 | 2.174 |

Através das *Tabelas 5.2* e *5.3* é possível concluir que em cenários simples, quando a complexidade é menor, não é possível aplicar taxas de redução muito altas devido à simplicidade do modelo.

Uma vez convertidos os modelos para VRML simples, manipulam-se alguns dos atributos de seus componentes com o uso da ferramenta 3DStudioMax. Separam-se os componentes animados dos componentes estáticos e tratam-se individualmente as características de textura ausentes ou de baixa qualidade advindas do modelo CAD.

Finalmente, após a obtenção do modelo em VRML simples, com o Vizup, e a conversão de tipos feitas com o 3DStudioMax, tem-se o Modelo RV-imersivo, passível de visualização no ambiente CAVE.

## 5.3 ETAPA DE VISUALIZAÇÃO

O cenário resultante da modelagem e conversão é então visualizado de forma distribuída entre *n* planos de projeção. No estudo de caso em questão foi realizado um experimento com implementação física para dois planos de projeção e testes simuláveis em três e quatro planos de projeção.

O *Unreal Tournament* e o pacote CaveUT foram instalados nos três computadores utilizados no experimento, sendo que a instalação do jogo *Unreal* contém o pacote com o editor UnrealED.

Com o editor UnrealED importaram-se os arquivos gerados pelo 3DStudioMax: os componentes animados (arquivo *psa*) e os componentes estáticos (arquivo *psk*). Criou-se o modelo do estaleiro no editor, conforme descrito no Apêndice A, denominando-o DMmapaEstaleiro.map. Foram então gerados os pacotes de animação (site.ukx) e as malhas estáticas (coppe.usx). Esses três arquivos foram copiados nas pastas *MAP*, *ANIMATION* e *STATICMESHES* no diretório *UT2004* do jogo.

A multiprojeção baseada no CaveUT requer um computador servidor que é conectado a uma rede local (LAN) com os computadores clientes. Cada cliente é responsável por projetar o cenário em uma tela. No experimento foram utilizados dois computadores clientes e um servidor.

O operador da máquina servidora deve iniciar o jogo *Unreal Tournament* no modo "host", e em seguida escolher um tipo de jogo *GameType*. Para o mapa criado DMmapaEstaleiro.map utiliza-se o "Death-Match". Entretanto pode-se escolher qualquer tipo de jogo. A seguir se escolhe o mapa ou cenário virtual, na guia (*Select Map*). Neste caso o cenário naval DMmapaEstaleiro.map. Na guia *mutator* seleciona-se a opção CaveUT e volta-se a guia *Select Map* para iniciar o servidor clicando em "Listen".

As máquinas clientes devem iniciar o jogo em seguida, clicando-se em "Join Game" e setando a guia LAN caso esteja executando em uma CAVE local, ou a guia Internet para executar de outra rede geograficamente separada. Neste trabalho utilizou-se a guia *(LAN)* e selecionou-se a opção "CaveUT Server" em cada uma das 2 máquinas clientes. Caso esteja-se executando de uma CAVE remota utiliza-se a guia *Internet*. Por fim, clicando em "Spectate", inicia-se a configuração dos parâmetros de projeção manualmente. Essas configurações podem ser modificadas diretamente no jogo ou no arquivo CaveUT.ini, conforme Apêndice B.

Cada espectador deve configurar os parâmetros de projeção: correção perspectiva, rotação e *offset* individualmente. Cada disposição física das telas da CAVE (número de telas) requer um valor distinto a ser calculado. Os valores utilizados para cada uma das telas de projeção foram configurados manualmente durante a execução do jogo e se encontram descritos no código CaveUT.ini dos espectadores, apresentado no Apêndice B. Para disposições diferentes da utilizada neste estudo podem-se calcular os parâmetros de acordo com o material disponível em [Jacobson 2004].

Ao configurar individualmente os parâmetros nas máquinas clientes cada espectador irá duplicar a visão do jogador do servidor, em cada cliente o código CaveUT rotaciona a visão do mundo virtual que executa no servidor e projeta para sua respectiva tela. Logo cada projeção cliente compõe parte da visão geral do jogador no servidor.

O *layout* do sistema de projeção, bem como as suas configurações são ilustrados na Figura 5.5 A visão de uma tela é rotacionada 45 graus para esquerda e a da outra 45 graus para a direita. Para instalação sem nenhum *head tracking*, como neste experimento, especificou-se um único ponto de visualização ideal para o *display* inteiro. Caso haja um rastreador não é necessário especificar um ponto de visualização pois o CaveUT deverá corrigir a perspectiva em tempo real, de modo a perseguir os movimentos do usuário.

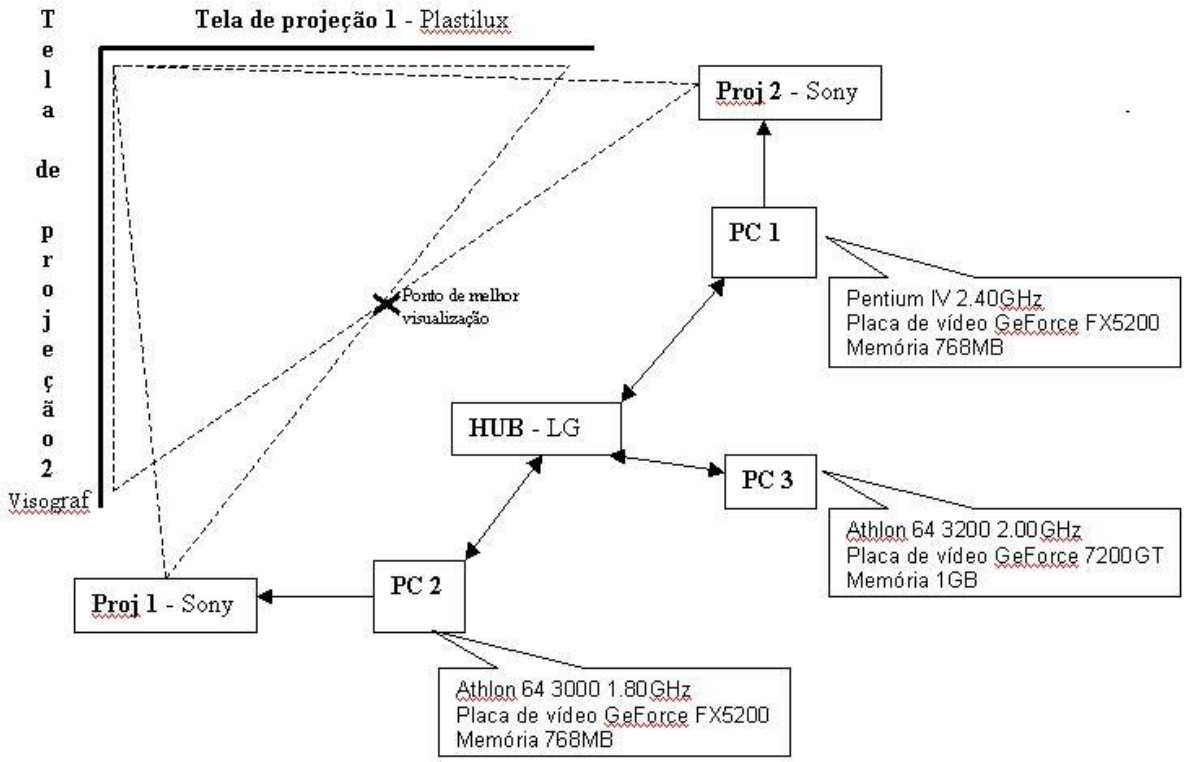



Figura 5.5. Layout do sistema de projeção utilizado no experimento (CAVE)

Foram utilizados três computadores, um modem ADSL de quatro portas funcionando como HUB, dois projetores e duas telas de projeção. Um computador Athlon 3200 foi utilizado como servidor (um jogador atuante - *player*) e os outros dois utilizados como clientes (dois jogadores espectadores - *spectates*). Os três sistemas foram conectados em uma rede local através do modem. As máquinas clientes agem como espectadoras da máquina servidora. A *engine* do jogo renderiza o cenário bem como se encarrega do gerenciamento de outras mídias como, por exemplo, o som.

Mesmo utilizando diferentes configurações de hardware para a CAVE, característica não ideal, os resultados do experimento se mostraram adequados, tanto em níveis de sincronização da visualização como em grau de imersão, comprovando a meta de heterogeneidade da proposta. É importante salientar que diferentes formatos de descrição de cenários, que não VRML, podem ser utilizados como entrada do sistema.

A multiprojeção implementada pelo CaveUT mostrou-se eficiente e simples, bastando apenas alguns ajustes nos parâmetros de configuração para que se pudesse obter as projeções corretas na CAVE em duas telas formando um ângulo de 90º.

Além disso, a fim de avaliar o desempenho do experimento no que tange a escalabilidade, testou-se as taxas de renderização para 3 e 4 planos de projeção, porém sem a montagem de um experimento real. A Tabela 5.4 mostra os resultados obtidos nas simulações realizadas, em termos de quadros por segundo.

Tabela 5.4. Resultados quanto a escalabilidade.

| **Simulação** | **Um plano de projeção.** | **Dois planos de projeção.** FPSm = FPS médio entre os 2 planos | **Três planos de projeção.** FPSm = FPS médio entre os 3 planos | **Quatro planos de projeção.** FPSm = FPS médio entre os 4 planos |
|---|---|---|---|---|
| Modelo 1 (84%) | FPS entre 500 e 520; | FPSm entre 485 e 510; | FPSm entre 413 e 424; | FPSm entre 387 e 418; |
| Modelo 1 (86%) | FPS entre 530 e 540; | FPSm entre 520 e 530; | FPSm entre 425 e 433; | FPSm entre 392 e 415; |
| Modelo 1 (88%) | FPS entre 575 e 585; | FPSm entre 554 e 575; | FPSm entre 427 e 462; | FPSm entre 392 e 415; |
| Modelo 1 (90%) | FPS entre 640 e 650; | FPSm entre 637 e 647; | FPSm entre 427 e 463; | FPSm entre 393 e 415; |
| Modelo 1 (original) | Não simulável devido ao número de polígonos | Não simulável devido ao número de polígonos | Não simulável devido ao número de polígonos | Não simulável devido ao número de polígonos |

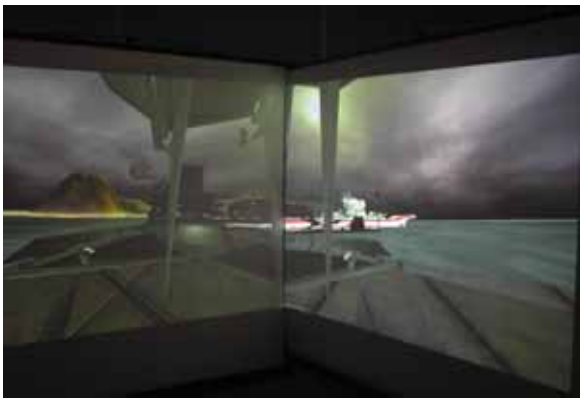
É possível observar que o aumento do número de planos de projeção acarretou uma ligeira diminuição no desempenho da etapa de visualização. Utilizando o servidor como espectador, para os casos de três e quatro planos de projeção com computadores de desempenho razoável (um Athlon 1.2GHz, placa gráfica nForce2 256MB e um Pentium IV 1.8GHz 512MB) os números de *frames* por segundo diminuíram em uma escala maior do que no experimento físico com dois planos.

