FÁBIO KAWAOKA TAKASE

DESENVOLVIMENTO DE UM SISTEMA DE MANIPULAÇÃO DE INFORMAÇÕES DE DEPENDÊNCIAS PARA APOIO A SISTEMAS CAD

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

São Paulo
1997

FÁBIO KAWAOKA TAKASE

DESENVOLVIMENTO DE UM SISTEMA DE MANIPULAÇÃO DE INFORMAÇÕES DE DEPENDÊNCIAS PARA APOIO A SISTEMAS CAD

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

Área de Concentração:
Engenharia Mecânica

Orientador:
Marcos de Sales Guerra Tsuzuki

São Paulo
1997

Aos meus pais e meus irmãos

Agradecimentos

Agradeço ao Prof. Dr. Marcos de Sales Guerra Tsuzuki pelo orientação na execução deste trabalho, pelo incentivo pessoal em momentos de desânimo e pela disposição e paciência dedicadas em momentos de dificuldades.

Agradeço aos colegas e amigos da Mecatrônica pelas discussões, críticas e sugestões que tanto ajudaram para a conclusão deste trabalho. Um agradecimento especial para Tomaz M. Sasaki, Critstina T. M. Matsusaki, Myrna Y. Kagohara, Song San Woei e Roberto Y. I. Ueda pela amizade demonstrada durante todo o período da realização do trabalho.

Agradeço ao Prof. Dr. José Reinaldo Silva pelas frutíferas sugestões para melhoria da qualidade deste trabalho e ao Prof. Dr. Paulo Eigi Miyagi pelo incentivo em participar do programa de mestrado da EPUSP.

Agradeço aos meus pais que sempre me incentivaram e apoiaram em qualquer atividade por mim realizada.

ÍNDICE

ÍNDICE ..... **IV**

LISTA DE FIGURAS ..... **VII**

LISTA DE TABELAS ..... **X**

RESUMO ..... **XI**

ABSTRACT ..... **XII**

1. INTRODUÇÃO **1**

1.1 OBJETIVO DO TRABALHO 1

1.2 MOTIVAÇÃO 1

1.2.1 Evolução dos Sistemas CAD *2*

1.2.2 Modelagem do Produto e Sistemas CAD Paramétrico e Variacional *10*

1.2.3 Importância da História de Execução de Operações e a Dependência entre elas. *11*

1.3 ORGANIZAÇÃO DO TRABALHO 13

2. CONCEITOS BÁSICOS **15**

2.1 MODELAGEM DE SÓLIDOS 15

2.1.1 Modelos *15*

2.1.2 O Modelo Sólido e suas Representações *16*

2.2 DIMENSÕES RELATIVAS 35

2.3 SISTEMAS DE MANUTENÇÃO DA VERDADE (TMS - TRUTH MAINTENANCE SYSTEMS) 37

2.3.1 O Sistema de Manutenção da Verdade baseado em Suposições (ATMS - Assumption-based Truth Maintenance System) *38*

3. REVISÃO BIBLIOGRÁFICA **42**

3.1 ABORDAGENS PRIMÁRIAS 42

3.2 ABORDAGENS ALGÉBRICAS 42

3.3 ABORDAGENS BASEADAS EM TÉCNICAS DE IA 44

4. METODOLOGIA E IMPLEMENTAÇÃO **47**

4.1 METODOLOGIA ADOTADA 47

4.2 FUNCIONALIDADES DO SISTEMA 48

4.2.1 UNDO *48*

4.2.2 REDO *50*

4.3 ARQUITETURA DO SISTEMA 50

4.3.1 O USPDesigner *51*

5.3.2 O Sistema de Apoio *53*

4.4 IMPLEMENTAÇÃO DO PROTÓTIPO 59

4.4.1 Implementação de Modificações no USPDesigner *59*

4.4.2 Implementação da Estrutura para Armazenamento da História de Execução *62*

4.4.3 Implementação da estrutura ATMS *62*

5. RESULTADOS EXPERIMENTAIS **63**

5.1 CASO SIMULADO A 64

5.2 CASO SIMULADO B 67

6. DISCUSSÃO **72**

6.1 CASO A 72

6.1.1 UNDO da Operação 5 ..... *72*

6.1.2. REDO da Operação 5 ..... *73*

6.2 CASO B 73

6.2.1 UNDO da Operação 5 ..... *73*

6.2.2 REDO da Operação 5 - Aumento do diâmetro do furo *74*

6.2.3 REDO da Operação 5 - Mudança para um furo quadrado *75*

7. CONCLUSÕES **76**

ANEXO A - SISTEMAS CAD PARAMÉTRICO E VARIACIONAL **78**

A.1 PROJETO PARAMÉTRICO 79

A.2 PROJETO VARIACIONAL 80

A.3 COMPARAÇÃO ENTRE OS PROJETOS PARAMÉTRICO E VARIACIONAL 81

REFERÊNCIAS BIBLIOGRÁFICAS **85**

LISTA DE FIGURAS

Figura 1.1 - As quatro gerações de sistemas CAD. 2

Figura 1.2 - Modelo de Produto. 6

Figura 1.3 - História e dependências entre operações. 12

Figura 2.1 - Diferença entre geometria e topologia de um objeto. (ZEID, 1991) 17

Figura 2.2 - Um modelo que pode ser ambíguo segundo a representação por superfície 18

Figura 2.3 - Classificação dos esquemas de representação. 19

Figura 2.4 - Parâmetros da equação do semi-plano cônico (a) e toroidal (b). 25

Figura 2.5 - Relações de Adjacência (WEILER, 1985). 28

Figura 2.6 - Operadores de Euler. 32

Figura 2.7 - Seqüência de criação de um cubo com furo passante (MÄNTYLÃ, 1988). 34

Figura 2.8 - Exemplo de criação de um sólido inválido auto-interceptante através da execução de uma operação local de extrusão. 35

Figura 2.9 - Comportamento de uma dimensão em um sistema paramétrico. 36

Figura 2.10 - Exemplo de uma rede ATMS. 40

Figura 4.1 - UNDO: convencional e consistente (INUI e KIMURA, 1993). 49

Figura 4.2 - Exemplo de uma seqüência de operações de modelagem 55

Figura 4.3 - Estrutura ATMS para o exemplo da figura 4.2. 58

Figura 4.4 - Exemplo de problemas na execução de UNDO em uma interface baseada na representação CSG. 60

Figura 4.5 - Estrutura em árvore da CSG utilizando um sólido base. 61

Figura 5.1 - Primeiro Caso simulado. 63

Figura 5.2 - Segundo caso simulado. 64

Figura 5.3 - Sólido gerado pelas operações 1, 2, 3 e 4 da Tabela 5.1 65

Figura 5.4 - Sólido obtido após a execução da operação 5 (criação do bloco de dimensões 5x5x2). . 65

Figura 5.5 - Sólido obtido após a execução das operações 6 e 7 (translação do cubo gerado pela operação 5 e operação booleana de diferença, gerando um furo passante). 66

Figura 5.6 - Sólido obtido após a execução de todas as operações de modelagem 66

Figura 5.7 - UNDO da operação 5 67

Figura 5.8 - REDO da operação 5. 67

Figura 5.9 - Sólido gerado pelas operações 1, 2, 3 e 4 da Tabela 5.2 68

Figura 5.10 - Sólido obtido após a execução da operação 5 (criação do cilindro). 69

Figura 5.11 - Sólido obtido após a execução das operações 6 e 7 (translação do cilindro gerado pela operação 5 e operação booleana de diferença, gerando um furo passante). 69

Figura 5.12 - Sólido obtido após a execução das operações 8 e 9 (criação de um bloco e sua translação para a posição adequada). 69

Figura 5.13 - Sólido obtido após a execução da operação 10 (criação de um bloco). 70

Figura 5.14 - Sólido obtido após a execução da operação 11 (translado do bloco). **70**

Figura 5.15 - Sólido obtido após a execução da operação 12 (operação booleana de subtração). **70**

Figura 5.16 - UNDO da operação 5. ..... **71**

Figura 5.17 - REDO da operação 5 (obtenção de um furo de maior diâmetro). **71**

Figura 5.18 - REDO da operação 5 (furo prismático quadrado). **71**

Figura 6.1 - UNDO da operação 5, caso A. **72**

Figura 6.2 - REDO da operação 5, caso A. **73**

Figura 6.3 - UNDO da operação 5, caso B. **74**

Figura 6.4 - REDO da operação 5, aumento do diâmetro do furo. **75**

Figura 6.5 - REDO da operação 5, furo prismático quadrado. **75**

Figura A.1 - Restrições acopladas (b) e não acopladas (a) **80**

Figura A.2 - Exemplo do triângulo de Chung e Schussel. (SINGH, 1996) **81**

LISTA DE TABELAS

Tabela 2.1 - Os quinze tipos de dimensões relativas (TSUZUKI, 1994)..... 37

Tabela 5.1 - Seqüência de operações para criação do sólido da Figura 5.1 64

Tabela 5.2 - Seqüência de operações para criação do sólido da Figura 5.2 68

RESUMO

Neste trabalho, a necessidade dos sistemas CAD de disponibilizar ao usuário meios de manipular de maneira mais flexível os modelos sólidos criados é enfatizada, mostrando a evolução histórica destes sistemas e a necessidade crescente de modelos capazes de representar um volume maior de informações e de manipulá-las consistentemente. Dentro deste contexto, as metodologias paramétrica e variacional se apresentam como uma boa opção para a manipulação consistente e automática de mudanças impostas pelo projetista, suportando desta forma a natureza interativa do processo de desenvolvimento de um produto. Diversas propostas nesta área são apresentadas e classificadas segundo a forma com que abordam o problema. A importância da seqüência de operações adotada pelo projetista ao definir o modelo computacional é ressaltada e duas funções, UNDO e REDO, foram identificadas como necessárias para uma manipulação satisfatória de modelos sólidos em sistemas CAD baseados na história de execução de operações. Para um suporte consistente para estas funções foi verificada a necessidade de informações de dependências entre as operações executadas. O sistema ATMS (*Assumption-based Truth Maintenance System*) é sugerido como uma estrutura conveniente para armazenar as informações de dependências necessárias. Um protótipo de um Sistema de Apoio que manipula estas informações foi desenvolvido e integrado ao modelador de sólido do Departamento de Engenharia Mecânica da EPUSP, o USPDesigner.

ABSTRACT

In this work, the need of CAD systems that supports the user in the task of manipulating solid models in a flexible manner is demonstrated. The historical evolution of these systems and the increasing need of models which are able to represent more product data and manipulate them with consistency are presented. In this context, the parametric and variational methods has shown to be good choices to manipulate the changes proposed by the designer in a consistent and automatic manner. The use of these methods leads to a good support to the product design process, which is a highly iterative task. Several proposals in this research area are presented and classified based on the way they approach the problem. The importance of building sequences of the designer's modeling operations is shown. Two tasks, UNDO and REDO were identified as being necessary to provide good suport to manipulate the solid models in CAD systems based on the history of modeling operation execution. To provide a consistent support for these tasks, the need of dependency data between the executed operations was verified. The ATMS (Assumption-based Truth Maintenance System) is used as a convenient structure to store and manipulate the dependency data needed. A prototype of a Support System that manipulates these information has been implemented and integrated to the solid modeling system of Mechanical Engineering Department of Escola Politécnica da Universidade de São Paulo, the USPDesigner.

1. INTRODUÇÃO

1.1 Objetivo do Trabalho

O presente trabalho tem por objetivos o estudo de técnicas de modelagem paramétrica e o desenvolvimento de um sistema de auxílio ao projeto de produtos baseado na história de criação de um modelo sólido, utilizando uma estrutura para armazenamento e manipulação de informações de dependências existentes entre as operações realizadas sobre o modelo.

1.2 Motivação

A tecnologia CAD/CAM (*Computer Aided Design/Computer Aided Manufacturing*) é uma tecnologia indispensável nas indústrias mecânicas e muita pesquisa tem sido realizada nesta área (INUI e KIMURA, 1990). Os sistemas CAD evoluíram da simples "informatização" da atividade de desenho para sistemas que suportam a natureza interativa da atividade de desenvolvimento de novos produtos e que permitem uma integração com outros sistemas computacionais de apoio ao projeto e fabricação.

A observação da evolução histórica dos sistemas CAD permite a verificação das diferentes ênfases dadas às diversas dificuldades encontradas no desenvolvimento de sistemas deste tipo.

1.2.1 Evolução dos Sistemas CAD

Do ponto de vista da representação computacional das informações de um produto, os sistemas CAD podem ser historicamente agrupados em quatro gerações (SUZUKI, 1994; CUGINI, 1989). Na Figura 1.1 podemos observar as quatro gerações de sistemas CAD.

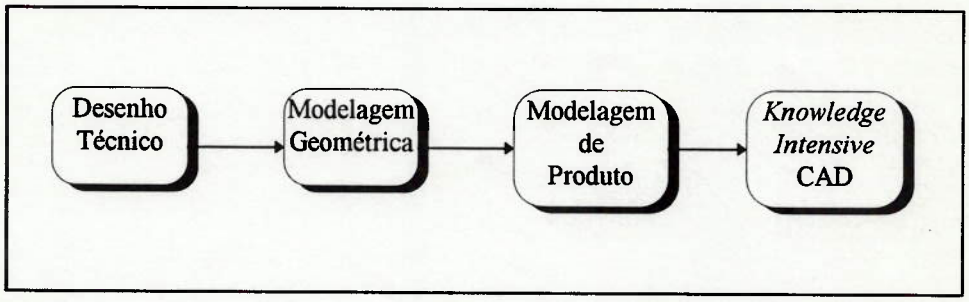

Figura 1.1 - As quatro gerações de sistemas CAD.

1.2.1.1 1ª Geração: Desenho Técnico (*Computer Aided Drafting*)

Esta primeira geração de sistemas CAD, largamente difundida nas indústrias nos anos 70, caracteriza-se por informatizar o processo de documentação do produto. Como uma primeira abordagem, estes sistemas procuraram informatizar aspectos visíveis da concepção e transferência de informações de projeto de um produto, ou seja, informatizaram o desenho.

O problema abordado apresenta-se de forma bem comportada, pois o desenho técnico é uma linguagem artificial bem formalizada e padronizada. Estes padrões bem definidos de representação geométrica foram utilizados para informatizar a produção de desenhos.

Entretanto, o resultado obtido consiste ainda em um conjunto de linhas, pontos e entidades geométricas que precisam ser interpretados para ter algum significado. Para realizar esta interpretação do desenho, é necessário um conhecimento prévio dos padrões e símbolos utilizados.

Dentre os principais motivos que levaram estes sistemas a informatizar o processo de geração de desenhos a fim de se obter uma maior eficiência temos (MEDLAND, 1992):

- A fase de desenho é a fase mais visível e quantitativa do processo de desenvolvimento de um produto, além de envolver muitas pessoas, tornando sua informatização economicamente atrativa.
- Os procedimentos utilizados são bem definidos, sendo por isto facilmente isolados e resolvidos do ponto de vista de desenvolvimento de sistemas de apoio.
- Os desenhos têm uma grande importância no campo industrial, pois eles são os resultados quantificados da fase de concepção do produto e são referência para as fases subseqüentes, como as fases de manufatura, montagem, etc..., além de serem documentos que contém e transferem informações relativas a peças, máquinas e disposição física (*lay-out*), entre todos aqueles que estão envolvidos no processo de produção do produto.

Uma referência para esta geração é o sistema SKETCHPAD desenvolvido por SUTHERLAND (1963). Em seu trabalho, SUTHERLAND (1963) inovou ao realizar a integração de computadores e dispositivos gráficos para tratar objetos bi e tri-dimensionais.

1.2.1.2 2ª Geração: Modelagem Geométrica (*Geometric Modeling*)

Esta geração de sistemas CAD, que atualmente é largamente utilizada na indústria, tem por característica principal a criação de modelos sólidos computacionais capazes de classificar qualquer ponto do espaço tridimensional como sendo internos, externos e sobre o contorno do sólido. A esta propriedade é dado o nome de endereçamento espacial (*space addressability*) (ZEID, 1991). Isto é possível de ser realizado por que o modelo possui informações sobre a topologia do sólido, isto é, linhas, pontos e superfícies "soltas" não possuem sentido; estes entes geométricos devem fazer parte de unidades com volume, isto é, são partes de um sólido. Duas formas de modelagem se destacam nesta geração, a modelagem por superfícies, onde a ênfase é dada sobre a geometria das superfícies que definem o contorno do sólido, e a modelagem de sólidos, onde a ênfase está nas informações sobre a topologia do sólido.

Abordando de uma maneira mais intuitiva, estes sistemas inovaram ao permitirem a criação de protótipos computacionais, isto é, uma representação computacional capaz de descrever um objeto real que pode fornecer informações que anteriormente eram obtidas pela construção de protótipos. O modelo foi separado do documento, uma vez que a representação computacional se tornou capaz de armazenar um volume maior de informações associadas ao produto que um desenho técnico. A principal implicação desta separação foi que a importância do documento foi alterada, pois informações sobre

o produto puderam a partir de então ser obtidas por consultas diretas ao modelo computacional sem a necessidade de interpretação e reconhecimento das informações por parte do usuário.

1.2.1.3 3ª Geração: Modelagem de Produto (*Product Modeling*)

Esta geração de sistemas CAD muda o enfoque das gerações anteriores. Para uma melhor compreensão do que é Modelagem de Produto é importante conhecermos o contexto que exigiu a sua criação.

O acirramento e endurecimento da competição no mercado levou as indústrias a procurarem cada vez mais a diminuição de tempo de produção, a melhora da qualidade, a redução de custos, etc... e uma maneira de atingir estas metas foi melhorar a eficiência empresarial nas fases de concepção e produção.

Neste contexto, uma melhora na utilização do conhecimento e da informação tornou-se urgente e foram iniciados estudos de sistemas capazes de contornar a falta de habilidade de manipular as diversas informações de engenharia associadas ao processo de desenvolvimento de novos produtos.

O modelo computacional do produto precisava ser capaz de descrever além das informações geométricas, as propriedades físicas e informações de engenharia, tais como propriedades de materiais, dimensões, tolerâncias e representações de montagem.

Basicamente, o que se procurou foi uma integração pela criação de um sistema com uma base de dados única com informações representativas de um produto, tanto nas atividades de concepção como nas de fabricação. Como o computador não é capaz de entender desenhos técnicos, foi proposto um modelo computacional capaz de representar todo o conhecimento presente nos desenhos técnicos (dados explícitos como geometria e

dimensões e dados implícitos que somente um usuário treinado na linguagem padronizada do desenho é capaz de entender). Para este modelo atribuiu-se o nome Modelo de Produto (*Product Model*) (KIMURA *et al.* 1987).

O processo de *design* passa a ser visto como consistindo de diversos processos de modelagem de um produto, e os sistemas CAD passam a ter a função de auxiliar em todos as fases ou processos do processo de *design*. WINGÅRD (1991) observou que projetistas que trabalham sobre um mesmo produto possuem interpretações diferentes com relação ao produto nas diferentes fases do processo de *design*. Ele ressalta a necessidade de diferentes modelos para cada fase de design. Estes modelos entretanto não devem ser totalmente independentes, precisando existir alguma forma de mapear as relações existentes entre cada um deles, pois uma modificação realizada em um destes modelos deve provocar alterações nos outros. Em outras palavras, um modelo de produto (*Product Model*) consiste de vários tipos de modelos, cada um com um propósito especial (SUZUKI *et al.* 1984), como mostra a Figura 1.2.

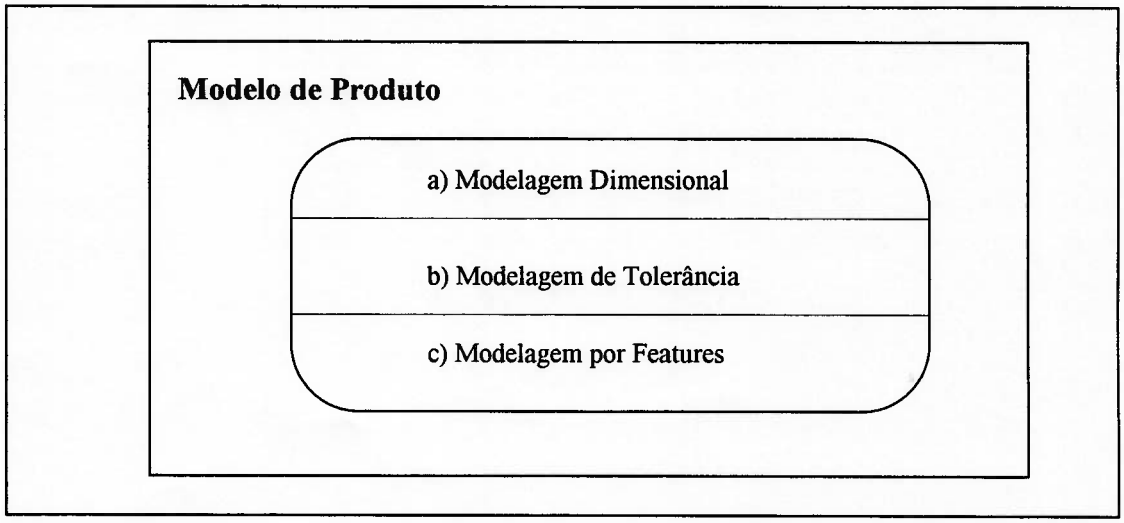

Figura 1.2 - Modelo de Produto.

a) Modelagem de Dimensões (*Dimension Modeling*)

As dimensões definem tamanhos e posições de partes de um produto em um desenho. Mesmo representando-se um produto com grande precisão através de modelos geométricos ainda é necessário um esquema dimensional, pois um mesmo produto pode possuir vários esquemas de dimensionamento. A utilização de um esquema de dimensionamento em detrimento de outros, significa que este grupo de dimensões representa de forma mais apropriada as intenções do projetista.

Outra faceta importante abordada pelo modelo dimensional é a modificação da forma do produto pela alteração de parâmetros. Esta característica é chamada de Projeto Dirigido por Dimensões (*Dimension-driven design*) e as metodologias que suportam esta característica são designadas por Projeto Paramétrico (*Parametric Design*) e Projeto Variacional (*Variational Design*).

