UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS- GRADUAÇÃO EM ENGENHARIA MECÂNICA

Modelamento Baseado em Features em um Conceito de Projeto para Fabricação e Montagem

DISSERTAÇÃO SUBMETIDA À UNIVERSIDADE FEDERAL DE SANTA CATARINA PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA MECÂNICA

Celson Pantoja Lima

FLORIANÓPOLIS, SETEMBRO DE 1994.

MODELAMENTO BASEADO EM FEATURES EM UM CONCEITO DE PROJETO PARA FABRICAÇÃO E MONTAGEM

Celson Pantoja Lima

ESTA DISSERTAÇÃO FOI JULGADA PARA OBTENÇÃO DO TÍTULO DE

MESTRE EM ENGENHARIA

ESPECIALIDADE ENGENHARIA MECÂNICA, ÁREA DE CONCENTRAÇÃO EM FABRICAÇÃO E APROVADA EM SUA FORMA FINAL PELO PROGRAMA DE PÓS- GRADUAÇÃO EM ENGENHARIA MECÂNICA, DA UNIVERSIDADE FEDERAL DE SANTA CATARINA.

Prof. Áureo Campos Ferreira, Ph. D - Orientador

Prof. Dr. J. J. Pamies Teixeira - Co-orientador

Prof. Antonio Fabio C. da Silva, Dr. - Coordenador do Curso

BANCA EXAMINADORA

Prof. Áureo Campos Ferreira, Ph.D - Presidente

Prof. Dr. J. J. Pamies Teixeira

Prof. João Carlos Espíndola Ferreira, Ph. D.

Bel. Norton Paim Moreira, M. Eng.

À Sra. Oscarina Pantoja Lima

minha mãe.

AGRADECIMENTOS

A famosa página dos agradecimentos. Quando o momento de escrevê-la é instanciado, significa que muitas pessoas e muitos outros momentos impulsionadores e catalizadores de todo esse trabalho já existiram, já aconteceram, já dele fazem parte sem que seja necessário citá-las nominalmente.

Não obstante essa não-necessidade, é impossível deixar que pelo menos as pessoas mais significativas para a conclusão dessa etapa tomem disso conhecimento, como a forma por mim escolhida para expressar minha sincera gratidão por tantas experiências boas.

A minha mãe Oscarina Lima e aos meus irmãos Samuca, Zu, Graça e Eró, que com certeza fazem parte de todas as etapas que serviram de preparação para mais esta, um profundo beijo de *valeu a força!* E sem vocês nada disso teria sido possível.

Ao prof. Pamies, valeu também a confiança em mim depositada durante toda a nossa mais-do-que-longa convivência (afinal, um ano saiu por dois!). Pelas discussões, pelas explicações e pela oportunidade, um grande abraço.

Ao prof. Áureo, que desde muito incentiva e acredita nas pessoas que com ele têm trabalhado. Valeu a oportunidade de concretizar mais essa experiência.

Ao meu querido amigo, agora mais do que nunca, Ricardo *Kadu* Rabelo, pela sua presença e sensatez, seus conselhos, e pelo seu trabalho como instrumento do cosmos e do holismo (e de revisor também, já agora!) ajudando-me às vezes a não piorar tanto; mostrando-me que existe sim um caminho (tortuoso e espinhento) que pode nos levar a uma etapa superior do processo maior, aquela do **PEACE & LOVE**; ensinando-me que existe virtuosismo em quem cala no tumulto como em quem fala ao silêncio; ensinando-me que o *yang sem o yin* não existe; e que o único processo imutável é o da própria mutação. Por todos os momentos **BONS** desses dois últimos anos, um grande e profundo abraço. *Valeu, irmão!*

Ao meu irmão Marcelo *Parceiro* Tavares, que dividiu comigo mais uma estação de chegada e partida nas nossas andanças pelo mundo. Pela catalização do processo de *vinda* (ou de *ida*, depende do referencial!), um grande e profundo abraço.

Dos amigos novos (que não são poucos, felizmente!) , dois não poderiam ser esquecidos : um, pela sua simplicidade e sorriso de criança mesmo já despontando na casa dos cinquentões, meu querido e campeiro amigo, Dr. Simão *Indio Velho* Toscani ; o outro, pelos longos papos e vinhos do porto (mais vinho que papo, obviamente!), o *gigantesco* amigo Adalberto Márcio Nogueira. A eles, meu profundo agradecimento por todas as imagens portuguesas das quais eles foram protagonistas.

SUMÁRIO

Índice das Figuras

Capítulo 1

Capítulo 2

Capítulo 3

Capítulo 4

Capítulo 5

GLOSSÁRIO

Todas as siglas e palavras de língua inglesa usadas no texto encontram-se descritas nesse glossário.

ACAD: Nome comercial do CAD adotado como suporte à implementação do FM, da *AutoDesk*.

AME: Advanced Modeling Extension. Modelador de Sólidos pertencente ao ACAD.

CAD: Desenho Auxiliado por Computador (Computer Aided Design).

CAM: Manufatura Auxiliada por Computador (Computer Aided Manufacturing). Nesse caso, *manufatura* está ligada especificamente ao processo de fabricação de um produto.

CAM-I: Computer Aided Manufacturing - International.

CAPP: Planejamento do Processo Auxiliado por Computador (Computer Aided Process Planning).

CIM: Manufatura Integrada por Computador (Computer Integrated Manufacturing). Aqui, *manufatura* relaciona-se ao processo completo de produção, desde a concepção até a entrega de um produto.

CSG/B-Rep: Construtive Solid Geometry e Boundary Representation, dois modelos de representação de primitivas sólidas.

DFA: Projeto para Montagem (Design For Assembly).

DFM: Projeto para Fabricação (Design For Manufacture).

DOS: Disk Operating System. Sistema operacional para computadores linha PC.

EXPRESS: linguagem orientada por objetos para definições de modelos de informações no PDES/STEP

Feature: traduzida em alguns trabalhos publicados por autores brasileiros como *característica* ou *atributo*, neste trabalho é utilizada em sua forma nativa motivado pelo fato de não existirem palavras ou palavra da língua portuguesa, que expressem exatamente o que significa *feature* no contexto de engenharia.

FFIM: Modelo de Informação de Features de Forma (Form Feature Information Model), desenvolvido pelo comitê PDES nos EUA.

FM: Modelador de Features.

IC: Intelligence Compiler. Sistema computacional de suporte à implementação de um dos módulos existentes no modelo conceituai ao qual pertence este trabalho, chamado *Módulo Abstrato*.

ISO: International Standard Organization.

NC: Comando Numérico (Numeric Control).

PDES: Especificação de troca de dados de produtos (**P**roduct **D**ata **E**xchange **S**pecification).

STEP: **P**adrão para troca de dados de produtos (**ST**andard for **E**xchange **P**roduct **D**ata).

Wireframe: forma de representação de primitivas geométricas sólidas, onde somente as arestas delineadoras dessas primitivas são desenhadas.

Resumo

A atividade de projetar é um processo de criação de peças (mecânicas) segundo requisitos e especificações funcionais, sob certas restrições. As ferramentas computacionais de auxílio a essa atividade, que sempre trataram as informações de natureza puramente geométrica dos projetos, caminham rumo ao modelamento baseado em features. A ideia primordial da utilização da tecnologia de features é a representação e manipulação de informações que representem conhecimentos relativos a diversas fases envolvidas na produção de um produto (planejamento do processo, fabricação, montagem, etc), de tal maneira que as análises, correções e melhoramentos prévios possam ser feitos.

Este trabalho apresenta um modelo conceitual que suporta as principais informações envolvidas na criação de um produto, além de oferecer um tratamento apriorístico dessas informações. A implementação desse modelo foi dividida em dois sub-sistemas : Gráfico e Abstrato. O sub-sistema Gráfico, objeto dessa dissertação, é responsavel pela entrada e visualização de todas as informações (geométricas, tecnológicas, de processo) pertencentes ao produto; o módulo Abstrato faz, sobre os produtos, inferências e raciocínios pertinentes a um grupo específico de aplicações (Seleção de Processos de Fabricação, Verificação de Projeto, Avaliação de Manufaturabilidade e Tecnologia de Grupo).

O projeto foi desenvolvido no Instituto UNINOVA, da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa - Portugal, no âmbito do programa da União Européia denominado BRITE.

Abstract

The design activity is a process of creating parts according to functional requirements and specifications, under a set of constraints. The available computer tools to aid this activity usually deal with product geometric informations. These tools are actually being improved towards the feature based modelling. The main idea in the features technology is the representation and manipulation of information describing knowledge about several areas involved in the production process of a product (process planning, manufacturing, assembly, etc.), in such a way that analysis and improvements can be done "a priori".

This work introduces a conceptual model handling the main informations involved in the creation process of a product. Further, it offers a "a priori" treatment of these informations. The model implementation was divided in two subsystems: Graphical and Abstract. The graphical subsystem, explored in this work, is responsible by the input process as well as necessary to show all informations (geometric, technological, etc.) related to a product; the abstract subsystem works in the products doing inference and reasoning related to a specific applications group (manufacture process selection, project validation, manufacture analysis and group technology).

This work was developed at UNINOVA-FCT, New University of Lisbon, Portugal, sponsored by the European BRITE program.

Capítulo 1

Enquadramento do Problema

O projeto é um dos objetivos essenciais da engenharia como forma de criar e conceber novos produtos, processos, sistemas de fabricação e organizações. Desde a sua gênese, o homem sempre construiu, concebeu, projetou coisas como suas ferramentas básicas de caça e suas vestimentas protetoras. Retratam-se grandes avanços desde os nossos primórdios como seres criativos até os dias atuais, passando pelos grandes gênios inventivos que sempre serviram de catalisadores e incentivadores da divulgação do processo criativo, os quais não servem como regra geral de comportamento e evolução desse processo.

No período de grande avanço tecnológico em que nos encontramos, notamos que esse processo criativo, que designamos de *projeto*, é tratado como uma atividade subjetiva e sem formalização, mais dependente da habilidade, talento e experiência dos projetistas envolvidos do que em técnicas e metodologias formais. Muitas ferramentas e conceitos surgiram com o objetivo de auxiliar os engenheiros envolvidos em tal atividade, os quais tentam entender o processo criativo e definir técnicas e métodos de trabalho. Porém normalmente essas técnicas pertencem a campos e/ou problemas específicos. Uma abordagem relativamente nova e pouco explorada é a ***Teoria axiomática do Projeto*** [Suh90], a qual tenta trazer o projeto para um âmbito mais científico, baseado em axiomas e teoremas que formalizam o processo de projetar.

Independente de metodologias e técnicas usadas no projeto de um produto, e sem considerar se esse projeto é para a produção de uma única peça ou de um produto composto de várias peças (um automóvel, por exemplo), qualquer produto possui um ciclo de vida que pode ser caracterizado nas seguintes fases [BRITE94]:

- Planejamento/desenvolvimento de Especificações
- Projeto Conceitual
- Projeto do produto
- Produção
- Desempenho
- Eliminação e/ou Reciclagem do produto

A 1ª fase tem por objetivo desenvolver uma diretriz ciara dos requisitos do produto em termos de performance, avaliação de tempo, recursos a investir, e outras especificações. Algumas vezes essa fase é chamada de pré-concepção, visando enfatizar a necessidade do entendimento completo do problema antes do desenvolvimento dos conceitos da solução. Em outras palavras, dá-se o reconhecimento das necessidades do produto.

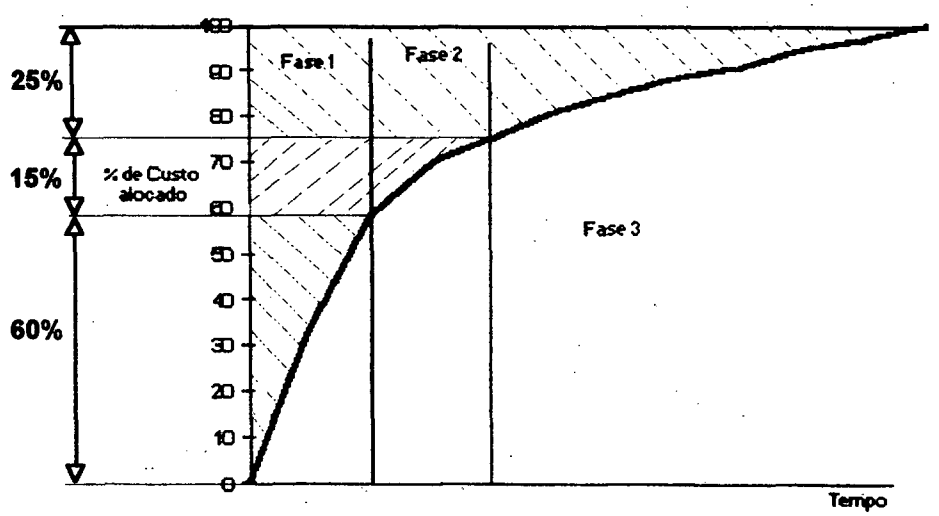
Na 2ª fase é desenvolvida uma idéia imprecisa sobre a funcionalidade e aspecto do produto. Tipicamente os projetos conceituais são representados através de esboços imprecisos e notas. Essa fase do processo de projeto tem sido a menos gerenciada, a menos documentada e a menos entendida. Os projetistas tendem a fazer um esboço conceitual rápido, e então gastar muito tempo no projeto do produto tentando fazer o conceito funcionar.

A 3ª fase é a que consome mais tempo dentro do processo de projeto. Ela começa com um conceito e acaba com um produto pronto-para-fabricar. Tradicionalmente, engenheiros de projeto completam seus trabalhos e passam seus desenhos e notas para os engenheiros de fabricação, os quais decidem então como manufaturar o produto. Existe pois uma comunicação pequena entre os engenheiros dessas duas áreas, o que não tem-se mostrado uma prática muito interessante, pois o projetista, com conhecimento limitado de processos de fabricação, gerará frequentemente produtos que serão difíceis de produzir e caros; os engenheiros de fabricação então frequentemente alteram os componentes para facilitar manufatura e montagem, sem considerar os impactos dessas alterações na operação do produto. A fraqueza dessa abordagem trouxe a *Engenharia Simultânea* ao produto e à produção, uma filosofia que encara o processo de projeto como um esforço de equipe.

De uma breve análise do ciclo-de-vida de um produto, a partir de uma ótica global, percebe-se claramente que as fases correspondentes ao projeto propriamente dito carregam em si a responsabilidade da qualidade de trabalho das fases subsequentes em todos os aspectos, ou seja, as decisões tomadas durante a fase de projeto refletem-se pela vida inteira do produto, seja a nível de custo, tempo e qualidade de fabricação do mesmo, seja a nível de sua funcionalidade no momento de utilização real pelo cliente. Suh [Suh90] estima que de 70% a 80% da produtividade de fabricação é determinada durante a fase de projeto.

Sem considerar que o produto sendo projetado é um sistema inteiro ou somente uma pequena parte de um produto maior, tanto o usuário como o administrador o quererão mais barato (custos de produção reduzidos), melhor (garantia de qualidade) e mais rápido (produção ágil e veloz). Isso significa que as três medidas do processo de projeto são *custo, qualidade e tempo*.

O projeto é tão determinante no custo do produto que tem sido prática corrente das empresas um investimento crescente nesta atividade. A título ilustrativo a figura 1.1 mostra esta dependência em função do tempo de evolução da vida do produto.



Fase 1 - Desenvolvimento das Especificações
Fase 2 - Projeto Conceitual
Fase 3 - Projeto do Produto

fig. 1.1 - Custo e Influência do Projeto[BRITE94]

Outro aspecto importante do projeto é a sua relação com a qualidade de um produto. É bem conhecida a tendência para uma maior exigência de desempenho, o que obriga a que a qualidade esteja contida no próprio produto, isto é, ao criar-se o produto, ele deve ser concebido de forma que, uma vez colocado no mercado, ele responda às exigências do seu consumidor. Este tem sido sem dúvida a ofensiva oriental, designadamente do Japão.

Além de afetar custos e qualidade, o processo de projeto também afeta o tempo de colocação de um novo produto no mercado. Considerando-se que a competitividade a nível internacional de produtos de consumo é cada vez mais intensa, os fabricantes estão optando por produzir cada vez mais rápido, com menor custo e com maior qualidade possível, tanto para satisfazer os clientes como para conseguir melhores colocações de preços. Portanto, reforça-se a importância do projeto na indústria mundial.

A Engenharia Simultânea, considerada como uma abordagem integrada de desenvolvimento de produto, reforça a necessidade do altíssimo nível de qualidade exigido dos projetos, pois todas as discussões pertinentes a um produto, acontecem e são avaliadas ainda durante o seu projeto. As tomadas de decisão são antecipadas por equipes multidisciplinares de trabalho, com conhecimentos específicos exigidos em cada uma das atividades efetivamente envolvidas na produção de um produto, suportadas por meios tecnológicos avançados e infra-estrutura computacional eficientemente bem concebida.

Ou seja, todas as pesquisas relacionadas à produção de produtos conduzém e reforçam a necessidade de um melhor tratamento do projeto *per si*.

1.1. Projeto para *X*[1]

Os aspectos discutidos na seção anterior indicam que o processo de projeto deve ser tanto mais abrangente e eficiente quanto possível, dada a sua importância e reflexos em toda a vida do produto. Como as soluções de projeto, apresentadas pelos projetistas, dependem fortemente de seus conhecimentos e experiências na área de concepção do projeto, conclue-se que é necessário de alguma maneira, trazer os conhecimentos referentes aos estágios posteriores para que eles possam ser ponderados e considerados durante o projeto, sem que os projetistas sejam transformados em especialistas nas áreas pós-projeto. Isto significa que é necessário que o projetista e/ou a equipe de projeto seja multi-disciplinar e tenha conhecimentos das várias áreas envolvidas no projeto de um produto.

Alguns exemplos de pesquisas a nível de projeto, que trazem conhecimentos de fases pós-projeto para serem consideradas durante o processo de projeto, são as metodologias conhecidas como Projeto para Montagem (*DFA*), Projeto para Fabricação (*DFM*), Projeto para Reciclagem (DFR), etc.. Basicamente essas metodologias foram desenvolvidas, cada uma visando um estágio específico posterior ao projeto, para que justamente os conhecimentos utilizados respectivamente em montagem, fabricação e reaproveitamento de material fossem considerados aprioristicamente ainda durante a fase de projeto. Tanto o DFA quanto o DFM formalizaram regras e práticas baseadas em experiências e consenso entre os engenheiros, incorporando-as ao projeto do produto. Garante-se assim um aumento do grau de confiabilidade na criação do produto, conferindo-lhe uma visão mais abrangente de sua funcionalidade.

Não sendo o objetivo fazer uma análise muito detalhada destes aspectos, é no entanto importante referir que em resultados publicados por Boothroyd [Boothroyd], a aplicação da

[1] Tradução de *Design For X*, utilizada na bibliografia em geral para designar todas as áreas envolvidas e dependentes do projeto. Exemplos mais comuns são *DFM* e *DFA*, respectivamente *Design for manufacture* e *Desin for Assembly*.

metodologia DFA pode reduzir os custos de fabricação em 20% e aumentar a produtividade de montagem de 100 a 200%.

A tendência actual, em particular na Europa, é o de se formalizarem procedimentos e regras para todas as formas do *Projeto para X* (sendo X qualquer das atividades) para se garantir uma perfeita integração das diferentes actividades relativas ao desenvolvimento de produtos, condição necessária para a minimização dos custos, aumento da qualidade e competividade das empresas.

É dentro desta perspectiva que se desenvolve o presente trabalho : uma tentativa de contribuir, com a sua parcela devida, para o desenvolvimento de ferramentas de apoio ao projetista no sentido de, na fase mais conceitual de desenvolvimento de um produto, possibilitar a integração de informações referentes a outros aspectos do ciclo de vida dos produtos.

Capítulo 2

Estado da Arte em Pesquisas Baseadas em Features

Primordialmente, as pesquisas baseadas em features são relativas ou a fase de projeto ou às fases posteriores interligadas ao projeto, como planejamento do processo, fabricação e montagem. Apresenta-se aqui uma revisão brevemente discutida dos aspectos mais importantes relacionados a essas pesquisas. Procura-se também evidenciar o aspecto integrador conferido às pesquisas baseadas em features, pois da tecnologia de features espera-se uma melhor abordagem de integração de projeto e aplicações dependentes das informações modeladas no projeto [Salomons92].

2.1. Introdução

Durante a criação de novos produtos, uma das fases mais críticas, em se tratando de qualidade e performance desses produtos, é sem dúvida a fase de projeto. Nos últimos anos, muitos trabalhos têm sido executados a fim de facilitar a criatividade e a performance dos projetistas.

A atividade de projetar é um processo de criação de peças segundo requisitos e especificações funcionais, sob certas restrições. Isso envolve tanto a definição de forma quanto a designação de atributos de um modo interativo, para se atingir otimização. Para ser tão útil e completo quanto possível, o CAD deveria contemplar inteiramente essas atividades, porém desde a sua concepção somente o desenho parece ter sido fornecido com êxito pelas ferramentas computacionais existentes. Por outro lado, os processos de fabricação necessários à construção de uma peça deveriam ser considerados desde o estágio inicial do desenvolvimento de um produto, para melhorar tanto qualidade quanto manufaturabilidade do mesmo.

Os sistemas CAD normalmente permitem a criação de informações gráficas através da manipulação de entidades geométricas e topológicas, usando várias representações para a descrição do modelo, as quais na prática são apenas opções válidas e aceitáveis para a implementação de sistemas de desenho e documentação.

Uma nova tendência no modelamento de produtos, baseado no conceito de features surgiu em meados da década de 70, com o desenvolvimento de métodos para a automatização da programação NC. Desde então muitos trabalhos foram realizados usando diferentes abordagens e também diferentes definições para o conceito **feature**. Nessa área existem abordagens e técnicas particulares para a implementação e utilização do conceito, que vai desde o reconhecimento automático de padrões geométricos definidos como features até o próprio projeto baseado no conceito, passando pelo reconhecimento interativo e conduzido pelo usuário.

2.2. Definições e Contextualizações

Segundo Shah[Shah91b], muito do trabalho inicial em features parece originar-se do desejo de encontrar métodos para extrair geometria de peças a partir de modeladores geométricos, a qual poderia ser usada na geração de planos de processos, códigos de tecnologia de grupo e programas de comando numérico. Dixon[Dixon87a] argumenta que as representações em termos de features originam-se nas necessidades funcionais de raciocínio geométrico, requeridas de sistemas CAD ditos inteligentes. Chung[Chung88] expande esse argumento dizendo que a geometria de uma peça deve ser representada por entidades de mais alto nível relacionadas diretamente a certas funcionalidades de projeto ou características de fabricação, visando facilicitar a atividade de raciocínio sobre o modelo geométrico dessa peça.

As features têm origem no processo de raciocínio usado em várias atividades de concepção, projeto e fabricação[Dixon87a]. O termo ***feature*** é usado com significados diferentes, contextualizado segundo a área onde ele é aplicado, inclusive comumente é sinônimo de ***feature-de-forma*** pois para peças mecânicas a forma geométrica é extremamente importante e muitas aplicações de engenharia requerem a geometria de um produto como informação de entrada . Shah [Shah91b] diz que as features comportam um significado de engenharia de partes da geometria de uma peça ou montagem, ou genericamente, buscam inter-relacionar forma geométrica e significado funcional ou semântica.

Como as pesquisas referentes a features são relativamente recentes e alguns investigadores consideram-nas ainda em fase incipiente, existem muitas definições para o termo, bem como tentativas de generalização do seu conceito. Percebe-se que essa multiplicidade de definições implica na existência de uma dependência conceitual das features ao seu contexto de aplicação [Bronsvoort93] [Falcidieno92] [Salomons92].

Em [Bronsvoort93], encontra-se o seguinte conjunto de definições de features :

- geometria que corresponde a operações básicas de usinagem (Wilson).
- uma peça distinta ou característica de uma peça definindo uma forma geométrica, a qual pode ser tanto específica para processos de usinagem como para fixação e/ou medição (van 't Erve).
- uma área de interesse na superfície de uma peça (Faux).
- uma região de interesse no modelo de uma peç (Wilson e Pratt).

Algumas definições mais gerais :

- um conjunto de informações relativas a descrição de uma peça (Shah).
- elementos usados na geração, análise ou avaliação de projetos (Wilson).
- uma forma de aspecto funcional para projeto e fabricação (van Emmerik)

Salomons [Salomons92] cita mais algumas definições :

- Padrões recursivos de informação relacionados à descrição de uma peça (Shah).
- Um agrupamento semântico usado para descrever uma peça e suas montagens. Feature agrupa de um modo funcionalmente relevante, informações de projeto e fabricação (Giacometti e Chang).

- Uma forma ou entidade geométrica cuja presença ou dimensões são solicitadas para executar pelo menos uma função CIM e cuja disponibilidade como primitiva permite ao processo de projeto sua ocorrência (Lubby e Dixon).

- Um transportador de informação do produto que pode auxiliar o projeto ou comunicação entre projeto e fabricação, ou entre tarefas de engenharia (Shah).

- Qualquer entidade usada em raciocínio sobre projeto, engenharia ou fabricação (CAM-I)

Uma das melhores definições, na opinião do autor e consoante a abordagem conduzida por esse trabalho, é dada por Shah[Shah91a]:

- São formas genéricas às quais os engenheiros associam certas propriedades ou atributos e conhecimentos úteis em processos de raciocínio[1] sobre o produto. Ou seja, as features podem ser vistas como ***primitivas de engenharia***.

Analisando-se as várias definições apresentadas, nota-se claramente que o conceito feature verdadeiramente relaciona forma geométrica e funcionalidade, ao mesmo tempo que reforça a grande dependência da área onde está sendo aplicado, sem que exista uma definição formal expressável matematicamente. Dixon [Dixon90] também deixa claro que o conceito de feature deve transcender a forma geométrica, e discute a não-formalização das definições como um dos problemas ainda não resolvidos da pesquisa em features.

Shah [Shah91a], quando advoga que dois componentes básicos da definição de features são forma geométrica e significado de engenharia, argumenta também que tais componentes não estão limitados a um número finito, o que configura um quadro difícil para padronização de definições e conceitos. Ao mesmo tempo Shah também defende que a aceitação de padrões poderia funcionar como um freio para as novas pesquisas.

Vários pesquisadores defendem que a dependência conceitual entre features, área de aplicação, tipo de produto e nível de abstração é tão grande que torna-se praticamente impossível a criação de um modelo genérico [Bronsvoort93] [Shah88a]. Outros advogam que a generalização é objetivo imediato a ser atingido [Dixon90] no direcionamento mais eficaz das pesquisas sobre o tema.

2.3. Abordagens baseadas em Features

Atualmente as linhas de pesquisas baseadas em features sustentam-se em duas variantes:

- Reconhecimento de features

- Projeto por features

Essas duas abordagens serão discutidas nas seções seguintes. Antes porém, acha-se apropriado fazer um breve resumo da evolução dos modeladores geométricos, utilizados pelos sistemas de auxílio ao projeto, visto que a pesquisa em features está intimamente relacionada com a representação da informação nos modeladores geométricos 3-D.

[1]Processos de raciocínio é uma tradução de ***reasoning process***, porém no contexto a expressão pode ser lida como *processos de análise*

2.3.1. Sistemas CAD tradicionais, evolução e limitações.

Do ponto de vista histórico e particular ao projeto mecânico, o CAD começou com ferramentas computacionais que baseavam-se em modeladores geométricos pobres, o que em termos práticos, pouco ou quase nada podiam oferecer em se tratando de auxílio real ao projeto, como a sigla sugere. Originariamente e como já exaustivamente discutido em todas as pesquisas relacionadas aos sistemas CAD, esses simplesmente serviam como ferramentas de substituição das pranchetas dos projetistas, oferecendo-lhes compassos, réguas, esquadros e todas suas ferramentas de desenho, através de um programa computacional.

Desde a sua gênese até os dias atuais, muitos benefícios foram trazidos com a divulgação e utilização de tais ferramentas. Porém, as necessidades ou os requisitos impostos pelos usuários e organizações dependentes de tais sistemas têm exigido mais de qualidade na representação do conhecimento envolvido, especialmente visando um aproveitamento mais direcionado das informações tratadas nos sistemas CAD com vistas a integração das atividades delas dependentes.

A representação de informações de natureza puramente geométrica, a qual é o objeto dos sistemas tradicionais, já não atende aos requisitos de inteligência exigidos pelas indústrias usuárias desses sistemas, mesmo considerando-se toda a evolução existente nessa representação desde os segmentos de retas bi-dimensionais até as primitivas volumétricas 3-D. Faz-se necessário a existência de informações de naturezas diversas porém relevantes à configuração de qualquer produto, como por exemplo as informações de materiais e processos de fabricação, ou simplesmente informações tecnológicas.

Muitas aplicações de engenharia e fabricação auxiladas por computador requerem uma definição geométrica mais completa de um produto e os modeladores geométricos atuais não conseguem responder adequadamente a esses requisitos, visto que eles representam um produto somente pela sua geometria nominal, definida em termos de entidades de baixo nível, enquanto que aquelas aplicações exigem níveis de abstração mais altos[Shah88a]. Trabalhando-se com os modeladores geométricos convencionais, a produção de um plano de processo simples obrigatoriamente passa pela interpretação manual do processista, sobre os desenhos criados pelo projetista.

Pode-se resumir os requisitos atualmente cobrados dos sistemas de auxílio ao projeto em uma única frase : ***elevação do nível de representação das informações pertencentes à especificação do projeto de um produto***. As pesquisas conduzidas tanto pelas instituições acadêmicas como pelos desenvolvendedores de sistemas CAD, visando conferir aspectos de inteligência a esses sistemas,reforçam esses requisitos.

O tratamento das informações manipuladas dentro dos sistemas CAD é dependente do modelador geométrico que os suportam. A figura 2.1 mostra a evolução desses modeladores nos últimos 20 anos, desde os mais simples até a última geração, que são os modeladores de features sobre representações sólidas. A diferença fundamental entre cada geração de modeladores é o nível de informação suportado [Shah88d]. Um dos grandes avanços dos modeladores geométricos foi a capacidade de representação e tratamento de formas 3-D de objetos sólidos, o que é conhecido como modelamento por sólidos. Vários métodos para o modelamento de sólidos (representação de formas e estruturas de dados relacionadas) têm sido desenvolvidos, e os mais populares são **CSG** (Construtive Solid Geometry) e **B-rep** (Boundary representation).

A representação B-rep considera que um sólido pode ser visto como sendo limitado por um conjunto de faces limitadas por um conjunto de arestas, as quais são limitadas por dois

vértices. A informação é armazenada sobre esses elementos de contorno e suas relações de adjacência [Bronsvoort93], as quais representam as informações topológicas do sólido e possibilitam a representação de um sólido B-rep como um espaço fechado. As faces e arestas são representadas por equações matemáticas apropriadas, enquanto que os vértices são representados por suas coordenadas (X, Y, Z). A estrutura de dados é um grafo com nodos para os elementos de contorno, e ligações para as referências entre esses elementos. Essas ligações entre os nodos representam as relações de adjacência entre eles. Muitos modeladores baseados nessa representação fornecem somente faces planares devido a facilidade de representação, simplicidade e velocidade de operação.

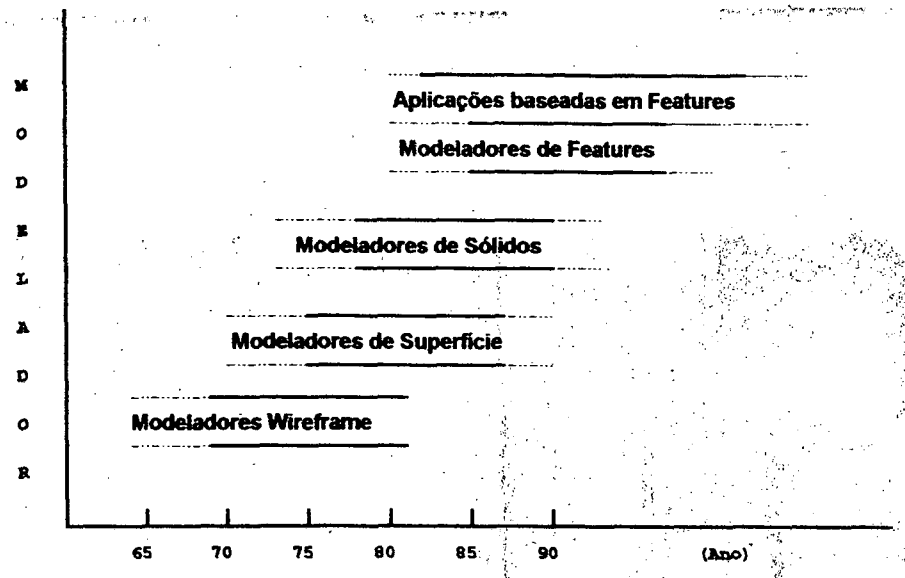


fig. 2.1 - Evolução dos Modeladores Geométricos[Shah88d]

A representação CSG é baseada em um conjunto de primitivas sólidas (paralelepípedos, cubos, cilíndros, esferas, etc) e um conjunto de operadores booleanos (união, intersecção e diferença). As primitivas podem ser combinadas através de operações boolenas para formar entidades mais complexas. A estrutura de dados de um modelo CSG é uma árvore binária, onde cada nodo contém ou uma primitiva sólida ou um operador booleano. Os nodos folhas serão sempre primitivas sólidas, enquanto que os nodos de operadores sempre terão nodos-filho (ou sub-árvores) direito e esquerdo. Nessa representação, ao contrário da B-rep, as informações sobre as primitivas são implicitamente definidas por um conjunto de parâmetros que instanciam os sólidos correspondentes.

As duas representações têm sido usadas nos modeladores de sólidos atuais, e potencialmente podem atender áreas distintas de aplicação. Os sistemas comerciais apresentam normalmente uma forma híbrida, combinando CSG para o modelamento (por sua simplicidade de definição, estrutura de dados simples, pela validade física de um sólido CSG) e derivando as informações da representação B-rep associada, o que é essencial para algumas áreas de aplicação (por exemplo, geração de programas NC) e de pesquisa, especialmente nas pesquisas envolvendo projeto por features.

Os modeladores sólidos, que significaram uma abstração superior aos modeladores 2-D e 2 ½ D, lançaram o embrião para a representação de outras informações de natureza não-geométrica dentro das bases de dados dos sistemas CAD. Os projetistas já não precisam mais pensar, ou melhor, traduzir seus projetos em termos de entidades de muito baixo nível

informacional como arcos e segmentos de retas, pois a forma de representação 3-D dos objetos já significa um nível superior de abstração.

Algumas áreas de aplicação, como por exemplo planejamento do processo e geração de programas NC, efetivamente aproveitaram-se desses modeladores para inferir e/ou associar informações adicionais às geométricas, elevando ainda mais o grau dos conhecimentos específicos diretamente associados aos modelos suportados pelos sistemas CAD, o que foi conceituado como ***features*** e enquadra-se na abordagem de reconhecimento de features, a ser discutida na seção 2.3.2. A primeira abordagem em features foi reportada em um sistema de auxílio ao planejamento do processo [Salomons92].

A abordagem features foi extendida até os sistemas CAD, trabalhando-se no conceito de **modelamento baseado features** ou projeto por features. As features são encaradas como entidades que efetivamente permitem a elevação do nível de informação dos dados manipulados pelos sistemas CAD, uma vez que se propõem a representar conjuntos de informações de natureza funcional e semântica dos objetos modelados, além da geometria. Segundo Dixon[Dixon90], a representação baseada em features no desenvolvimento de projetos de produtos é um pré-requisito essencial para uma nova geração dos chamados sistemas CAD inteligentes, que possam suportar o projeto conceitual, oferecer sugestões de fabricação, fornecer acesso automático para procedimentos de análise, e capturar e usar as intenções do projetista relativas a função e manufaturabilidade.

2.3.2. Reconhecimento de Features

Nessa abordagem, a idéia principal é fornecer técnicas automáticas de reconhecimento de features de interesse específico de uma aplicação, trabalhando-se sobre o modelo geométrico de um produto. Esses procedimentos de reconhecimento podem ser interativos e guiados pelo usuário, ou executados automaticamente. Além do modelo do produto, é necessário ter-se também a definição dos padrões a serem reconhecidos descritos de maneira compatível à forma de armazenamento das entidades presentes no modelo, possibilitando comparações.

Segundo Bronsvoort[Bronsvoort93], existem quatro classes de métodos de reconhecimento para modelos baseados na representação B-rep : *padrões sintáticos*, *baseados em regras*, *baseados em grafos* e *decomposição de volume*. Os métodos das três primeiras classes baseiam-se na comparação de padrões geométricos (faces, arestas e vértices), diferindo entre si na maneira pela qual os padrões são descritos. Os métodos da última classes buscam remover volumes contidos no produto que correspondam a operações de usinagem.

Henderson e Hwang[Hwang92] classificam as técnicas de reconhecimento em três abordagens gerais : *baseadas em regras*, *baseadas em grafos* e *redes neurais*. A abordagem baseada em regras usa técnicas de inteligência artifical para criar um conjunto de regras de features, determinando condições necessárias e suficientes para a definição das features. A abordagem baseada em grafos necessita de grafos definidores das features que devem ser usados como padrões na análise dos modelos sólidos. A abordagem de redes neurais utilizam exemplos geométricos para aprender a reconhecer a feature, através da assignação de valores numéricos à cada uma das faces componentes das features.