Disso se depara que são necessários testes com mais planos de projeção, em uma rede estável com máquinas de configurações idênticas, a fim de verificar o desempenho do sistema. A utilização de *hardwares* distintos, bem como possíveis atrasos devidos à configuração de rede, também podem ter causado essa diminuição no número de *frames* por segundo.

Avaliou-se, ainda, a possibilidade de imersão à distância, para validar a metodologia no que diz respeito ao suporte a ambientes colaborativos separados geograficamente. Foram feitos dois testes: no primeiro executou-se o sistema como um espectador da CAVE remota; e no segundo executou-se como jogador para navegar na CAVE remota.

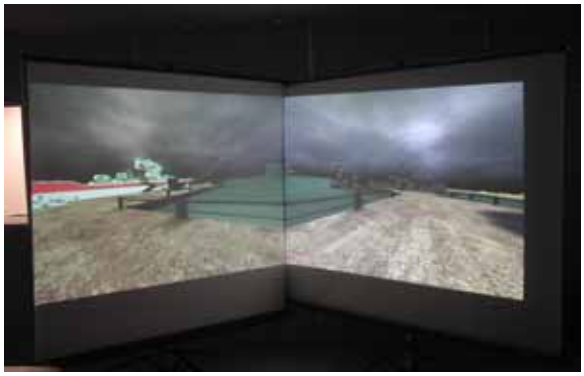
Para esses experimentos foram utilizados um computador Athlon 2.3GHz com 1GB memória e placa gráfica GeForce MX440. A navegação remota mostrou-se eficiente em níveis de sincronização tanto para o teste como jogador quanto para o teste do espectador, mantendo as taxas de *frames* por segundo aproximadamente iguais aos valores da Tabela 5.4 para o caso de 3 planos de projeção do modelo 1 com taxa de redução de 84%.

Os recursos multimídias disponíveis no UT foram testados através da colocação de objetos sonoros distribuídos no cenário. Por exemplo, associado a rampas de metal foram adicionados arquivos de som representando o caminhar de pessoas por estas superfícies, ver Figura 5.6. A sincronização dos *frames* entre os planos de projeção e o tratamento de obstáculos realizados pela *engine* também mostraram-se adequadas para os experimentos desenvolvidos. As Figuras 5.6 (a), (b), (c), (d), (e), (f), (g) e (h) mostram cenas dos experimentos e a CAVE utilizada para realizá-los.



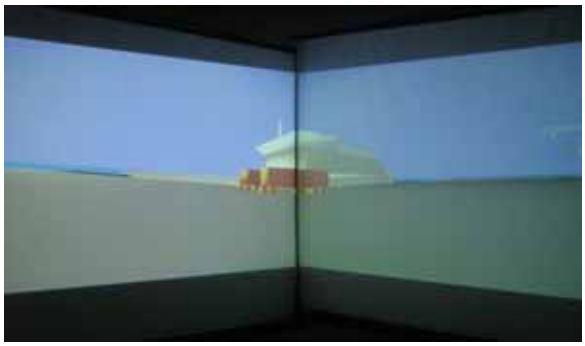
(a) Navios do UT, com textura.

(b) Plataforma *Offshore* do UT.

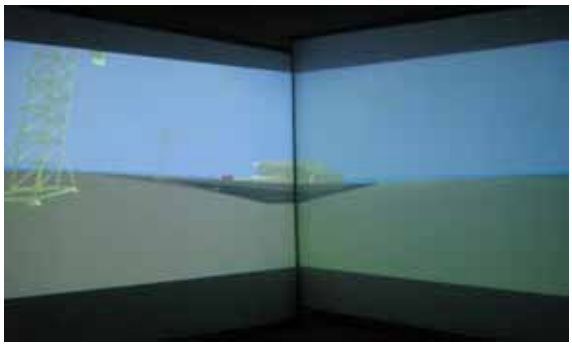


(c) Plataforma do UT.

(d) Rampa de embarque do UT.

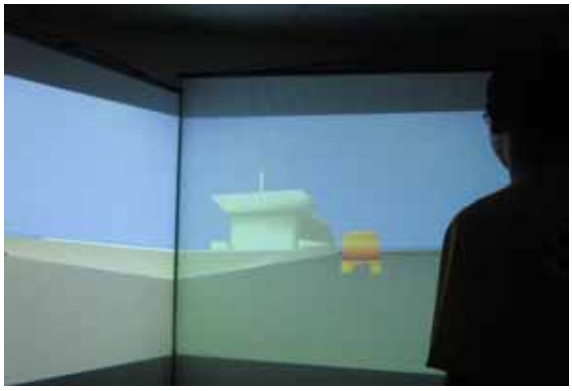


(e) Estaleiro 1.



(f) Guindastes e pátio do Estaleiro 1.

(g) Imersão do usuário no Estaleiro 1.



(h) Navio e caminhão do Estaleiro 1.

Figura 5.6. Visualização distribuída de cenários navais na CAVE (*a, b, c, d, e, f, g, h*).

Em termos gerais, pode ser dito que, no que se refere ao tema proposto, a metodologia apresentada cumpriu as expectativas esperadas: visualização distribuída em *n* planos em dispositivos do tipo CAVE. No que segue, são apresentadas propostas de futuros desenvolvimentos e conclusões sobre o trabalho realizado.

# 6 CONCLUSÕES E PERSPECTIVAS FUTURAS

*Apresentar conclusões e indicar possíveis estudos de continuidade da proposta.*

Este trabalho apresentou um estudo sobre o processo de manufatura associado à indústria naval/*offshore*. De forma mais precisa, foi apresentado um panorama desse mercado, focalizando-se nos aspectos relacionados a técnicas e ferramentas associadas à manufatura digital na construção naval.

Conceitos básicos relacionados à Realidade Virtual e sua utilização na manufatura digital também foram apresentados, de forma a fornecer subsídios ao entendimento da proposta principal deste trabalho: uma metodologia capaz de fornecer recursos de visualização imersiva de cenários associados ao processo de manufatura naval/*offshore*.

Após a identificação das limitações, restrições e necessidades associadas ao problema, foi proposto um conjunto de quatro etapas para a implementação de uma metodologia: a modelagem, a Conversão CAD/RV-Imersivo, a Visualização e a Interatividade. Essas etapas congregam procedimentos que integram diferentes ferramentas existentes, finalizando na possibilidade de projeção em *n* planos com características imersivas e recursos de RV. Ressalta-se que atualmente não existem ferramentas que, de forma integrada, realizem tal procedimento. Nem foi encontrada na bibliografia uma proposta de solução para esse problema.

A metodologia foi validada em uma aplicação real no âmbito da indústria naval e *offshore*. Cenários associados a plantas de estaleiros, projetos de navios e plataformas foram modelados, convertidos e sua visualização distribuída foi propiciada em uma CAVE.

Com base em cenários concebidos por ferramentas DMU, apresentadas no *Capítulo 3*, foram obtidos os modelos dinâmicos a serem visualizados. Após levantamento de possíveis soluções existentes, para a etapa de conversão foi proposta a utilização das ferramentas VizUp e 3DStudioMax. Estas permitem a redução de polígonos e a conversão dos modelos sem prejuízos ao processo de renderização.

A utilização da *engine* do jogo *Unreal* na manufatura digital é outra contribuição original deste trabalho. Um estudo dos diferentes módulos existentes, sua integração e desenvolvimento de solução apropriada para uso em CAVE foram realizadas. O eficiente

módulo de renderização, a possibilidade de multiprojeção a *n* planos e a interatividade, além de uso colaborativo em rede, são características do jogo que contribuíram para a qualidade dos resultados obtidos. A solução via utilização da *Engine* do UT se mostrou bastante simples, robusta e com custo de implementação bastante baixo. Entretanto, o jogo tem como limitação o número máximo de polígonos por entidade. No estudo de caso efetuado, tal limitação foi resolvida pela redução de complexidade realizada na etapa de conversão, sem comprometimento da qualidade final de visualização.

Um outro aspecto a ser considerado é a possibilidade da metodologia ser aplicada diretamente na planta por técnicos, no caso em estudo da área naval, não necessitando de grandes conhecimentos em programação. A metodologia proposta pode ser aplicada a diferentes tipos de arquivos de cenários, bem como a diferentes plataformas de hardware e sistemas operacionais. As ferramentas utilizadas na etapa de visualização, como a *game engine* UT e o CaveUT, apresentam versões Linux e Windows. Entretanto, na etapa de modelagem e conversão foram utilizados, no estudo de caso, versões para a plataforma Windows.

O trabalho mostrou-se útil na compreensão dos vários elementos envolvidos na utilização de RV na manufatura, constituindo uma contribuição aos sistemas digitais de manufatura, não só da indústria naval/*offshore* à qual o estudo foi aplicado, mas às diversas indústrias e setores produtivos. Se a modelagem e simulação dos processos de manufatura já são uma realidade, a utilização de interfaces elaboradas de visualização distribuída com recursos de RV, capazes de integrar as diferentes ferramentas de projeto e simulação, ainda é um desafio (mesmo que a visualização de projetos CAD seja amplamente utilizada).

Como perspectivas futuras, estabelecem-se a necessidade de automatização de algumas das etapas do processo através do desenvolvimento de um *framework* que integre as etapas de modelagem, conversão e visualização. Ainda a realização de testes com um maior número de planos de projeção, bem como com equipamentos interativos, como luvas e óculos estereoscópicos, podem ser realizados.

Finalmente, se sugere o aprimoramento dos aspectos relacionados à interatividade como, por exemplo, a inclusão de recursos de Realidade Mista ao UT, possibilitando a visualização distribuída de entidades representativas de operários e máquinas que de forma interativa e em tempo real atuam na planta real.

# APÊNDICES

## A – Manual para criação do cenário virtual (mapa UT)

**<u>Softwares necessários:</u>**

I. VizUp
II. 3DStudioMax (plugin ActorX )
III. UnrealEd

***Resumo de descrições de cada software:***

**I.** VizUp: Reduz polígonos das entidades em formato VRML. Sua função é tornar possível a importação de entidades complexas pois o UnrealEd permite importar como malha estática - *static mesh* - entidades com no máximo 61.000 polígonos, sendo necessária a redução até esse número.

