FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”
CENTRO UNIVERSITÁRIO “EURÍPIDES DE MARÍLIA” – UNIVEM
CURSO DE CIÊNCIA DA COMPUTAÇÃO

MARCELO HUGO ROMEU DIAS

MIRELLA SILENE BASTIANIK

FERRAMENTA PARA VISUALIZAÇÃO E CONVERSÃO DE ARQUIVOS VRML E DXF

MARÍLIA

2006

MARCELO HUGO ROMEU DIAS

MIRELLA SILENE BASTIANIK

FERRAMENTA PARA VISUALIZAÇÃO E CONVERSÃO DE ARQUIVOS VRML E DXF

Monografia apresentada ao curso Bacharelado Ciência da Computação do Centro Universitário Eurípides de Marília, mantido pela Fundação de ensino Eurípides Soares da Rocha, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador:
Prof. Dr. José Remo Ferreira Brega

MARÍLIA
2006

MIRELLA SILENE BASTIANIK

MARCELO HUGO ROMEU DIAS

FERRAMENTA PARA VISUALIZAÇÃO E CONVERSÃO DE ARQUIVOS VRML E DXF

Banca examinadora da monografia apresentada ao Curso de Bacharelado em Ciência da Computação do "UNIVEM/F.E.E.S.R", para obtenção do Título de Bacharel em Ciência da Computação.

Nota: _____(_____)

ORIENTADOR:. José Remo Ferreiro Brega

1º EXAMINADOR: _____

2º EXAMINADOR: _____

Marília, ____de _____de 2006.

AGRADECIMENTOS

“O maior pecado do ser humano é ignorar sua força interior, seu poder criador e sua herança divina. Agradecemos a Deus, que se fez presente em todos os momentos, a oportunidade de iniciarmos a operacionalização de nossas fábricas de sonhos e ilusões.”

Aos nossos pais presentes e ausentes, familiares e amigos queridos, que em todo momento colaboraram para a conclusão de um objetivo profissional.

Ao Prof. Dr. José Remo Ferreira Brega, que com muita sabedoria nos conduziu à realização deste trabalho.

BASTIANIK, Mirella Silene; DIAS ROMEU Marcelo Hugo **Desenvolvimento de uma Ferramenta para visualização e conversão de arquivos DXF e VRML.** 2006. Monografia (Conclusão de Curso em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

RESUMO

Apresentar um procedimento para visualização de dados de projeto de engenharia em Realidade Virtual. Tendo em vista que a Realidade Virtual (RV) tem sido cada vez mais utilizada na engenharia e áreas afins. Neste trabalho apresenta-se estudo sobre o estado da arte em Realidade Virtual, voltado ao desenvolvimento de uma ferramenta de visualização de edificação na área de Engenharia, baseado sobre ferramentas CAD. Os Projetos arquitetônicos utilizam também diversas técnicas de RV para aperfeiçoar a visualização dos Projetos.

Palavras-chave: Engenharia, CAD, Realidade Virtual, Visualização.

BASTIANIK, Mirella Silene; DIAS ROMEU Marcelo Hugo. **Desenvolvimento de uma Ferramenta para visualização e conversão de arquivos DXF e VRML.** 2006. Monografia (Conclusão de Curso em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

ABSTRACT

To present a procedure for visualization project data of engineering in Virtual Reality (R.V.). Where Virtual Reality (RV) has been each time more used in engineering and similar areas. In this work we present studies on the state of the art in Virtual Reality, come back to the development of a tool of visualization of construction in the area of Engineering, based on tools CAD. The Projects architectural also use diverse techniques of R.V. to perfect the visualization of the Projects.

Keywords: Engineering, CAD, Realidade Virtual, Visualization

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

2D: Bidimensional

3D: Tridimensional

AV: Ambiente Virtual

CAD/CAM: Projeto e Manufatura Auxiliados por Computador

CAD: *Computer Aided Design*: Projeto Auxiliado / Assistido por Computador

CAE: Engenharia Auxiliada / Assistida por Computador

CAM: Manufatura Auxiliada / Assistida por Computador

CAP: *Computer Aided Planning*

CAVE: *Cave Automatic Virtual Environment*

CRT: Tubos de Raios Catódicos

DNS: *Domain Name System*

DXF: *Data Interchange Format*

E/S: Entrada e Saída

EIA: *Eletronics Industry Association*

GM: General Motors

HMD: *Head Mounted Display*

HUD: *Heads-up-displays*

IEC: *International Eletrotechnical Commission*

LCD: *Crystal Liquid Display* – Visores de Cristal Líquido

MIT: *Massachusetts Institute of Technology*

MRP: Material *Requesting Planning*: Planejamento de Requisição de Materiais

NETBEUI: *Network Basic End User Interface*

RV: Realidade Virtual

SGBD: Sistema Gerenciador de Banco de Dados

UK: Inglaterra

VDU: *Visual Display Unit*: unidades de exibição visual

VRML: *Virtual Reality Modeling Language*

X3D: *Extensible 3D*

XML: *Extensible Markup Language*

XNS: *Xerox Network Systems*

SUMÁRIO

PROPOSTA E DESCRIÇÃO DO PROJETO

Criação de uma Ferramenta de Visualização e Conversão de arquivos DXF e VRML.

Objetivo

Desenvolver um aplicativo que permita a visualização de arquivos DXF e VRML, convertendo arquivos DXF no formato VRML. Este aplicativo será capaz de trabalhar com plantas baixas desenvolvidas no AutoCAD.

Metodologia

Na realização deste trabalho, foram empregados diversos métodos, técnicas e conceitos para o desenvolvimento da ferramenta de visualização e conversão de arquivos DXF e VRML, descritos abaixo:

- Estudo de Realidade Virtual;
- Estudo do arquivo DXF;
- Estudo de AutoCad;
- Importação de formatos de arquivos diferentes;
- Desenvolvimento da Interface;
- Conversão em 3D;
- Testes Teóricos e Práticos.
- Testes de compatibilidade do projeto final a ser gerado com algumas ferramentas de auxílio a projetos (CADs);
- Geração da Documentação Final;

Os procedimentos acima propostos conduziram a resultados satisfatórios para este projeto.

Plano de Trabalho

Foi definido um plano de trabalho de acordo com as necessidades e dificuldades esperadas para sua implementação e execução do projeto.

O plano é composto pelas seguintes tarefas:

Tarefa 01: Estudo e Planejamento

Tarefa 02: Desenvolvimento e Implementação do Projeto

Tarefa 03: Testes, Análise e Correções da Implementação do Projeto

Tarefa 04: Apresentação e Defesa da Dissertação para o grau de Bacharel em Ciência da Computação.

Cronograma

O cronograma descreve as tarefas que foram executadas de acordo com o Plano de Trabalho proposto.

Cronograma de desenvolvimento:										
Atividade	**Tempo previsto**									
	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov
1. Atividade 1	■									
2. Atividade 2	■									
3. Atividade 3		■	■							
4. Atividade 4		■	■	■	■	■	■			
5. Atividade 5				■	■	■	■	■	■	
6. Atividade 6								■	■	■
7. Atividade 7										■

INTRODUÇÃO

O objetivo deste projeto consiste em pesquisar uma ferramenta que contribua com o trabalho de empresas que atuam na Área de Engenharia e Computação.

Os modelos tridimensionais das edificações em Realidade Virtual (RV) são trabalhosos na sua produção e apresentam um custo razoável para sua confecção. A ferramenta proposta possibilita aos profissionais de Engenharia e Arquitetura "passearem" pela edificação mesmo antes dela ter sido construída, auxiliando na elaboração do projeto da edificação.

O escopo do trabalho visa investigar a melhor maneira de se desenvolver o aplicativo que irá converter um arquivo DXF do AutoCAD no formato VRML que se baseia nas informações do desenho da edificação. Um ambiente para permitir a visualização e a interatividade do modelo de Realidade Virtual (R.V.) também faz parte das pretensões desse trabalho de pesquisa.

A motivação para o desenvolvimento de uma Ferramenta de Auxílio a Projetos de Engenharia está na agilidade que esta proporcionará aos profissionais de Engenharia e Arquitetura, porque permitirá a interação com a edificação durante seu planejamento. Além disso, o aprendizado da Linguagem de Modelagem de Realidade Virtual e da Linguagem de Programação C++ foi um desafio que estimulou a pesquisa.

O trabalho aborda, inicialmente, a história e a evolução dos sistemas CAD, bem como algumas de suas utilizações. Abrange, também, sua estrutura funcional cujo conteúdo será usado no desenvolvimento do projeto.

Em seguida, aborda-se a Realidade Virtual (RV), relatando sua história, desenvolvimento e aplicações. Baseando-se nestes conceitos é que a Ferramenta de Visualização e Conversão de Arquivos DXF e VRML foi desenvolvida.

O primeiro capítulo descreve a evolução da Engenharia e o CAD, resumindo uma parte da sua historia e seu desenvolvimento.

O segundo capítulo aborda a história da Realidade Virtual (R.V.) e suas áreas de aplicações.

O terceiro capítulo descreve o formato do arquivo DXF e sua ligação com o CAD.

O quarto capítulo aborda as pesquisas realizadas durante o projeto, ressaltando as dificuldades encontradas.

O quinto capítulo descreve a Ferramenta VModel que é utilizada como base desse projeto para visualização e conversão de arquivos DXF e VRML.

CAPÍTULO 1 - PROJETOS – A ENGENHARIA E O CAD

1.1- Evolução

Até pouco tempo atrás, o uso de computadores em sua grande maioria destinava-se a aplicações de caráter administrativo.

Hoje, os métodos produtivos tradicionais estão cedendo caminho para os automatizados, implicando assim em uma diversificação no uso de computadores.

Os primeiros sistemas gráficos surgiram na década de 60, porém, devido ao custo extremamente elevado desses equipamentos, sua aplicação foi reduzida à indústria aeronáutica, onde o custo do produto final justificava o investimento. (AUTOCAD 10.0)

O advento dos mini e microcomputadores a um custo mais acessível, possibilitaram a sua implementação num número muito grande de empresas com resultados positivos.

A computação gráfica desperta, hoje, grande interesse na comunidade brasileira, porém de uma forma obscura e de resultados duvidosos, por ser uma tecnologia recente.

Desta forma, a introdução de conhecimentos adequados é necessária para que os técnicos usem o computador como uma ferramenta cotidiana.

No MIT, Steve Coons começou a desenvolver técnicas de desenho de superfícies baseadas na decomposição em partes que foram aplicados ao desenho de cascos de navios em 1964. (BARN, 1993).

A modelagem de sólidos teve um desenvolvimento mais lento. Os primeiros relatos são os trabalhos desenvolvidos no MIT por Coons de 1960 e 1965, que se concentraram na utilização de métodos numéricos a sólidos criados por varredura (extrusão).

Os primeiros trabalhos relacionados com o modelo de fronteiras aparecem na Universidade de Cambridge (Inglaterra - UK), no fim dos anos 60. (BARN, 2003)

No final dos anos 60 e início de 70, iniciou-se o desenvolvimento dos modeladores de sólidos. Entre eles o EUCLID, de J.M. Brun (França), o PADL-1 da Universidade de Rochester, o SHAPES do MIT e o TIPS-1 desenvolvido por Okino. (BARN, 2003)

"Uma das causas que leva ao fracasso de uma instalação de CAD é a esperança irreal do que ele deve fazer". (AUTOCAD10.0)

Hoje o AutoCad é o software gráfico mais utilizado em todo o mundo e também aqui no Brasil. O número de cópias de AutoCad em todo o mundo é extremamente grande. Possui uma infinidade de publicações, É adotado como padrão mundial de Cad. (AUTOCAD10.0)

1.2 - O Uso de Modelos

O computador tem grande aplicação em processos de simulação consistindo de uma série de cálculos numéricos e tomada de decisões, realizando conjuntos de regras específicas e que utiliza a matemática como uma das suas principais ferramentas.

O engenheiro costuma construir um modelo geométrico que representa a forma e a partir daí, trabalha com esse modelo nas outras etapas de um projeto, nas quais irá definir os materiais, suas dimensões e outras características.

O modelo geométrico criado se materializa por meio de modelos físicos (maquetes), ou por meio de modelos gráficos (desenhos técnicos). (LATERZA, 1994)

1.3- O desenho na era da informática

Apesar de o desenho ser um modelo adequado para a representação dos objetos, não é totalmente adequado para as tarefas de análise do projeto. Geralamente era necessária a

construção de modelos físicos ou matemáticos que representassem a geometria do produto para que as análises de engenharia pudessem ser realizadas.

Com a introdução do computador teve um grande avanço nas metodologias de cálculo e análise numérica, onde foram desenvolvidos métodos importantes, como por exemplo Método dos Elementos Finitos.

Era necessário, uma maneira numérica para representar a forma dos objetos, fazendo com que surgissem as primeiras pesquisas no campo hoje conhecido como Modelamento Geométrico Computacional

Então, notou-se que, se por um lado, a informação numérica é adequada à manipulação pelo computador, por outro, ela é muito difícil de ser manipulada pelo cérebro humano. Assim, tornou-se necessário o desenvolvimento de ferramentas computacionais para o desenvolvimento de dispositivos e programas computacionais que convertessem a informação geométrica do modo gráfico, adequado ao homem, para o modo numérico, adequado ao computador e vice-versa. Estava aberto o caminho para o desenvolvimento da Representação Gráfica Computacional (LATERZA, 1994).

1.4- História da computação gráfica

1.4.1- Origens da Computação Gráfica

A Computação Gráfica está presente em todas as áreas, desde os mais inconseqüentes jogos eletrônicos até o projeto dos mais modernos equipamentos para viagens espaciais, passando também pela publicidade, com as mais incríveis vinhetas eletrônicas e pela medicina onde a criação de imagens de órgãos internos ao corpo humano possibilitando o diagnóstico de males que em outros tempos somente seria possível com intervenções cirúrgicas complicadas e comprometedoras.

Parece existir consenso entre os pesquisadores da história da Computação Gráfica de que o primeiro computador a possuir recursos gráficos de visualização de dados numéricos foi o "*Whirlwind I*" (furacão), desenvolvido pelo MIT. Este equipamento foi desenvolvido, em 1950, com finalidades acadêmicas e também possivelmente militares, pois logo em seguida o comando de defesa aérea dos EUA desenvolveu um sistema de monitoramento e controle de vôos (*SAGE - Semi-Automatic Ground Enviroment*) que convertia as informações capturadas pelo radar em imagem em um tubo de raios catódicos (na época uma invenção recente) no qual o usuário podia apontar com uma caneta ótica. Ocorre que nesta época os computadores eram orientados para fazer cálculos pesados para físicos e projetistas de mísseis não sendo próprios para o desenvolvimento da Computação Gráfica (pucrs_2006).

Em 1962, surgiu uma das mais importantes publicações de Computação *Gráficade* todos os tempos, a tese do Dr. Ivan Sutherland ("*Sketchpad - A Man-Machine Graphical Communication System*"), propunha uma forma de inteção muito semelhante ao que hoje chamados de interfaces *WIMP – Window-Icon-Menu-Pointer*.

Esta publicação chamou a atenção das indústrias automobilísticas e aeroespaciais americanas. Os conceitos de estruturação de dados bem como o núcleo da noção de Computação Gráfica interativa levaram a General Motors a desenvolver o precursor dos primeiros programas de CAD. Logo em seguida diversas outras grandes corporações americanas seguiram este exemplo sendo que no final da década de 60 praticamente toda a indústria automobilística e aeroespacial se utilizava de softwares de CAD (pucrs_2006).

1.5- Histórico do CAD

CAD do inglês *Computer Aided Design*, que quer dizer Projeto Assistido por Computador, consiste basicamente de sistemas capazes de auxiliar um projetista (mecânico, elétrico, civil) a desenvolver suas idéias de forma mais rápida. Os sistemas de CAD são normalmente entendidos como programas capazes de fazer desenhos. De fato, são, em grande parte, isto, pois com um CAD o processo de criação e, principalmente, de alteração de desenhos fica muito facilitado. Porém, CAD não é somente isto, um dos principais avanços que alguns destes sistemas trazem em relação ao processo original de projeto é sua capacidade de fazer simulações. Por exemplo, existem sistemas capazes de determinar o comportamento de uma laje de concreto quando esta for submetida a um certo esforço, outros programas podem mostrar como ficaria a iluminação de uma sala com a colocação de uma janela em uma certa parede (pucrs_2006).

1.6- Editores gráficos (sistemas CAD-2D)

Após o surgimento dos primeiros monitores de vídeo gráficos acoplados aos computadores, que utilizavam a tecnologia dos CRT, tubos de raios catódicos desenvolvidas com outras finalidades (monitores de radar, osciloscópios etc.), surgiram também as *light-pens*, que permitiam ao usuário interagir com os desenhos apresentados no monitor.

Com o surgimento desses e outros periféricos gráficos, foram desenvolvidos os sistemas computacionais que ficaram conhecidos pela sigla CAD que hoje chamamos genericamente de “editores gráficos”. Tais sistemas permitem que as informações geométricas que compõem o desenho sejam criadas, alteradas e apresentadas graficamente no monitor e, ao mesmo tempo, guardadas numericamente na memória do computador. Além disso, permitem que essas informações, se necessário, sejam enviadas às impressoras gráficas (plotters). Tais sistemas trouxeram diversas vantagens, entre as quais podemos citar:

• Facilidade na criação e alteração do desenho;

• Melhoria da qualidade gráfica, pois o traço não depende da habilidade dos desenhistas;

• Maior facilidade no arquivamento, recuperação e transporte dos desenhos;

• Possibilita o reaproveitamento de informações existentes, tanto em projetos realizados como em detalhes nas bibliotecas de padrões;

• Organização das informações em camadas (*layers*), facilitando a criação de novas pranchas, combinando-se as camadas; e

• Maior consistência entre desenhos gerados por vários projetistas, pois passam a trabalhar sobre uma base de dados comum a todos.

Apesar das vantagens na mecanização e racionalização das tarefas de desenho, faltava ainda a integração total das etapas de projeto aos editores gráficos, uma vez que os modelos utilizados para a representação da geometria de um produto eram, na verdade, de natureza bidimensional, ou seja, um conjunto de linhas que representavam a geometria de uma ou mais projeções do objeto num plano, o mesmo que os desenhos convencionais. (LATERZA,1994).

1.6.1-O CAD

Os sistemas CAD/CAM foram criados para melhorar os desenhos de prancheta, mas, hoje, podem ser ameaça para operários, técnicos, engenheiros, copistas, projetistas e desenhistas. Deixando de lados os conceitos, o CAD não é mais somente uma ferramenta para projetos, quando prevalecia a tecnicidade do seu uso. (YOSHIDA, 2002).

Existem modelos de CAD específicos que simulam as condições de fabricação, ou seja, as ferramentas usadas no desenho são as mesmas disponíveis no chão de fábrica (estes são geralmente chamados programas CAM). Também na arquitetura existem CADs

específicos que desenham paredes, telhados e outras construções automaticamente. Os softwares mais avançados de CAD usam o chamado modelagem paramétrica, que permite modificações do desenho pela simples entrada de números indicando dimensões e relações entre as entidades ou objetos desenhados (pucrs_2006).

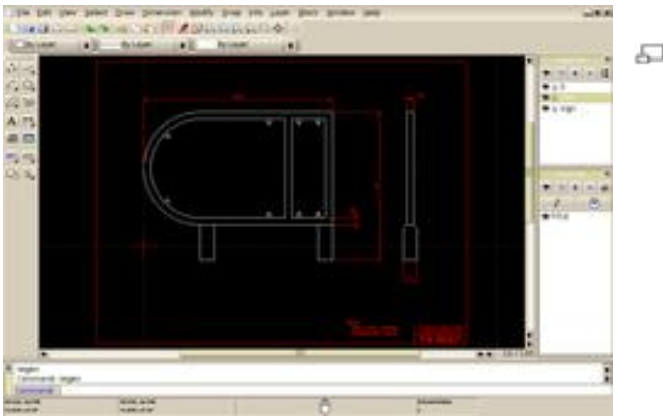

Figura 1.1 -Tela de um software CAD, representação de uma planta baixa (CAD_2006).

Os *softwares* CAD costumam ser utilizados por um nicho pequeno de usuários, devido a sua intensa especialização e seu alto custo. Existem poucas ferramentas livres nessa área, e em muitos aspectos ficam aquém dos *softwares* comerciais. Também costumam demandar *hardware* caro. Embora ao contrariar o paradigma comercial de computação, possamos utilizar versões antigas de CAD cuja necessidade de *hardware* é modesta e o desempenho é satisfatorio, pricipalmente em projetos bidimensionais. (pucrs_2006).

1.7- CAD-3D - Modeladores Geométricos Tridimensionais

Foi no final dos anos 60 e início dos 70, que começaram a surgir os primeiros sistemas que representassem a geometria de um produto em sua natureza tridimensional.

Tais sistemas, hoje chamados de modeladores geométricos tridimensionais, podem ser classificados em três categorias: modeladores de arestas, de superfícies e de sólidos. (LATERZA, 1994).

Qualquer representação gráfica de um objeto apresenta-se com duas dimensões (Altura e largura), mas com o auxílio de óculos especiais que fundem determinados pontos da figura, ou da computação gráfica entre outros recursos, pode-se fazer com que a figura dê a impressão de apresentar, também, profundidade, o que dá maior semelhança com o objeto representado.

A genialidade dos criadores das formas 3D, formularam idéias por todo o mundo, fazendo assim vários tipos de programas e instalações em 3D, se destacando nas formas para computador. A mais atual formulação feita para o 3D fora jogos totalmente criados em formas 3D para computadores, se destacando entre todos, o conhecido *Lineage 2*, um jogo com o peso em disco de 3 Mbs, todo 3D. Através da estereoscopia os olhos humanos conseguem fazer uma imagem 3D dos objetos vistos (3D_2006).

A grande vantagem desses sistemas está no fato de que o projetista deixa de trabalhar somente com as projeções do modelo e passa a trabalhar, de certa forma, diretamente com o próprio modelo em 3D do objeto.(LATERZA, 1994).

Outra vantagem, neste caso, é a possibilidade da visualização do objeto (modelo tridimensional) de diversos pontos de vista e por diferentes sistemas de projeção.

Assim, quando passa-se a trabalhar com modelos tridimensionais, está se saindo do campo do desenho propriamente dito e entrando no campo do modelamento geométrico tridimensional.

Os profissionais da área técnica têm acompanhado nos últimos anos a crescente utilização dos sistemas CAD 3D na indústria, facilitada por PCs cada vez mais poderosos e baratos. Foi derrubado o conceito de que trabalhar com modelos virtuais 3D é difícil e caro,

pois os modernos e poderosos sistemas CAD 3D são mais fáceis de usar e muito mais produtivos do que os sistemas 2D tradicionais. (LATERZA,1994).

Pensando em ter-se desenhos 2D para manufatura, os sistemas CAD 3D facilitam muito a criação, atualização e ampliação desses documentos, pois as vistas e cortes são gerados automaticamente a partir do modelo em 3D. Além disso, muitas coisas são faladas ou escritas sobre a eliminação do papel da indústria (*paperless*), onde os desenhos são acessados digitalmente através de softwares visualizadores, ao invés de serem impressos e distribuídos em papel (MOTTA, 2001).

Além da redução do ciclo de projeto, os sistemas CAD 3D modificaram o seu conceito tradicional. Como o projeto mecânico está relacionado a mudanças, os sistemas paramétricos permitiram, efetivamente, que engenheiros e projetistas possam experimentar mais alternativas de projeto, visualiza-los melhor, verificar interferências de montagem, simular virtualmente movimentos, e, o mais importante, dada as características de associatividade dos modelos 3D com os desenhos 2D, que todas as alterações realizadas no protótipo virtual em 3D fossem, automaticamente, reproduzidas nos desenhos 2D (MOTTA, 2001).

Toda essa integração é fruto de um novo conceito de interface de programação de aplicativos que permite, hoje, que cada empresa monte a solução de CAD adequada às suas necessidades, integrando aplicativos para otimização de suas tarefas específicas (MOTTA, 2001).