Os procedimentos de reconhecimento lidam com as bases de dados produzidas pelos modeladores geométricos existentes e várias pesquisas tem sido reportadas, prioritariamente trabalhando com modeladores sólidos. As 1ªs aplicações que usaram features basearam-se nessa abordagem.

Dentre as pesquisas reportadas encontra-se ainda em [Jared84] a descrição de um reconhecedor de features baseado no que foi denominado de *gramática de features*,

desenvolvida por Kyprianou. O reconhecedor agrupa e classifica as entidades presentes no modelo trabalhado e utiliza algoritmos baseados na gramática de features, para reconhecer a um nível genérico, protusões e depressões, as quais são reconhecidas em níveis mais detalhados como por exemplo, rasgos. Esse reconhecedor pode identificar features puramente prismáticas ou axis-simétrica classificando as arestas do objeto como côncavas ou convexas, e agrupando conjuntos de faces definindo a feature em *conjunto-de-faces*.

Henderson[Henders84] propôs uma abordagem baseada em regras para reconhecer features de um modelo B-rep usando um conjunto de regras de conectividade de faces e operações booleanas, em sua tese de doutoramento. O trabalho de De Floriani[Floriani88] propõe um método de reconhecimento de features baseado em um grafo denominado *Grafo de Decomposição do Objeto*, o qual é orientado por informações topológicas representadas explicitamente na descrição aresta-face do objeto.

Trabalhos mais recentes[Varady90] classificam os métodos de identificação de features em modelos sólidos em quatro grupos : *Rotulação*, *Identificação Nominal*, *Expressões Topológicas* e *Seleção Geométrica*. Rotulação consiste na atribuição de um identificador numérico ao sólido criado, pelo próprio modelador; identificação nominal é uma extensão da rotulação porém permitindo a atribuição de um mnemônico (nome) identificador dado pelo usuário; expressões topológicas consistem em descrições dos objetos através de linguagens especialmente concebidas; e finalmente seleção geométrica é baseada em restrições geométricas do tipo "vértice mais próximo a um dado ponto". Segundo [Varady90], todas essas técnicas de reconhecimento de features estão implementadas no trabalho chamado *Build/FF Solid Modeler*.

Hwang e Henderson[Hwang92] exploram a técnica de reconhecimento de features em um modelo sólido B-rep usando redes neurais. A rede deve ser treinada para aprender a reconhecer as features de interesse da aplicação, através de exemplos. Valores numéricos são atribuídos aos padrões *estudados* pela rede e aos modelos sólidos reconhecidos, o que confere um *fator de confiança* à feature reconhecida. Algumas vantagens desse método : ser extremamente veloz e o reconhecimento parcial das features é indicado pelo fator de confiança obtido pela rede.

Wang e Chamberlain[Wang93] exploram no sistema QTC-II uma abordagem para extração de features de fabricação através de decomposição de volumes chamada crescimento retroativo, onde a peça produto é usada como ponto de partida para a geração do bloco inicial de matéria bruta que deve contê-la, fazendo-se exatamente o processo inverso ao que será efetuado pela usinagem. O elemento central dessa pesquisa é o tratamento das features de protusão, as quais não associam diretamente à sua geometria qualquer operação de usinagem ao contrário das features de depressão (furos, cavidades, etc). O QTC II gera então conjuntos de features de depressão que gerem as features de protusão, podendo assim associar processos de fabricação à tais features.

2.3.2.1. Problemas e Limitações

Segundo [Bronsvoort93], existem diversos problemas nessa abordagem: o processo de reconhecimento é muito complexo e frequentemente trabalha com uma classe geométrica restrita; os algoritmos de reconhecimento são específicos de cada aplicação; não existe o reconhecimento de informações que não estejam presentes no modelo geométrico. Outro problema também apontado [Bronsvoort93] é a redundância do reconhecimento de features, partindo-se do princípio que o trabalho do projetista foi transformar suas entidades conceituais de projeto de alto nível em formas geométricas de baixo nível, que por sua vez devem voltar a ser vistas como entidades de alto nível pelo reconhecedor de features.

Observando-se que os reconhecedores geralmente exploram diretamente os modelos geométricos e que as features somente passam a existir após essa exploração, não se considera que informações (de natureza tecnológica, por exemplo) tenham já sido incorporadas antecipadamente às entidades geométricas.

Além do mais, o reconhecimento de features é um processo unidirecional, pois quando pensa-se numa abordagem de integração de sistemas dentro do projeto e fabricação de um produto, as alterações posteriores ao projeto não poderiam ser re-enviadas ao CAD automaticamente, pela impossibilidade de se alterar a base de dados do CAD através de um reconhecedor de features.

2.3.3. Projeto por Features

Essa abordagem sustenta-se na idéia do fornecimento de conjuntos de features e procedimentos de utilização desse conjunto ao nível da fase de projeto de tal maneira que o produto seja já concebido em termos dessas primitivas de engenharia. Ou seja, o produto é criado usando-se entidades de conhecimento de alto nível tanto mais próximas quanto possível da abstração conceitual do projetista. Os sistemas de auxílio ao projeto devem ser suportados então pelo que se convencionou chamar ***modelamento por features***.

O projeto por features, que é apontado por muitos pesquisadores como a abordagem mais promissora no tocante às features, apresenta alguns requisitos a serem cumpridos [Bronsvoort93] pelos sistemas desenvolvidos sobre ela: a) existência de um mecanismo para definir as descrições das features, a fim de se eliminar o problema de conjuntos restritos de features, o que se traduz por uma biblioteca aberta de features [Pamies89], onde geometria e topologia das features possam ser definidas dinamicamente; b) instanciamento de features deve ser paramétrico e preferencialmente baseado em vários conjuntos de parâmetros; c) restrições devem estar disponíveis para garantir a validade do uso das features e das relações inter-features; d) disponibilizar hierarquias de features para o tratamento de features compostas; e) permitir tanto o projeto top-down quanto bottom-up.

Existem duas categorias dentro dessa abordagem que são o *Modelamento Destrutivo* e *Construtivo* [Shah91b]. No modelamento Destrutivo o modelo da peça é criado por subtrações booleanas de um conjunto de features de fabricação pré-definidas, a partir de um bloco de matéria prima, permitindo a geração simultânea do plano de fabricação da peça. Exemplos dessa abordagem: Arbab em sua tese de doutoramento [Arbab82], Cutkosky e Tenenbaum com o sistema First-Cut [Cutkosky].

No Construtivo, o modelo da peça é obtido por adição e/ou subtração de features entre si, sem partir de uma bloco inicial de matéria-prima. As features são agrupadas em bibliotecas dinâmicas (customizáveis ou genéricas) ou estáticas (específicas para aplicação). É a abordagem com mais trabalhos publicados.[Lin92] [Shah88c] [Requicha89] [Pratt90] [Pamies93a] [Pamies93b] [Dixon87a], e atualmente é a preferida pelos pesquisadores da área. Dentre os sistemas comerciais disponíveis que adotam essa abordagem ou similares a essa, várias referências [Shah91b] [Bronsvoort93] são feitas aos sistemas Pro/Engineer e IDEAS.

Um aspecto importante dessa abordagem retratado por vários trabalhos [Dixon90][Shah90b], é a definição do conjunto ideal de features que devem ser disponibilizadas ao projetista, visando fornecer as features essenciais segundo a ótica do projetista. Algumas pesquisas [Cutkosky91] definem para o projeto features com características de fabricação, o que pode não ser desejável considerando-se que o projetista não tem a obrigação de raciocinar em termos de fabricação, mas sim puramente de projeto, ou mesmo que o projetista não possui o conhecimento relativo à fabricação. Ao mesmo tempo que, definindo-se um conjunto de

features de projeto, obrigatoriamente cria-se a figura de um conversor de features, pois essas features dificilmente servirão como features de montagem, por exemplo. Um conversor, diferente de um reconhecedor, deverá atuar sobre as features de projeto e gerar features específicas da aplicação a que pertence (por exemplo, planejamento do processo, montagem, etc).

Esse mesmo aspecto é ponto convergente e de apoio para a abordagem de projeto integrado, ou seja, para a concentração de esforços de uma equipe de projeto multi-disciplinar, constituída por elementos das diversas áreas inter-relacionadas e dependentes das decisões tomadas durante a fase de projeto. Naturalmente o modelamento de informações de projeto incorporando conhecimentos e atributos relevantes às aplicações posteriores ou seja, baseado em features no seu sentido mais amplo, leva ao conceito de *Engenharia Simultânea*, o que também será discutido nesse trabalho a partir de um ponto-de-vista genérico, considerando que naturalmente a Engenharia Simultânea necessita de apoio organizacional e de sistemas para poder existir.

2.3.3.1. Problemas e Limitações

Os problemas e limitações existentes no projeto por features serão discutidos nos tópicos de interesse particular dessa abordagem, item 2.6.

2.3.4. Abordagem Híbrida

Dixon[Dixon90] defende intuitivamente que um terceiro tipo de abordagem combinando as duas anteriores seria a ideal. Entretanto, os reconhecedores não levam em consideração se as informações de projeto foram construídas por features ou não, eles simplesmente avaliam os modelos geométricos e as features são redefinidas [Wang93], o que invalida todo o esforço envolvido no projeto por features e dificulta a conceitualização da abordagem híbrida.

Muitos aspectos deveriam ser considerados em uma abordagem híbrida, porém acredita-se neste trabalho que em realidade essa abordagem deveria ser tida não como híbrida, mas como uma abordagem integrada para projeto e aplicações (planejamento do processo, análises de engenharia, fabricação, montagem, etc), onde o elemento integrador seria um sistema de informação baseado em features, e cada aplicação teria um conversor particular ao seu domínio, o que é diferente de se ter um reconhecedor de features específico para cada aplicação. Dois trabalhos que defendem essa idéia são discutidos brevemente nos próximos parágrafos.

Shah[Shah88a] formaliza matematicamente a definição dos chamados *espaços de features*, valendo-se de conceitos da teoria dos conjuntos e álgebra linear. Os espaços seriam domínios particulares (áreas de aplicação) onde cada feature seria vista como um vetor pertencente a esse domínio. Considerando que as features dependem do tipo de produto, do processo de raciocínio das aplicações e do nível de abstração, ele discute as diferentes representações das features nos ditos espaços particulares e suas transformações inter-espaços, criando uma representação primária e obtendo representações secundárias através de aplicativos mapeadores com conhecimentos específicos sobre cada espaço.

Falcidieno e equipe[Falcidieno92] também apresentam uma abordagem semelhante, onde uma representação primária puramente geométrica serve de base para o trabalho de um conversor que cria os modelos baseados em features, dependentes do contexto. A representação primária está baseada em um grafo hierárquico chamado *SFOG* (*Shape Feature Object Graph*), o qual contém as features existentes no modelo e suas relações de

adjacência. O conversor incorpora diretamente ao SFOG atributos específicos a cada uma das apiicacões, criando um modelo de features particularizado a um contexto.

No item 2.10 serão tecidas mais considerações a respeito de uma abordagem integrada baseado em um sistema de informações que modela uma representação primária de features, a partir da qual representações secundárias podem ser obtidas.

2.4. Classificação de Features

Shah [Shah91b] diz que o número de features não é finito mas existe a possibilidade de categorizá-lo em grupos ou classes, permitindo um tratamento das features a nível dessas classes, ao invés de individualizado a cada feature. As discussões em torno das classificações podem levar também a taxonomias comuns. A vantagem mais importante da classificação é a utilização dessas classes no desenvolvimento de padrões de troca de dados de produtos.

Pratt e Wilson lançaram os requisitos funcionais para o suporte de features-de-forma em um ambiente de modelagem de sólidos, criando uma hierarquia dos tipos de features com maior classificação feito sob categorias *Explícitas e Implícitas* [Pratt85]. Features Implícitas são definidas sem ambiguidade, por exemplo através de uma descrição genérica e um número de parâmetros para uma ocorrência específica, mas que não estão avaliadas dentro de uma descrição geométrica explícita. Features Explícitas são features cuja forma está explicitamente descrita por entidades geométricas. Features implícitas foram sub-divididas em *genéricas* e *modificadoras*.

A classificação dada por Wilson e Pratt foi a primeira, e serviu como base para Modelo de Informação de Features-de-Forma (*FFIM*), do PDES (Product Data Exchange Specification). O PDES classifica as features em vários tipos, sendo que as duas classes principais adotadas foram *Explícitas* e *Implícitas*, adotando as mesmas definições dadas por Pratt e Wilson. As Features Implícitas por sua vez subdividem-se em : *passagens, depressões, protusões, transições, deformações* e *features de área*.

Shah[Shah88d] dividiu as features em 3 grupos : *features-de-forma* - entidades geométricas que definem forma e dimensão de uma peça; *features de precisão* - desvios aceitáveis da geometria nominal da peça (tolerância e acabamento superficial), e *features de material* - especificam tipo, classe e propriedades dos materiais em geral. Shah sugere também que outros conjuntos lógicos podem ser definidos, como Features de Montagem, Features Funcionais e Features de Análises.

Wingard [Wingard91] propôs uma taxonomia dependente da área de aplicação, sugerindo uma divisão de features-de-forma em duas sub-classes : *Atômicas* e *Compostas*. As atômicas, que são features simples as quais não podem ser decompostas em features ainda mais simples, que por sua vez são sub-divididas em *Peça* , *Modificador* e *Grupo*, onde *peça* é definida como uma feature atômica com existência própria; *modificador* é existencialmente dependente de uma outra feature; e grupo reune várias features que possuem coletivamente um significado de engenharia. *Features Compostas* são formadas por features atômicas, e sub-divididas em *padrões* e *complexas*.

Varady [Varady90] classifica as features em ordem crescente de complexidade em: *features geométricas simples* que são os elementos básicos constituintes de um modelo sólido, como arestas, vértices e faces; *features geométricas compostas* que podem ser definidas como um conjunto de features simples particularmente adjacentes; *features funcionais* as quais representam funcionalidades básicas ou tecnológicas bem definidas, como um furo; e por

último e sem taxonomia, objetos separados como componentes de uma montagem são tratados como features.

Bronsvoort [Bronsvoort93] reforça que é muito difícil criar uma classificação independente da aplicação : diferentes aplicações simplesmente requerem diferentes features, concluindo-se que uma classificação geral de features talvez seja quase impossível. Ele argumenta que as mesmas entidades básicas podem ocorrer em muitos lugares na classificação, porém semanticamente diferentes, exemplificando com um furo cilíndrico que poderia ser uma feature-de-forma funcional no projeto, uma feature de análise com propriedades especiais em análise de Elementos Finitos, uma feature de Planejamento de Processo para uma operação de furação, e uma feature de montagem para conexão a outras peças. Falcidieno[Falcidieno92] também advoga que contextos diferentes usam vocabulários diferentes para descrever o mesmo objeto com diferentes necessidades de representação das features.

Neste trabalho defende-se que todos os esforços de classificação, com vistas à padronização na pesquisa de features são válidas e devem ser incentivados, motivados principalmente pela integração automatizada das tarefas de projeto e fabricação de um produto que necessitam de padrões de troca de informações entre os sistemas envolvidos. Como o conceito de features pode ser aplicado a praticamente todas as áreas envolvidas na produção (projeto e fabricação) de um produto, é mister buscar-se padrões de features mesmo sabendo-se de antemão que as features são completamente dependentes da área de aplicação, e que uma mesma entidade pode ser vista como duas features distintas em ambientes distintos. A norma 48 do PDES/STEP é a maior e mais completa tentativa de padronização, e os trabalhos futuros devem sempre considerá-la sem no entanto limitarem o aspecto da investigação propriamente dita, uma vez que as pesquisas e os conceitos sobre features estão ainda saindo da fase embrionária.

2.5. Representação das Features

As features podem ser representadas em vários níveis. Shah[Shah91a][Shah91b] classifica a representação de features em dois tipos : *Procedural* e *Enumerativa*, onde procedural é representada por procedimentos que podem construir as features enquanto que enumerativa utiliza primitivas geométricas pré-arranjadas espacialmente para expressar as features. A representação procedural é mais abstrata pois permite que as features sejam especificadas em formato neutro sem a ligação direta com o modelo geométrico, porém têm a desvantagem de necessitar de um *conversor* para expressá-las geometricamente. A representação enumerativa é mais específica e de mais baixo-nível, valendo-se dos elementos geométricos para existir.

Em se tratando de features associadas diretamente a geometria (enumerativas), e considerando as representações geométricas oferecidas pelos modeladores sólidos atuais, um questionamento específico dirige-se à escolha da representação a usar : B-rep, CSG ou uma representação híbrida.

Pratt[Pratt88] apresenta uma discussão profunda sobre essa escolha e problemas inerentes a cada uma das representações, e aponta como preferência genérica a representação B-rep em detrimento da CSG. As desvantagens apontadas por Pratt à CSG residem na não-facilidade de detecção de intersecções entre primitivas e na ambiguidade no entendimento de modelos CSG, esta última fortemente relacionada ao reconhecimento de features. Porém a nível de operações como inclusões e exclusões de primitivas, o modelo CSG é muito mais simples de operar. É interessante notar que a possibilidade da utilização da abordagem híbrida é pouco considerada nessa discussão.

No mesmo trabalho, Pratt conduz outra grande discussão quanto a utilização de features de superfície ou de volume dentro da representação B-rep. A diferença existente entre os dois tipos reside em um pequeno conjunto de faces delimitadoras do volume. Os argumentos citados a seguir levam a conclusão que as features de volume são mais apropriadas:

- As interações entre features são mais fáceis de tratar com volumes;
- A operação de remoção das features é simplificada;
- Elimina-se o problema de crescimento das faces;
- Contorna-se o problema das faces parciais, que são faces coplanares e adjacentes de primitivas distintas;
- Facilidade de decomposição automática de volumes a serem removidos como elementos de usinagem.

Considerando que algumas aplicações dependentes do projeto e das informações disponibilizadas nos modelos de features, a representação híbrida parece ser a mais adequada em se tratando de *projeto por features* pois atende tanto as aplicações que necessitam de informações detalhadas fornecidas pela B-rep, como as que trabalham sobre as informações mais genéricas fornecidas pela CSG.

2.6. Tópicos de Interesse Particular na Abordagem de Projeto por Features

Dentro da abordagem de *projeto por features*, alguns tópicos são de particular interesse e devem ser discutidos. Como citado anteriormente essa abordagem sustenta-se na idéia do fornecimento de conjuntos de features e procedimentos de utilização desse conjunto ao nível da fase de projeto de tal maneira que o produto já seja concebido em termos dessas primitivas.

Os tópicos a serem discutidos nessa seção são :

- Features Genéricas : pré-definição, abrangência, natureza, limitações, flexibilidade e manipulação.
- Modelos paralelos de features : Unicidade das Features.
- Contiguidade
- Funções.
- Tolerâncias e Inter-dimensionamento.

2.6.1. Features Genéricas

Um dos grandes aspectos de interesse exatamente por sua complexidade e posição estratégica nas pesquisas conduzidas em *projeto por features*, é o fornecimento de conjuntos de features genéricas. Considerando que a atividade de projeto detalhado dá-se basicamente sobre primitivas geométicas e que as features, como primitivas de engenharia que se propõe a ser, devem primordialmente manter um ambiente de projeto não-restritivo quanto a criatividade do projetista.

Nesse ambiente não-restritivo, a flexibilidade e os recursos necessários para a criação de peças complexas passa pelo grupo de entidades disponíveis para a expressão das intenções do projetista. Shah [Shah91a] advoga que dois componentes básicos da definição de features são *forma geométrica e significado de engenharia*, e que tais componentes não estão limitados a um número finito, o que configura um quadro difícil para a concepção de um grupo de features genéricas.

Nas pesquisas inicialmente feitas sobre o projeto por features, normalmente ofereciam-se bibliotecas de features contextualizadas dentro do domínio de uma aplicação [Lubby86][Cutkosky]. Isso traz consigo o problema da limitação da capacidade criativa da atividade de projeto, que então restringe-se a utilização de features contidas naquelas bibliotecas. Uma solução adotada que embora ainda mantenha alguns problemas, é o oferecimento de bibliotecas de features configuráveis pelos próprios usuários. Faz-se então uma particularização dinâmica das features segundo as necessidades correntes dos usuários.

Ainda assim, um pequeno problema permanece intrínseco à solução: a forma funcional da definição das bibliotecas configuráveis de features. O ponto em questão é : oferece-se um grupo de features básicas e um grupo de relações e regras que definam a criação de novas features, porém serão previsíveis todas as formas possíveis de serem criadas dentro dessas condições? Para que isso aconteça, as regras e relações definidoras das novas features devem ser tão precisas quanto restritas sem perder a flexibilidade e a generalidade.

Um exemplo simples é mostrado na figura 2.2, para ilustrar mais claramente a discussão: supõe-se que as duas primitivas sólidas (paralelepípedo e cilíndro) são features básicas oferecidas em uma biblioteca e que uma das funções disponíveis na criação de novas features é a operação de união. Pode-se prever todas as formas possíveis de serem construídas sobre essas duas primitivas, segundo essa função? A resposta é não, obviamente. Então, como se deve tratar elementos que não são previsíveis? Incorporando-se regras ou restrições que permitam formalizar as novas criações, onde se percebe o aspecto limitante da formalização.

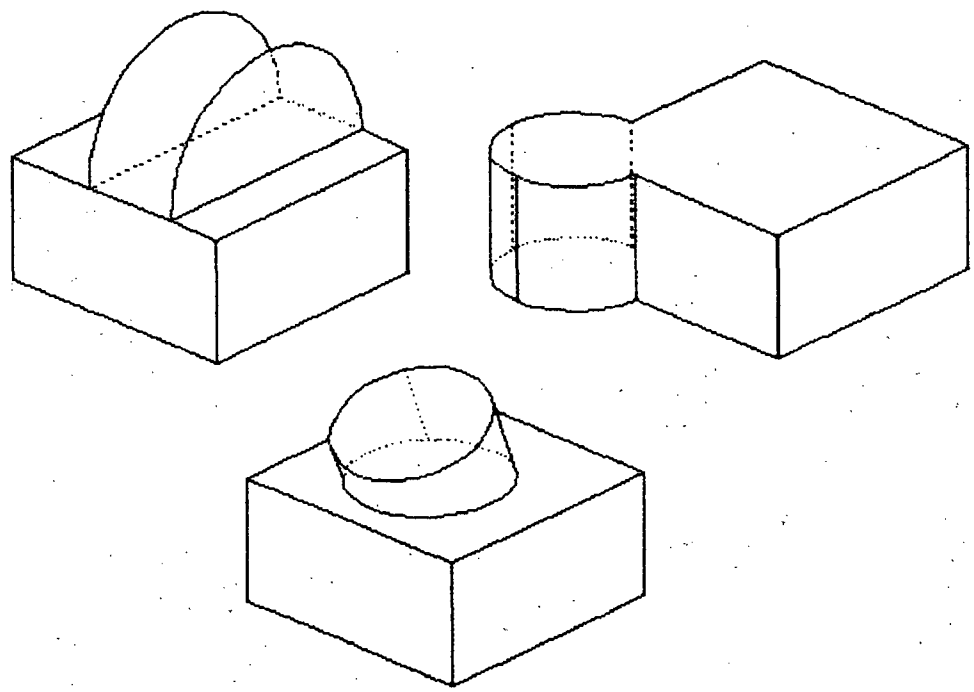


fig. 2.2 - Formas complexas resultantes de duas primitivas e uma função

No exemplo anterior, duas restrições ajudariam a reduzir e tratar o conjunto das combinações possíveis : não permitir a intersecção das features e a adjacência entre elas seria baseada em faces planas de contato.

Discute-se também a forma de definição das features nessas bibliotecas : se proceduralmente ou se implícitamente através de parâmetros definidores. A definição procedural implica no agrupamento de instruções relacionadas ao arranjo topológico de elementos geométricos básicos (por exemplo, a definição de um furo cilíndrico pode se dar através de : *duas faces planares, circulares e paralelas interligadas por uma face cilíndrica perpendicuiar às duas, na região do espaço contida entre as duas faces planares*). A definição paramétrica baseia-se em um ou mais conjuntos de parâmetros que caracterizam a feature : um furo cilíndrico é composto por um *raio ou diâmetro* e uma *profundidade ou altura*. Qualquer que seja a forma de definição, opções alternativas de criação da mesma feature devem ser consideradas.

2.6.2. Modelos Paralelos : Unicidade das Features

Outro ponto clássico de discussão reside na unicidade das features presentes em um modelo, intimamente relacionado com as diferentes vistas que podem ser aplicadas sobre uma feature. O exemplo tradicional é o apresentado na figura 2.3, onde a mesma região de uma peça pode ser vista de duas maneiras distintas.

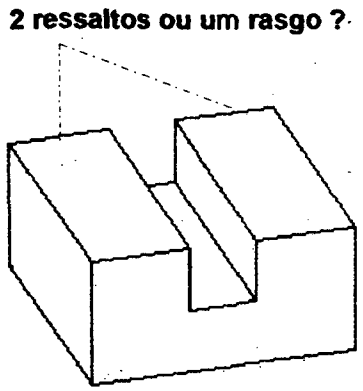


fig. 2.3 - Modelos alternativos

A interpretação dada àquela região será certamente dependente do domínio da aplicação interessada na peça. A discussão vai para a coexistência de modelos paralelos baseados em features, sobre um mesmo modelo geométrico. A manipulação de modelos paralelos toma-se demasiadamente complicada, visto que uma alteração em um dos modelos pode implicar em grandes alterações ou até mesmo na destruição das informações existentes sobre a mesma região em um outro modelo. No exemplo anterior, se um dos ressaltos é removido o rasgo deixa de existir.

A necessidade de vistas alternativas sobre um mesmo modelo parece tratável através das definições de modelos *primários* e *derivados* [Shah91a][Falcidieno92], onde os derivados atenderiam vistas particulares, independentes entre si e mantendo uma relação direta com o modelo primário, *sem no entanto coexistirem*. Ou seja, a definição de cada feature deve ser única dentro de um mesmo modelo, o que não necessariamente se traduz pela associação

unívoca entre geometria e semântica. Uma mesma forma pode executar diversas funções e uma mesma função pode ser executada por várias formas.

2.6.3. Contiguidade

Todos os elementos geométricos presentes na definição de uma feature qualquer devem ser necessariamente contíguos? Ou existem situações onde tal característica é dispensável? Pensando- se em um furo definido sobre duas paredes paralelas não adjacentes de uma peça (fig. 2.4), existe uma região espacialmente descontínua do furo. As perguntas voltam-se para: nessa situação, o furo na verdade não deveria ser dividido em dois furos relacionados por restrições geométricas que garantissem sua validade funcional (supondo que, por exemplo, esse furo servisse de guia para o encaixe de um pino)? Ou então que se criasse um elemento geométrico *virtual* de ligação para que o furo continuasse a ser visto como uma única feature, por razões de simplicidade de manipulação? Ou ainda uma abordagem radicalmente restritiva, onde não se permitisse tal situação e esse furo, que pode ser chamado *composto*, tivesse que ser dividido em dois furos independentes e a manutenção de suas relações funcionais recaíssem no projetista?

Essas discussões podem ser enriquecidas acrescentando-se um aspecto relacionado com a ocorrência da descontinuidade, a qual normalmente é oriunda de intersecções entre features. Dentro da literatura base utilizada nesse trabalho, não foi encontrada qualquer referência a features cujas entidades componentes, sejam conjuntos de faces ou volumes fechados, apresentassem elementos descontínuos, o que serve para reforçar a idéia da intersecção ser a origem da descontiguidade. Normalmente ela é gerada por remoções de porções sólidas de uma peça, o que pode ser visto como uma mudança na forma de relacionamento das features contidas nessas remoções.

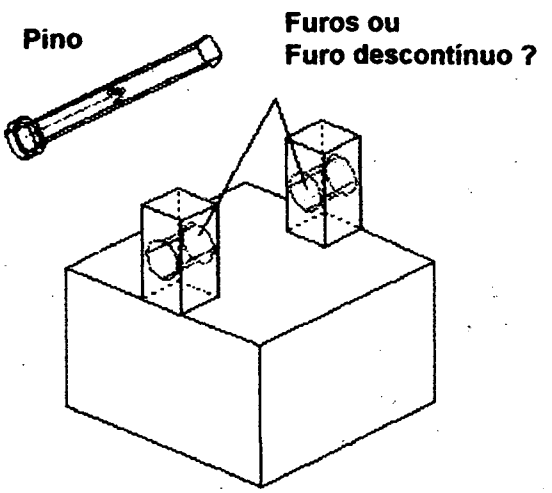


fig. 2.4 - Contiguidade das features

O trabalho de Mayer, Su e equipe[Mayer94] endereça especificamente o tratamento das intersecções entre as features, propondo uma representação primária para as features chamada *ECTOF* baseada em grafos. As features são re-arranjadas e particionadas segundo as intersecções que as envolvem, e a *ECTOF* é sempre mantida atualizada com as features no último estágio da criação.

2.6.4. Relações inter-features

Os parâmetros descritores de algumas features encontram-se implicitamente contidos na própria definição das features e mais, estão associados a atributos descritores de outras features. O exemplo clássico é o *furo passante*, onde qualquer que seja o tipo e raio do furo, o seu comprimento ou altura será diretamente dependente da(s) feature(s) que o contiver(em). O que se apresenta então, é uma das relações também clássicas de dependência *pai-filho*, onde as features filhas são instanciadas segundo atributos das features pais.

Logicamente algum mecanismo de associação entre features dessa natureza deve ser fornecido para que se mantenha a validade funcional e semântica. Ainda no caso do furo passante, de algum modo o comprimento deste deve ser derivado das dimensões de seu(s) pai(s).

Outra relação inter-features que existe aplica-se aos chamados *padrões* de features, onde existe uma restrição matemática regendo essa relação. Por exemplo, tomando o exemplo de um padrão de *n* furos circulares, um de seus parâmetros definidores é o raio ou diâmetro de uma circunferência sobre a qual os *n* furos se acham dispostos. Outro parâmetro é o ângulo existente entre cada furo ou o ângulo total onde os furos devem ser dispostos. Parâmetros dessa natureza permitem estabelecer leis matemáticas de formação entre as features geradoras de um padrão.

Outras duas relações inter-feature mais sutis que podem existir são as de *posicionamento* e *adjacência*. As features, quando instanciadas a partir de uma biblioteca, podem ser posicionadas em uma peça de modo relativo a uma outra feature pertencente a essa mesma peça, o que estabelece um posicionamento geométrico relativo entre tais features. Ainda no instanciamento, uma feature pode ser posicionada espacialmente de modo adjacente a uma outra feature pertencente a mesma peça, o que estabelece uma adjacência baseada em elementos geométricos de baixo nível pertencentes às features. Essas relações serão de extrema importância no modelo estudado nesse trabalho.

2.6.5. Tolerâncias e Inter-dimensionamento

Embora as dimensões definam a geometria nominal exata de uma peça, os processos de fabricação disponíveis para sua produção podem não ser capazes de garantir tal precisão. As tolerâncias entram no cenário como intervalos aceitáveis de variação da geometria nominal das peças, com garantia de manutenção da funcionalidade projetada nas mesmas. As tolerâncias também representam restrições de montagens e são importantes para uma correta definição dos planos de processos relacionados a uma peça.

Por outro lado, as tolerâncias têm um papel decisivo nos custos de fabricação das peças uma vez que o cumprimento de algumas tolerâncias pode implicar em processos de fabricação especiais, sofisticados e precisos, o que significa dizer *caros*. Uma outra visão da situação baseia-se no fato que as máquinas disponíveis em um determinado chão-de-fábrica podem não possuir capacidade necessária para produzir em tolerâncias muito pequenas, o que significa aquisição ou contratação de máquinas tecnologicamente mais avançadas.

Segundo a teoria de zonas de tolerâncias proposta por Requicha [Requicha84], o valor de uma tolerância representa a largura ou diâmetro de uma zona, dentro da qual um ponto, uma aresta ou uma face de uma feature devem estar contidos.

Jasthi [Jasthi94] agrupa as tolerâncias em dois tipos para garantir um modo simples de relacionar tolerâncias às features: *Intra-Feature* e *Inter-Feature*. As tolerâncias Intra-Feature modelam aquelas que dependem de uma única feature (acabamento superficial, tolerância dimensional, cilindricidade,etc.), e podem ser acopladas às primitivas geométricas de baixo nível que compõe quaisquer features (arestas, faces, vértices). As tolerâncias Inter-Feature (*Interdimensionamento*) relacionam duas features e requerem a criação de primitivas geométricas especialmente construídas para representá-las.

Mullins e Anderson [Mullins91] apresentam um método para representar e manipular tolerâncias em um ambiente de modelamento baseado em features. O método é baseado numa representação orientada por objetos e usa vetores toleranciados para representar dimensões e tolerâncias no modelo. Mullins considera que as features podem encapsular as informações necessárias para o toleranciamento de uma maneira compacta, localizada e computacionalmente eficiente. Mullins cita ainda exemplos de aplicações que utilizam a sua representação de tolerâncias: planejamento do processo e aplicações de síntese de tolerâncias.

2.7. Modelos Paradigmáticos baseados em Features

Dentro das pesquisas em features, alguns trabalhos servem de referência por suas contribuições tanto na resolução dos problemas quanto na sinalização de novas abordagens de investigação. Destacam-se nomeadamente o *ASU Features Testbed*, desenvolvido na Universidade Estadual do Arizona sob a coordenação de J. J. Shah, o qual será discutido com mais propriedade; o projeto *IMPPACT* desenvolvido sob o âmbito do programa da União Européia denominado *ESPRIT*; os sistemas *GEKO/KALEIT*, *DICAD*, *IIICAD*, *First-Cut* e *QTC* (Quick Turnaround Cell). A tabela da figura 2.5 apresenta alguns desses trabalhos.

2.7.1. First-Cut

O First-Cut é um sistema desenvolvido por Cutkosky e equipe[Cutkosky], que integra projeto baseado em features com planejamento de processo. O trabalho baseia-se na premissa que o projeto para fabricação pode ser atingido melhor quando a fase de projeto incorpora simultaneamente, uma análise dos processos que devem ser usados para fabricar o produto projetado.

Atendo-se a peças usináveis em máquinas NC, o First-Cut baseia-se no modelamento destrutivo onde o projetista, a partir de um bloco de matéria bruta, define as operações necessárias para a obtenção do peça final. Tais operações são associadas diretamente aos passos do plano de fabricação necessário para produzir a referida peça, de tal maneira que o resultado final da fase de projeto é não somente o modelo geométrico sólido da peça criada, mas também o respectivo plano de processo para fabricá-la. O projetista conta com o auxílio de pequenos sistemas especialistas (obviamente baseados em técnicas de inteligência artificial) que, possuindo conhecimentos específicos sobre aspectos da fabricação das peças oriundos de fatos dispostos em bases de dados, servem de conselheiro ao projetista sem que este efetivamente deva ser um profundo conhecedor da área de fabricação.

Resumindo, as peças são projetadas em termos de planos de fabricação que possam produzí-las, contando com o apoio de especialistas detentores de conhecimentos acerca da fabricação e que os tomam disponíveis aos projetistas, em forma de supervisão, onde os verdadeiros engenheiros de fabricação não se fazem presentes.