**II.** 3DStudioMax: Modelador 3D usado para importar entidades em formato VRML e exportar em formato ASE - *Ascii Scene Export* - (formato importável pelo UnrealEd), para os casos de entidades com comportamento estático. Para entidades animadas o *plugin* ActorX é utilizado para separar a entidade no formato VRML em duas partes: o esqueleto da malha (psk) e o comportamento deste esqueleto, ou, em outras palavras, a animação propriamente dita (psa), sendo esses arquivos importáveis pelo editor.

**III.** UnrealEd: Editor de cenários simuláveis do *Unreal Tournament*. Nesse editor são importados as entidades e seus respectivos comportamentos dentro dos cenários (mapas), além disso, o editor permite inserir iluminação, texturas e outros aspectos relacionados à criação de cenários virtuais.

***Etapas necessárias:***

**Passo 1: Reduzindo os polígonos do modelo VRML com VizUp**

Após executar o programa VizUp, basta abrir o arquivo desejado e clicar no botão <*Reduce*> como mostra a *Figura A.1*:

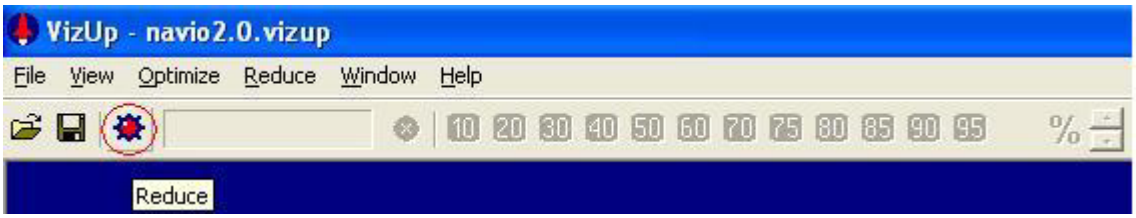



Figura A.1. Redução de polígonos no VizUp.

Essa ação faz com que o programa reduza o número de polígonos do modelo ou da entidade. Porém é necessário indicar manualmente a porcentagem de redução que será realizada. Os botões numéricos e as setas pra cima e pra baixo, mostradas na *Figura A.1*, indicam as taxas de redução que podem ser utilizadas. A redução é necessária para entidades com mais de 61.000 polígonos, tendo que se reduzir até esse número.

**Passo 2. Convertendo formatos com o 3DStudioMax(3DS)**

Esta etapa converte a entidade ou o modelo, de formato VRML para formatos importáveis pelo UnrealEd. Existem dois procedimentos para este tipo de conversão no 3DS: um para malhas estáticas, isto é, entidades sem animação, e outro para animações, isto é, VRMLs animados.

- **Importação de malhas estáticas**

Após clicar em *<File>Import*, como mostra a *Figura A.2*, tem-se a tela da *Figura A.3*, onde é preciso especificar o tipo de arquivo que queremos importar, neste caso VRML. Após, basta escolher o arquivo a ser importado e desmarcar todas as opções como mostra a *Figura A.4.*.

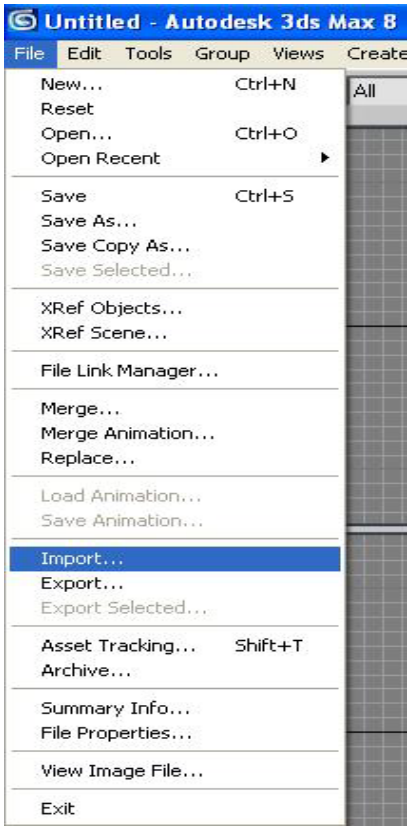



Figura A.2. Importar VRML.

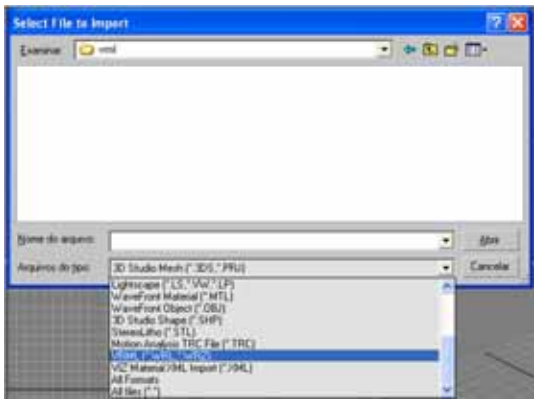



Figura A.3. Selecionando VRML.

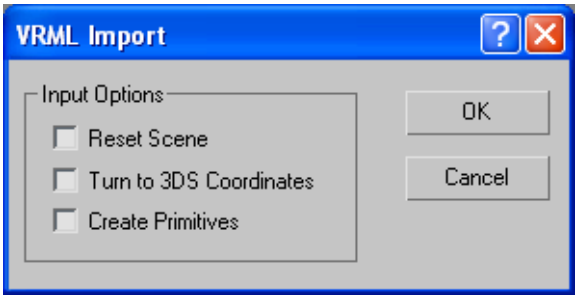



Figura A.4. Opções de importação VRML.

Passa-se então para a etapa de exportação.

- **Exportação de malhas estáticas**

Usa-se a opção do menu <*File*>*Export* como na *Figura A.2*, porém escolhendo a opção <*Export*>. A seguir, se salva o arquivo no formato ASE através da opção <*Salvar como:* ASCII *Scene Export* (ASE)>, como mostra a *Figura A.5*. As opções de exportação devem estar exatamente como na *Figura A.6*.

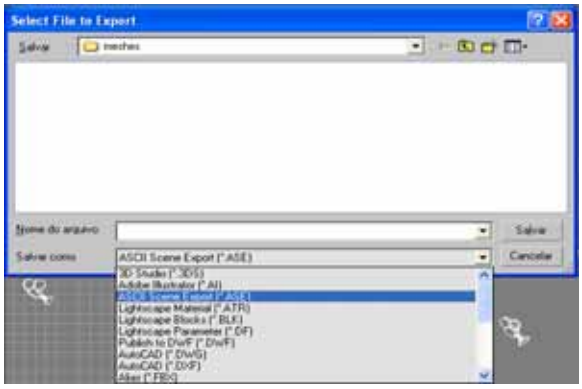



Figura A.5. Exportar ASE.

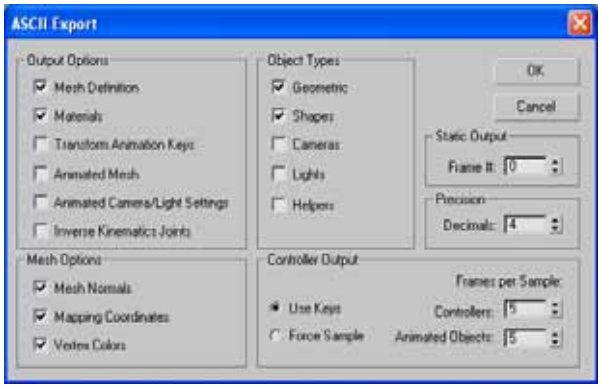



Figura A.6. Opções de exportação.

Clicando em <*OK*>, a etapa de exportação estará finalizada e então se realiza a etapa de importação deste arquivo recém gerado no UnrealEd.

- **Exportação de animações**

Após importar o VRML usa-se o *plugin* ActorX do 3DS fornecido pelo fabricante do jogo. Clicando no ícone utilidades com a figura de um "martelo" no canto direito da tela, conforme *Figura A.7*, e em seguida em <*More*>, aparecerá a janela "Utilities". Deve-se selecionar o *plugin* <*ActorX*> e confirmar em <*OK*>. Note que devido ao fato de não ser um *plugin* nativo do 3DS o ActorX deve ser previamente instalado antes de sua utilização.

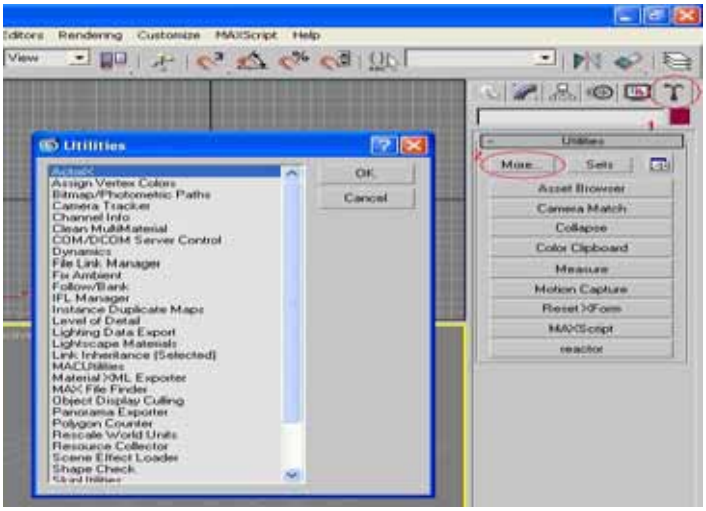



Figura A.7. Utilizando plugin ActorX.

No primeiro campo de configuração do plugin, conforme *Figura A.8*, seleciona-se o diretório para onde serão exportados os arquivos do esqueleto da malha (arquivo psk) e a animação (arquivo psa). No segundo campo especifica-se o nome do arquivo psk que será gerado. Ao clicar no botão <*Save mesh/refpose*> este será exportado.

Em seguida realiza-se a exportação da animação (arquivo psa), conforme mostra a *Figura A.9*. Torna-se necessário preencher o campo <*animation file name*> e <*animation sequence name*> com um nome qualquer e então clicar no botão <*Digest Animation*>.

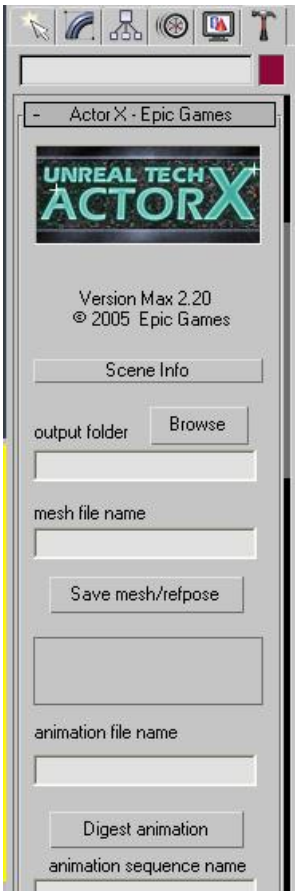



Figura A.8. Diretório de exportação psa e psk.