Um conceito importante neste modelo é o de restrições geométricas (*geometric constraints*), pois estas restrições definem a geometria da forma do objeto.

Os diversos métodos propostos para lidar com as dimensões podem ser classificados segundo a forma como as dimensões são transformadas em restrições geométricas e como a solução para estas restrições geométricas são encontradas.

b) Modelagem de Tolerâncias (*Tolerance Modeling*)

Durante o processo de desenvolvimento de um produto, é de elevada importância a consideração das relações entre as variações de forma de cada peça e seus

efeitos na funcionalidade do produto, e na possível utilização destas peças em outros produtos (intercambialidade) (SUZUKI *et al.* 1995).

SUZUKI *et al.* (1995) classifica os sistemas CAT (*Computer Aided Tolerancing*) em dois grupos, segundo as diferentes formas de abordarem o problema. O primeiro grupo, chamado de sistemas baseados em conhecimento por utilizarem conhecimentos acumulados por especialistas, tais como: qual a precisão que se consegue obter com um determinado processo de usinagem, seus custos, durabilidade da peça, etc... encontram muitas aplicações, entretanto não oferecem muito suporte quando se deseja realizar projetos inovativos.

O segundo grupo, baseado em simulação, mostra-se mais genérico, ao avaliar o comportamento fisico de produtos com erros geométricos. Desta forma, as alterações de comportamento devido aos erros geométricos podem ser analisadas, como por exemplo, a conseqüência do acúmulo de erros geométricos na montagem de um produto pode ser avaliada por simulações computacionais da própria montagem utilizando o produto com diversas variações dimensionais.

Verifica-se que uma importante tarefa do projetista é conseguir o máximo em funcionalidade para um dado produto respeitando as restrições impostas pelos recursos e processos de manufatura disponíveis.

c) Modelagem por Features (*Form Feature Modeling*)

A modelagem por features tem por objetivo permitir que o modelo seja definido por meio de vocabulários diferentes, isto é, de diferentes pontos de vista segundo pessoas das diversas áreas que participam do seu desenvolvimento. Imagina-se que esta modelagem permitirá uma eficiente integração entre sistemas CAD e

CAM, duas áreas intrinsicamente integradas no processo de *design* e que no entanto possuem vocabulários muito diferentes.

As features são elementos cujas características são definidas a partir dos diversos pontos de vista de projeto (TSUZUKI, 1994).

Os sistemas baseados em features podem ser divididos em dois tipos básicos, aqueles que utilizam features durante o projeto do produto (*Design-by-features*) e aqueles que a partir de um modelo computacional reconhecem features previamente definidas, adicionando ao modelo características de manufatura associadas à feature reconhecida.

1.2.1.4 4ª Geração: *Knowledge Intensive CAD* / Sistemas Integrados

A importância de informações gerais, tais como dados de manufatura, montagem, custos, que suportam uma decisão na escolha de materiais, formas, dimensões e tolerâncias passa a ser enfocada nesta geração de sistemas CAD.

As informações tratadas nesta geração de sistemas CAD caracterizam-se por definir a experiência de um projetista. Esta geração de sistemas CAD têm como fatores alavancadores o suporte a produtos de alta qualidade, o suporte ao ciclo de vida completo do produto, compartilhamento e reutilização de conhecimento relacionado ao processo de desenvolvimento e por fim, suporte à engenharia simultânea (SUZUKI, 1994) .

1.2.2 Modelagem do Produto e Sistemas CAD Paramétrico e Variacional

Uma das dificuldades existentes no processo de *design* é em como criar um modelo computacional capaz de mapear as idéias do projetista, suas intenções e permitir uma fácil e rápida criação e manutenção do modelo, uma vez que o projetista, principalmente nas fases preliminares do projeto, ainda não possui uma idéia bem definida de como deve ser o produto. O modelo utilizado deve ser de fácil manutenção, para permitir mudanças geométricas e dimensionais que por ventura sejam necessárias e além disto, permitir a reutilização de projetos anteriores, uma vez que o processo de desenvolvimento de produtos complexos pode representar um grande investimento considerando o tempo necessário para realizá-lo.

Os modelos paramétrico e variacional foram desenvolvidos para satisfazer estas necessidades. Os significados destes dois termos se confundem muito, tanto em contextos técnicos como principalmente em contextos comerciais, pois do ponto de vista do usuário final, os dois sistemas praticamente não possuem diferenças, uma vez que ambos fornecem suportes semelhantes ao processo de desenvolvimento de produtos (SHAH e MÄNTYLÄ, 1995), em que o usuário cria um modelo do produto e descreve as propriedades requeridas em termos de restrições geométricas. O sistema, então avalia o modelo, verificando a integridade e a existência de inconsistências pela execução de um algoritmo geral de solução de restrições, por fim, o usuário pode criar variações do modelo alterando valores das variáveis restringidas. Uma discussão mais detalhada sobre este assunto é realizada no Anexo A.

1.2.3 Importância da História de Execução de Operações e a Dependência entre elas.

Modelos paramétricos e variacionais trazem como grande contribuição aos sistemas CAD a automatização das alterações, isto é, para um dado conjunto de alterações no modelo, uma nova instância é gerada automaticamente. Para um processo interativo como o projeto de produtos, esta capacidade é fundamental.

O projeto de um novo produto pode ocorrer de duas maneiras, na primeira, o projetista não faz idéia de como o novo produto deve ser como um todo e tem em mãos apenas as restrições funcionais; e na segunda, o projetista já possui um modelo mental do produto com um conjunto suficiente de detalhes para criá-lo autilizando-se de operações de modelagem em um sistema CAD.

No segundo caso a história de execução de operações se apresenta como uma forma de mapear as intenções do projetista, em outras palavras, desta maneira o projetista pode converter o seu modelo mental em um modelo computacional. CUGINI (1989) ressalta a importância desta característica em sua hipótese: "o procedimento de trabalho adotado pelo projetista experiente não é casual e não objetiva apenas a produção de gráficos, mas que indiretamente reflete as conexões funcionais entre as partes representadas no desenho".

Outras características desejáveis em um sistema CAD são apresentadas por INUI e KIMURA (1993): o suporte consistente à função UNDO (desfazer uma operação) e REDO (alteração de uma operação já executada), como uma forma de se conseguir uma manutenção consistente do modelo.

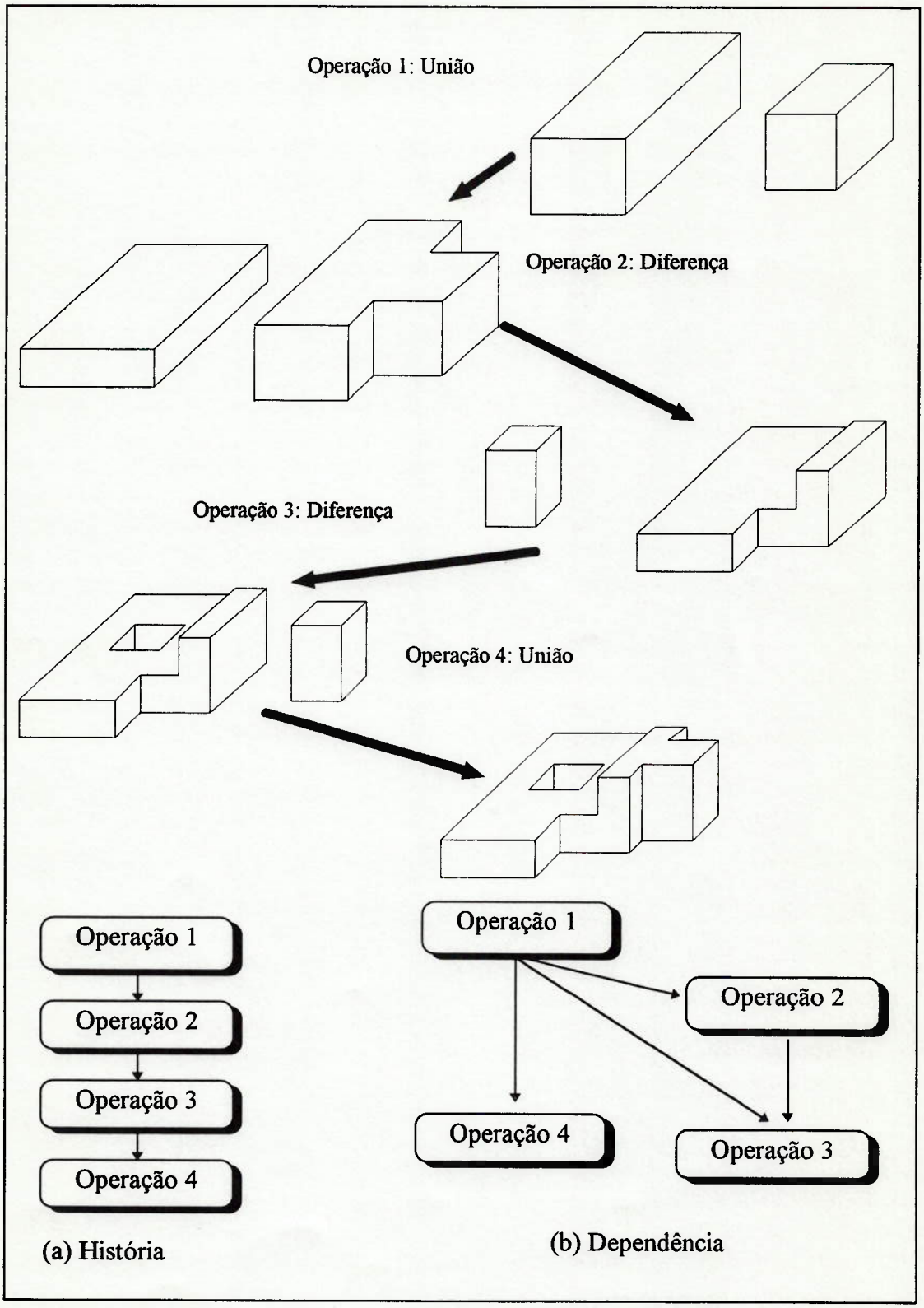

Figura 1.3 - História e dependências entre operações.

Através da utilização da história de execução de operações sobre o modelo paramétrico, a manutenção consistente do modelo pode ser obtida pela re-execução (REDO) ou anulação (UNDO) de todas as operações que foram executadas após a operação que foi alterada (REDO) ou desfeita (UNDO) pelo usuário. Entretanto, para suportar um UNDO/REDO consistente, as informações de história de execução não são suficientes, pois a alteracao de uma operação irá provocar a anulação/re-execução de todas as outras operações que foram executadas posteriormente. Isto pode ser evitado se informações sobre dependências entre as operações estiverem disponíveis para o sistema.

A Figura 1.3 mostra um exemplo de relações de dependência entre operações para uma dada seqüência histórica de operações. Neste exemplo verificamos que se a Operação 1 for alterada ou desfeita, todas as outras precisarão ser re-executadas ou desfeitas. Mas caso a Operação 2 seja desfeita ou alterada, apenas a Operação 3 depende dos resultados dela e, portanto, somente a Operação 3 deve ser desfeita ou re-executada, permanecendo a Operação 4 inalterada. Note que se fossem utilizadas apenas informações de história de execução (figura 1.3 (a)), a operação 4 também seria desfeita ou re-executada.

A história de operações realizadas além de permitir a definição das possíveis operações que dependem daquela que foi modificada, garante que uma operação já realizada não venha a depender de uma operação realizada posteriormente.

1.3 Organização do Trabalho

Este trabalho está organizado de forma que no capítulo 2, entitulado Conceitos Básicos, são apresentados conceitos básicos como modelagem de sólidos, suas

representações computacionais, com ênfase para a representação por fronteiras (*B-Rep - Boundary Representation*), operadores de Euler, dimensões relativas e estruturas de manutenção de dependências.

No capítulo 3, entitulado Revisão Bibliográfica, diversos trabalhos realizados na área de CAD Variacional e Paramétrico são apresentados segundo uma classificação conveniente.

No capítulo 4, entitulado Metodologia e Implementação, as diversas fases da metodologia adotada para a execução do trabalho é apresentada.

No capítulo 5, entitulado Resultados Experimentais, os resultados obtidos experimentalmente são apresentados.

No capítulo 6, entitulado Discussão, uma análise dos resultados obtidos para cada caso simulado experimentalmente é realizada.

No capítulo 7, entitulado Conclusões, a avaliação geral do trabalho é realizada, os resultados obtidos são discutidos e propostas para desenvolvimentos futuros são sugeridas.

No Anexo A, entitulado Sistemas CAD Paramétrico e Variacional, uma discussão sobre sistemas CAD paramétricos e variacionais é realizada procurando evidenciar as diferenças entre as metodologias. Um exemplo é utilizado para esclarecer melhor estas diferenças.

2. CONCEITOS BÁSICOS

Neste capítulo alguns conceitos básicos de modelagem de sólidos, suas representações, características e propriedades são apresentados. A estrutura de manutenção de verdade (*TMS - Truth Maintenance System*) proposta por DOYLE (1979) e o TMS baseado em asserções (*ATMS - Assumption-based Truth Maintenance System*), proposto por DE KLEER (1986a) são apresentados.

2.1 Modelagem de Sólidos

A modelagem de sólidos é um ramo da modelagem geométrica que tem como objetivo a obtenção de um modelo geométrico que possa ser utilizado para diversas aplicações sem a intervenção do usuário, isto é, um modelo cuja representação seja "completa" em informações geométricas (MÄNTYLÄ, 1988). Segundo esta definição é conveniente que façamos uma breve discussão sobre o que é um modelo no contexto deste trabalho antes de abordarmos propriamente o tema Modelagem de Sólidos.

2.1.1 Modelos

Um modelo no contexto deste trabalho é um objeto abstrato, artificialmente definido, cuja finalidade é permitir que determinadas propriedades de um objeto real

possam ser evidenciadas e mais facilmente observadas (MÄNTYLÄ, 1988). Segundo esta definição, observamos que um conjunto de desenhos técnicos constitui um modelo, uma vez que, a partir destes desenhos bidimensionais, um projetista com o conhecimento apropriado consegue construir um modelo tridimensional.

Uma propriedade importante dos desenhos técnicos é que além deles possuírem simbolismos para a precisa representação geométrica das peças do projeto, eles servem como meio de comunicação entre as diversas etapas do processo de desenvolvimento de um produto e entre integrantes das diferentes equipes de desenvolvimento.

Para suportar o processo de desenvolvimento de um produto, um modelo computacional deve substituir o desenho técnico e representar as informações de maneira completa, consistente e precisa para permitir a automatização das operações no meio industrial.

Além das características citadas, este modelo computacional deve suportar a natureza interativa do processo de desenvolvimento de um produto, precisando ser muitas vezes impreciso e indefinido quando apropriado (MÄNTYLÄ, 1988).

2.1.2 O Modelo Sólido e suas Representações

Antes de analisarmos as características de um modelo sólido e suas representações, vamos fazer uma breve comparação entre a modelagem de sólidos e outras duas formas de representação geométrica, as representações geométricas por fio de arame (*wireframe*) e por superfícies.

O modelo sólido se diferencia de outras representações, como por fio de arame (*wireframe*) e por superfícies, por representar informações geométricas e uma maior

quantidade de informações sobre a topologia do sólido modelado. Uma comparação entre informações topológicas e geométricas podem ser observadas na figura 2.1.

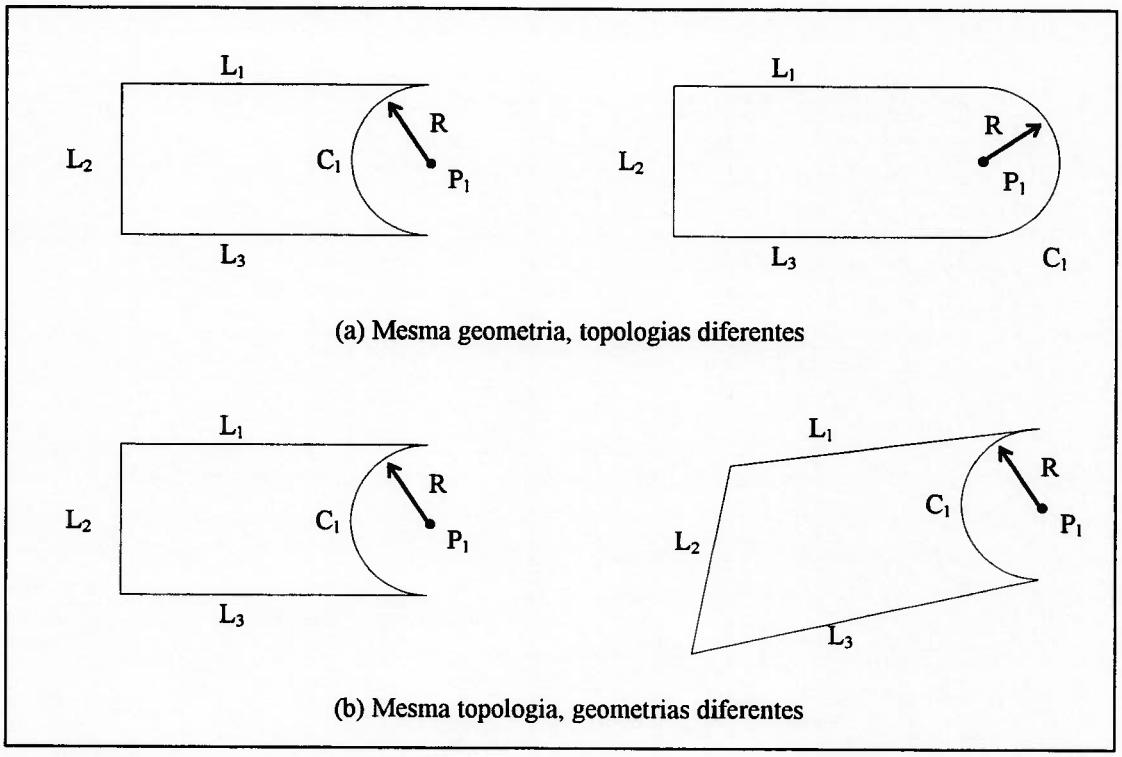

Figura 2.1 - Diferença entre geometria e topologia de um objeto. (ZEID, 1991)

Embora alguns autores afirmem que um modelo sólido não pode ser definido segundo uma representação por superfície é importante notar que isto não é totalmente verdadeiro. Um modelo sólido é representado por um conjunto de faces coerentemente orientadas, normais apontando para o exterior do sólido, por exemplo. As infornações topológicas, no caso de representação por superfície pode ser obtida por métodos algorítmicos. Entretanto, como exemplificado na figura 2.2, situações de ambigüidade podem ocorrer. O problema consiste na verificação da existência ou não de conexão entre as faces f_1 e f_2.

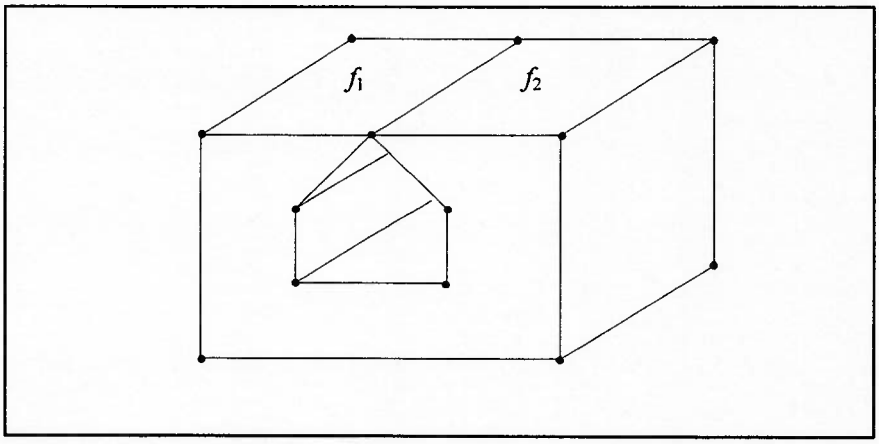

Figura 2.2 - Um modelo que pode ser ambíguo segundo a representação por superfície

Um modelo sólido, por conter informações sobre a adjacência, informações topológicas, entre as faces f_1 e f_2, é capaz de informar sem ambigüidades qual a relação existente entre estas duas faces. Já um modelo por superfícies, por interpretar a topologia do objeto de forma algorítmica, e não será capaz de gerar a mesma informação sem incorrer em ambigüidades (mais de uma interpretação).

Esta propriedade ficará mais clara ao explicarmos os diversos esquemas de representação.

Um esquema de representação é definido como uma relação $s: D \rightarrow V$ que mapeia um objeto $o \in D$ (domínio da representação) contido em W (espaço do objeto) em um modelo válido $v \in V$ (imagem do espaço do objeto) contidno em R (espaço da representação) (MÄNTYLÄ, 1988).

Um esquema de representação pode ser (a) não ambíguo e único; (b) não ambíguo e não único e (c) ambíguo, como ilustrado na figura 2.3.