Pesquisador Universidade de Referência	Nome Sistema	Representação Geométrica Feature-de-Forma	Representação das Features	Inter-relações de Features	Dimensões Tolerâncias	Validação Features	Tratamento múltiplas visões	Aplicações Subsequentes	Plataforma software comercial	Linguagem usada
D.C. Anderson Purdue Univ. USA	QTC Quick Turnaround Cell	CSG / B-rep	Objetos : . Atributos . Métodos	São inferidos de curtões; Features compostas e padrões presentes	Tolerâncias dimensionais presentes	Presente Faces de contorno das features identificadas	Através de refinamento de features	Planejamento do Processo	TWIN solid m. X-windows KEE expert sys	Common Lisp e C
M.R. Cutkosky Stanford Univ. USA	First-Cut	NURBS	Objs (frames) : . Atributos . Restrições	??	Dimensões representadas por restrições	??	Por reconhecimento de features	Planejamento de Processo	ALPHA-1 (modelador sólido baseado em NURBS)	Hyperclass Lucid Common Lisp
J.R. Dixon University of Massachusetts USA	Vários sistemas para projetar produtos específicos	Estrutura de dados com representação tipo CSG/B-rep	Objs (frames) : . Atributos . Restrições	Grafos relação features primárias, secundárias e add-on	Somente dimensionamento	??	Através de conversão de features	Análise	Geomod modelador sólido	Common Lisp
M van Emmerik TU Delft NL	GeoNODE	CSG (semi-espaços)	Objetos : . Atributos . Métodos	Nodos (sistema de coordenadas locais)	Somente dimensões e restrições dimensionais	Através de regras contidas nos objetos	–	Análise Movimentos	X11	C++
M.Mantyla Helsinki Univ. Finlandia	HUTFPD	CSG / B-rep (?)	Objs (frames) : . Atributos . Métodos	Por construção geométrica	–	–	Através de relaxation de features	Planejamento de Processo	GWB e HutWindows (modeladores sólidos acadê.	Lisp e C
T. Ozsoy Lehigh Univ. USA	??	CSG / B-rep	Objetos	Grafos relação features primárias, secundárias	Presentes usando features-forma primitivas e restrições	Presentes em parâmetros e restrições	–	Análise de Tolerâncias	??	??
M.J. Pratt Cranfield Inst. UK	??	CSG / B-rep	Objetos implícitos e explícitos	??	??	??	??	??	Romulus modelador sólido	??
A. Requicha U. South Calif. USA	AI/SM testbed (Artificial intelligence modelling	CSG	Objetos : . Atributos . Métodos	CSG tree	Presentes usando Grafos-V	Por regras : presença , não-intrusão acesso e dimensional	Através de reconhecimento de features	fabricação	PADL-2 modelador sólido Knowledgecraft	Common Lisp
J.J. Shah Arizona State U. USA	ASU Features Testbed	CSG / B-rep	Objetos : . Atributos . Métodos	Grafos relação features primárias, secundárias	Presentes	Através de regras	Através de mapeamento de features	Codif. GT Planejamento de Processo Avaliação de fabricação	Solidesign modelador sólido e outros	C

fig. 2.5 - Pesquisas baseadas em Features[Salomons92]

2.7.1.1. Conclusões sobre o First-Cut

Projetar em termos de planos de fabricação, limitando-se somente a features simples fabricáveis facilmente, parece ser uma abordagem interessante. Porém quando se pensa em termos de operações de fabricação complexas, várias fixações, formas resultantes sofisticadas, essa abordagem tal como apresentada no First-Cut deixa muito a desejar. Quanto às intersecções de features nenhuma referência foi feita, o que também em termos de fabricação pode trazer um simples problema, mostrado na peça da figura 2.6. Como o rasgo significa uma remoção de material, como se poderia construir a referida peça?

A única solução viável exige do projetista a remoção individual de cada um dos retângulos imaginários que circundam o ressalto do centro do rasgo. Ou seja, complicam-se funcionalmente os procedimentos criativos do projetista.

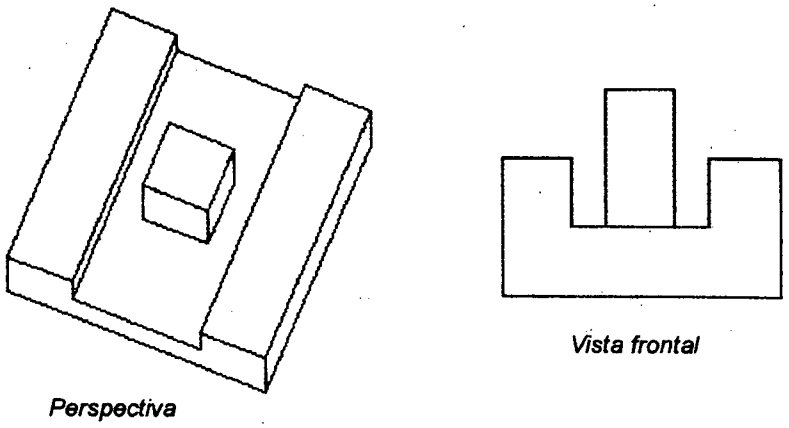


fig. 2.6 - Contra-exemplo para o Modelamento Destrutivo

2.7.2. ASU Features TestBed

O **A.S.U. Features Testbed** é um conjunto de módulos de projeto, documentação e avaliação de peças mecânicas que oferecem recursos para uma descrição unificada de produto. Esses módulos estão divididos em dois sistemas : um para modelamento do produto através de features e outro para interpretação flexível desse produto (mapeamento e aplicações).

O sistema de modelamento do produto está dividido em vários modeladores, cada um deles possuindo uma representação particular de features, porém todos eles concorrendo para uma definição integrada do produto. Existem mapeadores que permitem a transmissão de informações entre os modeladores. O conhecimento a respeito das features é fornecido pelos próprios usuários, garantido ao mesmo tempo flexibilidade e adequação das entidades tratadas pelo sistema.

O sistema de mapeamento facilita a implementação de aplicações baseadas nas features, como avaliação de manufaturabilidade, codificação GT, análise de fadiga, etc. Bases de conhecimentos devem ser adicionadas ao sistema fornecendo descrições das informações a

2.7.2.1. Representação das Features

As features-de-forma são tratadas como volumes positivos, negativos ou modificadores, sendo que as features de volume positivo e negativo são representadas por sólidos independentes, enquanto que as modificadores não possuem representação sólida, ou seja, são existencialmente dependente dos outros dois tipos. O ASU usa dizer que as features positivas e negativas são *FPV* (features que produzem volume, considerados como sólidos independentes). As features-de-forma são usadas como a representação fundamental e estão sempre associadas a uma entidade geométrica.

As features-de-forma são definidas segundo uma abordagem orientada por objetos, onde cada classe corresponde a uma feature, e onde os mecanismos naturais da abordagem orientada por objetos permitem a herança e hierarquia entre as features. Linguagens interpretadas foram criadas para suportar a escrita de regras e expressões em forma de lista, como parte da definição das features. Essas regras e expressões são usadas para criar uma rede de heranças e para validação das features. Regras de herança e expressões são usadas para determinar os valores dos parâmetros derivados dos pais das features. Regras de validação são usadas para verificar se a feature está sendo usada de modo válido. Restrições podem ser colocadas em como a feature é usada, através de regras de compreensão. Essas regras interligam o ambiente de modelamento através de dimensão e restrições de posicionamento nas features. A linguagem de regras de compreensão é muito similar a linguagem de regras de herança, exceto que comparações algébricas são suportadas.

Para cada feature genérica, um sólido representativo é criado usando o conjunto de primitivas (cilindro, paralelepípedos, sólidos de revolução, etc.). Esses sólidos são combinados usando-se operadores booleanos(união, intersecção e subtração) para definir as *FPV*, em uma linguagem específica.

2.7.2.2. Conclusões sobre o ASU

Conceitualmente falando, a abordagem de projeto por features explorada no ASU busca a integração do modelamento de um produto através de features e aplicações correlacionadas. Existem poucos problemas reportados quanto a definição conceituai do ASU, e na literatura específica somente dificuldades quanto à definição das tolerâncias dimensionais entre as features no modelador de tolerâncias, que segundo Wang [Wang93], não são nem fáceis de usar nem definidas segundo conceitos de engenharia. Cabem ainda os seguintes questionamentos:

- Qual o grau de dificuldade encontrado por um usuário comum para a definição de sua(s) biblioteca(s) particular(es) de features-de-forma?
- A exigência de um número muito elevado mapeadores inter-modeladores **n * (n-1)**.
- O tratamento das features se dá realmente não segundo classes genéricas, mas segundo classes específicas, pois de outra maneira os parâmetros de herança e hierarquia não seriam permitidos. Isso pode ser mais abrangente?
- A não-associação do significado das features diretamente à geometria das primitivas, mas sim a comandos do modelador sólido que constroem a representação sólida das features.

2.8. Apresentação do Modelo UNINOVA/CESO

O modelo defendido nesse trabalho baseia-se na abordagem conceitual explorada pela pesquisa desenvolvida no âmbito do projeto BRITE-EURAM, dentro da União Européia. O referido projeto deve fornecer indicativos de pesquisa dentro da concepção integrada de um produto, partindo-se de uma definição fornecida por uma entidade usuária e culminando com a entrega desse produto àquela entidade. Uma das tarefas desse projeto é denominada *Sistema de Apoio ao Projeto*, e situa-se na fase de criação de um produto. O objetivo primário dessa tarefa é fornecer um sistema de apoio a decisão que permitirá incluir considerações de fabricação durante a fase de projeto, ou seja, um sistema de *projeto-para-fabricação* onde exista a possibilidade de criar e avaliar diferentes alternativas de produção para um dado produto, integrando diferentes aplicações.

Fundamentalmente, essa tarefa visa estudar e propor uma estrutura informacional que suporte a integração de algumas aplicações existentes na produção de produtos genéricos, que são : *Projeto, Seleção de Processos, Tecnologia de Grupo, Verificação de Projeto e Avaliação de Manufaturabilidade*. Por *produtos genéricos* entenda-se peças mecânicas formadas por componentes individuais ou montagens de componentes, atendo-se a caracterização desses produtos ao escopo dos produtos tipicamente encontrados na indústria mecânica.

Por estar inserida dentro de um projeto maior, essa tarefa está sujeita a algumas imposições de desenvolvimento oriundas do próprio BRITE e que são pertinentes a plataformas de hardware e software, que serão consideradas quando necessário.

2.8.1. O Modelo Conceitual

O modelo representa um conjunto de entidades tipicamente encontradas nas atividades de projeto e fabricação de um produto, segundo os objetivos das aplicações especificadas. Pela natureza dessas aplicações e pela maneira natural de desenvolvimento de produtos mecânicos, as informações contempladas no modelo são as especificações de um produto em termos de forma e geometria, materiais, processos tecnológicos e máquinas de construção. O escopo do trabalho abrange o desenvolvimento de um sistema CAD que efetivamente auxilie o engenheiro de projeto, segundo a filosofia de Projeto para Fabricação.

Na figura abaixo representa-se de uma forma hierárquica a estrutura das informações necessárias ao desenvolvimento de um produto. No topo dessa hierarquia representam-se três classes distintas de informação envolvidas nas atividades de projeto para fabricação de um produto: as peças componentes, as máquinas disponíveis para a fabricação e as informações gerais relativas ao produto.

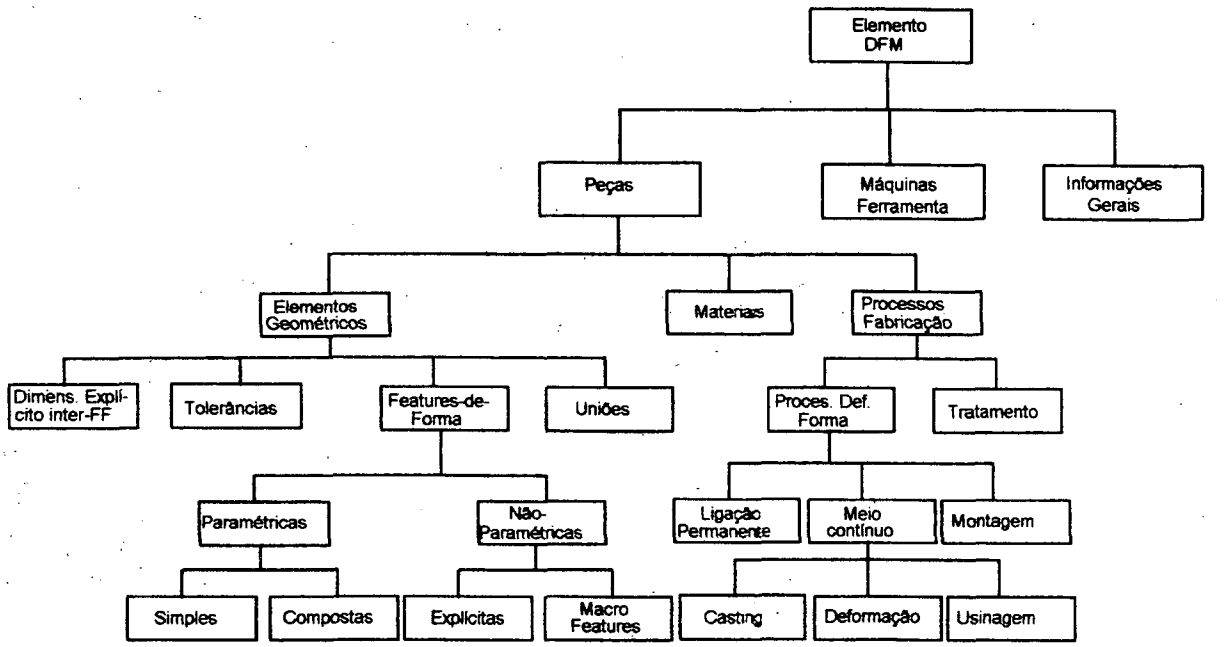


fig. 2.7 - Hierarquia de informações do Modelo Conceitual

Identificou-se então a necessidade de representar três classes distintas de informação envolvidas nas atividades de projeto para fabricação de um produto : as peças componentes, as máquinas disponíveis para a fabricação e as informações gerais relativas ao produto. As peças representam unidades individuais ou montagens; as máquinas são caracterizadas por seus parâmetros nominais (capacidade, dimensão, velocidades, etc) e as informações gerais englobam toda a classe de informação pertencente ao produto que sirva para algum tipo de raciocínio, ou que seja auxiliar nos processos de tomadas de decisão relativas àquele produto.

Sendo este modelo parte do trabalho de outros elementos da equipe de investigação onde o autor se insere, ir-se-á apenas apresentar de uma forma resumida o modelo consubstanciado pela estrutura apresentadas anteriormente. Assim:

- **Peças**: formam a classe principal de representação das informações do modelo, pois peça é a **entidade-produto**. A entidade *peças*, como mostra a figura 2.7, agrupa três classes distintas de informação (Elementos Geométricos, Material e Processos de Fabricação) e estão virtualmente sub-divididas em peças *simples* e *complexas*. Essa classificação baseia-se no tipo do processo de ligação, que caracteriza a junção existente entre as features componentes da peça : *simples* significam que as features são definidas sobre um bloco de material contínuo; enquanto que *complexas* significa a existência de um processo de ligação (fixa ou amovível) entre as features componentes. As peças complexas por sua vez, subdividem-se em *compostas e agregadas* (figura 2.8), segundo a natureza daquele processo de ligação : se o processo é permanente (por exemplo soldagem e colagem), as peças são ditas compostas; se o processo é uma montagem as peças são ditas agregadas.

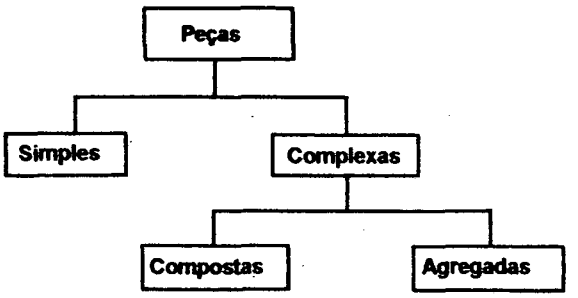


fig. 2.8 - Classificação das Peças

Um aspecto importante referente as peças é como sua representação pode ser entendida de diferentes maneiras, segundo uma visão particular. O projetista irá focar sua atenção nas informações de geometria e de material, da peça. Ao mesmo tempo, o engenheiro de processo tem uma visão da peça mecânica como um conjunto de procedimentos de produção e recursos tecnológicos associados para produzí-la.

- **Máquinas-Ferramenta**: representam as máquinas que devem ser usadas para a construção da forma geométrica de um produto, e são descritas em termos de suas especificações nominais. Ao mesmo tempo, máquinas de transporte ou inspeção podem também serem modeladas. Nesse trabalho foram consideradas somente máquinas de usinagem (torno, fresadora, centros de usinagem, etc).

- **Informações Gerais**: informações que são relevantes a nível global do projeto.

A nível das peças, a entidade mais importante existente no modelo, encontram-se os seguintes grupos de informações:

- **Elementos Geométricos**: correspondem à abstrações obtidas em relação a geometria e topologia dos elementos componentes das peças, e estão divididos em quatro entidades distintas : *features-de-forma, uniões, dimensionamento inter-features-de-forma, e tolerâncias geométricas*. Features-de-forma são classificadas em *paramétricas* e *não-paramétricas*, que por sua vez classificam-se *simples, compostas, macro e explícitas*. As features-de-forma são combinadas espacialmente para definir a forma geométrica de uma peça. O elemento União representa a relação de adjacência existente entre duas features-de-forma pertencentes a uma peça, tanto em termos de elementos geométricos quanto de processos de fabricação envolvidos na ligação. Os elementos Dimensionamento e Tolerância descrevem as distâncias e ângulos entre as superfícies e eixos das features-de-forma, podendo ser utilizados no processo de análise e síntese de tolerâncias, bem como na própria seleção do processo.

- **Materiais**: entidades que representam *Tipo* e *Forma Comercial* de cada material. O *Tipo* descreve as propriedades mecânicas do material, que servem de base para as recomendações de processos de fabricação mais adequados. *Forma Comercial* descreve as condições disponíveis de obtenção do material (blocos, barras, vergalhão, etc), ao mesmo tempo em que pode servir de ligação com as informações de controle de estoque.

- **Processos de Fabricação**: essas entidades estão sub-divididas em *Processos de Definição de Forma* e *Processos de Tratamento*. Processos de Definição de forma estão sub-divididos em três classes : processos para a produção de peças discretas (Forjamento, Usinagem), processos de união permanente de peças discretas (soldagem, colagem) e Montagens. Processos de Tratamento referem-se à tratamentos térmicos e de superfície.

Uma das particularidades deste modelo é que o conceito de *feature* é generalizado a qualquer característica informativa de um componente do produto com representação física ou não. Outra particularidade é a existência de uma *feature* virtual que define a forma como as diferentes features de forma são colocadas - a feature **união**. Esta feature é a que permite a definição da topologia da peça ou montagem de peças.

2.8.2. O Sistema para tratar o Modelo

O sistema desenvolvido para cumprir os objetivos da tarefa foi dividido em dois módulos : ***Gráfico*** e ***Abstrato***. O módulo gráfico, também referenciado como sistema de auxílio ao projeto, é responsável pelo ambiente gráfico-interativo de entrada das informações de projeto. O módulo abstrato, que congrega todas as outras aplicações contempladas no modelo, é responsável pelo recebimento das informações oriundas do módulo gráfico e pela execução dos raciocínios exigidos sobre essas informações. Segundo Chang [Lin92] e Shah [Shah91a], a representação mais adequada para modelar as informações dentro de um sistema de auxílio ao projeto quando necessita-se automaticamente extraí-las para diversos aplicativos, é a representação baseada em features. As features possuem representações distintas nos dois módulos, porém mantêm entre si uma integridade conceitual rigorosa que garante a troca e tratamento de informação entre eles, trabalhando-se unicamente o formato dos dados.

A representação baseada em features é traduzida ao nível do módulo gráfico pela abordagem de *Síntese de Volumes Elementares*, dentro do *projeto por features* (ou *Modelamento Construtivo*, segundo a classificação de Shah). Portanto, as primitivas básicas oferecidas ao projetista são formas volumétricas sólidas elementares que incorporam informações e atributos relevantes às demais aplicações (como seleção de processos e materiais), e são arranjadas espacialmente segundo um conjunto de regras e relações. Ao nível do módulo abstrato, a representação é traduzida pela criação de bases de dados de features orientadas por objetos, onde todo o conhecimento relativo a cada uma das aplicações deve ser precisamente modelado em termos de regras e fatos que possam atuar sobre tais bases de dados. O módulo abstrato consegue dessa maneira, usufruir de todos os benefícios implícitos do paradigma de orientação por objetos, e em especial, dos mecanismos de herança.

O módulo gráfico trabalha sobre um modelador de sólidos e permite que o produto seja concebido em termos das peças que o compõe. As peças são formadas por conjuntos de features volumétricas, paramétricas e variacionais que se inter-relacionam espacialmente segundo regras definidas em termos de posicionamento, orientação, inter-dependência, processos e superfícies de ligação inter-features, bem como de restrições inerentes à manipulação dessas features. As features oferecidas como primitivas básicas de criação são orientadas à fabricação e visam atender o conjunto de aplicações tratadas pelo modelo, que são dependentes do projeto do produto. Portanto, além das especificações geométricas, esse módulo recebe informações topológicas, de material e de processos referentes às peças modeladas.

O módulo abstrato trabalha com um conjunto de features definidos compativelmente com aquelas reconhecidas pelo módulo gráfico, contemplando os requisitos das seguintes aplicações : *seleção de processos de fabricação, tecnologia de grupo, verificação de projeto e avaliação de manufaturabilidade*. Esse módulo baseia-se em conhecimentos - representados por *regras* e *fatos* - para sugerir processos de fabricação e reprojetos para as peças, operando sobre uma base de dados de features orientada por objetos contendo as informações necessárias ao raciocínio exigido pelas aplicações.

Este módulo representa a parte do modelo onde estão contidos os conhecimentos necessários para apoiar o projetista. Trata-se de fato de uma base de conhecimento estruturada com

regras, **fatos** e ***frames*** associados a um **motor de inferência**, que é o responsável pelo desencadeamento dos diferentes raciocínios. Existem vários conjuntos de regras e fatos em função das diferentes aplicações existentes no sistema, designadamente para sugestão de processos de fabricação, para codificação (TG), análise da manufacturabilidade, sugestão de materiais, etc.

Uma interface direta permite que questionamentos sejam dirigidos ao módulo abstrato, a partir do módulo gráfico, o que permite obter simultaneamente análises e sugestões pertinentes aos problemas de especificação da peça e do produto sendo trabalhado. Isso dá ao projetista maior flexibilidade e amplitude de criação. Como todo o conhecimento existente no módulo abstrato está representado através de fatos e regras, é muito fácil configurar esse conjunto de informações para se obter a melhor ajuda possível por parte do módulo, sem obrigar o projetista a possuir grandes conhecimentos a nível de fabricação e montagem.

Como já dito anteriormente, a base de representação das informações são as entidades denominadas *features*. Para o escopo desse trabalho, features são definidas como ***um conjunto de informações ou entidades usadas em atividades de projeto ou fabricação, dependentes do modelo e do contexto.*** Essa definição deixa claro que feature ***não*** é sinônimo de feature-de-forma, visto que features podem ser entidades sem qualquer representação física ou forma geométrica. Entretanto, convenciona-se nesse documento que o termo *feature* quando usado isoladamente significa *feature-de-forma*, e que as outras classes de features serão sempre explicitadas (por exemplo, *feature-união, feature-de-material, feature-de-processos*). Esse documento tratará mais especificamente das discussões envolvidas com o módulo gráfico onde as features mais relevantes são as features-de-forma.

As features-de-forma estão agrupadas em duas classes de bibliotecas : uma ***canônica***, que contém um conjunto de primitivas sólidas oferecidas pelo ACAD/AME acrescidos dos atributos de caracterização da natureza das primitivas, e os padrões de features (que são grupos de feature que obedecem relações matemáticas de construção); e outra ***aberta***, a qual permite a definição de features genéricas, baseadas nas features canônicas.

A comunicação do sistema com quaisquer outras aplicações se dará segundo protocolos específicos concebidos para atender aplicações particulares, sem no entanto perder de vista o objetivo maior em termos de troca de dados que é a adoção da norma 48, do PDES/STEP. Ou seja, a concepção dos procedimentos de troca de informação do sistema para/com outros sistemas deve ser concebida o mais flexível possível, a fim de permitir sua fácil e confortável adaptação aos protocolos do STEP.

Maiores detalhes conceituais e de implementação do modelo serão discutidos nos capítulos 3 e 4 deste documento.

2.9. Contribuição do modelo na resolução das limitações apresentadas

O modelo trata em termos de features as entidades mais representativas existentes na concepção de uma peça genérica. Portanto, visa tratar de um conjunto específico de aplicações integradas do processo completo de produção de um produto, reportando problemas e apresentando soluções para problemas novos e/ou já detectados antes, em pesquisas da mesma natureza. Basicamente, os seguintes pontos de contribuição podem ser destacados :

- Integração de informações referentes a material, processos, geometria e topologia dentro de uma única abordagem, baseada em features.

- Associação e tratamento de características de material e processo ainda na fase de projeto, bem como considerações de fabricação e avaliação de manufaturabilidade.
- Tratamento de uma biblioteca aberta e genérica de features, baseada em restrições e regras de relacionamento inter-feature.
- Experiência com plataforma comercial e de pequeno porte como suporte de desenvolvimento (ACAD e computadores linha PC 486) para aplicações integradas baseadas em features.
- Validação do modelo conceitual com vistas a utilizá-lo em um ambiente mais completo e poderoso, segundo os conceitos de engenharia concorrente suportando modelamento integrado de produtos.
- Consideração precoce de manufaturabilidade e dos processos durante a fase de projeto.
- Tratamento dos códigos de Tecnologia de Grupo definidos dinâmica e contextualmente, através de regras de derivação e atribuição automática de quaisquer tipos de código, consoante as características das peças, sem perder a forma tradicional de tratamento dos códigos específicos associados às famílias de peças.

2.10. Integração Baseada em features. Formatos neutros. Step e Express

As diversas áreas envolvidas na produção de um produto lidam com a mesma informação básica, porém segundo sua visão particular, o que ficou caracterizado nas diferentes definições de features. Ou seja, o desenvolvimento de pesquisas baseadas em feature carregam consigo uma forte componente de integração, que irá depender da capacidade de representação e mapeamento das visões específicas das features. A representação de features é um passo necessário para superar a distância existente entre os sistemas computacionais que auxíliam várias áreas da engenharia, e um importante passo rumo a uma verdadeira Manufatura Integrada por Computador (CIM) [Klauck93].

Quando áreas distintas de aplicação tentam integrar-se, normalmente os problemas se originariam na capacidade de transformação das informações consideradas em cada uma das áreas para tornã-las acessíveis as outras correlacionadas. A propagação de alterações em informações compartilhadas dentro de um modelo pode ser extremamente difícil. A replicação das informações particularizadas a cada aplicação requer um mecanismo de atualização global completo e sofisticado, a fim de manter a integridade total dos dados trabalhados.

Por outro lado, a centralização de informações dentro de um modelo primário exige que, além da completude necessária desse modelo (que pode ir desde dados geométricos até custos de ferramentas e prazos de entrega, por exemplo), mecanismos de disponibilização dessas informações a cada aplicação sejam construídos, o que parece ser uma perpectiva factível. Dentro dela, e considerando que as diversas áreas cooperantes sejam suportadas por sistemas baseados em features, uma arquitetura ideal de solução dos problemas de integração poderia ser a apresentada na figura 2.9.

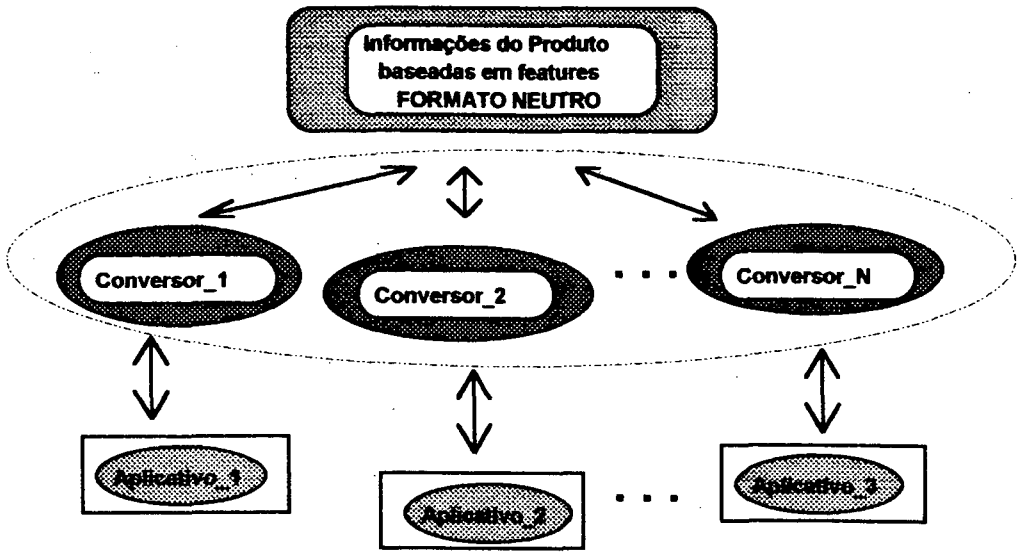


fig. 2.9 - Arquitetura Integrada para Aplicativos baseados em features

Muitos problemas estão implícitos nessa arquitetura tida como ideal, dos quais o mais grave é a representação de todas as informações de um produto em um único formato, a partir do qual essas informações possam ser extraídas pelos conversores, trabalhadas pelos aplicativos e atualizadas consistentemente. Esse processo exige uma representação uniforme dos dados no dito formato neutro, que é a única maneira de permitir o funcionamento determinístico dos conversores.

Recentemente muitos esforços tem sido direcionados para a afirmação das normas sugeridas pelo PDES/STEP [Lis92][Qiao93][Tavares93], com especial interesse voltado aos trabalhos referentes às features. Parece que, dada a diversidade de definições e áreas de pesquisas envolvidas com features, a proposta da norma 48 pelo menos centraliza trabalhos rumo a uma padronização de linguagem e, consequentemente, a um caminho mais natural de integração. Portanto, a dedicação de novas pesquisas englobando esse protocolo e verdadeiramente integrando aplicações diversas dependentes do projeto de um produto, são vitais para a sua consolidação.

Capítulo 3

O Modelo UNINOVA/CESO

O objetivo primário do modelo concebido é suportar conceitualmente uma estrutura de informações que sirva de base para um sistema de apoio a decisão, que permitirá incluir considerações de fabricação durante a fase de projeto de um produto. Isso caracteriza um sistema de *projeto-para-fabricação* onde pode-se criar e avaliar diferentes alternativas de produção para um dado produto. O conhecimento que deve ser incorporado e tratado ainda na fase de projeto de um produto é oriundo, nesse modelo, do domínio das seguintes aplicações: *seleção de processos de fabricação, tecnologia de grupo, verificação de projeto e avaliação de manufaturabilidade.*

O modelo conceitual, que foi brevemente discutido no capítulo anterior, será mais detalhado nesse capítulo, em termos das entidades representativas desse modelo no contexto das aplicações consideradas. Como citado no capítulo 2, o sistema para suportar o modelo foi dividido em dois módulos : *Gráfico* e *Abstrato*, e a representação das informações é baseada em features. Os módulos devem possuir representações de features rigorosamente compatíveis, para permitir a integração das aplicações. Para o escopo desse modelo, features foram definidas como *um conjunto de informações ou entidades usadas em atividades de projeto ou fabricação, dependentes de modelo e contexto.*

No módulo gráfico, as features sempre serão entidades geométricas (volumétricas ou não) acrescidas de atributos caracterizadores das necessidades das aplicações dependentes. No módulo abstrato as features não possuirão obrigatoriamente informações geométricas, podendo em alguns casos serem formadas de atributos puramente textuais.

O objetivo desse documento é discutir mais profundamente o desenvolvimento do módulo gráfico: representação das features, classificação, forma de implementação, problemas e limitações, conclusões. O módulo abstrato é assunto de uma tese de doutoramento, que se encontra em fase de conclusão junto ao Departamento de Engenharia Mecânica da Faculdade de Ciências e Tecnologia, da Universidade Nova de Lisboa. Portanto, todas as considerações mais profundas pertinentes ao módulo abstrato serão devidamente referenciadas àquela tese.

A discussão dos próximos itens ainda englobarão o modelo como um todo, onde serão definidas as características globais do modelo e a arquitetura proposta para suportá-lo. A partir desse item a discussão será direcionada explicitamente para o módulo gráfico.

3.1. Características Principais do Modelo

Dentre os características mais importantes do modelo, enumeram-se os seguintes:

- Contemplar o princípio de *projeto-para-fabricação*, onde as considerações dos seguintes aplicativos devem ser consideradas ainda na fase de projeto: *Seleção de Processos, Tecnologia de Grupo, Verificação de Projeto e Avaliação de Manufaturabilidade.*

- Representação das informações presentes no modelo *baseada em features*, tendo escolhido o método de *projeto por features* suportado na abordagem de *Modelamento Construtivo (síntese de elementos volumétricos)* para descrever a configuração da peça. O conhecimento referente aos principais elementos que figuram no projeto e fabricação é representado através de regras e fatos que operem sobre uma base de dados orientada por objetos.

- Suportar o problema dos conjuntos específicos de features através da definição de bibliotecas de features, configuráveis.

- O modelo não é um gerador de soluções para a integração de aplicações baseadas em features, mas sim concebido sob a perspectiva de auxílio ao Projeto para a Fabricação, integrando um grupo específico de aplicações.

- Garantir a factibilidade de um produto projetado, através de restrições e validações de naturezas funcionais, geométricas, topológicas e de material, devendo inclusive sugerir reprojetos ou alterações.

- Alto nível de abstração na captura das intenções do projetista.

3.2. Arquitetura

O modelo está dividido em dois módulos principais, conforme mostrado na arquitetura da figura 3.1, que são o *Módulo Gráfico* e o *Módulo Abstrato*, este último formado pelas *Aplicações* e pelo *Sub- Sistema Abstrato*. Como a representação das informações existentes no modelo é baseada em features, ambos os módulos (Gráfico e Abstrato) possuem modeladores que devem tratá-las segundo uma visão compatível entre as diversas aplicações. As características globais do modelo serão apresentadas a seguir, fazendo-se as considerações devidas a cada módulo.

Essa arquitetura suporta os requisitos essenciais para o desenvolvimento de um sistema de projeto-para-fabricação. As features relevantes às aplicações são modeladas na base de dados do módulo abstrato, contemplando as visões específicas de cada uma delas. As features relevantes à definição de forma, geometria e topologia do produto são trabalhadas primariamente no módulo gráfico, que deve incorporar ao seu conjunto de features, atributos que satisfaçam as necessidades funcionais das aplicações contidas no módulo abstrato.

As informações referentes aos produtos concebidos devem ser trocadas entre os módulos da maneira mais automática possível, para validar o conceito de projeto-para-fabricação. Isso significa que deve existir um formato particular de troca de informações, especificamente definido para compatibilizar as representações dos os módulos. As considerações de fabricação devem ocorrer necessariamente nos estágios iniciais do projeto, a fim de direcioná-lo para a melhor solução.

O formato de troca de informações entre os dois módulos, concebido para atender exclusivamente as necessidades de informação daqueles módulos, está definido como um arquivo-texto contendo linhas do tipo *<IDENTIFICADOR_INFORMAÇÃO* ***VALOR***>. Um exemplo pode ser visto no anexo **3**.

3.3. O Módulo Abstrato

Esse módulo é composto pelas *Aplicações* e pelo *Sub-sistema Abstrato*. O elo de ligação entre as aplicações e o sub-sistema abstrato é uma base de dados orientada por objetos, a qual contém uma descrição hierárquica das features relevantes ao módulo em forma de árvore, representando todas as entidades presentes no modelo conceitual. As aplicações seguem os conceitos de inteligência artificial, suportadas por bases de regras e de fatos.

O módulo abstrato deve receber do módulo gráfico os dados referentes á cada uma das peças a serem analisadas pelas aplicações, devolvendo ao módulo gráfico o resultado das análises, as quais variam desde a recomendação de processos de fabricação até reprojetos de elementos contidos nas peças.

Todas as informações a respeito desse modulo encontram-se devidamente descritas em Cunha [Cunha94], na sua tese de doutoramento.

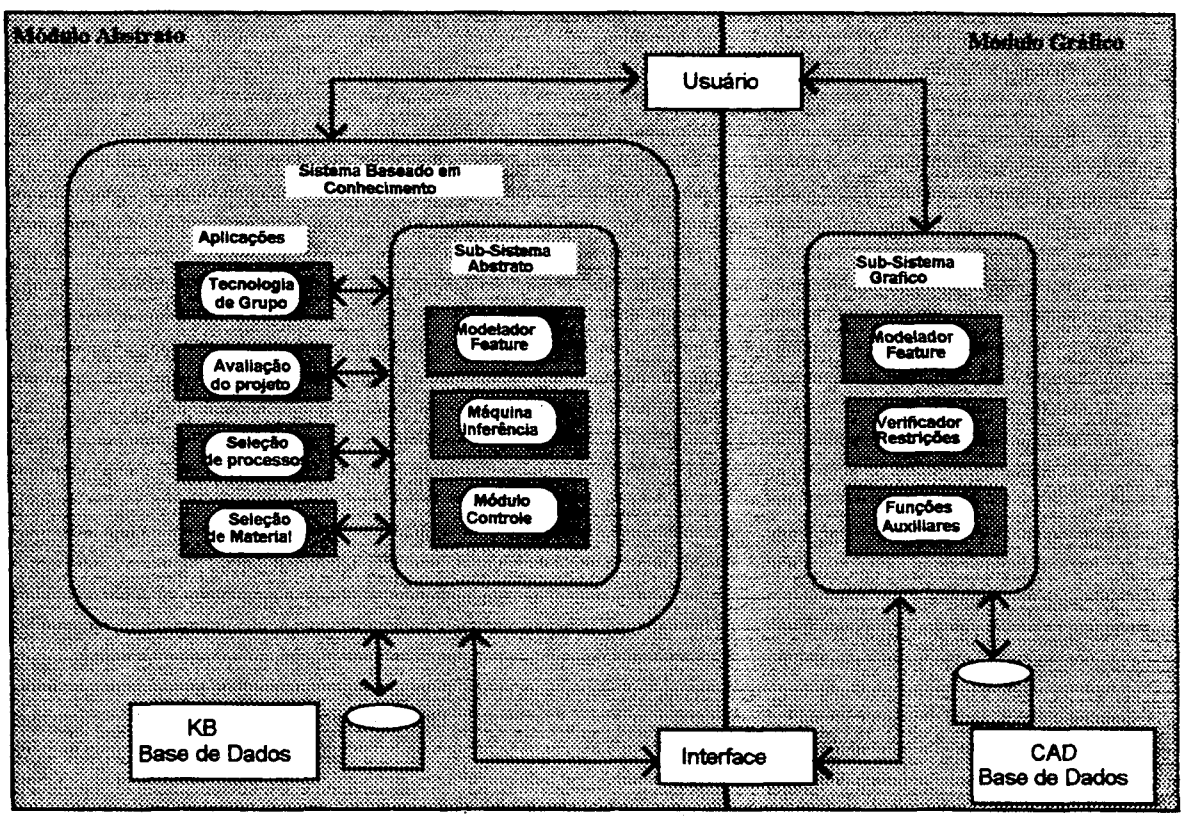


fig. 3.1 - Arquitetura do Sistema concebido para o Modelo

OBS. A simbologia usada neste diagrama, excessão feita aos símbolos tradicionais de Banco de Dados, representam basicamente módulos funcionais dentro da arquitetura, identificados por mnemônicos significativos.