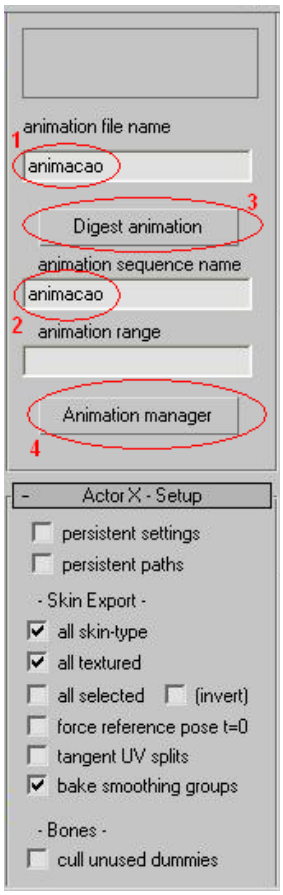



Figura A.9. Exportando psa.

Feito isso, pressionando o botão <*Animation Manager*> obtém-se a janela mostrada na *Figura A.10.*. Basta então clicar no ícone “seta para a direita” e <*SaveAs...*>, escolher um nome para o arquivo e clicar em salvar. Tem-se então o arquivo psa gerado.

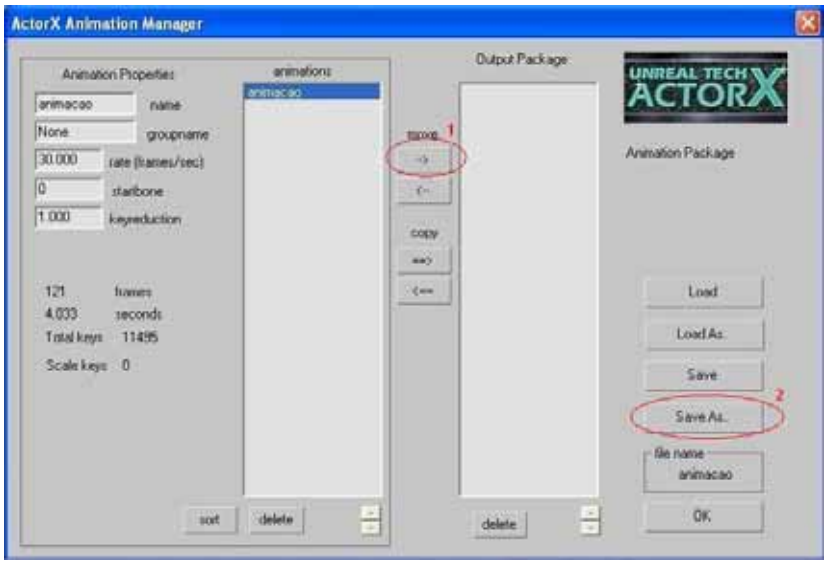



Figura A.10. Gerando o arquivo psa de animação.

**Passo 3. Criando o ambiente virtual no UnrealEd**

A *Figura A.11.* mostra os componentes básicos da barra de ferramentas do editor utilizados nesta etapa.

Ícone 1 - Actor Class Browser
Ícone 2 - Texture Browser
Ícone 3 - Static Mesh Browser
Ícone 4 - Animation Browser

Figura A.11. Componentes utilizados da barra de ferramentas do UnrealED

- **Estrutura básica do cenário**

Ao clicar com o botão direito do mouse sobre o botão que apresenta a figura de um cubo na barra de ferramentas da *Figura A.12.*, a janela "CubeBuilder" aparecerá. É preciso então definir a altura, largura e comprimento do cubo, que são os 3 primeiros valores da lista. Feito isso, basta clicar em <Build> e logo após em <*subtract*>. Esse cubo indicará os limites do mundo virtual. Tem-se então a estrutura básica: chão, teto e paredes, criados com uma textura padrão.

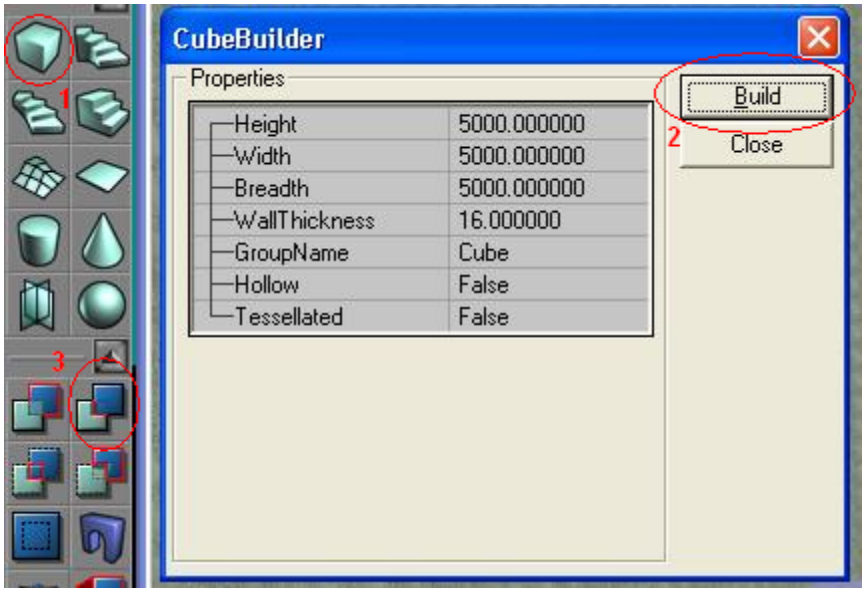



Figura A.12. Criando os limites do cenário virtual.

Para mudar a textura de qualquer superfície basta selecionar uma ou várias superfícies, segurando a tecla <ctrl> e clicando no ícone <*Texture Browser*>, mostrado na *Figura A.11*. Aberta a janela *Textures* basta clicar no ícone "1" (conforme indicado na *Figura A.13.*) e então abre-se um pacote de texturas. Para alternar entre diferentes grupos do mesmo pacote utiliza-se a caixa de seleção "2". Para selecionar e aplicar a textura sobre as superfícies selecionadas basta escolher a textura desejada.

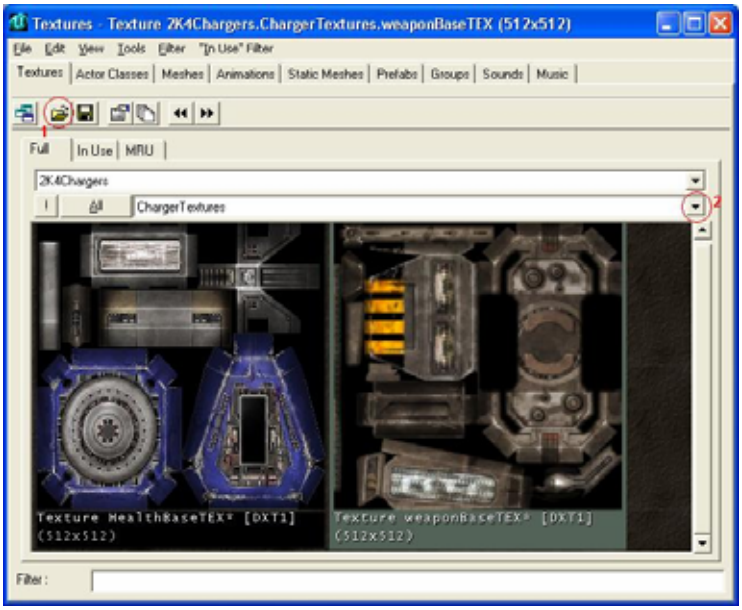



Figura A.13. Aplicando texturas.

- **Trabalhando com Malhas Estáticas**

Abre-se o *<Static Mesh Browser>* mostrado na *Figura A.11.* e usa-se o menu *<File>Import...>* como mostra a *Figura A.14.* Para importar o arquivo ASE desejado, será necessário escolher um nome para o pacote, o grupo e os objetos que serão importados, como mostra a *Figura A.15.*

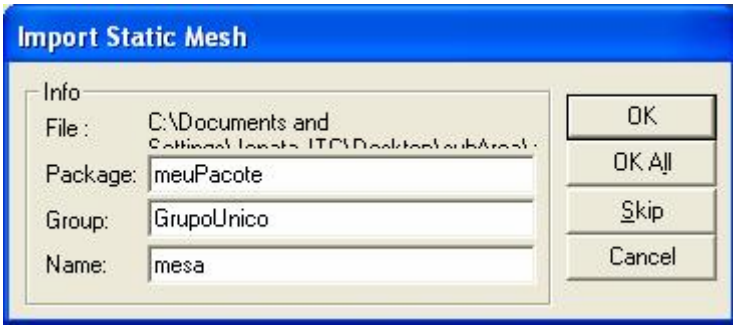



Figura A.14. Editando pacote de importação das malhas estáticas.

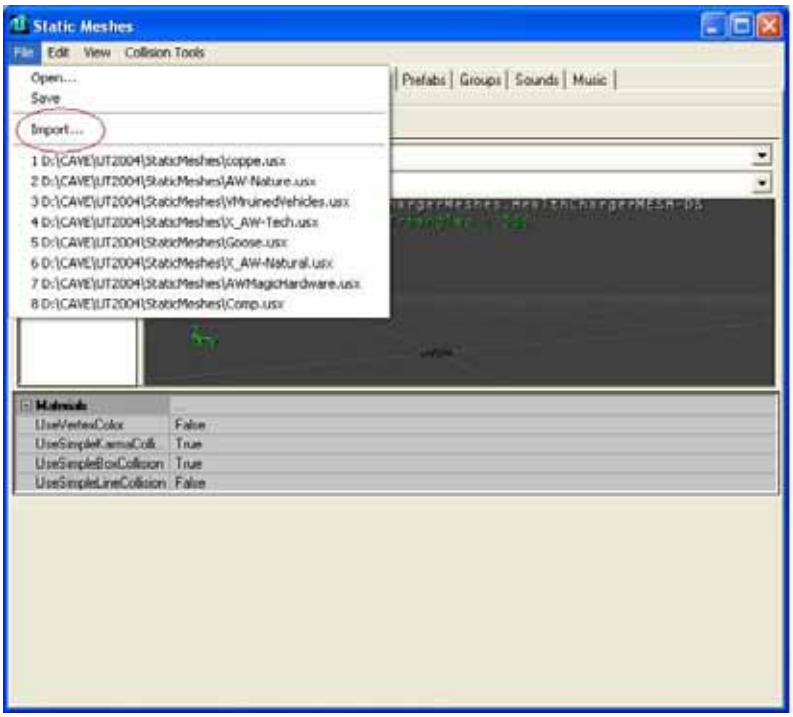



Figura A.15. Importando as malhas estáticas.