1.8- O 3D na engenharia

Desde o surgimento de tecnologias em 3D, tem-se dado consultorias sobre a real necessidade de seu uso na engenharia. Mais ou menos 97% dos produtores definem seus softwares como 3D.

Uma solução de CAD 3D seria essencial para a engenharia. Se o mundo real é em 3D, seu produto não deve ser simplesmente uma chapa plana.

Mesmo clientes que entraram para o “mundo tridimensional” desde o tempo que ainda não podíamos ver ou rotacionar uma peça , puderam obter um projeto completo de uma máquina, além de disponibilizar esses modelos para seus fornecedores. Isso sem ter a necessidade de utilizar recursos só encontrados em softwares caros e de grande porte.

Nos últimos anos, com a popularização dos PC’s e, consequentemente, com os preços de máquinas cada vez mais baixos, o software CAD 3D evoluiu muito.

Quanto à implantação dos sistemas CAD 3D na Engenharia, existem algumas peculiaridades.

Mas as mesmas dificuldades aparecem no trabalho com todo software. Conseguem-se ótimos resultados com engenharias em que os projetistas estão utilizando 3D como ferramenta de projeto. Sabe-se que é quase imensurável sua produtividade e que nunca é menor que o 2D. Diz-se que com o 3D o cliente não vai ganhar somente em velocidade de projeto (STUTZ, 2002).

Pode-se citar ainda que a grande maioria dos usuários que usam os recursos 3D consegue chegar ao produto com erro “zero”, ou seja, problemas como peças que não se encaixam ao serem fabricadas, quantidade de peças a serem produzidas, entre dezenas de outros fatores, simplesmente não fazem mais parte dos problemas da engenharia. O que faz um cliente optar em usar o 3D é aquilo que o faz ter a visão de que a tecnologia é produtiva e lucrativa. Os que não conseguem ter essa visão sempre ficarão na prancheta ou no 2D, com menor lucro (STUTZ, 2002).

1.9- As Aplicações dos sistemas CAD

Existem vários pacotes de programas CAD para diversos tipos de plataformas de computadores (PC, estações de trabalho etc.). Cada pacote tem suas funções e segmentos de mercado bem definidos e, consequentemente, um grupo de usuários específicos. Por exemplo, existem sistemas para as áreas de mecânica, eletricidade, arquitetura, calçados, têxtil etc. (BIB, 2006)

1.10- Projeto auxiliado por computador

Computer Aided Design (CAD), ou desenho auxiliado por computador, é o nome genérico de sistemas computacionais (*software*) utilizados pela engenharia, geologia, arquitetura, e design para facilitar o projeto e desenho técnicos. No caso do design, este pode estar ligado especificamente a todas as suas vertentes (produtos como vestuário, eletroeletrônicos, automobilísticos, etc.), de modo que os jargões de cada especialidade são incorporados na interface de cada programa (CAD_2006).

Consistem estes sistemas numa série de ferramentas para construção de entidades geométricas planas (como linhas, curvas, polígonos) ou mesmo objetos tridimensionais (cubos, esferas, etc.). Também deve haver ferramentas para relacionar essas entidades ou esses objetos, por exemplo: criar um arredondamento (filete) entre duas linhas ou subtrair as formas de dois objetos tridimensionais para obter um terceiro (CAD_2006).

1.10-1. O ambiente de projeto

É importante observar que as técnicas e as ferramentas do Modelamento Geométrico Computacional não trouxeram somente melhorias de produtividade e qualidade para o projeto de engenharia, mas também mudaram radicalmente a maneira pela qual ele é desenvolvido.

No modelamento de sólidos, por exemplo, o conceito de composição das formas a partir de sólidos primitivos e operações lógicas Booleanas (união, diferença e intersecção), oferece um método completamente diferente para projetar objetos, sem paralelo na representação gráfica tradicional. É como se o projetista, atuando como um escultor, modelasse a forma dos objetos, acrescentando ou retirando volumes (LATERZA, 1994).

A interface de alguns dos sistemas de modelamento de sólidos com o usuário, colocaram à disposição do projetista operações análogas às utilizadas nos processos de fabricação, tais como tornear, furar, vazar, extrudar, puncionar e estampar. Tais sistemas de modelamento por features conferiram ao projeto um caráter extremamente lógico e intuitivo.

Enfim, o conceito de modelamento paramétrico, pelo qual a forma dos objetos é definida topológicamente por meio de parâmetros, restrições e relacionamentos entre as entidades geométricas, também tem modificado radicalmente a forma de projetar dos engenheiros. É como se o usuário fosse transmitindo a sua “intenção de projeto” ao sistema, que as traduz em traçados geométricos. As progressivas alterações que o produto vai sofrendo ao longo do processo são facilmente gerenciadas pelo sistema, que se encarrega de compatibilizá-las com as restrições e relacionamentos previamente estipulados.

Assim, apesar do projeto de engenharia continuar baseado na imaginação, no modelamento e na comunicação de idéias, a grande diferença está na forma como o modelamento e a comunicação de idéias são efetuados (LATERZA, 1994).

Num ambiente de modelamento geométrico computacional, os desenhos técnicos de engenharia, entendidos como representações gráficas no papel (vistas ortográficas, cortes

etc.), passam a ser necessários apenas na fase final de documentação do projeto. Em alguns casos, inclusive nessa etapa, a representação gráfica em papel está se tornando desnecessária.

Por conseguinte, neste ambiente de projeto, baseado no modelamento geométrico computacional, caso os desenhos sejam necessários, serão gerados automaticamente a partir de um modelo sólido, em vez de construídos vista por vista, linha por linha, como acontecia anteriormente (LATERZA, 1994).

1.10-2. Processo de projeto

Um projeto segue os seguintes passos (ver Figura 2.1): (MASS, 1987).

•**Definição:** Consiste em especificar as propriedades e qualidades relevantes do sistema a ser projetado.

•**Concepção do modelo:** É o núcleo de um processo de projeto. O engenheiro_ou projetista concebe um modelo de sistema que satisfaça as especificações. O modelo deverá ser documentado.

•**Desenho de detalhe:** A maior parte dos objetos que se fabricam tem algum tipo de representação gráfica natural, que se utiliza como descrição 'formal' do elemento a construir. Por esse motivo, antes de passar ao processo de construção deve-se gerar grande quantidade de 'planos' (ou descrições gráficas em geral). O conjunto de documentos gerados deve ser bastante claro para descrever o modelo e com suficiente detalhamento para permitir a fabricação de protótipos, com o intuido de aprovar o projeto. Este passo pode requerer cerca de 50% do esforço na criação e confecção do projeto.

•**Construção de protótipos:** Para elementos que vão se submeter a um processo de fabricação em série ou em cadeia é normal fabricar previamente protótipos, fora da cadeia de montagem.

Às vezes se utiliza protótipos com elementos que não se fabrica em série, como em engenharia civil ou na arquitetura. Nesta situação cabe utilizar as maquetes para estudos de resistência de materiais, o comportamento aerodinâmico e as maquetes de arquitetura.

- **Realização de ensaios:** Realizando ensaios sobre o protótipo pode-se descobrir deficiências ou falhas no modelo ou na própria definição do sistema, o que obrigará a retomar o processo, revisando o projeto. Deve-se observar que o detalhamento do desenho está, inicialmente, dentro deste ciclo de re-projeto.

- **Documentação:** Uma vez validado o projeto passa-se a documentá-lo. A documentação deve conter informação suficiente para poder atacar (efetivar) a construção/fabricação do objeto ou equipamento do sistema. A documentação pode ser apresentada com a descrição do sistema e de todos os seus componentes, esquemas de montagem, lista de componentes, etc.

O processo de projeto segue um esquema interativo, no qual o projetista trata de encontrar um projeto que satisfaça determinados requisitos como mostra na figura a seguir 1.2, explorando possibilidades, seguindo um ciclo de proposta – valorização (custo x benefício).

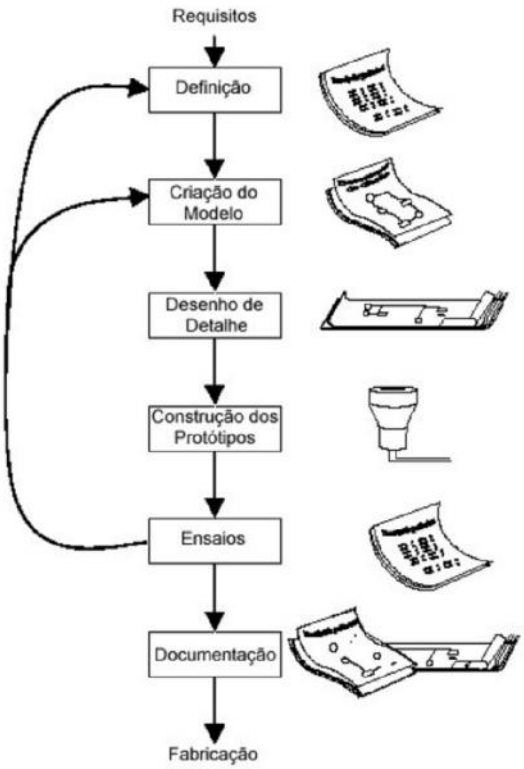

Figura 1.2: Processo clássico de projeto (BARN, 2003)

1.10-3. Conceito de sistema CAD

Pode-se entender o Desenho Assistido por Computador (CAD) como a "aplicação da informática ao processo de projeto". Entende-se por Sistema CAD, um sistema informatizado que automatiza o processo de projeto de um produto (edifício, automóvel, peça, modelo, conjunto, etc.) (SALMON, 87).

Os meios informatizados podem ser utilizados na maior parte das etapas do processo, sendo o desenho em si o que mais tem sido utilizado. Uma Ferramenta CAD é um software que aborda a automatização global do processo de projeto de um determinado tipo de entidade.

O sucesso da utilização dos sistemas CAD resulta na redução de tempo investido nos ciclos de exploração, fundamentalmente pelo uso de sistemas gráficos interativos que permitem realizar as modificações no modelo e observar imediatamente as mudanças refletidas no projeto.

O desenvolvimento de um sistema CAD baseia-se na representação computacional do modelo. Isto permite produzir, elaborar e apresentar automaticamente os detalhes do desenho e a sua documentação, além de possibilitar a utilização de métodos numéricos para realizar simulações sobre o modelo, como uma alternativa à construção de protótipos.

Os ciclos do projeto que utilizam sistemas CAD afetam-se pela inclusão de uma etapa de simulação entre a criação de um modelo e a geração dos seus esboços. Esta simples modificação supõe uma redução importante na duração do processo de desenho. Na Figura 1.3 é mostrado o ciclo de projeto utilizando uma ferramenta CAD. A área sombreada mostra a atuação do sistema CAD (SALMON, 87).

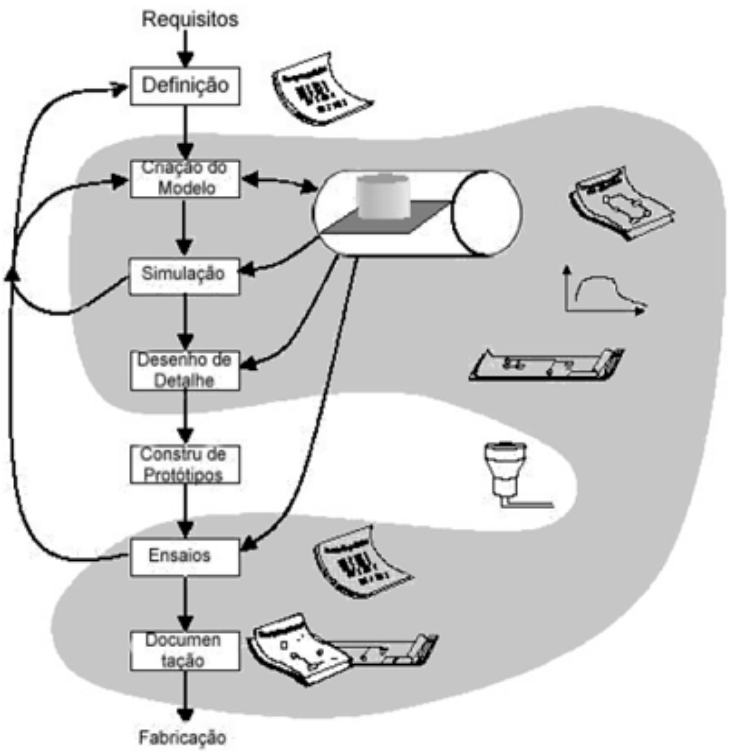

Figura 1.3: Processo de projeto usando uma ferramenta CAD.(BARN, 2003)
A área sombreada mostra a abrangência do sistema CAD

Apenas as etapas de definição e testes de ensaio se utilizam de protótipos que se encontram fora do âmbito de um sistema CAD. As demais tarefas são realizadas utilizando o sistema CAD. A importância da realização de ensaios com protótipos dependerá da natureza do objeto que queremos projetar, e da possibilidade de substituí-los por simulações numéricas. Quando não existe um processo de fabricação em série a construção de protótipos não pode acontecer. Outro aspecto importante da automatização do projeto é a possibilidade de utilizar a informação do modelo como base para um processo de fabricação assistida por computador (SALMON, 87).

1.10-4. Estrutura de um sistema CAD

O projeto é um processo muito iterativo de definição de um objeto, ou seja, o desenvolvimento de um sistema CAD deve basear-se no estabelecimento de um ciclo de edição suportado por técnicas de representação, de edição e de visualização do modelo. Num nível mais concreto, um sistema CAD deve realizar as seguintes funções (BRUN, 86):

Definição interativa do objeto;

- Visualização múltipla;
- Cálculo de propriedades, simulação;
- Modificação do modelo;
- Geração de planos e documentação;
- Conexão com CAM.

É difícil de se estabelecer um modelo universal de sistema de projeto. Não obstante, em nível geral, e com base nas funções a desempenhar, pode-se estabelecer que todos os

sistemas de projeto pedem ao menos os componentes descritos no modelo abaixo. (BRUN, 86).

1.10-4.1. Modelo

É a representação computacional do objeto que está sendo projetado. Deve conter toda a informação necessária para descrevê-lo, tanto em nível geométrico como de características. É o elemento central do sistema, onde vários outros componentes trabalham sobre o mesmo. Portanto, determinará as propriedades e limitações do sistema CAD.

- **Subsistema de edição**. Permite a criação e edição do modelo tanto em nível geométrico quanto especificando propriedades abstratas do sistema. Em qualquer caso, a edição deve ser interativa para facilitar a exploração das diversas possibilidades.

- **Subsistema de visualização**. É encarregado de gerar imagens do modelo. Normalmente interessa poder realizar distintas representações do objeto, bem porque existe mais de um modo de representar graficamente o objeto que está sendo representado, para permitir visualizações rápidas durante a edição, junto com imagens mais elaboradas para avaliar o desenho.

- **Subsistema de cálculo**. Permite o cálculo das propriedades do modelo bem como a realização de simulações.

- **Subsistema de documentação**. Se encarrega da geração da documentação do modelo.

No ciclo de desenvolvimento de projeto com um sistema CAD, pode-se ver uma sucessão de modificações - visualização do modelo. Uma sessão de trabalho com um sistema CAD pode ser vista como a criação de um “programa”, o modelo, que se especifica interativamente com uma seqüência de ordens de edição.

Indubitavelmente, tanto as técnicas de representação e edição do modelo, como a visualização, o cálculo ou a documentação, dependem do tipo de objeto a modelar.

Portanto, não é possível construir sistemas CAD universais. (Ver Figura 1.4).

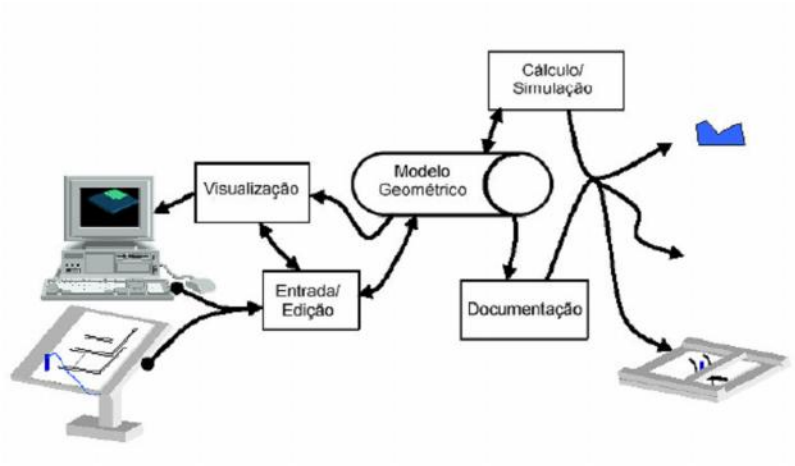

Figura 1.4: Esquema geral de um sistema CAD (BARN, 2003)

1.11- Campos de aplicação

Há um grande número de aplicações que automatizam parte de um processo de projeto. Hoje, para quase todos processos de fabricação e elaboração dispõem-se de ferramentas informatizadas que ajudam este processo. Porém, os três campos clássicos de aplicação são a engenharia civil, o desenho industrial e o projeto (fabricação) de máquinas.

É possível encontrar no mercado aplicações específicas para um determinado campo junto com aplicações de tipo geral, que basicamente são editores de um modelo geométrico, sobre as quais pode-se acoplar módulos de simulação, de cálculos específicos para um certo campo específico. Este último é o caso do AUTOCAD (Autodesk), 3D-Studio (Autodesk) e MICROSTATION (Bentley).

O desenho industrial é o campo típico de aplicação e o que comercializa mais aplicações. Utiliza-se modelos 3D com os quais são realizados cálculos e simulações mecânicas. A natureza das simulações depende do tipo de elemento a projetar. No projeto de veículos é normal simular o comportamento aerodinâmico; no projeto de peças mecânicas

pode-se estudar a flexão, ou a colisão entre duas partes móveis. Entre as aplicações comerciais do tipo geral cabe destacar CATIA (IBM), IDEAS (SDRC) e PRO/ENGINEER (PTC). (Bentley).

No projeto de máquinas e equipamentos pode-se encontrar desde aplicações para o projeto de placas de circuitos impressos e circuitos integrados. Neste último campo é fundamental a realização de simulações do comportamento elétrico do circuito que está se projetando. Muitas destas aplicações são 2D e incluem ou permitem a conexão com um sistema CAM.

Na Engenharia Civil e na Arquitetura pode-se encontrar aplicações 2D, especialmente em arquitetura, e aplicações 3D. As simulações realizadas podem estar relacionadas com o estudo da resistência e a carga do elemento. (Bentley).

1.11-1. As Vantagens

Pode-se destacar vários motivos na utilização de sistemas de auxílio de projeto:

Aumenta a capacidade do projetista/engenheiro: ajuda ao projetista a visualizar o produto e subsistemas;

Melhorias na qualidade do projeto: permite análise de engenharia completa (concepção ao dimensionamento fina) e propicía maiores alternativas para serem verificadas em pouco tempo bem como redução de erros dimensionais de projeto levando a um projeto melhor;

Melhorias na qualidade da comunicação: fornece melhores desenhos, maior padronização nos detalhamentos, melhor documentação para o projeto, menos erros dimensionais e maior clareza de detalhes e legibilidade;

Criação de um banco de dados para manufatura: automaticamente é gerado um banco de dados com as informações geométricas podendo serem enviadas à uma máquina de controle numérico, etc..

Um sistema CAD, se bem implantado, pode aumentar significativamente a produtividade do departamento de projetos, através da implantação de vários tipos de técnicas complementares:

- Personalizando o CAD, utilizando rotinas do dia a dia em formas práticas de utilização;
- CAE, simulações e cálculos feitos a partir do desenho de cada peça;
- CAM, integrando computador e máquina de comando numérico. (McELENY, 2004).

1.11-2. As Desvantagens

Em relação às desvantagens do CAD, estas são poucas, embora consideráveis, destacando-se os custos de aquisição de Hardware e Software.

Quanto ao Custo de aquisição do Hardware, normalmente estão associados a estas aplicações máquinas com características especiais, como por exemplo:

- Velocidade de processamento;
- Placas gráficas com elevada velocidade de processamento;
- Monitor mínimo recomendado de 17” (polegadas).
- Custo associado à formação de utilizadores:
- Apesar de já existirem bastante centros de formação, os preços relativos à formação necessária ainda não são propriamente econômicos. A quantidade/qualidade dos cursos necessários, depende, obviamente, das necessidades específicas do departamento de projeto de cada empresa..

Custo associado à aquisição do Software:

• Existem no mercado diversas soluções, umas econômicas, outras nem tanto. O seu custo vai depender das necessidades específicas de cada Empresa (McELENY, 2004).

1.12- A integração das ferramentas no futuro

As novas maneiras de leitura das informações, verificação de medidas ou tolerâncias, serão vistas via programas auxiliares instalados em microcomputadores, workstations ou notebooks e utilizadas nos canteiros de obra, na engenharia civil, chão de fábrica ou em qualquer lugar distante do local de desenvolvimento de um projeto (BARBERATO, 2001).

Espera-se que em 10 anos, o projeto de produtos continue sendo uma peça importante na economia mundial. Se bem que o número de fornecedores de aplicativos para automação de projetos será bem menor do que hoje. Pode-se até arriscar, no entanto, que nesse período o mundo já terá completado sua transição de 2D para 3D. Futuramente, projetistas e fabricantes compartilharão informações e vão se comunicar em um nível muito mais alto de integração. Os desenhos ainda deverão ser uma forma de troca de dados e informações, mas sim como informações dimensionais e de tolerâncias e serão apresentados não em modo “texto” para em modo “entidade inteligente”.

Espera-se que no futuro, os fabricantes entreguem os melhores produtos ao mercado o mais rápido possível, a um preço mais acessível. (McELENY, 2004).

Mas, para muitos projetistas (àqueles que têm uma visão mais abrangente), bons produtos a preços baixos são uma exigência, pois somente após isso, conseguirão, ao final dos projetos, manter a qualidade exigida pela empresa e pelo mercado, dando sempre um passo a frente na busca constante pela excelência (ANDRADE, 2004).

CAPÍTULO 2 - REALIDADE VIRTUAL

Realidade Virtual (RV) pode ser definida de uma maneira simplificada como sendo a forma mais avançada de interface do usuário de computador até agora disponível. Com aplicação na maioria das áreas do conhecimento, senão em todas, e com grande investimento das indústrias na produção de hardware, software e dispositivos de E/S especiais, a realidade virtual vem experimentando um desenvolvimento acelerado nos últimos anos e indicando perspectivam bastante promissoras para os diversos segmentos vinculados com a área. (ACKERMAN, M. J.).

A Realidade Virtual é uma técnica de interface avançada que permite ao usuário realizar imersão, navegação e interação em um ambiente sintético tridimensional gerado por computador, utilizando canais multi-sensoriais (JACOBSON, 1991), (KRUEGER, 1991) e (BURDEA, 1994).

Na prática, a RV permite que o usuário navegue e observe um mundo tridimensional, em tempo real e com até seis graus de liberdade, exigindo que o software e o hardware reconheçam até seis tipos de movimento: para frente e para trás, acima e abaixo, esquerda e direita, inclinação para cima e para baixo, angulação à esquerda e à direita e rotação à esquerda e à direita. Na essência, a RV é uma simulação da realidade física, na qual um indivíduo que existe em três dimensões tenha a sensação do tempo real e a capacidade de interagir com o mundo ao seu redor. Equipamentos de RV podem simular essas condições, chegando-se ao ponto em que o usuário pode “tocar” os objetos de um mundo virtual além de fazer com que eles respondam, ou mudem, de acordo com suas ações.

A nova interface em RV propõe um controle tridimensional e altamente interativo dos processos computacionais. O usuário entra no espaço virtual das aplicações, visualiza, manipula e explora os dados da aplicação em tempo real, usando para isso os seus sentidos, particularmente os movimentos naturais tridimensionais do corpo. A grande vantagem é que

o conhecimento intuitivo do usuário sobre o mundo físico pode ser transportado ao mundo virtual. Para que se possa suportar esse tipo de interação o usuário utiliza dispositivos não convencionais, como capacetes de visualização e controle (HMD), e luvas de dados (datagloves). O uso desses dispositivos dá ao usuário a sensação de que a aplicação está funcionando no ambiente tridimensional real, permitindo a exploração do ambiente e a manipulação natural dos objetos com o uso das mãos (KIRNER, 1996).