3.4. O Módulo Gráfico

O módulo gráfico, além de atender as especificações do modelo conceitual do projeto como um todo, contempla o seguinte conjunto de requisitos :

- Ser uma ferramenta de visualização e manipulação da geometria dos projetos a um nível apropriado de detalhe, o que significa mostrar e representar formas 3-D relativamente complexas, além de gerenciar entidades sólidas consistentes com essas formas.
- Fornecer um conjunto padrão de features e permitir a criação de conjuntos de features específicos às aplicações.
- Oferecer um conjunto básico de ferramentas de edição das features, o que envolve a implementação de funções para criação, eliminação, modificação e análise do produtos.
- Fornecer relações que permitam a criação de peças mecânicas complexas, através do agrupamento das features.
- Permitir a incorporação, aos atributos das features, de informações relativas à fabricação das peças, nomeadamente : especificação de material, de processos de fabricação, de códigos de classificação para Tecnologia de Grupo e de dispositivos de fabricação. Esses atributos são a base de funcionamento do módulo Abstrato.

O módulo gráfico é responsável pelo ambiente para criação das peças componentes de um produto, onde são fornecidas as informações relativas à geometria, topologia, material e processos utilizados nos produtos. Esse módulo está funcionalmente dividido em três sub-módulos : um ***Modelador de Features***, o qual terá nas features as suas primitivas básicas de representação de informação; um ***Verificador de Restrições*** responsável pela consistência das informações geométricas, topológicas e de fabricação, das peças construídas. Por último, um grupo de ***Funções Auxiliares*** que congregam todas as funções que devem ser oferecidas pelo módulo gráfico, para que o seu funcionamento seja o mais confortável e completo possível para o usuário.

O Modelador de Features é o elemento principal do módulo gráfico, responsável pelo fornecimento das primitivas de criação dos produtos, que são as features. Esse modelador, valendo-se dos procedimentos oferecidos pelo sub-módulo Funções Auxiliares, oferece ao projetista um ambiente de trabalho gráfico-interativo de alto nível.

O Verificador de Restrições valida as informações de processos e manufaturabilidade através de questionamentos específicos ao módulo abstrato, de tal maneira que o projetista pode ter acesso a considerações de fabricação durante a fase inicial do projeto.

3.4.1. Entidades Gráficas

Como já dito anteriormente, o módulo gráfico deve manter coerência e consistência com o modelo conceituai global. Assim, as entidades trabalhadas no módulo foram concebidas visando primordialmente garantir essa coerência. As figuras 3.2 e 3.3 mostram hierarquicamente essas entidades.

O *modelo* é a entidade representativa de um produto como um todo, possuindo todas as informações geométricas e tecnológicas relativas a esse produto, seja ele uma peça simples ou uma montagem de peças. Um modelo completo é formado por *peças, features livres e informações genéricas*. A entidade *meta-peça* possui um caráter auxiliar temporário, o que significa que ela não pertence a composição final de um produto.

As *peças* são agrupamentos de elementos tecnológicos, geométricos e topológicos. Os elementos tecnológicos representam as informações de material, processos e máquinas. Os elementos geométricos são baseados nas features, features livres e macro-features. Os elementos topológicos são representadas por um conjunto de relações que definem o comportamento das ligações inter-features (de posicionamento, de adjacência e de dependência).

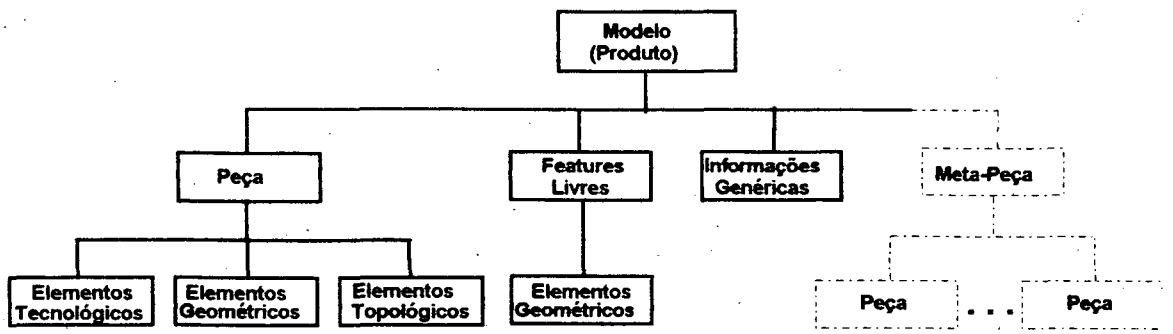


fig. 3.2 - Hierarquia das Entidades Gráficas - 1º Nível

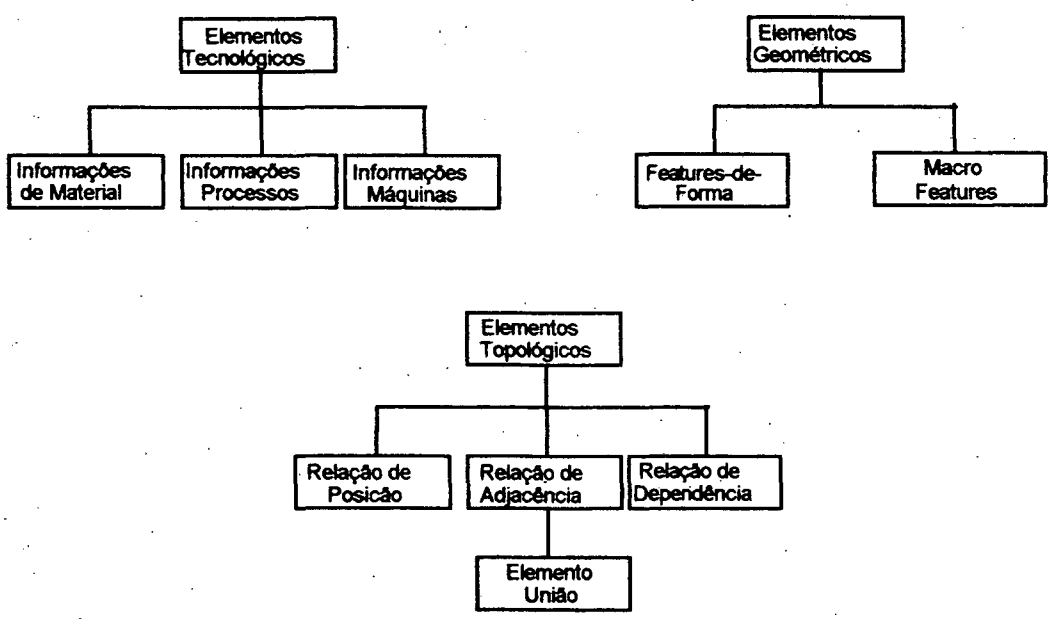


fig. 3.3 - Hierarquia das Entidades Gráficas - 2º Nível

Os *elementos tecnológicos* referem-se ao material (informações de tipo e forma comercial disponível), aos processos de fabricação (definição de forma e tratamento) e às máquinas (existentes) associados a uma peça. Eles servem apenas como parâmetros de questionamentos dirigidos ao módulo abstrato, e não existe qualquer tratamento gráfico associado a eles.

Os *elementos geométricos* possuem nas features sua principal entidade de representação, as quais fornecem as informações geométricas da peças baseando em primitivas sólidas. As features são classificadas e discutidas com mais propriedade e detalhe nas próximas secções. Uma característica importante das features é a possibilidade da incorporação direta de elementos tecnológicos sobre os elementos geométricos de baixo nível das features, como as faces.

Os *elementos topológicos* são representados por um conjunto de três relações que servem de guia para a definição da topologia das peças contruídas. A adição das features à uma peça segue as regras definidas nessas relações. Resumidamente, a relação de posicão fornece as regras para o posicionamento e orientação espacial das features; a relação de adjacência fornece as regras para a justaposição das features, o que traduz-se por um existência de um elemento de *união* entre as features, e a relação de dependência fornece as regras para o inter-dependência existencial que pode ocorrer entre as features. As relações serão descritas mais detalhadamente nos próximos itens.

As *features livres* foram criadas para permitir a existência de elementos auxiliares (por exemplo, linhas de centro) e a existência de features (canônicas ou abertas) isoladamente. A característica marcante dessas features é que o número de elementos pertencentes a elas é sempre muito reduzido (1 ou dois elementos, no máximo), o que enfatiza o caráter auxiliar de construção dessas features.

As *macro-features* representam features criadas pelo próprio usuário, segundo suas necessidades específicas. Essas features são constituídas por features ligadas topologicamente segundo as relações de posição, adjacência e dependência, e são agrupadas em bibliotecas ditas *bibliotecas abertas*.

As *informações genéricas* representam informações de natureza diferentes das representadas pelas outras entidades existentes no modelo, por exemplo, as informações de caráter administrativo.

As *meta-peças*, que estão representadas de uma maneira distinta das outras entidades na figura 3.2., são entidades temporárias constituídas de um grupo de peças. As meta-peças são criadas com um único propósito: o de permitir que operações geométricas sejam realizadas simultaneamente sobre várias peças. Por esse motivo, elas não fazem parte da estrutura final de informações do produto.

3.4.2. O Modelador de Features

Núcleo central do módulo gráfico, o Modelador de Features tem nas features a sua primitiva básica de representação das informações relativas aos produtos e suas peças componentes. As features são elementos que incorporam diretamente sobre a geometria e topologia de primitivas sólidas, atributos de material e processos de fabricação. Elas são agrupadas segundo um conjunto de relações, a fim de constituirem as peças componentes de um produto.

3.4.2.1. Features-de-Forma

Como a abordagem escolhida para o projeto por features foi a *síntese de elementos volumétricos*, as features são representadas por primitivas volumétricas fechadas cujas

superfícies ou faces componentes são univocamente identificáveis por exigências dos aplicativos do módulo abstrato, pois assim é possível associar quaisquer atributos às faces.

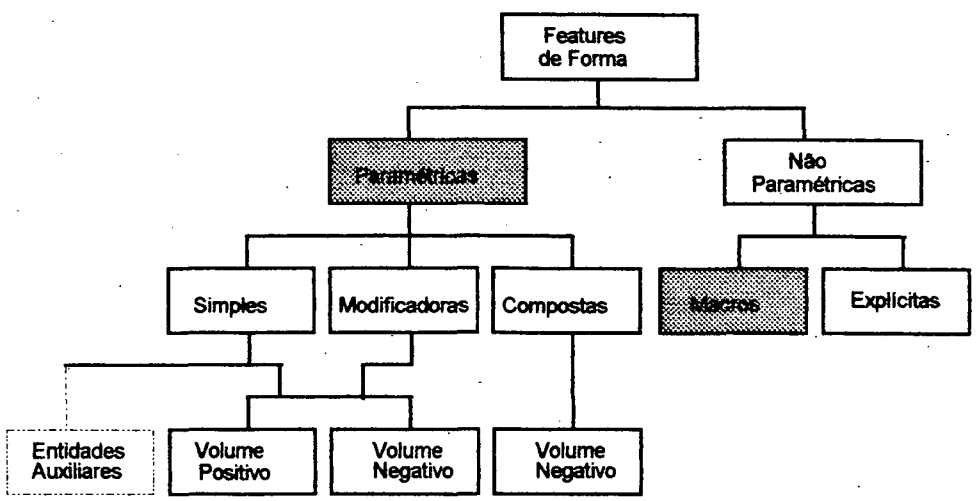
As features associam atributos representativos das informações existentes no modelo conceitual, diretamente às primitivas geométricas. Esses atributos somente serão utilizados no módulo abstrato.

3.4.2.1.1. Classificação

As features estão classificadas primariamente conforme apresentado na figura 3.4., em *Paramétricas e Não-Paramétricas*.

As features *Paramétricas* são aquelas que podem ser instanciadas através de um ou mais conjuntos de parâmetros geométricos, e sub-dividem-se em Simples, Modificadoras e Compostas.

As features *Simples* são as primitivas volumétricas básicas (paralelepípedos, troncos de cone, cilindros, etc). As features *Modificadoras*, como originalmente concebidas, representam alterações na geometria e na topologia de uma primitiva geométrica básica (concordâncias e chanfros). Por restrições impostas pela plataforma de implementação, esse conceito foi adaptado para features de volume positivo e negativo, o que será melhor detalhado no capítulo 4. As features *Compostas* são aquelas formadas por conjuntos de features simples, agrupados segundo parâmetros e relações matemáticas. Nesse trabalho, as features compostas são os chamados *padrões* de features, que são conjuntos circulares ou matriciais de cilindros de volume negativo.



OBS. As features paramétricas e as não-paramétricas-macros são chamadas ***Features Implícitas.***

fig. 3.4 - Classificação Primária das Features-de-forma

As features *Não-Paramétricas* são definidas ou através de conjuntos de entidades geométricas de baixo nível explicitamente identificadas (*explícitas*), ou através de conjuntos específicos de features paramétricas (*macros*). Um furo pode ser definido tanto através de um conjunto de parâmetros constituído de raio e profundidade, como através de suas superfícies lateral, de entrada e de saída. Do 1º modo tem-se uma definição paramétrica do furo, enquanto que no 2º tem-se uma definição não-paramétrica explícita.

A definição explícita porém exige que as entidades geométricas que formarão a feature tenham sido previamente construídas, o que foge ao escopo desse trabalho. Portanto, as features não-paramétricas a serem consideradas serão somente as abertas, uma vez que não existirão no modelo entidades geométricas não-pertencentes a uma feature qualquer, e portanto passíveis de serem incorporadas às features explícitas.

Como mostrado na figura 3.4, as features paramétricas e as não-paramétricas-macros podem ser classificadas como *implícitas*, pois são implicitamente definidas através dos parâmetros caracterizadores ou através do agrupamento de features.

Os elementos de volume positivo e negativo são derivados de uma característica de *natureza* existencial de cada feature e naturalmente voltados às operações de usinagem, onde *volume positivo* significa que a feature possui um sólido representativo com existência independente de quaisquer outras features e *volume Negativo* significa a existência de uma dependência com uma ou mais features positivas, visto que as features negativas somente existem se contidos em uma ou mais features de volume positivo. Traçando um paralelo com os processos de fabricação, as positivas significam presença de material enquanto que as negativas significam remoção. No contexto desse trabalho, as features de volume negativo também são referidas como simplesmente *features negativas*, o mesmo aplicando-se às de volume positivo.

Ainda referente à *natureza* das features, a existência de features tipo *furos, rasgos, cavidades*, ou genericamente as *depressões* (como classificadas pelo FFIM) necessitam dessa característica para sua definição. Um furo é um cilíndro de volume negativo contido em uma ou mais features de volume positivo, e assim por diante.

As *Entidades Auxiliares* contemplam os elementos geométricos não volumétricos, usados como auxílio na criação de peças, como linhas de centro, planos de referência, etc.

3.4.2.1.2. Representação das Informações das Features

Cada feature paramétrica é geometricamente definida por um conjunto de faces que delimitam um volume fechado. Todas as faces possuem identificadores únicos dentro de uma feature, de tal maneira que cada face pode receber atributos específicos a ela. Além disso, cada feature possui um sistema de coordenadas que lhe é particular chamado FCS *(Feature Coodinate System* - Sistema de Coordenadas da Feature*)*, que define sua orientação e posicionamento espacial.

As features associam diretamente às primitivas geométricas, atributos representativos das informações existentes no modelo conceitual e que somente serão utilizados no módulo abstrato. Esses atributos estão representados na figura 3.2 (Hierarquia das entidades gráficas) pelos *elementos tecnológicos*, e a nível de features basicamente são identificadores de tipo e código comercial do material do qual a feature será fabricada.

Existe ainda um sistema de coordenadas associado a cada uma das faces de uma primitiva, de tal maneira que as definições geométricas inerentes à uma face podem ser trabalhadas sempre em um plano XY. Tal sistema será extremamente útil na construção das peças, quando da inserção das features.

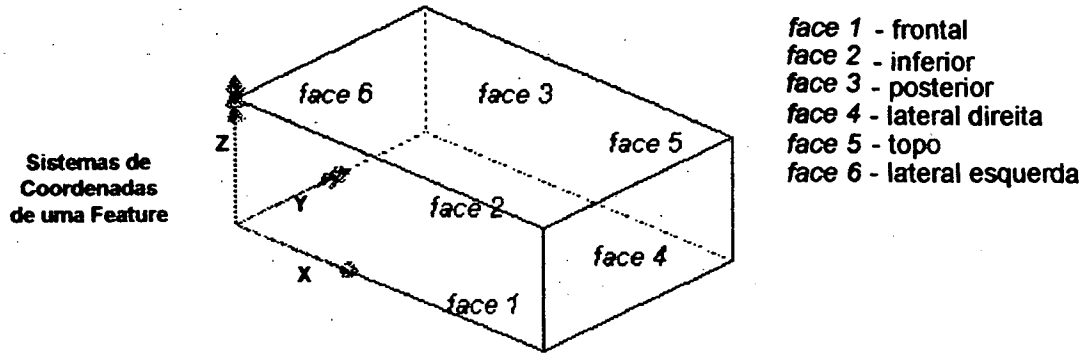


fig. 3.5 - Representação Geométrica

Uma consideração adicional é feita ainda para as features negativas. Como citado no item anterior as features negativas significam uma remoção de volume, e como tal, dependem existencialmente de uma feature positiva. Portanto, o posicionamento e a adjacência entre uma feature negativa e uma positiva se dá sobre faces que se intersectam e serão removidas. Essa face de contato da feature negativa é denominada *face de entrada* e a ela podem ser associadas regras auxiliares para, por exemplo, aproximação de ferramentas durante a usinagem. Logo, uma feature negativa quando posicionada sobre uma positiva, adiciona faces que não existiam na entidade e elimina regiões de faces já existentes.

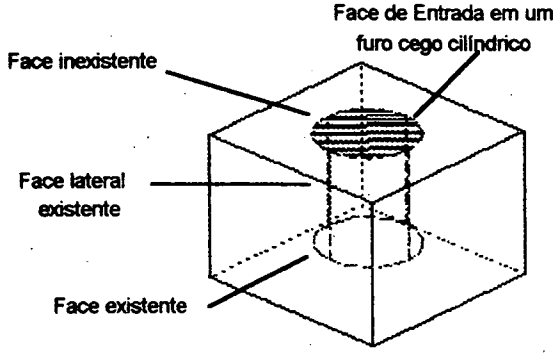


fig. 3.6 - Faces em uma feature negativa

3.4.2.2. Relações Inter-Features-de-Forma

Para definir a forma segundo a qual as features serão interligadas durante o processo de criação das peças (simples ou complexas), existem basicamente três tipos de relações que são: *Posicionamento, Adjacência e Dependência*. A relação de Posicionamento rege a colocação espacial de cada feature, levando em consideração tanto o sistema de coordenadas relativo à peça a qual essa feature pertencerá, como um sistema de coordenadas absoluto; a relação de Adjacência trata da justaposição e contiguidade das features que compõe uma peça, tanto a nível geométrico quanto de processos de fabricação; por último a relação de Dependência trata das intersecções existentes entre features negativas e positivas.

A relação de posicionamento está baseada geometricamente em uma *matriz de transformação homogênea*, onde consegue-se definir o posicionamento e orientação espacial de cada primitiva sólida. Logo, cada feature possui um atributo representativo dessa matriz, o que permite posicioná-las em relação a uma feature vista como *referência* dentro de uma peça, ou mesmo em termos absolutos em relação a um sistema de coordenadas global.

A relação de adjacência rege a instanciamento das features durante a criação das peças, tratando da definição de faces e eixos de contato entre elas. Ou seja, toda nova feature é inserida em uma peça qualquer de maneira contígua a uma feature já existente, e essa contiguidade baseia-se nas faces a serem justapostas e em dois eixos, os quais definem dois pontos pertencendo a cada uma das faces que deverão ser sobrepostos no instanciamento. A relação de adjacência suporta também a existência do elemento *união*, o qual especifica o tipo de processo de ligação entre features adjacentes e serve de parâmetro de classificação do tipo de peça, segundo a classificação apresentada no capítulo 2, item 2.8.1

Para as features negativas, que significam a remoção do volume por elas definido, a face de adjacência da feature positiva que a suporta é também chamada *face de entrada*, uma vez que algum processo de usinagem para a remoção da referida feature será definido. Além do mais, a feature negativa é sempre posicionada ortogonalmente à uma das faces da positiva que a suporta, ou seja, o modelo não contempla o tratamento da definição de um *ângulo de entrada* para as features negativas.

A relação de dependência é definida sobre a relação existente entre uma feature negativa e um conjunto de features positivas, visto que as features negativas não existem isoladamente, segundo sua própria definição. Dessa relação nascem os conceitos de *feature-Mãe e feature-Filha*, onde *Filha* é sempre uma feature negativa e *Mãe* são todas as features positivas das quais uma negativa depende para existir. Exemplo elucidativo: na figura 3.7 tem-se a *FF4*, um cilíndro de volume negativo que está contido nas features *FF1* e *FF2*.. Logo, *FF4* é uma feature-filha do conjunto de features-mãe formado por *FF1* e *FF2*.

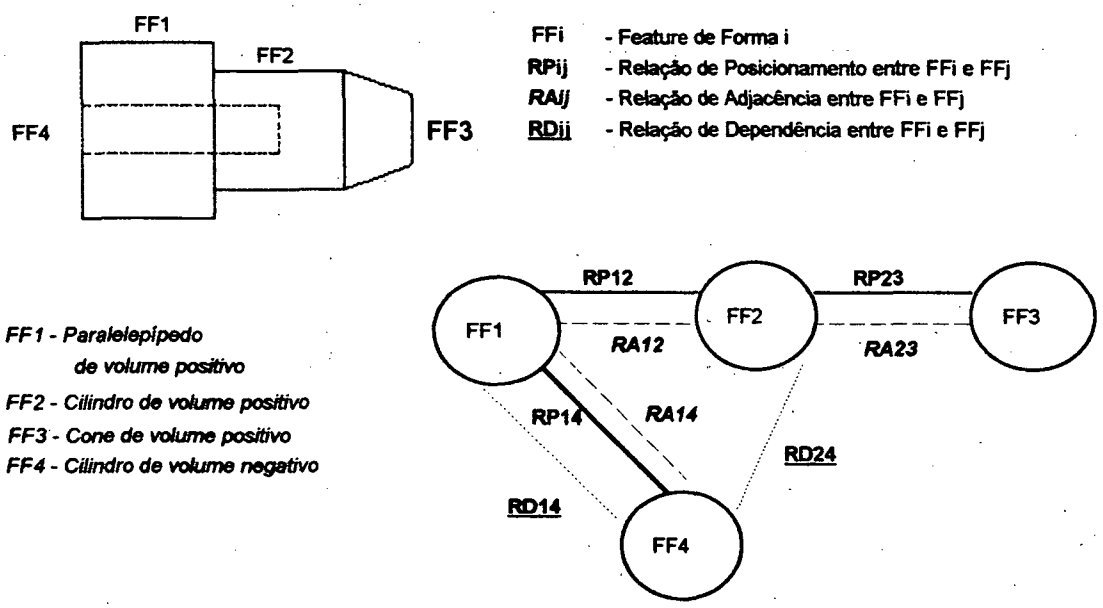


fig. 3.7 - Peça de exemplo geral das relações

Embutido nas regras de definição dessas relações, encontram-se algumas das validações efetuadas pelo módulo gráfico, principalmente aquelas ligadas aos aspectos topológicos e de fabricação das peças. As relações de posicionamento e adjacência garantem a contiguidade

física das peças uma vez que elas obrigam que as features pertencentes a uma peça sejam dispostas espacialmente de forma contígua e segundo entidades (primariamente faces) de contato explicitamente definidas, além da associação de um processo de definição de forma que valide essa ligação. A relação de dependência garante e retrata a factibilidade de operações de fabricação, pois por exemplo um processo onde haja remoção de material deve obrigatoriamente ser feito sobre features que produzam volume.

As regras que regem as relações são:

- Sempre existe uma relação de dependência entre uma feature de volume negativo e as features de volume positivo que ela intersecta.
- Não pode existir relação de posicionamento ou dependência entre duas features que não possuam uma relação de adjacência, ou seja, que não possuam faces de contato.
- Relações de adjacência vazias não são permitidas.
- Não pode existir intersecção entre duas features de volume positivo.
- Não existe relação de dependência entre duas features de mesma natureza (positivas ou negativas).
- O posicionamento de uma feature negativa é sempre perpendicular a uma face planar de uma feature positiva.

Para ilustrar melhor as relações, toma-se o exemplo da peça mostrada na figura 3.7, e o grafo representativo do conjunto das relações existentes na peça.

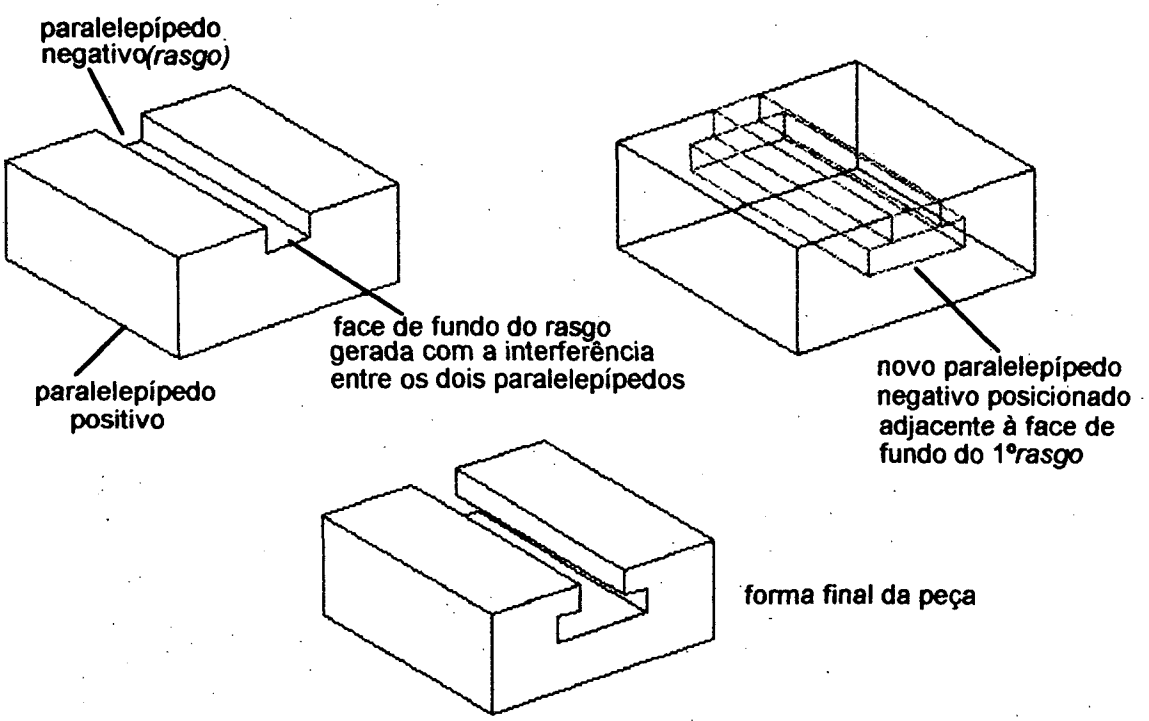


fig. 3.8 - Interferência entre features negativas e positivas

Um aspecto a considerar ainda pertinente às relações inter-features porém não formalizável nesse modelo, são as interferências entre features positivas e negativas. Tais interferências

criam formas inesperadas e que requerem um tratamento mais refinado [Mayer94], o que efetivamente não foi contemplado nesse trabalho. A única consideração implicitamente derivada das relações e naturezas das features é a aceitação da formação de novas faces oriundas daquelas interferências, as quais são reconhecidas e passíveis de serem utilizadas na construção do modelo do produto. Um rasgo escalonado é o exemplo desse conceito, quando uma feature negativa é posicionada adjacente a uma outra feature negativa, sobre uma face que foi *gerada* por uma interferência (fig. 3.8).

3.4.2.3. Bibliotecas de Features de Forma

As features agrupam-se em dois grandes conjuntos ou bibliotecas : as ***canônicas*** e as ***abertas***. As canônicas são representadas geometricamente pelas primitivas volumétricas sólidas básicas do modelador escolhido para suportar o módulo gráfico, sendo uma biblioteca padrão e não configurável. As features abertas são representadas geometricamente por agrupamentos de features criadas pelo usuário, segundo as relações de posicionamento, adjacência e dependência. A aplicação das relações na construção das features abertas lhes confere um certo grau de formalização, ao mesmo tempo em que garante a validade física do sólido representativo dessas features. Existirão tantas bibliotecas abertas de features quantas necessárias, enquanto que existirá sempre somente uma única biblioteca canônica.

Nesse trabalho, o conjunto de features canônicas oferecido é composto por: ***paralelepípedos***, ***cilindros***, ***troncos de cone***, ***cunhas***, ***fillets***, ***chanfros***, ***padrões circulares e matriciais de cilindros***. As features abertas são peças que serão usadas como features componentes de outras peças, o que significa que as features abertas são construídas formalmente sob a regência das mesmas regras que validam a criação das peças.

3.4.2.4. Features Livres

As features livres são formadas por features canônicas ou abertas, ou ainda de entidades geométricas auxiliares simples. Essas features são vistas como uma entidade indivisível, mesmo que elas sejam formadas por uma feature aberta complexa contendo muitas features. Primariamente, elas foram concebidas para contemplar a existência dos elementos auxiliares simples e permitir a existência isolada de features canônicas ou abertas, livres da obrigação de pertencerem a qualquer peça.

3.4.2.5. Features de mais alto Nível

As features de mais alto nível são construídas a partir das features primitivas, com seus atributos e características. Toma-se como exemplo a definição de furos cilíndricos tratada a um nível mais geral. Supõe-se que existem somente duas classes de furos a tratar: cegos e passantes. Ambos podem ser representados por cilindros de volume negativo contidos em features positivas, ambos possuem faces de entrada, ambos possuem diâmetro e profundidade. A profundidade do furo cego deve ser especificada nominalmente, enquanto do furo passante deve ser derivada de seus pais, ou seja, da relação de dependência.

Em outras palavras, a partir da definição das features primitivas, o modelo contempla facilmente a definição de mais alto níveis das seguintes features : *Furos, Rasgos, Cavidades, Ressaltos.*

3.4.3. O Verificador de Restrições

Da arquitetura do sistema proposto para contemplar o modelo conceitual, fica claro que a maioria das considerações de fabricação efetuadas durante a fase de projeto serão executadas pelo módulo abstrato. Basicamente, as informações tecnológicas recebidas pelo módulo gráfico serão repassadas sob a forma de questionamentos ao módulo abstrato, onde as restrições e/ou sugestões relativas à processos e fabricação serão efetuadas.

Entretanto, algumas restrições devem ser efetuadas pelo módulo gráfico. Elas referem-se sempre ao aspecto de validação geométrica dos modelos construídos ou ainda à validações funcionais que requeiram análises geométricas. Pode-se dizer também que as regras que regem a manipulação das features e suas inter-relações são constantemente validadas pelo módulo gráfico. As validações encontram-se espalhadas pelo modelador de features.

- *Restrição de Interpenetrabilidade de features de mesma natureza* : evitar que features positivas ou negativas se interpenetrem. O raciocínio efetuado pelo módulo abstrato exige que a topologia das peças seja a mais precisa, o que deixa de ser válido quando por exemplo, duas features positivas se intersectam.

- *Restrição de Inacessibilidade* : evitar que quando as faces de *entrada* das features negativas sejam obstruídas por outras features positivas.

- *Restrição Existencial 1*: não permitir que features negativas não estejam totalmente contidas por uma ou mais features positivas.

- *Restrição Existencial 2* : não permitir que a 1ª feature inserida em uma peça seja de natureza negativa, o que geraria uma situação paradoxal. As features negativas somente podem ser instanciadas após a instância de pelo menos uma feature positiva.

3.4.4. Funções Auxiliares

Far-se-ão necessárias muitas funções auxiliares para o tratamento do modelo, recuperação e armazenamento das peças e entidades existentes no projeto.

Para manipulação do modelo a nível global, fazem-se necessárias funções de: leitura, gravação, criação de novos modelos, consulta das entidades existentes em um modelo.

Para manipulação das peças fazem-se necessárias funções de: criação de uma peça nova, leitura, armazenamento, inclusão de features na peça (canônicas, padrões, e abertas), remoção de features, modificação dos atributos das features, operações geométricas sobre a peça (translação, rotação e escalonamento), remoção da peça, cópia, união de peças e features livres e definição dos parâmetros tecnológicos.

Para tratamento das features livres fazem-se necessárias funções de: criação de uma nova feature livre, leitura, armazenamento, inclusão de features (canônicas e abertas), operações geométricas (translação, rotação e escalonamento), modificação das features, remoção da feature livre, cópia e definição dos parâmetros tecnológicos.

Para tratamento das meta-peças fazem-se necessárias as funções de: montagem e desmontagem da meta-peça, inclusão e exclusão de peças, operações geométricas sobre a meta-peça (translação, rotação e escalonamento).

Para o tratamento das features abertas fazem-se necessárias as funções de: criação das bibliotecas abertas, criação das features abertas, inserção de features abertas nas bibliotecas, remoção de features abertas das bibliotecas.

Existe ainda um conjunto de funções ditas *de apoio* que mantêm relação direta com a implementação do modelo, em se tratando do conforto de utilização. Essas funções serão citadas no capítulo da implementação do modelo (cap. 4).

3.4.5. Conceitos Funcionais Básicos

Todo o trabalho desenvolvido nesse capítulo até o momento resumiu-se à descrição das estruturas representativas do modelo do ponto de vista do modelador gráfico, bem como suas inter-relações e conteúdos. Um outro aspecto importante a ser tratado é a funcionalidade conferida ao modelador na sua amplitude total a fim de fornecer o melhor aproveitamento das informações por ele manipuladas, o que será feito a partir das entidades gráficas suportadas.

3.4.5.1. Peças

Conforme citado no capítulo 2, item 2.8.1, as peças são classificadas segundo o tipo do processo de definição de forma que caracteriza a ligação existente entre as features componentes dessas peças. As peças *simples* (ou *discretas*) são definidas sobre um meio de material contínuo. As peças *complexas* são aquelas que necessitam de um processo de ligação (fixa ou amovível). As peças complexas por sua vez, subdividem-se em *compostas e agregadas*, segundo a natureza daquele processo de ligação : se o processo é permanente (por exemplo soldagem e colagem), as peças são ditas compostas; se o processo é uma montagem as peças são ditas agregadas. Faz-se necessário salientar que as peças complexas, vistas sob uma outra perpectiva, são conjuntos de peças simples interligadas por algum processo (figura 3.9).

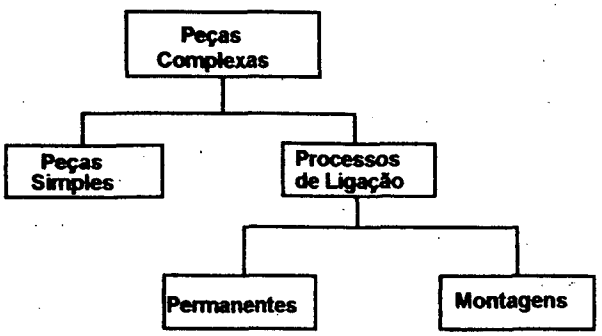
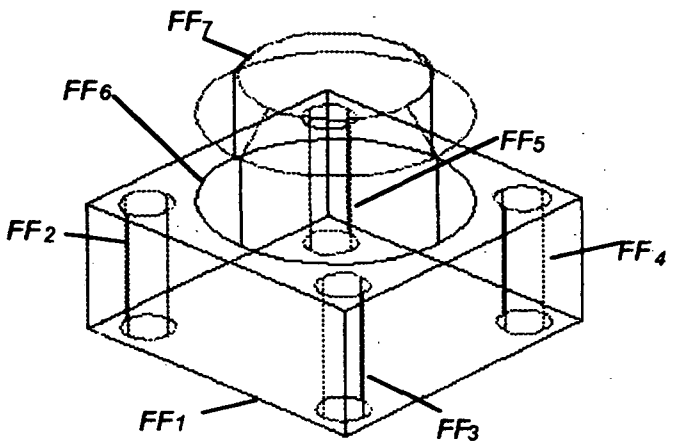


fig. 3.9 - Conceito definidor das Peças Complexas

No modelador gráfico, a entidade *peças* é composta por uma lista de features agrupadas segundo um conjunto de regras e relações. As peças possuem também atributos que definem informações tecnológicas, nomeadamente o tipo de material do qual elas serão fabricadas. Entretanto, alguns aspectos relativos à construção das peças deve ser ressaltados.



$$PEÇA = (FF_1 \cup FF_6) - (FF_2 \cup FF_3 \cup FF_4 \cup FF_5 \cup FF_7)$$

fig. 3.10 - Exemplo de uma Peça

Cada peça possui um sistema de coordenadas particular denominado **WCS** *(Workpiece Coordinate System)*, o qual é coincidente com o sistema de coordenadas da 1ª feature inserida na peça. Todas as operações geométricas efetuadas sobre uma peça serão relativas ao seu **wcs** particular.