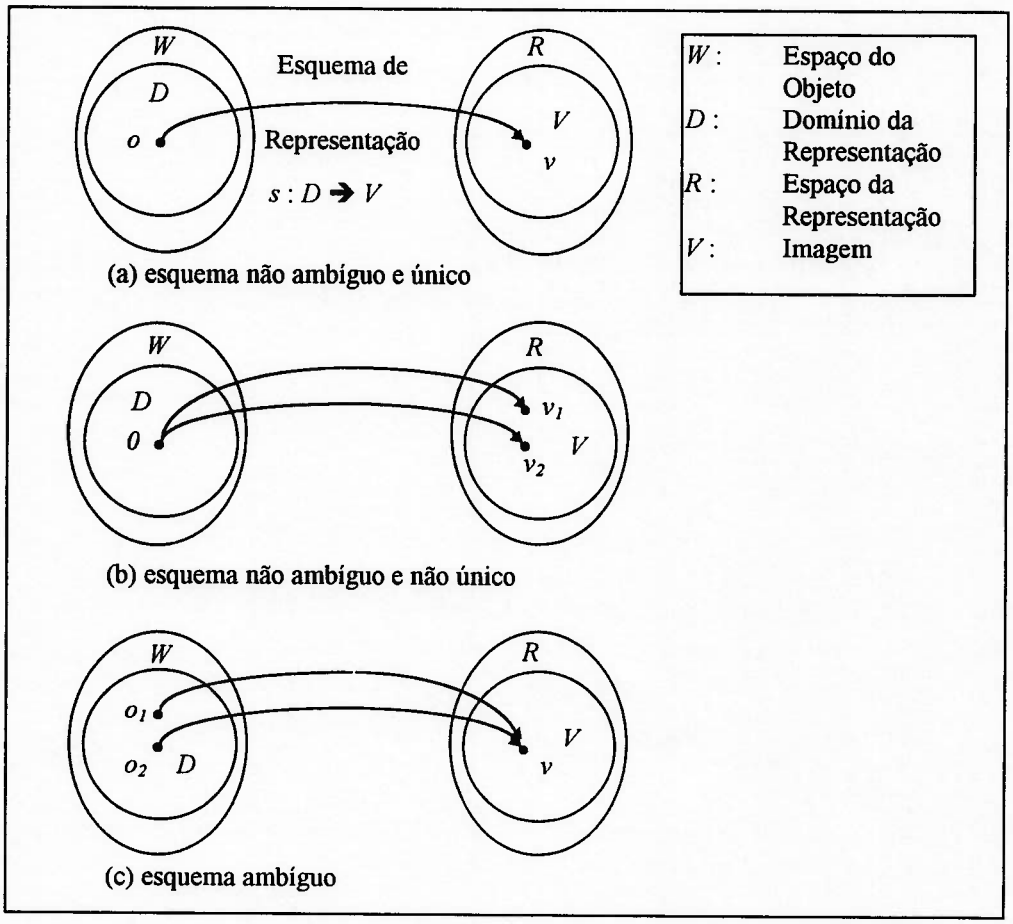

Figura 2.3 - Classificação dos esquemas de representação.

Quatro características dos esquemas de representação são bastante importantes e merecem ser analisadas com maior cuidado. São elas: o domínio da representação, sua validade, ambigüidade e unicidade.

a) Domínio

O domínio do esquema de representação indica quais são os objetos que o esquema pode representar, isto é, quais geometrias ele é capaz de representar. Um esquema de representação ideal é aquele em que o conjunto D tende a ser

igual ao conjunto W, isto é, o esquema de representação é capaz de representar todos os objetos.

b) Validade

A validade de um esquema de representação é o conjunto de representações válidas, representações estas que podem ser consideradas modelos computacionais associados a objetos. A validade é importante pois ela é uma das condições para que algoritmos automáticos de análise possam ser aplicados.

A validade pode ser obtida de três formas. A primeira é através da realização de constantes verificações na base de dados do sistema. A segunda é através da verificação da validade durante a criação do modelo computacional e a terceira é a utilização de primitivas pré-definidas com validade assegurada que serão então manipuladas. O problema de verificação da validade de um modelo é ainda dividido em duas partes, a primeira é a verificação da validade topológica do modelo e a segunda é a verificação da validade geométrica do modelo, pois um modelo pode ser topologicamente válido mas devido a sua geometria não estar associado a nenhum objeto de D.

c) Não ambigüidade

A não ambigüidade é uma outra condição para que algoritmos automáticos de análise possam ser aplicados. Observando a figura 2.3 verificamos que uma representação é não ambígua quando verificamos que se o mapeamento s de D em V de d_1é v_1 e o de d_2 também é v_1, então d_1 é igual a d_2, como observa-se na equação 2.1 .

$$s(d_1,v_1) \wedge s(d_2,v_1) \rightarrow d_1 = d_2 \qquad (2.1)$$

d) Unicidade

Através desta propriedade um esquema pode verificar ou não, se dois modelos computacionais representam o mesmo objeto. Esta propriedade é raramente verificada em esquemas de representação (ZEID, 1991).

A seguir diversas formas de representação para a modelagem de sólidos serão brevemente discutidas, com ênfase nas duas representações mais significativas, a CSG (*Constructive Solid Geometry*) e a B-Rep (*Boundary Representation*) (ZEID, 1991).

Neste trabalho, fazendo uma simplificação de linguagem, um modelo definido por uma representação de modelagem de sólidos será referenciado apenas por sólido, e o objeto real, físico apenas por objeto.

2.1.2.1 Representação por Varredura (*Sweep Representation*)

A representação por varredura é bastante conveniente para a criação de modelos sólidos de objetos 2,5-dimensionais. Estes sólidos são aqueles que podem ter uma espessura constante (sólidos criados por extrusão) ou uma simetria em torno de um eixo (sólidos criados por revolução). Isto acontece porque neste tipo de representação o sólido é um volume gerado pela varredura translacional ou rotacional de lâminas bi-dimensionais. Em uma varredura translacional uma lâmina plana bidimensional define um volume segundo uma curva tridimensional. Esta curva é chamada de trajetória. Em uma varredura rotacional o sólido é criado pela rotação da lâmina em torno de um eixo de rotação segundo um ângulo dado. Este eixo é o eixo de simetria do objeto.

Este tipo de representação é limitada pois o seu domínio se restringe a sólidos simples e a verificação da validade do sólido criado não é uma tarefa computacionalmente trivial, pois dependendo da trajetória um modelo inválido pode ser criado (ZEID, 1991).

2.1.2.2 Representação por Instanciação de Primitivas (*Primitive Instantiation*)

O método de instanciação de primitivas é baseado no agrupamento de objetos que possuem a mesma topologia e geometrias diferentes em famílias, chamadas de primitivas genéricas (ZEID, 1991). Pré-definindo a topologia de primitivas, o usuário precisa informar ao sistema apenas as dimensões parametrizadas do objeto e a família a que ele pertence.

Através da utilização deste método a utilização e a criação de sólidos consistentes e não ambíguos é uma tarefa bastante simples; entretanto, para uma utilização efetiva desta representação é necessário um estudo preliminar para identificar os tipos de primitivas que serão necessárias na aplicação. Uma vantagem desta representação é que ao invés de desenvolvermos algoritmos genéricos, podemos utilizar algoritmos específicos para cada primitiva. Assim, o cálculo de propriedades tais como volume, massa, momento de inércia, etc... pode ser realizado de maneira específica e otimizada para cada tipo de primitiva existente (MÄNTYLÄ, 1988).

2.1.2.3 Representação por Decomposição em Células (*Cell Decomposition*)

Neste esquema de representação, um objeto é decomposto em diversas células menores, contíguas e que não se interpenetram. Embora as células possam possuir qualquer forma e tamanho, normalmente em representações computacionais, elas são cubóides de tamanhos idênticos. Basicamente três tipos de células são criadas durante a decomposição de um objeto: vazias, cheias ou semi-cheias. Para este tipo de representação por decomposição, dois esquemas se destacam (SINGH, 1996): a utilização de uma grade simples, com células tridimensionais de tamanho fixo, iguais e contíguas, também chamada na literatura por enumeração exaustiva (*exhaustive enumeration*) (MÄNTYLÄ, 1988) e a utilização de uma grade com células de tamanhos variáveis, definidas recursivamente. Este último método é conhecido na literatura por *octree* (SINGH, 1996) ou esquema de subdivisão adaptativa (*adaptative subdivision scheme*) (MÄNTYLÄ, 1988).

2.1.2.4 Representação por Semi-espaços (*Half-Spaces*)

O semi-espaço é considerado o elemento básico de contornos de sólidos. Matematicamente um semi-espaço H é definido como um conjunto regular de pontos no espaço Euclidiano tridimensional E^3:

$$H = \{ V : f(V) < 0, V \in E^3 \} \qquad (2.2)$$

onde V é um ponto de E^3 e $f(V) = 0$ define a equação da superfície que limita o semi-espaço.

Embora $f(V)<0$ represente qualquer superfície de contorno, normalmente, os semi-espaços mais utilizados são o plano, o cilíndrico, o esférico, o cônico e o toroidal, que são dados por:

Plano: $H = \{ (x,y,z): z<0 \}$ (2.3)

Cilíndrico: $H = \{ (x,y,z): x^2 + y^2 < R^2 \}$ (2.4)

Esférico: $H = \{ (x,y,z): x^2 + y^2 + z^2 < R^2 \}$ (2.5)

Cônico: $H = \{ (x,y,z): x^2 + y^2 < [(\tan \alpha/2)z]^2 \}$ (2.6)

Toroidal: $H = \{ (x,y,z): (x^2 + y^2 + z^2 - R_1^2 - R_2^2)^2 < 4 R_2^2(R_1^2 - 1) \}$ (2.7)

Na figura 2.4 podemos verificar a representação gráfica dos parâmetros das equações para os semi-espaços cônico (equação 2.6) e toroidal (equação 2.7).

Utilizando estes semi-espaços, pode-se formar diversos sólidos por meio de operações de conjunto. Por exemplo, um bloco pode ser formado por seis semi-espaços planos utilizando-se operadores AND.

Uma grande vantagem desta representação é sua concisão, uma vez que se baseia em um número limitado de primitivas. Entretanto esta representação pode criar modelos sólidos não limitados se não for utilizada com cuidado pelo projetista (ZEID, 1991).

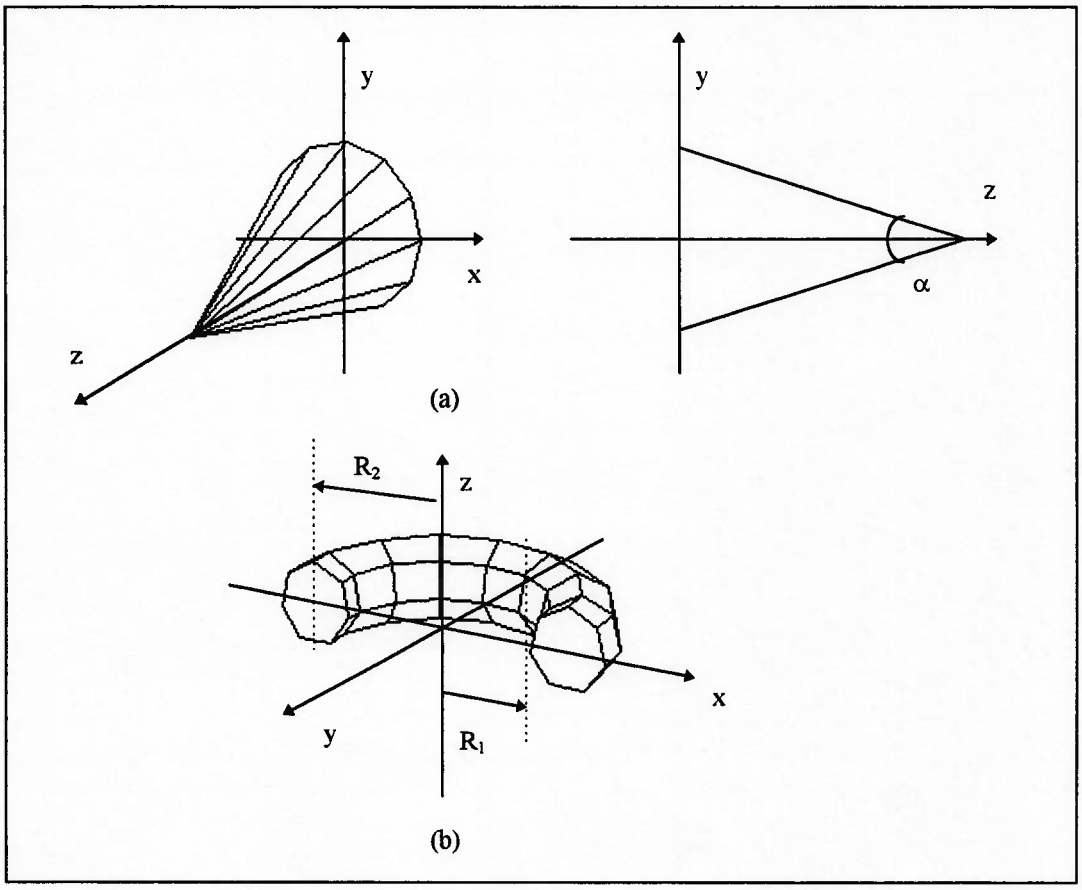

Figura 2.4 - Parâmetros da equação do semi-plano cônico (a) e toroidal (b).

2.1.2.5 Representação por geometria sólida construtiva (*CSG - Constructive Solid Geometry*)

A CSG é uma representação bastante utilizada em modelagem de sólidos. Pela CSG, um sólido é definido pela combinação de sólidos simples. Estes sólidos simples são chamados de primitivas ou entidades sólidas. Um sólido é representado segundo uma árvore binária onde as suas folhas são primitivas e os nós internos são operadores booleanos que relacionam os dois nós inferiores. A construção de modelos CSG combinando primitivas por operadores booleanos é relativamente fácil, pois é uma tarefa intuitiva. A estrutura de armazenamento de dados é bem concisa, pois um sólido

representado pela CSG não possui informações explícitas sobre quais são as faces, arestas e vértices que compõem o sólido. Caso seja necessário determiná-las, isto pode ser realizado de forma algorítmica. Devido ao que foi anteriormente descrito, a representação por CSG é lenta para exibir os sólidos e por isto normalmente ela é convertida para uma representação B-Rep (ver abaixo) para este fim. Por isto, muitos sistemas são híbridos e utilizam as duas representações, suportando um ambiente intuitivo de criação e flexibilidade na visualização de sólidos.

Várias primitivas são fornecidas aos usuários dependendo do sistema CAD/CAM utilizado. As primitivas disponibilizadas dependem da aplicação a que estão orientados os sistemas. As primitivas utilizadas pela representação CSG podem ser interpretadas como sendo uma combinação de semi-espaços, dando suporte para um formalismo matemático para esta representação. Alguns destes sistemas ainda fornecem a possibilidade de se definir novas primitivas.

A validade dos sólidos criados é mantida pela sua construção a partir de primitivas válidas e pela utilização de operações booleanas *regularizadas* do tipo UNIÃO, INTERSEÇÃO e DIFERENÇA (ZEID, 1991). As operações booleanas regularizadas produzem como resultado apenas sólidos válidos, isto é, não produzem pontos, nem linhas e nem superfícies isoladas.

2.1.2.6 Representação por Fronteiras (*B-Rep - Boundary Representation*)

A representação B-Rep descreve um sólido definindo um conjunto de faces que o limitam, que constituem as suas fronteiras, classificando assim os pontos do espaço em internos ao sólido, externos ao sólido, e sobre o contorno do sólido. Cada face do sólido é limitada por um conjunto de arestas, e as arestas por sua vez são limitadas por vértices.

Nos sistemas de modelagem de sólidos podemos definir três níveis de representação de um sólido (TSUZUKI *et al.* 1991). O primeiro nível, o nível mais alto de abstração, interage com o usuário através de operações para construir, modificar e armazenar os sólidos. Neste nível o sólido é descrito por um conjunto de operações de alto nível, que permite o interfaceamento com o usuário. Entre o primeiro nível e o segundo nível de abstração localiza-se toda a infra-estrutura matemática e algorítmica do sistema de modelagem que suportam e tornam executáveis as operações disponibilizadas ao usuário no primeiro nível. No segundo nível, o sólido é descrito por um conjunto de operadores específicos chamados de operadores de Euler, que serão tratados em maiores detalhes a seguir. No terceiro nível, o nível inferior de abstração, o sólido é representado por uma estrutura de dados computacional.

Por ser uma representação de modelos sólidos, a representação B-Rep precisa armazenar informações de como as faces, as arestas e os vértices de um sólido estão inter-relacionadas (relações de adjacência). Estas informações são denominadas por informações topológicas. As informações que descrevem a geometria da superfície, a descrição da geometria das arestas e a localização dos vértices são denominadas por informações geométricas. As informações topológicas criam um vigamento no qual as informações geométricas são aglutinadas. As nove relações de adjacência, definidas pela combinação dois a dois entre os três elementos primitivos da B-Rep (face, aresta e vértice) podem ser vistas na figura 2.5. Nesta figura, a relação VV é entre um vértice e outros vértices que com este definem uma aresta, a relação EV é entre uma aresta e os dois vértices que a definem, a relação FV é entre uma face e os vértices que a definem, a relação VE é entre um vértice e as arestas limitadas por ele, a relação EE é entre uma aresta e outras quatro arestas que possuem a mesam face como contorno, a relação FE é

entre uma face e as arestas que a limitam, a relação VF é entre um vértice e as faces que o possuem, a relação EF é entre uma aresta e as duas faces que ela separa e finalmente, a relação FF é entre uma face e as outras que a circundam.

As diversas estruturas computacionais para a representação B-Rep devem ser capazes de fornecer as relações de adjacência expostas na figura 2.5 (WEILER, 1985). Dentre elas citamos três que são baseadas na aresta como elemento de referência: a estrutura "winged-edge" (BAUMGART, 1975), a estrutura "aresta-dividida" (*half-edge*) (WEILER, 1985) e a estrutura unificada. Em um sistema de modelagem de sólidos, estas estruturas constituem o terceiro nível de representação de um sólido.

Por conterem todas as informações sobre as adjacências (topologia), estas estruturas computacionais anteriormente expostas são bastante complexas e a sua manipulação exige muitos cuidados para que a consistência dos dados seja mantida.

Para contornar o problema acima descrito, um conjunto de operadores foi

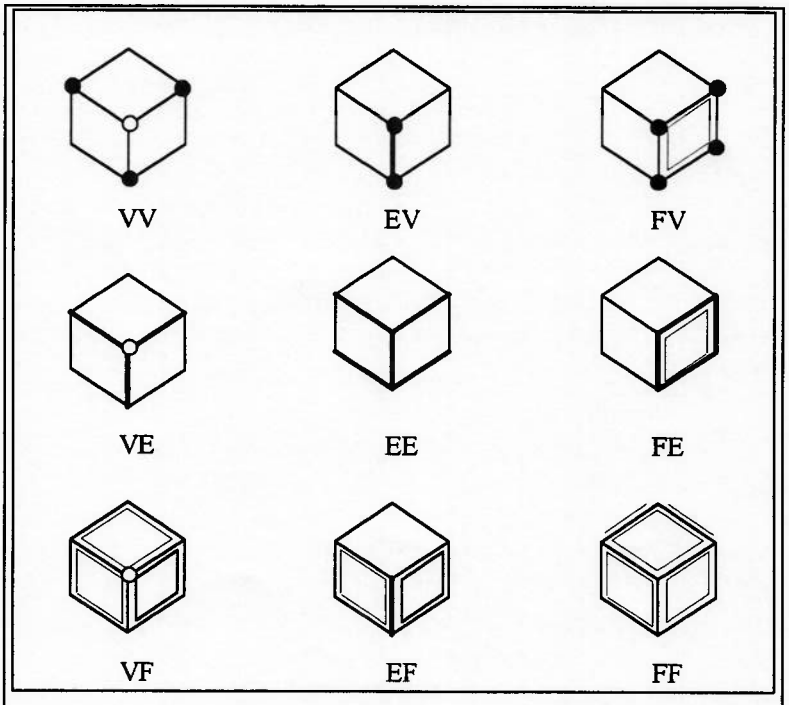

Figura 2.5 - Relações de Adjacência (WEILER, 1985).

desenvolvido para manipular as estruturas de dados da representação B-Rep de forma consistente. Estes operadores formam um conjunto mínimo de operadores e são chamados por Operadores de Euler. Eles permitem que a construção do sólido possa ser executada passo a passo escondendo todos os detalhes de implementação da representação, estes operadores formam o segundo nível de abstração da representação B-Rep.

Dentre as várias vantagens da utilização deste nível de abstração destacamos:

a) Construção incremental do modelo

A utilização de operadores de Euler permite que sólidos uni, bi ou tridimensionais possam ser criados passo a passo, realizando alterações locais no modelo.

b) Esconder detalhes do nível de abstração inferior

Vários autores (TSUZUKI,1991; MÄNTYLÄ, 1988) afirmam que um conjunto mínimo de cinco operadores são suficientes para construir qualquer sólido. Desta forma, os comandos disponibilizados ao usuário podem ser descritos por seqüências de execução de apenas cinco operadores de Euler, independentemente da estrutura de dados computacional escolhida para o sistema.

c) Manutenção automática da validade topológica do sólido criado

Conforme já comentado, a base de dados de uma representação B-Rep contém informações topológicas e geométricas. Pela utilização dos operadores de Euler a

consistência topológica pode ser obtida automaticamente segundo a equação conhecida por equação de Euler-Poincaré.

Os elementos básicos constituintes de um sólido são face, aresta e vértice. Entretanto, algumas extensões são necessárias para definir alguns casos especiais. Por exemplo, uma face deve ser limitada por um conjunto de arestas, entretanto, verificamos que estes conjuntos podem ser disconexos, como é o caso de uma face com um furo, e que estes conjuntos de arestas formam laços fechados. Uma forma de contornar esta situação é a utilização de arestas pontes, que ligam os laços que definem uma face, formando um único grande laço. Um outro elemento é inserido, o anel, que nada mais é que uma face com um furo. Por isto, o número de anéis em um sólido é definido como a diferença entre o número de laços e o número de faces. Caso não houvessem faces com furos, o número de laços seria igual ao número de faces. Um sólido em criação pode possuir ainda peças disconexas, chamadas por *shell* (*s*) em situações intermediárias de operações complexas.

A equação de Euler-Poincaré diz que um sólido poliédrico é topologicamente válido se a seguinte relação entre seus elementos constituintes for verificada (ZEID, 1991):

$$v - e + 2f = 2(s - h) + l \qquad (2.8)$$

onde v é o número de vértices do sólido, e o número de arestas, f o número de faces, s o número de peças disconexas do sólido, h o número de furos e l o número de laços. A forma mais conhecida na literatura da equação de Euler-Poincaré supõe ainda a existência de r anéis no sólido, onde $r = l - f$. Ficando a equação da seguinte forma (MÄNTYLÄ, 1988; WILSON, 1985):

$$v - e + f = 2(s - h) + r \qquad (2.9)$$

Abaixo são relacionados e detalhados os cinco operadores de Euler mais comumente utilizados na literatura (TSUZUKI, 1991; MÄNTYLÄ, 1988) e uma representação gráfica para cada operador pode ser observada na figura 2.6:

- **Operador mvsf** (*Make Vertex Solid Face*): este operador cria um sólido incial com apenas uma face e um vértice.
- **Operador mev** (*Make Edge Vertex*): este operador adiciona a um sólido uma aresta e um vértice. A aresta é criada conectando-se um vértice já existente ao novo vértice criado.
- **Operador mef** (*Make Edge Face*): este operador adiciona ao sólido uma aresta e uma face. A face é criada pela divisão de uma face já existente acrescentando-se a nova aresta.
- **Operador kemr** (*Kill Edge Make Ring*): este operador divide o contorno de uma face em dois laços pela remoção de uma aresta-ponte.
- **Operador kfmrh** (*Kill Face Make Ring Hole)* : este operador une duas faces de maneira que o laço da face removida se torne o anel da segunda face.