Feita a importação, vai-se novamente para o menu <File> e se salva o pacote recém criado. Logo após, para colocar uma malha estática no cenário, basta selecioná-la no <*Static Mesh Browser*>, ir para a área do cenário e clicar com o botão direito no local onde ficará a malha estática e selecionar a opção <*Add Static Mesh:*[nome da malha selecionada]>, conforme mostrado na *Figura A.16.*

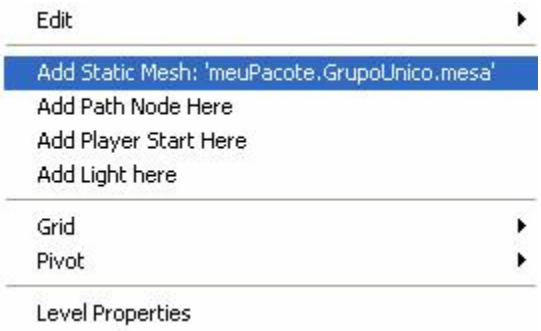



Figura A.16. Adicionando no cenário componentes estáticos.

Caso a malha/componente fique pequena no cenário, é possível aumentar a sua escala acessando as propriedades da malha através da tecla F4. Com a malha selecionada ou duplo clique sobre a malha, aumenta-se o valor da propriedade <*Display*>*DrawScale* como mostra a *Figura A.17.*

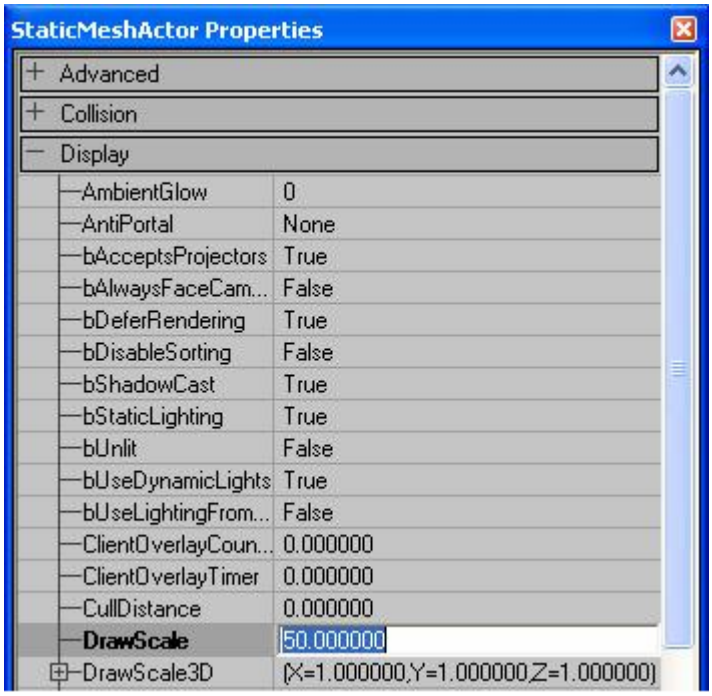



Figura A.17. Aumentando o tamanho da malha.

- **Trabalhando com Animações**

O processo para a utilização de animações no UnrealEd é um pouco mais trabalhoso. Precisa-se trabalhar com os atores (*Actors*) e definir suas ações (*Actions*) através de um script. Primeiramente é preciso adicionar <*xIntroPawn*>. Para isso é necessário clicar no ícone <*Actor Class Browser*> conforme *Figura A.11*. Em seguida seleciona-se <*Pawn*>*UnrealPawn*>*xIntroPawn*> como mostra a *Figura A.18.* Feito isso, basta clicar com o botão direito na área do cenário em que deverá ficar a animação e escolher <*add xIntroPawn*>, que a princípio não aparecerá, pois ainda não se selecionou a malha de animação. Para isso é necessário pressionar as teclas <ctrl + Z> e após a tecla F4 para abrir as propriedades do ator.

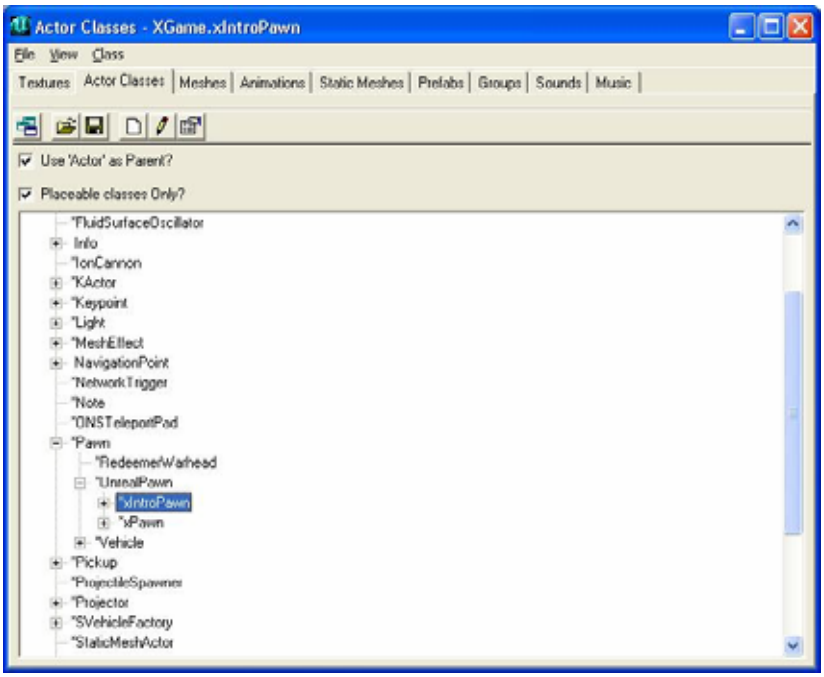



Figura A.18. Definindo atores e suas ações.

Torna-se então necessário importar a animação que foi previamente exportada, seleciona-se para isso o *<Animation Browser>*, conforme ícone 4 da *Figura A.11*. Importa-se primeiramente o esqueleto da malha *<Mesh import>* e logo após a animação *<Animation import>*, como mostra a *Figura A.19.* Note que será preciso dar um nome qualquer para o pacote, o grupo e o objeto que iremos importar. É importante não trocar o nome padrão que aparecerá.

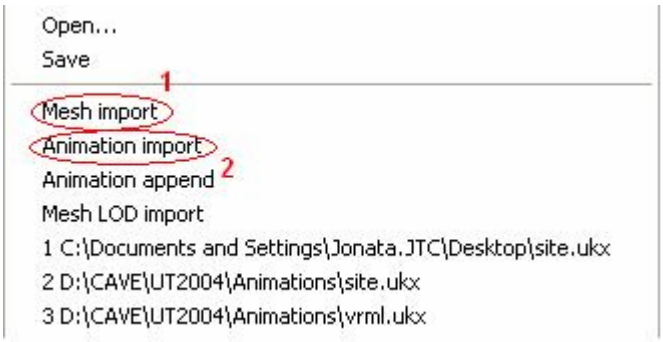



Figura A.19. Importando animação.

Depois disso será preciso clicar no menu *<Edit>linkup anim and mesh>* como mostra a *Figura A.20.* Feito isso salva-se o pacote.

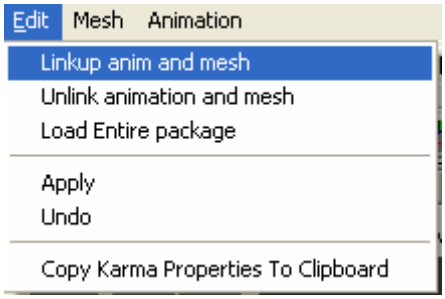



Figura A.20. Ligando a animação a malha estática.

Deixando o objeto selecionado que acabou de ser importado no <A*nimation Browser*>, voltam-se as propriedades do ator, conforme a *Figura A.21*. Clica-se no botão <*Use*> da propriedade *Mesh* e coloca-se o valor "script" na propriedade *AIScriptTag*. Caso a malha fique muito pequena no cenário, pode-se utilizar a propriedade "DrawScale" da mesma forma que em malhas estáticas.

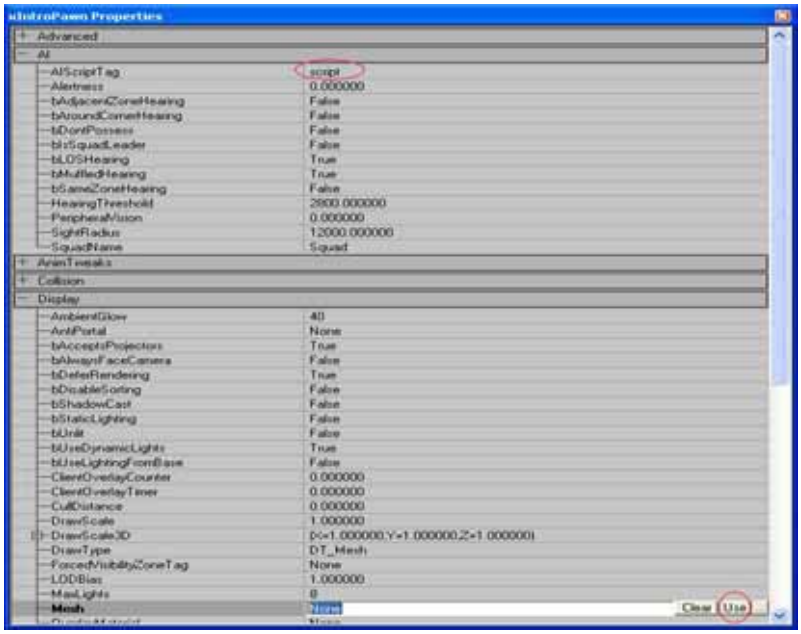


Figura A.21. Propriedades do ator.

É necessário então criar um script para definir o comportamento do ator. Para isso retorna-se ao ícone <A*ctor Class Browser*> e faz-se a seleção como mostra a *Figura A.22.*

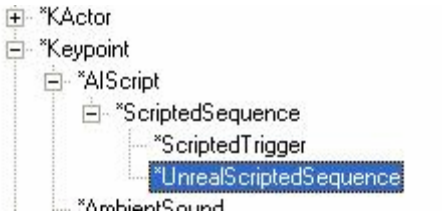



Figura A.22. Ações do ator.

Em seguida, clica-se com o botão direito do mouse na área do cenário onde ficará o script e seleciona-se *<Add>UnrealScriptedSequence>*. Logo após será preciso acessar as propriedades desse script e inserir uma nova ação selecionando *<Action_PLAYANIM>* e clicando no botão *<New>* que aparecerá no lado direito da ação.

Como mostra a *Figura A.23.*, é preciso colocar na propriedade "BaseAnim" o nome dado para a animação que foi importada e mais abaixo em "Events>Tag" colocar o valor "script".