Uma vez que é possível interagir e explorar esse mundo por meio de dispositivos de E/S (entrada e saída), ele se transforma em um ambiente virtual, ou ambiente de Realidade Virtual. A RV é, freqüentemente, confundida com animação, CAD (Computer Aided Design) ou multimídia. Em comparação com essas tecnologias, a RV é (LESTON, 1996):

- Orientada ao usuário, o observador da cena virtual;
- Mais imersiva, por oferecer uma forte sensação de presença dentro do mundo virtual;
- Mais interativa, pois o usuário pode modificar e influenciar o comportamento dos objetos;
- Mais intuitiva, pois existe pouca ou nenhuma dificuldade em manipular as interfaces computacionais entre o usuário e a máquina.

Além disso, RV pressupõe rendering (processo de transformação de modelos em imagens) em tempo real, isto é, imagens são atualizadas assim que a cena sofre qualquer tipo de modificação, incluindo uma descrição funcional dos objetos, estendendo a descrição puramente geométrica e topológica do CAD.

O desenvolvimento de um sistema de RV requer estudos e recursos ligados a percepção sensorial, hardware, software, interfaces de, e com o usuário, fatores humanos e aplicações, sendo necessário, também, algum domínio sobre dispositivos não convencionais de E/S, computadores de alto desempenho, sistemas paralelos e/ou distribuídos, modelagem

geométrica 3D, simulação em tempo real, navegação, detecção de colisão, avaliação, impacto social e projeto de interfaces (KIRNER, 1996).

2.1 - Breve Histórico

A RV começou na indústria de simulação, com os simuladores de vôo que a força aérea do Estados Unidos começou a construir logo após a Segunda Guerra Mundial (JACOBSON, 1994). A indústria de entretenimento também teve um papel importante, ao construir um simulador chamado Sensorama. O Sensorama era uma espécie de cabine que combinava filmes 3D, som estéreo, vibrações mecânicas, aromas, e ar movimentado por ventiladores; tudo para proporcionar ao espectador uma viagem multisensorial (PIMENTEL, 1995.

Os primeiros trabalhos científicos na área surgiram em 1958, quando a Philco desenvolveu um par de câmeras remotas e o protótipo de um capacete com monitores que davam ao usuário um sentimento de presença quando dentro de um ambiente virtual. (COMEAU, 1961). Posteriormente, esse equipamento passou a se chamar head-mounted display, ou simplesmente HMD (ELLIS, 1994).

Alguns anos depois, Ivan Sutherland, conhecido como o precursor da RV, apresentou à comunidade científica a idéia de desenhar objetos diretamente na tela do computador com uma caneta ótica, sendo este o início da Computação Gráfica. Sutherland tornou-se o precursor da atual indústria de CAD e desenvolveu o primeiro vídeo-capacete totalmente funcional para gráficos de computador no projeto "*The Ultimate Display*". Esse vídeo-capacete permitia ao usuário observar, movimentando a cabeça, os diferentes lados de um cubo representado em uma estrutura fio-de-arame flutuando no espaço (FISHER, 1990) (MACHOVER, 1994).

Na mesma época em que Sutherland criava seu vídeo-capacete na Universidade de Utah, Krueger M. experimentava combinar computadores e sistemas de vídeo, criando Realidade Artificial na Universidade de Wisconsin (PIMENTEL, 1995). Em 1975, Krueger criou o VIDEOPLACE, onde uma câmera de vídeo capturava a imagem dos participantes e a projetava em uma grande tela. Os participantes podiam interagir uns com os outros e com objetos projetados nessa tela, tendo seus movimentos capturados e processados. Essa técnica tornou-se conhecida como Realidade Virtual de Projeção (JACOBSON, 1994).

Em 1982, Thomas Furness demonstrava à Força Aérea Americana, o VCASS (*Visually Coupled Airborne Systems Simulator*), conhecido como "*Super Cockpit*". Tratava-se de um simulador que usava computadores e vídeocapacetes interligados para representar a cabine de um avião em 3D. Os vídeo-capacetes integravam as componentes de áudio e vídeo. Sendo assim, os pilotos podiam aprender a voar e lutar em trajetórias com até 6 graus de liberdade sem decolar verdadeiramente. O VCASS possuía alta qualidade de resolução nas imagens além de ser bastante rápido na atualização de imagens complexas. No entanto, o custo representava um problema: milhões de dólares eram necessários apenas para o capacete (PIMENTEL, 1995).

Com a nova tecnologia dos visores de cristal líquido (LCD), McGreevy começou a trabalhar no projeto VIVED (*Virtual Visual Environment Display*) em 1984 na NASA, no qual seriam geradas imagens estereoscópicas. A resolução dessas imagens eram limitadas em comparação com o VCASS, mas o custo era bastante atrativo (RHEINGOLD, 1991). Componentes de áudio e vídeo foram montados sobre uma máscara de mergulho utilizando dois visores LCD com pequenos auto-falantes acoplados. Scott Fisher juntou-se a esse projeto em 1985, com o objetivo de incluir nele luvas de dados, reconhecimento de voz, síntese de som 3D, além de dispositivos de feedback (resposta) tátil.

Figura 2.1: Protótipo do Sensorama.

http://www.sensomatic.com/sensorama/

Figura 2.2: Piloto usando capacete do projeto "Super Cockpit" de Tom Furness

(PIMENTEL, 1995)

Figura 2.3: Super-Cockpit

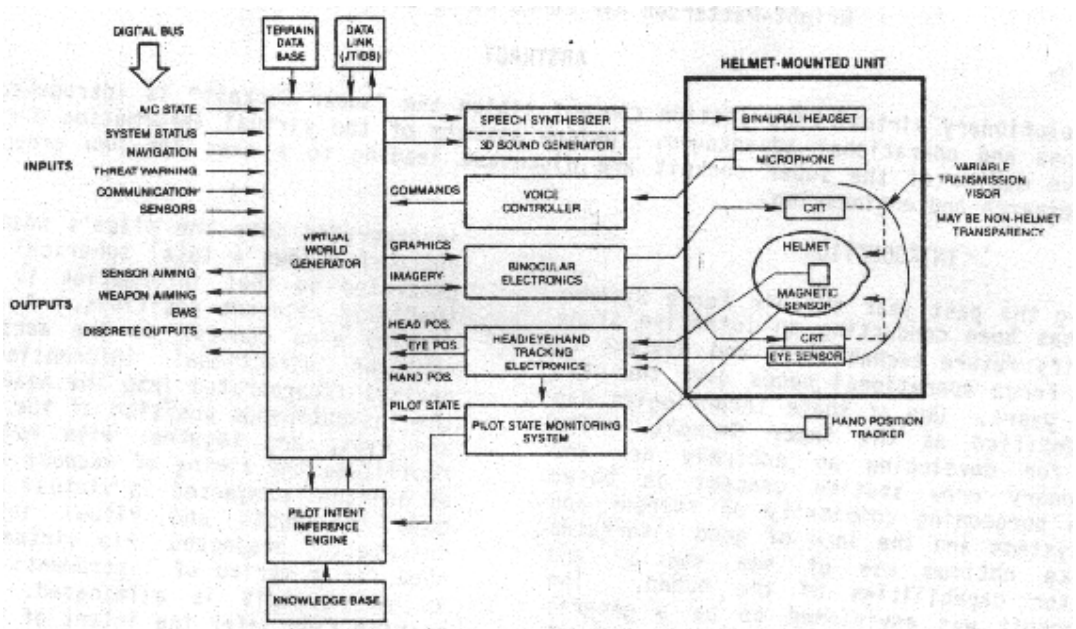

Figura 2.4: Esquema do Super-Cockpit

Fonte: http://www.hitl.washington.edu/publications/m-86-1/m861fig1.gif

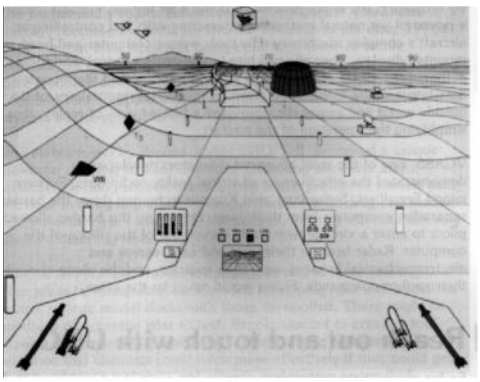

Figura 2.5: Visão do usuário do "Super Cockpit" (PIMENTEL, 1995)

Em 1985, Thomas Zimmerman e Jaron Lanier fundam a VPL Research, tendo como primeiro produto uma luva de dados, chamada DataGlove, desenvolvida por Zimmerman e capaz de captar os movimentos e inclinação dos dedos de uma mão. No mesmo ano uma dessas luvas foi comprada para o projeto VIVED.

No final de 1986 a equipe da NASA já possuía um ambiente virtual que permitia aos usuários ditar comandos por voz, de escutar fala sintetizada, som 3D, e manipular objetos virtuais diretamente por meio do movimento das mãos. (JACOBSON, 1994).

2.2- Tipos de sistemas de RV

Diversos livros e artigos abordam conceitos e definições sobre RV ou Ambiente Virtual (KALAWSKY, 1993); (LATTA, 1994); (EARNSHAW, 1995); (VINCE, 1995); (BRUNETTI et al., 2000); (VALERIO NETTO, 2000). Como dito, há várias definições aceitas devido, em parte, à natureza interdisciplinar da área e à sua evolução, pois, de uma maneira ou de outra, os sistemas de RV acabaram vindo de sistemas de mesa, simuladores e sistemas de tele-operação, entre outros (KIRNER, 1996).

Os sistemas de RV diferem entre si de acordo com os níveis de imersão e de interatividade proporcionado ao usuário. Níveis estes determinados pelos tipos de dispositivos de entrada e saída de dados do sistema, além da velocidade e potência do computador que o hospeda. Ainda não existe um critério claro para classificação dos sistemas de RV. Shepherd em (SHEPHERD, 1993), identifica duas grandes classes: tele-presença, em que um ambiente sintético comum é compartilhado entre várias pessoas como uma extensão do conceito de trabalho cooperativo suportado por computador, e tele-operação, onde robôs agem sobre um elemento, seja ele um corpo humano ou um produto em sua manufatura. Entretanto, esses termos sofreram vários desdobramentos e mesmo inversões.

Aplicações de RV, em geral, são classificadas da seguinte maneira: tele-colaboração, tele-presença, visualização científica, visualização de dados 3D e outros.

Pimentel em (PIMENTEL, 1995) considera que sistemas ou estilos de RV podem ser classificados como sendo RV de Simulação, RV de Projeção, Augmented Reality (Realidade Realçada ou Aumentada), Tele-presença, Visually Coupled Displays (“Displays Visualmente Acoplados”) e RV de Mesa.

A RV de Simulação corresponde ao tipo mais antigo, originado com os simuladores de vôo desenvolvidos pelos militares americanos após a Segunda Guerra Mundial (JACOBSON, 1994). Um sistema desse tipo imita o interior de um avião, jato, carro, colocando o usuário dentro de uma cabine com controles. Nesta, telas de vídeo e monitores apresentam um mundo virtual que reage à comandos do usuário. Como um sistema de RV de Simulação não processa imagens em estéreo, as imagens são geradas de forma bem acelerada. Em alguns sistemas as cabines são montadas sobre plataformas móveis, e os controles oferecem feedback tátil e auditivo.

A RV de Projeção também é conhecida como Realidade Artificial, foi criada na década de 70 por Myron Krueger. Na RV de Projeção o usuário está fora do mundo virtual,

entretanto, pode se comunicar com personagens ou objetos nele contidos. (JACOBSON, 1994).

A Realidade Realçada ou Aumentada (*Augmented Reality*) utiliza dispositivos visuais transparentes presos na cabeça do usuário. Como esses displays são transparentes, o usuário pode ver dados, diagramas, animações e gráficos 3D sem deixar de enxergar o mundo real, tendo informações geradas pelo computador sobrepostas ao mundo real. Tais *displays* transparentes são chamados *heads-up-displays* (HUDs). O usuário pode, por exemplo, estar consertando algo e visualizando nos óculos os dados necessários a esta operação.

A Telepresença, como já mencionado, utiliza câmeras de vídeo e microfones remotos para envolver e projetar o usuário profundamente no mundo virtual. Controle de robôs e exploração planetária são alguns exemplos de pesquisas em desenvolvimento. No entanto, existe um grande campo de pesquisa no uso de tele-presença em aplicações médicas.

Médicos já utilizam câmeras de vídeo e cabos de fibra óptica em cirurgias para visualizar os corpos de seus pacientes. Através da RV eles podem, literalmente, “entrar” no paciente, indo direto ao ponto de interesse e/ou vistoriar a operação feita por outros.

Os *Displays* Visualmente Acoplados (*Visually Coupled Displays*) correspondem a classes de sistemas nas quais as imagens são exibidas diretamente ao usuário, que olha em um dispositivo que deve acompanhar os movimentos de sua cabeça. Esse dispositivo geralmente permite imagens e sons estéreo, além de conter sensores especiais que detectam a movimentação da cabeça do usuário e usam essa informações para alterar as imagens exibidas.

A RV de Mesa (*Desktop VR*) é um subconjunto dos sistemas tradicionais de RV em que, ao invés de headmounted displays (HMD), utiliza-se grandes monitores ou algum sistema de projeção para apresentação do mundo virtual. Alguns sistemas permitem ao

usuário ver imagens 3D no monitor com óculos obturadores, polarizadores ou filtros coloridos.

O conceito de CAVE (*Automatic Virtual Environment*) surgiu como uma nova proposta para interfaces de sistemas de RV (CRUZ-NEIRA, 1992). Um CAVE (ou uma Caverna, em português) consiste de uma sala nas quais paredes, teto e chão são telas semi-transparentes aonde as imagens são projetadas, permitindo que uma ou mais pessoas fiquem imersas no ambiente virtual. A projeção das imagens é feita por projetores que ficam posicionados atrás das telas e pode ser estereoscópica, exigindo dos usuários o uso de óculos obturadores. A grande vantagem de sistemas desse tipo é a total imersão do usuário no ambiente virtual.

Sistemas do tipo CAVE também incorporam projeção acústica tridimensional, dispositivos de rastreamento de posição e de interação. A estrutura computacional envolvida no acionamento e funcionamento de um CAVE é bastante avançada e deve processar os pares estereoscópicos das imagens (um total de 12 imagens, imaginando-se uma CAVE de 6 lados) além de gerenciar os dispositivos de interação, auditivos e projetores. A Figura 2.6 mostra o esquema de um CAVE com o posicionamento dos projetores.

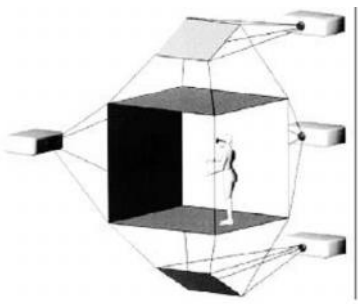

Figura 2.6: Esquema de CAVE com o posicionamento de projetores atrás das telas.

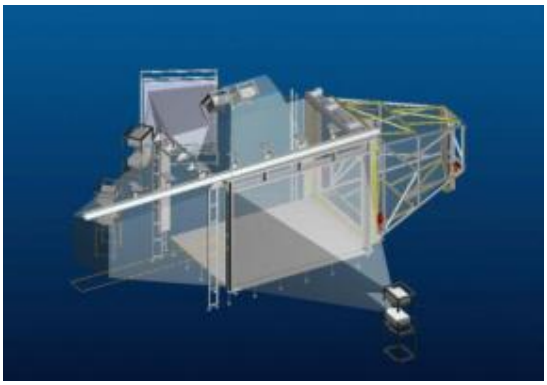

Figura 2.7: Esquematização de uma CAVE

Figura 2.8: Cave em funcionamento

2.3- Imersão, Interação e Envolvimento

A RV pode ser caracterizada pela coexistência de três idéias básicas: imersão, interação e envolvimento (MORIE, 1994) (KIRNER, 1996). A idéia de imersão está ligada ao sentimento de fazer parte do ambiente. Basicamente, um sistema imersivo é obtido com o uso de capacete de visualização, ou CAVEs (CRUZ-NEIRA, 1992). Além dos fatores visuais, alguns dispositivos ligados aos demais sentidos também são importantes para o sentimento de imersão, como o som, o posicionamento automático da pessoa e também dos movimentos da cabeça, controles de força e reativos, etc. (BEGAULT, 1994) (GRADECKI, 1994). A visualização de uma cena 3D em um monitor é considerada não imersiva. Assim, tem-se também a conceituação de RV imersiva e não imersiva (LESTON, 1996).

De modo geral, do ponto de vista da visualização a RV imersiva utiliza capacete ou CAVEs, enquanto a não imersiva usa monitores. Porém, dispositivos baseados nos demais sentidos podem introduzir algum grau de imersão à RV que usa monitores (ROBERTSON, 1993). Os monitores ainda apresentam alguns pontos positivos, como a facilidade de uso e o baixo custo, evitando as limitações técnicas e problemas decorrentes do uso do capacete. Porém, a tendência deve ser a utilização da RV imersiva na maioria das aplicações futuras.

A interação está ligada à capacidade do computador detectar as entradas do usuário modificando instantaneamente o mundo virtual em função das ações efetuadas sobre ele (capacidade reativa). Pessoas devem ser e são cativadas por uma boa simulação em que as cenas mudam em resposta aos seus comandos, que é característica mais marcante da nova geração de jogos e vídeo games. Para que um sistema de RV seja mais realista, o ambiente virtual inclui objetos simulados. Outros artifícios para aumentar o realismo são empregados como , por exemplo, a texturização dos objetos do ambiente e a inserção de sons tanto ambientais quanto sons associados a objetos específicos (ARAÚJO, 1996).

O envolvimento, porém, está ligado ao grau de motivação para o engajamento de uma pessoa em determinada atividade. O envolvimento pode ser passivo (ler um livro ou assistir televisão), ou ativo (participar de um jogo com parceiros). A RV tem potencial para os dois tipos de envolvimento, por permitir a exploração do ambiente virtual e propiciar a interação do usuário com o mundo virtual dinâmico.

Embora a percepção visual seja nosso sentido primário, outros sentidos também devem ser estimulados visando proporcionar uma completa imersão; entre os quais o retorno auditivo, o tato e a força de reação.

2.4- RV passiva, exploratória ou interativa

Um aplicativo de RV pode proporcionar sessões sob três formas diferentes: Passiva, Exploratória ou Interativa. Uma sessão passiva proporciona ao usuário uma exploração do ambiente automática e sem sua interferência. As dimensões, velocidades, rotas e os pontos de observação são pré-determinados e, controlados exclusivamente pelo software. O usuário não tem controle algum, exceto talvez, para sair da sessão.

Uma sessão de RV exploratória proporciona uma exploração do ambiente controlada pelo usuário. Este pode escolher a rota e os pontos de observação, mas não pode interagir de outra forma com entidades contidas na cena.

Já uma sessão interativa proporciona uma exploração do ambiente totalmente dirigida pelo usuário e, além disso, as entidades virtuais do ambiente respondem e reagem às suas ações. Por exemplo, se o usuário move o ponto de observação em direção à porta, esta pode abrir-se, permitindo ao usuário passar por ela. (ADAMS, 1994).

2.5- Aplicações de RV

A Realidade Virtual ainda se encontra em um estágio inicial de desenvolvimento, porém seu uso tende a ser bem abrangente. Como ferramenta de manufatura (CAE/CAD/CAM), possibilita, entre outras aplicações simular a fabricação de uma peça mecânica em 3D, visualização de Prédios e detecção de problemas estruturais, etc.

A Realidade Virtual traz aos usuários os seguintes benefícios:

- Identificação rápida e fácil de possíveis falhas num projeto;
- Correção imediata com baixo custo;
- Facilidade na apresentação do projeto a outros grupos de especialistas externos e internos;

- Armazenamento de informações; e
- Facilidade de manutenção das partes que compõem produtos mais complexos.

Os testes feitos em ambientes virtuais são muito mais baratos, não colocando em risco a vida dos usuários. Quanto mais rápidas e precisas forem a manutenção e a reposição de peças danificadas de um produto, menor será também o custo do trabalho na área de Engenharia de Automação. Periféricos de Realidade Virtual possibilitam operar com máquinas prejudiciais à saúde humana.

Na medicina esses periféricos também podem permitir a execução de cirurgias complexas por médicos em locais remotos, de difícil acesso e em casos em que o paciente não pode ser removido.

Na indústria aeroespacial, a Realidade Virtual permite simulações de pilotagens.

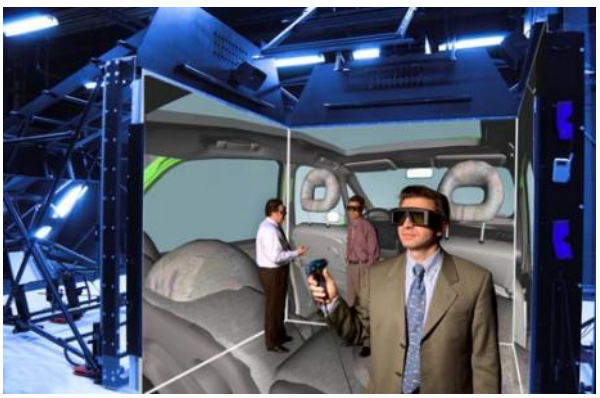

Figura 2.9: Exemplo de Aplicação de RV

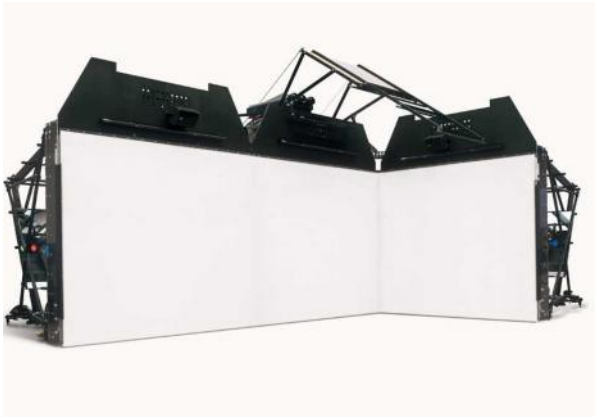

Figura 2.10: Exemplo de Cave para Aplicação de RV

2.6- Visualização em Realidade Virtual

A ampla compreensão da natureza qualitativa de fenômenos complexos é muitas vezes comprometida devido à enorme quantidade de dados por eles produzidos. Técnicas tradicionais têm seu foco voltado à exatidão da informação, fornecendo subsídios importantes para a percepção quantitativa. O desenvolvimento de técnicas avançadas de visualização e realidade virtual, por sua vez, tem proporcionado um mecanismo eficiente e intuitivo na compreensão da natureza qualitativa de dados complexos, agilizando sua interpretação e fornecendo real apoio no domínio da decisão em aplicações de diversas áreas, incluindo a engenharia. (Ande93).

A interface com realidade virtual envolve um controle tridimensional altamente interativo de processos computacionais. O usuário entra no espaço virtual das aplicações e visualiza, manipula e explora os dados da aplicação em tempo real, usando seus sentidos, particularmente os movimentos naturais tridimensionais do corpo. A grande vantagem desse tipo de interface é que o conhecimento intuitivo do usuário a respeito do mundo físico pode ser transferido para manipular o mundo virtual. Para suportar esse tipo de interação, o usuário utiliza dispositivos não convencionais como capacete de visualização e controle, luva, e

outros. Estes dispositivos dão ao usuário a impressão de que a aplicação está funcionando no ambiente tridimensional real, permitindo a exploração do ambiente e a manipulação natural dos objetos com o uso das mãos, por exemplo, para apontar, pegar, e realizar outras ações (ufscar_2006).