Os cinco operadores detalhados acima são todos construtivos, entretanto, para que o sólido possa ser alterado satisfatoriamente ele precisa também de algumas operações destrutivos. Por isto, cada operador construtivo possui um operador correspondente destrutivo. São eles:

- **kvsf** (*Kill Vertex Solid Face*) ←→ **mvsf** (*Make Vertex Solid Face*)
- **kev** (*Kill Edge Vertex*) ←→ **mev** (*Make Edge Vertex*)
- **kef** (*Kill Edge Face*) ←→ **mef** (*Make Edge Face*)
- **mekr** (*Make Edge Kill Ring*) ←→ **kemr** (*Kill Edge Make Ring*)
- **mfkrh** (*Make Face Kill Ring Hole*) ←→ **kfmrh** (*Kill Face Make Ring Hole*)

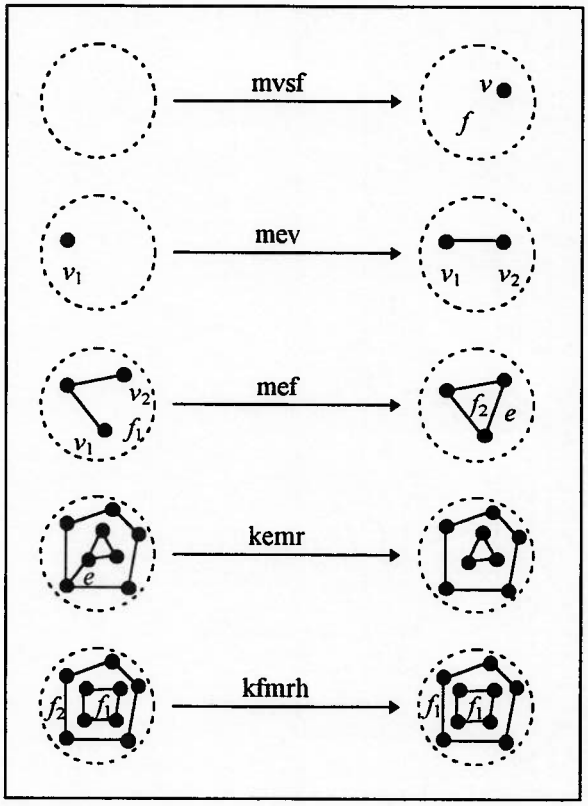

Figura 2.6 - Operadores de Euler.

Durante o processo de construção de um sólido pela utilização de operadores de Euler, a validade topológica do mesmo é mantida observando-se a equação de Euler-Poincaré. Entretanto, é comum o agrupamento de operadores de Euler em uma certa seqüência para mantermos também a geometria válida. É importante observarmos que não há como manter a geometria válida em todos os estágios intermediários da

construção. Por isto, os operadores de Euler devem ser agrupados em seqüências que possuem algum significado (TSUZUKI, 1991).

A combinação de operadores de Euler em seqüências com significado é realizada pela interface existente entre o nível mais alto de abstração do sistema, e o segundo nível de abstração.

Na figura 2.7 um exemplo pode ser observado em que um operador de alto nível (criar cubo com furo passante) aciona uma seqüência de operadores de Euler. Estes operadores de alto nível podem ser organizados em dois grupos; operadores locais e operadores booleanos.

Os operadores locais permitem que algumas características dos sólidos possam ser modificadas diretamente pelo usuário preservando as consistências topológica e geométrica do local modificado. Por não realizarem alterações em áreas externas à localidade das características a que elas se aplicam, verificações de validade não são realizadas sobre o sólido após a realização de uma operação local. Entretanto existe a possibilidade de se criar um sólido inválido utilizando-se operações locais, como por exemplo, um sólido auto-interceptante (figura 2.8). A verificação da validade do sólido não é uma tarefa fácil, ficando, geralmente, dependente da correta utilização destas operações pelo usuário. Como exemplos de operadores locais temos os processo de arredondamento, extrusão, chanframento e colagem (*gluing*).

Os operadores booleanos são utilizados para determinar a união, intersecção e diferença entre dois sólidos, e são necessários em modeladores de sólidos B-Rep quando se disponibiliza ao usuário uma interface semelhante a de sistemas baseados na CSG. Os

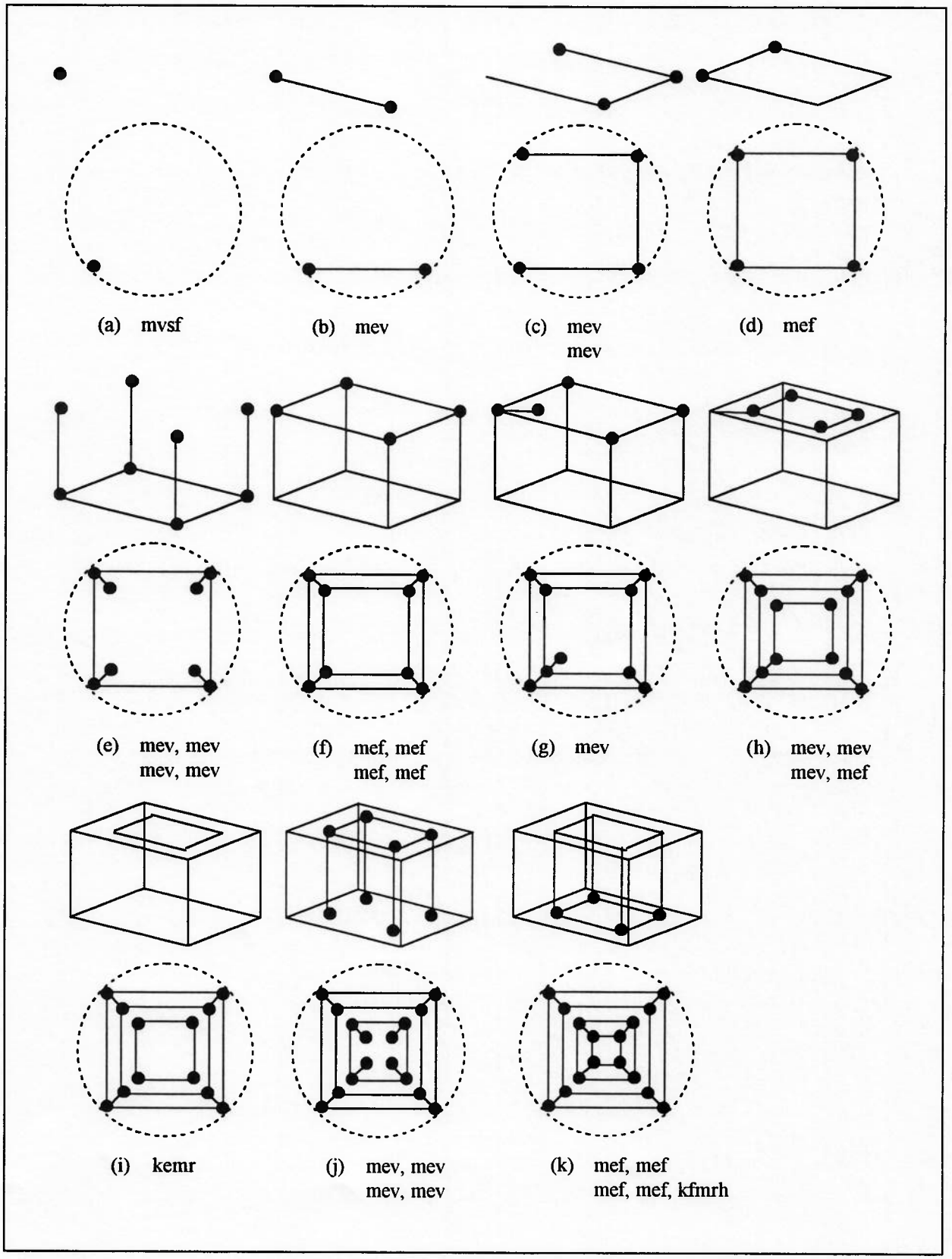

Figura 2.7 - Seqüência de criação de um cubo com furo passante (MÄNTYLÄ, 1988).

algoritmos envolvidos são complexos pois os operadores booleanos são operadores que não atuam localmente, isto é, eles não alteram o modelo localmente. Uma característica interessante destes operadores é que, por serem baseados em operações de conjunto regularizadas, e por manipularem o sólido de forma global, eles sempre resultam em sólidos válidos, não havendo necessidade de verificação.

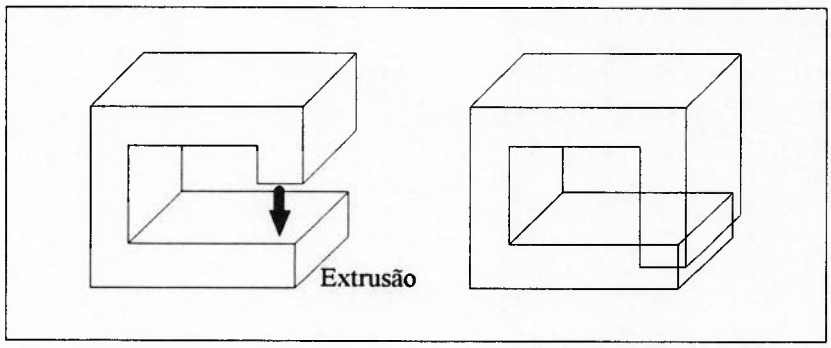

Figura 2.8 - Exemplo de criação de um sólido inválido auto-interceptante através da execução de uma operação local de extrusão.

2.2 Dimensões Relativas

Ao trabalharmos com sólidos nos deparamos com dois tipos de dimensões: dimensões locais e dimensões relativas. As dimensões locais são aquelas que têm sua definição e aplicação em um mesmo sólido. Já as dimensões relativas são aquelas que definem posicionamentos entre sólidos independentes, tornando ou não a combinação destes sólidos única e rígida (TSUZUKI, 1994).

Uma característica de um sistema CAD paramétrico é dar suporte na definição de esquemas de dimensionamento, principalmente quando modificações em algumas

dimensões já utilizadas são necessárias. A implicação destas modificações não é trivial, uma vez que todo o esquema de dimensionamento deve ser reavaliado de forma automática.

Utilizando classificações do tipo válida para quando uma dimensão está de acordo com as especificadas pelo projeto e inválida para quando não estiver de acordo, pode-se criar um mecanismo em que o usuário torna este conjunto de dimensões inválidas quando deseja alterar dimensões do sólido, e o sistema paramétrico transforma estas dimensões inválidas em dimensões válidas novamente. A figura 2.9 ilustra este mecanismo.

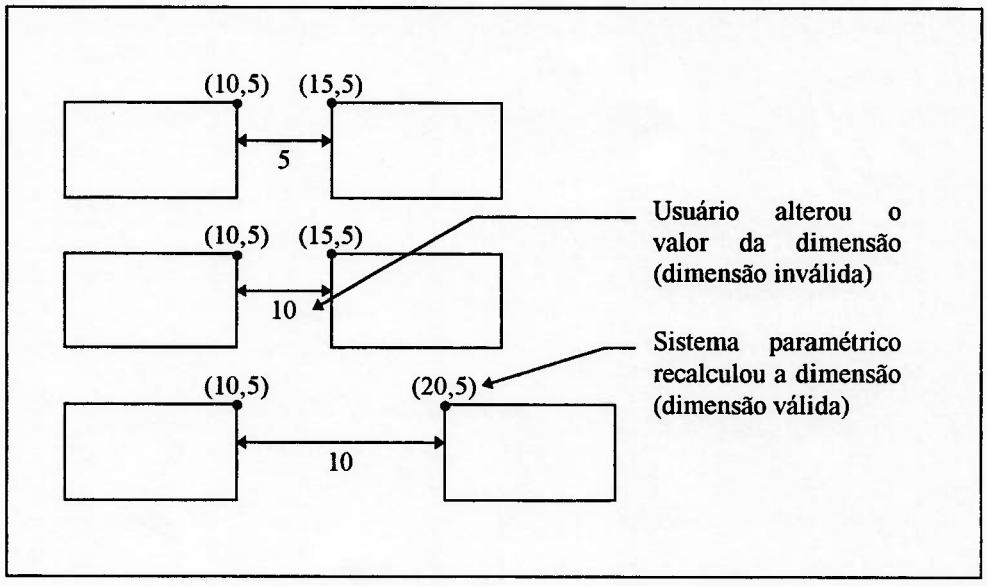

Figura 2.9 - Comportamento de uma dimensão em um sistema paramétrico.

As dimensões relativas são definidas sempre entre dois sólidos, um considerado fixo (referência) e outro considerado móvel. Os tipos de dimensões relativas, que totalizam quinze, podem ser vistos na tabela 2.1.

Tabela 2.1 - Os quinze tipos de dimensões relativas (TSUZUKI, 1994)

Tipo de Restrição
Distância plano-plano com normais coincidentes
Distância plano-plano com normais opostas
Ângulo plano-plano
Distância plano-linha (e vice-versa)
Ângulo plano-linha (e vice-versa)
Distância plano-ponto (e vice-versa)
Distância linha-linha com direções coincidentes
Distância linha-linha com direções opostas
Ângulo linha-linha
Distância linha-ponto (e vice-versa)
Distância ponto-ponto

A manipulação dos sólidos para se conseguir satisfazer uma dimensão relativa é realizada por meio de transformações. O problema de satisfazer uma dimensão relativa é então resumido à busca de um conjunto de transformações que modifique o estado das dimensões relativas inválidas para válidas. Este problema é tratado por TSUZUKI (1994).

2.3 Sistemas de Manutenção da Verdade (TMS - Truth Maintenance Systems)

DOYLE (1979) define o sistema de manutenção de verdade (TMS) como um subsistema de resolução de problemas que permite que um programa possa assumir algumas hipóteses iniciais (asserções) e mais tarde revisar suas crenças quando descobertas posteriores vierem a contradizer as hipóteses iniciais. Isto é obtido pelo armazenamento e manutenção das razões para cada crença. Estas razões armazenadas são utilizadas ainda na construção de explicações para as ações realizadas.

O TMS se diferencia de outros sistemas de raciocínio por permitir uma abordagem não monotônica. Os sistemas convencionais de raciocínio baseiam-se na monotonicidade da seqüência de estados das crenças do sistema de raciocínio. Desta forma, enquanto suas crenças forem verdadeiras o sistema de raciocínio vai aumentando o conjunto de crenças consideradas verdadeiras com mais crenças verdadeiras (DOYLE,1979). Um problema gerado pela monotonicidade é que se uma das crenças não for mais verdadeira, dificilmente se conseguirá verificar quais crenças que devem ter seu estado válido modificado.

2.3.1 O Sistema de Manutenção da Verdade baseado em Suposições (*ATMS - Assumption-based Truth Maintenance System*)

Os sistemas ATMS diferenciam-se do TMS de Doyle por não manipularem apenas as justificativas, mas também por manipularem os conjuntos de suposição (*assumptions*) (DE KLEER, 1986a). Há diversas vantagens na utilização de sistemas ATMS para sistemas de busca de soluções, tais como possibilidade de trabalhar de modo consistente e eficiente com informações inconsistentes, a não restrição do espaço de busca, permitindo a mudança fácil de contexto, sem a necessidade de retroceder na seqüência de inferências e como conseqüência, poder explorar todo o espaço de busca e encontrar múltiplas soluções.

O ATMS é uma ferramenta que permite armazenar informações de dependências entre predicados. Estes predicados podem ser de dois tipos: suposições (*assumptions*) ou relações derivadas (*derived relations*). As suposições caracterizam-se por não dependerem de nenhum outro predicado, podendo ser utilizadas para representar

escolhas ou decisões. As relações derivadas dependem de outros predicados e em última instância de um conjunto de asserções.

O ATMS monta uma rede de dependências cujo elemento básico é o nó. O nó possui três propriedades: descrição (*datum*), rótulo (*label*) e justificativa (*justification*). A descrição de um nó contém informações descritivas do nó. O rótulo armazena os conjuntos de suposições do qual depende o nó, cada conjunto de suposições a partir do qual o nó pode ser derivado é chamado de ambiente (*environment*). A justificativa armazena o conjunto de nós dos quais o nó é diretamente derivado, podendo ser suposições ou relações derivadas. A representação de um nó é dada por:

```
Nome do nó: < descrição, rótulo, justificativa >
```

No ATMS existe ainda um terceiro tipo de nó, chamado de inconsistente (*no-good*). A descrição deste tipo de nó expressa textualmente o problema, seu rótulo contém os conjuntos de suposições que se contradizem e sua justificativa indica os nós dos quais o nó deriva diretamente.

Na figura 2.10 temos a representação gráfica da rede ATMS para as relações expressas pelas equações em 2.1, adotando a notação de AN_i (*Assumption Node*) para o i-ésimo nó suposição, DN_i (*Derived Node*) para o i-ésimo nó derivado e FALSO para os nós contradição. Na figura 2.10 foi utilizada ainda uma notação para expressar a dependência do tipo "e" pela utilização de uma seta que se inicia na junção de dois arcos provenientes dos nós dos quais a relação derivada depende. Observamos também que o nó DN_3 possue uma relação do tipo "ou" com os nós DN_1 e DN_2.

Observando as relações de 2.1, verificamos facilmente que A, B, D e G são suposições, e que C, E e F são derivados de outros nós.

$A \wedge B \rightarrow C$ (2.1)

$B \wedge D \rightarrow E$

$C \vee E \rightarrow F$

$G \wedge E \rightarrow$ inconsistente

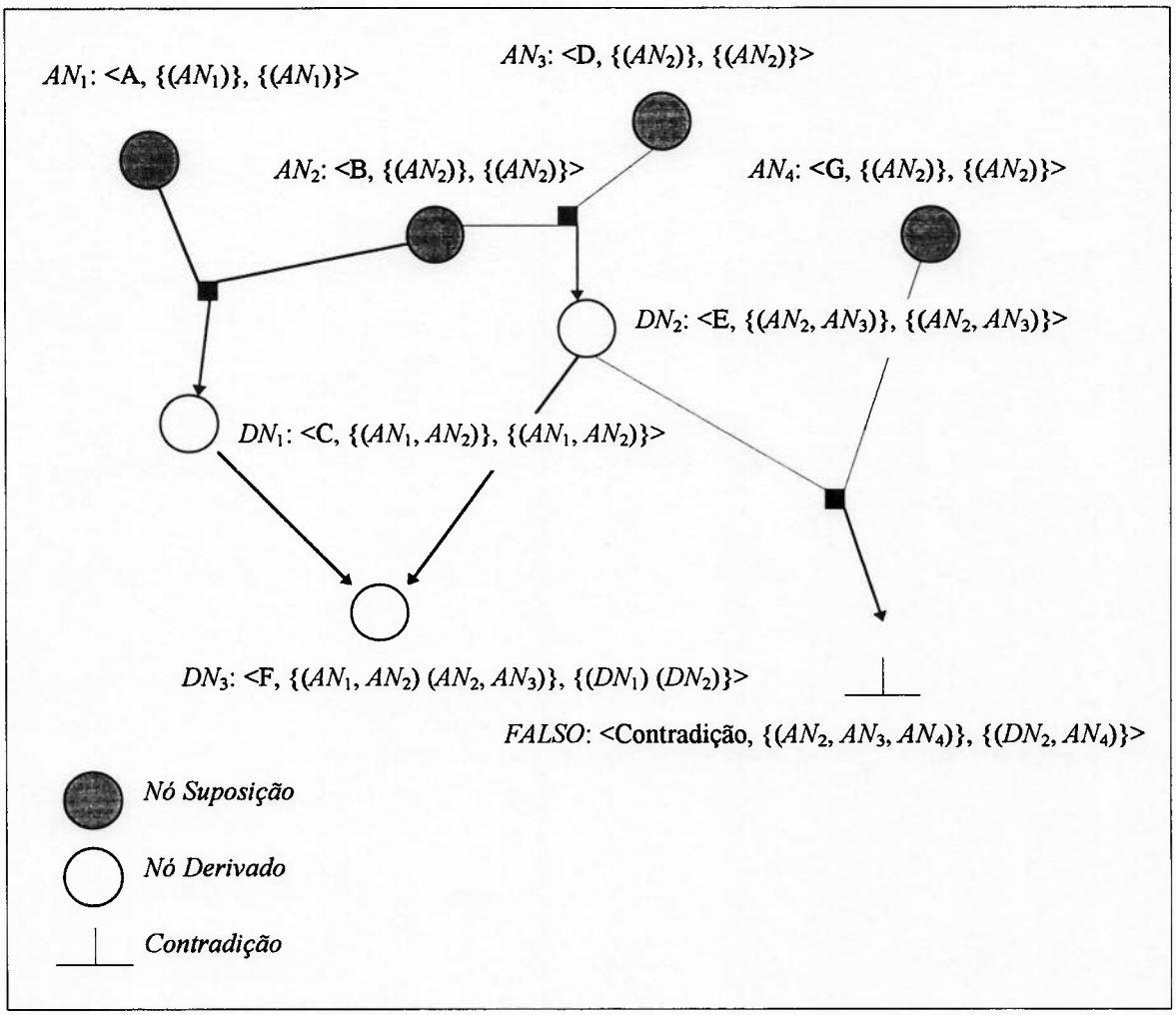

Figura 2.10 - Exemplo de uma rede ATMS.

Uma propriedade importante dos nós derivados é a possibilidade de verificação de sua validade somente utilizando as informações constantes em seu rótulo e o conjunto

de suposições que são consideradas válidas em um dado instante. Este conjunto de suposições é chamado de contexto (*Context*).