3.4.5.1.1. Lista de Features

As peças são compostas por agrupamentos de features, as quais manipuladas segundo as regras e as relações vistas nas seções anteriores, definem a forma e a topologia de uma peça. Se considerarmos que as features positivas significam presença de volume (material) e que as features negativas significam remoção, uma peça é vista como:

$$Peça_i = (FF_{i1} \cup FF_{i2} \ldots \cup FF_{in}) - (FF_{in+1} \cup FF_{in+2} \ldots \cup FF_{im})$$

Onde FF_{i1} a FF_{in} representam features de volume positivo ou features abertas, e FF_{in+1} a FF_{im} representam as features de volume negativo (figura 3.10).

3.4.5.1.2. O Elemento União

Quando as features são inseridas em uma peça, elas são conectadas obrigatoriamente a alguma das features já existentes. Obviamente essa regra não se aplica a 1ª feature inserida na peça. A partir desta, entretanto, todas as features são incluídas tomando como referência uma feature já existente nessa peça. Essa referência é utilizada para definição das relações de adjacência e posicionamento. Para as features negativas, a referência serve também para a definição da dependência.

O elemento união é definido sobre faces e pontos de ligação. Cada feature fornece uma face e um ponto, sobre o qual incidirá um eixo perpendicular. Os eixos serão dispostos colinearmente, pois as faces da união serão postas em contato considerando os pontos definidos como pontos de contato. Assim, é possível definir um atributo que expresse exatamente o tipo de processo

de definição de forma envolvido nessa ligação. Cada peça possui uma lista de todas as uniões existentes entre as suas features componentes.

Entretanto, a fim de evitar que o projetista deva informar para cada feature de cada peça um atributo referente ao processo de ligação dessas features, as peças trabalhadas serão sempre peças simples pois a definição de material formador da feature foi elevada ao nível da peça. As peças complexas somente surgem no momento em que duas ou mais peças simples - feitas de materiais distintos ou não - forem explicitamente unidas, o que configura a necessidade incondicional de um processo de ligação nessa nova peça.

As faces envolvidas na união das features positivas de peças simples são classificadas em *parcial* e *virtual*, onde *parcial* é uma face de uma feature que é adjacente a uma face de outra feature, porém **não** em toda sua extensão. *Virtual* é aquela face que está totalmente contida em uma adjacência. Se a peça é simples e portanto construída sobre um meio contínuo, logicamente que a região de contato envolvida na adjacência inter-features não existirá fisicamente, o que permite a utilização dessa classificação das peças para fins de análise de seleção de processos.

No exemplo da figura 3.11., a face do paralelepípedo envolvida na união é do tipo parcial enquanto que a face do cilindro é do tipo virtual.

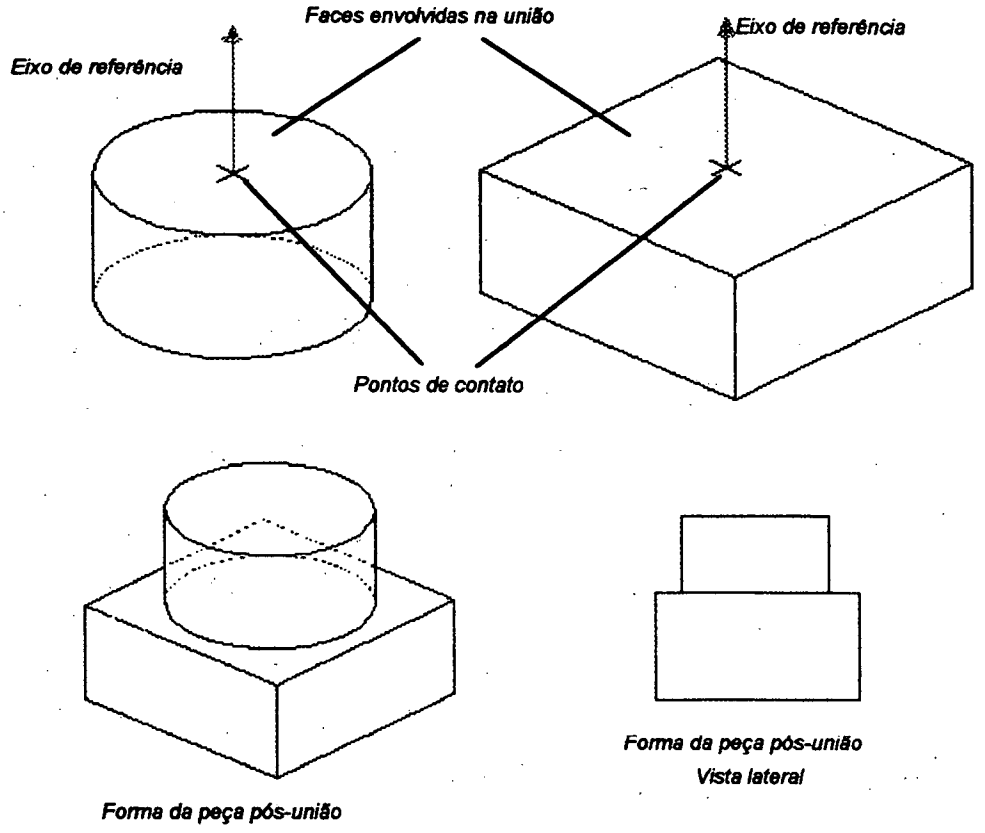


fig. 3.11 - Exemplo de peça ressaltando o elemento União

As faces envolvidas na união de features positivas e negativas são classificadas em *existentes* e *inexistentes*, o que reforça a idéia de remoção ou retirada da região sólida envolvida por uma feature negativa. O sólido referente a feature negativa deve ser removido de uma ou mais

features positivas, o que significa a face de união e todas as outras faces intersectadas pela feature negativa deixam de existir.

3.4.5.1.3. Sistema de Coordenadas das Faces

Como já citado anteriormente, existe um sistema de coordenadas associado a cada uma das faces de uma feature. Na construção das peças tal sistema é extremamente útil pois facilita a definição dos pontos de contato sobre as faces adjacentes, conforme descrito no item anterior. A figura 3.12. mostra exatamente a utilização desse sistema, quando da construção de uma peça.

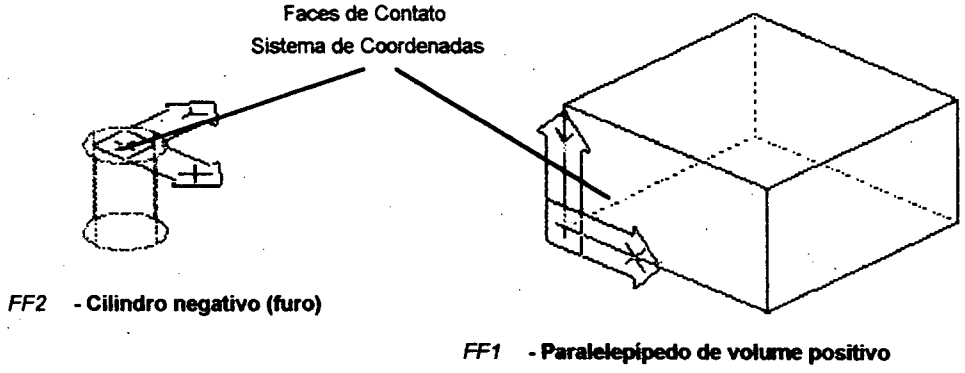


fig. 3.12 - Sistema de Coordenadas das Faces de cada feature

3.4.5.1.4. Peças Complexas

As peças complexas, como já citado anteriormente, serão tratadas como verdadeiras uniões de peças simples. Ou seja, uma peça complexa pode ser vista como

$Peça\_Complexa_i = Peça\_simples_1 \cup Peça\_simples_2 \ldots \cup Peça\_simples_n$

De modo análogo à composição das peças simples, a união das peças simples se dará sobre faces e pontos de contato, pertencentes a duas das features existentes nas peças simples envolvidas na formação da peça complexa. Portanto, pode-se atribuir um processo de ligação que caracterize a união e consoante o tipo desse processo, surgirão as peças *agregadas ou compostas*.

3.4.5.1.5. Features de Referência

O conceito de *feature de referência* (*ou feature raiz*) é aqui introduzido, e aplica-se àquelas features usadas como base para a inserção de novas features nas peças. Logo, feature de referência é toda feature que possui uma ou mais features adjacentes ou dependentes. Esse conceito é importante pois rege as operações de modificação e remoção das features pertencentes a uma peça.

Considere-se a peça da figura 3.13., composta por 4 features de volume positivo. Qual a forma final da peça se a feature FF_2 fosse removida?

Para ajudar a discussão do exemplo, supomos que o número de cada uma das features indica a ordem em que elas foram inseridas na peça. Portanto, segundo a regra de adjacência e adotando o conceito recém-introduzido de *feature de referência* , concluímos que a *FF2* foi introduzida tomando como referência a *FF1*, e que as features *FF3* e *FF4* tiveram como referência a *FF2*. Logo, a remoção da *FF2* deixaria um vazio no conjunto das relações de adjacência, uma vez que não existe qualquer relacionamento entre as features *FF1, FF3* e *FF4*. A peça resultante tem forma indefinida.

Nesse exemplo, tem-se então duas feautures de referência: *FF1* e *FF2*. Uma peça pode ter ***N*** features de referência.

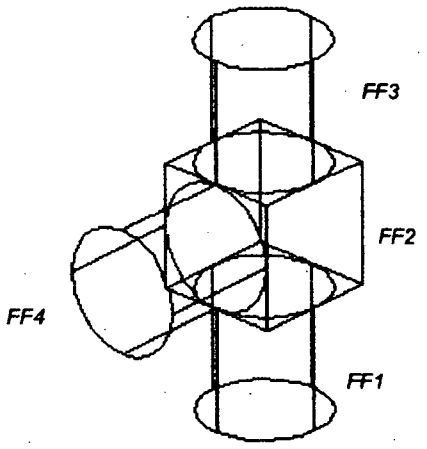


fig. 3.13 - Feature de Referência

Introduz-se aqui uma nova regra que rege a construção das peças dentro do módulo gráfico:

As features de referência não podem ser removidas ou mesmo alteradas. Tais operações somente são permitidas sobre *features folhas*, que são as features que não estão sendo usadas como referência.

No exemplo da figura 3.13., *FF3* e *FF4* são features folhas.

3.4.5.2. Meta-Peças

As *meta-peças* são entidades temporárias que agrupam diversas peças, mantendo entre elas uma relação espacial, definida em termos do posicionamento relativo existente entre as peças. As operações geométricas de translação, rotação e escalonamento podem ser aplicadas sobre a meta-peça, o que garante a aplicação sobre cada uma das peças isoladamente, porém mantendo o posicionamento relativo já existente.

No modelo mostrado na figura 3.14., assume-se que as peças *peça1, peça2, peça3 e peça4* formam uma meta-peça (um conjunto de pinos), e que devem ser encaixados nos 4 furos existentes na *peça5*. A meta-peça permite, portanto, a aplicação de uma unica operação de translação sobre o conjunto todo, para encaixar os pinos nos respectivos furos.

3.4.5.3. Macro-Features e Bibliotecas

As *Macro-features* são, em última análise, peças (simples ou complexas) que devem ser tratadas como uma primitiva ou feature canônica. Elas são formadas por features agrupadas segundo as relações que regem a construção das peças, com uma única restrição :

Uma macro-feature não pode ser incluída na peça que serviu de base para sua geração, pois poderia-se incorrer na formação de **loops** na construção dessa peça.

As *macro-features* são conceitualmente idênticas as features canônicas, pois são vistas como uma entidade única e indivisível, sobre as quais aplicam-se as mesmas regras aplicadas sobre as demais features.

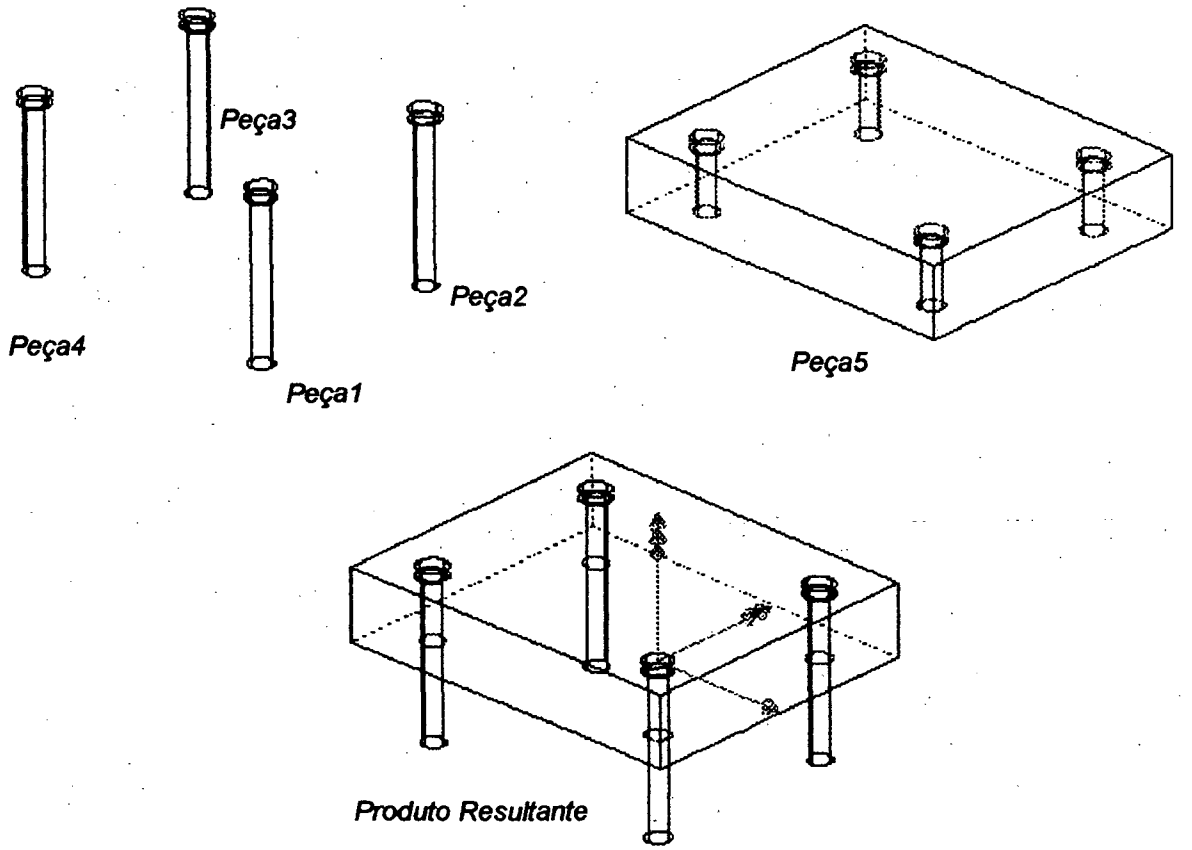


fig. 3.14 - Exemplo de translação de uma Meta-peça

As *macro-features* são agrupadas em bibliotecas, segundo critérios arbitrariamente definidos pelos usuários. Todas as restrições impostas pelo modelamento baseado em conjuntos específicos de features são contornadas com a utilização dessas bibliotecas, onde as features são construídas de maneira dinâmica e flexível.

Do mesmo modo em que as features canônicas necessitam de um padrão para instanciamento (por exemplo, um conjunto de parâmetros), as *macro-features* necessitam de uma lista

contendo a descrição das primitivas envolvidas e suas inter-relações. Essa lista é armazenada em um arquivo quando da criação de uma *macro-feature*, e usada como padrão para o seu instanciamento. O formato desse arquivo será descrito no capítulo que discute a implementação do modelo.

Como uma macro-feature pode ser instanciada em qualquer peça indistintamente e existe somente uma lista descritora de cada feature, configura-se a necessidade da criação de um fator de escala que esteja associado a cada uma dessas instâncias para permitir dimensionamentos diferentes adequados a cada peça.

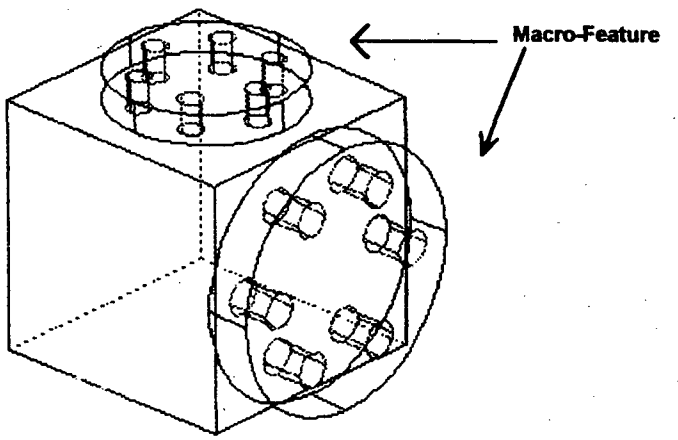


fig. 3.15 - Macro-Feature instanciada em escalas distintas

Capítulo 4

Implementação do Modelo UNINOVA/CESO

Após a apresentação do modelo conceituai que suporta o trabalho no capítulo anterior, passa-se à discussão dos aspectos relacionados com a implementação daquele modelo, soluções adotadas para atender os conceitos concebidos, restrições e técnicas de implementação referentes apenas ao *Módulo Gráfico*.

Em sua implementação o *Módulo Gráfico* recebeu o nome de *Feature Modeller*. Nesse capítulo ele será simplesmente referenciado como ***FM***.

4.1. Introdução

Como já citado anteriormente, esse trabalho faz parte de um projeto desenvolvido no âmbito do programa **BRITE-EURAM**, dentro da União Européia. O referido programa impôs algumas restrições quanto a plataforma de suporte à implementação e, dentre elas, duas foram determinantes: a utilização de computadores de pequeno porte e a utilização de sistemas computacionais de suporte quando necessários, do tipo *comercial*. Ou seja, disponíveis no mercado mundial. As duas restrições destinam-se à redução dos custos de trabalho e aplicação de seus resultados em empresas de pequeno e médio porte.

4.2. Plataforma e Ferramentas Utilizadas

Relembrando, a tarefa que contém este trabalho foi dividida em dois módulos, chamados *Abstrato* e *Gráfico*. Concebidos dentro do mesmo modelo, devem funcionar em simultâneo de maneira integrada. Em se tratando de hardware, a imposição do Brite restringiu o conjunto de opções à partida, e os equipamentos escolhidos foram computadores da linha *PC* modelo *486* para o FM, e *386* para o módulo abstrato. Contudo, as diferenças existentes entre as necessidades de cada módulo, oriundas de suas naturezas conduziram a adoção de sistemas de apoio totalmente distintos, como já esperado.

O módulo abstrato, por seus requisitos de raciocínios, regras e fatos, baseou-se em programas com recursos de *Inteligência Artificial*. Mais especificamente falando, o software foi o *Intelligence Compiler (IC)*, um sistema especialista baseado em regras, fatos e com gerenciamento de bases de dados orientadas por objeto. O módulo gráfico, por sua vez, utilizou como base o software *AutoCAD* na sua versão *R12*, acompanhada do modelador de sólidos *AME*, o qual utiliza uma abordagem de representação informação híbrida (*CSG* e *B-rep*) para as suas primitivas geométricas. Ambos os programas (*IC* e *ACAD*) trabalham sob o sistema operacional *DOS*.

A característica de sistema mono-usuário do *DOS*, aliada aos requisitos de operação dos módulos em tempo real, levou a busca de uma solução que permitisse integrar os dois módulos de modo transparente ao usuário. Isso foi conseguido através de um programa que possibilita a virtualização de áreas de trabalho inter-computadores, baseado em uma rede local. Tal programa chama-se *Neteye*.

A figura 4.1. retrata o cenário da plataforma de implementação da tarefa como um todo.

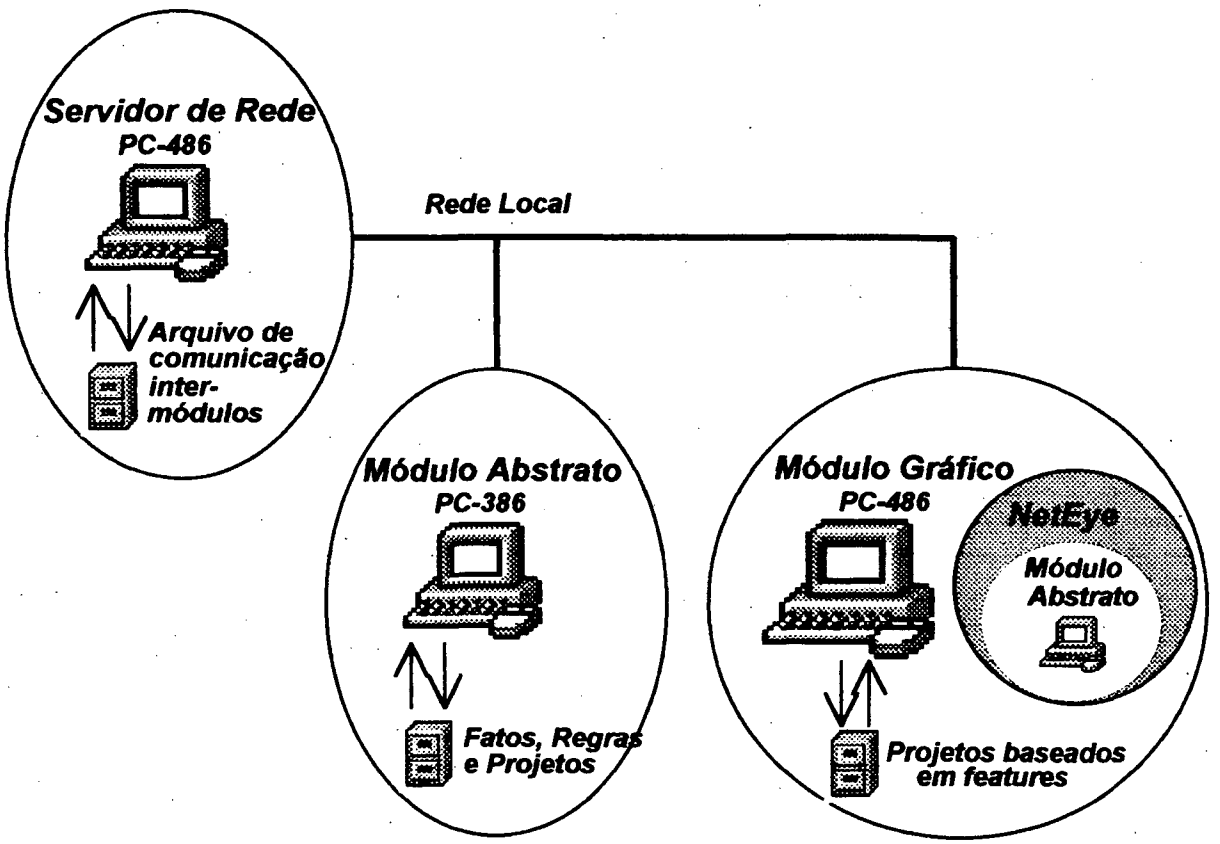


fig. 4.1 - Cenário da Implementação da Tarefa

4.3. Paradigmas de Orientação por Objetos

A implementação *Orientada por Objetos* tem sido bastante utilizada no desenvolvimento de pesquisas envolvendo features. Normalmente as features são agrupadas em conjuntos (*classes*), onde existem parâmetros (*atributos*) compartilhados e específicos a cada um deles. Alem do mais, a existência de hierarquização entre os tais conjuntos justifica ainda mais a orientação por objetos, pois ela conduz à utilização dos mecanismos de herança inter-classes.

O conhecimento sobre as features, genéricas ou específicas, é encapsulado sob forma de atributos e procedimentos em unidades chamadas *classes*. Consegue-se uma padronização fácil da manipulação dos procedimentos associados às features, além do que o encapsulamento dos atributos permite mudanças precisas e de impacto localizado sobre uma certa classe. Mecanismos de herança permitem o compartilhamento de procedimentos e atributos comuns a classes hierarquicametne dependentes. Ou ainda as propriedades de herança permitem explorar similaridades entre membros de uma família de features [Shah91b].

Dentro da plataforma de desenvolvimento adotada para o FM, a escolha de uma linguagem do tipo *Orientada por Objetos* resumia-se àquelas que pudessem ser utilizadas nas aplicações

As features *simples, modificadoras e compostas* são agrupadas em uma biblioteca chamada *Canônica*. Como features paramétricas, todas elas possuem conjuntos de parâmetros através dos quais são instanciadas. As *macro-features* são agrupadas em bibliotecas criadas e identificadas segundo especificação dos usuários, chamadas *bibliotecas abertas*.

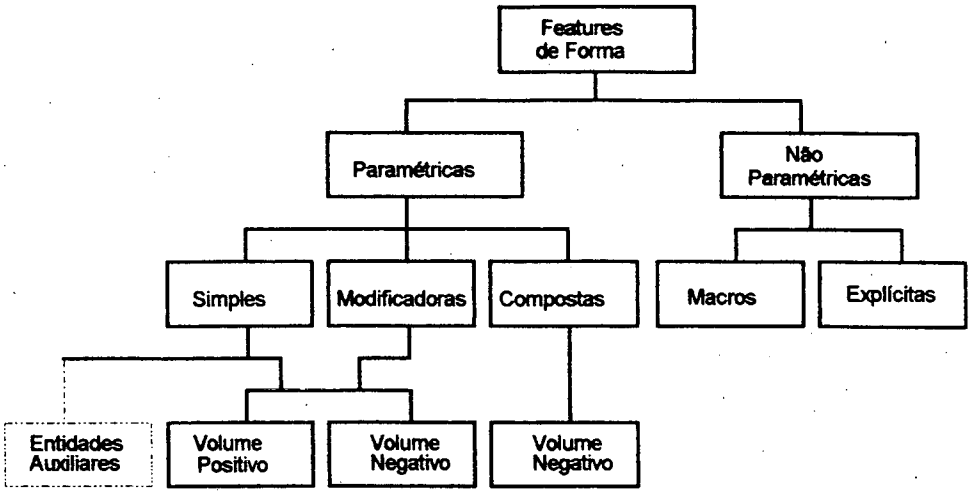


fig. 4.3 - Classificação das features

Das features de mais alto nível, as únicas contempladas para validar o modelo foram os furos definidos através dos padrões matriciais e circulares. Entretanto todos os parâmetros devem ser especificados nominalmente, não tendo sido implementada a derivação de parâmetros das features *pais* para os furos passantes.

4.4.1.1. Aspectos de Implementação particulares às Features

Como já dito nas secções anteriores, as peças e as features livres são compostas por features (canônicas e macros) que são agrupadas obedecendo a um conjunto de regras e relações. Alguns aspectos particulares à implementação do tratamento das features serão discutidos nessa seção, para facilitar o entendimento das abordagens discutidas a *posteriori*.

O primeiro relaciona-se com o instanciamento das features. O modelador de sólidos adotado trabalha com um sistema de coordenadas *absoluto*, porém permite que sistemas relativos sejam definidos uma vez que o ponto de origem do sistema e a orientação dos eixos são definíveis. As primitivas geométricas, que traduzirão a forma das features baseiam-se no ponto ***(0,0,0)*** absoluto para sua origem e operações geométricas. As features devem ser inseridas nas peças segundo orientação do projetista, o que significa que existem recursos para que as operações necessárias sejam efetuadas.

Portanto, aproveitando o conceito de instanciamento do modelador sólido de suporte, as features serão sempre instanciadas no ponto ***(0,0,0)*** absoluto e depois posicionadas, segundo especificação do usuário, na entidade a que pertencem. Para a 1ª feature pertencente a uma entidade, no caso às peças, esse conceito não se aplica pois tal feature é sempre posicionada na origem do sistema de coordenadas corrente, o qual é adotado como o sistema de coordenadas da peça.

Para permitir o posicionamento das features nas entidades, o FM serve-se da informação referente à cada face das features e cria elementos descritores da união inter-feature. Uma das características mais particulares é a definição dinâmica de um sistema de coordenadas bi-dimensional sobre cada uma das faces envolvidas na adjacência inter-feature, a fim de facilitar a identificação dos eixos diretores da união. Além do mais, cada face existente em uma feature é univocamente identificável através de um número.

Para vizualização das features adotou-se a *wireframe*, embora não seja a melhor em termos visuais. As operações booleanas que deviam automaticamente serem executadas quando da inclusão de novas features em uma peça não o são. A justificativa para as duas abordagens anteriores vem da necessidade de identificação de cada feature individualmente, dentro do grupo que constitue, por exemplo, uma peça. O *wireframe* permite a identificação unívoca de cada uma das features e a não execução das operações booleanas também, pois por restrição do ACAD/AME os sólidos resultantes de operações booleanas recebem novos identificadores internos, o que dificultaria a manipulação das features individualmente.

No entanto, são disponibilizadas funções que permitem contornar o problema de vizualização das entidades, como será detalhado na discussão referente às peças.

Existe uma associação direta entre uma primitiva instanciada pelo FM e um sólido fornecido pelo modelador sólido adotado, que é um *identificador numérico* atribuído a esse sólido. Esse identificador é o código através do qual o *ACAD* recupera informações sobre o sólido correspondente ou executa operações boolenas e geométricas sobre ele. Portanto, evita-se a utilização de comandos internos do *ACAD* que efetivamente altere os identificadores das primitivas, o que traria problemas à supervisão do FM.

4.4.1.2. Atributos

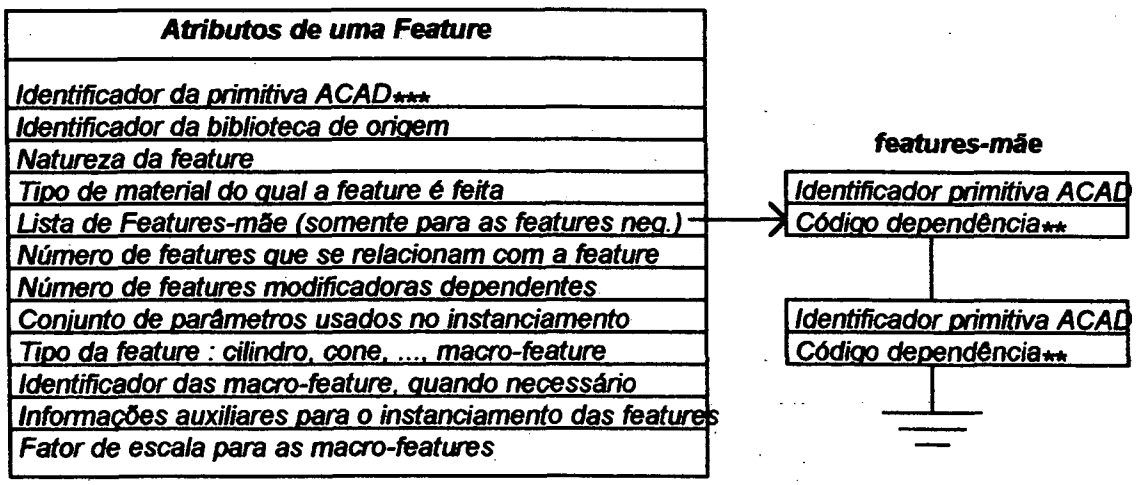
Para contemplar todo o conhecimento existente a respeito da classe *Features*, os seguintes atributos foram definidos :

- Identificador do sólido correspondente à feature, fornecido pelo modelador sólido. Esse identificador serve de ligação entre as features tal como concebidas no FM e sua representação geométrica no ACAD. Todas as informações de mais baixo nível tratadas pelo ACAD são acessadas através desse identificador.
- Identificador da biblioteca da qual a feature é originária.
- Natureza da feature : *positiva* e *negativa*, onde positiva significa necessariamente a *presença* de volume e negativa a *ausência*.
- Tipo de Material : herdado da entidade a qual a feature pertence (peça ou feature livre).
- Lista que contém as features *Mãe* de cada feature negativa.
- Quantidade de *Filhas* que a feature possui, ou seja, o número de features que por alguma relação se inter-relacionam com a referida feature.
- Quantidade de features modificadoras existentes em uma feature.
- Identificador do conjunto de parâmetros utilizado no instanciamento da feature.
- Tipo da feature : Paralelepípedo, Cilindro, Cone, Cunha, Macro, Padrão Matricial, Padrão Circular, Chanfro e Concordância.

- Identificador das macro-features.
- Informações auxiliares para as features modificadoras e compostas.
- Fator de escala associado somente as macro-features.

Como já dito anteriormente, o modelador de sólidos AME fornecido pelo ACAD baseia-se na representação híbrida (CSG e B-rep) das entidades sólidas. O FM utilizou as operações booleanas para validar o conceito de volume positivo e negativo, associando-os respectivamente as operações *união* e *subtração*. Ou seja, na construção de uma peça as features positivas são *unidas* e as negativas *subtraídas* do conjunto da peça, de tal maneira que a forma final da peça esteja sempre presente.

Como existe uma restrição oriunda do modelador sólido em relação a identificação unívoca das faces existentes em sólidos compostos (unidos e/ou subtraídos), e como tal característica é essencial para o funcionamento do modelo concebido para o FM, as operações de união e subtração não serão consideradas quando do instanciamento de cada feature, mas sim passíveis de serem efetuadas a qualquer momento sob requisição do usuário.



** - descrito mais apropriadamente na relação de dependência
*** - possibilita o acesso a todas as informações geométricas de baixo nível manipuladas pelo ACAD, referentes à uma primitiva geométrica

fig. 4. 4 - Atributos de uma feature

4.4.1.3. Features Canônicas

O conjunto das features canônicas, totalmente dependente das primitivas sólidas oferecidas pelo ACAD, é composto de: paralelepípedos, cilindros, troncos de cones, cunhas, chanfros, concordâncias, padrões de furos matriciais e circulares, conforme mostrado na figura 4.5.

Cada uma das features canônicas é instanciada baseando-se em parâmetros geométricos que as caracterizam univocamente. Algumas dessas features também necessitam do atributo de caracterização de suas *naturezas*, como *positiva* ou *negativa*. O FM tem conjunto de parâmetros

para cada feature, os quais são apresentados a seguir em suas formas positivas e negativas (no 1° sólido, por exemplo, o cilindro externo é positivo enquanto que o interno é negativo).

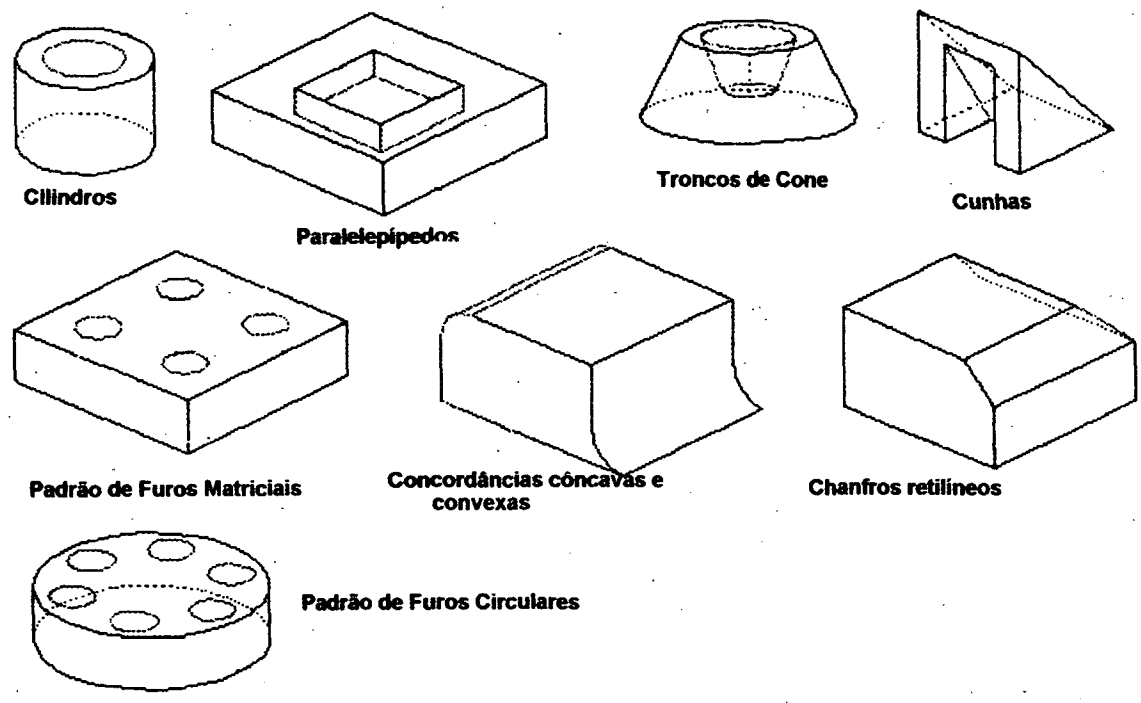


fig. 4.5 . features canônicas

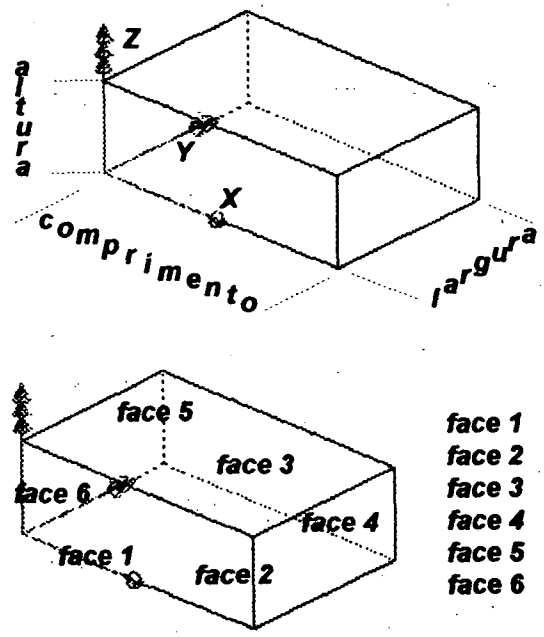


fig. 4.6 - Parâmetros descritores dos Paralelepípedos

4.4.1.3.1. Paralelepípedos

O FM fornece dois conjuntos de parâmetros para o instanciamento dos paralelepípedos. A figura 4.6. é um exemplo do instanciamento através de um desses conjuntos. Na figura mostra-se também o sistema de coordenadas e as faces que identificáveis em um paralelepípedo.