2.7- Visualização em Engenharia e RV

A modelagem tridimensional ajuda a visualização e interpretação do objeto projetado. No processo de modelagem tridimensional o usuário constrói um modelo digital do objeto (maquete eletrônica) ao invés de desenhar vistas isoladas deste objeto, como: vista superior, vista frontal e vistas laterais. Uma vez construído o objeto, o usuário poderá posicionar-se adequadamente em relação ao modelo e obter a representação desejada. Sejam vistas ortográficas, projeções axonométricas ou mesmo perspectivas, cortes e seções. A Figura 2.11 mostra a diferença entre a visualização 2D e 3D.

Fonte: http://www.hitl.washington.edu/publications/m-86-1/m861fig1.gif

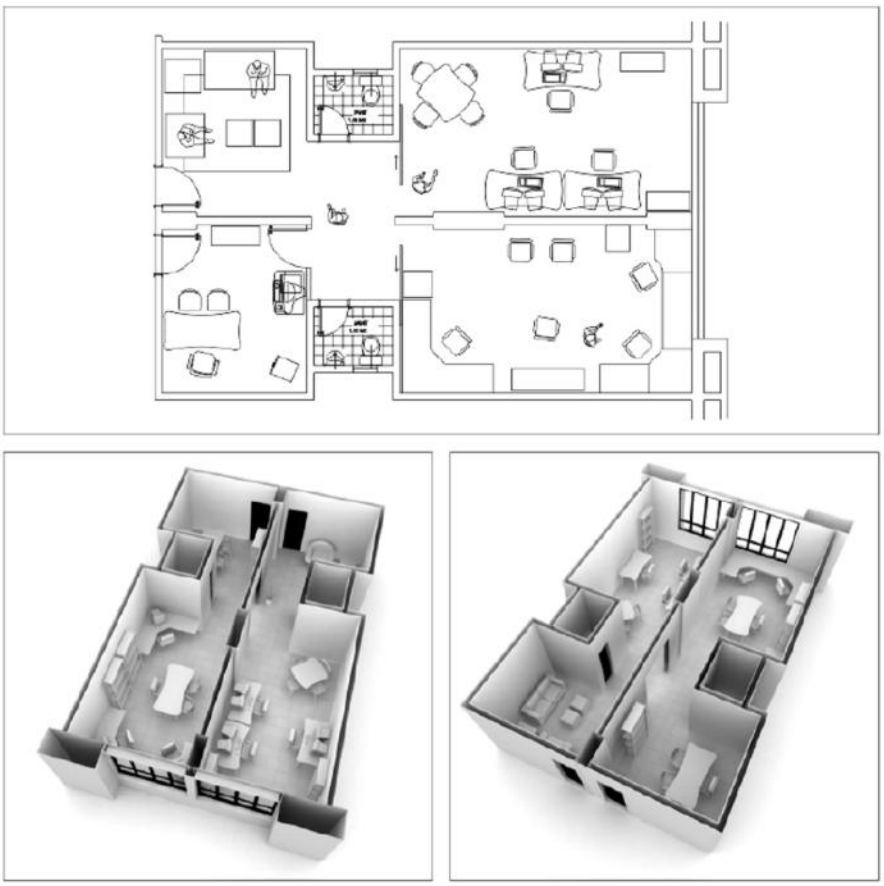

Figura 2.11: Diferenças na visualização de um desenho gráfico em 2D e 3D

Esta técnica de representação de objetos permite a geração automática de perspectivas a partir do posicionamento do observador em qualquer posição do espaço em torno do objeto, e até mesmo posicionar o observador dentro do modelo olhando para fora, particularmente útil em trabalhos de arquitetura.

Os modelos tradicionais podem formar a base para a construção de desenhos 2D. Planos, seções e elevações podem ser extraídas de um modelo tridimensional, e utilizados como uma base sobre a qual é possível adicionar dimensões, notas e símbolos.

Com um modelo digital tridimensional, é possível discutir não só as questões estéticas da edificação, mas também, todo o processo construtivo. Isto se deve a possibilidade de trabalhar com os eixos (X,Y e Z) e a capacidade de simular diferentes pontos de observação, tanto internos quanto externos. O realismo constitui um dos pontos-chave dos modelos gerados por computador, devido a tridimensionalidade, a representação dos elementos arquitetônicos em escala e a simulação de texturas e efeitos luminosos nas superfícies e volumes. A Figura 2.12 mostra posicionamento das câmeras e seus resultados obtidos através do 3D.

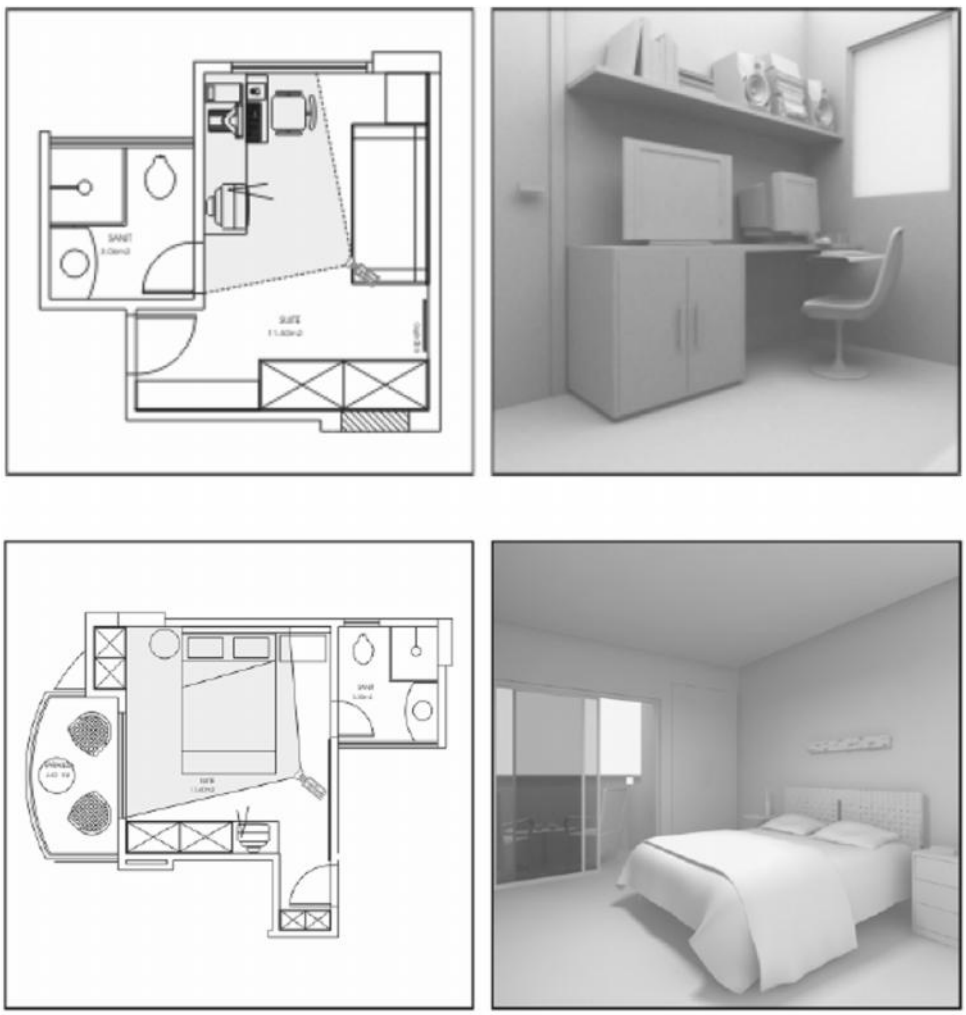

Figura 2.12: Planta baixa mostrando área de atuação da câmera e sua vista em 3D

Na construção civil, pode-se tirar bastante proveito da Realidade Virtual. O leque de possibilidades é bastante amplo, especialmente na área de ensino de Engenharia e Arquitetura, análise de projetos, projetos colaborativos, análise estrutural e desenho urbano. A Arquitetura pode ser potencialmente reformulada pela Realidade Virtual e contribuir para alterar as tecnologias disponíveis. (MACLEOD, 1992).

Os modelos desenvolvidos em sistemas CAD podem ser convertidos através de softwares específicos, para os formatos VRML ou X3D e visualizado com o COSMO Player ou CORTONA, entre outros. (MACLEOD, 1992).

A maioria dos programas destinados à elaboração de maquetes eletrônicas, como o AutoCAD, o Microstation, o 3D Studio Max, entre outros, já disponibilizam filtros para exportar a malha 3D para VRML. Desta forma, qualquer usuário CAD pode, sem maiores dificuldades, exportar desenhos 3D para visualização e interação em ambiente virtual no interior de um browser.

Mas, também tem suas limitações: Dependendo do arquivo gerado, e no caso de uma conversão direta, terá um “custo computacional” significativo, exigindo equipamentos High-end para rodar o modelo. Não basta exportar o modelo gerado, este deve ser criado para o VRML, observando-se suas características: malha, iluminação, propriedades dos materiais e o tamanho do arquivo. O modelo deve ser otimizado para o uso em RV.

Tratando-se do futuro da realidade virtual, é preciso olhar com cuidado para o X3D. Padrão aberto para distribuir conteúdo 3D, combinando geometria e descrições de comportamentos instantâneos em um simples arquivo que tem inúmeros formatos de arquivos disponíveis para isso, incluindo o Extensible Markup Language (XML). É a próxima revisão da especificação ISO VRML97, incorporando os avanços dos recursos disponíveis nos últimos dispositivos gráficos comerciais tanto quanto melhorias na sua arquitetura. (MACLEOD, 1992)

CAPÍTULO 3- DXF (DATA INTERCHANGE FORMAT)

DXF é o formato de dados disponibilizado pela *Autodesk,* empresa que desenvolve o programa de CAD chamado de AutoCAD. O padrão DXF é considerado um padrão de fato.

Devido a grande utilização do sistema AutoCAD pela comunidade industrial e científica. O DXF é muito mais que uma linguagem ou um meta arquivo gráfico (exemplos de *metafile graphics1: CGM . Computer Graphics Metafile, CGI . Computer Graphics* Interface) do que um simples formato de imagem (exemplos de bitmap: *GIF . Graphics Interchange Format, TIFF . Tagged Image File Format*). Assim como num formato de imagem, a exata localização ou a ordem em que os dados gráficos aparecem no arquivo não é importante. Entretanto, como linguagem, o contexto no qual os comandos aparecem pode ter significado diferenciado, dependendo de qual tipo de informação é repassada (DXF_2006).

Inicialmente a edificação em projeto é desenhada utilizando o AutoCAD, neste, as paredes, portas, janelas, telhados, calçamentos são representados num espaço de coordenadas 3D. O AutoCAD além de poder salvar os desenhos em um arquivo padrão (DWG), pode também gerar um arquivo vetorial de intercâmbio, o DXF.

O formato DXF foi criado para ser compatível com a maioria dos CADs existentes. O formato DXF é padrão ASCII, ou seja, é interpretado pelo programa (no caso, o CAD) através da leitura de descrições textuais do conteúdo nele contido.

A estrutura geral do DXF é composta de uma linha de comando que é precedida de um número inteiro alinhado à esquerda chamado Código de Grupo. Ele especifica o que vem a ser o código da próxima linha: se é uma variável ou um nome que indica o início ou o fim das seções do arquivo ou mesmo início e fim do próprio arquivo. A estrutura geral de um arquivo DXF é como segue:

- HEADER: É a seção Cabeçalho. Contém as variáveis de configuração do CAD e seus valores de inicialização. Útil apenas quando da criação de um ambiente de trabalho em outro arquivo.
- TABLES: É a seção Tabelas. Contém a definição dos tipos de linha (*Linetype*), estilos de texto (*TextStyle*) e níveis de informação referenciados nas seções *Blocks* e *Entities*.
- BLOCKS: Seção Blocos. Contém todos os blocos utilizados no desenho. Os blocos podem ser conjuntos de linhas, de pontos, de "primitivas" do DXF.
- ENTITIES: É a seção de Entidades. Nessa seção estão incluídas todas as entidades do desenho, ou seja, pontos, retas, círculos, as "primitivas" do DXF. Cada entidade pode ter até 5 atributos: *Layer, Linetye, Color, Elevation, Tickess.*

• EOF: Seção que indica o fim do arquivo ("*End Of File*") (DXF_2006).

3.1 - FORMATO DO ARQUIVO DXF

Tudo num arquivo DXF é composto de pares chamados de grupos. Cada grupo tem um código, seguido por um número ou string de caracteres, chamado de valor do grupo. O código do grupo é um inteiro que indica o tipo de valor que o segue. Alguns intervalos de códigos são reservados a certos tipos de dados.

Um exemplo da estrutura de seções de um arquivo DXF é observado na Figura 3.1.

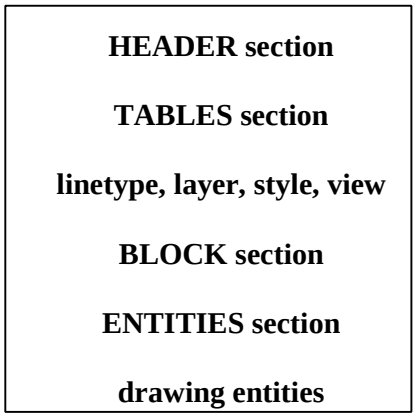

Figura 3.1: Estrutura de seções de um arquivo DXF.

Uma característica dos arquivos DXF é que eles são muito longos. Isso se deve ao fato de que para cada dado, são escritas no mínimo duas linhas no arquivo: uma linha diz o que é o dado, e a outra traz o valor do dado.

Com os avanços da tecnologia de CAD e sua ampla aplicação no meio industrial, alguns programas comerciais de CAD se tornaram bastante utilizados. A grande utilização de arquivos oriundos de tais programas fez com que surgissem os chamados padrões de fato.

Um padrão de fato é um formato de arquivo de dados geométricos que se popularizou, e atingiu um número tão considerável de usuários, que o fez tornar-se um padrão não oficial, mas de fato. O exemplo mais clássico disso é o formato de arquivo DXF *(Drawing Interchange File)*. Diversos desenvolvedores de programas CAD/CAM/CAE *(Computer-Aided Engineering)*, além de gerarem arquivos num formato próprio, adicionaram também saídas em formato DXF, como forma até de aceitação dos seus produtos pelo mercado.

Outra solução, que teve origens no meio comercial e acabou sendo utilizada como forma de trocar dados geométricos entre sistemas CAD foi os chamados núcleos *(kernels)* de modelagem geométrica de sólidos. Diversos programas de CAD adotam e embutem essas bibliotecas de funções de modelagem sólida na sua arquitetura computacional, para geração de sólidos a partir de operações típicas de modelagem sólida. A grande adoção de certas

bibliotecas, por parte dos desenvolvedores de programas CAD, fez com que elas tornassem um meio de trocar dados entre diferentes sistemas CAD. As soluções comerciais, embora adotadas pelo mercado, não podem ser consideradas como oficiais e aceitas de consenso pela comunidade mundial.

Essa comunidade é formada por centros de pesquisa, escritórios de normalização e padronização nacionais e internacionais, indústrias de tecnologia (aeronáutica, automobilística), grandes desenvolvedores de programas, órgãos militares, e outros que influenciam. Os interesses desses grupos são diversos e a luta de poder também faz parte das decisões. O passo seguinte então, foi empenhar-se em desenvolver padrões de formatos de arquivos para troca de dados geométricos e não geométricos entre sistemas de CAD (DXF_2006).

3.2 - O CAD E O ARQUIVO DXF

O CAD pode ser usado por si só como um editor de desenho completo. Em algumas aplicações outros programas têm que examinar os desenhos criados pelo AutoCAD ou gerar desenhos vistos e modificados (3D/DXF_2006).

Por exemplo, se você fez um desenho arquitetônico com AutoCAD,partes inseridas usando para representar janelas, portas, e assim por diante, você pode processar o arquivo de desenho e produzir uma conta de materiais de todos artigos usados no desenho, ou até mesmo fazer cálculos baseados na área e no número e no tipo de janelas usados. Outra possível aplicação é usar AutoCAD para descrever estruturas e então chamar as descrições a um computador mais poderoso finito. Pode-se computar tensões e deslocamentos e mandar de volta para informação exibir o formato de estrutura como um desenho de AutoCAD.

Um formato de arquivo de intercâmbio é o DXF. Toda implementação de AutoCAD aceita este formato e pode converter isto para a representação de arquivo de desenho interno .

AutoCAD também apóia a Especificação de Troca de Gráficos Inicial (IGES) formato de arquivo. A informação que inclui um desenho de AutoCAD pode ser escrita fora em IGES formate, e podem ser lidos arquivos de IGES e convertido ao AutoCAD formato interno (3D/DXF_2006).

O formato DXF (*Drawing Interchange file*) foi desenvolvido para auxiliar a comunicação entre o AUTOCAD e outros programas. Estes arquivos podem ser transformados para outros sistemas CAD ou outros programas.

Os arquivos DXF são arquivos de texto no formato ASCII.

O SPRING utiliza as seções de entidades e tabelas no momento da importação do formato DXF para o SPRING.

O SPRING importa dados DXF dos modelos temáticos e numéricos e para tal, é necessário que os dados tenham sido digitalizados segundo alguns critérios. Para dados temáticos o SPRING importa somente linhas e pontos. As linhas devem ter sido digitalizados no AUTOCAD no formato "*line*" ou "*polyline*", e todo cuidado deve ser adotado no momento de correção das linhas (DXF_2006).

O identificador que representa a característica de cor da linha deve ser do tipo "*Point*" com uma cor associada e deve ser colocado em qualquer lugar do polígono. No momento da digitalização o usuário deve ter associado uma cor ao objeto digitalizado ou a todo plano de informação. Deve-se ter como produto para importação, polígonos fechados com cor específica pelo identificador, e linhas poligonais com a cor da linha ou do plano.

No momento da importação das linhas do SPRING faz automaticamente um ajuste de erros menores que 0.5 mm. Polígonos com erro maior que 0.5 mm é importado como linha poligonal. Após a importação de Linhas e Identificadores o sistema executa a poligonalização.

Linhas que não foram ajustados ou não constituem polígonos são transformados em linhas poligonais, e podem ser devidamente corrigidos e identificados dentro do ambiente SPRING.

Para dados numéricos no formato DXF, o SPRING importa linhas cotadas e pontos cotados, isto é, dados no formato (X, Y, Z), em que X e Y são as coordenadas planas e Z o valor da cota. As linhas cotadas devem ser digitalizadas no AUTOCAD como "line" ou "Polyline", e os pontos cotados como "Point".

Um arquivo DXF pode ser importado em coordenadas planas (metros, quilômetros, milímetros, pés ou polegadas), ou coordenadas geográficas (grau decimal ou segundo decimal), mas somente exportado em coordenadas planas (metros) (DXF_2006).

CAPÍTULO 4 - FORMAS DE CONVERSÃO DXF PARA VRML

A proposta principal do projeto foi a busca de uma forma para converter um arquivo DXF para VRML.

No inicio foi muito difícil encontrar uma ferramenta que pudesse auxiliar, porém depois de muitas pesquisas conseguiu-se acreditar que o **Vmodel** atenderia a proposta.

O Vmodel foi o programa escolhido que obteve sucesso depois de alguns ajustes. Porém antes de informar que ele seria a melhor opção procurou-se outras alternativas citadas a seguir:

SE Drawing Extractor é um programa que extrai facilmente desenhos 2D e 3D para o formato DXF e DWG (AutoCAD) para fácil integração com programas de terceiros, como sistemas CAM e software CNC. Este programa parecia a principio ser útil porém, era pago, sua funcionalidade estava de acordo com a proposta buscada, mas, sua desvantagem era o preço da licença era de custo elevado.

Deep Exploration é um visualizador de gráficos 3D como os do 3D Studio, *LightWave, Caligari, Direct X, AutoCAD,* etc. Com suporte a OpenGL, o programa permite rápidas operações na visualização como zoom instantâneo, e etc. Este sim parecia ser o programa perfeito, funcionava corretamente, fazia todas as conversões necessárias, mas infelizmente não fornecia o código fonte então foi descartado.

DWG DXF Convert 1.0 é um conversor bidirecional de arquivos DWG e DXF. Ele converte arquivos DWG do AutoCAD para arquivos DXF e vice-versa sem necessidade de ter o AutoCAD instalado. Também suporta conversão de múltiplos arquivos de uma só vez.

Pode ser executado em Windows 95, Windows 98, Windows NT, Windows 2000, Windows Millenium, Windows XP. Parecia excelente, mas, não faz a conversão de DXF para VRML, ou seja, também foi eliminado.

SurfX3D As superfícies matemáticas dos lotes de superfície do explorador 3D nas equações 3D de superfície definidas usuário podem ser explícitas, implícitas, ou paramétricas. As equações da superfície e da cor são especificadas pelo usuário. Pode exportar-se para o papel, POV, DXF, impressora. O código de fonte do MFC C++ incluiu. Freeware. A principio parecia muito interessante, mas depois da instalação, não satisfez a conversão necessária para VRML.

DXF to VRML97 é um programa que faz a conversão de DXF para arquivos VRML97, porém continha muitos erros e não era possível identificar todos e não era possível rodá-lo, e tinha-se pressa para encontrar brevemente um programa que fosse útil.

Além destes foram encontradas outras possibilidades, os seus endereços eletrônicos estão citados logo a seguir:

- http://www.filehungry.com/portuguese/product/windows_software/multimedia_&_graphics/3d_graphic_tools/surfx3d
- http://superdownloads.uol.com.br/download/64/d
- http://www.soliddwg.com/?gclid=CJrniZC5yogCFQSSHgodrChTgA
- http://www.allworldsoft.com/folders/alpha/graphic-apps/cad/
- http://fraktali.849pm.com/graphcon.html
- http://www.dc.ufscar.br/~grv/vrml/tutoriais/vrml10/
- http://www.ia.hiof.no/~michaell/home/vr/mac/dxf2vrml/

CAPÍTULO 5 - VMODEL

O Vmodel foi criado com uma utilidade simples e muito interessante para conversão e visualização 3D de arquivos DXF para VRML, (é importante ressaltar que o programa VModel utiliza o VRML 1.0) . É possível até o contrário com uma representação do wireframe. A sua tela de apresentação é mostrada abaixo na figura 5.3.

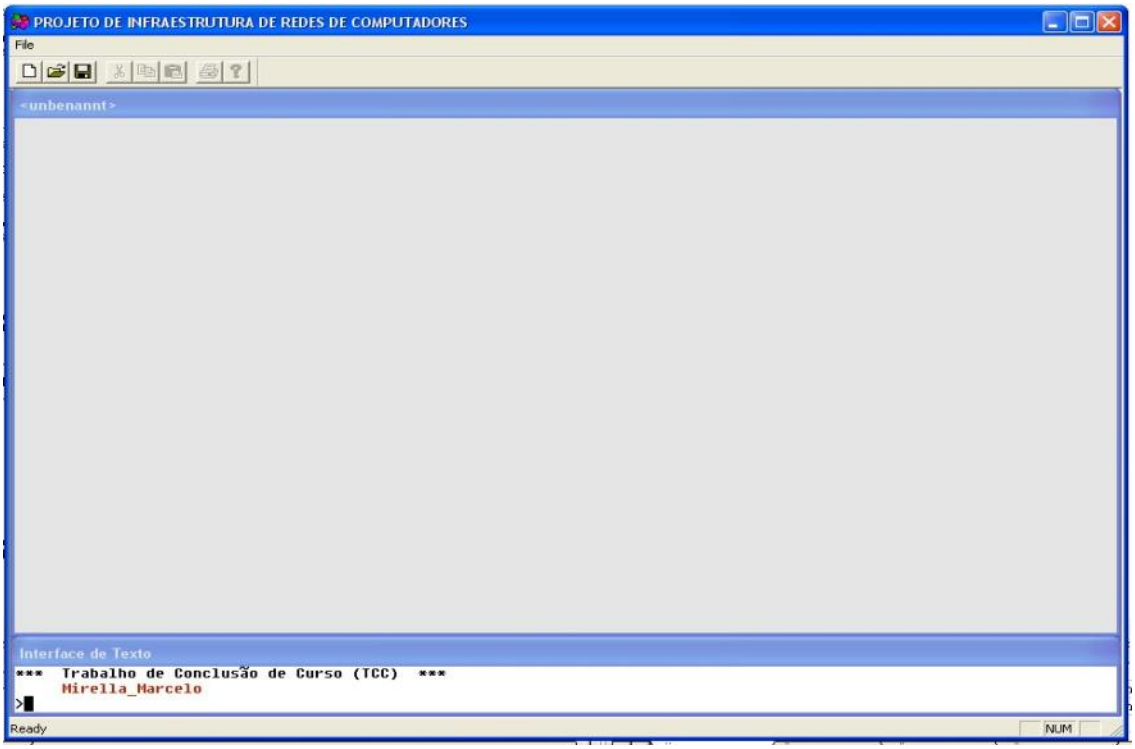

Figura 5.1 – Tela inicial do Vmodel

A funcionalidade deste 3D-Viewer era mantida durante os anos e uma versão ao Windows foi feita.