Os nós do tipo FALSO permitem que a exclusão de determinadas combinações de suposições possa ser realizada, diminuindo o número de contextos válidos para a busca de soluções.

Para melhor compreensão da nomenclatura utilizada na figura 2.10, o nó DN_3 será detalhado. Observamos que sua descrição (datum) contém a descrição do nó, no caso F. Sua justificativa contém dois conjuntos de nós dos quais ele depende diretamente. No caso, ele depende de DN_1 ou de DN_2, por isto contém dois conjuntos de nós. O seu rótulo contém dois conjuntos de suposições, dos quais o nó depende. Observamos que estes conjuntos de suposições são os mesmos dos quais os nós DN_1 e DN_2 também dedependem.

3. REVISÃO BIBLIOGRÁFICA

As diversas propostas de sistemas paramétricos e variacionais podem ser classificadas, conforme ROLLER (1989) em abordagens primárias, algébricas e baseadas em técnicas de inteligência artificial.

3.1 Abordagens Primárias

As abordagens primárias caracterizam-se por manipularem um número muito limitado de restrições e por utilizarem algoritmos muito específicos, restringindo assim a área de utilização dos sistemas desenvolvidos nesta plataforma. Um trabalho deste tipo de abordagem é apresentado por FITZGERALD (1981), onde apenas se considera dimensões lineares nas direções vertical e horizontal. Pela construção de um grafo adequado, pode-se verificar quando um elemento está ou não bem determinado.

3.2 Abordagens Algébricas

As abordagens algébricas caracterizam-se por traduzir todas as restrições em equações, transformando o projeto em um problema de resolução numérica de um sistema de equações não lineares, definindo assim as coordenadas de pontos importantes do sólido.

Uma vez que qualquer restrição pode ser traduzida em equações algébricas, estas abordagens têm como característica serem muito genéricas e poderosas, suportando tanto modelos bidimensionais como tridimensionais.

Pelo fato de resolverem simultaneamente o conjunto de equações, estas abordagens são capazes de lidar com problemas em que as restricões impostas possuem um alto grau de dependências.

Estas abordagens apresentam entretanto, duas limitações sérias. A primeira é relacionada ao fato de utilizarem um método numérico iterativo para solucionar o sistema de equações, isto é, o sistema encontra apenas uma solução para casos em que há múltiplas soluções, e esta solução encontrada ainda depende dos valores iniciais fornecidos. A segunda limitação é o alto custo computacional que os métodos numéricos iterativos possuem (TSUZUKI, 1994).

HILLYARD e BRAID (1978a, 1978b) descrevem como uma forma geométrica pode ser manipulada pela alteração das coordenadas de pontos singulares satisfazendo as restrições impostas. Basicamente o usuário necessita informar a topologia da forma, isto é, fornecer todos os dados relativos às diversas relações entre vértices, faces e arestas. Assim o sistema pode manipular estes dados como uma estrutura do tipo "treliça" onde os comprimentos das arestas são desconhecidos. Assim monta-se um sistema de equações, e à medida que o usuário acrescenta dimensões, são introduzidos elementos "enrigecedores" na "treliça" que vão limitando os graus de liberdade.

Outro trabalho expressivo da abordagem algébrica é o apresentado por LIGHT e GOSSARD (1982). Nesta proposta, um vetor geométrico, que contém as coordenadas de todos os pontos característicos é confrontado com um vetor de dimensões por

funções que representam as restrições. A solução numérica é encontrada pelaa utilização do método de Newton-Raphson.

3.3 Abordagens Baseadas em Técnicas de IA

São numerosas as abordagens baseadas em técnicas de IA. Neste tipo de abordagem os métodos adotados utilizam mecanismos de inferência para parametrizar o projeto.

ALDEFELD (1988) propõe um método em que a entrada de restrições e o processamento das mesmas ocorre em duas fases. Na primeira fase as restrições estruturais são lidas e verificadas pelo sistema uma a uma, evitando assim informações redundantes ou conflitantes. Ainda nesta primeira fase, um sistema de inferência aplica um conjunto de regras sobre as restrições estruturais conhecidas, derivando uma série de outras restrições. Estas inferências são armazenadas na forma de um histórico de inferências a partir do qual um plano de construção é gerado. Na segunda fase o sistema utiliza o plano de construção, dimensões introduzidas pelo usuário (restrições métricas), o protótipo geométrico e procedimentos de execução (aplicação de métodos numéricos para encontrar a solução do problema), além de um ponto e uma orientação de referência para encontrar uma solução. Os procedimentos de execução (cálculo numérico) são aplicados no modelo de acordo com o plano de construção e o protótipo geométrico é utilizado para solucionar os casos em que há mais de uma solução para um conjunto de equações, optando-se sempre pela solução mais próxima do protótipo. Um ponto a ser ressaltado na abordagem de ALDEFELD é que para um mesmo conjunto de restrições, independentemente da ordem de inserção, um mesmo plano de contrução será obtido.

INUI e KIMURA (1993) propõem a utilização de um ATMS (*Assumption-based Truth Maintenance System*) para armazenar informações sobre a dependência entre operações realizadas sobre um modelo geométrico tridimensional, com ênfase na possiblidade de se realizar uma operação de UNDO consistente e manutenção do modelo. Não indicam entretanto como a conexão entre o sistema de manutenção de verdade e a representação gráfica do sólido é realizada.

No sistema proposto por QIU e HUANG (1994) todo o processamento é dividido em três fases. A primeira fase, criação e edição de elementos gráficos, permite a definição de relações topológicas (passos diferentes na fase de desenho geram diferentes topologias) e a introdução de restrições geométricas que não podem ser modificadas (somente pode ser realizada a remoção da restrição, e mesmo assim, somente após a remoção dos elementos associados). Na segunda fase, é realizada a inserção de restrições dimensionais. Estas restrições podem ser posteriormente removidas ou modificadas. Na terceira fase, projeto variacional dirigido por dimensões, a regeneração do modelo por suas dimensões é realizada, mantendo sempre as relações topológicas e satisfazendo as restrições geométricas entre os elementos.

Em CUGINI (1989) o sistema GIPS (*Graphic Interactive Parametric System*) é apresentado e a atenção é chamada para a importância do procedimento pelo qual o projetista traduz o seu modelo funcional em um desenho. Baseando-se na hipótese de que um procedimento de trabalho adotado por um projetista não é arbitrário nem apenas objetivando a produção de um desenho, mas que indiretamente reflete as interconexões funcionais entre os diversos elementos do desenho, o sistema guarda estas informações e gera um modelo procedural, isto é, um modelo descrito pelas operações executadas para a sua criação. Este modelo procedural é armazenado na forma de um grafo de relações.

Desta maneira, uma classe de produtos pode ser armazenada em um grafo de relações, pois para que uma nova instância seja gerada, basta executar as operações como armazenadas no grafo para um novo conjunto de parâmetros.

4. METODOLOGIA E IMPLEMENTAÇÃO

4.1 Metodologia Adotada

Para o desenvolvimento de um sistema CAD que suporte as tarefas de criação de sólidos, sua visualização e manipulação, foi adotada a metodologia descrita a seguir.

Em primeiro lugar, foram definidas as especificações das funcionalidades desejadas para todo o sistema. As funcionalidades desejadas são exemplificadas em alguns casos típicos.

Em seguida, a definição da arquitetura do sistema, seus blocos funcionais principais, o fluxo de informações e os meios em que isto ocorre, são definidos. Nesta fase as descrições são mais detalhadas e servem como ponto de partida para a implementação dos blocos funcionais e das interfaces do sistema. Ainda nesta fase, para cada bloco funcional definido, uma breve discussão das necessidades é realizada e alguns conceitos importantes são apresentados para que depois a implementação possa ser detalhada.

Com o sistema implementado, um conjunto de testes é realizado para que os resultados experimentais possam ser discutidos e avaliados. Estes resultados são apresentados no capítulo 5 e discutidos no capítulo 6.

Finalmente, após a realização da análise de resultados, propostas para outros trabalhos que deêm continuidade ao que foi realizado são apresentadas.

4.2 Funcionalidades do Sistema

O sistema a ser implementado deve permitir que o usuário construa um sólido por meio de uma interface amigável além de permitir que este sólido seja manipulado de maneira consistente. Para tanto, funções para construção, visualização e manutenção de sólidos devem ser disponibilizadas ao usuário. Por funções de manutenção de sólidos, entendemos que sejam funções pelas quais o usuário pode modificar o sólido alterando operações já realizadas. Basicamente, duas funções realizam esta tarefa, as funções de UNDO e REDO.

4.2.1 UNDO

Os sistemas CAD devem ser capazes de anular os efeitos de uma alteração no sólido que não produziu os resultados esperados pelo usuário. Sistemas CAD baseados na história de execução permitem que isto seja realizado de maneira seqüencial, isto é, permitem que a ultima operação seja anulada, voltando sempre para um estado anterior.

Entretanto no sistema a ser implementado a operação de UNDO deve ser realizada de forma consistente, em que o usuário seleciona a operação que deve ser anulada, e somente as alterações provocadas por esta operação devem ser desfeitas.

Na figura 4.1 um exemplo das diferentes abordagens para a função UNDO é ilustrada. Uma peça (a) sofre uma operação de furação (b) e em seguida passa por um outro processo em que outro furo é realizado e um rasgo retangular é introduzido (c). O

projetista então chega a conclusão que o ideal é que na parede vertical do objeto o furo cilíndrico seja substituído por um rasgo retangular, por motivos quaisquer, tais como maior facilidade de montagem ou menor custo de usinagem. Um sistema baseado somente na história de execução de operações irá desfazendo seqüencialmente as operações até que a operação anulada seja desfeita, voltando para o estado incial (a). O UNDO consistente da operação de furação indicada, deveria dar como resultado o estado (d) da figura 4.1, permitindo então a criação do rasgo retangular, como exemplificado no estado (e). O fluxo indicado pelas setas pontilhadas ilustram a seqüência de estados gerados por um UNDO convencional para se passar do estado (c) para o (e). O fluxo indicado pela seta cheia indica a seqüência de estados gerados por um UNDO consistente.

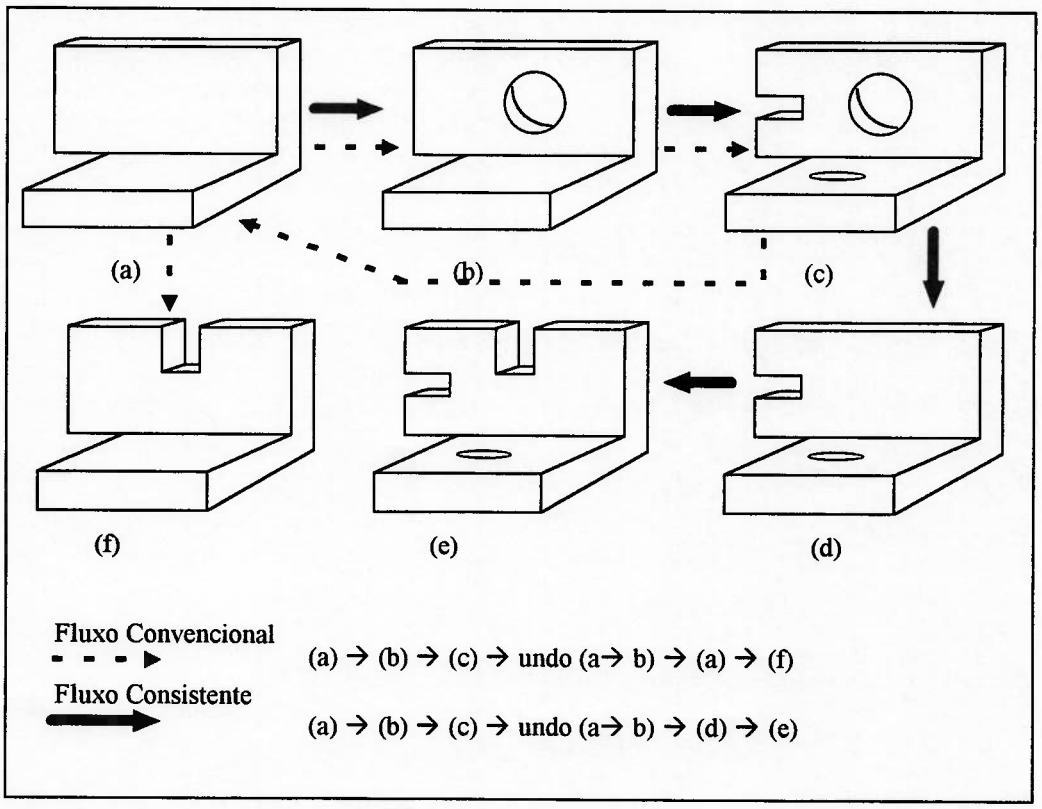

Figura 4.1 - UNDO: convencional e consistente (INUI e KIMURA, 1993).

4.2.2 REDO

Além de serem capazes de anular os efeitos de uma operação, os sistemas CAD devem ser capazes de refazer uma operação com, por exemplo, outro conjunto de parâmetros. A esta funcionalidade damos o nome de REDO. Sistemas CAD baseados na história de execução identificam qual operação foi alterada, realizam um UNDO seqüencial até esta operação, executam esta operação com o novo conjunto de parâmetros e executam novamente todas as outras na seqüência.

No sistema a ser implementado, a operação de REDO deve ser realizada de uma forma mais consistente, em que somente as operações que possuem alguma dependência em relação àquela que foi selecionada (para ser executada novamente com um novo conjunto de parâmetros), devem ser desfeitas e executadas novamente.

Para o caso ilustrado na figura 4.1, uma operação de REDO poderia realizar a mudança desejada (do estado (c) ao (e)) através da re-execução da operação de furação substituindo-a por uma de criação de um rasgo retangular.

4.3 Arquitetura do Sistema

O sistema desenvolvido foi definido como uma composição do modelador de sólidos do Departamento de Engenharia Mecânica da USP, o USPDesigner, e um Sistema de Apoio. Desta forma, funções básicas de modelagem de sólidos já são disponibilizadas ao usuário pelo USPDesigner, necessitando apenas acrescentar as

funções de UNDO e REDO consistentes ao sistema formado pelo USPDesigner e o Sistema de Apoio.

A interface entre estes dois sistemas deve ser realizada por meio de arquivos de tipo texto. Isto é desejável, pois como o Sistema de Apoio ainda é um protótipo, é interessante o isolamento entre os dois sistemas para avaliação isolada das saídas e entradas de ambos.

Neste protótipo o usuário ainda interage com os dois sistemas, gerando o sólido no USPDesigner, que passa então informações de execução de operações em um arquivo de tipo texto para o Sistema de Apoio. O usuário pode então realizar operações de UNDO ou REDO no Sistema de Apoio que envia informações sobre o que deve ser realizado no USPDesigner também por meio de um arquivo de tipo texto.

4.3.1 O USPDesigner

O USPDesigner é um modelador de sólidos onde o sólido possui três níveis de representação. No nível superior, o sólido é descrito por operações de alto nível, segundo uma representação semelhante à CSG. Estas operações são então tratadas por algoritmos numéricos e geométricos e no segundo nível de representação o sólido é descrito por operadores de Euler, segundo uma representação B-Rep e finalmente, a estrutura computacional adotada para armazenar todas as informações no terceiro nível de representação é a de aresta dividida (discutada na seção 2.1.2.6).

A função de UNDO seqüencial do modelador foi implementada como uma fila de operações do tipo LIFO (*Last In First Out*) que é preenchida com a operação complementar da operação executada pelo usuário. A lista de operações de UNDO é

gerada no segundo nível de representação, isto é, é composta por uma série de operadores de Euler, e isto se deve ao fato de que se fossem utilizadas as operações do primeiro nível, os algoritmos numéricos e geométricos precisariam ser executados novamente, podendo representar um custo computacional elevado. A manipulação de informações diretamento no terceiro nível de representação também não seria adequada, pois as relações de dependências entre as diversas informações não são facilmente detectadas, além da dificuldade de se manipular um volume grande de informações de forma consistente. O mecanismo de funcionamento da operação de UNDO seqüencial do USPDesigner consiste portanto na execução destes operadores de Euler complementares de forma a anular os efeitos das operações originais.

O USPDesigner é, portanto, um modelador de sólidos que pode gerar um arquivo texto com as informações de UNDO (lista ordenada de operadores de Euler complementares), com informações de REDO (lista ordenada com os operadores de Euler originais) e informações dos comandos de alto nível executados pelo usuário.

Além de gerar arquivos com as informações acima descritas, o USPDesigner é capaz de executar operações a partir de um arquivo texto, sejam estas operações de alto nível (representação por CSG) ou de baixo nível (operadores de Euler).

Alguns detalhes de como o USPDesigner manipula os elementos básicos da representação B-Rep devem ser analisados para que estas propriedades possam ser utilizadas pelo sistema de apoio.

O USPDesigner manipula e identifica os elementos básicos da representação B-Rep utilizando-se de identificadores numéricos únicos para cada elemento. A atribuição de valores é realizada de forma incremental. Assim, não existirão duas faces com um mesmo identificador, embora possa existir um sólido, uma face, um vértice, uma aresta e

um shell com um mesmo identificador, não ocorrendo, entretanto, conflito, pois eles são de tipos diferentes. Desta forma o modelador é sempre capaz de identificar um elemento de forma única.

Os operadores de Euler possuem como parâmetros os elementos que precisam estar presentes para que a operação possa ser realizada, bem como o identificador da operação de nível superior realizada..

Algumas operações, as chamadas transformações (rotação e translação), caracterizam-se por possuírem como parâmetro apenas o identificador de um sólido, pois alteram o posicionamento de todos os vértices do sólido, embora não provoquem alterações de topologia e nem de geometria.

5.3.2 O Sistema de Apoio

O Sistema de Apoio é constituído de duas estruturas principais, uma estrutura que armazena a seqüência de operações executadas pelo usuário, chamada de Sistema de História, e uma estrutura que armazena e manipula as informações de dependências entre as operações executadas, chamada de Sistema ATMS. A seguir uma breve explicação do funcionamento do conjunto formado por estes módulos é fornecida.

Os arquivos gerados pelo USPDesigner são lidos pelo Sistema de Apoio que insere em um Sistema de História, informações combinadas de operações de alto nível executadas pelo usuário e as informações de REDO, chamaremos esta estrutura por Histórico de REDO para referências futuras. Em um outro Sistema de História, uma combinação das informações das operações de alto nível executadas pelo usuário e as

informações de UNDO são introduzidas, chamaremos esta estrutura por Histórico de UNDO para diferenciá-la da anterior.

A partir do Histórico de REDO, o Sistema de Apoio constrói a estrutura de dependências, que chamaremos por ATMS. Dispondo desta estrutura pronta, o usuário pode realizar operações de UNDO ou REDO de operações executadas no USPDesigner, e o ATMS informa quais as operações que serão ou não alteradas por esta escolha.

No caso do UNDO, com as informações fornecidas pelo ATMS, o Sistema de Apoio constrói um arquivo texto com uma seqüência de operadores de Euler do Histórico de UNDO. Pela verificação de quais operações que foram executadas posteriormente à que foi anulada e que não foram alteradas, acrescenta-se uma lista de operadores de Euler do Histórico de REDO a este arquivo. Este arquivo é então enviado ao USPDesigner que executa os operadores listados e reconstrói o sólido com as alterações desejadas.

No caso do REDO, os mesmos passos seguidos pela operação de UNDO são realizados. Entretanto, no arquivo enviado ao USPDesigner, algumas informações adicionais são enviadas: a operação de alto nível que deve ser refeita é colocada no arquivo, assim como todas as operações de alto nível que foram invalidadas por sua re-execução.

5.3.2.1 O Sistema de Armazenamento da História de Execução

Como descrito anteriormente, este módulo é responsável pela leitura dos arquivos gerados pelo USPDesigner e por sua organização em uma lista, permitindo a consulta de informações de forma ordenada. As informações devem ser armazenadas de forma hierárquica e seqüencial, pois uma ordem cronológica das operações de alto nível

deve ser respeitada, assim como a ordem cronológica dentro de cada conjunto de operações de Euler que correspondem à cada operação de alto nível.

5.3.2.2. O Sistema de Armazenamento de Informações de Dependências

Para armazenar e manipular informações de dependências entre operações de modelagem executadas pelo usuário, uma estrutura ATMS foi implementada. Para uma melhor compreensão de como estas dependências foram armazenadas, é importante uma explicação de como as operações podem ser representadas no ATMS. Com o objetivo de tornar isto mais claro, basearemos a explicação no exemplo da figura 4.2.

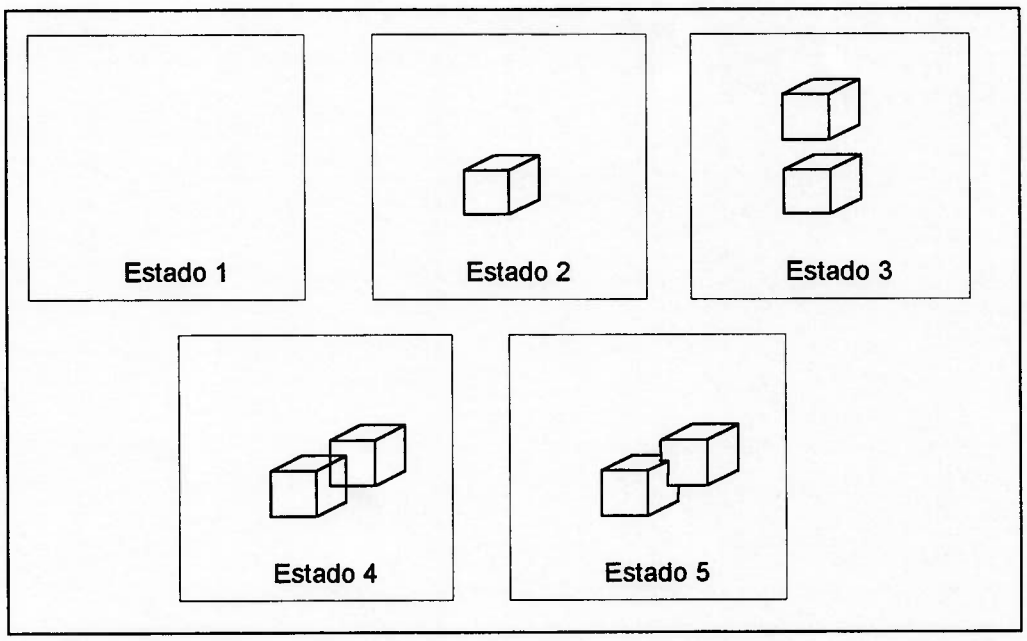

Figura 4.2 - Exemplo de uma seqüência de operações de modelagem.