Para o conjunto ***<Comprimento>*** um cubo é instanciado segundo a dimensão especificada. Para o conjunto ***<Comprimento, Largura, Altura>*** um paralelepípedo é instanciado tendo o *comprimento* especificado no sentido positivo do eixo **X**, sua *largura* especificada no sentido positivo do eixo **Y**, e sua *altura* especificada no sentido positivo do eixo **Z**.

Os paralelepípedos de natureza negativa são vistos como *cavidades*, *rasgos*, embora todas as dimensões devam ser explicitamente especificadas, não havendo sido implementado o tratamento da derivação ou herança de parâmetros.

4.4.1.3.2. Cilindros

O FM fornece um único conjunto de parâmetros para o instanciamento dos cilíndros, onde requisita-se **<Raio, Altura>**, respectivamente o *raio* e a *altura ou profundidade* do cilindro especificidado no sentido positivo do eixo **Z**. Com a atribuição da natureza da feature, o cilindro de volume *negativo* é tratado como um furo, onde as dimensões são explicitamente descritas.

Não foi implementado também a categorização global existente para os furos, que são os *furos passantes* e *cegos*. O furo passante, como já discutido anteriormente, herdaria sua profundidade da(s) feature(s) onde ele se localizasse.

A figura 4.7 identifica cada um dos parâmetros descritores de um cilindro.

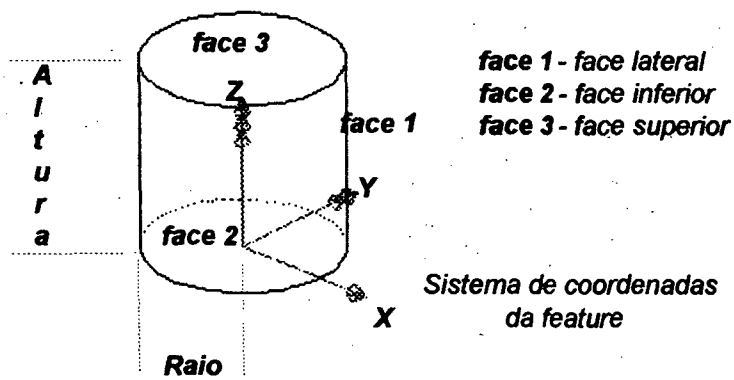


fig. 4.7 - Parâmetros descritores dos Cilindros

4.4.1.3.3. Troncos de Cone

Para o instanciamento dos troncos de cone, o FM fornece também um único conjunto de parâmetros, onde requisita-se ***<Raio da Base, Raio Topo, Altura>***, respectivamente *raio da base, raio de topo e altura* do cone no sentido positivo do eixo **Z**. Os cones de natureza negativa

são tratados como *furos cônicos*. A figura 4.8 identifica cada um dos parâmetros descritores de um tronco de cone.

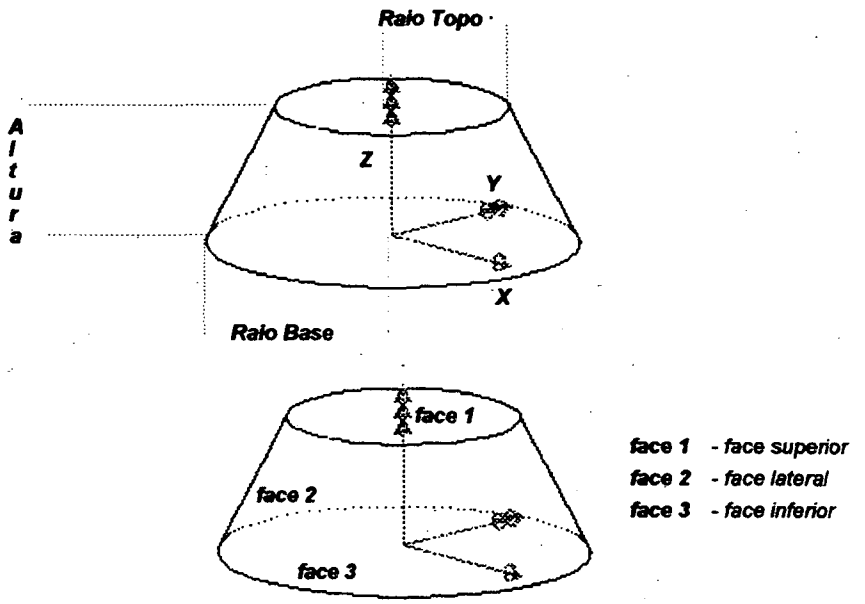


fig. 4.8 - Parâmetros descritores dos Cones

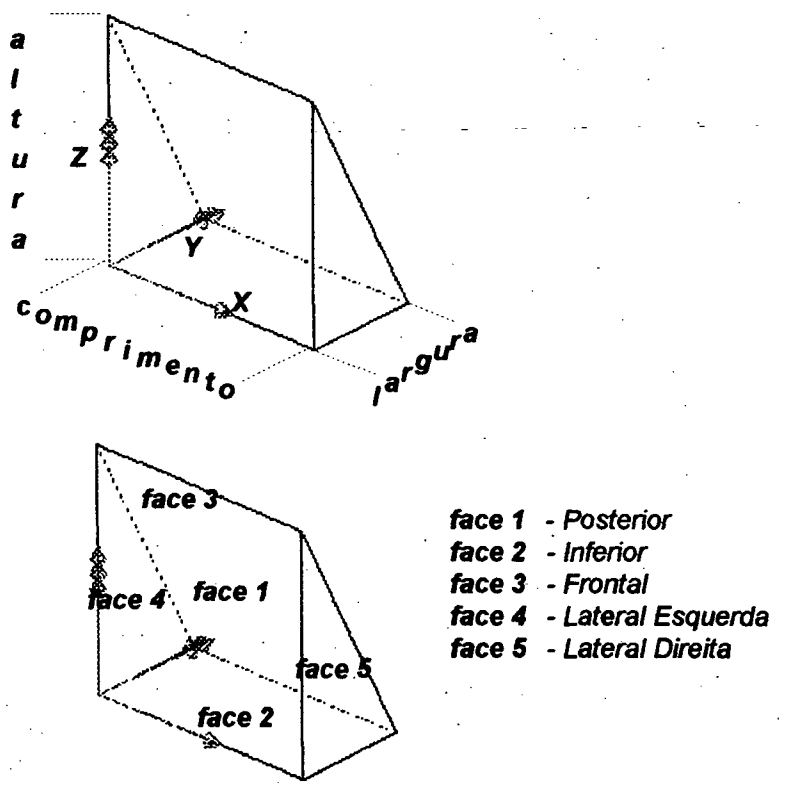


fig. 4.9 - Parâmetros descritores das cunhas

4.4.1.3.4. Cunha

Para o instanciamento de cunhas, o FM oferece apenas um conjunto de parâmetros onde requisita-se ***<Comprimento, Largura, Altura>***, respectivamente seu *comprimento* (sentido positivo do eixo **X**), sua *largura* (sentido positivo do eixo **Y**) e sua *altura* (sentido positivo do eixo). As cunhas de natureza negativa também são considerados, embora não exista uma taxonomia particular para definí-los.

A figura 4.9 identifica cada um dos parâmetros descritores de uma cunha.

4.4.1.3.5. Chanfros

O FM trata dois tipos de chanfros : os *retilíneos*, aplicáveis sobre paralelepípedos e cunhas, e os *circulares*, aplicáveis sobre cilindros e cones. A seleção do tipo de chanfro a ser instanciado é feita automaticamente pelo FM baseado na seleção do tipo de aresta a ser chanfrada. Os chanfros possuem apenas um conjunto de parâmetros nominais, porém necessitam da seleção de entidades gráficas auxilares, que são: *aresta a ser chanfrada, face de referência, arestas base e adjacente*. Essas entidades serão descritas para cada tipo.

Para os chanfros retilíneos, as seguintes entidades devem ser selecionadas :

- aresta *retilínea* a ser chanfrada.
- *face de referência*: uma das faces perpendiculares e adjacentes à aresta a ser chanfrada.
- *aresta base*: aresta que pertence à face de referência e a uma outra face perpendicular àquela, chamada de *face base do chanfro* (base face).
- *aresta adjacente*: aresta que pertence à face de referência e a uma outra face perpendicular àquela, chamada de *face adjacente ao chanfro.*

Para os chanfros circulares, apenas as seguintes entidades devem ser selecionadas :

- aresta *circular* a ser chanfrada.
- *face de referência*: face planar adjacente à aresta a ser chanfrada.

Os parâmetros existentes no único conjunto aplicável aos dois tipos de chanfro são ***<Distância Face Adjacente, Distância Face Base>***, respectivamente as distâncias entre a aresta chanfrada e as faces que efetivamente serão chanfradas.

Os chanfros, que juntamente com os concordâncias são classificados como *features modificadoras*, possuem uma particularidade existencial. Conforme concepção do modelo, as features modificadoras deveriam ser incorporadas diretamente sobre a geometria da feature-destino, porém por restrições do modelador sólido escolhido os chanfros são tratados como primitivas volumétricas negativas. A forma gerada pela inclusão de chanfros só é visualizável na feature-destino após a execução de uma operação booleana de subtração, embora esta operação não seja parte do modelo.

A fim de permitir um melhor entendimento dos parâmetros descritores de um chanfro, um exemplo é mostrado na figura 4.10.

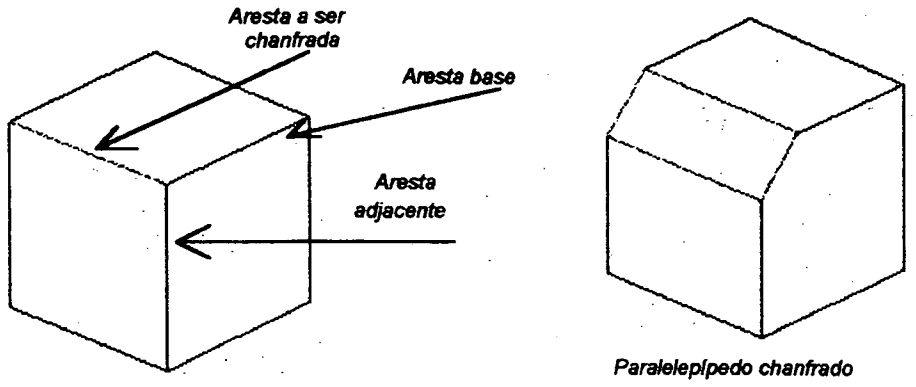


fig. 4.10 - Instanciamento Chanfro Retilíneo

4.4.1.3.6. Concordâncias

Pertencendo ao conjunto das *features modificadoras* juntamente com o chanfro, as concordâncias possuem mais particularidades de implementação. Elas são classificados em *côncava* e *convexa*, o que representa exatamente o atributo *natureza* das features simples. Ou seja, a concordância côncavo significa presença de volume enquanto a convexa significa remoção.

Como para os chanfros, existem dois tipos de concordâncias : *retilíneos* - aplicáveis nos paralelepípedos e cunhas - e *circulares* - aplicáveis sobre cilindros e cones. O tipo de concordância é automaticamente decidido pelo FM baseando-se no tipo de aresta a ser arredondada. Além do mais, as concordâncias circulares são tratadas como *convexas*.

As concordâncias possuem apenas um conjunto de parâmetros nominais, porém necessitam da seleção de entidades gráficas auxilares, que são : *aresta a ser arredondada* e *face de referência*. Essas entidades serão descritas para cada tipo.

Para as concordâncias retilíneas :

- aresta *retilínea* a ser arredondada.
- *face de referência*: uma das faces perpendiculares e adjacentes à aresta a ser arredondada.

Para as concordâncias circulares :

- aresta *circular* objeto da concordância.
- *face de referência*: a face planar que contem a aresta objeto da concordância.

As concordâncias, que também são *features modificadoras*, possuem uma particularidade quanto à suas existências. Conforme concepção do modelo, as features modificadoras deveriam ser incorporadas diretamente sobre a geometria da feature-destino, porém por restrições do modelador sólido escolhido, as concordâncias são tratadas como primitivas volumétricas negativas. A forma gerada pela inclusão de concordância só é visualizável na feature-destino após a execução de uma operação booleana de subtração, embora esta operação não seja parte do modelo.

4.4.1.3.7. Padrões de Furos

O FM trata dois tipos de padrões de furos : *circulares* e *matriciais*. Como já citado anteriormente os furos são cilindros de volume negativo, obedecendo à especificação de parâmetros nominais precisos. Por essa característica de dependência existencial conferida às features negativas os padrões não existem de *per si*, o que os obriga a estarem contidos em alguma feature positiva. O FM fornece apenas um conjunto de parâmetros para cada um dos padrões, os quais estão descritos a seguir.

Para os padrões circulares, requisita-se :

- *Raio* de cada furo.
- *Profundidade* de cada furo.
- *Quantidade de furos* que compõe o padrão.
- *Ângulo*, em graus, existente entre um furo e outro.
- *Raio base* de uma circunferência sobre a qual os furos serão dispostos.

Para os padrões matriciais, requisita-se :

- *Raio* de cada furo.
- *Profundidade* de cada furo.
- *Quantidade de furos na direção X* ou numero de linhas da matriz.
- *Quantidade de furos na direção Y* ou número de colunas da matriz.
- *Distância* entre dois furos na direção X.
- *Distância* entre dois furos na direção Y.

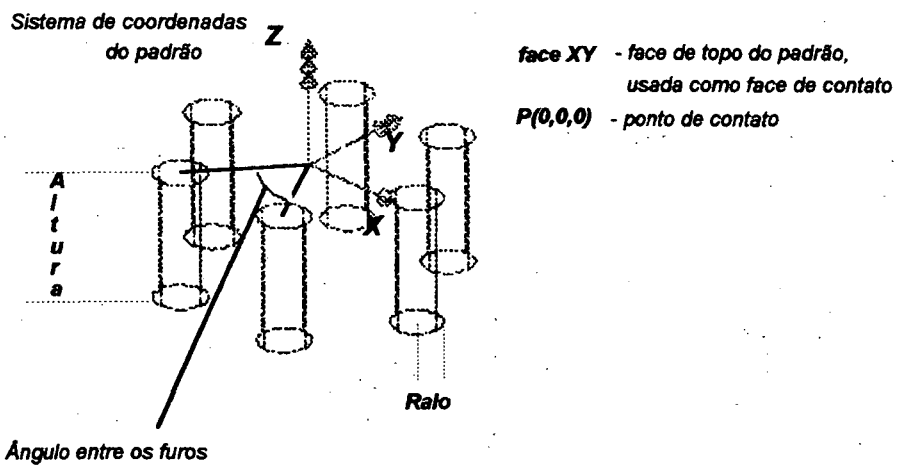


fig. 4.11 - Padrão Circular de furos

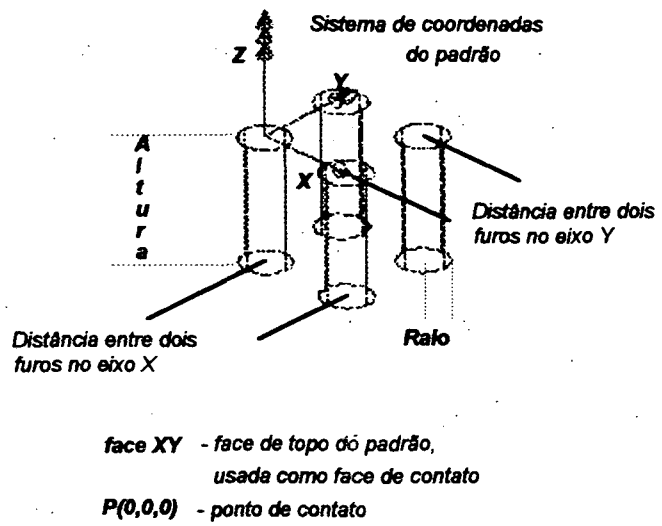


fig. 4.12 - Padrão Matricial de Furos

OBS. *Direção X* e *Y* referem-se às orientações respectivas dos eixos **X** e **Y** da face topo do padrão.

4.4.5. Macro-Features

Pode-se dizer que o conjunto de features canônicas por ser fortemente dependente do modelador sólido de suporte, reduz-se a um número muito pequeno de primitivas de criação. A idéia naturalmente emergente de aumentar esse conjunto para atender minimamente os requisitos de criatividade do usuário foi contemplado com as macro-features. Conforme discutido no capítulo 3, seção 3.4.5.3., as macro-features são peças que devem ser tratadas como uma feature única, independente do número de features que as formam.

Uma macro-feature é conceitualmente idêntica a uma feature canônica, exceção feita à sua natureza que é sempre positiva pois as peças, que **sempre** produzem volume, são a base da criação das macro-features. Assim sendo, uma macro-feature é também uma primitiva de criação de *features livres* e *peças*.

Algumas restrições para a criação das macro-features fazem-se necessárias e são as seguintes :

- A *peça-base* (usada como modelo para a macro-feature) deve estar armazenada em arquivo, para tornar factível sua recuperação e instanciamento. Sem esse pré-requisito, o FM não permite que uma peça sirva de base à criação de uma macro-feature.

- Para manter a coerência com as features canônicas que são pré-instanciadas no ponto ***(0,0,0)*** absoluto, a *peça-base* deve estar posicionada e orientada segundo o sistema de coordenadas absoluto, com origem no ponto ***(0,0,0)*** e orientação segundo os eixos ***X***, ***Y*** e ***Z***. Por *peça-base* entenda-se aquela a partir da qual uma macro-feature é gerada.

- Uma peça não pode ter como feature componente uma macro-feature que corresponda a própria peça, pois isto geraria uma situação de *loop*. Essa restrição é para eliminar a recursão infinita de *peça que inclue macro-feature que inclue peça.*

Quando uma macro-feature é instanciada em uma peça, a estrutura de dados que a representa não é repetida dentro da peça mas simplementes referenciada, acompanhada do respectivo fator de escala para o seu instanciamento nesta peça. Em outras palavras, uma macro-feature pode ser incluída em diversas peças, com fatores de escala distintos e posicionada segundo orientações particulares sem que isso implique na existência de um arquivo de dados para cada macro-feature.

As macro-features são armazenadas em bibliotecas organizadas livremente segundo critérios do usuário. As funções disponibilizadas para manipulação das bibliotecas são as de *inclusão* e *remoção* das macro-features; *criação, abertura* e *fechamento* das bibliotecas.

4.4.2. Relações

O FM baseia-se em três relações básicas para reger as interações entre as features, que são as relações de *posicionamento*, *adjacência* e *dependência*. Uma vez que as features serão dispostas lado a lado ou interpenetrando-se, durante a concepção das peças, o tratamento dessas relações é fundamental para uma boa definição das mesmas. Os detalhes de implementação das relações são discutidos a seguir.

4.4.2.1. Relação de Posicionamento

A relação de posicionamento fornece uma matriz de transformação homogênea que contém a localização e orientação espacial de cada feature instanciada, em relação a um sistema de coordenadas absoluto. Isso significa que durante a inclusão de uma feature em uma entidade (peça ou feature livre), embora o posicionamento seja feito relativamente a outras features, a informação é mantida em relação ao sistema absoluto.

Como os sólidos fornecidos pelo ACAD já possuem essa matriz definida implicitamente, tomou-se desnecessário replicá-la para as necessidades do FM. Essa informação é requisitada e fornecida pelo ACAD sempre que necessário.

O posicionamento das features é feito tendo como referência faces pertencentes as duas features, sendo que o ângulo existente entre as duas faces é sempre 0^{c}. Ou seja, duas faces são sobrepostas.

4.4.2.2. Relação de Adjacência

A relação de adjacência fornece as faces e pontos envolvidos na justaposição de duas features. Todas as features são instanciadas em uma entidade tendo como referência duas faces e dois eixos definidores de sua inclusão. Tal regra não se aplica à 1ª feature intanciada em uma entidade.

A fim de facilitar a implementação do conceito, somente são aceitas faces *planas* como definidoras de adjacência. Ou seja, as faces laterais dos cilindros e troncos de cones não são válidas.

Os atributos descritores da relação são :

- Identificador da *feature de referência*, que é a feature-destino na adjacência.

- Identificador da *face de referência* pertencente a feature de referência, a qual é usada como base na adjacência.

- Coordenadas do *ponto de referência*, localizado na face de referência e sobre o qual incide um eixo perpendicular que serve de guia para a adjacência. Em outras palavras, o ponto-de-contato na face de referência.

- Identificador da *feature-fonte*, que é a feature que será instanciada justaposta a feature-destino.

- Identificador da *face-fonte* pertencente a feature-fonte, a qual é usada como base na adjacência.

- Coordenadas do *ponto-fonte*, localizado na face-fonte e sobre o qual incide um eixo perpendicular que serve de referência para a adjacência. Em outras palavras, o ponto-de-contato fonte.

- Percentuais da localização do ponto de contato em relação à dimensão da face. Permite que nas modificações das dimensões nominais de uma feature, o ponto-de-contato da adjacência dessa feature-fonte seja relativamente mantido, a fim de evitar inconsistências posicionais.

- Natureza da feature-fonte : positiva ou negativa.

- Processo de fabricação que pode ser associado à ligação, onde é possível especificar quaisquer dos processos tratáveis pelo módulo abstrato.

As informações referentes às relações de adjacência são tratadas a nível das entidades como um todo e não a nível de cada uma das features. Isso significa que a entidade (peça ou feature-livre) possui uma lista de objetos que retratam as adjacências existentes entre suas features componentes.

O tratamento da adjacência, apesar da não geração automática da forma final da entidade criada, o que se traduz na não criação das novas faces oriundas das interferências entre as features, implementa-se baseando-se nos identificadores das faces particulares a cada uma das features envolvidas.

4.4.2.3. Relação de Dependência

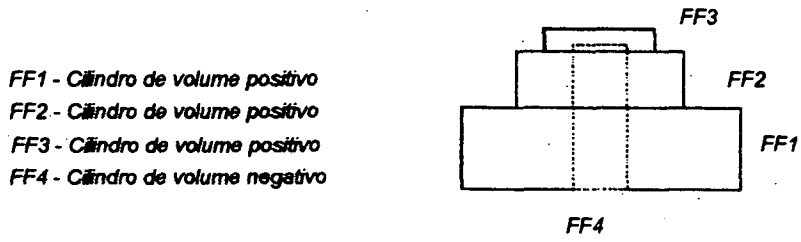
Aplica-se somente as relações envolvendos features positivas e negativas. Como as features negativas significam remoção de volume, elas dependem das features positivas para existir. Cada feature positiva que possue uma ou mais features negativas dependentes é chamada de *feature-mãe*, e cada feature negativa é chamada de *feature-filha*.

Os atributos descritores da relação são :

- Identificador de uma *feature-mãe*

- Tipo da dependência, onde o domínio de valores possíveis é uma combinação de três dígitos que permitem identificar : a feature-mãe pela qual a filha foi instanciada (*origem*), as features-mãe pelas quais a filha trespassa (*passante*) e a feature-mãe na qual a feature-filha termina (*final*). Cada um dos dígitos assume os valores ***0*** ou ***1***.

A informação referente à dependência é tratada pelas próprias features, o que significa que cada uma das features negativas (*features-filhas*) possui uma lista de objetos que identificam as features positivas que fazem parte do seu conjunto de *features-mãe*.



RDij* - relação de dependência entre *feature i* e *feature j

	Origem	Passante	Fim
RD14	***1***	***1***	***0***
RD24	***0***	***1***	***0***
RD34	***0***	***0***	***1***

fig. 4.13 - Conjunto de Features-mãe em uma relação de dependência

4.4.3. Features Livres

As features livres são entidades criadas no modelo conceitual para permitir a existência isolada de features canônicas e macro-features, sem que estas obrigatoriamente estejam contidas em peças. Isto significa que as features livres são compostas de apenas uma feature, seja ela canônica ou macro.

Entretanto, a implementação das features livres está ligeiramente alterada em relação à definição conceitual, pois as features livres poderiam ser compostas por até duas features, sendo estas uma feature modificadora inserida numa feature simples. Como a modelador de suporte forçou a criação e tratamento de chanfros e concordâncias (features modificadoras) como primitivas de volume também, a aceitação dessa condição implicaria em ter uma peça composta por duas features tratada como uma feature livre. Por esse motivo, decidiu-se alterar ligeiramente o conceito das features livres.

Caso ocorra essa necessidade de se ter uma feature livre composta por uma feature simples e uma modificadora, o FM permite que uma peça com a referida composição seja transformada em uma meta-peça e a partir daí, tratada como uma feature livre.

4.4.3.1. Aspectos de Implementação particulares às Features Livres

Para melhor clareza na descrição das *features livres*, faz-se necessário definir alguns conceitos básicos referentes a estas entidades.

- **Representação**: tanto features livres quanto peças serão representadas em **wireframe**, visto que as features também o são. Isso significa dizer que somente as arestas das primitivas geométricas são traçadas.

- Cada feature livre possui um sistema de coordenadas que é composto por um ponto de origem e três eixos orientados *X, Y* e *Z*. Esse sistema fornece a base de orientação de uma feature livre. As operações geométricas são efetuadas sobre esse sistema de coordenadas.

- Cada instância de uma feature livre tem um arquivo de dados particular. Isso gera uma replicação de dados, porém simplifica manipulação. Ou seja, se o usuário desejar ter duas features livres iguais no mesmo modelo, com orientação e/ou posição diferentes, ele deve fazer uma cópia da 1ª feature atribuindo-lhe um novo identificador e posição, o que significa realmente criar uma nova feature livre.

- Cada feature livre será composta de *uma e somente uma* feature, canônica ou macro. Em função da complexidade que pode estar presente em uma macro-feature, elas não são alteráveis como as canônicas.

- Dentre todas as features livres presentes no modelo, somente uma delas está *ativa* num dado instante de tempo. Isso significa que apenas a feature livre ativa pode ser manipulada através das funções disponibilizadas para tal.

- O modelador sólido de suporte trabalha com um sistema de coordenadas variável, chamado *Sistema de Coordenadas do Usuário* (***UCS***) o qual foi utilizado pelo *FM* para servir de origem às features livres. Ou seja, no instanciamento da 1ª feature pertencente a uma feature livre, o FM busca diretamentente do ACAD esse sistema e o define como *sistema de coordenadas* da feature livre.

4.4.3.2. Atributos

Para contemplar todo o conhecimento existente a respeito da classe *Features Livres*, os seguintes atributos foram definidos:

- *Identificador* : mnemônico pela qual a entidade é referenciada dentro do modelo, atribuído pelo usuário.

- *Procedência* da feature, ou seja, se ela é *canônica* ou oriunda de uma biblioteca de *macro-features.*

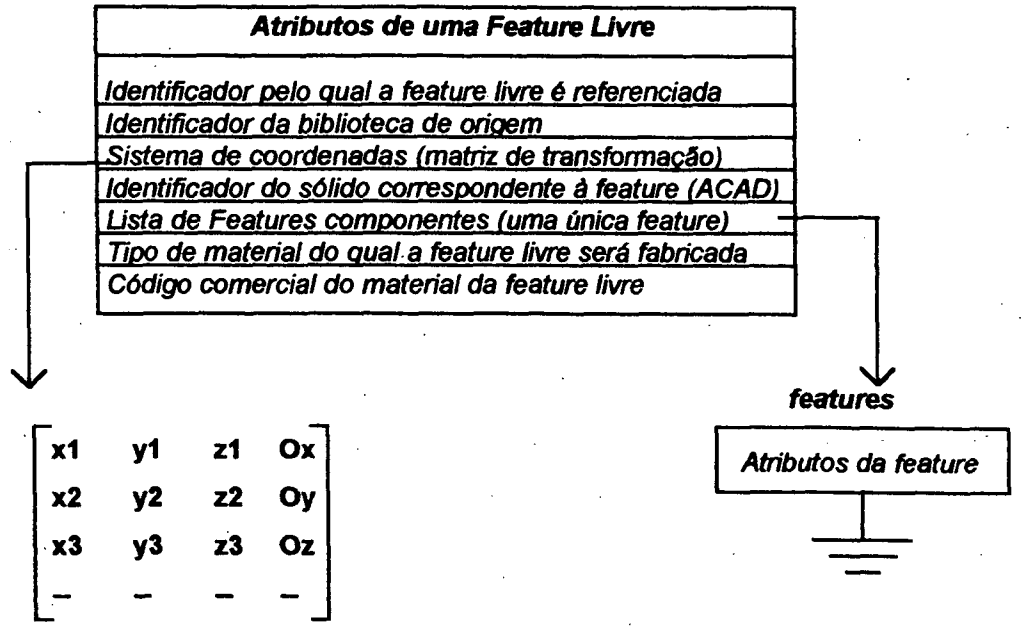
- *Sistema de coordenadas*, representado por uma matriz de transformação homogênea contendo origem e eixos de orientação *X, Y* e *Z*.

- *Identificador do sólido* correspondente à feature livre, criado pelo modelador sólido de suporte.

- *Lista de features* que compõe a feature livre. Tal lista possui sempre somente um elemento.

- *Tipo do material* do qual a feature livre será fabricada

- *Código da forma* comercial disponível do material (bloco, barra, vergalhão, etc).



Ox, Oy, Oz - *coordenadas do ponto de origem do sistema*
x1, x2, x3 - *cossenos diretores do eixo X*
y1, y2, y3 - *cossenos diretores do eixo Y*
z1, z2, z3 - *cossenos diretores do eixo Z*
– - *valores não-usados*

fig. 4.14 - Feature livre

4.4.3.3. Instanciamento

O instanciamento de uma feature livre exige os atributos de *identificação*, do *tipo de material* e *código de forma comercial*. Não é necessário a inclusão imediata de features (*de-forma*) à feature livre. Essa construção é distinta consoante a procedência da feature a ser incluída na feature livre.

A inclusão de uma feature canônica baseia-se na escolha dentre as existentes no conjunto canônico oferecido, e no posicionamento regido por uma matriz de transformação homogênea, que representará o *sistema de coordenadas* da feature livre. É claro que somente são aceitas features canônicas de natureza *positiva*.

A inclusão de uma macro-feature exige que uma biblioteca de macros-features tenha sido construída previamente, o que será discutido com mais rigor em itens subsequentes. Em termos funcionais, porém, desenrola-se exatamente o mesmo processo definido para as features canônicas.

Uma restrição definida sobre o instanciamento visa não permitir a transformação de *features livres* em *peças*, inibindo a inclusão de novas features após a 1ª ter sido incluída.

4.4.3.4. Operações Geométricas e Auxiliares

Sobre uma feature livre podem ser aplicadas funções geométricas (*rotacionar, transladar e escalonar*), e funções auxiliares (*exclusão, cópia, modificação, ativação e consulta*).

As operações geométricas aplicam-se diretamente sobre o sistema de coordenadas da feature livre. Na *translação* o ponto de origem do sistema é movido. Na *rotação* a feature gira sobre o ponto de origem, ângulos especificados em relação a cada um dos eixos de orientação. O escalonamento aplica à feature livre um fator especificado pelo usuário, sem no entanto alterar qualquer informação do sistema de coordenadas daquela feature livre.

As operações auxiliares visam ajudar a manipulação das entidades. A *remoção* traduz-se pela exclusão sumária da feature livre do modelo, o que significa dizer que tal feature simplesmente deixa de pertencer ao modelo atual. As informações referentes à tal feature, caso já existam armazenadas em arquivos, continuam intactas.

A *modificação* aplica-se somente as features livres formadas por features canônicas, onde as alterações processam-se a nível de parâmetros geométricos e tecnológicos (tipo de material e código de forma comercial). As features livres formadas por macro-features não são passíveis de alterações, uma vez que tal operação poderia envolver mudanças simultâneas nas diversas features formadoras das macro-features. A necessidade de alterações dessa natureza são contornadas por alterações efetuadas nas próprias macro-features.

A *cópia* visa permitir a criação, dentro de um mesmo modelo, de uma nova instância de uma feature livre já existente. Ou seja, é um recurso para permitir a existência de múltiplas instâncias da mesma feature livre cujas diferenças básicas são suas posições e orientações espaciais. Todos as demais informações são rigorosamente as mesmas, desde que não sejam explicitamente alteradas.

A *ativação* consiste em definir qual, dentre as features livres existentes no modelo corrente, é aquela passível de ser trabalhada pelas funções disponíveis para tais features. Somente existe uma feature livre ativa em um dado instante. Para que a seleção seja visível adota-se o traçado de um símbolo indicador da origem do sistema de coordenadas da feature livre ativa, quando selecionada.

Disponibiliza-se também uma função para consultar os parâmetros e atributos descritores da feature livre *ativa*, para fins estritamente de visualização.

Para as operações de impactos mais significativos como remoção e modificação, adota-se o bom procedimento de pré-validação das mesmas.

4.4.3.5. Operações de Armazenamento e Leitura

Para permitir que as informações referentes às features livres sejam armazenadas e recuperadas, o FM fornece recursos para gravação e leitura dessas entidades.

Tais funções permitem que uma feature livre pertença simultaneamente a vários modelos, o que em conceitualmente falando não vai contra nenhuma das restrições apresentadas. Entretanto existe um aspecto implicitamente contido nessa possibilidade : uma feature livre que esteja contida em vários modelos se apresentará em todos eles com a mesma orientação espacial, mesmas dimensões e atributos, visto que existe somente um arquivo de dados por feature livre.

O formato de armazenamento dos dados é particular ao FM, constando basicamente os atributos descritores da feature livre mais as informações particulares à feature(canônica ou macro) pertencente àquela feature livre.

Para se obter quaisquer diferenças entre duas instâncias da mesma feature livre faz-se necessária a utilização da função de *cópia*, discutida no item anterior, onde todos as informações referentes à uma dada feature são replicados e podem ser alterados independentemente, sob um novo identificador.

Para fins de comunicação com o módulo abstrato, as features livres podem ser armazenadas também em formato particular ao módulo abstrato, de tal maneira que os raciocínios a ele atribuídos possam ser executados.

4.4.4. Peças

As peças são as entidades mais completas e mais importantes no modelo pois conceitualmente elas representam os componentes de um produto. As peças são agrupamentos de features, as quais são manipuladas segundo as regras e as relações que regem a construção de uma peça.

As peças são classificadas segundo o tipo do processo de definição de forma que caracteriza a ligação existente entre as features componentes dessas peças. As peças *simples* (ou *discretas*) são definidas sobre um meio de material contínuo. As peças *complexas* são aquelas que necessitam de um processo de ligação (fixa ou amovível). As peças complexas por sua vez, subdividem-se em *compostas e agregadas*, segundo a natureza daquele processo de ligação : se o processo é permanente (por exemplo soldagem e colagem), as peças são ditas *compostas*; se o processo é uma montagem as peças são ditas *agregadas*. Faz-se necessário salientar que as peças complexas, vistas sob uma outra perpectiva, são conjuntos de peças simples interligadas por algum processo (figura 3.9).

A nível de implementação somente as peças simples estão plenamente atendidas, enquanto as peças complexas estão implementadas parcialmente, porém a um nível suficiente de validação do conceito.

4.4.4.1. Aspectos de Implementação particulares às Peças

Antes de discutir a implementação do conceito, alguns tópicos básicos fazem-se necessário para aclarar o entendimento da discussão. Esses tópicos estão listados a seguir :

- As peças são representadas em **wireframe**, ou seja, somente as arestas das features são traçadas. Em se tratando de visualização essa seguramente não é a melhor representação que uma peça pode ter, porém como um dos requisitos do FM é precisamente permitir a edição dos componentes presentes em uma entidade, a representação *wireframe* facilita a identificação e seleção das features.