O VModel funciona no programa Visual C++, onde são necessárias as seguintes bibliotecas: stdlib.h, string.h, stdio.h.

As bibliotecas que seguem abaixo são necessárias no programa VModel para o funcionamento adequado da conversão de arquivos DXF para VRML (IMPORT, EXPORT).

(Utiliza-se o comando # include para incluir as bibliotecas que seguem abaixo):

"stdafx.h" /*liga-se afxwin.h,afxext.h,afxcmn.h*/

"general.h" /*liga-se vectors.h*/

"Primitives.h" /*liga-se materials.h,function.h*/

"VModel.h" /*tem ligação com resource.h*/

"VModelFile.h" /*não tem ligação*/

"VModelDoc.h" /*não tem ligação*/

"TextWin.h" /*não tem ligação*/

A seguir nota-se a importação de um arquivo DXF na figura 5.2.

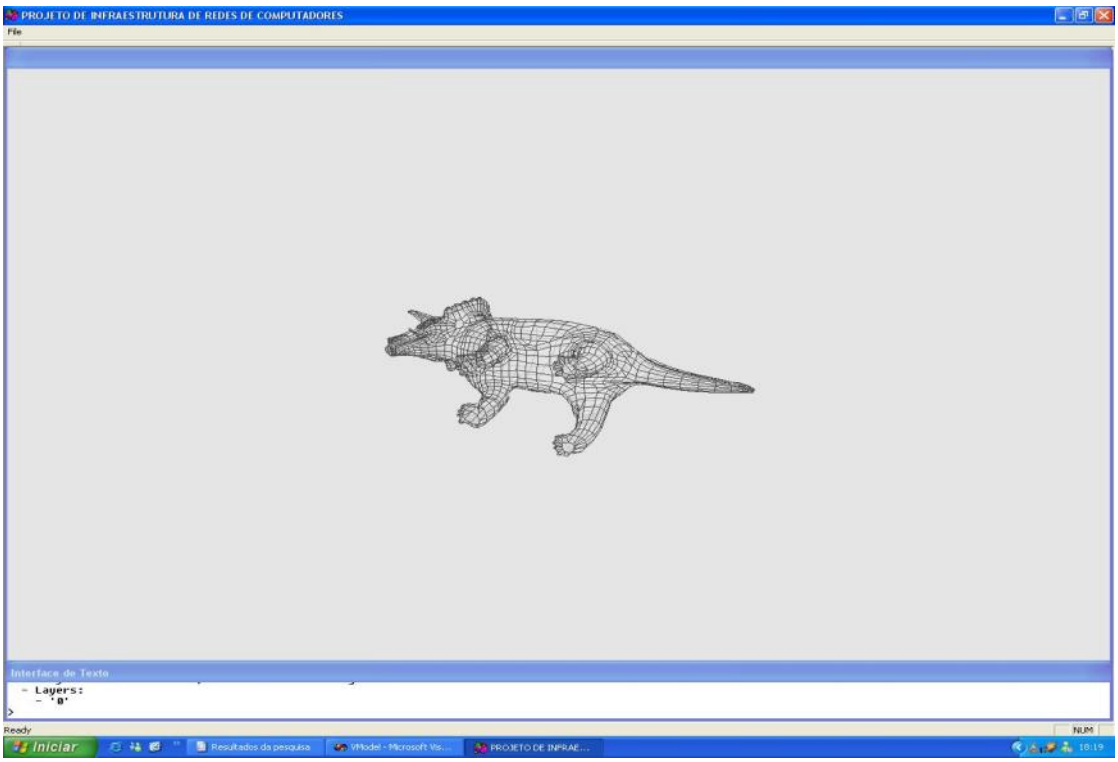

Figura 5.2 – Tela do Vmodel importando um arquivo DXF

O VModel é como um visualizador e um conversor, as características do objeto-criação são ainda fracas. Na prática constrói-se modelos em AutoCAD, importa-se etapa por etapa através dos arquivos DXF, ajusta-se vistas e normas em VModel, adiciona-se materiais

e primitivas especiais através de um texto-editor, edita-se e inspeciona-se este em VModel outra vez e chega-se finalmente ao DXF ou VRML. Nota-se que a versão atual é uma inspeção prévia, aquela que é distante afastada de ser terminada. Não obstante pode ser já completamente útil, especial para editar cenas de Windows.

Na figura 5.3 nota-se a exportação para um arquivo VRML.

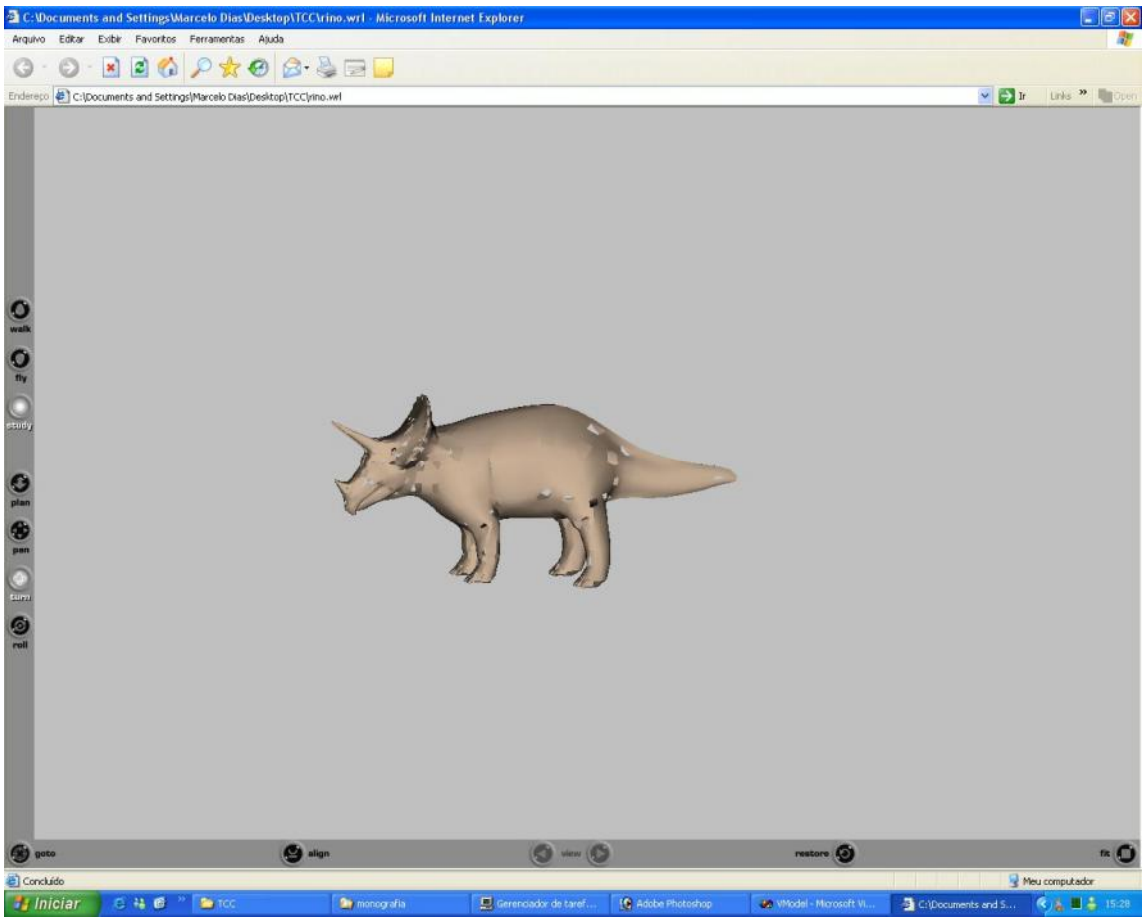

Figura 5.3 – Exportando um arquivo VRML

VModel é *freeware copyrighted*. É permitido você usar livremente e distribuir o pacote para a finalidade não comercial. Abaixo mostra-se na figura 5.4 a tela inicial do VModel com a conversão de uma arquivo DXF para VRML. Há certamente muito a fazer, pois existe varias formas para a conversão de diversos arquivos além do DXF e VRML (VModel_2006).

Figura 5.4 – Tela inicial do Vmodel. Fonte:(VModel_2006).

Na figura 5.5 pode-se notar a entrada de luz no ambiente passando em um prisma refletindo as cores.

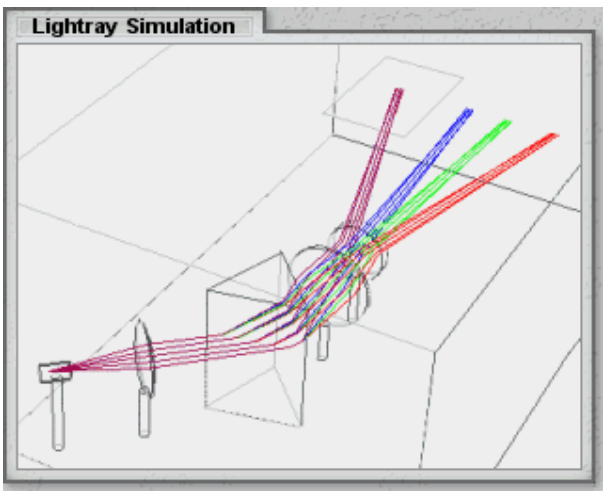

Figura 5.5 – Foto Modelador / Extrator Fonte:(VModel_2006).

CONCLUSÃO

As possibilidades de utilização de computadores como auxiliares no projeto arquitetônico há muito tempo deixaram de restringir-se apenas à transposição do desenho de papel para o desenho de computador. Modelos tridimensionais construídos no computador vêm substituindo as maquetes de apresentação (verdadeiras representações mais próximas da realidade com um tamanho muitas vezes reduzido).

Partindo dessa realidade, idealizou-se este projeto de pesquisa baseado em uma aplicação na linguagem de programação Visual C++ para gerar um código VRML (linguagem para modelagem em Realidade Virtual) a partir de um arquivo DXF (proveniente do AutoCAD) contendo o desenho de uma edificação no espaço tridimensional.

Um desenho 3D de uma edificação feita no AutoCAD poderia ser salvo em um arquivo DXF, transformado em um modelo VRML e poderia ser visualizado por meio uma aplicação C++. Esta é a idéia global desse projeto de pesquisa, que propõe o desenvolvimento da aplicação escrita na linguagem de programação C para converter num desenho 3D DXF para uma representação VRML e também o desenvolvimento da aplicação escrita em C++ para permitir a visualização e a interatividade do usuário no modelo VRML.

É importante também ressaltar que este projeto deverá se estender para continuidade de outro futuro projeto de graduação, sendo que este deverá ser aperfeiçoado para permitir ao profissional acrescentar cor, textura aos modelos tridimensionais VRML, além de converter da linguagem VRML 1.0 para 2.0.

ANEXO

```
|================================================|
|                                                |
|      Drawing Interchange and File Formats      |
|                  Release 12                    |
|   Copyright (c) 1982-1990, 1992 Autodesk, Inc. |
|              All Rights Reserved               |
|                                                |
|================================================|
```

AutoCAD can be used by itself as a complete drawing editor. In some applications, however, other programs must examine drawings created by AutoCAD or generate drawings to be viewed, modified, or plotted with AutoCAD.

For example, if you've made an architectural drawing with AutoCAD, using inserted parts to represent windows, doors, and so on, you can process the drawing file and produce a bill of materials of all items used in the drawing, or even make energy-use calculations based on the area and the number and type of windows used. Another possible application is to use AutoCAD to describe structures and then send the descriptions to a more powerful computer for finite-element structural analysis. You can compute stresses and displacements and send back information to display the deformed structure as an AutoCAD drawing.

Since the AutoCAD drawing database (.dwg file) is written in a compact format that changes significantly as new features are added to AutoCAD, we do not document its format and do not recommend that you attempt to write programs to read it directly. To assist in interchanging drawings between AutoCAD and other programs, a Drawing Interchange file format (DXF) has been defined. All implementations of AutoCAD accept this format and are able to convert it to and from their internal drawing file representation.

AutoCAD also supports the Initial Graphics Exchange Specification (IGES) file format. The information comprising an AutoCAD drawing can be written out in IGES format, and IGES files can be read and converted to the AutoCAD internal format.

ASCII Drawing Interchange (DXF) Files

****

This section describes the AutoCAD DXF (drawing interchange file) format and the commands provided to read and write these files. DXF files are standard ASCII text files. They can easily be translated to the formats of other CAD systems or submitted to other programs for specialized analysis. AutoCAD can also produce or read a binary form of the full DXF file. This feature is described in detail later in this chapter.

DXFOUT Command - Writing a DXF File

==========

You can generate a drawing interchange file from an existing drawing by means of the DXFOUT command:

```
   Command: dxfout
```

When AutoCAD prompts you, respond with a filename or press 5 to accept the default.

The default name for the output file is the same as that of the current drawing, but with a file type of .dxf. If you specify an explicit filename, you do not need to include a file type; .dxf is assumed. If a file with the same name already exists, the existing file is deleted. If you specify the file using a file dialogue box, and a file with the same name already exists, AutoCAD tells you; allowing you to OK or cancel the deletion. Next, DXFOUT asks what precision you want for floating-point numbers and permits output of a partial DXF file containing only selected objects.

```
   Enter decimal places of accuracy (0 to 16)/Entities/Binary <6>:
```

The Binary option is described later in this chapter.

If you respond with "entities" (or just "e"), DXFOUT asks you to select the objects you want written to the DXF file. Only the objects you select are included in the output file - symbol tables (including Block Definitions) will not be included. Once you've selected the desired objects, AutoCAD again prompts you for the numeric precision:

```
   Enter decimal places of accuracy (0 to 16)/Binary <6>:
```

DXFIN Command - Loading a DXF File

A drawing interchange file can be converted into an AutoCAD drawing by means of the DXFIN command:

```
   Command: dxfin
```

When AutoCAD prompts you, respond with the name of the drawing interchange file to be loaded.

Full DXFIN

To load a complete DXF file, you must use DXFIN in an empty drawing, before any entities have been drawn and before any additional Block definitions, layers, linetypes, text styles, dimension styles, named views, named coordinate systems, or named viewport configurations have been created.

NOTE: If the drawing you are using as a prototype is not empty, you might find it helpful to open a new drawing using the No Prototype... button of the Create New Drawing dialogue box, as described in chapter 4 of the "AutoCAD Reference Manual." You should also be aware that some third-party applications include an acad.lsp or .mnl file that modifies your drawing upon startup.

If any errors are detected during the input, the new drawing is discarded. Otherwise, an automatic ZOOM All is performed to set the drawing extents.

Partial DXFIN

If the current drawing is not empty, DXFIN loads only the ENTITIES section of the DXF file, adding the entities found there to the current drawing. In this case, DXFIN displays the message:

```
Not a new drawing -- only ENTITIES section will be input.
```

If errors are detected during such partial DXF input, the drawing is returned to the state it was in before the DXFIN command. Otherwise, the newly added entities are drawn.

Auditing DXF Files

To ensure that corrupt data is not imported into your drawing, you can instruct AutoCAD to perform an audit after importing DXF files into your drawing with DXFIN. When you use DXFIN, the default action is to perform no automatic auditing. To activate automatic auditing, use the CONFIG command:

```
Command: config
```

Your current AutoCAD configuration appears. Press 5 to continue. From the Configuration menu select this option:

```
7. Configure operating parameters
```

From the Operating parameter menu select this option:

```
9. Automatic Audit after IGESIN, DXFIN, or DXBIN
```

Answer Y to this question:

```
Do you want an automatic audit after IGESIN, DXFIN, or DXBIN?
<N>: y
```

Return to the graphics screen by pressing 5 three times.

NOTE: This kind of audit only displays the errors AutoCAD finds; it does not correct them. To correct problems, use the AUDIT command on the drawing while you are in AutoCAD, or manually edit the DXF file.

DXF File Format

This section describes the format of a DXF file in detail. It contains technical information that you need only if you write your own programs to process DXF files or work with entity information obtained by certain AutoLISP and ADS functions.

It would probably be helpful to produce a DXF file from a small drawing, print it out, and refer to it occasionally while reading the information presented next.

General File Structure

A Drawing Interchange File is simply an ASCII text file with a file type of .dxf and specially formatted text. The overall organization of a DXF file is as follows:

1. HEADER section - General information about the drawing is found in this section of the DXF file. Each parameter has a variable name and an associated value (see table 11-3 for a list of the header variables).

2. TABLES section - This section contains definitions of named items.

   - o Linetype table (LTYPE)
   - o Layer table (LAYER)
   - o Text Style table (STYLE)
   - o View table (VIEW)
   - o User Coordinate System table (UCS)
   - o Viewport configuration table (VPORT)
   - o Dimension Style table (DIMSTYLE)
   - o Application Identification table (APPID)

3. BLOCKS section - This section contains Block Definition entities describing the entities that make up each Block in the drawing.

4. ENTITIES section - This section contains the drawing entities, including any Block References.

5. END OF FILE

If you use DXFOUT's Entities option, the resulting DXF file contains only the ENTITIES section and the END OF FILE marker, and the ENTITIES section reflects only the objects you select for output.

NOTE: If you select an INSERT entity, the corresponding Block definition is not included in the output file.

A DXF file is composed of many groups, each of which occupies two lines in the DXF file. The first line of a group is a group code, which is a positive nonzero integer output in FORTRAN I3 - that is, right-justified and blank filled in a three-character field (the exception to this is the four-digit extended entity data group codes, which are output in FORTRAN I4). The second line of the group is the group value, in a format that depends on the type of group specified by the group code. Although DXFOUT output has a fixed format, the DXFIN format is free.

The specific assignment of group codes depends on the item being described in the file. However, the type of the value this group supplies is derived from the group code in the following way:

Table 11-1. Group code ranges

Group code range	Following value
0 - 9	String
10 - 59	Floating-point
60 - 79	Integer
140 - 147	Floating-point
170 - 175	Integer

210 - 239	Floating-point
999	Comment (string)
1010 - 1059	Floating-point
1060 - 1079	Integer
1000 - 1009	String

Thus a program can easily read the value following a group code without knowing the particular use of this group in an item in the file. The appearance of values in the DXF file is not affected by the setting of the UNITS command: coordinates are always represented as decimal (or possibly scientific notation if very large) numbers, and angles are always represented in decimal degrees with zero degrees to the east of origin.

Variables, table entries, and entities are described by a group that introduces the item, giving its type and/or name, followed by multiple groups that supply the values associated with the item. In addition, special groups are used for file separators such as markers for the beginning and end of sections, tables, and the file itself.

Entities, table entries, and file separators are always introduced with a 0 group code that is followed by a name describing the item.

NOTE: The maximum DXF file string length is 256 characters. If your AutoCAD drawing contains strings that exceed this number, those strings are truncated during DXFOUT. If your DXF file contains strings that exceed this number, DXFIN will fail.

Group Codes

Group codes are used both to indicate the type of the value of the group, as explained earlier, and to indicate the general use of the group. The specific function of the group code depends on the actual variable, table item, or entity description. This section indicates the general use of groups, noting as "(fixed)" any that always have the same function.

Table 11-2. AutoCAD entity group codes (by number)

Group code	Value type
0	Identifies the start of an entity, table entry, or file separator. The type of entity is given by the text value that follows this group
1	The primary text value for an entity
2	A name: Attribute tag, Block name, and so on. Also used to identify a DXF section or table name
3-4	Other textual or name values
5	Entity handle expressed as a hexadecimal string

	(fixed)
6	Line type name (fixed)
7	Text style name (fixed)
8	Layer name (fixed)
9	Variable name identifier (used only in HEADER section of the DXF file)
10	Primary X coordinate (start point of a Line or Text entity, center of a Circle, etc.)
11-18	Other X coordinates
20	Primary Y coordinate. 2n values always correspond to 1n values and immediately follow them in the file
21-28	Other Y coordinates
30	Primary Z coordinate. 3n values always correspond to 1n and 2n values and immediately follow them in the file
31-37	Other Z coordinates
38	This entity's elevation if nonzero (fixed). Exists only in output from versions prior to R11
39	This entity's thickness if nonzero (fixed)
40-48	Floating-point values (text height, scale factors, etc.)
49	Repeated value - multiple 49 groups may appear in one entity for variable length tables (such as the dash lengths in the LTYPE table). A 7x group always appears before the first 49 group to specify the table length
50-58	Angles
62	Color number (fixed)
66	"Entities follow" flag (fixed)
67	Identifies whether entity is in model space or paper space
68	Identifies whether viewport is on but fully off screen, is not active, or is off
69	Viewport identification number
70-78	Integer values such as repeat counts, flag bits, or modes
210, 220,	X, Y, and Z components of extrusion direction (fixed)

230	
999	Comments
1000	An ASCII string (up to 255 bytes long) in extended entity data
1001	Registered application name (ASCII string up to 31 bytes long) for XDATA (fixed)
1002	Extended entity data control string ("{" or "}") (fixed)
1003	Extended entity data Layer name
1004	Chunk of bytes (up to 127 bytes long) in extended entity data
1005	Extended entity data database handle
1010, 1020, 1030	Extended entity data X, Y, and Z coordinates
1011, 1021, 1031	Extended entity data X, Y, and Z coordinates of 3D world space position
1012, 1022, 1032	Extended entity data X, Y, and Z components of 3D world space displacement
1013, 1023, 1033	Extended entity data X, Y, and Z components of 3D world space direction
1040	Extended entity data Floating-point value
1041	Extended entity data distance value
1042	Extended entity data scale factor
1070	Extended entity data 16-bit signed integer
1071	Extended entity data 32-bit signed long

Comments
========
The 999 group code indicates that the following line is a comment string. DXFOUT does not currently include such groups in a DXF output file, but DXFIN honors them and ignores the comments. Thus, you can use the 999 group to include comments in a DXF file you've edited. For example:

```
999
This is a comment.
999
This is another comment.
```

File Sections
==========
The DXF file is subdivided into four editable sections, plus the END OF FILE marker. File separator groups are used to delimit these file sections. The following is an example of a void DXF file with only the section markers and table headers present:

```
  0            (Begin HEADER section)
 SECTION
  2
 HEADER
              <<<<Header variable items go here>>>>
 0
 ENDSEC       (End HEADER section)
  0           (Begin TABLES section)
 SECTION
  2
 TABLES
  0
 TABLE
  2
 VPORT
  70
 (viewport table maximum item count)
              <<<<viewport table items go here>>>>
 0
 ENDTAB
 0
 TABLE
 2
 APPID, DIMSTYLE, LTYPE, LAYER, STYLE, UCS, VIEW, or VPORT
 70
 (Table maximum item count)
              <<<<Table items go here>>>>
 0
 ENDTAB
 0
 ENDSEC       (End TABLES section)
 0            (Begin BLOCKS section)
 SECTION
 2
 BLOCKS
              <<<<Block definition entities go here>>>>
 0
 ENDSEC       (End BLOCKS section)
 0            (Begin ENTITIES section)
 SECTION
 2
 ENTITIES
              <<<<Drawing entities go here>>>>
 0
 ENDSEC       (End ENTITIES section)
 0
 EOF          (End of file)
```

HEADER Section

The HEADER section of the DXF file contains settings of variables associated with the drawing. These variables are set with various

commands and are the type of information displayed by the STATUS command. Each variable is specified in the header section by a 9 group giving the variable's name, followed by groups that supply the variable's value. The following list shows the header variables and their meanings.