No caso exemplo, partimos de um estado inicial em que temos um espaço tridimensional vazio, sem nenhum elemento geométrico. Criamos então um cubo,

chamado por cubo1, isto é, um sólido s_1, 6 faces de f_1 a f_6, doze arestas de e_1 a e_{12}, e oito vértices de v_1 a v_8. O segundo estado, consiste na existência de um cubo. Em seguida um segundo cubo, chamado por cubo2, é criado, isto é, um sólido s_2, 6 faces de f_7 a f_{12}, doze arestas de e_{13} a e_{24}, e oito vértices de v_9 a v_{16}. O terceiro estado representado consiste de dois cubos, cubo1 e cubo2. Transladamos o cubo2, isto é, o sólido s_2 para uma posição conveniente. O quarto estado também é formado por dois cubos, entretanto o cubo2 está em uma posição diferente. Por fim uma operação de união entre os cubos, cubo1 e cubo2, é realizada resultando em um sólido chamado por um nome qualquer, como união, designado por s_3. A seqüência destas operações pode ser observada na figura 4.2.

5.3.2.2.1 Modelagem de Operações no ATMS

Para modelar operações e estados no processo de criação de modelos sólidos, utilizaremos a formulação proposta por MORRIS e NADO (1986). A seqüência de modelagem é representada por uma seqüência de nós ATMS que representam mundos (*World*), sendo o estado inicial (W_0) representado por um nó tipo mundo WN_0 (*World Node*) cuja justificativa contém apenas um nó tipo suposição WA_0 (*World Assumption*) e sua descrição pode ser Estado1.

Ao executarmos uma operação de modelagem (no exemplo, a criação de um cubo), criamos um segundo nó tipo mundo WN_1, cuja justificativa contém o nó suposição WA_1, que é criado junto com o nó WN_1, e o nó WN_0. A descrição do nó WN_1 poderia ser Estado2. Inserimos também o nó operação ON_1 (*Operation Node*) e sua suposição OA_1 (*Operation Assumption*). O nó ON_1 possui em sua justificativa os nós OA_1 e WN_1, e sua descrição pode ser Cria cubo1. A operação insere no modelo um

sólido, seis faces, doze arestas e oito vértices. A inserção da face f_1 no modelo, por exemplo, seria representada pela inserção de um nó tipo FN_1 (*Fact Node*) e de um nó NDA_1 (*Non Deletion Assumption*), sendo que a jusificativa do nó FN_1 deve conter o nó NDA_1 e o nó ON_1, e sua descrição poderia ser `Face1`.

A execução da segunda operação, `Cria cubo2`, é semelhante à descrita anteriormente, com índices de identificação alterados.

A execução da terceira operação, `Translação do cubo2`, introduz um conjunto de nós representando um novo estado e um conjunto de nós representando a operação. Ela se diferencia por não introduzir nenhum elemento novo (*Fact Node*) no modelo, mas por alterar um sólido já existente. Por isto, a execução desta operação está condicionada a existência do sólido `Cubo2`. Esta dependência é expressa ao se colocar na justificativa do nó operação correspondente, além de seu nó suposição e do nó mundo que indica o estado corrente, o nó operação que introduziu o nó fato `Sólido1`.

Por fim, a execução da última operação, a união entre os sólidos, introduz um terceiro sólido `União` que depende das operações que inserem os sólidos `Cubo1` e `Cubo2`.

A representação de uma operação de UNDO é realizada pela inserção de mais um estado (conjunto de nós mundo), inserindo um nó tipo FALSE cuja justificativa contém o nó mundo inserido e o nó suposição da operação anulada. O grafo descrevendo os nós do ATMS para a seqüência exemplificada pode ser observado na figura 4.3.

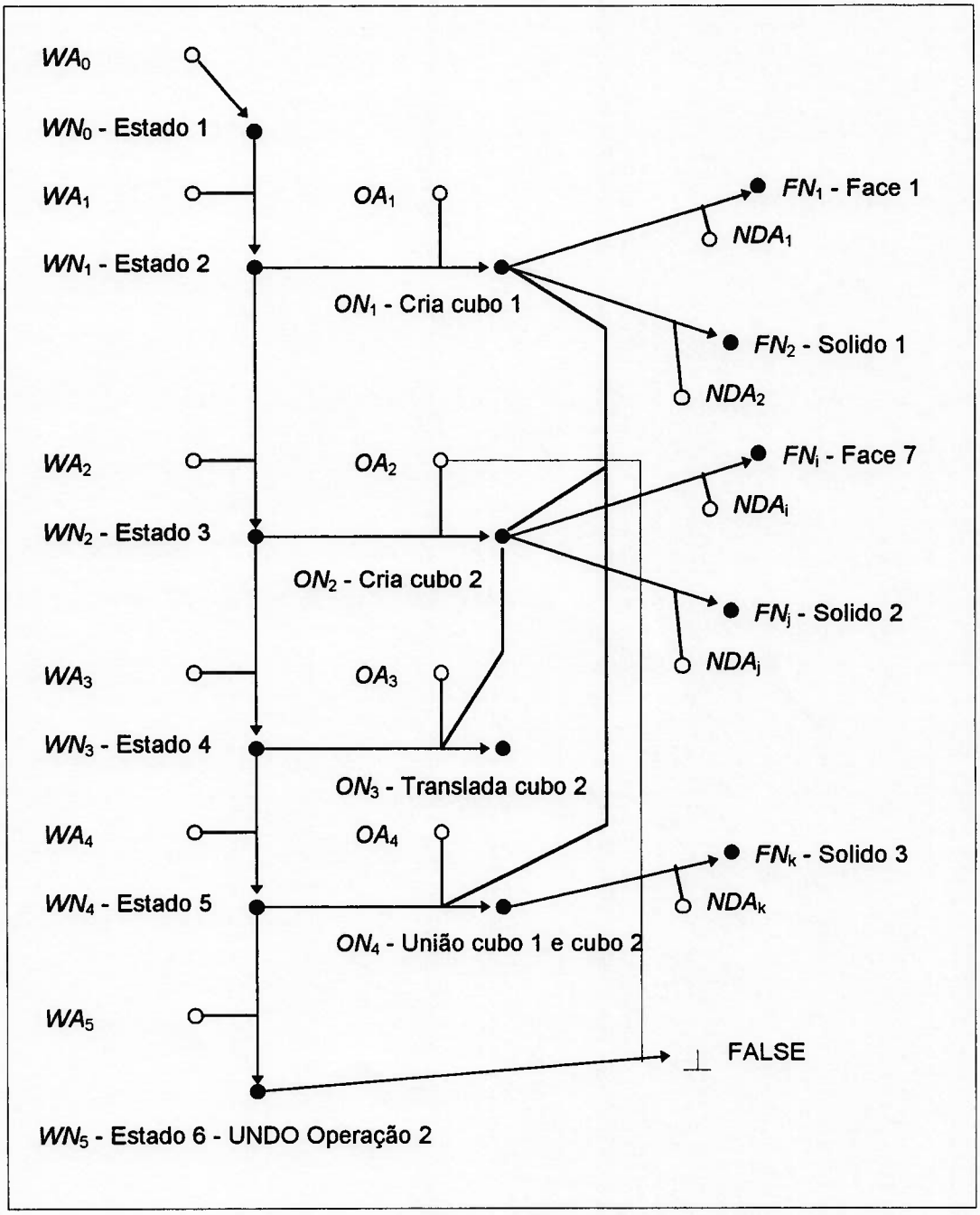

Figura 4.3 - Estrutura ATMS para o exemplo da figura 4.2.

4.4 Implementação do Protótipo

Para a implementação do protótipo do sistema, foram necessárias as implementações de algumas modificações no USPDesigner, e a implementação do Sistema de Apoio, isto é, a implementação da estrutura de armazenamento da história de execução de operações, tanto para informações de UNDO como de REDO, a implementação da estrutura ATMS e por fim, a integração destas estruturas no Sistema de Apoio.

Como o USPDesigner foi desenvolvido utilizando o compilador de linguagem C++ versão 4.52 da Borland para ambiente DOS, todo o Sistema de Apoio também foi desenvolvido neste ambiente. Cada sub sistema do Sistema de Apoio foi implementado como uma classe, para que uma integração direta com o USPDesigner possa ser realizada, eliminando a utilização de arquivos texto como meio de comunicação.

4.4.1 Implementação de Modificações no USPDesigner

Algumas modificações foram necessária no USPDesigner. A primeira e mais simples, foi a necessidade de se gerar um arquivo de REDO semelhante ao arquivo de UNDO, como descrito anteriormente.

A segunda, foi devida a um problema que encontramos devido à interface baseada na representação por CSG do USPDesigner. Observamos que se as operações booleanas são executadas entre dois sólidos, s_1 e s_2, gerando sempre como resultado um

terceiro sólido s_3, então ao realizarmos um UNDO desta operação, todas as operações que alteram o sólido s_3 após ela também serão anuladas, mesmo que venha só a alterar os sólidos s_1 ou s_2 separadamente. Ocorrendo isto, dificilmente conseguiríamos obter uma operação de UNDO consistente. No exemplo da figura 4.4, verificamos que ao realizarmos o UNDO da operação 3, a operação 5 também será anulada. Entretanto, observamos que a operação 5 poderia ser perfeitamente realizada considerando apenas a existência dos sólidos 1 e 4.

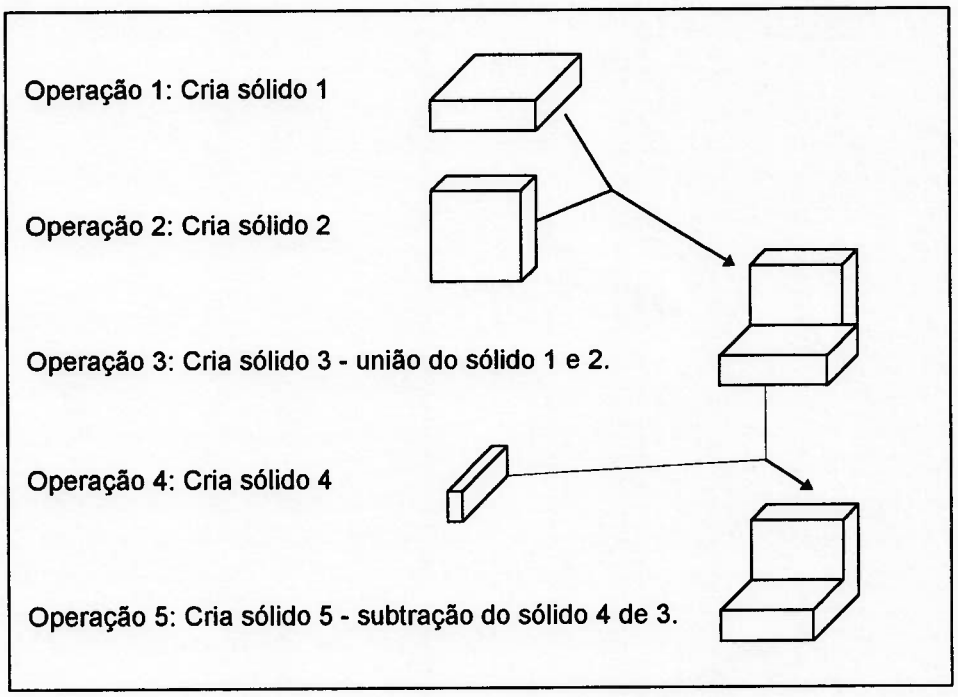

Figura 4.4 - Exemplo de problemas na execução de UNDO em uma interface baseada na representação CSG.

Para contornar este problema, utilizamos um sólido base, cuja identificação é associada ao sólido resultante das operações booleanas. Desta forma, no exemplo da figura 4.4, se o sólido base escolhido fosse o sólido 1, o UNDO da operação 3 não iria

invalidar a operação 5. Para que isto fosse possível, foram criadas operações booleanas que devolvem o resultado em um sólido com o identificador do sólido base.

Uma implicação desta escolha é que o sólido base não pode sofrer nenhuma transformação de rotação ou translação, pois ao se desfazer uma operação booleana anterior a transformação, não será possível verificar se esta operação deve ou não ser re-executada para o sólido que foi adicionado ou subtraído do sólido base, uma vez que nos parâmetros fornecidos desta operação de transformação somente constará o identificador do sólido base. Isto, entretanto não constitui problema, pois estas transformações sempre podem ser aplicadas ao sólido anexado antes da execução da operação booleana. A estrutura genérica em árvore da representação CSG, utilizando um sólido base fica como ilustrado na figura 4.5.

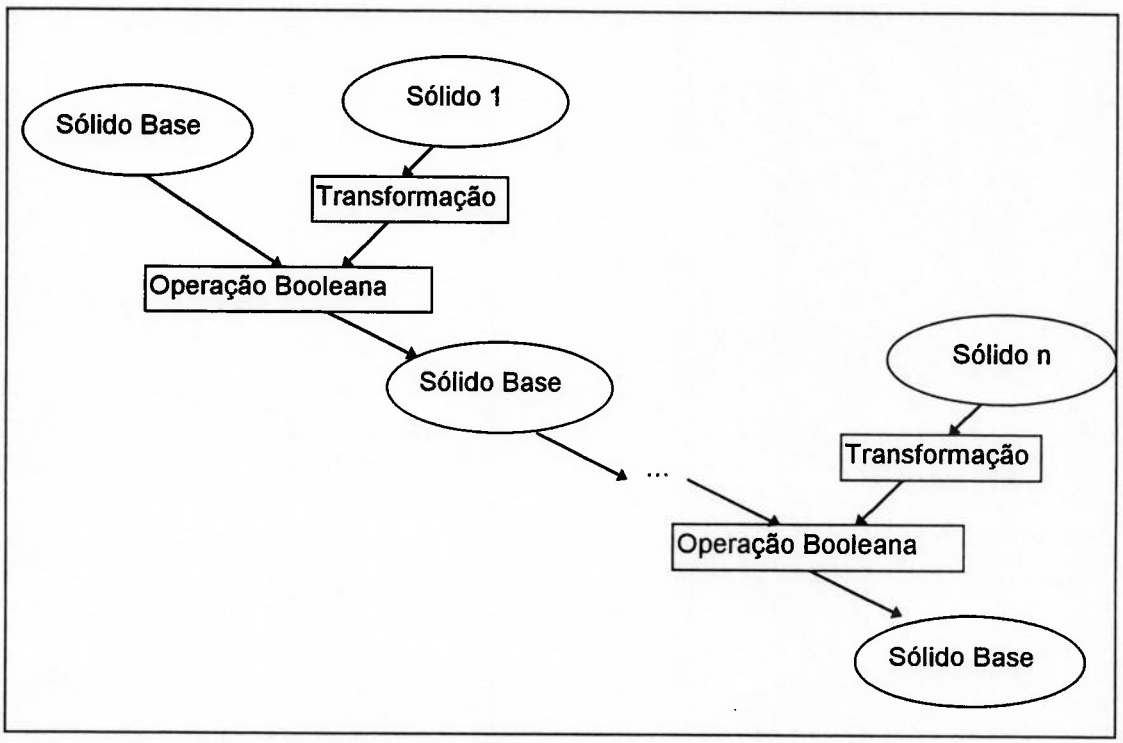

Figura 4.5 - Estrutura em árvore da CSG utilizando um sólido base.

4.4.2 Implementação da Estrutura para Armazenamento da História de Execução

A estrutura implementada para armazenamento da história de execução das operações consiste em apenas uma lista ligada onde cada elemento consiste em uma descrição da operação de alto nível e uma lista ligada de operadores de Euler associados a esta operação. Cada elemento de dados é inserido seqüencialmente à lista à medida que as operações são lidas dos arquivos enviados pelo USPDesigner.

A utilização de listas ligadas foi escolhida porque o número de operações executadas é ilimitado. Basicamente a classe implementada para este sistema disponibiliza funções ao usuário para ler arquivos provenientes do USPDesigner, funções para inserir operações e operadores de Euler nas listas adequadas e recuperar estes dados, através de descrições ou identificadores.

4.4.3 Implementação da estrutura ATMS

A classe implementada para o sistema ATMS manipula uma lista ligada de nós, uma lista de contextos do tipo *nogood* e um contexto corrente. Ela disponibiliza ao sistema: funções para inserir nós, alterar seus rótulos, alterar suas justificativas, manipular o contexto corrente e recuperar qualquer nó através de seu ID ou sua descrição.

5. RESULTADOS EXPERIMENTAIS

Duas situações foram simuladas a fim de verificar o funcionamento do Sistema de Apoio implementado. No primeiro caso, daqui em diante referenciado por caso A, o sólido da figura 5.1 foi criado no USPDesigner. Duas funções foram testadas, a função UNDO, para a operação que cria o furo passante quadrado e a operação REDO, transformando o furo quadrado em um rasgo retangular.

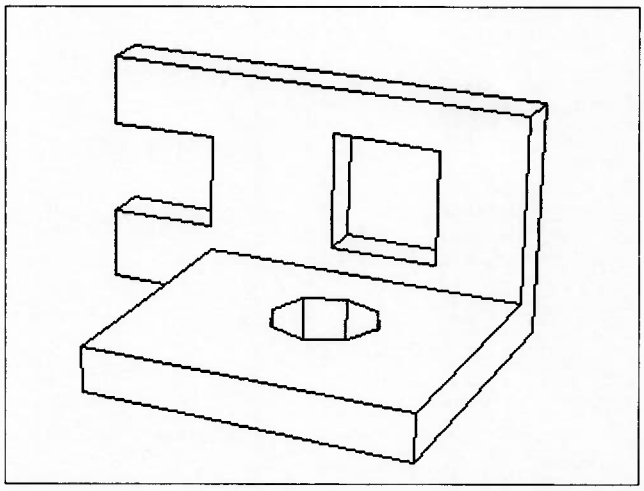

Figura 5.1 - Primeiro Caso simulado.

No segundo caso, caso B, o sólido da figura 5.2 foi criado no USPDesigner. Neste exemplo, primeiramente foi realizada uma operação de UNDO para a operação de criação do cilindro que dá origem ao furo. Outro teste foi realizado, simulando uma mudança no projeto da peça, o aumento do diâmetro do furo em 1 unidade. Para tanto realizamos a operação de REDO para a operação que cria o cilindro que posteriormente

dá origem ao furo cilíndrico. Mais um teste da função REDO foi realizada, na tentiva de substituir o furo cilíndrico por um furo prismático quadrado.

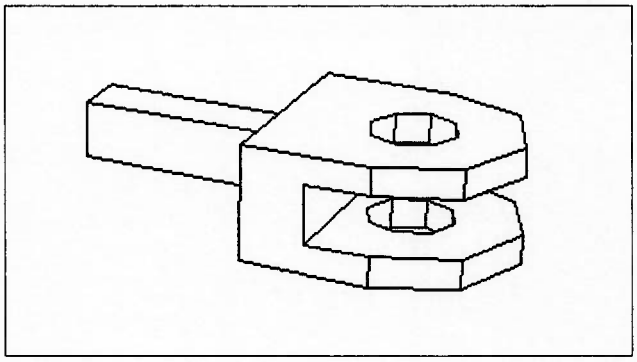

Figura 5.2 - Segundo caso simulado.

5.1 Caso Simulado A

O sólido da Figura 5.1 foi gerado no modelador de sólidos pela execução da seqüência de operações listadas na tabela 5.1:

Tabela 5.1 - Seqüência de operações para criação do sólido da Figura 5.1

Operação 1	idcube 1 2 20 10
Operação 2	idcube 2 15 15 2
Operação 3	trans 2 -1 0 5 0
Operação 4	idunion 1 2 3
Operação 5	idcube 4 2 5 5
Operação 6	trans 4 -1 0 11 3
Operação 7	idminus 1 4 5
Operação 8	idcube 6 2 5 4
Operação 9	trans 6 -1 0 0 3
Operação 10	idcyl 8 8 2.5 2
Operação 11	trans 8 -1 7.5 13 0
Operação 12	idminus 1 8 9
Operação 13	idminus 1 6 7

O resultado das operações 1, 2, 3 e 4 pode ser observado na figura 5.3. O resultado da execução da operação 5 pode ser observado na figura 5.4 e das operações 6 e 7 na figura 5.5. Na figura 5.6 podemos observar o resultado da execução de todas as operações da tabela 5.1.

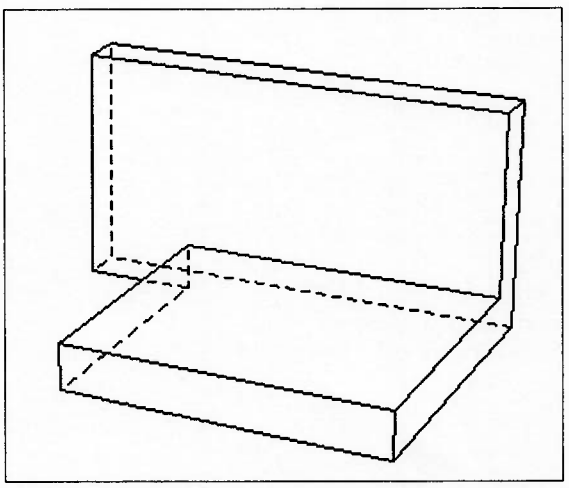

Figura 5.3 - Sólido gerado pelas operações 1, 2, 3 e 4 da Tabela 5.1.