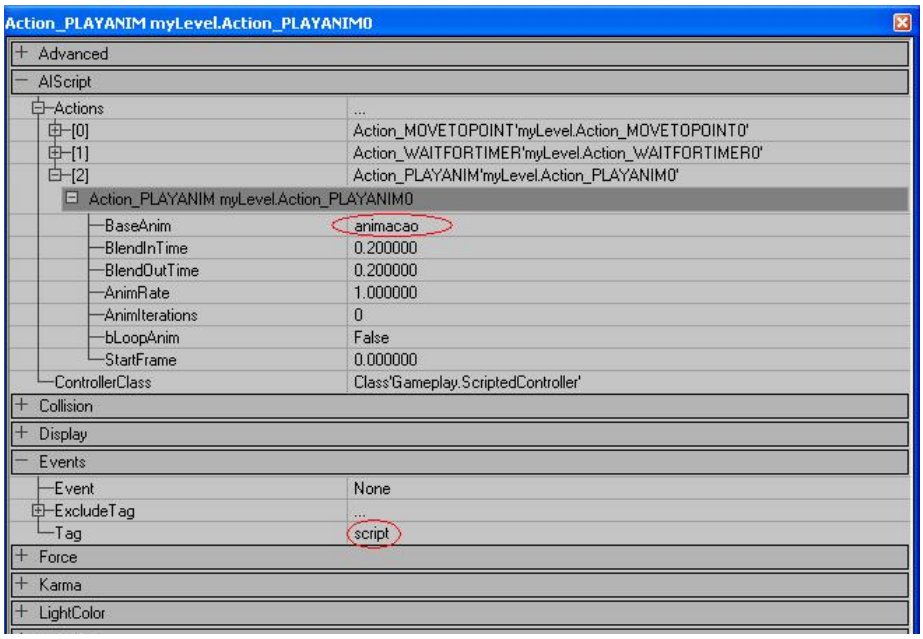



Figura A.23. Propriedades das ações do ator.

Para fazer a animação executar continuamente, basta reproduzir as propriedades que aparecem na *Figura A.24.*

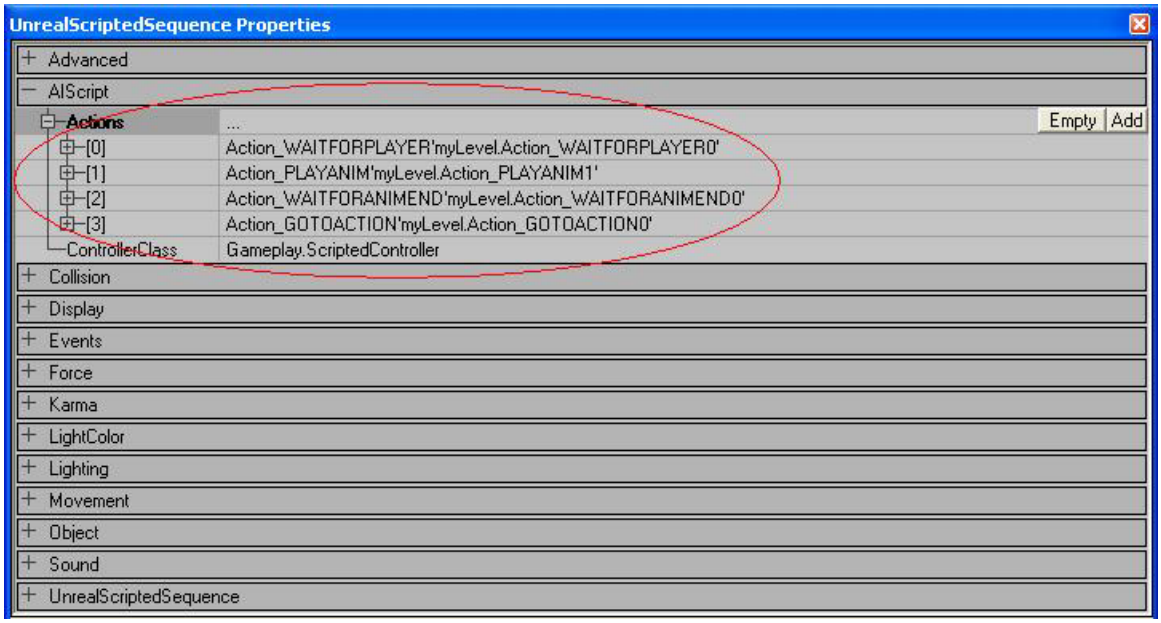



Figura A.24. Executando a animação.

Tendo-se executado os três passos de implementação deste manual: Passo1. Reduzindo os polígonos do modelo VRML com VizUp; Passo 2. Convertendo formatos com o 3DStudioMax(3DS) e Passo 3. Criando o ambiente virtual no UnrealEd tem-se então o cenário naval que será visualizado pela *engine* do UT conforme é descrito na *Sessão 5.3* da etapa de visualização.

## B – Códigos modificados.

São apresentados os códigos modificados da *engine* que permitem a visualização distribuída em *n* planos. O *CaveUT.u* é um código aberto que pode ser livremente alterado para diferentes implementações. O código *caveut.ini* são os parâmetros de configuração utilizados para a CAVE de dois planos implementada neste trabalho.

**a)** Código do jogo modificado (*CaveUT.u*)

```
//-----------------------------------------------------
//parâmetros de visualização em n planos
//-----------------------------------------------------

class Spectator extends xPlayer
	config(CaveUT);

var config float CaveFOV;
var config float CaveRoll;
var config float CavePitch;
var config float CaveYaw;
var config float CaveOffsetX;
var config float CaveOffsetY;
var config float CaveOffsetZ;

var config bool CaveSmoothing;
var int UncompressedViewRotation;
var rotator OriginalCameraRotation;

replication
{
	reliable if (Role==ROLE_Authority)
		UncompressedViewRotation;
}

//-----------------------------------------------------
// Cálculo de visualização de rotação da câmera funções (Jonge)
//-----------------------------------------------------

struct Matrix3
{
	var float m11;
	var float m12;
	var float m13;

	var float m21;
	var float m22;
	var float m23;

	var float m31;
	var float m32;
	var float m33;
};

function Matrix3 Euler2Matrix( rotator rot )
{
	local Matrix3 R;

	local float a1;
	local float a2;
	local float a3;

	local float c1,s1;
	local float c2,s2;
	local float c3,s3;

	a1 = Unreal2Rad(rot.roll);
```

```
        a2 = Unreal2Rad(rot.pitch);
        a3 = Unreal2Rad(rot.yaw);

        c1 = cos(a1);
        s1 = sin(a1);

        c2 = cos(a2);
        s2 = sin(a2);

        c3 = cos(a3);
        s3 = sin(a3);

        R.m11 = c2*c3;
        R.m12 = c3*s1*s2  - c1*s3;
        R.m13 = -c1*c3*s2 - s1*s3;

        R.m21 = c2*s3;
        R.m22 = c1*c3 + s1*s2*s3;
        R.m23 = c3*s1 - c1*s2*s3;

        R.m31 = s2;
        R.m32 = -c2*s1;
        R.m33 = c1*c2;

        return R;
}

function rotator Matrix2Euler( Matrix3 R )
{
        local rotator rot;

        local float a1;
        local float a2;
        local float a3;
        local float c2;

        c2 = sqrt( R.m11*R.m11 + R.m21*R.m21 );

        if (c2>0.00000001)
        {
                a1 = Atan( -R.m32, R.m33 );
                a2 = Atan(  R.m31, c2 );
                a3 = Atan(  R.m21, R.m11 );
        }
        else
        {
                a1 = Atan( R.m23, R.m22 );
                a2 = Atan( R.m31, c2 );
                a3 = 0;
        }

        rot.roll  = Rad2Unreal(a1);
        rot.pitch = Rad2Unreal(a2);
        rot.yaw   = Rad2Unreal(a3);

        return rot;
}

final static function float Unreal2Rad(int u)
{
        local float r;
        r = (float(u)*PI)/32768.0;
        return r;
}

final static function int Rad2Unreal(float r)
{
        local int u;
        u = int( (r*32768.0)/PI );
        return u;
}

final static function int Deg2Unreal(float d)
{
  local float value;
  value = d * 182.0444444444444444444444444444;
  if(value < 0) value = 65536.0 + value;
```

```
  return round(value);
}

final static function Matrix3 MulMatrix( Matrix3 A, Matrix3 B)
{
        local Matrix3 M;

        M.m11 = A.m11*B.m11 + A.m12*B.m21 + A.m13*B.m31;
        M.m12 = A.m11*B.m12 + A.m12*B.m22 + A.m13*B.m32;
        M.m13 = A.m11*B.m13 + A.m12*B.m23 + A.m13*B.m33;

        M.m21 = A.m21*B.m11 + A.m22*B.m21 + A.m23*B.m31;
        M.m22 = A.m21*B.m12 + A.m22*B.m22 + A.m23*B.m32;
        M.m23 = A.m21*B.m13 + A.m22*B.m23 + A.m23*B.m33;

        M.m31 = A.m31*B.m11 + A.m32*B.m21 + A.m33*B.m31;
        M.m32 = A.m31*B.m12 + A.m32*B.m22 + A.m33*B.m32;
        M.m33 = A.m31*B.m13 + A.m32*B.m23 + A.m33*B.m33;

        return M;
}

function rotator GetCaveViewRotation( rotator camrot )
{
        local rotator ViewRotation;
        local Matrix3 Rcam;
        local Matrix3 Rcave;
        local Matrix3 R;
        local rotator caverot;

        // Assign values from CaveUT.ini value, and convert to Unreal degrees
        caverot.yaw   = Deg2Unreal(CaveYaw);
        caverot.pitch = Deg2Unreal(CavePitch);
        caverot.roll  = Deg2Unreal(-CaveRoll);

        Rcam  = Euler2Matrix(camrot);

        Rcave = Euler2Matrix(caverot);

        R = MulMatrix(Rcam, Rcave);

        ViewRotation = Matrix2Euler(R);

        return ViewRotation;
}

//--------------------------------------------------
// fim do cálculo de rotação
//--------------------------------------------------

function string GetCaveRotation() {
  return "CavePitch " $ CavePitch $ " CaveRoll " $ CaveRoll $ " CaveYaw " $ CaveYaw;
}

function string GetCaveLocation() {
  return "CaveOffsetX " $ CaveOffsetX $ " CaveOffsetY " $ CaveOffsetY $ " CaveOffsetZ " $
CaveOffsetZ;
}

function string GetCaveFOV() {
  return "CaveFOV " $ CaveFOV;
}

function SetX(float X) {
  CaveOffsetX += X;
  SaveConfig();
}

function SetY(float Y) {
  CaveOffsetY += Y;
  SaveConfig();
}

function SetZ(float Z) {
  CaveOffsetZ += Z;
  SaveConfig();
}
```

```
function SetPitch(float P) {
  CavePitch += P;
  SaveConfig();
}

function SetRoll(float R) {
  CaveRoll += R;
  SaveConfig();
}

function SetYaw(float Y) {
  CaveYaw += Y;
  SaveConfig();
}

function SetFOV(float FOV) {
  CaveFOV += FOV;
  SaveConfig();
}

function int PackRotator( rotator CameraRotation )
{
    CameraRotation.Yaw = (CameraRotation.Yaw + 65536) & 65535;
    CameraRotation.Pitch = (CameraRotation.Pitch + 65536) & 65535;
    return (CameraRotation.Yaw << 16) + CameraRotation.Pitch;
}

function rotator UnpackRotator( int Packed )
{
    local rotator NewCameraRotation;
    NewCameraRotation.Yaw = Packed >> 16;
    NewCameraRotation.Pitch = Packed & 65535;
    return NewCameraRotation;
}


// Get the PTarget, and then adjust all the camera location and rotation values
// Do these camera adjustments in pretty much all cases, even when we aren't a spectator
    PTarget = Pawn(ViewTarget);
    if ( PTarget != None && !bBehindView ) { // This should always be true
      CameraRotation = GetCaveViewRotation(CameraRotation);
      CameraLocation += PTarget.EyePosition();
      CameraLocation += vector(PTarget.GetViewRotation()) * vect(-10,-10,-20);
      CameraLocation.X += CaveOffsetX;
      CameraLocation.Y += CaveOffsetY;
      CameraLocation.Z += CaveOffsetZ;
    }
}

class CaveUTMut extends Mutator
  notplaceable;

var bool bHasInteraction;

function ModifyLogin(out string Portal, out string Options) {
  Level.Game.PlayerControllerClassName = "CaveUT.Spectator";

  Level.Game.PlayerControllerClass =
    Class<PlayerController>(DynamicLoadObject(Level.Game.PlayerControllerClassName,
Class'Class'));

  Super.ModifyLogin(Portal, Options);
}

function Class<Weapon> GetDefaultWeapon() {
  return None;
}


simulated function Tick(float DeltaTime)
{
    local PlayerController PC;

    // If the player has an interaction already, exit function.
    if (bHasInteraction) {
      Return;
```

```
    }

    PC = Level.GetLocalPlayerController();

    if(PC == None) {
      Return;
    }

// Run a check to see whether this mutator should create an interaction for the player
    PC.Player.InteractionMaster.AddInteraction("CaveUT.CaveUTInteraction", PC.Player); //
Create the interaction
    bHasInteraction = True; // Set the variable so this lot isn't called again
    log("CaveUT: Interaction Added");


    // hide all of the HUD display, but leave the code intact since we play nice that way
    if(PC.myHUD.bCrosshairShow )
      PC.ToggleScreenShotMode(); // (also hides the weapon)

    Disable('Tick');

}

function ModifyPlayer(Pawn Other)
{
  if(Other.Weapon != None) {
    Other.Weapon.Destroy();
    Other.Weapon = None;
  }
  Super.ModifyPlayer(Other);
}

function string GetInventoryClassOverride(string InventoryClassName)
{
  return "";
}

class CaveUTInteraction extends Interaction
        config(CaveUT)
        notplaceable;

var protected LevelInfo Level;
var private bool bPressedShift;
var private bool bPressedCtrl;
var private bool bShowDebug;

var config float RotateIncrement;
var config float OffsetIncrement;
var config float FOVIncrement;

event Initialized() {
  log("Interaction Initialized");
  Level = ViewportOwner.Actor.Level;
}

function PostRender(Canvas Canvas) {
  local vector location;
  local rotator rotation;
  local vector SizeCharacter;
  local vector LocationScreenHelp;
  local Spectator S;

  if(!bShowDebug) return;

  S = Spectator(ViewportOwner.Actor);

  Canvas.Style = 1;  // STY_Normal;
  Canvas.SetDrawColor(255,255,255,255);
  Canvas.Font = Canvas.TinyFont;
  Canvas.TextSize("X", SizeCharacter.X, SizeCharacter.Y);

  LocationScreenHelp.X = 0;
  LocationScreenHelp.Y = 0;

  Canvas.SetPos(LocationScreenHelp.X, LocationScreenHelp.Y);
  Canvas.GetCameraLocation(location, rotation);
  Canvas.DrawText("Rotation: " $ rotation $ " Location: " $ location);
```

```
  Canvas.CurY += SizeCharacter.Y * 1.5;
  Canvas.DrawText(S.GetCaveRotation());
  Canvas.CurY += SizeCharacter.Y * 1.5;
  Canvas.DrawText(S.GetCaveLocation());
  Canvas.CurY += SizeCharacter.Y * 1.5;
  Canvas.DrawText(S.GetCaveFOV());
}

function bool KeyEvent(out EInputKey InputKey, out EInputAction InputAction, float Delta) {
  local Spectator S;

  S = Spectator(ViewportOwner.Actor);

  if (InputKey == IK_Shift)
    if (InputAction == IST_Press)
      bPressedShift = True;
    else if (InputAction == IST_Release)
      bPressedShift = False;

  if (InputKey == IK_Ctrl)
    if (InputAction == IST_Press)
      bPressedCtrl = True;
    else if (InputAction == IST_Release)
      bPressedCtrl = False;

  if (bPressedShift && InputAction == IST_Press) {
    switch (InputKey) {
    case IK_A: S.SetX(OffsetIncrement); break;
    case IK_S: S.SetY(OffsetIncrement); break;
    case IK_D: S.SetZ(OffsetIncrement); break;
    case IK_F: S.SetFOV(FOVIncrement); break;

    case IK_Z: S.SetPitch(RotateIncrement); break;
    case IK_X: S.SetRoll(RotateIncrement); break;
    case IK_C: S.SetYaw(RotateIncrement); break;

    case IK_H: bShowDebug = true; break;
      }

    return True;
    }

  if (bPressedCtrl && InputAction == IST_Press) {
    switch (InputKey) {
    case IK_A: S.SetX(-OffsetIncrement); break;
    case IK_S: S.SetY(-OffsetIncrement); break;
    case IK_D: S.SetZ(-OffsetIncrement); break;
    case IK_F: S.SetFOV(-FOVIncrement); break;

    case IK_Z: S.SetPitch(-RotateIncrement); break;
    case IK_X: S.SetRoll(-RotateIncrement); break;
    case IK_C: S.SetYaw(-RotateIncrement); break;

    case IK_H: bShowDebug = false; break;
      }

    return True;
    }

  return Super.KeyEvent(InputKey, InputAction, Delta);
  }
```