Although this list is very similar to the list of system variables in Appendix A of this manual, the two lists are not identical. Be sure you're referring to the proper list.

NOTE: $AXISMODE and $AXISUNIT are no longer functional in Release 12.

Table 11-3. DXF system variables

Variable	Type	Description
$ACADVER	1	The AutoCAD drawing database version number; AC1006 = R10, AC1009 = R11 and R12
$ANGBASE	50	Angle 0 direction
$ANGDIR	70	1 = clockwise angles, 0 = counterclockwise
$ATTDIA	70	Attribute entry dialogs, 1 = on, 0 = off
$ATTMODE	70	Attribute visibility: 0 = none, 1 = normal, 2 = all
$ATTREQ	70	Attribute prompting during INSERT, 1 = on, 0 = off
$AUNITS	70	Units format for angles
$AUPREC	70	Units precision for angles
$AXISMODE	70	Axis on if nonzero (not functional in Release 12)
$AXISUNIT	10, 20	Axis X and Y tick spacing (not functional in Release 12)
$BLIPMODE	70	Blip mode on if nonzero
$CECOLOR	62	Entity color number; 0 = BYBLOCK, 256 = BYLAYER
$CELTYPE	6	Entity linetype name, or BYBLOCK or BYLAYER
$CHAMFERA	40	First chamfer distance
$CHAMFERB	40	Second chamfer distance
$CLAYER	8	Current layer name
$COORDS	70	0 = static coordinate display, 1 = continuous update, 2 = "d<a"

		format
$DIMALT	70	Alternate unit dimensioning performed if nonzero
$DIMALTD	70	Alternate unit decimal places
$DIMALTF	40	Alternate unit scale factor
$DIMAPOST	1	Alternate dimensioning suffix
$DIMASO	70	1 = create associative dimensioning, 0 = draw individual entities
$DIMASZ	40	Dimensioning arrow size
$DIMBLK	2	Arrow block name
$DIMBLK1	1	First arrow block name
$DIMBLK2	1	Second arrow block name
$DIMCEN	40	Size of center mark/lines
$DIMCLRD	70	Dimension line color, range is 0 = BYBLOCK, 256 = BYLAYER
$DIMCLRE	70	Dimension extension line color, range is 0 = BYBLOCK, 256 = BYLAYER
$DIMCLRT	70	Dimension text color, range is 0 = BYBLOCK, 256 = BYLAYER
$DIMDLE	40	Dimension line extension
$DIMDLI	40	Dimension line increment
$DIMEXE	40	Extension line extension
$DIMEXO	40	Extension line offset
$DIMGAP	40	Dimension line gap
$DIMLFAC	40	Linear measurements scale factor
$DIMLIM	70	Dimension limits generated if nonzero
$DIMPOST	1	General dimensioning suffix
$DIMRND	40	Rounding value for dimension distances
$DIMSAH	70	Use separate arrow blocks if nonzero
$DIMSCALE	40	Overall dimensioning scale factor
$DIMSE1	70	First extension line suppressed if nonzero
$DIMSE2	70	Second extension line suppressed

		if nonzero
$DIMSHO	70	1 = Recompute dimensions while dragging, 0 = drag original image
$DIMSOXD	70	Suppress outside-extensions dimension lines if nonzero
$DIMSTYLE	2	Dimension style name
$DIMTAD	70	Text above dimension line if nonzero
$DIMTFAC	40	Dimension tolerance display scale factor
$DIMTIH	70	Text inside horizontal if nonzero
$DIMTIX	70	Force text inside extensions if nonzero
$DIMTM	40	Minus tolerance
$DIMTOFL	70	If text outside extensions, force line extensions between extensions if nonzero
$DIMTOH	70	Text outside horizontal if nonzero
$DIMTOL	70	Dimension tolerances generated if nonzero
$DIMTP	40	Plus tolerance
$DIMTSZ	40	Dimensioning tick size: 0 = no ticks
$DIMTVP	40	Text vertical position
$DIMTXT	40	Dimensioning text height
$DIMZIN	70	Zero suppression for "feet & inch" dimensions
$DWGCODEPAGE	70	Drawing code page. Set to the system code page when a new drawing is created, but not otherwise maintained by AutoCAD
$DRAGMODE	70	0 = off, 1 = on, 2 = auto
$ELEVATION	40	Current elevation set by ELEV command
$EXTMAX	10, 20, 30	X, Y, and Z drawing extents upper-right corner (in WCS)
$EXTMIN	10, 20, 30	X, Y, and Z drawing extents lower-left corner (in WCS)
$FILLETRAD	40	Fillet radius
$FILLMODE	70	Fill mode on if nonzero

$HANDLING	70	Handles enabled if nonzero
$HANDSEED	5	Next available handle
$INSBASE	10, 20, 30	Insertion base set by BASE command (in WCS)
$LIMCHECK	70	Nonzero if limits checking is on
$LIMMAX	10, 20	XY drawing limits upper-right corner (in WCS)
$LIMMIN	10, 20	XY drawing limits lower-left corner (in WCS)
$LTSCALE	40	Global linetype scale
$LUNITS	70	Units format for coordinates and distances
$LUPREC	70	Units precision for coordinates and distances
$MAXACTVP	70	Sets maximum number of viewports to be regenerated
$MENU	1	Name of menu file
$MIRRTEXT	70	Mirror text if nonzero
$ORTHOMODE	70	Ortho mode on if nonzero
$OSMODE	70	Running object snap modes
$PDMODE	70	Point display mode
$PDSIZE	40	Point display size
$PELEVATION	40	Current paper space elevation
$PEXTMAX	10, 20, 30	Maximum X, Y, and Z extents for paper space
$PEXTMIN	10, 20, 30	Minimum X, Y, and Z extents for paper space
$PLIMCHECK	70	Limits checking in paper space when nonzero
$PLIMMAX	10, 20	Maximum X and Y limits in paper space
$PLIMMIN	10, 20	Minimum X and Y limits in paper space
$PLINEGEN	70	Governs the generation of linetype patterns around the vertices of a 2D Polyline 1 = linetype is generated in a continuous pattern around

		vertices of the Polyline 0 = each segment of the Polyline starts and ends with a dash
$PLINEWID	40	Default Polyline width
$PSLTSCALE	70	Controls paper space linetype scaling 1 = no special linetype scaling 0 = viewport scaling governs linetype scaling
$PUCSNAME	2	Current paper space UCS name
$PUCSORG	10, 20, 30	Current paper space UCS origin
$PUCSXDIR	10, 20, 30	Current paper space UCS X axis
$PUCSYDIR	10, 20, 30	Current paper space UCS Y axis
$QTEXTMODE	70	Quick text mode on if nonzero
$REGENMODE	70	REGENAUTO mode on if nonzero
$SHADEDGE	70	0 = faces shaded, edges not highlighted 1 = faces shaded, edges highlighted in black 2 = faces not filled, edges in entity color 3 = faces in entity color, edges in black
$SHADEDIF	70	Percent ambient/diffuse light, range 1-100, default 70
$SKETCHINC $SKPOLY	40 70	Sketch record increment 0 = sketch lines, 1 = sketch polylines
$SPLFRAME	70	Spline control polygon display, 1 = on, 0 = off
$SPLINESEGS	70	Number of line segments per spline patch
$SPLINETYPE	70	Spline curve type for PEDIT Spline (See your AutoCAD Reference Manual)
$SURFTAB1	70	Number of mesh tabulations in first direction
$SURFTAB2	70	Number of mesh tabulations in second direction
$SURFTYPE	70	Surface type for PEDIT Smooth (See your AutoCAD Reference Manual)

$SURFU	70	Surface density (for PEDIT Smooth) in M direction
$SURFV	70	Surface density (for PEDIT Smooth) in N direction
$TDCREATE	40	Date/time of drawing creation
$TDINDWG	40	Cumulative editing time for this drawing
$TDUPDATE	40	Date/time of last drawing update
$TDUSRTIMER	40	User elapsed timer
$TEXTSIZE	40	Default text height
$TEXTSTYLE	7	Current text style name
$THICKNESS	40	Current thickness set by ELEV command
$TILEMODE	70	1 for previous release compatibility mode, 0 otherwise
$TRACEWID	40	Default Trace width
$UCSNAME	2	Name of current UCS
$UCSORG	10, 20, 30	Origin of current UCS (in WCS)
$UCSXDIR	10, 20, 30	Direction of current UCS's X axis (in World coordinates)
$UCSYDIR	10, 20, 30	Direction of current UCS's Y axis (in World coordinates)
$UNITMODE	70	Low bit set = display fractions, feet-and-inches, and surveyor's angles in input format
$USERI1 - 5	70	Five integer variables intended for use by third-party developers
$USERR1 - 5	40	Five real variables intended for use by third-party developers
$USRTIMER	70	0 = timer off, 1 = timer on
$VISRETAIN	70	0 = don't retain Xref-dependent visibility settings, 1 = retain Xref-dependent visibility settings
$WORLDVIEW	70	1 = set UCS to WCS during DVIEW/VPOINT, 0 = don't change UCS

The following header variables existed prior to AutoCAD Release 11 but now have independent settings for each active viewport. DXFIN honors these variables when read from DXF files, but if a VPORT

symbol table with *ACTIVE entries is present (as is true for any DXF file produced by Release 11 or higher), the values in the VPORT table entries override the values of these header variables.

Table 11-4. Revised VPORT header variables

Variable	Type	Description
$FASTZOOM	70	Fast zoom enabled if nonzero
$GRIDMODE	70	Grid mode on if nonzero
$GRIDUNIT	10, 20	Grid X and Y spacing
$SNAPANG	50	Snap grid rotation angle
$SNAPBASE	10, 20	Snap/grid base point (in UCS)
$SNAPISOPAIR	70	Isometric plane: 0 = left, 1 = top, 2 = right
$SNAPMODE	70	Snap mode on if nonzero
$SNAPSTYLE	70	Snap style: 0 = standard, 1 = isometric
$SNAPUNIT	10, 20	Snap grid X and Y spacing
$VIEWCTR	10, 20	XY center of current view on screen
$VIEWDIR	10, 20, 30	Viewing direction (direction from target, in WCS)
$VIEWSIZE	40	Height of view

The date/time variables ($TDCREATE and $TDUPDATE) are output as real numbers in the following format:

<Julian date>.<Fraction>

The elapsed time variables ($TDINDWG and $TDUSRTIMER) have a similar format:

<Number of days>.<Fraction>

The date and time variables are described on page 299.

TABLES Section

The TABLES section contains several tables, each of which contains a variable number of table entries.

The order of the tables may change, but the LTYPE table will always precede the LAYER table. Each table is introduced with a 0 group with the label TABLE. This is followed by a 2 group identifying the particular table (VPORT, LTYPE, LAYER, STYLE, VIEW, DIMSTYLE, UCS or APPID) and a 70 group that specifies the maximum number of table entries that may follow. Table names are always output in uppercase characters.

The tables in a drawing can contain deleted items, but these are not written to the DXF file. Thus, fewer table entries may follow the table header than are indicated by the 70 group, so don't use the count in the 70 group as an index to read in the table. This group is provided so that a program which reads DXF files can allocate an array large enough to hold all the table entries that follow.

Following this header for each table are the table entries. Each table item consists of a 0 group identifying the item type (same as table name, e.g., LTYPE or LAYER), a 2 group giving the name of the table entry, a 70 group specifying flags relevant to the table entry (defined for each following table), and additional groups that give the value of the table entry. The end of each table is indicated by a 0 group with the value ENDTAB.

The 70 group flag bit values that apply to all table entries are described in the following chart. Additional 70 group values that apply to LAYER, STYLE, and VIEW table entries are described in the appropriate sections below.

Table 11-5. Group 70 bit codes that apply to all table entries

Flag bit value	Meaning
16	If set, table entry is externally dependent on an Xref
32	If this bit and bit 16 are both set, the externally dependent Xref has been successfully resolved
64	If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands; it can be ignored by most programs that read DXF files, and need not be set by programs that write DXF files)

The following are the groups used for each type of table item. All groups are present for each table item.

APPID 2 (user-supplied application name), 70 (standard flag values).
These table entries maintain a set of names for all applications registered with a drawing.

DIMSTYLE 2 (dimension style name), 70 (standard flag values), and the following, described by dimension variable name:
3 (dimpost), 4 (dimapost), 5 (dimblk), 6 (dimblk1), 7 (dimblk2), 40 (dimscale), 41 (dimasz), 42 (dimexo), 43 (dimdli), 44 (dimexe), 45 (dimrnd), 46 (dimdle), 47 (dimtp), 48 (dimtm), 140 (dimtxt), 141 (dimcen), 142 (dimtsz), 143 (dimaltf), 144 (dimlfac), 145 (dimtvp), 146 (dimtfac), 147 (dimgap), 71 (dimtol), 72 (dimlim), 73 (dimtih), 74 (dimtoh), 75 (dimse1), 76 (dimse2), 77 (dimtad), 78 (dimzin), 170 (dimalt), 171 (dimaltd), 172 (dimtofl), 173 (dimsah), 174 (dimtix), 175 (dimsoxd), 176 (dimclrd), 177 (dimclre), 178 (dimclrt).

LTYPE 2 (linetype name), 70 (standard flag values), 3 (descriptive text for linetype), 72 (alignment code; value is always 65, the ASCII code for `A'), 73 (number of dash length items), 40 (total pattern length), and optionally: 49 (dash length 1), 49 (dash length 2), and so on.

LAYER 2 (layer name), 70 (standard flag values), 62 (color number, negative if layer is off), 6 (linetype name). In addition to the standard flags, the 70 group flag is bit coded as follows:

Table 11-6. Group 70 bit codes for LAYER table

Flag bit value	Meaning
1	If set, layer is frozen
2	If set, layer is frozen by default in new Viewports
4	If set, layer is locked

If no value (0) is set, the layer is on and thawed. The fourth bit (8) and the eighth bit (128) are not used. Xref-dependent layers are output during DXFOUT. For these layers, the associated linetype name in the DXF file is always CONTINUOUS.

STYLE 2 (style name), 70 (standard flag values), 40 (fixed text height; 0 if not fixed), 41 (width factor), 50 (oblique angle), 71 (text generation flags), 42 (last height used), 3 (primary font filename), 4 (big-font file name; blank if none).

If the third bit (4) is set in the 70 group flags, this is a vertically oriented text style.

A STYLE table item is used to record shape file LOAD requests also. In this case the first bit (1) is set in the 70 group flags and only the 3 group (shape filename) is meaningful (all the other groups are output, however).

The text generation flags are a bit-coded field with the following bit meanings:

Table 11-7. Group 71 bit codes for STYLE table

Flag bit value	Meaning
2	Text is backward (mirrored in X)
4	Text is upside down (mirrored in Y)

UCS 2 (UCS name), 70 (standard flag values), 10, 20, 30 (origin), 11, 21, 31 (X axis direction), 12, 22, 32 (Y axis direction). All in World coordinates.

VIEW — 2 (name of view), 70 (standard flag values), 40 and 41 (view height and width, in DCS), 10 and 20 (view center point, in DCS), 11, 21, 31 (view direction from target, in WCS), 12, 22, 32 (target point, in WCS), 42 (lens length), 43 and 44 (front and back clipping planes - offsets from target point), 50 (twist angle), 71 view mode (see VIEWMODE system variable in appendix A).

If the first bit (1) is set in the 70 group flags, this is a paper space view.

(See chapter 2 of the "AutoLISP Programmer's Reference" for information on DCS, the Display Coordinate System.)

VPORT — 2 (viewport name), 70 (standard flag values), 10 and 20 (lower-left corner of viewport; 0.0 to 1.0), 11 and 21 (upper-right corner), 12 and 22 (view center point, in WCS), 13 and 23 (snap base point), 14 and 24 (snap spacing, X and Y), 15 and 25 (grid spacing, X and Y), 16, 26, 36 (view direction from target point), 17, 27, 37 (view target point), 40 (view height), 41 (viewport aspect ratio), 42 (lens length), 43 and 44 (front and back clipping planes; offsets from target point),50 (snap rotation angle), 51 (view twist angle), 68 (status field), 69 (ID), 71 (view mode; see VIEWMODE system variable in appendix A), 72 (circle zoom percent), 73 (fast zoom setting), 74 (UCSICON setting), 75 (snap on/off), 76 (grid on/off), 77 (snap style), 78 (snap isopair).

The VPORT table is unique in that it may contain several entries with the same name (indicating a multiple-viewport configuration). The entries corresponding to the active viewport configuration all have the name *ACTIVE. The first such entry describes the current viewport.

BLOCKS Section

The Blocks section of the DXF file contains all the Block Definitions. This section contains the entities that make up the Blocks used in the drawing, including anonymous Blocks generated by the HATCH command and by associative dimensioning. The format of the entities in this section is identical to those in the Entities section described later, so see that section for details. All entities in the Blocks section appear between Block and Endblk entities. Block and Endblk entities appear only in the Blocks section. Block definitions are never nested (that is, no Block or Endblk entity ever appears within another Block-Endblk pair), although a Block definition can contain an INSERT entity.

External References are written in the DXF file as any Block Definition, except they also include a text string (group code 1) of the path and filename of the External Reference. This is the text string format:

```
Xref filename
```

ENTITIES Section

Entity items appear in both the BLOCK and ENTITIES sections of the DXF file. The appearance of entities in the two sections is identical.

The following gives the format of each entity as it appears in the file. Some groups that define an entity always appear, and some are optional and appear only if they differ from their default values. In the following discussion, groups that always occur are given by their group number and function, while optional groups are indicated by -optional N following the group description. N is the default value if the group is omitted.

Programs that read DXF files should not assume that the groups describing an entity occur in the order given here. The end of the groups that make up an entity is indicated by the next 0 group, beginning the next entity or indicating the end of the section.

Remember that a DXF file is a complete representation of the drawing database, and that as AutoCAD is further enhanced, new groups will be added to entities to accommodate additional features. Accommodating DXF files from future releases of AutoCAD will be easier if you write your DXF processing program in a table-driven way, ignoring any groups not presently defined, and making no assumptions about the order of groups in an entity.

Each entity begins with a 0 group identifying the entity type. The names used for the entities are given on the following pages. Every entity contains an 8 group that gives the name of the layer on which the entity resides. Each entity may have elevation, thickness, linetype, or color information associated with it.

If handles are enabled, every entity has a 5 group containing its handle (as a string representing a hexadecimal number).

The following groups are included only if the entity has nondefault values for these properties. When a group is omitted, its default value upon input (when using DXFIN) is indicated in the third column. If the value of a group is equal to the default, it is omitted upon output (when using DXFOUT).

Table 11-8. Group codes common to all entities

Group code	Meaning	If omitted, defaults to...
6	Linetype name (if not BYLAYER). The special name BYBLOCK indicates a floating linetype	BYLAYER
38	Elevation (if nonzero). Exists only in output from versions prior to R11. Otherwise, Z coordinates are supplied as 3x-groups as part of each of the entity's defining points	0
39	Thickness (if nonzero)	0
62	Color number (if not BYLAYER). Zero indicates the BYBLOCK (floating) color. 256 indicates the BYLAYER color	BYLAYER

67	Absent or zero indicates entity is in model space. One indicates entity is in paper space, other values are reserved	0
210, 220, 230	These groups are included for each Line, Point, Circle, Shape, Text, Arc, Trace, Solid, Block Reference, Polyline, Dimension, Attribute, and Attribute Definition entity if its extrusion direction is not parallel to the World Z axis. They indicate the X, Y, and Z components of the entity's extrusion direction	0,0,1

The rest of the groups that make up an entity item are described next. Many of the entities include "flag" groups. These are integer codes (6x or 7x groups) that encode various pieces of information regarding the entity, and are specific to the particular entity type. In the following descriptions, the term bit-coded means that the flag contains various true/false values coded as the sum of the bit values given. Any bits not defined in the following section should be ignored in these fields and set to zero when constructing a DXF file.

LINE 10, 20, 30 (start point), 11, 21, 31 (endpoint).

POINT 10, 20, 30 (point).

Point entities have an optional 50 group that determines the orientation of PDMODE images. The group value is the negative of the Entity Coordinate Systems (ECS) angle of the UCS X axis in effect when the point was drawn. The X axis of the UCS in effect when the point was drawn is always parallel to the XY plane for the point's ECS, and the angle between the UCS X axis and the ECS X axis is a single 2D angle. The value in group 50 is the angle from horizontal (the effective X axis) to the ECS X axis. Entity Coordinate Systems (ECS) are described later in this section.

CIRCLE 10, 20, 30 (center), 40 (radius).

ARC 10, 20, 30 (center), 40 (radius), 50 (start angle), 51 (end angle).

TRACE Four points defining the corners of the trace: (10, 20, 30), (11, 21, 31), (12, 22, 32), and (13, 23, 33).

SOLID Four points defining the corners of the solid: (10, 20, 30), (11, 21, 31), (12, 22, 32), and (13, 23, 33). If only three points were entered (forming a triangular solid), the third and fourth points will be the same.

TEXT 10, 20, 30 (insertion point), 40 (height), 1 (text value), 50 (rotation angle -optional 0), 41 (relative X-scale factor -optional 1), 51 (oblique angle -optional 0), 7 (text style name -optional STANDARD), 71 (text generation flags -optional 0), 72 (horizontal justification type -optional 0), 73 (vertical justification type -optional 0)

11, 21, 31 (alignment point -optional, appears only if 72 or 73 group is present and nonzero).

The "text generation flags" are a bit-coded field with meanings as follows:

Table 11-9. Group 71 bit codes for Text entity

Flag bit value	Meaning
2	Text is backward (mirrored in X)
4	Text is upside down (mirrored in Y)

The justification-type value (group codes 72 and 73, not bit-coded) indicates the text-justification style used on the text, as shown in the following table:

Table 11-10. Group 72 & 73 bit codes for Text entity

Group 73 (vertical alignment)	Group 72 (horizontal alignment)					
	0	1	2	3	4	5
3 (Top)	TLeft	TCenter	TRight			
2 (Middle)	MLeft	MCenter	MRight			
1 (Bottom)	BLeft	BCenter	BRight			
0 (Baseline)	Left	Center	Right	Aligned	Middle	Fit

If the justification is anything other than baseline/left (groups 72 and 73 both 0), group codes 11, 21, and 31 specify the alignment point (or the second alignment point for Align or Fit).

DXFOUT handles ASCII control characters in text strings by expanding the character into a ^ (caret) followed by the appropriate letter. For example, an ASCII Control-G (BEL, decimal code 7) is output as ^G. If the text itself contains a caret character, it is expanded to ^ (caret, space). DXFIN performs the complementary conversion.

SHAPE 10, 20, 30 (insertion point), 40 (size), 2 (shape name), 50 (rotation angle -optional 0), 41 (relative X-scale factor -optional 1), 51 (oblique angle -optional 0).

BLOCK 2 (Block name), 3 (this is also the Block name), 70 (Block type flag), 10, 20, 30 (Block base point), and if the Block is an Xref Block it will also contain group code 1 (Xref pathname). Block entities appear only in the BLOCKS section, not in the ENTITIES section. The "Block type flag" (group 70) is bit-coded, with the following bit meanings:

Table 11-11. Group 70 bit codes for Block table

Flag bit value	Meaning
1	This is an anonymous Block generated by hatching, associative dimensioning, other internal operations, or an application
2	This Block has Attributes
4	This Block is an external reference (Xref)
8	not used
16	This Block is externally dependent
32	This is a resolved external reference, or dependent of an external reference
64	This definition is referenced

ENDBLK No groups. Appears only in BLOCKS section.

INSERT 66 (Attributes follow flag -optional 0), 2 (Block name), 10, 20, 30 (insertion point), 41 (X- scale factor -optional 1), 42 (Y scale factor -optional 1), 43 (Z-scale factor -optional 1), 50 (rotation angle -optional 0), 70 and 71 (column and row counts -optional 1), 44 and 45 (column and row spacing -optional 0).

If the value of the "Attributes follow" flag is 1, a series of Attribute (Attrib) entities is expected to follow the Insert, terminated by a sequence end (Seqend) entity.