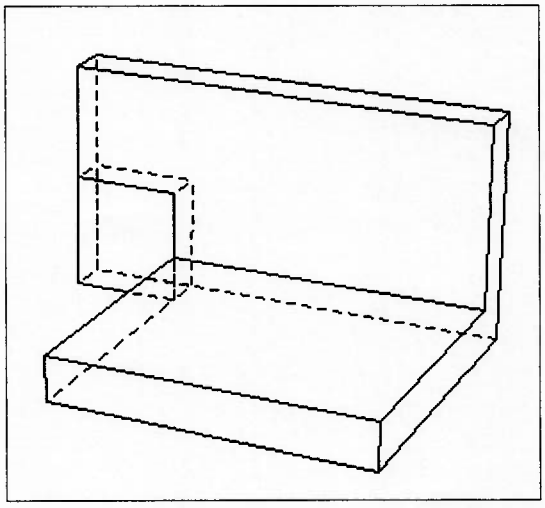

Figura 5.4 - Sólido obtido após a execução da operação 5 (criação do bloco de dimensões 5x5x2).

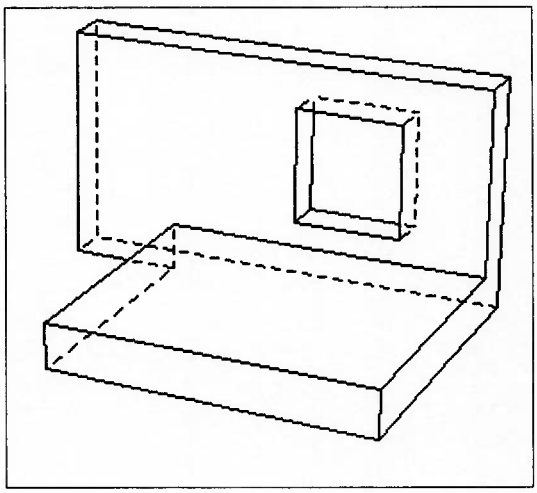

Figura 5.5 - Sólido obtido após a execução das operações 6 e 7 (translação do cubo gerado pela operação 5 e operação booleana de diferença, gerando um furo passante).

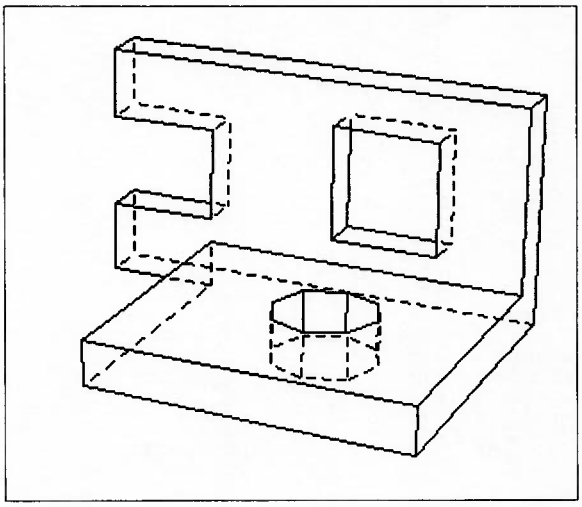

Figura 5.6 - Sólido obtido após a execução de todas as operações de modelagem.

Na figura 5.7, o resultado do UNDO da operação 5 é apresentado, e na figura 5.8, o sólido obtido pela re-execução da operação 5, com dimensões diferentes para o bloco é ilustrado.

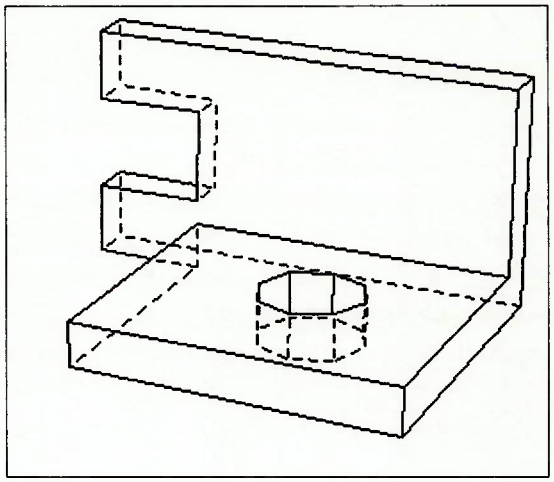

Figura 5.7 - UNDO da operação 5.

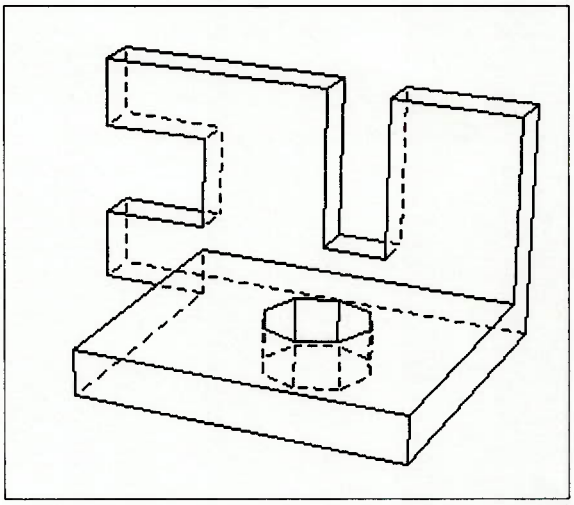

Figura 5.8 - REDO da operação 5.

5.2 Caso Simulado B

O sólido da Figura 5.2 foi gerado no modelador de sólidos pela execução da seqüência de operações listadas na tabela 5.2:

Tabela 5.2 - Seqüência de operações para criação do sólido da Figura 5.2

Operação 1	idcube 1 10 6 5
Operação 2	idcyl 2 8 5 5
Operação 3	trans 2 -1 5 6 0
Operação 4	idunion 1 2 3
Operação 5	idcyl 4 8 2 5
Operação 6	trans 4 -1 5 6 0
Operação 7	idminus 1 4 5
Operação 8	idcube 6 2.5 10 2.5
Operação 9	trans 6 -1 2.5 -10 1.25
Operação 10	idcube 8 10 10 2.5
Operação 11	trans 8 -1 0 3 1.25
Operação 12	idminus 1 8 9

O resultado das operações 1, 2, 3 e 4 pode ser observado na figura 5.9. O resultado da execução da operação 5 pode ser observado na figura 5.10 e das operações 6 e 7 na figura 5.11. O resultado da execução das operações 8 e 9 pode ser verificada na figura 5.12, da operação 10 na figura 5.13, da operação 11 na figura 5.14 e da operação 12 na figura 5.15.

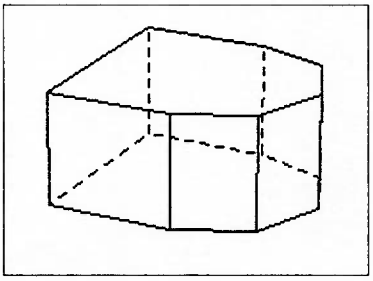

Figura 5.9 - Sólido gerado pelas operações 1, 2, 3 e 4 da Tabela 5.2.

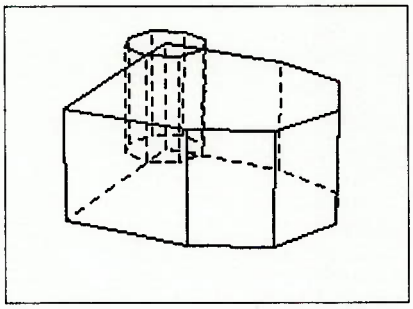

Figura 5.10 - Sólido obtido após a execução da operação 5 (criação do cilindro).

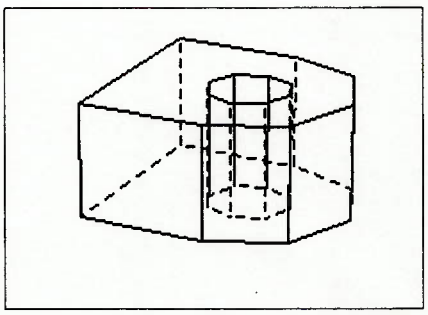

Figura 5.11 - Sólido obtido após a execução das operações 6 e 7 (translação do cilindro gerado pela operação 5 e operação booleana de diferença, gerando um furo passante).

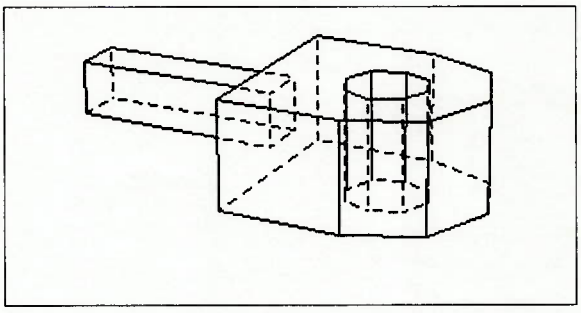

Figura 5.12 - Sólido obtido após a execução das operações 8 e 9 (criação de um bloco e sua translação para a posição adequada).

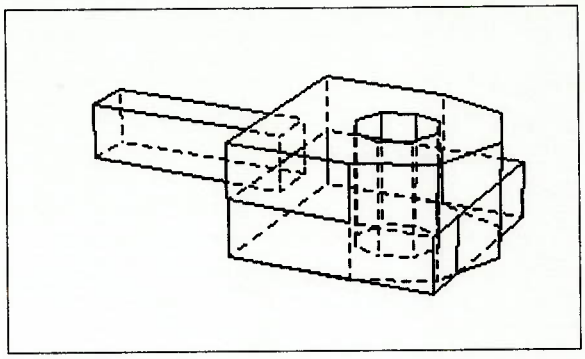

Figura 5.13 - Sólido obtido após a execução da operação 10 (criação de um bloco).

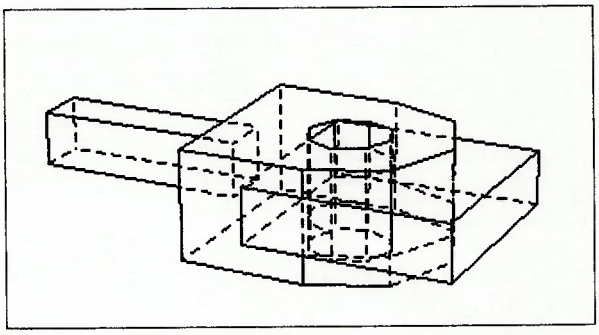

Figura 5.14 - Sólido obtido após a execução da operação 11 (translado do bloco).

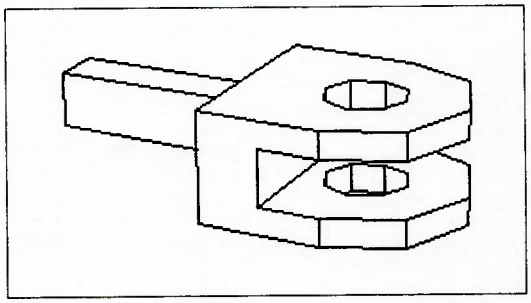

Figura 5.15 - Sólido obtido após a execução da operação 12 (operação booleana de subtração).

Na figura 5.16 é apresentado o resultado do UNDO da operação 5. Na figura 5.17, pode ser observado o resultado do REDO da operação 5 para obtenção de um furo com diâmetro maior. E finalmente, na figura 5.18 o sólido obtido pela execução do REDO da operação 5 para um furo prismático quadrado é ilustrado.

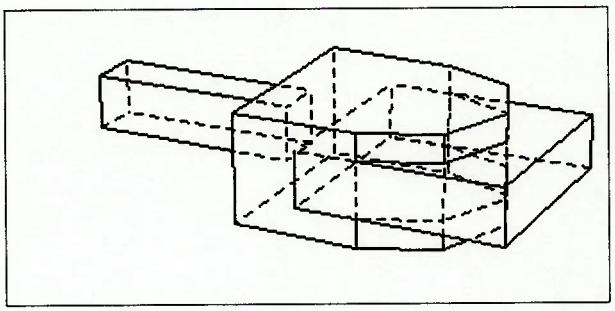

Figura 5.16 - UNDO da operação 5.

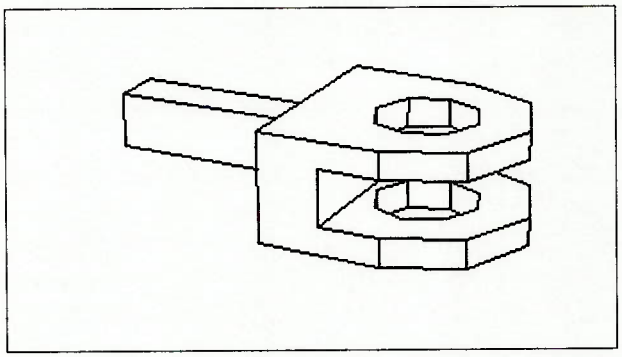

Figura 5.17 - REDO da operação 5 (obtenção de um furo de maior diâmetro).

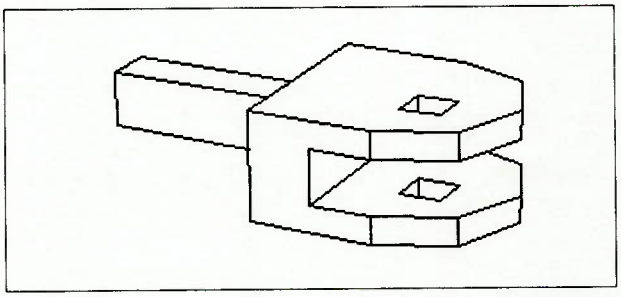

Figura 5.18 - REDO da operação 5 (furo prismático quadrado).

6. DISCUSSÃO

6.1 Caso A

6.1.1 UNDO da Operação 5

Para a realização do UNDO da operação 5, criação de um bloco de dimensões 2x5x5, que posteriormente através de uma operação de conjunto de subtração gera um furo passante quadrado em uma posição adequada do objeto (operações 6 e 7), verificamos que o Sistema de Apoio foi capaz de informar corretamente ao modelador de sólidos sobre quais operações retrair e quais manter, mantendo intactas as operações sobre o sólido que não possuem relação de dependência com as operações retraídas, dando como resultado o sólido exibido na figura 6.1.

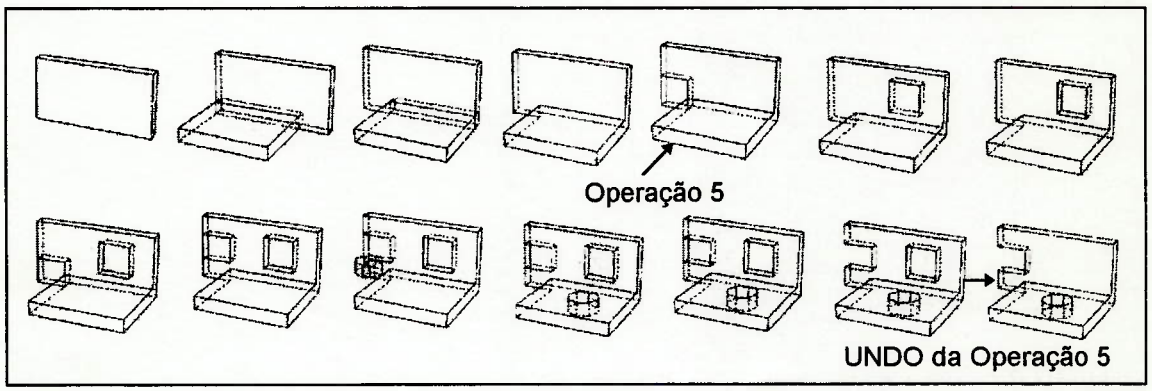

Figura 6.1 - UNDO da operação 5, caso A.

6.1.2. REDO da Operação 5

Para a realização do REDO da operação 5, com outros parâmetros dimensionais, verificamos que o sistema foi capaz de identificar as operações que deveriam ser re-executadas por possuírem algum tipo de dependência com a operação alterada, e o sólido obtido foi o ilustrado na figura 6.2.

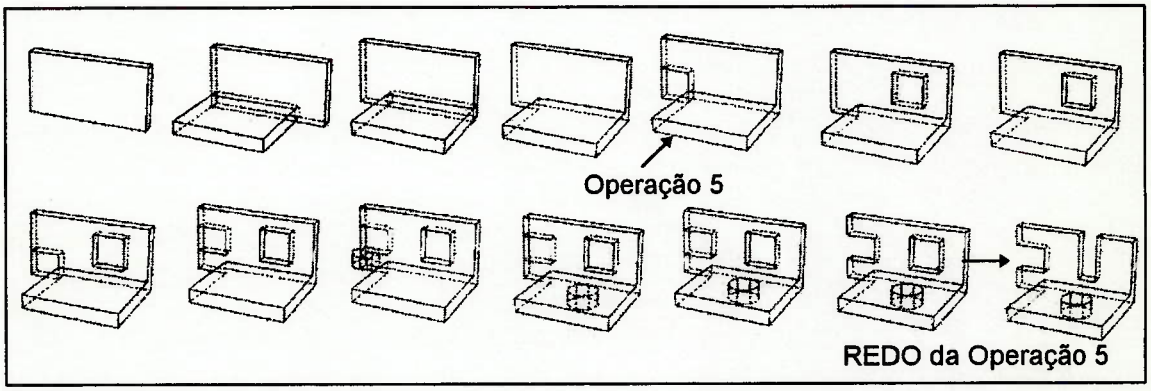

Figura 6.2 - REDO da operação 5, caso A.

6.2 Caso B

6.2.1 UNDO da Operação 5

Neste caso, o UNDO da operação que cria o cilindro que posteriormente gera o furo passante na peça (operação 5, figura 5.10), provoca a retração das operações 6 e 7 (translação e operação booleana que envolvem o cilindro que deixou de existir) e da operação 12, de subtração de um bloco. A operação 12 foi retraída, pois utiliza faces,

arestas e vértices que foram gerados pela operação booleana 7. Por isto, na figura 6.3 verificamos a existência do bloco gerado na operação 10 que é transladado para a posição em que se encontra pela operação 11.

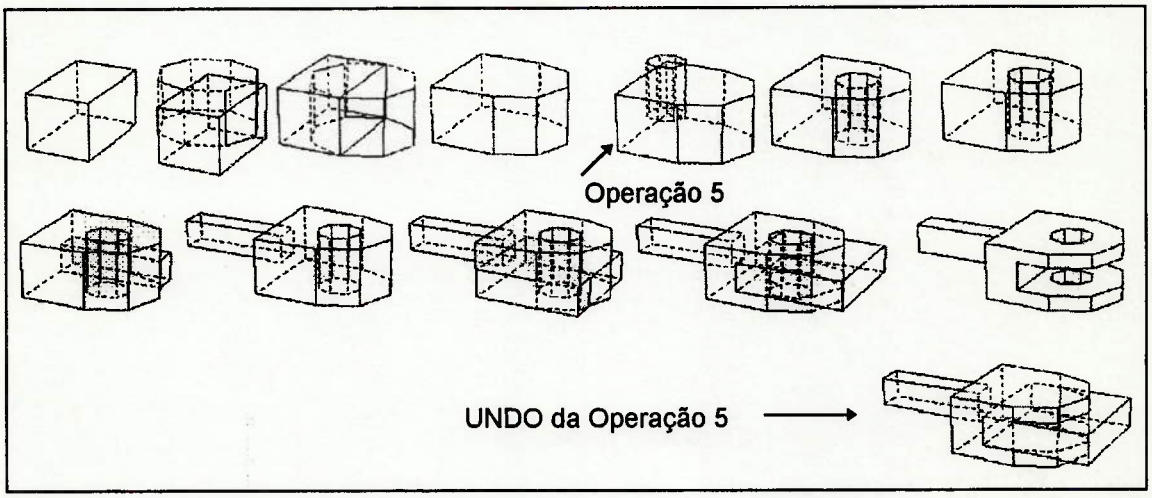

Figura 6.3 - UNDO da operação 5, caso B.

6.2.2 REDO da Operação 5 - Aumento do diâmetro do furo

A operação de REDO da operação 5, criando um cilindro com diâmetro de uma unidade maior que o anterior, gera o sólido apresentado na figura 6.4. Podemos verificar que esta operação foi executada de maneira consistente e que o resultado foi o esperado.

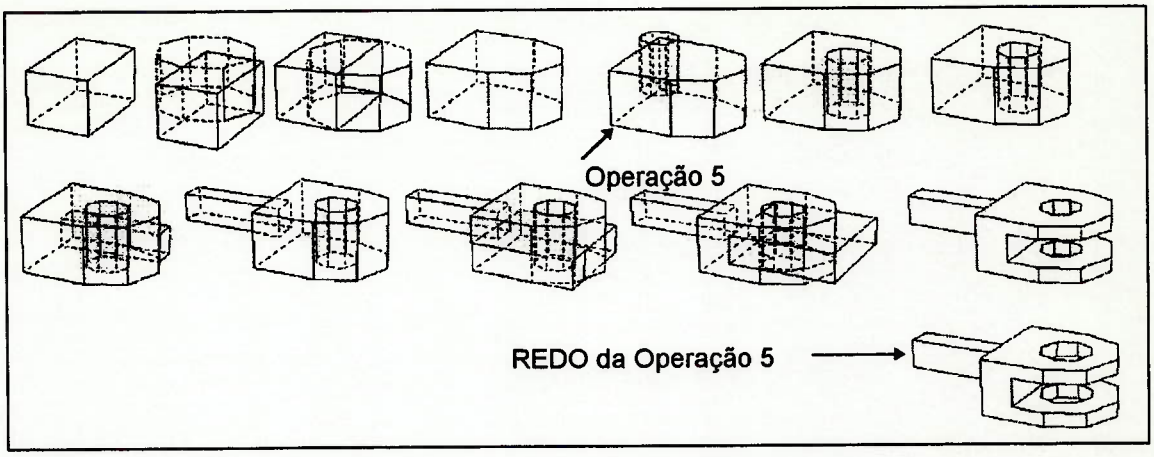

Figura 6.4 - REDO da operação 5, aumento do diâmetro do furo.

6.2.3 REDO da Operação 5 - Mudança para um furo quadrado

A operação de REDO da operação 5, criando um bloco prismático de seção quadrada no lugar de um cilindro gerou o sólido apresentado na figura 6.5. Comparando na figura 6.5 o sólido antes da operação de REDO e depois, verificamos que o furo cilíndrico foi substituído por um furo prismático quadrado como desejado.