- A representação das peças é também afetada pelo fato que o modelador gráfico de suporte não permite a identificação unívoca das primitivas geométricas envolvidas em operações booleanas. Tais operações serão necessárias para incorporar as features nas peças, segundo suas naturezas. Entretanto a identificação e o reconhecimento das features torna-se complicado após a execução de operações booleanas. Assim, para as peças que possuam mais de uma feature componente, existem funções que permitem representá-las como um único sólido.

- Como já citado nos itens referentes a chanfros e concordâncias, uma restrição do modelador sólido de suporte impede que tais primitivas sejam instanciadas por aplicações externas, diretamente sobre os sólidos. Isso obriga ao tratamento de chanfros e concordâncias como primitivas volumétricas. Ou seja, a visualização da forma real gerada por tais features é disponibilizada através de funções que efetuam as operações booleanas correspondentes às aplicações das concondâncias e dos chanfros.

- Cada peça possui um sistema de coordenadas composto por um ponto de origem e um sistema de eixos *X, Y* e *Z*, chamado ***WCS*** (workpiece coordinate system). O sistema de coordenadas fornece a base de orientação da peça e sobre ele aplicam-se operações geométricas disponíveis às peças.

- Cada instância de uma peça tem um arquivo de dados particular. Isso gera a replicação de dados em um modelo com várias ocorrências de uma mesma peça, porém simplifica a manipulação. De qualquer modo, duas instâncias de uma peça em um modelo significam na verdade duas peças distintas, uma vez que pelo menos a localização das duas é obrigatoriamente diferente. As diferenças, portanto, justificam a replicação dos dados ao invés da atribuição de informações particulares a uma entidade genérica.

- A 1ª feature instanciada em uma peça deve necessariamente ser de volume positivo, e por razões de simplificação funcional, se dará sempre sobre um ponto definido pelo usuário como a origem do sistema de coordenadas corrente. As features subsequentes, no entanto, serão instanciadas em um sistema auxiliar e então incluídas na peça segundo faces e eixos de adjacência especificados pelo usuário.

- Uma peça não permite que features de volume positivo tenham intersecção não-vazia. A interferência entre elas será baseada somente em faces adjacentes. Features negativas, como definido no modelo conceitual, não poderiam intersectar-se. Em função da restrição referente aos chanfros e concordâncias, o FM não invalida intersecção entre tais tipos de features negativas.

- Cada peça será composta tanto de features canônicas como de macro-features. Em função da complexidade que pode estar envolvida em uma macro-feature, essas não são alteráveis como as canônicas.

- Existirá sempre, em um dado instante, uma única peça ativa dentro de um modelo e sobre ela aplicam-se todas as funções e tratamentos relativos à entidade peças.

4.4.4.2. Atributos

Para contemplar todo o conhecimento existente a respeito da classe *Peças*, os seguintes atributos foram definidos:

- *Identificador* da peça : mnemônico pelo qual uma determinada peça é referenciada dentro de um modelo.

- *Natureza* da peça : para fins de raciocínio do módulo abstrato, esse atributo indica se uma peça é composta somente por features positivas ou se existe quaisquer outros tipos de features (negativas e/ou modificadoras).

- Sistema de coordenadas de uma peça, representado por uma matriz de transformação homogênea contendo origem e orientação do referido sistema.

- *Lista* das features pertencentes à uma peça, sobre a qual aplicam-se todas as operações executadas sobre a referida peça.

- *Lista das adjacências*, a qual contém os elementos descritores de todas as relações de adjacência existentes em uma peça.

- Identificador do sólido representativo de uma certa peça, que é gerado pelo modelador sólido de suporte através da execução das operações booleanas de união e subtração, sobre a lista de features pertencentes à peça.

- *Tipo de material* do qual a peça será fabricada. O FM trabalha com uma tabela de materiais de características e comportamentos conhecidos, sobre os quais se pode efetuar raciocínios específicos.

- *Código da forma* comercial disponível do material do qual a peça será fabricada (bloco, barra, vergalhão, etc).

4.4.4.3. Construção

As peças são listas de features(canônicas ou macro-features), agrupadas segundo especificação precisa do usuário. O Instanciamento de uma peça inicia pela definição de *identificador, tipo e código do material da peça*, sendo o identificador arbitrariamente definido pelo projetista e tipo e código são obtidos de tabelas previamente definidas contendo materiais catalogados e disponíveis no mercado, como mostrados na tabela abaixo.

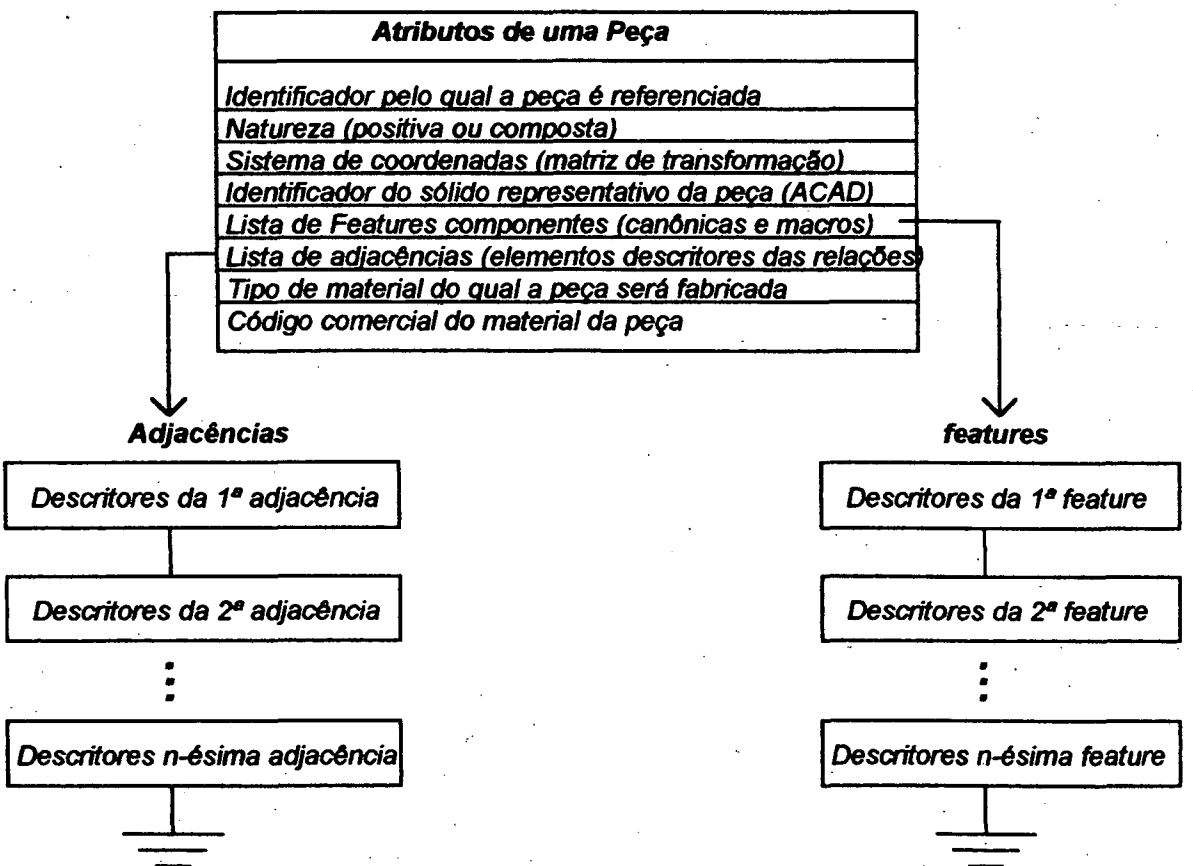
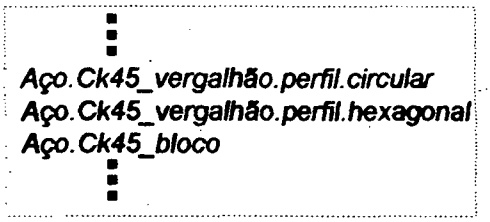


fig. 4.15 - Peça



Parte de uma Tabela de Materiais

4.4.4.3.1. Instanciamento da 1ª feature

A seguir faz-se necessário incluir as features que definirão suas forma e topologia. O instanciamento da 1ª feature da peça é completamente diferente dos outros instanciamento, visto que a 1ª feature não poderá ser adjacente a qualquer outra e não poderá ser de natureza negativa. A 1ª feature pode ser canônica positiva ou uma macro-feature, sendo que a segunda opção obrigatoriamente necessita da criação prévia de uma biblioteca de macro-features a partir da qual estas podem ser instanciadas.

Em termos de posicionamento e orientação da 1ª feature, são arbitrariamente definidos pelo usuário tanto o ponto de origem quanto o sistema de eixos coordenados sobre o qual aquela feature deve ser disposta.

4.4.4.3.2. Instanciamento e inclusão de outras features

Após o instanciamento da 1ª feature, os requisitos para a inciusão de novas features são maiores pois existe a necessidade de captura de informações relacionadas as faces e as ligações que envolvem as relações inter-features. Basicamente, as relações de adjacência, posicionamento e dependência regem os instanciamentos.

As novas features são pré-instanciadas em um sistema de coordenadas fixo, antes de serem efetivamente inciuidas à peça corrente. A pré-instância justifica-se pela necessidade de visualização dessas novas features por parte do usuário, para que as informações referentes à ligação, posicionamento e orientação sejam fornecidas. Faces e eixos de adjacência devem ser explicitamente identificamente pelo usuário para que a inciusão da nova feature seja validada segundo regras e restrições.

O pré-instanciamento não se aplica aos chanfros e concordâncias, em função da restrição apresentada pelo modelador sólido de suporte (cap.3, item 3.4.2.1.1.). Eles são instanciados diretamente sobre a aresta destino, não havendo portanto a necessidade do pré-instanciamento. Entretanto, todas as demais features seguem o mesmo procedimento de instanciamento. A figura 4.16. mostra o instanciamento de uma feature negativa, onde explicita-se as informações solicitadas.

O FM liga as duas features baseado nos eixos de contato, os quais serão coincidentes. Como cada face envolvida na justaposição possui um sistema de coordenadas particular onde cada eixo X,Y e Z têm suas orientações mais apropriadas, existe a possibilidade de orientação inesperada das faces após a ligação. Para contornar esse problema, o FM fornece a posssibilidade de rotação após o justaposição, como mostrado na figura 4.17.

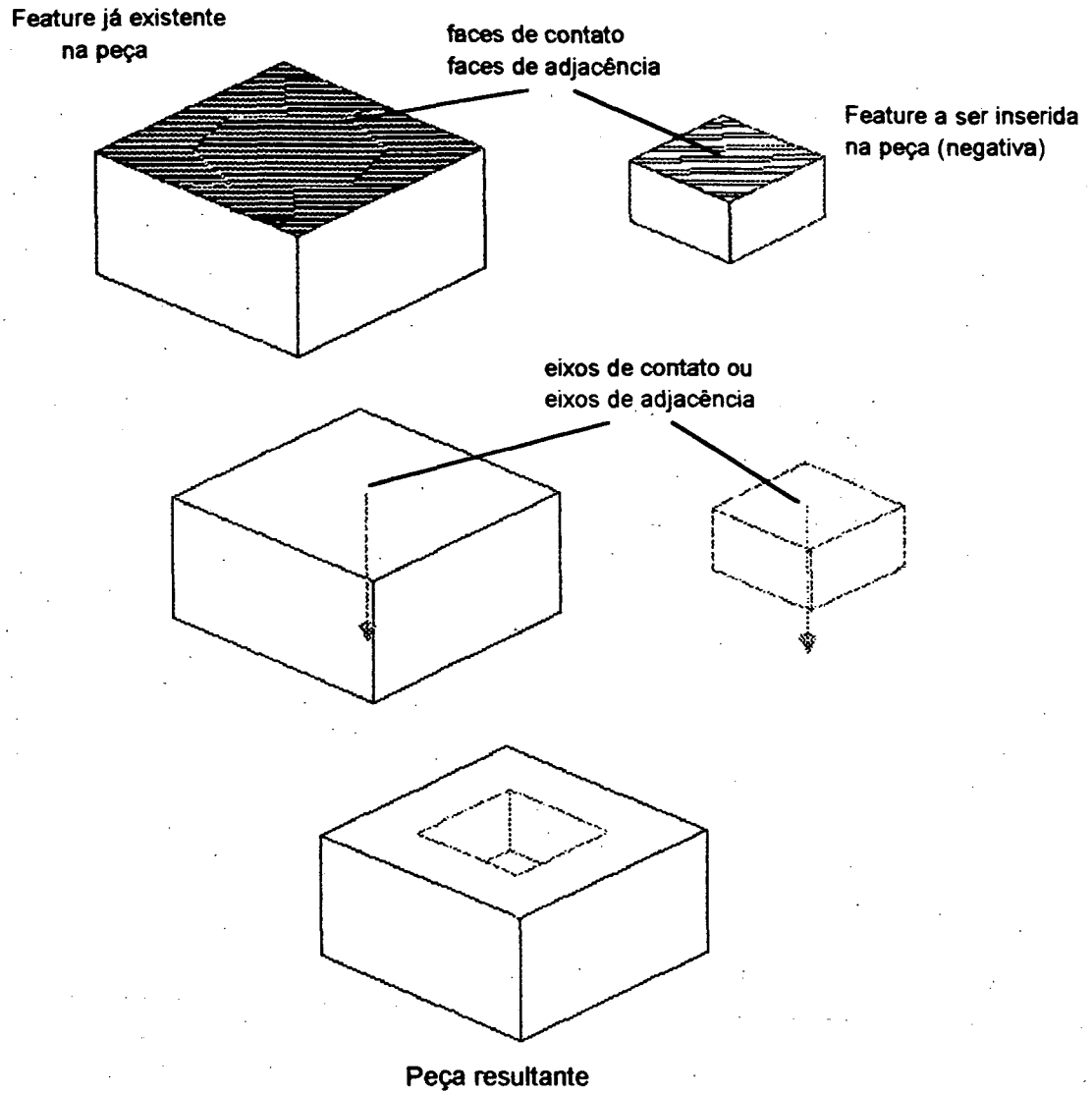


fig. 4.16 - Instanciamento de uma cavidade

O FM permite então a rotação de uma feature recém-instanciada, baseando-se no ponto de contato definido na feature-destino. Dessa maneira, a feature é orientada não mais segundo a arbitrariedade da orientação dos sistemas de coordenadas das faces envolvidas na adjacência, mas segundo as intenções reais do projetista.

Para faciiitar a entrada dos pontos de contato, o FM serve-se dos recursos fornecidos pelo ACAD de identificação de pontos, como intersecções de retas, centros de arcos, etc. Além disso, fornece uma função extra que permite identificar o centro geométrico do *envelope* de uma face, onde envelope é o menor retângulo que envolve tal face.

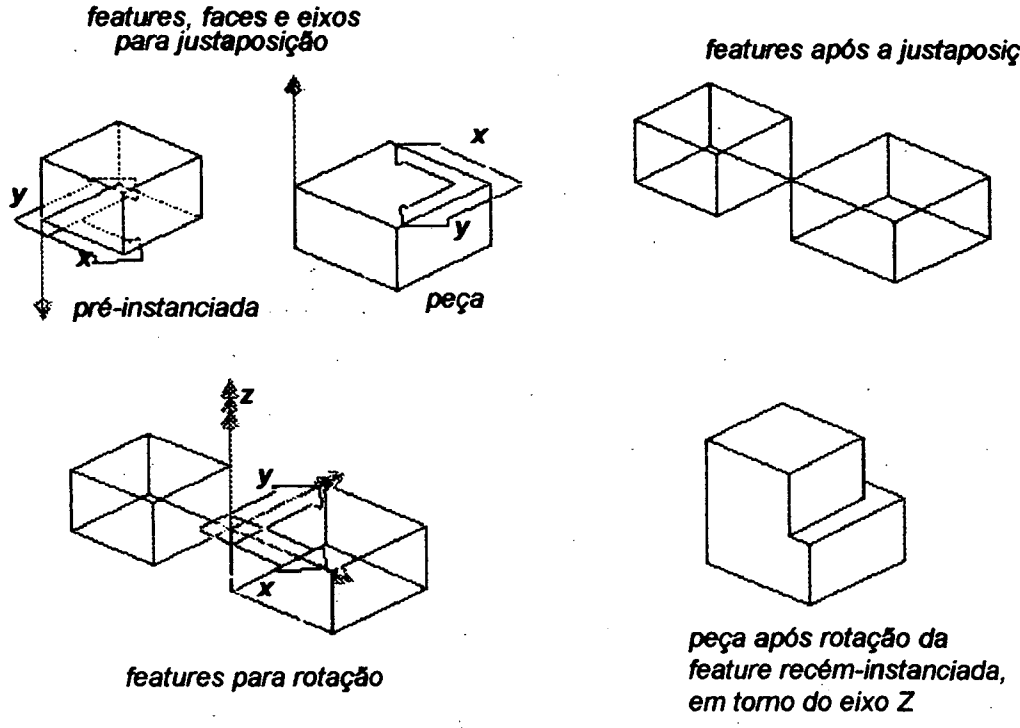


fig. 4.17 - Rotação após justaposição

Toda feature que serve de *guia* para a justaposição de qualquer outra feature é chamada ***feature de referência***. As features que não possuem qualquer feature ligada a si ou que possuem apenas features modificadoras adjacentes são chamadas ***features folhas***. As features de referência não são nem alteráveis e nem removíveis, uma vez que tais operações poderiam implicar em processamentos muito complexos e não-determinísticos por parte do FM. Portanto, tais operações devem ser hierarquizadas e executadas a partir das features folhas.

As features modificadoras não conferem o estado de feature de referência àquelas features que as suportam, no tocante apenas a remoção. Isso significa que a operação de remoção pode ser aplicada sobre uma feature que contenha chanfros e concordâncias, os quais são também removidos.

4.4.4.3.3. Operações aplicáveis às features

Algumas funções para manipulação das peças fazem-se necessárias. As operações estão divididas em classes: geométricas, de armazenamento e leitura, auxiliares.

Geométricas

As seguintes operações podem ser aplicadas sobre uma peça: *translação, rotação e escalonamento*. Recorda-se que cada peça possui um sistema de coordenadas particular representado por uma matriz de transformação homogênea, formada por ponto de origem e eixos X, Y e Z. Todas as operações geométricas aplicam-se sobre esse sistema.

Para transladar uma peça para qualquer ponto do espaço, o FM requisita o ponto destino e move a peça. Na verdade, somente a origem do sistema de coordenadas é transladada. Os recursos de identificação de ponto fornecidos pelo modelador sólido de suporte são aproveitados, tais como identificação de intersecções, centros de arcos, ponto médio de segmentos de reta, etc.

A rotação de uma peça se dá também sobre os eixos que formam o seu sistema de coordenadas, com rotações individualizadas para cada eixo. Diferente da translação, os eixos do sistema de coordenadas não são rotacionados na operação.

No escalonamento de uma peça aplica-se um fator único não particularizado por eixo, o que significa multiplicar todos os parâmetros especificados para a instância de cada uma das features existentes na peça pelo fator de escala. O fator de escala pode ser maior ou menor que 1, permitindo ampliação ou redução das peças.

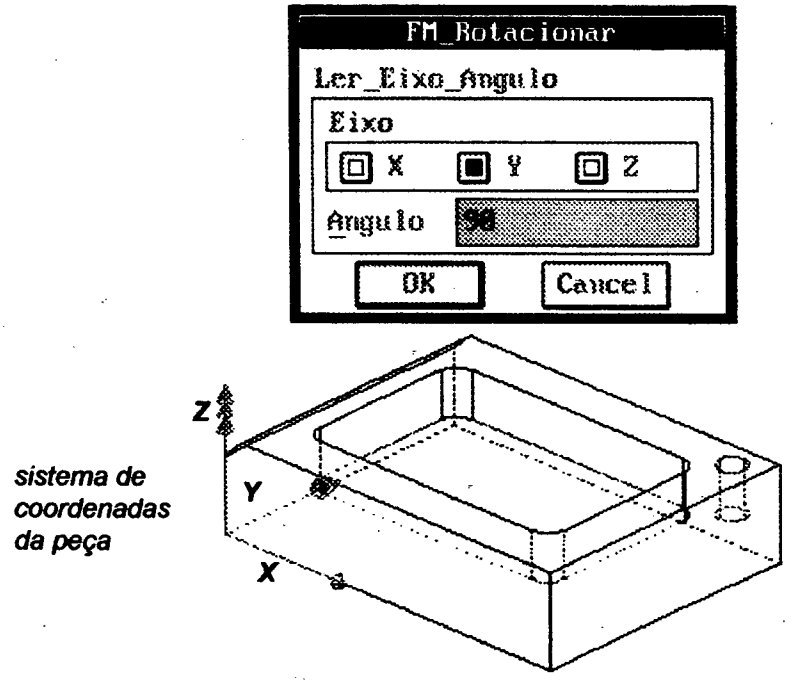


fig. 4.18 - Rotação de uma peça

De Armazenamento e Leitura

Para permitir que as peças sejam armazenadas e recuperadas, o FM fornece recursos de gravação e leitura dessas entidades. As estruturas concebidas para o armazenamento das peças guardam, em formato particular, as informações necessárias às suas reconstruções. As informações são basicamente os atributos descritores de uma peça, mais as informações existentes sobre cada uma das features em particular.

As peças são reconstruídas no momento da leitura, o que torna a carga de peças com muitas features um pouco lenta. Não foi possível aproveitar os recursos internos do modelador sólido de suporte para carga de modelo já em seu próprio formato, por conseguinte mais veloz, uma vez que os identificadores das primitivas sólidas associadas às features não são mantidos de uma sessão de trabalho à outra.

Para fins de comunicação entre o FM e o módulo abstrato, as peças podem ser armazenadas também em um formato particularmente definido para tal, contendo as informações necessárias aos raciocínios e inferências realizadas pelo módulo abstrato. A recuperação de informações oriundas do módulo abstrato não foi implementada por motivos óbvios : aquele módulo simplesmente sugere e aponta problemas, não tendo portanto poder de alteração. Logo, todas as mudanças relativas a um produto deverão ***sempre*** ocorrer no FM.

A implementação dessas funções utiliza recursos oferecidos pelo modelador sólido de suporte, que permitem navegação por entre os subdiretórios de qualquer ambiente DOS.

Auxiliares

A entidade peças, como a mais completa e mais representativa em termos de produto, possui uma série de funções que auxiliam sua construção e manipulação. Algumas aplicam-se às features pertencentes à peça e outras aplicam-se à peça como um todo, e estão descritas a seguir.

Aplicadas às features componentes : *modificação de parâmetros* e *remoção*. Aplicadas à peça como um todo: *cópia, seleção, consulta, liberação, alinhamento, edição de atributos.*

- ***Modificação e Remoção***

As funções de modificação e remoção encontram uma restrição em relação as *features de referência* (item 4.4.3.2. desse capítulo) pois tais tipos de features não são passíveis de serem modificadas ou removidas, pelas razões apresentadas naquele item. Tais funções aplicam-se somente às *features folhas.*

A modificação permite alteração nos valores nominais dos parâmetros de instanciamento de uma feature. A justaposição da feature **não** é afetada, mantendo-se relativamente no mesmo eixo de contato. A remoção elimina uma feature da lista de features componentes da peça em questão. Caso a feature a ser removida possua features modificadoras instanciadas nela, a remoção se dá a nível das modificadoras também.

No exemplo da fig. 4.19., o problema recai na segunda cavidade cuja face de entrada depende da 1ª cavidade. A remoção, se válida, cortaria o acesso a 2ª cavidade, tranformando a peça em uma espécie de paralelepípedo oco.

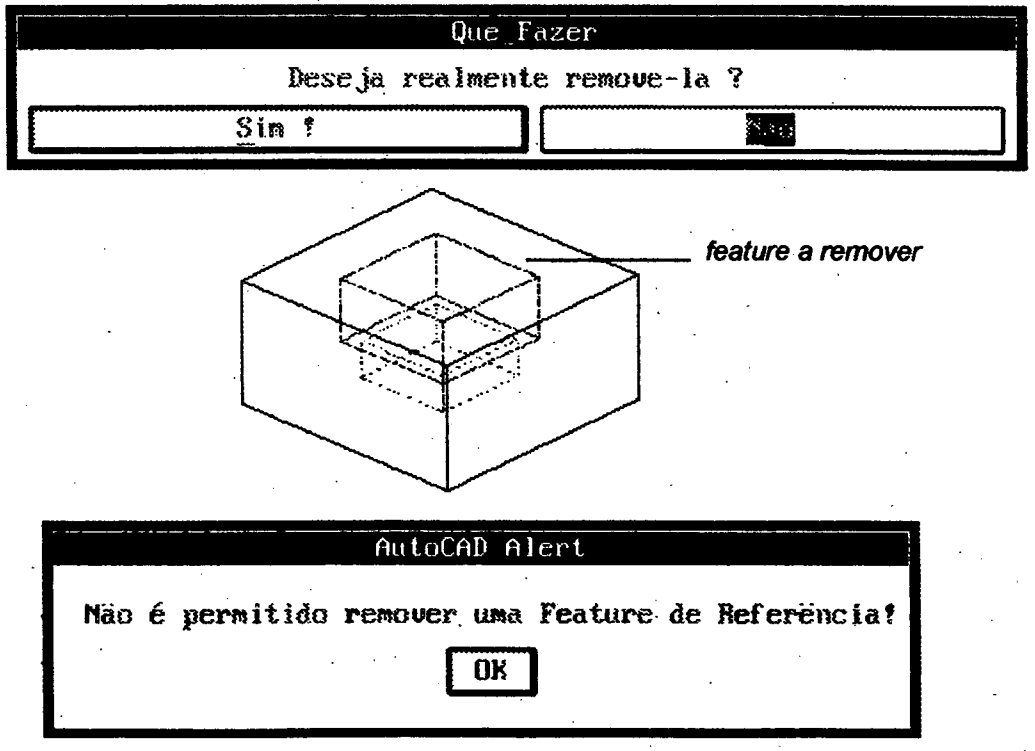


fig. 4.19 - Remoção Inválida

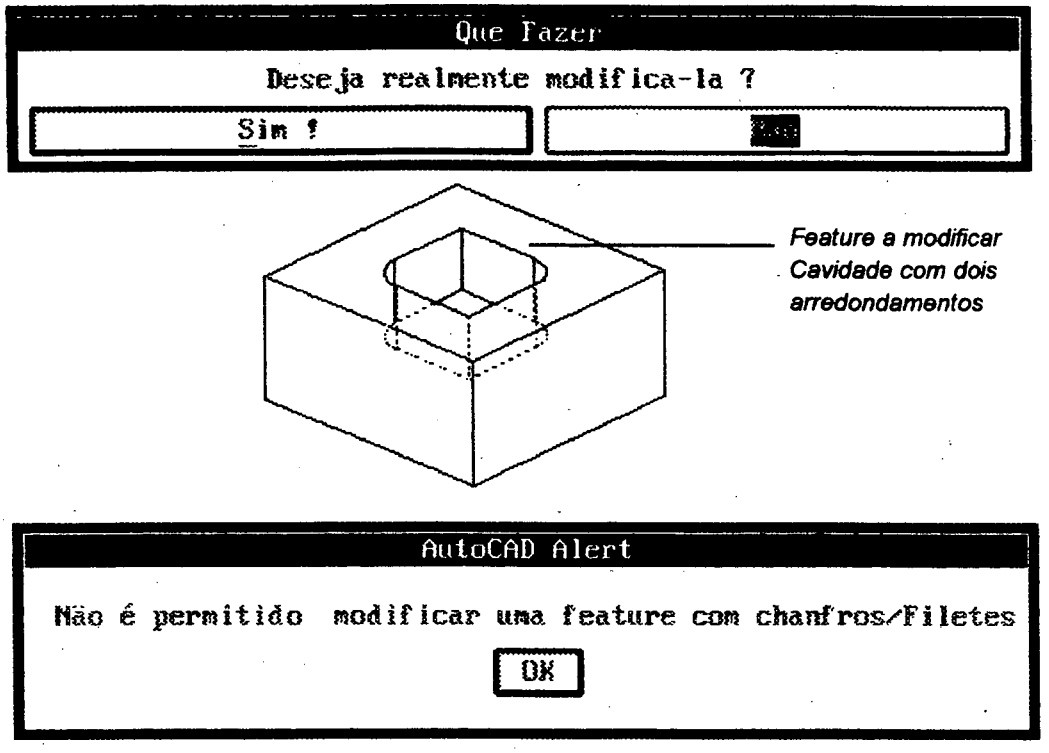


fig. 4.20 - Modificação inválida

Conforme a implementação atual do **FM**, a modificação mostrada na figura **4.20**. não é válida. As features modificadoras (chanfros e concordâncias), as quais não constituem uma dependência absoluta pois elas não evitam a remoção das features que as contêm, constituem no entanto uma dependência relativa em relação a modificação. FM não permite que as features que contêm chanfros ou concordâncias sejam modificadas.

A inclusão de regras de relacionamento posicional e validação de dimensões pode-se contornar essa restrição.

- ***Cópia***

Copiar uma peça significa duplicar todas as informações existentes sobre a peça, sob um novo identificador e normalmente em uma nova posição no espaço. A cópia permite ter duas ocorrências de uma mesma peça dentro de um único modelo.

- ***Seleção***

Função para seleção da peça ativa, visto que modelo pode conter ***n*** peças. Sobre a peça ativa atuam todas as funções disponibilizadas para as peças.

- ***Consulta***

Função que permite que todas as informações referentes às peças sejam visualizadas e consultadas. As informações referem-se não só ao nível de atributos da peça, mas também ao nível dos atributos descritores das instâncias das features que compõe a peça. Uma restrição do modelador sólido de suporte impediu que no momento da consulta dos atributos de uma certa feature, esta fosse realçada a nível de desenho.

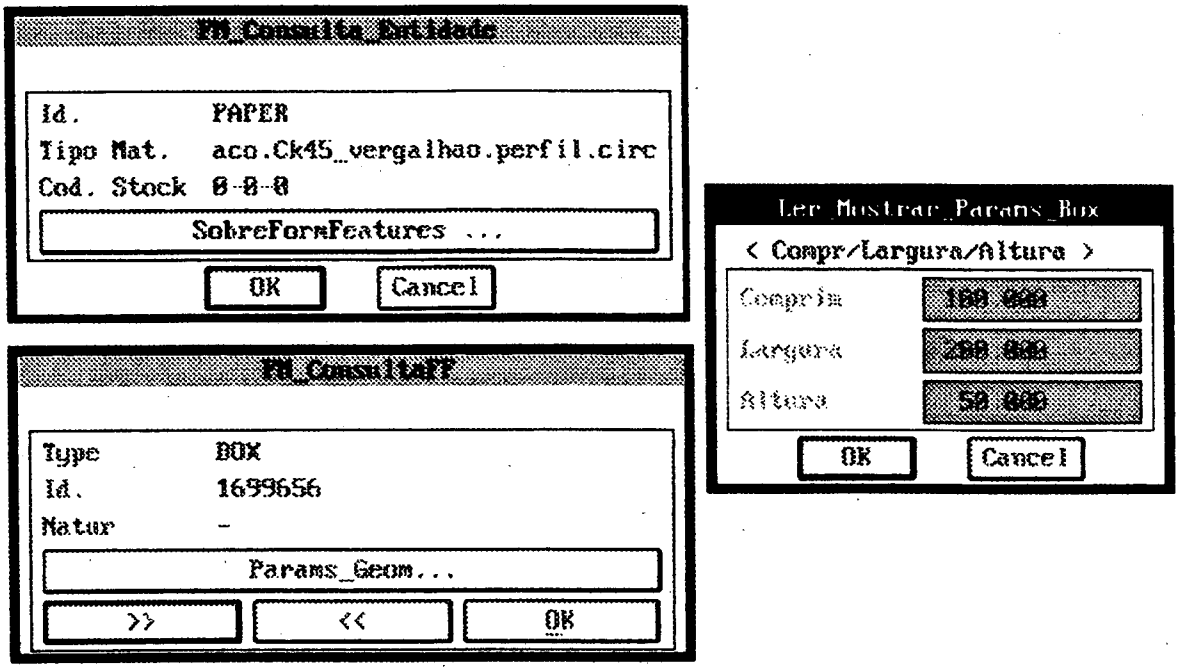


fig. 4.21 - Consulta de uma Peça

- ***Liberação***

Para possibilitar a remoção de uma peça existente em um modelo. A peça ativa se auto-remove.

- ***Alinhamento***

Para fins de orientação das peças, estas podem ser alinhadas segundo quaisquer faces. A orientação se dá apenas a nível dos eixos componentes do sistema de coordenadas de uma peça.

- ***Edição de atributos***

Para fins de alteração dos atributos referentes a *tipo e código comercial* do material que será usado na fabricação da peça.

4.4.4.3.4. Peças Complexas

Conforme apresentado no capítulo 3, as peças classificam-se em *simples ou complexas*. O elemento caracterizador das peças complexas é a existência de processos de ligação entre suas features componentes. Ou ainda como mostrado no item 3.4.5.1.4 (cap. 3), as peças complexas serão tratadas como verdadeiras uniões de peças simples. Ou seja, uma peça complexa pode ser vista como:

$$Peça\_Complexa_i = Peça\_simples_1 \cup Peça\_simples_2 \ldots \cup Peça\_simples_n$$

De modo análogo à composição das próprias peças simples, a união entre elas se dará sobre faces e pontos de contato, pertencentes a duas das features existentes nas peças simples envolvidas na formação da peça complexa, podendo-se atribuir um processo de ligação que

caracterize a união. Consoante o tipo desse processo, surgirão as peças *agregadas ou compostas*.

Assim sendo, o FM fornece recursos para que duas peças sejam justapostas, ou mais precisamente aglutinadas. Exatamente como na justaposição de features, faces e eixos de contatos nas duas peças devem ser informados. Todas as features da peça a ser aglutinada são incorporadas na outra peça (ativa), sempre considerando-se que as peças são do mesmo material.

Por restrição do modelador sólido de suporte, a peça a ser aglutinada deve estar posicionada e orientada segundo o sistema de coordenadas absoluto.

Quanto à face e ponto de contato da peça a ser aglutinada, o FM serve-se do sistema de coordenadas dessa peça como orientação de adjacência, não sendo portanto arbitrária a definição dessas informações. A face arbitrária para o contato é a face ***XY*** da feature, e o ponto de contato será o ponto ***(0,0)*** nessa face. Já na peça ativa, face e ponto de contato são definíveis.

A implementação desse conceito não está completa pois não se pode informar o processo associado à ligação das duas peças, e uma das peças é absorvida pela outra. Na verdade uma outra classe de peça deveria surgir no lugar das duas peças envolvidas na justaposição, que seria efetivamente uma instância de peça complexa conforme definição do modelo.

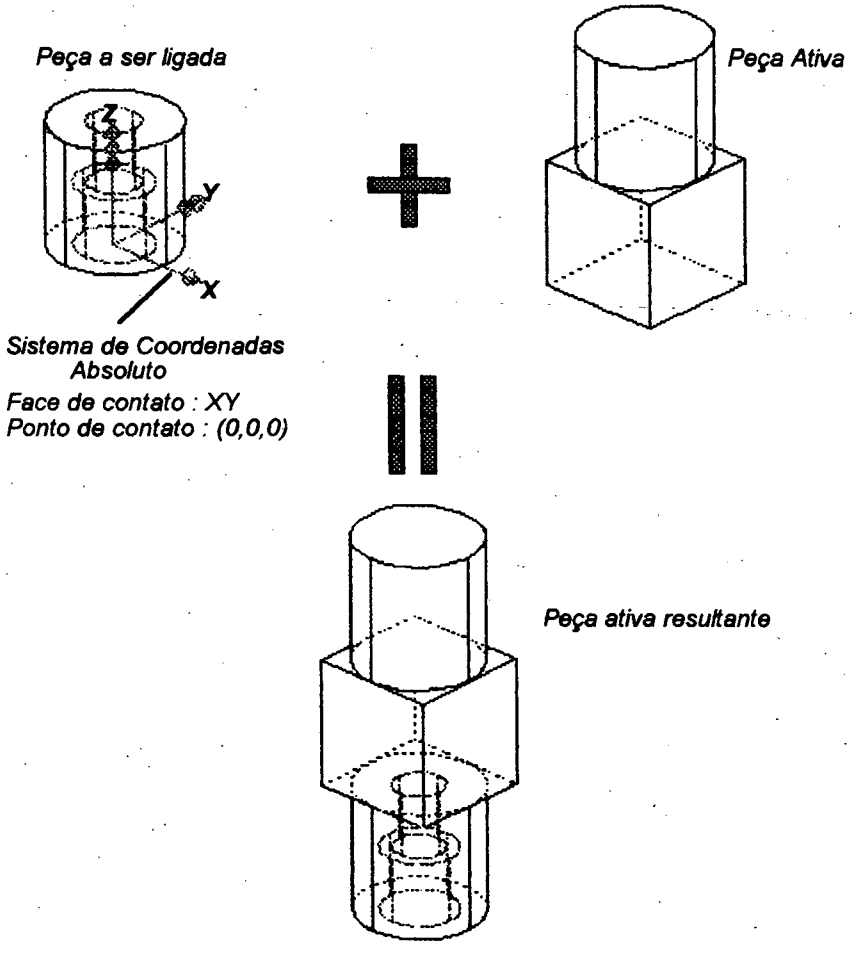


fig. 4.22 - União de duas peças

4.4.4.3.5. Ligação Peças-Features Livres

O FM fornece recursos também para que uma feature livre seja aglutinada à uma peça. De modo análogo à aglutinação de peças, faces e eixos de contatos nas duas peças devem ser informados. A feature livre é incorporada à peça ativa, considerando-se que ambas são do mesmo material, visto que o FM somente trata peças simples.