**b)** Parâmetros de configuração das duas telas (*CaveUT.ini*)

```
[CaveUT.Spectator]
CaveRoll=0.000000
CavePitch=0.000000
CaveYaw=0.000000
CaveOffsetX=0.000000
CaveOffsetY=0.000000
CaveOffsetZ=0.000000
CaveFOV=90.000000
OffAxis=Yes
FOVleft=-45.0
FOVright=45.0
FOVtop=36.86
```

```
FOVbottom=-36.86
CaveSmoothing=False
bDebuggingVoiceChat=False
bAutoDemoRec=False
EnemyTurnSpeed=45000
InputClass=Class'Engine.PlayerInput'
MidGameMenuClass=GUI2K4.UT2K4DisconnectOptionPage
DemoMenuClass=GUI2K4.UT2K4DemoPlayback
AdminMenuClass=GUI2K4.RemoteAdmin
ChatPasswordMenuClass=GUI2K4.UT2K4ChatPassword

[CaveUT.CaveUTInteraction]
RotateIncrement=1.000000
OffsetIncrement=5.000000
FOVIncrement=5.000000
 [XGame.xPlayer]
bClassicTrans=False
[UnrealGame.UnrealPlayer]
CustomizedAnnouncerPack=
CustomStatusAnnouncerPack=UnrealGame.FemaleAnnouncer
CustomRewardAnnouncerPack=UnrealGame.MaleAnnouncer
MaxRecentServers=25
bDontShowLoginMenu=False
RecentServers=192.168.1.33:7777
RecentServers=192.168.1.108:7777
RecentServers=192.168.1.20:7777
RecentServers=66.219.41.218:7777
RecentServers=192.168.1.112:7777
RecentServers=192.168.1.100:7777
RecentServers=192.168.1.101:7777
RecentServers=192.168.1.106:7777
RecentServers=127.0.0.1:7777

[Engine.PlayerController]
bLookUpStairs=False
bSnapToLevel=False
bAlwaysMouseLook=True
bKeyboardLook=True
bNeverSwitchOnPickup=False
bAlwaysLevel=False
bNoVoiceMessages=False
bNoTextToSpeechVoiceMessages=True
bNoVoiceTaunts=False
bNoAutoTaunts=False
bAutoTaunt=False
bNoMatureLanguage=False
bDynamicNetSpeed=True
bSmallWeapons=True
bLandingShake=True
bAimingHelp=False
bEnablePickupForceFeedback=False
bEnableWeaponForceFeedback=False
bEnableDamageForceFeedback=False
bEnableGUIForceFeedback=False
bEnableInitialChatRoom=True
bEnableStatsTracking=False
bOnlySpeakTeamText=False
AutoJoinMask=5
AnnouncerLevel=2
AnnouncerVolume=1
TextToSpeechVoiceVolume=1.000000
DesiredFOV=90.000000
DefaultFOV=90.000000
ScreenFlashScaling=1.000000
TimeMarginSlack=0.001000
StatsUsername=
StatsPassword=
DynamicPingThreshold=400.000000
LastActiveChannel=
VoiceChatCodec=CODEC_48NB
VoiceChatLANCodec=CODEC_96WB
ChatPassword=
DefaultActiveChannel=

[Engine.Controller]
Handedness=2.000000
```

**REFERÊNCIAS**


3DStudioMax (2004), World Wide Web, http://www4.discreet.com/3dsmax Acessado em: fev 2006.

ActorX (2006), *"ActorX plugins for 3DS Max"*, World Wide Web, http://udn.epicgames.com/Two/StaticMeshes Acessado em: ago2006.

ALTERNE, Consortium (2004) "*Alterne – Alternative Realities in Network Environment*" World Wide Web, http://www.alterne.info. Acessado em: mar 2006.

Araújo, R. B.(1996), "*Especificação e análise de um sistema distribuído de Realidade Virtual*", São Paulo, Junho, 144, p.p. Tese de Doutorado, Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da Universidade de São Paulo.

ASE (2004), "*AscII Scene Export*" World Wide Web, http://wiki.beyondunreal.com/wiki/ASE_File_Format Acessado: dez 2006.

Aukstakalnis, S. and Blatner, D. (1992), "Silicon Mirage: The Art and Science of Virtual Reality", Peatpich Press Berkeley, CA.

AutoDesk (1985), World Wide Web, http://www.autodesk.com, Acessado: fev 2006.

Barboza, T. (2005), "*O atual cenário da construção naval civil e militar no mundo, incluindo o subcenário brasileiro.*" World Wide Web, http:// www.emgepron.mar.mil.br/cenario_construcao_naval.pdf. Acessado em: jul 2006.

Bastos, N. C. (2005), "*Arquitetura para Dispositivos Não-Convencionais de Interação Utilizando Realidade Aumentada*" , Monografia do curso Ciência da Computação – Universidade Federal de Pernambuco.