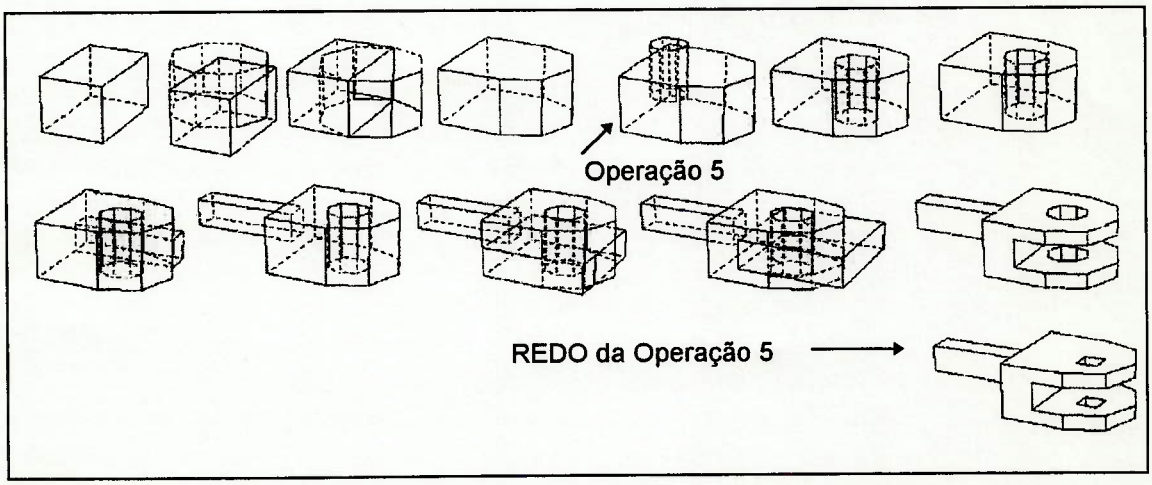

Figura 6.5 - REDO da operação 5, furo prismático quadrado.

7. CONCLUSÕES

Neste trabalho um panorama das tecnologias de sistemas CAD foi apresentado, com ênfase nas metodologias paramétrica e variacional.

O Sistema de Apoio implementado provou ser capaz de fornecer um suporte consistente a funções de UNDO e REDO. A integração com o sistema de modelagem (USPDesigner) se deu de maneira suave, e independente dos algoritmos de cálculo geométrico do modelador. Isto é importante se verificarmos que tanto as operações locais como as globais puderam ser utilizadas.

Alguns problemas foram confrontados, e a utilização de um sólido base provou ser de grande utilidade. Outras limitações do sistema de apoio se devem a pequenos detalhes de implementação, que não foi otimizada, por se tratar ainda de um protótipo. Entre elas temos a necessidade excessiva de espaço em memória, o que limita ainda a utilização deste sistema de apoio a sólidos simples e a integração pouco amigável com o USPDesigner através de arquivos texto, que foi implementada desta forma para isolar os dois sistemas e permitir a avaliação de cada um separadamente.

A escolha pela utilização do ATMS como estrutura para armazenar informações de dependências provou ser acertada, principalmente pela necessidade de se verificar constantemente o estado válido ou inválido de cada operação, além de permitir a possibilidade de trabalhar em diferentes contextos e de verificar a validade de cada contexto. Uma vantagem significativa é que os contextos não precisaram ser gerados combinatorialmente, pois hipóteses iniciais foram assumidas, e outras foram sendo acrescentadas, testando se não introduziam alguma incoerência. Esta conferência no

ATMS é uma tarefa simples, pois basta realizar a comparação com os conjuntos denominados por *nogood*. Por armazenar informações sobre o conjunto de asserções do qual cada nó depende, a verificação da validade dos nós para cada contexto no ATMS também é uma comparação simples de conjuntos.

Os objetivos propostos para este trabalho foram portanto atingidos e para dar continuidade ao que foi realizado, os seguintes tópicos podem ser explorados:

- Definição e implementação de estruturas que mantenham informações de persistência de identificação e propriedades de elementos em um sistema CAD.
- Aprimorar a interface do sistema de suporte com o modelador, de tal forma que este passe a pertencer ao modelador, sem que seja necessário ao usuário verificar quando está trabalhando com comandos de um ou outro sistema.
- Otimizar a utilização de memória do sistema de apoio para que este não seja um fator limitador da complexidade do modelo sólido que pode ser criado e manipulado pelo sistema.
- Utilizar na estrutura em árvore do CSG com utilização de um sólido base (figura 4.5), as dimensões relativas no lugar das transformações, deixando a tarefa de identificação das transformações adequadas para algoritmos computacionais de busca de solução do modelador de sólidos.

ANEXO A - SISTEMAS CAD PARAMÉTRICO E VARIACIONAL

Os termos paramétrico e variacional são utilizados de maneira indistinta no meio comercial. Esta confusão se deve ao fato de sistemas baseados nestes dois métodos serem, do ponto de vista do usuário, muito semelhantes, apesar de serem abordagens diferentes para a solução de um mesmo problema.

Os dois métodos surgiram para lidar com o problema dos projetistas que em etapas iniciais do projeto muitas vezes não possuem uma definição completa e detalhada do produto ou peça em projeto. Esta indefinição se traduz em freqüentes modificações nos modelos geométrico e dimensional do produto, que devem ser suportadas e facilitadas pelo sistema CAD.

Segundo SHAH e MÄNTYLÄ (1995), uma análise mais cuidadosa dos modelos paramétrico e variacional revela que ambos diferem na maneira com que lidam com as restrições e na forma com que as definem. Os sistemas paramétricos encontram soluções para as restrições impostas atribuindo valores para as variáveis do modelo. Estes valores são calculados a partir de valores previamente obtidos pelo sistema. A ordem de cálculo dos valores é definida pelo algoritmo de propagação de restrições. Os sistemas variacionais tratam as restrições de forma diferente, eles resolvem as restrições construindo um sistema de equações que as representam. O sistema de equações obtido é, então, resolvido simultaneamente por métodos numéricos ou outros métodos equivalentes.

A.1 Projeto Paramétrico

Antes de definirmos o que é um projeto paramétrico vamos esclarecer alguns conceitos como: o que é um modelo dirigido por dimensões (*Dimension-driven Model*) e o que é um conjunto de restrições geométricas não acopladas.

Um modelo dirigido por dimensões é aquele definido por um conjunto de dimensões cujos valores podem ser definidos pelo projetista, podendo ser alteradas e, neste caso, o modelo é modificado de modo que a dimensão possua o valor correto. Esta característica é que diferencia os sistemas paramétricos dos sistemas tradicionais nos quais, por exemplo, quando uma linha é desenhada ela não pode ser alterada, mas somente redesenhada.

Restrições geométricas especificam relações entre entidades geométricas, tais como paralelismo, tangência, dimensões lineares e angulares. Estas restrições geométricas podem ser classificadas em dois conjuntos: restrições geométricas acopladas e restrições geométricas não acopladas. As restrições geométricas não acopladas são restrições que definem a geometria do objeto sem que seja necessária a resolução simultânea de restrições, já no caso de restrições geométricas acopladas o conjunto de restrições deve ser tratado simultaneamente para se obter uma solução. No exemplo da Figura A.1 verificamos que no caso (a), se a linha L_1 tiver os seus parâmetros definidos, definimos também os parâmetros de L_2, verificando a restrição de paralelismo D_1 e também podemos definir os pontos P_1 e P_2 satisfazendo a restrição de comprimento D_2. Observa-se que neste caso as restrições podem ser verificadas e satisfeitas em qualquer ordem, por isto este conjunto de restrições é chamado de conjunto de restrições não

acopladas. O caso (b) da mesma figura ilustra restrições acopladas, isto é, não podemos definir os comprimentos B e C a partir de A de maneira independente.

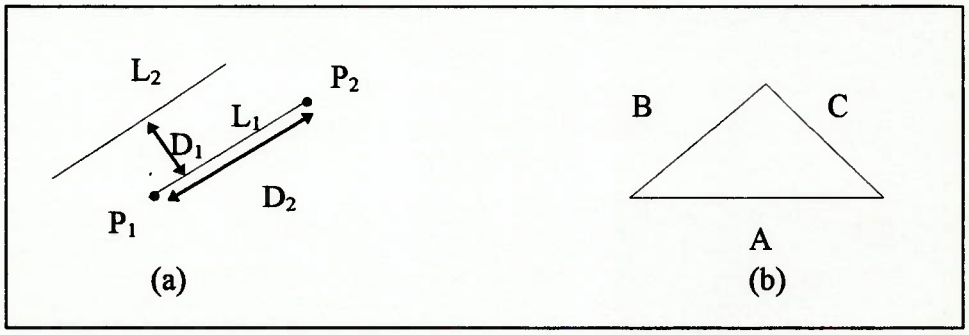

Figura A.1 - Restrições acopladas (b) e não acopladas (a)

SINGH (1986) define o projeto paramétrico como uma metodologia que utiliza técnicas de busca e solução baseadas em casos especiais para se obter um modelo dirigido por dimensões que é aplicada principalmente para manipulação de restrições geométricas não acopladas.

A.2 Projeto Variacional

No projeto variacional, a geometria, as equações de engenharia e o esquema dimensional do projeto são todos tratados como restrições. Os sistemas baseados nesta metodologia, por resolverem simultaneamente as equações geradas a partir do conjunto de restrições, e por estas poderem ser acopladas arbitrariamente, conseguem obter uma configuração de projeto que satisfaz todos os critérios de projeto. Pela utilização do método variacional é possível resolver os problemas de projeto em que o método

paramétrico é bem sucedido, entretanto o inverso não é verdadeiro, sendo portanto o método variacional mais geral que o paramétrico.

As restrições e equações de engenharia no projeto variacional são representadas na forma de redes de restrições. Utilizando teoria de grafos, uma grande rede de restrições pode ser decomposta em pequenos conjuntos de equações simultâneas de forma que as redes possam ser resolvidas eficientemente. Esta decomposição é realizada pela identificação de subconjuntos independentes de restrições acopladas (SINGH, 1996).

A.3 Comparação entre os Projetos Paramétrico e Variacional

Para ilustrar a diferença entre estes dois métodos, apresentamos um exemplo retirado de Singh (1996) como descrito na figura A.2.

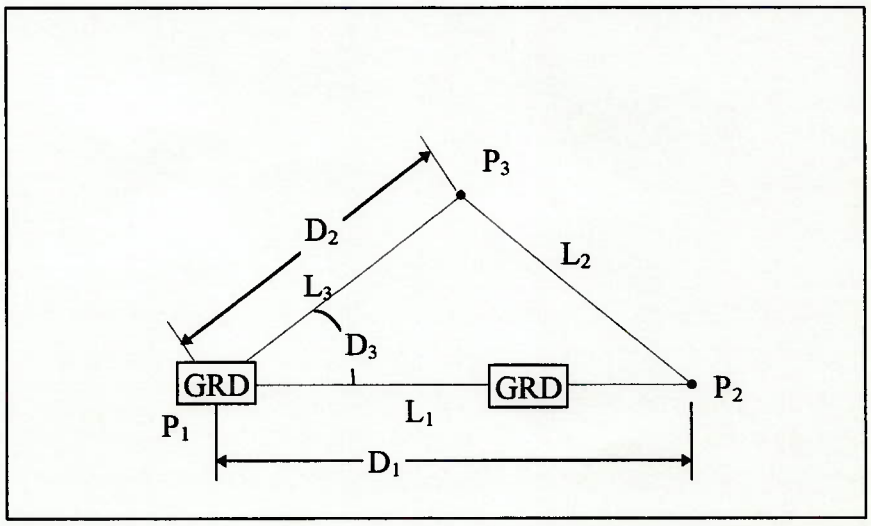

Figura A.2 - Exemplo do triângulo de Chung e Schussel. (SINGH, 1996)

No caso do método variacional, todas as variáveis simbólicas necessárias para definir completamente a geometria são geradas automaticamente. Estas variáveis ser relacionadas por equações de engenharia e equações que expressam restrições geométricas. O conjunto de restrições define o projeto. Quando uma restrição ou equação é especificada, o sistema gera internamente equações matemáticas relacionando as variáveis simbólicas. Para o triângulo do exemplo, o conjunto de equações de restrições gerada pelo sistema seria:

$$\sigma_3 = \sigma_1 + D_3 + 180 \qquad \text{(A.1)}$$

$$\sqrt{(X_1 - X_3)^2 + (Y_1 - Y_3)^2} = D_2 \qquad \text{(A.2)}$$

$$\sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} = D_1 \qquad \text{(A.3)}$$

$$Y_1 \cdot \cos(\sigma_3) - X_1 \cdot \text{seno}(\sigma_3) + DO_3 = 0.0 \qquad \text{(A.4)}$$

$$Y_3 \cdot \cos(\sigma_3) - X_3 \cdot \text{seno}(\sigma_3) + DO_3 = 0.0 \qquad \text{(A.5)}$$

$$Y_3 \cdot \cos(\sigma_2) - X_3 \cdot \text{seno}(\sigma_2) + DO_2 = 0.0 \qquad \text{(A.6)}$$

$$Y_2 \cdot \cos(\sigma_2) - X_2 \cdot \text{seno}(\sigma_2) + DO_2 = 0.0 \qquad \text{(A.7)}$$

$$\sigma_1 = 0.0 \qquad \text{(A.8)}$$

$$Y_2 \cdot \cos(\sigma_1) - X_2 \cdot \text{seno}(\sigma_1) + DO_1 = 0.0 \qquad \text{(A.9)}$$

$$Y_1 \cdot \cos(\sigma_1) - X_1 \cdot \text{seno}(\sigma_1) + DO_1 = 0.0 \qquad \text{(A.10)}$$

onde σ_1 é o ângulo formado entre GRD e L_1, σ_2 é o ângulo formado entre GRD e L_2 e σ_3 é o ângulo formado entre GRD e L_3. DO_1, DO_2 e DO_3 são as componentes constantes das equações de L_1, L_2 e L_3, que estão na forma:

$$Y_i \cdot \cos(\sigma_i) = X_i \cdot \text{sen}(\sigma_i) + DO_i \qquad \text{(A.11)}$$

Estas restrições são geradas automaticamente baseadas na entrada de dados e na intenção do projetista. Por exemplo, a equação (A.8) é uma equação gerada automaticamente onde a igualdade reflete a interpretação de que a linha L_1 é horizontal.

Há um mecanismo de solução genérica não linear de equações simultâneas em sistemas de CAD/CAM baseados no método variacional. Depois da geração das equações a partir das restrições, as soluções para todas as variáveis são obtidas pela resolução do conjunto de equações segundo o mecanismo de solução. Técnicas de decomposição são utilizadas para aumentar a eficiência do processo de solução. A seqüência de solução é a seguinte: σ_1 é encontrado pela equação (A.8), os valores de σ_3 (equação A.1) e DO_1 (equações A.9 e A.10) podem ser então calculados. Por fim, os valores de DO_3 (equações A.4 e A.5), DO_2 e σ_2 (equações A.6 e A.7) também são obtidos.

Utilizando o método paramétrico o problema é abordado de forma diferente. No método paramétrico a geometria global do projeto é vista como uma combinação de vários casos especiais. Por exemplo, o ponto P_1 é definido como GRD, isto é, este ponto é uma referência, um caso geométrico especial, podendo ser chamado de um ponto fixo. L_1 é outro caso especial, sendo definida pelos pontos P_1 e P_2. Pela solução de casos especiais como estes, os valores de todas as variáveis podem ser obtidos. Entretanto há muitos casos especiais e a combinação entre eles cresce combinatorialmente. Por isto, problemas podem ser encontrados devido ao aumento de combinações.

De maneira bastante simples podemos concluir que o projeto paramétrico é baseado em casos especiais da geometria analítica e o projeto variacional é baseado em teorias de grafos e análise numérica.

Segundo SHAH e MÄNTYLÄ (1995), por serem baseados na satisfação explícita e seqüencial de restrições, os modelos paramétricos podem ser instanciados rapidamente. Entretanto, sistemas que utilizam modelos paramétricos não são capazes de lidar com restrições mutuamente acopladas. Por outro lado, os modelos variacionais que utilizam técnicas implícitas de satisfação de restrições podem lidar com restrições acopladas, mas são lentos no processo de reavaliação de uma nova instância do modelo e possuem limitações maiores na manipulação de modelos especificados de maneira incompleta.

REFERÊNCIAS BIBLIOGRÁFICAS

ALDEFELD, B. Variation of Geometries based on a Geometric-reasoning Method. **Computer Aided Design**, v.20, n.3, p.117-126, Abr. 1988.

BAUMGART, B.G. A Polyhedron Representation for Computer Vision. **AFIPS Proc.**, v.44, p 589 - 596, AFIP Press, Arlington, Va, 1975.

CUGINI, U. Capturing Context Dependent Rules from Interaction Sequences: An Example for Mechanical Design. In: SATA, T., ed. **Organization of Engineering Knowledge for Product Modelling in Computer Integrated Manufacturing**. p.385-410, Elsevier, Amsterdã, 1989.

DE KLEER, J. An Assumption-based TMS. **Artificial Intelligence,** v.28, n.2, p.127-162, 1986(a).

DE KLEER, J. Extending the ATMS. **Artificial Intelligence,** v.28, n.2, p.163-196, 1986(b).

DE KLEER, J. Problem Solving with the ATMS. **Artificial Intelligence,** v.28, n.2, p.197-224, 1986(c).

DOYLE, J. A Truth Maintenance System. **Artificial Intelligence,** v.12, n.3, p.231-272, 1979.

FITZGERALD, W.J. Using Axial Dimensions to Determine the Proportions of Line Drawings in Computer Graphics. **Computer Aided Design,** v.13, n.6, p.377-382, Nov. 1981.

HILLYARD, R.C. BRAID, I.C. Analysis of Dimensions and Tolerances in Computer-aided Mechanical Design. **Computer Aided Design**, v.10, n.3, p.161-166, Mai. 1978.

HILLYARD, R.C. BRAID, I,C. Characterizing Non-ideal Shapes in Terms of Dimensions and Tolerances. **Computer Graphics**, v.12, n.3, p234-238, 1978.

INUI, M., KIMURA, F. Representation and Manipulation of Design and Manufacturing Processes by Data Dependency. In: H., Yoshikawa et al. eds.,.**Intelligent CAD II**. Elsevier Science Publishers B.V. (North-Holland) IFIP, 1990.

INUI, M., KIMURA, F. Using a Truth-maintenace System to Assist Product-model Construction for Design and Process Planning. **Computer Aided Design,** v.5, n.1, p59-70, Jan. 1993.

KIMURA, F. SUZUKI, H. WINGÄRD L. A Uniform Approcah to Dimensioning and Tolerancing in Product Modeling. In: K., Bo et al. eds. **Computer Aided Production Engineering**. p.165-178, Amsterdam, North Holland, 1987.

LIGHT, R. GOSSARD, D. Modification of Geometric Models through Variational Geometry. **Computer Aided Design**, v.14 n.4, p.209-214, Jul. 1982.

MÄNTYLÄ, M. **An Introduction to Solid Modeling**. Maryland, Computer Science Press, 1988.

MEDLAND, A.J. **The computer-based Design Process**. London, Chapman&Hall, 1992.

MORRIS, P.H. NADO, R.A. Representing Actions with an Assumption-based Truth Maintenance System. **Proc. 5th National Conference Artificial Intelligence**, p.13-17, Philadelphia, PA, USA, 1986.

QIU, Z.Y., HUANG, Y. Research on a Dimensional-Driven Variational Design System. **Proc. Of the 6th ICECGDG**. Tokyo, 1994.

ROLLER, D.; SCHONEK, F.; VERROUST, A. Dimension-driven Geometry in CAD: A Survey. **Theory and Practice of Geometric Modelling**. Springer-Verlag, 1989. P509-523.

SHAH, J.J., MÄNTYLÄ, M. **Parametric and Feature-based CAD/CAM**. John Wiley & Sons, USA. 1995.

SINGH, N. **Systems Approach to Computer-integrated Design and Manufacturing**. John Wiley & Sons, USA. 1996.

SUTHERLAND, I.E. Sketchpad a Man-machine Graphical Communication System. In **Proc. of 23rd SJCC**, p.329-346, 1963.

SUZUKI, H. KIMURA, F. SATA, T. Treatment of Dimensions on Product Modeling Concept. **International Symposium on Design and Synthesis**. p.619-624, Tokyo, Jul. 11-13, 1984.

SUZUKI, H. On Representing Product Information in CAD - Present Status and Future Directions. **Proc. of the 6th ICEEGDG**. Tokyo, 1994.

SUZUKI, H. KASE, K. KIMURA, F. Physically Based Modelling for Evaluating Shape Variations. **Proceedings of the 4th CIRP Seminar on Computer Aided Tolerancing**. p.173 - 180, University of Tokyo, Tokyo, Japan. Abr.5-6, 1995.

TSUZUKI, M.S.G. Miyagi, P.E. Moscato, L.A. Modelagem de Sólidos: Representação por Fronteira (B-rep). **XI Congresso Brasileiro de Engenharia Mecânica**, p.611-614, São Paulo, Dez. 1991.

TSUZUKI, M.S.G. **Contribuição para a Representação de "Features" Paramétricas Aplicada a Sistemas CAD/CAE/CAM**. São Paulo, 1994. 85p. Tese (Doutorado) - Escola Politécnica, Universidade de São Paulo.

TSUZUKI, M.S.G. MIYAGI, P.E. MOSCATO, L.A. Representing Dimensions and Features in a Product Model. In Computer Applications in Production Engineering, **Proceedings of CAPE'95**, p.52-61, IFIP, Chapman & Hall, 1995.

WEILER, K. Edge-based Data Structures for Solid Modeling in Curved-Surface Environments. **IEEE Computer Graphics & Applications**, v.1,n.5, p.21-39, Jan. 1985.

WILSON, P.R. Euler Formulas and Geometric Modeling. **IEEE Computer Graphics & Applications,** v.5, n.8, p.24-36, Ago. 1985.

WINGÅRD, L. **Introducing Form Features in Product Models. A Step Towards CADCAM with Engineering Terminology**. Stockolm, 1991. 87p. Licentiate Thesis - The Royal Institute of Technology.

ZEID, I. **CAD/CAM Theory and Practice**. International Edition, McGraw-Hill, Singapore, 1991.