ATTDEF 10, 20, 30 (text start), 40 (text height), 1 (default value, see "Text" on page 260 for handling of ASCII control characters), 3 (prompt string), 2 (tag string), 70 (Attribute flags), 73 (field length -optional 0), 50 (text rotation - optional 0), 41 (relative X scale factor -optional 1), 51 (oblique angle -optional 0), 7 (text style name -optional STANDARD), 71 (text generation flags -optional 0, see "Text" on page 260), 72 (horizontal text justification type - optional 0, see "Text" on page 260), 74 (vertical text justification type -optional 0 see group 73 in "Text" on page 260), 11, 21, 31 (alignment point -optional, appears only if 72 or 74 group is present and nonzero).

The "Attribute flags" (group code 70) are a bit-coded field in which the bits have the following meanings:

Table 11-12. Group 70 bit codes for Attdef entity

Flag bit value	Meaning
1	Attribute is invisible (does not display)
2	This is a constant Attribute

	4	Verification is required on input of this Attribute
	8	Attribute is preset (no prompt during insertion)

ATTRIB 10, 20, 30 (text start), 40 (text height), 1 (value, see "Text" on page 260 for handling ASCII control characters), 2 (Attribute tag), 70 (Attribute flags; see Attdef), 73 (field length -optional 0), 50 (text rotation -optional 0), 41 (relative X scale factor -optional 1), 51 (oblique angle -optional 0), 7 (text style name -optional STANDARD), 71 (text generation flags -optional 0, see "Text" on page 260), 72 (horizontal text justification type -optional 0, see "Text" on page 260), 74 (vertical text justification type -optional 0, see group 73 in "Text" on page 260), 11, 21, 31 (alignment point -optional, appears only if 72 or 74 group is present and nonzero).

POLYLINE 66 (vertices-follow flag), 10, 20, 30 (polyline elevation - 30 supplies elevation, 10 and 20 are always set to zero), 70 (Polyline flag -optional 0), 40 (default starting width - optional 0), 41 (default ending width -optional 0), 71 and 72 (polygon mesh M and N vertex counts -optional 0), 73 and 74 (smooth surface M and N densities -optional 0), 75 (curves and smooth surface type -optional 0). The default widths apply to any vertex that doesn't supply widths (see later).

The "vertices follow" flag is always 1, indicating that a series of Vertex entities is expected to follow the Polyline, terminated by a sequence end (Seqend) entity. The polyline flag (group code 70) is a bit-coded field with bits defined as follows:

Table 11-13. Group 70 bit codes for Polyline entity

Flag bit value	Meaning
1	This is a closed Polyline (or a polygon mesh closed in the M direction)
2	Curve-fit vertices have been added
4	Spline-fit vertices have been added
8	This is a 3D Polyline
16	This is a 3D polygon mesh. Group 75 indicates the smooth surface type as follows: 0 = no smooth surface fitted 5 = quadratic B-spline surface 6 = cubic B-spline surface 8 = Bezier surface
32	The polygon mesh is closed in the N direction

64	This Polyline is a polyface mesh
128	The linetype pattern is generated continuously around the vertices of this Polyline

A polyface mesh is represented in DXF as a variant of a Polyline entity. The Polyline header is identified as introducing a polyface mesh by the presence of the 64 bit in the Polyline flags (70) group. The 71 group specifies the number of vertices in the mesh, and the 72 group, the number of faces. While these counts are correct for all meshes created with the PFACE command, applications are not required to place correct values in these fields, and AutoCAD actually never relies upon their accuracy.

Following the Polyline header is a sequence of Vertex entities that specify the vertex coordinates and faces that compose the mesh. Vertices such as these are described in the following subsection on Vertex.

Applications might want to represent polygons with an arbitrarily large number of sides in polyface meshes. However, the AutoCAD entity structure imposes a limit on the number of vertices that a given face entity can specify. You can represent more complex polygons by decomposing them into triangular wedges. Their edges should be made invisible to prevent visible artifacts of this subdivision from being drawn. The PFACE command performs this subdivision automatically, but when applications generate polyface meshes directly, the applications must do this themselves.

The number of vertices per face is the key parameter in this subdivision process. The PFACEVMAX system variable provides an application with the number of vertices per face entity. This value is read-only, and is set to 4.

Polyface meshes created with the PFACE command are always generated with all the vertex coordinate entities first, followed by the face definition entities. The code within AutoCAD that processes polyface meshes does not, at present, require this ordering; it works even with interleaved vertex coordinates and face definitions as long as no face specifies a vertex with an index that appears after it in the database. Programs that read polyface meshes from DXF would be wise to be as tolerant of odd vertex and face ordering as AutoCAD is.

VERTEX 10, 20, 30 (location), 40 (starting width -optional, see earlier), 41 (ending width -optional, see above), 42 (bulge -optional 0), 70 (vertex flags -optional 0), 50 (curve fit tangent direction -optional). The bulge is the tangent of 1/4 the included angle for an arc segment, made negative if the arc goes clockwise from the start point to the endpoint; a bulge of 0 indicates a straight segment, and a bulge of 1 is a semicircle. The meaning of the bit-coded Vertex flag (group code 70) is shown in the following table

Table 11-14. Group 70 bit codes for Vertex entity

Flag bit value	Meaning
1	Extra vertex created by curve-fitting
2	Curve-fit tangent defined for this vertex. A curve-fit tangent direction of 0 may be omitted from the DXF output, but is significant if this bit is set
4	Unused (never set in DXF files)
8	Spline vertex created by spline-fitting
16	Spline frame control point
32	3D Polyline vertex
64	3D polygon mesh vertex
128	Polyface mesh vertex

Every Vertex that is part of a polyface mesh has the 128 bit set in its Vertex flags (70) group. If the entity specifies the coordinates of a vertex of the mesh, the 64 bit is set as well and the 10, 20, and 30 groups give the vertex coordinates. The vertex indexes are determined by the order in which the Vertex entities appear within the Polyline, with the first numbered 1.

If the Vertex defines a face of the mesh, its Vertex flags (70) group has the 128 bit set but not the 64 bit. The 10, 20, and 30 (location) groups of the face entity are irrelevant and are always written as zero in a DXF file. The vertex indexes that define the mesh are given by 71, 72, 73, and 74 groups, the values of which are integers specifying one of the previously defined vertices by index. If the index is negative, the edge that begins with that vertex is invisible. The first zero vertex marks the end of the vertices of the face. Since the 71 through 74 groups are optional fields with default values of zero, they are present in DXF only if nonzero.

SEQEND No fields. This entity marks the end of vertices (Vertex type name) for a Polyline, or the end of Attribute entities (Attrib type name) for an Insert entity that has Attributes (indicated by 66 group present and nonzero in Insert entity).

3DFACE Four points defining the corners of the face: (10, 20, 30), (11, 21, 31), (12, 22, 32), and (13, 23, 33). 70 (invisible edge flags -optional 0). If only three points are entered (forming a triangular face), the third and fourth points will be the same. The meanings of the bit-coded "invisible edge flags" are shown in the following table:

Table 11-15. Group 70 bit codes for 3D Face entity

Flag bit value	Meaning
1	First edge is invisible
2	Second edge is invisible
4	Third edge is invisible
8	Fourth edge is invisible

VIEWPORT 10,20,30 (center point of entity in paper space coordinates), 40 (width in paper space units), 41 (height in paper space units), 68 (viewport status field), 69 (viewport ID, permanent during editing sessions, but mutable between sessions; the paper space viewport entity always has an ID of 1).

The value of the viewport status field (68) is interpreted as follows:

-1 On, but is fully off-screen or is one of the viewports not active because the $MAXACTVP count is currently being exceeded.

0 Off.

<positive value> On, active and the value indicates the order of "stacking" for the viewports, with 1 applying to the active viewport, which is also the highest, 2 applying to the next viewport in the stack, and so on.

In addition, the extended entity data groups in the following table apply to viewports.

NOTE: In contrast to normal entity data, the same extended entity group code can appear multiple times, and order is important.

Table 11-16. Extended entity group codes for Viewports

Group	Description
1001	Application name. This field will always be the string "ACAD"
1000	Begin viewport data. This field will always be the string "MVIEW". Other data groups may appear in the future
1002	Begin window descriptor data. This field will always be the string "{"
1070	Extended entity data version number. For Releases 11 and 12, this field will always be

	the integer 16
1010	View target point X value
1020	View target point Y value
1030	View target point Z value
1010	View direction vector X value
1020	View direction vector Y value
1030	View direction vector Z value
1040	View twist angle
1040	View height
1040	View center point X value
1040	View center point Y value
1040	Perspective lens length
1040	Front clip plane Z value
1040	Back clip plane Z value
1070	View mode
1070	Circle zoom
1070	Fast zoom setting
1070	UCSICON setting
1070	Snap ON/OFF
1070	Grid ON/OFF
1070	Snap style
1070	Snap ISOPAIR
1040	Snap angle
1040	Snap base point UCS X coordinate
1040	Snap base point UCS Y coordinate
1040	Snap X spacing
1040	Snap Y spacing
1040	Grid X spacing
1040	Grid Y spacing
1070	Hidden in plot flag
1002	Begin frozen layer list (possibly empty).

	This field will always be the string "{"
1003...	The names of layers frozen in this viewport. This list may include Xref-dependent layers. Any number of 1003 groups may appear here
1002	End frozen layer list. This field will always be the string "}"
1002	End viewport data. This field will always be the string "}"

DIMENSION 2 (name of pseudo-Block containing the current dimension entity geometry), 3 (dimension style name), 10, 20, 30 (definition point for all dimension types), 11, 21, 31 (middle point of dimension text), 12, 22, 32 (dimension block translation vector), 70 (Dimension type), 1 (dimension text explicitly entered by the user. If null or "<>", the dimension measurement is drawn as the text, if " " [one blank space], the text is suppressed. Anything else is drawn as the text). 13, 23, 33 (definition point for linear and angular dimensions), 14, 24, 34 (definition point for linear and angular dimensions), 15, 25, 35 (definition point for diameter, radius, and angular dimensions), 16, 26, 36 (point defining dimension arc for angular dimensions), 40 (leader length for radius and diameter dimensions), 50 (angle of rotated, horizontal, or vertical linear dimensions).

The dimension type (group code 70) is an integer-coded field with the following values:

Table 11-17. Group 70 integer codes for Dimension entity

Group	Description
0	Rotated, horizontal, or vertical
1	Aligned
2	Angular
3	Diameter
4	Radius
5	Angular 3-point
6	Ordinate
64	Ordinate type. This is a bit value (bit 7) used only with integer value 6. If set, ordinate is X-type; if not set, ordinate is Y-type
128	This is a bit value (bit 8) added to the other group 70 values if the dimension text has been positioned at a user-defined location rather than at the default location

In addition, all dimension types have an optional group (code 51) that indicates the horizontal direction for the Dimension entity. This determines the orientation of dimension text and dimension lines for horizontal, vertical, and rotated linear dimensions. The group value is the negative of the Entity Coordinate Systems (ECS) angle of the UCS X axis in effect when the Dimension was drawn. The X axis of the UCS in effect when the Dimension was drawn is always parallel to the XY plane for the Dimension's ECS, and the angle between the UCS X axis and the ECS X axis is a single 2D angle. The value in group 51 is the angle from horizontal (the effective X axis) to the ECS X axis. Entity Coordinate Systems (ECS) are described later in this section.

Linear dimension types with an oblique angle have an optional group (code 52). When added to the rotation angle of the linear dimension (group code 50) this gives the angle of the extension lines. The optional group code 53 is the rotation angle of the dimension text away from its default orientation (the direction of the dimension line).

For all dimension types, the following groups represent 3D WCS points:

10, 20, 30
13, 23, 33
14, 24, 34
15, 25, 35

For all dimension types, the following groups represent 3D ECS points:

11, 21, 31
12, 22, 32
16, 26, 36

Linear	(13,23,33)	The point used to specify the first extension line.
	(14,24,34)	The point used to specify the second extension line.
	(10,20,30)	The point used to specify the dimension line.

Refer to figure 11-1 on page 267 of the "AutoCAD Customization Manual"

Angular	(13,23,33) and (14,24,34)	The endpoints of the first extension line.
	(10,20,30) and (15,25,35)	The endpoints of the second extension line.
	(16,26,36)	The point used to specify the dimension line arc.

Refer to figure 11-2 on page 267 of the "AutoCAD Customization Manual"

Angular	(15,25,35)	The vertex of the angle.
(3-point)	(13,23,33)	The endpoints of the first extension line.
	(13,23,33)	The endpoints of the first extension line.
	(14,24,34)	The endpoints of the second extension line.

	(10,20,30)	The point used to specify the dimension line arc.
	Refer to figure 11-3 on page 267 of the "AutoCAD Customization Manual"	
Diameter	(15,25,35)	The point used to pick the circle/arc to dimension.
	(10,20,30)	The point on that circle directly across from the pick point.
	Refer to figure 11-4 on page 268 of the "AutoCAD Customization Manual"	
Radius	(15,25,35)	The point used to pick the circle/arc to dimension.
	(10,20,30)	The center of that circle.
	Refer to figure 11-5 on page 268 of the "AutoCAD Customization Manual"	
Ordinate	(13,23,33)	The point used to select the feature.
	(14,24,34)	The point used to locate the leader end point.
	Refer to figure 11-6 on page 268 of the "AutoCAD Customization Manual"	

Entity Coordinate Systems (ECS)

To save space in the drawing database (and in the DXF file), the points associated with each entity are expressed in terms of the entity's own Entity Coordinate System (ECS). The Entity Coordinate System allows AutoCAD to use a much more compact means of representation for entities. With ECS, the only additional information needed to describe the entity's position in 3D space is the 3D vector describing the Z axis of the ECS, and the elevation value.

For a given Z axis (or extrusion) direction, there are an infinite number of coordinate systems, defined by translating the origin in 3D space and by rotating the X and Y axes around the Z axis. However, for the same Z axis direction, there is only one Entity Coordinate System. It has the following properties:

- Its origin coincides with the WCS origin.
- The orientation of the X and Y axes within the XY plane are calculated in an arbitrary, but consistent manner. AutoCAD performs this calculation using the arbitrary axis algorithm (described later).

For some entities, the ECS is equivalent to the World Coordinate System and all points (DXF groups 10 - 37) are expressed in World coordinates. See the following table.

Table 11-18. Coordinate systems associated with an entity type

Entities	Notes
Line, Point, 3DFace, 3D Polyline, 3D Vertex, 3D Mesh, 3D Mesh vertex	These entities do not lie in a particular plane. All points are expressed in World coordinates. Of

	these entities, only Lines and Points can be extruded; their extrusion direction can differ from the World Z axis
Circle, Arc, Solid, Trace, Text, Attrib, Attdef, Shape, Insert, 2D Polyline, 2D Vertex	These entities are planar in nature. All points are expressed in Entity coordinates. All of these entities can be extruded; their extrusion direction can differ from the World Z axis
Dimension	Some of a Dimension's points are expressed in WCS, and some in ECS
Viewport	Expressed in World coordinates
Others	The remaining entities have no point data and their coordinate systems are therefore irrelevant

Once AutoCAD has established the ECS for a given entity, here's how it works:

- The elevation value stored with an entity indicates how far along the Z axis to shift the XY plane from the WCS origin to make it coincide with the plane that the entity is in. How much of this is the user-defined elevation is unimportant.

- Any 2D points describing the entity that were entered through the UCS are transformed into the corresponding 2D points in the ECS, which (more often than not) is shifted and rotated with respect to the UCS.

These are a few ramifications of this process:

- You cannot reliably find out what UCS was in effect when an entity was acquired.

- When you enter the XY coordinates of an entity in a given UCS and then do a DXFOUT, you probably won't recognize those XY coordinates in the DXF file. You'll have to know the method by which AutoCAD calculates the X and Y axes in order to work with these values.

- The elevation value stored with an entity and output in DXF files will be a sum of the Z-coordinate difference between the UCS XY plane and the ECS XY plane, and the elevation value that the user specified at the time the entity was drawn.

Arbitrary Axis Algorithm

The arbitrary axis algorithm is used by AutoCAD internally to implement the arbitrary but consistent generation of Entity Coordinate Systems for all entities except Lines, Points, 3D Faces, and 3D Polylines, which contain points in World coordinates.

Given a unit-length vector to be used as the Z axis of a coordinate system, the arbitrary axis algorithm generates a corresponding X

axis for the coordinate system. The Y axis follows by application of the right-hand rule.

The method is to examine the given Z axis (also called the normal vector) and see if it is close to the positive or negative World Z axis. If it is, cross the World Y axis with the given Z axis to arrive at the arbitrary X axis. If not, cross the World Z axis with the given Z axis to arrive at the arbitrary X axis. The boundary at which the decision is made was chosen to be both inexpensive to calculate and completely portable across machines. This is achieved by having a sort of "square" polar cap, the bounds of which is 1/64, which is precisely specifiable in 6 decimal fraction digits and in 6 binary fraction bits.

In mathematical terms, the algorithm does the following (all vectors are assumed to be in 3D space, specified in the World Coordinate System):

Let the given normal vector be called N.
Let the World Y axis be called Wy, which is always (0,1,0).
Let the World Z axis be called Wz, which is always (0,0,1).

We are looking for the arbitrary X and Y axes to go with the normal N. They'll be called Ax and Ay. N could also be called Az (the arbitrary Z axis):

If (abs (Nx) < 1/64) and (abs (Ny) < 1/64) then
            Ax = Wy N (where "" is the cross-product operator).
Otherwise,
            Ax = Wz N.

Scale Ax to unit length.

The method of getting the Ay vector would be:
Ay = N Ax. Scale Ay to unit length.

Extended Entity Data

Extended entity data is created by applications such as the Advanced Modeling Extension (AME), or by routines written with AutoLISP or ADS. Extended entity data is also produced by creating PostScript output with PSOUT. If an entity contains extended data, it follows the entity's normal definition data.

The group codes 1000 through 1071 describe extended entity data. The following is an example of an entity containing extended entity data in DXF format.

```
                ___
0                  |
INSERT             |
 8                 |
0                  |
 5                 |
5                  |
F11                |
15                 |-- Normal entity definition data.
 2                 |
BLOCK_A            |
 10                |
0.0                |
```

```
 20                |
0.0                |
 30                |
0.0             ___|
1001               |
AME_SOL            |
1002               |
{                  |
1070               |
 0                 |
1071               |
 1.95059E+06       |
1070               |
 519               |
1010               |
2.54717            |
1020               |
2.122642           |-- Extended entity data.
1030               |
2.049201           |
1005               |
ECD                |
1005               |
EE9                |
1005               |
0                  |
1040               |
0.0                |
1040               |
1.0                |
1000               |
MILD_STEEL      ___|
```

Figure 11-7. Example of extended entity data

Organization of Extended Entity Data
====================================

As you can see in the above example, group code 1001 indicates the beginning of extended entity data. This is followed by one or more 1000 group codes. Application names are string values (in the example, the application name is AME_SOL). In contrast to normal entity data, the same group code can appear multiple times, and order is important.

Extended entity data are grouped by registered application name, and each registered application's group begins with a 1001 group code with the registered application name as the string value. Registered application names correspond to APPID symbol table entries, which are essentially placeholders for registered application names.

An application can use as many APPID names as needed, although one will often suffice. APPID names are permanent, although they can be purged if they aren't currently used in the drawing.

Each APPID name can have no more than one data group attached to each entity. Within an application's group, the sequence of extended entity data groups and their meaning is defined by the application.

NOTE: PostScript images and PostScript fill requests for Polylines are stored in the AutoCAD database as extended entity data belonging

to the AUTOCAD_POSTSCRIPT_FIGURE application.

As the example in the previous figure shows, the group codes for extended entity data begin at 1000 and currently extend to 1071. The following list of extended entity data group codes are supported by AutoCAD, which maintains and manipulates their values as described:

Table 11-19. extended entity data group codes and descriptions

Entity Name	Group code	Description
String	1000	Strings in extended entity data can be up to 255 bytes long (with the 256th byte reserved for the null character)
Application name	1001 also a string value	Application names can be up to 31 bytes long (the 32d byte is reserved for the null character). Use of application names is described in more detail later in this section CAUTION: Do not add a 1001 group into your extended entity data, as AutoCAD will assume it is the beginning of a new application extended entity data group
Control string	1002	An extended data control string can be either "{"or "}": these braces enable applications to organize their data by subdividing the data into lists. The left brace begins a list, and a right brace terminates the most recent list; lists can be nested When AutoCAD reads the extended entity data for a particular application, it checks to ensure that braces are balanced correctly
Layer name	1003	Name of the layer associated with the extended entity data
Binary data	1004	Binary data is organized into variable-length chunks.The maximum length of each chunk is 127 bytes. Binary data is represented as a string of hexadecimal digits, two per binary byte, in ASCII DXF files
Database handle	1005	Handles of entities in the drawing database NOTE: When a drawing with handles and extended entity data handles is imported into another drawing using INSERT, INSERT *, XREF Bind, XBIND, or partial DXFIN, the extended entity data handles are translated in the same manner as their corresponding entity handles,

		thus maintaining their binding. This is also done in the EXPLODE Block operation, or for any other AutoCAD operation. When AUDIT detects an extended entity data handle that doesn't match the handle of an entity in the drawing file, it is considered an error. If AUDIT is fixing entities, it sets the handle to 0.
3 reals	1010, 1020, 1030	Three real values, in the order X, Y, Z. They can be used as a point or vector record. AutoCAD never alters their value
World space position	1011, 1021, 1031	Unlike a simple 3D point,the World space coordinates are moved, scaled, rotated, and mirrored along with the parent entity to which the extended data belongs. The world space position is also stretched when the STRETCH command is applied to the parent entity and this point lies within the select window
World space displacement	1012, 1022, 1032	Also a 3D point that is scaled, rotated, and mirrored along with the parent (but not moved or stretched)
World direction	1013, 1023, 1033	Also a 3D point that is rotated and mirrored along with the parent (but not moved, scaled, or stretched).
Real	1040	A real value
Distance	1041	A real value that is scaled along with the parent entity
Scale factor	1042	Also a real value that is scaled along with the parent. The difference between a distance and a scale factor is application-defined
Integer	1070	A 16-bit integer (signed or unsigned)
Long	1071	A 32-bit signed (long) integer

For more information on extended entity data and the APPID table, refer to the "AutoCAD Development System Programmer's Reference" and the "AutoLISP Programmer's Reference."

Writing DXF Interface Programs

Writing a program that communicates with AutoCAD via the DXF mechanism often appears far more difficult than it really is. The DXF file contains a seemingly overwhelming amount of information, and examining a DXF file manually may lead to the conclusion that the task is hopeless.

However, the DXF file has been designed to be easy to process by program, not manually. The format was intentionally constructed to make it easy to ignore information you don't need while easily reading the information you do need. Just remember to handle the groups in any order and ignore any group you don't care about.