Bastos, Thiago; Silva, Romano; Raposo Alberto; Gattass, Marcelo. (2004) "*ViRAL:Um Framework para o Desenvolvimento de Aplicações de Realidade Virtual"*, VII Symposium on Virtual Reality pp. 51-62.

Baxter III, W.V., Sud, A., Govindaraju, N.K. and Manocha, D. (2002) "*GigaWalk: Interactive WalkThrough of Complex Environments*". Eurographics Workshop on Rendering.

Benson, D. (1997) *"Simulation Modeling And Optimization Using Promodel"*, Proceedings of the 1997 Winter Simulation Conference, Volume , Issue , 7-10 Dec 1997 pp. 587 - 593

Bessant, C. B. (1986), "*CAD/CAM – Projeto e Fabricação com Auxílio de Computador*", Editora Campos Rio de Janeiro.

Bierbaum, A. (2000) *"VR Juggler: A Virtual Platform for Virtual Reality Application Development"*, Master Thesis, Iowa State University.

Binotto, A.; Brunetti, G.; Gierlinger, T.; Pereira, C.; Santos, P.; Stork, A.(2006) "*Computer Graphics Applications in Virtual Engeneering*", Proceedings of Virtual Concept 2006 Playa Del Carmen, Mexico, – December, 2006.

Bowskill, J. and Downie, J. (1995) "*Extending the Capabilities of the Human Visual System: An Introduction to Enhanced Reality*", Computer Graphics - ACM, 29(2):61-65, May 1995.

Brunetti, G. Amicis, R. and Rix, J. "*Virtual Engineering and Using Virtual Reality for Engineering."* In Associazone Nazionale Disegno di Macchine: International Conference on Design Tools and Methods in Industrial Engineering, 1999.

Bylund, M., Espinoza, F. (2002) *"Testing and demonstrating context aware services with Quake III Arena"*, Communications of the ACM Volume 45 , Issue 1 (January 2002) pp. 46 - 48

Carpin, S., Birk, A.(2005) *"Stochastic map merging in rescue environments"*. In: Robocup 2004: Robot Soccer World Cup VIII. Springer

Carpin, S., Wang, J., Lewis, M., Birk, A. , Jacoff, A. (2006) *"High fidelity tools for rescue robotics: results and perspectives,"* in Robocup 2005: Robot Soccer World Cup IX, ser. LNCS, 2006, pp. 301–311.

Castelltort, X. (1988), "*CAD/CAM Metodologia e Aplicações Práticas*", Editora McGraw-Hill, São Paulo.

CaveUT (2004), World Wide Web, http://www.planetjeff.net/ut/CaveUT.html Acessado em: abr 2006.

Corseuil, E. T. L., Raposo, A. B. et al. (2004): *"ENVIRON – Visualization of CAD Models In a Virtual Reality Environment."* In Eurographics Symposium on Virtual Environments (EG-VE 2004) Grenoble, France , pp. 79-82.

Corseuil, E. T. L., Raposo, A. B. et al. (2003): *"Buscando o Uso Operacional de Realidade Virtual em Grande Modelos de Engenharia "*, SVR 2003 - VI Symposium on Virtual Reality, Ribeirão Preto, SP, Brasil, Out. 2003, pp.187-198.

Cruz-Neira, C.; D.J. Sandin, T.A. DeFanti R.V. Kenyon and J.C. Hart, (1992) "*The Cave Automatic Virtual Environment*", Communications of the ACM, 35(2): 64-72, June 1992.

Dai, F; Reindl, P.(1996) , "*Enabling digital mock up with virtual reality techniques-vision, concept, demonstrator*", Proceedings of 1996 ASME Design Engineering Technical Conference and Computer in Engineering. Irvine, California.

Dassault Systemes (2005) *"Product Lifecycle Management Solutions"*, World Wide Web, http://www.3ds.com/products-solutions/ Acessado: dez 2005.

Deep, Exploration (1999) *"Right Hemisphere - Visual Product Communication and Collaboration"* World Wide Web, http://www.righthemisphere.com/products/dexp/ , Acessado: mai 2005.

Delmia (2002) "*DELMIA VIRTUAL. NC.*", World Wide Web, http://www.delmia.com/. Acessado: jul 2005.

Dimension Technologies, Inc. (2004) "*DTI – The 3-Dimensional Technology Specialists*". World Wide Web, http://www.dti3d.com/ Acessado: jul 2006.

Division Reality (2003), *"Parametric Technology Corporation – PTC Division Reality"*, World Wide Web, http://www.ptc.com. Acessado: dez 2005.

Ferreira, A. G. (1999) *"Uma arquitetura para visualização distribuída de ambientes virtuais"*, Dissertação de mestrado, PUC-Rio - Pontifícia Universidade Católica do Rio de Janeiro.

Hammann, J. E. and Markovitch, N. A. (1995) *"Introduction to Arena"*, Proceedings of the 1995 Winter Simulation Conference, pp. 519-523

Intersense (2004), World Wide Web, http://www.isense.com. Acessado: dez 2005.

Jacobson, Jeffrey (2004). "*Planett Jeff*", World Wide Web, http://www.planetjeff.net/, Acessado: dez 2005.

Kaminka, G. A., Veloso, M.M., Schaffer, S., Sollitto, R.(2002) , *"GameBots: A flexible test bed for multiagent team"*, Communications of the ACM, Volume 45, pp. 43-45.

Kim, Hongtae ; Lee, Jong-Gap; Lee, Soon-Sup; Park, Jin. H.(2003a) *"A Simulation-based Shipbuilding System for Evaluation of Validity in Design and Manufacturing"*; IEEE International Conference on Systems, Man and Cybernetics, 2003; Volume: 1, pp. 522- 529

Kim, Hongtae; Lee, Jong-Kap; Park, Jin. H., Park, Beom-J.; Jang, Dong-Sik (2002b) "*Applying digital manufacturing technology to ship production and the maritime environment*"; Journal: Integrated Manufacturing Systems, ISSN: 0957-6061, Year: Aug 2002 Volume: 13 Issue: 5 Page: 295 - 305

Kirner, C. and Tori, R. (2004), "*Realidade Virtual: Conceitos e Tendências*", Pré Simpósio SVR2004.

Kosa, Korea Iron and Steel Association (2005) World Wide Web, http:// www.kosa.or.kr/ Acessado: dez 2005.

Laird, J.E. (2001), *"Using a computer game to develop advanced AI"*, IEEE Computer, Volume 34, pp.70-75.

Latta, J. N. and Oberg, D. J. (1994). "*A conceptual virtual reality model*". IEEE Computer Graphics and Applications pp. 23-29, Jan.

Machover, C., Tice, S. E. (1994) "Virtual reality". IEEE Computer Graphics and Applications.

Marcicano, P. (2003), "*Introdução ao Controle Numérico*" , World Wide Web, http:// www.poli.usp.br . São Paulo. Acessado: mar 2006.

McAllister, D. F. (2002) "*Stereo and 3-D Display Technologies*", In Encyclopedia of Imaging Science and Technology, John Wiley & Sons, p. 1327-1344.

Mota, S. (2002), "*Controle Numérico*", World Wide Web, http://www.poli.usp.br . São Paulo. Acessado: dez 2006.

MR Toolkit, M. (2006), World Wide Web, http://web.cs.ualberta.ca/graphics/mrtoolkit.html. Acessado: set 2006.

Okino PolyTrans (1991), World Wide Web, http://www.okino.com Acessado: dez 2005.

Osfield, R. and Burns, D. (2003), "*Open Scene Graph*", World Wide Web, http://www.openscenegraph.org. Acessado: dez 2005.

Pessôa, M and Spinola, M. (2005) "*Sistemas CAD/CAE/CAM - Integração da manufatura*" - Apostila da Escola Politécnica da Universidade de São Paulo PRO - Departamento de Engenharia de Produção, World Wide Web, http://www.prd.usp.br/ Acessado: jul 2006.

Piekarski, W., Thomas B.(2002), *"ARQUAKE: the outdoor augmented reality gaming system"*, Communications of the ACM Volume 45 , pp. 36 - 38

Pimentel, K.; Teixeira, K. (1995) "*Virtual Reality -Through the New Looking Glass*". 2.ed. New York, McGraw-Hill.

Pires, F.; Estefen, S. (2005) "*Construção Naval no Brasil*". Seminário Construção Naval: Tecnologia e Gestão – COPPE UFRJ. World Wide Web, http://www.oceanica.ufrj.br. Acessado: mar 2006.

Quake (1997) "*ID Software*" World Wide Web, http://www.idssoftware.com . Acessado: abr 2006.

Ribeiro, D. Rosenfeld H. (1999), "*CAM Computer Aided Manufacturing*" , World Wide Web, http://www.numa.org.br. Acessado: mar 2006.

Sense8, "*World Up*", World Wide Web, http://www.sense8.com. Acessado: abr 2005.

Sherman, W. R. and Craig, A. B., (2003) "*Understanding Virtual Reality: Interface, Application, and Design"* Morgan Kaufmann.

Shiratuddin, M.F. and Thabet, W. (2002) "*Virtual Office Walkthrough Using a 3D Game Engine*", International Journal of Design Computing, 4, ISSN 1329-7147.

Silva, Andre Koide (2006), "*Método para Avaliação e seleção de softwares de simulação de eventos discretos aplicados à análise logística*", Escola Politécnica de São Paulo. Dissertação de mestrado.

Silva, R.J.M.; Raposo, A.B.; Gattas, M. (2004) "*Grafo de cena e Realidade Virtual"*, Monografia em Ciência da Computação, ISSN 0103-9741, PUC-Rio - Pontifícia Universidade Católica do Rio de Janeiro.

Unreal, Tournament (2004), World Wide Web, http://www.unrealtournament.com. Acessado: dez 2005.

UnrealED (2000), "*Epic Games Editor Unreal*", World Wide Web, http://www.unrealwiki.com. Acessado: mar 2006.

VizUp Technology, "*Vizup*", World Wide Web, http://www.vizup.com/ Acessado: jun 2006.

VRML (2000), "VRML – *Virtual Reality Modeling Language*", World Wide Web, http://www.vrmlsite.com. Acessado: jan 2006.

Walkinside(2006). "*A visualization tool for viewing complex 3D CAD models in real time*". World Wide Web, http://www.walkinside.com Acessado: mar 2006.

Zerbone, E. (1995), "*Tecnologias informáticas na implementação do CIM*" Editora Campus Ltda, Rio de Janeiro. Manufatura Integrada por computador – Sistemas integrados de produção: Estratégia, Organização, Tecnologia e Recursos Humanos.

Zuffo, J. A; Soares, L. P.; Zuffo M.K.;Lopes-R.D (2001). "*Caverna Digital – Sistema de Multiprojeção Estereoscópico Baseado em Aglomerados de PCs para Aplicações Imersivas da Realidade Virtual*", 4th SBC Symposium on Virtual Reality pp. 139-147. Florianópolis, Outubro de 2001.