As an example, the following is a Microsoft BASIC program that reads a DXF file and extracts all the Line entities from the drawing (ignoring lines that appear inside Blocks). It prints the endpoints of these lines on the screen. As an exercise you might try entering this program into your computer, running it on a DXF file from one of your drawings, then enhancing it to print the center point and radius of any circles it encounters. This program is not put forward as an example of clean programming technique nor the way a general DXF processor should be written; it is presented as an example of just how simple a DXF-reading program can be.

```
1000   REM
1010   REM Extract lines from DXF file
1020   REM
1030   G1% = 0
1040   LINE INPUT "DXF file name: "; A$
1050   OPEN "i", 1, A$ + ".dxf"
1060   REM
1070   REM Ignore until section start encountered
1080   REM
1090   GOSUB 2000
1100   IF G% <> 0 THEN 1090
1110   IF S$ <> "SECTION" THEN 1090
1120   GOSUB 2000
1130   REM
1140   REM Skip unless ENTITIES section
1150   REM
1160   IF S$ <> "ENTITIES" THEN 1090
1170   REM
1180   REM Scan until end of section, processing LINEs
1190   REM
1200   GOSUB 2000
1210   IF G% = 0 AND S$ = "ENDSEC" THEN 2200
1220   IF G% = 0 AND S$ = "LINE" THEN GOSUB 1400 : GOTO 1210
1230   GOTO 1200
1400   REM
1410   REM Accumulate LINE entity groups
1420   REM
1430   GOSUB 2000
1440   IF G% = 10 THEN X1 = X : Y1 = Y : Z1 = Z
1450   IF G% = 11 THEN X2 = X : Y2 = Y : Z2 = Z
1460   IF G% = 0 THEN PRINT "Line from (";X1;",";Y1;",";Z1;") to
       (";X2;",";Y2;",";Z2;")":RETURN
1470   GOTO 1430
2000   REM
2010   REM Read group code and following value
2020   REM For X coordinates, read Y and possibly Z also
2030   REM
2040   IF G1% < 0 THEN G% = -G1% : G1% = 0 ELSE INPUT #1, G%
```

```
2050   IF G% < 10 OR G% = 999 THEN LINE INPUT #1, S$ : RETURN
2060   IF G% >= 38 AND G% <= 49 THEN INPUT #1, V : RETURN
2080   IF G% >= 50 AND G% <= 59 THEN INPUT #1, A : RETURN
2090   IF G% >= 60 AND G% <= 69 THEN INPUT #1, P% : RETURN
2100   IF G% >= 70 AND G% <= 79 THEN INPUT #1, F% : RETURN
2110   IF G% >= 210 AND G% <= 219 THEN 2130
2115   IF G% >= 1000 THEN LINE INPUT #1, T$ : RETURN
2120   IF G% >= 20 THEN PRINT "Invalid group code";G% : STOP
2130   INPUT #1, X
2140   INPUT #1, G1%
2150   IF G1% <> (G%+10) THEN PRINT "Invalid Y coord code"; G1% :
       STOP
2160   INPUT #1, Y
2170   INPUT #1, G1%
2180   IF G1% <> (G%+20) THEN G1% = -G1% ELSE INPUT #1, Z
2190   RETURN
2200   CLOSE 1
```

Writing a program that constructs a DXF file is more difficult, because you must maintain consistency within the drawing in order for AutoCAD to find the file acceptable. AutoCAD lets you omit many items in a DXF file and still obtain a usable drawing. The entire HEADER section can be omitted if you don't need to set any header variables. Any of the tables in the TABLES section can be omitted if you don't need to make any entries, and the entire TABLES section can be dropped if nothing in it is required. If you define any linetypes in the LTYPE table, this table must appear before the LAYER table. If no Block Definitions are used in the drawing, the BLOCKS section can be omitted. If present, however, the BLOCKS section must appear before the ENTITIES section. Within the ENTITIES section, you can reference layer names even though you haven't defined them in the LAYER table. Such layers are automatically created with color 7 and the CONTINUOUS linetype. The EOF item must be present at the end-of-file.

The following Microsoft BASIC program constructs a DXF file representing a polygon with a specified number of sides, leftmost origin point, and side length. This program supplies only the ENTITIES section of the DXF file, and places all entities generated on the default layer 0. This may be taken as an example of a minimum DXF generation program. Since this program doesn't create the drawing header, the drawing limits, extents, and current view will be invalid after performing a DXFIN on the drawing generated by this program. You can do a ZOOM E to fill the screen with the drawing generated. Then adjust the limits manually.

```
1000   REM
1010   REM Polygon generator
1020   REM
1030   LINE INPUT "Drawing (DXF) file name: "; A$
1040   OPEN "o", 1, A$ + ".dxf"
1050   PRINT #1, 0
1060   PRINT #1, "SECTION"
1070   PRINT #1, 2
1080   PRINT #1, "ENTITIES"
1090   PI = ATN(1) * 4
1100   INPUT "Number of sides for polygon: "; S%
1110   INPUT "Starting point (X,Y): "; X, Y
1120   INPUT "Polygon side: "; D
1130   A1 = (2 * PI) / S%
1140   A = PI / 2
```

```
1150   FOR I% = 1 TO S%
1160   PRINT #1, 0
1170   PRINT #1, "LINE"
1180   PRINT #1, 8
1190   PRINT #1, "0"
1200   PRINT #1, 10
1210   PRINT #1, X
1220   PRINT #1, 20
1230   PRINT #1, Y
1240   PRINT #1, 30
1250   PRINT #1, 0.0
1260   NX = D * COS(A) + X
1270   NY = D * SIN(A) + Y
1280   PRINT #1, 11
1290   PRINT #1, NX
1300   PRINT #1, 21
1310   PRINT #1, NY
1320   PRINT #1, 31
1330   PRINT #1, 0.0
1340   X = NX
1350   Y = NY
1360   A = A + A1
1370   NEXT I%
1380   PRINT #1, 0
1390   PRINT #1, "ENDSEC"
1400   PRINT #1, 0
1410   PRINT #1, "EOF"
1420   CLOSE 1
```

The DXFIN command is relatively forgiving with respect to the format of data items. As long as a properly formatted item appears on the line on which the data is expected, DXFIN will accept it (of course, string items should not have leading spaces unless these are intended to be part of the string). This program takes advantage of this flexibility in input format, and does not try to generate a file appearing exactly like one generated by AutoCAD.

In the case of error loading a DXF file using DXFIN, AutoCAD reports the error with a message indicating the nature of the error and the last line processed in the DXF file before the error was detected. This may not be the line on which the error occurred, especially in the case of errors such as omission of required groups.

Binary Drawing Interchange Files

The ASCII DXF file format described in the preceding sections of this chapter is a complete representation of an AutoCAD drawing in an ASCII text form easily processed by other programs. In addition, AutoCAD can produce or read a binary form of the full DXF file, and accepts limited input in another binary file format. These binary files are described in the following sections.

Binary DXF Files

The DXFOUT command provides a Binary option that writes binary DXF files. Such a file contains all of the information present in an ASCII DXF file, but in a more compact form that takes, typically, 25% less file space and can be read and written more quickly

(typically 5 times faster) by AutoCAD. Unlike ASCII DXF files, which entail a trade-off between size and floating-point accuracy, binary DXF files preserve all of the accuracy in the drawing database. AutoCAD Release 10 was the first version to support this form of DXF file; it cannot be read by older versions.

A binary DXF file begins with a 22-byte sentinel consisting of:

```
AutoCAD Binary DXF<CR><LF><SUB><NUL>
```

Following the sentinel are (group, value) pairs as in an ASCII DXF file, but represented in binary form. The group code is a single-byte binary value, and the value that follows is one of the following:

- A two-byte integer with the least-significant byte first and the most-significant byte last.
- An eight-byte IEEE double precision floating-point number stored with the least-significant byte first and the most-significant byte last.
- An ASCII string terminated by a zero (NUL) byte.

The type of the datum following a group is determined from the group code according to the same rules used in decoding ASCII DXF files. Translation of angles to degrees, and dates to fractional Julian date representation, is performed for binary files as well as for ASCII DXF files. The comment group, 999, is not used in binary DXF files.

Extended entity data group codes are represented in Binary DXF as a single byte with the value 255, followed by a 2-byte integer value containing the actual group code, followed by the actual value.

Extended entity data long (group code 1071) values occupy 4 bytes of data. Extended entity data binary chunks (group code 1004) are represented as a single-byte, unsigned integer length, followed by the specified number of bytes of chunk data. For example, to transfer an extended entity data long group, the following values would appear, occupying 1, 2, and 4 bytes respectively:

```
255       Escape group code.
1071      True group code.
999999    Value for the 1071 group code.
```

DXFOUT writes binary DXF files with the same file type (.dxf) as for ASCII DXF files. The DXFIN command automatically recognizes a binary file (by means of its sentinel string) and loads the file. There is no need for you to identify it as a binary file.

If DXFIN encounters an error in a binary DXF file, it reports the byte address within the file where the error was detected.

Binary Drawing Interchange (DXB) Files

The DXF file formats described earlier in this chapter are complete representations of an AutoCAD drawing that can be written and read by AutoCAD and other programs. However, AutoShade and programs

executed via the external commands facility (chapter 3) often need to supply simple geometric input to AutoCAD. For these purposes, another file format even more compact than the binary DXF format is supported. This format, called DXB (for drawing interchange binary) is limited in the entities it can represent.

DXBIN Command

To load a DXB file produced by a program such as AutoShade, enter the DXBIN command:

```
Command: dxbin
```

When AutoCAD prompts you, respond with the name of the file you want to load. You don't need to include a file type; .dxb is assumed.

DXB File Format

IMPORTANT: This information is for experienced programmers and is subject to change without notice.

The format of a DXB file is as follows:

```
Header: "AutoCAD DXB 1.0" CR LF ^Z NUL   (19 bytes)
Data: Zero or more data records
Terminator: NUL                          (1 byte)
```

Each data record begins with a single byte identifying the record type, followed by data items. The data items have various forms of representation and encoding. In the descriptions following, each data item is prefixed with a letter and a hyphen. The meaning of the letter codes is as follows:

w- 16-bit integer, byte reversed in the standard 80x86 style (least- significant byte first, most-significant byte second).

f- IEEE 64-bit floating-point value stored with lsb first, msb last (as stored by an 80x87).

l- 32-bit integer with the bytes reversed 80x86 style.

n- Number which may be either a 16-bit integer or a floating-point number depending on the most recent setting of the number mode data item. The number mode defaults to 0, signifying integers. If set to 1, all n- items will be read as floating-point.

u- Item which is either a 32-bit integer or a floating-point number depending on the most recent number mode setting. If a 32-bit integer, the value is scaled by multiplying it by 65536 (2^16). If a floating-point value, no scaling is applied.

a- Item representing an angle. If number mode is integer, this is a 32-bit integer representing an angle in units of millionths of a degree (range 0 to 360,000,000). If a floating-point number, represents degrees.

In the following table, the lengths include the item-type byte and assume the number mode is set to zero (integer mode). If number mode is floating-point, add 6 bytes to the length for each n- item

present and 4 bytes for each a-, or u- item present.

Table 11-20. Byte length for item types

Item type	Code (decimal)	Data items	Length (bytes)
Line	1	n-fromx n-fromy n-tox n-toy n-fromx n-fromy n-fromz n-tox n-toy n-toz	13
Point	2	n-x n-y	5
Circle	3	n-ctrx n-ctry n-rad	7
Arc	8	n-ctrx n-ctry n-rad a-starta a-enda	19
Trace	9	n-x1 n-y1 n-x2 n-y2 n-x3 n-y3 n-x4 n-y4	17
Solid	11	n-x1 n-y1 n-x2 n-y2 n-x3 n-y3 n-x4 n-y4	17
Seqend	17	(none)	1
Polyline	19	w-closureflag	3
Vertex	20	n-x n-y	5
3Dface	22	n-x1 n-y1 n-z1 n-x2 n-y2 n-z2 n-x3 n-y3 n-z3 n-x4 n-y4 n-z4	25
Scale Factor	128	f-scalefac	9
New Layer	129	"layername" NUL	layername length + 2
Line Extension	130	n-tox n-toy	5
Trace Extension	131	n-x3 n-y3 n-x4 n-y4	9
Block Base	132	n-bx n-by	5
Bulge	133	u-2h/d	5
Width	134	n-startw n-endw	5
Number Mode	135	w-mode	3
New Color	136	w-colornum	3
3Dline Extension	137	n-tox n-toy n-toz	7

The Line Extension item extends the last line or line extension from its To point to a new To point:. The Trace Extension item similarly extends the last trace solid, or Trace Extension from its x3,y3-x4,y4 ending line to a new x3,y3--x4,y4 line.

The Scale Factor is a floating-point value by which all integer coordinates are multiplied to obtain the floating-point coordinates used by the actual entities. The initial scale factor when a file is read is 1.0. The New Layer item creates a layer if none exists, giving the new layer the same defaults as the LAYER New command, and sets that layer as the current layer for subsequent entities. At the end of the DXB file load, the layer in effect before the command is restored.

The Block Base item specifies the base (origin) point of a created Block. The Block base must be defined before the first entity record is encountered. If DXB is not defining a Block, this specification will be ignored.

A Polyline consists of straight segments of fixed width connecting the vertices, except as overridden by the Bulge and Width items described below. The closure flag should be 0 or 1; if it is 1, then there is an implicit segment from the last vertex (immediately before the Seqend) to the first vertex.

A Bulge item, encountered between two Vertex items (or after the last Vertex of a closed Polyline), indicates that the two vertices are connected by an arc rather than a straight segment. If the line segment connecting the vertices would have length d, and the perpendicular distance from the midpoint of that segment to the arc is h, then the magnitude of the Bulge is (2 * h / d). The sign is negative if the arc from the first vertex to the second is clockwise. A semicircle thus has a bulge of 1 (or -1). If the number mode is 0 (integer), Bulge items are scaled by 2 16. If the number mode has been set to floating-point, then the floating-point value supplied is just 2*h/d (not scaled).

The Width item indicates the starting and ending widths of the segment (straight or curved) connecting two vertices. This width stays in effect until the next width item or the Seqend. If there is a Width item between the Polyline item and the first Vertex, it is stored as a default width for the Polyline; this saves considerable database space if the Polyline has several segments of this width.

The Number Mode item controls the mode of items with types given in the table above as n-, a-, or u-. If the value supplied is zero, these values will be integers, otherwise floating-point. The storage and implicit scaling conventions for these values in both modes are described earlier.

Lines share the same cells to remember the last to-point, so you shouldn't mix extension groups for the two entities without an initial group before the extension. There is no extension group for 3Dfaces, as there's no obvious edge to extend from.

The New Color group specifies the color for subsequent entities in the DXB file. The w-colornum word argument is in the range from 0 to 256. 0 means color by block, 1-255 are the standard AutoCAD colors, and 256 means color by layer. A color outside the range from 0 to 256 sets the color back to the current entity color (you can do this deliberately, and it can be quite handy). The initial entity color

of material added by DXBIN is the current entity color.

All points specified in the DXB file are interpreted in terms of the current UCS at the time the DXBIN command is executed.

Writing DXB Files

There is no direct AutoCAD command to write a DXB file, but the special ADI plotter driver can write such a file. If you want to create a DXB file from an AutoCAD drawing, configure the ADI plotter and select its DXB file output option.

Initial Graphics Exchange Specification (IGES) Files

Using the commands described in this section, you can instruct AutoCAD to read and write IGES-format interchange files.

NOTE: The format of IGES files and the mapping performed to translate between AutoCAD drawing information and IGES are described in the separate AutoCAD/IGES Interface Specifications document.

IGESOUT Command

You can generate an Initial Graphics Exchange Specification (IGES) interchange file from an existing AutoCAD drawing by means of the IGESOUT command:

```
Command: igesout
```

When AutoCAD prompts you, respond with a filename or press 5 to accept the default.

The default name for the output file is the same as that of the current drawing, but with a file type of .igs. If you specify an explicit filename without including a file type, .igs is assumed. If a file with the same name already exists, it is deleted. If FILEDIA is on, and a file with the same name already exists, AutoCAD tells you; allowing you to OK or cancel the deletion.

IGESIN Command

An IGES interchange file can be converted into an AutoCAD drawing by means of the IGESIN command:

```
Command: igesin
```

When AutoCAD prompts you, respond with the name of the IGES file to be loaded.

To load a complete IGES file, you must use IGESIN in an empty drawing, before any entities have been drawn and before any additional Block definitions, layers, linetypes, text styles, named views, named coordinate systems, or named viewport configurations have been created.

NOTE: If the drawing you are using as a prototype is not empty, you might find it helpful to open a new drawing using the No Prototype... button of the Create New Drawing dialogue box, as

described in chapter 4 of the AutoCAD Reference Manual. You should also be aware that some third-party applications include an acad.lsp or .mnl file that modifies your drawing upon startup.

If a serious error is encountered, the input process stops and an error message is displayed reporting where the error was found. The partial drawing is not discarded.

REFERÊNCIAS BIBLIOGRÁFICAS

ARRO99 E ARROYO, J L Los Arcos. SRV: A Virtual Reality Application to Electrical Substations Operation Training. IEEE, 1999.

ANDE93 M D ANDERSON, H J Pottinger, C M Electrical Substations Operation Training. IEEE, 1999.

ADAMS, L. Visualização e realidade virtual, Ed. Makron Books, pp. 255-259, São Paulo, 1994.

ANDRADE, Gustavo de Oliveira. Integração de sólidos e superfícies. Revista CADesign, São Paulo: Market Press, ano 10, n. 100, s./d., p. 50-51.

ARAÚJO, R. B. Especificação e análise de um sistema distribuído de realidade virtual, São Paulo, Junho, 144 Pp., Tese (Doutorado), Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da Universidade de São Paulo, 1996.

AUTOCAD 10.0, ENG.º Alexandre L. C. Censi

B. MUKHERJEE, Optical Communication Networks, McGraw-Hill, 1997.

BARBERATO, Edson. A criação de novos produtos e a tecnologia CAD/CAM/CAE. Revista CADesign, São Paulo: Market Press, ano 7, n. 68 s./d., p.66.

BARN (1993/2003) em http://lsi.ugr.es/~jctorres/cad/teoria/trTema1.pdf. Acessado em 16 mar. 2006

BEGAULT, D. R. 3-D Sound for virtual reality and multimedia, Academic Press, Cambridge, MA, 1994.

BIB - http://www.bibvirt.futuro.usp.br/. Acessado em 10 mar.2006

BREE95 P T BREEN, W G Scott. Virtual Reality Applications in T&D

BRUNET, P.: Diseño gráfico y modelado geométrico. MOMPÍN, J. (Ed.): Sistemas CAD/CAM/CAE. Diseño y fabricación por ordenador.. Barcelona: Marcombo, 1986.

BRUNETTI, G. Virtual reality techniques supporting the product and process development, 5º Seminário Internacional de Alta Tecnologia, UNIMEP, Santa Bárbara d´oeste, pp. 83-98, Outubro, 2000.

BURDEA, G. & COIFFET, P. Virtual reality technology, John Wiley & Sons, New York, N.Y, 1994.

CAMPBELL, Patrick T.. Instalando redes em pequenas e médias empresas : resolvendo os problemas de rede em pequenos e médios ambientes. São Paulo: Makron Books, 1997.

CEREDA, Ronaldo Luiz Dias; et al. ATM o futuro das redes. São Paulo: Makron Books, 1997.

COMEAU, C. P. & BRYAN, J. S. Headsight television system provides remote surveillance, Electronics, pp. 86-90, 1961.

ELLIS, S. R. What are virtual environments?, IEEE Computer Graphics and Application, pp. 17-22, 1994

FISHER, S. S. & TAZELAAR, J. M. Living in a virtual world, Byte, pp. 215-221, 1990.

GRADECKI, J. Kit de montagem da realidade virtual. São Paulo, Berkeley, 1994.

GRADECKI, J. The virtual reality construction kit, John Wiley & Sons, 340 Pp., 1995.

IEC Online Education. “Optical Access.” http://www.iec.org/online/tutorials/opt_acc/

JACOBSON, L. Realidade virtual em casa. Rio de Janeiro, Berkeley, 1994.

JACOBSON, L. Virtual reality: A status report, AI Expert, pp. 26-33, 1991.

KALAWSKY, R. S. The science of virtual reality and virtual environments, Ed. Addison-Wesley, 405 Pp., 1993.

KIRNER, C. Apostila do ciclo de palestras de realidade virtual, Atividade do Projeto AVVIC-CNPq (Protem - CC - fase III) - DC/UFSCar, São Carlos, pp. 1-10, 1996.

KRUEGER, M. W. Artificial reality II, Addison-Wesley, Reading, MA, USA, 1991.

LATERZA, Luiz Bandeira de Mello. Do desenho técnico ao modelamento de sólidos. Revista CADesign, São Paulo: Market Press, ano_1994 , n.1, s./d., p.18-21, 1994

LESTON, J. Virtual reality: the it perspective, Computer Bulletin, pp. 12-13, 1996.

MACHADO, L.S. (2003) *A Realidade Virtual no Modelamento e Simulação de Procedimentos Invasivos em Oncologia Pediátrica*. Tese de Doutorado. Março de 2003. Escola Politécnica da Universidade de São Paulo. Disponível em: www.teses.usp.br/teses/disponiveis/3/3142/tde-07052003-123257/

MACHOVER, C. Four decades of computer graphics, IEEE Computer Graphics and Application, pp. 14-19, 1994.

MACLEOD, D. Computer: Virtual Reality. Progressive Architecture April 1992; Pento Pub., Ohio, pp. 55-56.

MOTTA, Gustavo Dias. O mundo é tridimensional. **Revista CADesign**, São Paulo: Market Press, ano 7, n. 78, s./d., p. 66.

PIMENTEL, K. & TEIXEIRA, K. Virtual reality - through the new looking glass. 2.ed. New York, McGraw-Hill, 1995.

SOARES NETO, Vicente; BOSCATO C. JUNIOR, Mário; SILVA, Adelson de Paula. Telecomunicações – Redes de Alta Velocidade – Cabeamento Estruturado. São Paulo: Érica, 2001

SOLINHO, José Luiz Guedes. Novas perspectivas para aplicações de CAD. Revista CADesign, São Paulo: Market Press, ano 8, n. 8, s./d., p. 66.

STUTZ, Jeferson. 3D na engenharia. Revista CADesign, São Paulo: Market Press, ano 8 n. 89, s./d., p. 66.

VALERIO NETTO, A. Prototipação em ambientes virtuais para o desenvolvimento de máquinas-ferramenta, Revista PESQUISA & TECNOLOGIA - Publicação da FEI, n. 19, pp. 19- 23, 2000.

VENDITTI, M.V.R. Desenho técnico sem prancheta com AutoCAD 2002. Florianópolis: Visual Books, 2003. p.207.

VINCE, J. Virtual Reality Systems, Addison-Wesley, Reading, MA, USA, 1995.

VOISINET, Donald D. CAD – projeto e desenho auxiliados por computador. Introdução – conceitos – aplicações. São Paulo: McGraw-Hill, 1998.

W. GORALSKI, Optical Networking and WDM, Osborne/McGraw-Hill, Berkeley, CA, 2001.

YOSHIDA, Waldomiro. O CAD nosso de cada dia. Revista CADesign, São Paulo: Market Press, ano 8, n. 82, s./d., p. 66.

RHEINGOLD, H. Virtual reality. New York, Touchstone, 1991.

(DXF_2006), www.dpi.inpe.br/spring/usuario/dxf_format.htm acessado em setembro de 2006.

(3D/DXF_2006), http://www.dcs.ed.ac.uk/home/mxr/gfx/3d/DXF12.spec acessado em setembro de 2006.

BR&u=http://www.bearcave.com/dxf/dxfintro.htm&prev=/search%3Fq%3Dpaint%252BCA

http://www.dc.ufscar.br/~grv/vrml/tutoriais/vrml10/

http://broadwin.com/Manual/EngMAN/10.11.12_Open_DXF.htm

(ufscar_2006), http://www.dc.ufscar.br/~grv/tutrv/tutrv.htm acessado em outubro de 2006.

(CAD_2006), http://www.mundocad.com.br/publicacoes/historia.php acessado em junho de 2006.

(CAD_2006), http://pt.wikipedia.org/wiki/AutoCAD acessado em novembro 2006.

http://pt.wikipedia.org/wiki/3D

(pucrs_2006) ,http://www.inf.pucrs.br/~pinho/CG/Aulas/Intro/intro.htm acessado em novembro 2006

http://www.tqs.com.br/servicos/duvidas.htm

(VModel_2006), http://www.winosi.onlinehome.de/VModel.htm acessado em agosto de 2006.

ftp.autodesk.com/prodsupp/downloads/acad_dxf.pdf.
(3D_2006), http://www.amazon.com.br/~jbassalo/artigos.htm acessado em outubro de 2006.

http://www.dc.ufscar.br/~grv/vrml/tutoriais/vrml10.

www.ocnus.com/models/models.html.

BASTIANIK, Silene Mirella; DIAS, Marcelo Hugo Romeu e
Ferramenta para Visualização e Conversão de arquivos
VRML e DXF orientador:José Remo Ferreira Brega.
Marília, SP: [s.n], 2006.

Apresentação de Monografia (Graduação em Ciência da Computação) – Centro Universitário Eurípides de Marília – Fundação de Ensino Eurípides Soares da Rocha.

1. Realidade Virtual. 2. Engenharia. 3. AutoCad

CDD: 006