Quanto à face e ponto de contato da feature livre, o FM também serve-se do sistema de coordenadas dessa feature, não sendo portanto arbitrária a definição dessas informações. Já na peça ativa, face e ponto de contato são definíveis.

A feature livre a ser aglutinada é automaticamente transladada até a origem do sistema de coordenadas absoluto e orientada segundo os eixos absolutos, caso ela esteja posicionada em outro lugar do espaço. Isso é factível em virtude da dimensão das features livres, que é sempre de apenas uma feature. A face arbitrária para o contato é a face ***XY*** da feature, e o ponto de contato será o ponto ***(0,0)*** nessa face.

4.4.5. Modelos

Como a entidade de nível mais abstrato trabalhada no FM, os modelos representam os produtos sendo concebidos. Como mostrado no esquema da fig. 4.2, os modelos são conjuntos válidos de peças e/ou features livres. Somente existirá um único modelo ativo no FM, ou seja, da classe *modelo* somente será instanciado um objeto.

Os atributos descritores dessa classe são:

- *Identificador do modelo,* fornecido pelo próprio usuário, para permitir manipulações de gravação e leitura sobre modelos.
- *Lista dos identificadores* de todas as *peças* existentes no modelo. Tais peças podem ter sido incluídas ou totalmente construídas dentro de cada modelo.
- *Lista dos identificadores* de todas as *features livres* existentes no modelo. De modo análogo às peças, as features livres podem ter sido incluídas ou construídas dentro de cada modelo.

Um objeto modelo possui portanto, informações sobre todas as entidades *peças* e *features-livres* existentes nesse modelo. Essas informações devem ser disponibilizadas para consulta por parte do usuário.

Levanta-se na implementação desse conceito um ponto de discussão referente as entidades pertencentes a um ou mais modelos. Nenhuma restrição conceitual foi feita, porém o que pode acontecer se uma mesma peça pertencer a dois modelos? Como o modelo somente conhece os identificadores da peça, assume-se que a peça esteja corretamente definida segundo cada modelo. Entretanto, como foi visto anteriormente no item 4.4.4, cada peça possui uma matriz contendo sua localização e orientação espacial dentro do modelo. Isso elimina a possibilidade da mesma peça possuir duas orientações e duas localizações, e garante que se uma mesma peça pertence a dois modelos, ela está colocada exatamente na mesma origem e orientada da mesma maneira nos dois modelos.

Uma solução poderia ser a definição dessas matrizes a nível de modelo e não mais de peça. Porém, a integridade da peça ainda não pode ser garantida na mesma, por exemplo em termos de dimensões. Ou seja, sugere-se que cada modelo seja composto por suas próprias peças e features livres, e que o compartilhamento somente exista se as entidades forem rigorosamente iguais.

Um modelo não aceita que uma entidade qualquer figure mais que uma vez. Caso exista essa necessidade, tal entidade deverá ser replicada tantas vezes quantas necessário, porém recebendo um novo identificador para cada réplica.

Outro ponto de discussão passa pela *validação* das entidades (peças e features livres) existentes em um modelo, onde a única restrição do tipo geométrica é a não-permissão da sobreposição de entidades. A cada criação e/ou inclusão de uma nova entidade, tal processo deveria ser ativado para cumprir essa função, porém como essa tarefa pode tomar-se demasiadamente lenta em função da dimensão do modelo e das operações envolvendo os sólidos existentes, ela somente se apresenta disponível como uma validação global do modelo em um estágio final, ou sempre que o usuário o requeira.

4.4.6. Meta-Peças

As meta-peças são entidades temporárias formadas por um grupo de peças, cujos posicionamentos relativos são mantidos entre si. O objetivo básico de uma meta-peça e possibilitar a atuação de transformações geométricas sobre várias peças simultaneamente.

Quando o instanciamento da meta-peça é ativado, o FM desativa a manipulação das peças, uma vez que alterações indevidas nas peças pertencentes à meta-peça poderiam causar erros irrecuperáveis na operação normal do FM. Isso implica que é necessário desativar a meta-peça para se ter acesso novamente as operações sobre peças.

As funções disponibilizadas para o tratamento da meta-peça são : *Montagem, Desmontagem, Inclusão, Remoção, Translação, Rotação e Escalonamento.*

Montagem e Desmontagem, como o nome sugere, servem para instanciar e liberar a entidade meta-peça. Inclusão e Remoção são utilizadas para agregar/remover peças à/da meta-peça. Translação, Rotação e Escalonamento são as operações geométricas aplicáveis sobre a meta-peça.

A 1ª peça inserida na meta-peça é definida como a origem da entidade, e sobre o sistema de coordenadas dessa peça serão aplicadas todas as transformações geométricas.

4.4.7. Entidades Auxiliares

Para auxiliar uma implementação funcional do FM algumas entidades auxiliares foram construídas, dentre elas : lista de Features Canônicas, lista de Entidades, lista de Materiais, lista de Mensagens.

A lista de Features Canônicas contém a descrição de cada uma das primitivas geométricas disponibilizadas pelo ACAD/AME, e que servem de base para o FM.

A lista de entidades contém a relação e localização de todas as entidades construídas no FM (*peças, modelos, features livres*), para permitir suas recuperações através de operações de leitura adequadas a cada uma delas.

A lista de Materiais contém a descrição dos materiais disponíveis para a fabricação dos produtos concebidos, para fins de análise e seleção de processos.

A lista de Mensagens contém todas as frases enviadas pelo FM ao usuário, sejam de erro, de orientação ou de alerta.

4.5. Interface

A única interface implementada para o FM está particularizada às necessidades do módulo abstrato, baseando-se em arquivos descritores das peças construídas. A via de comunicação é uni-direcional, no sentido FM→módulo abstrato.

4.6. Aspectos Adicionais Relativos às Técnicas de Programação

Em termos de ajustamento da programação orientada por objetos às pesquisas na área de features, pode-se dizer com a experiência desse trabalho que o modelamento das classes hierarquizadas, a derivação de parâmetros entre features interligadas e a especificidade no tratamento e concepção do objeto feature é automática.

O tratamento particularizado conferido a cada instância de uma classe permite que regras e raciocínios sejam efetuados independentemente sobre cada feature presente em um modelo. Os mecanismos de herança permitem a derivação implícita de parâmetros e regras entre as features de classes distintas.

Por outro lado, esta técnica permite uma boa ligação à Base de Conhecimentos e Sistema Especialista que suporta o usuário, uma vez que o conhecimento está organizado através de frames e regras, que manipulam esses frames.

Capítulo 5

Discussão e Conclusões

Este trabalho apresentou uma pesquisa que introduz um modelo baseado em features concebido para representação de informações de projeto e fabricação referentes a um produto (peça mecânica), relacionadas as análises de manufaturabilidade, tecnologia de grupo e seleção de processos de fabricação. O objetivo principal do trabalho consistia em testar a validade conceituai desse modelo, através da implementação de dois módulos específicos trabalhando de maneira integrada: um para contemplar a fase de projeto propriamente dita (*Feature Modeller*) e outro para efetuar os raciocínios necessários e pertinentes à fabricação de um produto (*Subsistema Abstrato*)

A pesquisa seguiu a abordagem de *projeto por features*, cuja idéia central é construir um projeto desde o início já valendo-se de primitivas de criação de alto nível chamadas features. Ambos os módulos trabalham sob o paradigma de orientação por objetos, sendo que o módulo abstrato é também um sistema especialista baseado em regras e fatos. As limitações impostas por essa abordagem foram atacadas e algumas conclusões e resultados extraídos positivamente.

As features foram definidas para o escopo dessa pesquisa como *um conjunto de informações ou entidades usadas em atividades de projeto ou fabricação, dependentes de modelo e contexto.* Essa definição deixa claro que feature **não** é sinônimo de feature-de-forma, visto que features podem ser entidades sem qualquer representação física ou forma geométrica.

O enfoque principal de discussão deste trabalho reside sobre a implementação do módulo gráfico, que se adotou chamar *Feature Modeller* (FM). O FM baseia-se na abordagem de *Síntese de Elementos Volumétricos*, e fornece recursos para criação de projetos através de features, permitindo efetuar considerações de fabricação ainda nas fases iniciais de um projeto. A abordagem de Síntese de Elementos Volumétricos trabalha na junção de primitivas de volume que são definidas segundo seus significados de projeto e/ou fabricação.

As features foram implementadas valendo-se das primitivas sólidas fornecidas pelo ACAD/AME para representar suas formas, constituindo um conjunto dito canônico (figura 5.1). Atributos de natureza (*positivo e negativo*) conferiram àquelas primitivas uma característica de independência existêncial, onde as features positivas significam acréscimo de volume enquanto que as negativas significam retirada. Esta é a camada básica sobre a qual os padrões de features e as features de alto nível podem ser construídas, porém permanece a possibilidade da utilização das primitivas geométricas puras sem que o projetista seja *obrigado* a trabalhar ou raciocinar em termos de features pertinentes a outras àreas.

Atributos de material, de processos de fabricação, de classificação de simetria, de tipo da feature e de estado das faces foram adicionados às primitivas geométricas para atribuir significado de engenharia. Padrões de furos também foram implementados, embora com parametrização explícita e nominal.

Como o número de primitivas fornecido pelo ACAD é pequeno, o número de features oferecidas pelo FM também o é. Assim sendo, sentiu-se a necessidade de oferecer um maior domínio de features, o que foi feito através da criação das *macro-features*. Macro-features são construídas segundo as necessidades do projetista, servindo-se das features canônicas e amparado por um conjunto de regras e relações que norteiam a criação das macro-features.

As macro-features foram agrupadas em bibliotecas denominadas *abertas*. Tais bibliotecas também são construídas segundo as necessidades particulares dos usuários. A limitação dos conjuntos restritos de features no projeto por features foi então contornada.

As features são agrupadas para dar origem às peças, entidades mais representativas e completas de um produto. O uso de elementos de ligação inter-features possibilita a captura da topologia da peça, permitindo assim a obtenção das relações topológicas e associação processos de fabricação a esses elementos de ligação. A especificação dos processos de fabricação indicam a classe dos processos de geração da forma, os quais são supostos gerarem as features e, consequentemente, as peças (usinagem, forjamento, torneamento). Essa informação relaciona-se com a análise de seleção do processo.

As peças compõe o modelo de um produto, sobre o qual são feitas análises de manufaturabilidade, seleção e recomendação de processos de fabricação, ainda em tempo de projeto.

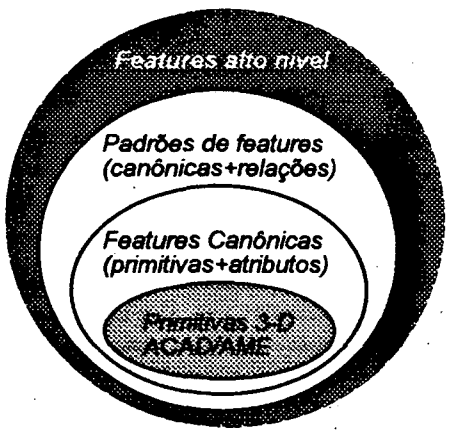


fig. 5.1 - Features no FM

5.1. Problemas e Restrições de Implementação

Os principais problemas para a implementação e validação dos conceitos presentes no modelo concebido originaram-se a partir da restrição imposta pelo programa BRITE, quanto ao caráter *comercial* das ferramentas de suporte.

As dificuldades impostas por modeladores como o ACAD+AME no tocante à manipulação das informações geométricas de mais baixo nível que servem de base para raciocínios e representações das features geraram dificuldades, uma vez que as funções disponibilizadas pela camada destinada aos aplicativos não foram concebidas para atender necessidades específicas como as contidas neste trabalho, indo desde as transformações geométricas até o tratamento de imagens 3-D como sólidos sombreáveis.

Sem dúvida uma ferramenta como a biblioteca PADL-2, desenvolvida na Universidade de Rochester - USA, facilitaria enormente a atuação direta do FM sobre as entidades por ele manipuladas. Ter-se- ia mais trabalho em implementar uma interface de alto-nível, mas uma vez que o centro das atenções do trabalho não se foca em interface, bastaria que esta atingisse um

nível aceitável. O investimento mais profundo, como foi feito, centra-se na validação dos aspectos teóricos do modelo.

Enumerativamente, os principais problemas encontrados no desenvolvimento do trabalho :

- A busca de uma linguagem orientada por objetos que pudesse trabalhar com o ambiente de desenvolvimento fornecido pelo ACAD.

- A não-existência de um identificador unívoco para as primitivas sólidas que se mantivesse inalterado entre sessões de trabalho. Isso impossibilitou a utilização dos modelos criados pelo ACAD e obrigou a reconstrução de todas as primitivas durante o processo de leitura. Felizmente a velocidade de execução do FM não é parâmetro de avaliação dos resultados.

- A impossibilidade de se identificar individualmente primitivas envolvidas em operações booleanas. Assim sendo, as peças não são tratadas como um sólido único mas sim como um conjunto de features inter-relacionadas, às quais podem ser identificadas e alteradas independentemente. Sem o tratamento de sólido único, as peças somente exibem a sua forma final quando explicitamente solicitado pelo usuário.

- A indisponibilidade de se executar chanfros e filetes via programas aplicativos dentro do ACAD obrigou a implementação de tais primitivas como primitivas de volume, criados como polylines extrudadas segundo o comprimento da aresta a ser chanfrada/filetada.

- A plataforma de hardware também não era a mais indicada, por razões de porte da pesquisa e natureza multi-usuária de projetos integrados.

5.2. Resultados Obtidos e Testes

Tomando-se os requisitos a serem cumpridos pelos sistemas de mesma natureza do que se encontra aqui implementado [Dixon90], pode-se avaliar mais especificamente os resultados aqui conseguidos. Os requisitos são :

- Existência de um grande conjunto de features implícitas para permitir conforto aos projetistas no uso de sistemas CAD baseados em feature.

- Possibilidade de definição de conjuntos particulares de features, pelos próprios usuários. Isso confere um caráter *aberto* ao sistema.

- Unicidade das features, não permitindo que diferentes features tenham formas similares ou equivalentes, levando a ambiguidade na compreensão da geométria da peça na fase de análise.

- Interações geométricas devem ser dirigidas ao usuário e não ao sistema CAD.

- Topologia da peça deve ser representada apropriadamente, de modo que relações de contiguidade, dependência e adjacência entre features-de-forma possam ser fácil e univocamente derivadas do modelo geométrico da peça.

- O objetivo do projeto deve ser capturado ao máximo, permitindo que a avaliação do projeto vá tão fundo quanto necessário, de acordo com a complexidade da análise.

- Cada tipo de feature (geometria nominal, topologia, atributos tecnológicos) pode ser representado.

- Visões diferentes das abstrações das peças podem ser consideradas.

Analisando-se sob a ótica do grupo de requisitos acima citados, pode-se dizer que o trabalho de concepção do modelo e implementação do sistema teve êxito. O oferecimento das features canônicas e dos recursos para a criação de bibliotecas de macro-features particularizadas às necessidades de cada usuário, amplia o domínio de features tratáveis a uma dimensão considerável, ao mesmo tempo em que confere o caráter aberto ao sistema.

A unicidade das features no nível em que elas foram implementadas é garantida, mesmo trabalhando-se com representação CSG. A implementação das relações de adjacência, posicionamento e dependência permitem fazer um mapeamento completo da topologia das peças a serem analisadas. A incorporação de atributos tecnológicos diretamente sobre a geometria das features conduz as análises a níveis de detalhes de faces e arestas.

A simplicidade das interações geométricas implementadas, valendo-se dos próprios recursos auxiliares do ACAD para identificação de pontos e entidades gráficas, mais as funções introduzidas pelo FM trazem enorme conforto às operações executadas pelo usuário. Exemplo das facilidades : identificações de *centro de face, intersecções e pontos-médios*.

A integração do sub-sistema abstrato, módulo dito inteligente, permitiu validar a operação de funções pós-projeto ainda na fase de concepção. As recomendações de fabricação apontam precocemente falhas de projeto, reduzem tempo e custos, e aumentam qualidade, uma vez que as correções se dão em estágios adequados. A seleção de processos aumenta a eficiência da preparação para fabricação.

Além do mais, o trabalho como ferramenta de auxílio à criação de *peças-base* foi testado em uma indústria de moldes portuguesa envolvida no projeto BRITE 3302. Como o resultado da experiência, pode-se dizer que o protótipo apresentado mostrou-se eficiente e de fácil utilização na geração das peças-bases dos moldes, falhando no ponto referente ao tratamento de superfícies complexas, não contemplado pelo modelo. Em todo o caso, o presente trabalho foi um ensaio que, por ter provado a exequibilidade dos conceitos presentes no modelo testado, permite pelo menos o estabelecimento completo das especificações para um desenvolvimento de caráter comercial.

5.3. Importância do trabalho

O trabalho mostrou que a consideração de fatores de fabricação ainda durante a fase de projeto é facilmente conseguida baseando-se em uma abordagem de projeto por features, e que as entidades presentes em todas as fases do desenvolvimento de um produto, podem ser consideradas e tratadas com auxílio de sistemas especialistas, ainda durante a fase de projeto. Ao mesmo tempo em que também se conclui que o modelo conceitual pode ser expandido para atender a Engenharia Simultânea (acrescido de novas entidades e funções) trabalhando na mesma abordagem de projeto por features, com uma representação primária do produto que pode ser exportável e tratável pelos n-processos simultâneos envolvidos.

A concepção e implementação das classes que traduzem o conhecimento acerca das entidades do modelo podem ser reaproveitadas, com os melhoramentos devidos, para trabalhos futuros na mesma linha de pesquisa.

5.4. Sugestões para a continuação do trabalho.

Apesar dos problemas relatados anteriormente, conseguiu-se a validação dos conceitos mais importantes teorizados no modelo. Com a implementação da camada de features canônicas e das macro-features, a implementação das features de alto nível consiste apenas da implementação das classes respectivas, onde o conhecimento sobre a semântica e topologia dessas features pode ser facilmente definido através dos atributos correspondentes.

Lembra-se que quanto mais se eleva o nível de definição das features mais específicas elas são. Por exemplo, um furo circular escalonado passante é muito mais preciso em termos de definição e portanto requer mais conhecimento a seu respeito, que um simples furo. Além da natureza, far-se-ão necessários definir passos de escalonamento, eixo de ligação, profundidades para cada passo respeitando a profundidade máxima derivada do adjetivo passante, e assim por diante.

Por exemplo, a implementação de uma camada mais abstrata de features consiste em definir os conhecimentos que são aplicados na geração de tais features, as quais se servirão das features de mais baixo nível (canônicas) para poderem existir. O conhecimento pode ser traduzido em regras e atributos. A implementação de elementos normalizados para uma indústria específica (no caso da indústria de moldes : extratores, guias, ...) também entra nessa camada mais abstrata. A figura 5.2 representa o conceito atualizado da concepção das features.

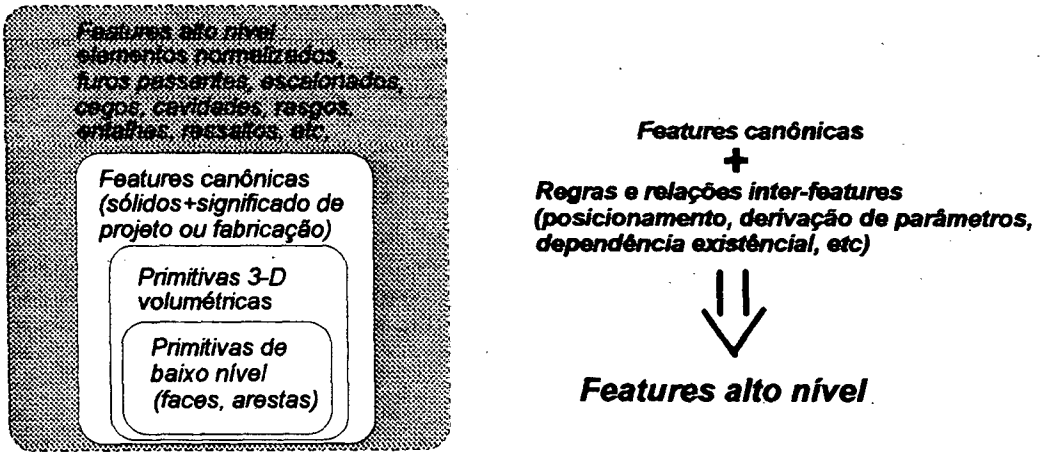


fig. 5.2 - Níveis de Abstração em Features

Pela importância e aceitação atualmente existente na indústria mundial em relação aos padrões de troca de dados definidos pelas normas do PDES/STEP; pela facilidade de definição e criação dos esquemas para modelamento de informação através do *EXPRESS*, e pela adoção do paradigma de orientação à objetos no desenvolvimento deste trabalho, torna-se extremamente fácil desenvolver uma interface que efetivamente permita ao protótipo implementado, comunicar-se com quaisquer outros sistemas via essa interface.

Como o modelo de informação concebido para suportar a definição dos produtos (*peças mecânicas*) pode ser extendido para servir de modelo-base em um ambiente de engenharia concorrente, onde as várias aplicações integradas utilizariam um único modelo de informação devidamente convertido à visão de cada uma das aplicações, o STEP entra naturalmente como padrão para o formato neutro do modelo primário das informações.

Capítulo 6

Referências Bibliográficas

Referências

[Boothroyd] G. Boothroyd, P. Dewhurst : 'Design For Assembly - Selecting the Right Method'.

[Arbab82] F. Arbab : 'Requirements and Architecture of CAM Oriented CAD Systems for Design and Manufacture of Mechanical Parts', *PhD Dissertation,* University of California, Los Angeles, USA (1982).

[Lin92] A. C. Lin, T. C. Chang : 'Product Modeling and Process Planning for 3-D Mechanical Assembly', Computers in Industry 19, Elsevier, pp. 175-184 (1992).

[Cutkosky] Cutkosky, M.R., Tenenbaum, J.M., : 'CAD/CAM Integration Through Concurrent Process and Product Design'.

[Jared84] Jared, G. E. M.: 'Shape Features in Geometric Modeling', edited by M.S. Picket and J.W. Boyse, Plenum Press (1984).

[Requicha84] Requicha, A.C.: 'Representation of Tolerances in Solid Modelling : Issues and Alternative Approaches', Solid Modeling by computers, from Theory to applications, edited by M.S. Picket and J.W. Boyse, Plenum Press, N.Y., 1984.

[Henders84] M. R. Henderson : 'Extraction of Feature Information from Three Dimensional CAD Data', Ph. D. Thesis, Purdue University, 1984.

[Miner85] Miner, R.H.: 'A Method for the representation and manipulation of geometric features in a Solid Model', *MS Thesis* Dep. Mechanical Engineering Massachisetts Institute of Technology, USA (July 1985).

[Pratt85] M. J. Pratt, P. R. Wilson : 'Requirements for Support of Form Features in a Solid Modelling System - Final Report', *CAM-I Report r-85-ASPP-01* (jun 1985).

[Luby86] Luby, S.C., Dixon, J.R., Simmons, M.K.: 'Designing with Features : Creating and Using a Feature data base for Evaluation of Manufacturability of Castings', *ASME Computers in Engineering Conference* (CIE), 1986, pp. 285-292.

[Dixon87a] Dixon, J.R., Cunningham, J., Simmons, M.K.: 'Research in Designing with Features', in Intelligent CAD, I, edited by H. Yoshikawa and D. Gossard, *Proceeding IFiP TC 5/WG. Workshop on Intelligent CAD*, Elsevier, 1987, pp. 137-148.

[Floriani88] Floriani, L.: 'Representation and Extraction of shape features in a Solid Model', NATO ASI Series, Vol. F40, Theorical Foundations of Computer Graphics and CAD. Edited by R. A. Earnshaw, Springer-Veriag Berlin Heidelberg 1988.

[Shah88a] Shah, J.J.: 'Feature Transformations Between Application Specific Feature Spaces', Computer- Aided Design, vol 23, number 4, may 1991, Butterworth, pp. 282-296.

[Chung88] Chung, J.C.H., Cook, R.L. e Patel, D.: 'Feature Based Geometry Construction for Geometric Reasoning', Proceedings for ASME Computers in Engineering Conference, San Francisco, California, USA, 1988, pp. 497-503.

[Pratt88] M. J. Pratt : 'Synthesis of an Optimal Approach to Form Feature Modelling', Proceedings for ASME Computers in Engineering Conference, San Francisco, California, USA, July-August 1988, pp. 263-274.

[Shah88b] J. J. Shah, A. Bhatnagar e D. Hsiao : 'Feature Mapping and application Shell', Proceedings for ASME Computers in Engineering Conference, San Francisco, California, USA, 1988, pp. 489-496.

[Shah88c] Shah, J.J., Rogers, M.T.: 'Feature Based Modelling Shell : Design and Implementation', Proceedings of ASME Computers in Engineering Conference, July 1988, pp. 255-261.

[Shah88d] Shah, J.J., Rogers, M.T.: 'Expert Form Feature Modelling Shell', *Computer Aided Design*, Vol. 20. No. 9, 1988, pp. 515-524.

[Pamies89] J. J. Pamies Teixeira, Gilberto D. Cunha : 'A Feature-Based Method Applied to Part Classification and Computer-Aided Design', 2nd International Conference on Automation Technology, July, 92, Taiwan.

[Requicha89] A. A. G. Requicha, J. H. Vanderbrande : 'Form Features for Mechanical Design and Manufacturing', *Proceedings ASME International Computers in Engineering Conference* (CIE), Anaheim CA, 1989, pp. 47-52.

[Suh90] Nam P. Suh : 'The Principles of Design', Oxford series on Advanced Manufacturing, 1990, by Oxford University Press Inc.

[Shah90a] Shah, J.J., Rogers, M.R., Sreevalsan, P. C., Hsiao, D., Mathew A., Bhatnagar, A., Liou B., Miller, D.W.: 'The A.S.U Features Testbed : an Overview',.*ASME Computers in Engineering Conference* (CIE), 1990 pp. 233-241.

[Faux90] I. D. Faux : 'Modelling of Components and Assemblies in terms of Shape Primitives Based on Standard Dimensioning and Tolerancing Surface Features', *Geometric Modeling for Product Engineering*, M. J. Wozny, J. U. Turner and K. Preiss (Editors), Elsevier Science Publisher B.V. (North Holland), IFIP, 1990, pp. 259-275.

[Pratt90] M. J. Pratt : 'An Hybrid Feature-based Modelling System', 1990, *Advanced Geometric Modelling for Engineering Applications*, edited by F. L. Krause and H. Jansen, Elsevier Science Publishers B.V (North Holand), 1990, pp. 189-201.

[Dixon90] J. R. Dixon, E. C. Libardi, E. H. Nielsen : 'Unresolved Research in Development of Design-with-Features Systems', *Geometric Modeling for Product Engineering, M. J. Wozny, J. U. Turner and K. Preiss* (editors), Elsevier Science Publishers B.V (North-Holland), IFIP 1990, pp. 193-196.

[Varady90] T. Várady, B. Gaál, G. E. M. Jared : 'Identifying Features in Solid Modelling', *Computers in Industry,* Elsevier (1990) **14**, pp. 43-50.

[Shah90b] J.J. Shah, Mathew A.: 'Experimental Investigation of the STEP Form Feature Information Model', *CAD*, Vol. 23, No. 4, 1991, pp. 282-296.

[Shah90c] J. J. Shah, D. Hsiao, R. Robinson : 'A Framework for Manufacturability Evaluation in a Feature Based CAD System', *NSG Design & Manufacturasbility Research Conference*, Tempe, AZ, January 1990, pp. 61-66.

[Wingard91] L. Wingard : 'Introducing Form Features in Product Models, A Step Towards CADCAM with Engineering Terminology', *Licenciate Thesis*, Departament of Manufacturing Systems, Royal Institute of Technology, Stockholm, 1991.

[Shah91a] J. J. Shah : 'Conceptual Development of Form Features and Feature Modelers', *Research in Engineering Design*, Vol. 2, 1991, pp. 93-108.

[Shah91b] Shah, J.J.: 'Assessment of Features Technology', *Computer-Aided Design,* Vol. 23, No. 5, june 1991, pp. 331-343.

[Mullins91] S. H. Mullins, D. C. Anderson : 'Feature-based Tolerance Representation for Design and Analysis', *Journal of Design and Manufacturing*, Chapman, (1991), **1**, pp. 107-118.

[Cutkosky91] M.R. Cutkosky, J. M. Tenenbaum : 'Providing Computational Support for Concurrent Engineering', *International Journal of Systems Automation : Research and Applications* (SARA) 1, pp. 239-261 (1991).

[Salomons92] W. Salomons, F. J. A. M. van Houten, H. J. J. Kals : 'Review of Research in Feature-based Design', *Journal of Manufacturing Systems*, Vol. 12, No. 2, pp. 113-132, 1992.

[Falcidieno92] Falcidieno, F. Giannini, Porzia, C. and Spagnuolo, M.: 'A Uniform approach to represent features in different application contexts', Computers in Industry **19**, Elsevier, pp. 175-184 (1992).

[Hwang92] J. L. Hwang, M. R. Henderson : 'Applying the Perceptron to three-dimensional feature recognition', *Journal of Design and Manufacturing* (1992) **2**, Chapman, pp. 187-198.

[Lis92] T. H. Lis, G. W. Fischer: 'Developing Feature-based Manufacturing Applications Using PDES/STEP', 1992.

[Gardan93] Y. Gardan, C. Minich : 'Feature-Based Models for CAD/CAM and their Limits', *Computers in Industry* **23** (1993), Elsevier, pp. 3-13.

[Wang93] M. T. Wang, M. A. Chamberiain, A. Jonea and T. C. Chang : 'Manufacturing Feature Extraction and Machined Volume Decomposition in a Computer-Integrated Feature-Based Design and Manufacturing Planning Environment', *Computers in Industry* **23** (1993), Elsevier, pp. 75-86.

[Klauck93] C. Klauck, J. Schwagereit : 'GGD - Graph Grammar Developer for features in CAD/CAM', 1993, pp. 573-580

[Bronsvoort93] W. F. Bronsvoort, F. W. Jansen : 'Feature Modelling and Conversion : Key Concepts to Concurrent Engineering', 1993, *Computers in Industry* 21, Elsevier (1993), pp. 61-86.

[Pamies93a] Pamies, J.J.T., Cunha, G.D.: 'A Feature-Based Model for Design and Manufacturing Knowledge Representation',Proceedings of 12th. Brazilian Congress of Mechanical Engineering, Brasilia, dec 1993.

[Pamies93b] Pamies, J.J.T., Cunha, G.D.: 'The Design for Manufacture Applied to the Computer-Aided Design of Mechanical Parts', Proceedings of 12th. Brazilian Congress of Mechanical Engineering, Brasilia, dec 1993.

[Qiao93] L. H. Qiao, C. Zhang, T. H. Liu, H. P. B. Wang, G. W. Fischer : 'A PDES/STEP Based Product Data Preparation Procedure for Computer-Aided Process Planning', *Computers in Industry,* Elsevier (1993) **21**, pp. 11-22.

[Tavares93] M. Tavares, R. J. Gonçalves, M. M. Barata, A. S. Steiger-Garção, 'Integração de Aplicações em Ambientes de Manufactura : Rumo a Padronização', 1993

[Mayer94] R. J. Mayer, C. J. Su, T. L. Sun, T. A. Wysk : 'ECTOF : a Feature Representation Technique for Concurrent Engineering Applications', *Journal of Design and Manufacturing*, Chapman (1994) **4**, pp. 49-65.

[Jasthi94] S. R. K. Jasthi, A. V. S. R. K. Prasad, G. Manidhar, P. N. Rao, U. R. K. Rao, N. K. Tewari : 'A Feature-based Part Description System for Computer Aided Process Planning', *Journal of Design and Manufacturing*, Chapman (1994) **4**, pp. 67-80.

[Cunha94] G. D. Cunha, *Tese de Doutoramento a ser apresentada junto ao Depto. de Engenharia Mecânica da Faculdade de Ciências e Tecnologia, da Universidade Nova de Lisboa.*

[BRITE94] Documentos de Especificação do Projeto ***BRITE-EURAM 3302***.

Bibliografia

Industrial Automation System - Product Data Representation and Exchange - Part 48 : Form Features

Manuais de Auxílio ao Desenvolvimento de Aplicativos e utilização do AME no ambiente ACAD.

Manuais da Linguagem de desenvolvimento do protótipo HighC/C++.

[Stoll88] Stoll, H. W.: 'Design for Manufacture',.*Manufacturing Engineering*, January 1988, pp. 67-73.

[SPRIT90] F. L. Krause, S. Kramer, E.. Rieger : 'Feature as a Basis for Intelligente Product Modelling', Projeto IMPPACT - ESPRIT, Comunidade Européia.

[Warman90] E. A. Warman : 'Integration Revisited - An Appraisal of the State of the Integration of CAD', *Computers in Industry* **14**, 1990, pp. 59-65.

[Sycara91] K. P. Sycara, C. M. Lewis : 'Modeling Group Decision Making and Negotiation in Concurrent Product Design', *International Journal of Systems Automation : Research and Applications* (SARA) **1**, pp. 217-238 (1991).

[Shah91c] J.J. Shah, S. Sen, S. Ghosh :'An Intelligent Environment for Routine Mechanical Design', Computer in Engineering Conference, ASME, San Jose, 1991.

[Fu92] Z. Fu, A. de Pennington : 'Reasoning About Machinable Features for Automated Process Planning', *Journal of Design and Manufacturing* (1992) **2**, Chapman, pp. 225-237.

[Nnaji92] B. O. Nnaji, P. B. Jagtap, J. B. Sadrach and S. C. Yeh : 'Automatic Precedence and Spanning Vector Generation for Assemly Planning', *Journal of Design and Manufacturing* (1992) **2**, Chapman, pp. 211-224.

[Kim92] J. Y. Kim, Ravio, Mitall, P. O'Grady, R. E. Young : 'Process Selection for Concurrent Engineering in the Domain of Rotational Parts', *Journal of Design and Manufacturing* (1992) 2, pp. 199-209.

[Liou92] F. W. Liou and D. J. Suen : 'The Development of a Feature-Based Fixture Process Planning System for Flexible Assembly', *Journal of Manufacturing Systems* Vol 11/No. 2, pp.102-112.

[Kang92] T. S. Kang, B. O. Nnaji : 'Feature Representation and Classification for Automatic Process Planning Systems', *Journal of Manufacturing Systems*, Vol. 12, No. 2, pp. 133-145, 1992.

[Zeiler91] W. Zeiler :'Object-Oriented Hybrid Intelligent CAD System', *Computers in Industry* **20** (1992), Elsevier, pp. 1-9.

[Gardan93] Y. Gardan, C. Minich : 'Feature-Based Models for CAD/CAM and their Limits', *Computers in Industry* 23 (1993), Elsevier, pp. 3-13.

[Schulte93] M. Schulte, C. Weber, R. Stark : 'Functional features for design in Mechanical Engineering', *Computers in Industry* **23** (1993), Elsevier, pp. 15-24.

[Nadir93] Y. Nadir, M. Chaabane, C. Marty : 'PROCODE - Automated coding system in group technology for rotational parts', *Computers in Industry* **23** (1993), Elsevier, pp. 39-47.

[Bernard93] A. Bernard : 'The Feature Approach for the Integrated Design and Machining of Forming Dies', *Robotics & Computer-Integrated Manufacturing*, Vol. 10, No. 1/2, pp. 71-76, 1993.

[Eversheim93] W. Eversheim and J. Schneewind : 'Computer-Aided Process Planning - State of the Art and Future Development', *Robotics & Computer-Integrated Manufacturing*, Vol. **10**, No. 1/2, pp. 65-70, 1993.

[Qiao93] L. H. Qiao, C. Zhang, T. H. Liu, H. P. B. Wang, G. W. Fischer : 'A PDES/STEP Based Product Data Preparation Procedure for Computer-Aided Process Planning', *Computers in Industry,* Elsevier (1993) **21**, pp. 11-22.

[Liu93] T. H. Liu, G. W. Fischer : 'Developing Feature-Based Manufacturing Applications Using PDES/STEP', *Concurrent Engineering : Research and Applications* (1993) **1**, pp. 39-50.

ANEXO 1

Sessão de Trabalho com o FM

Esse anexo contém imagens da execução do Feature Modeller, trabalhando no ambiente ACAD.

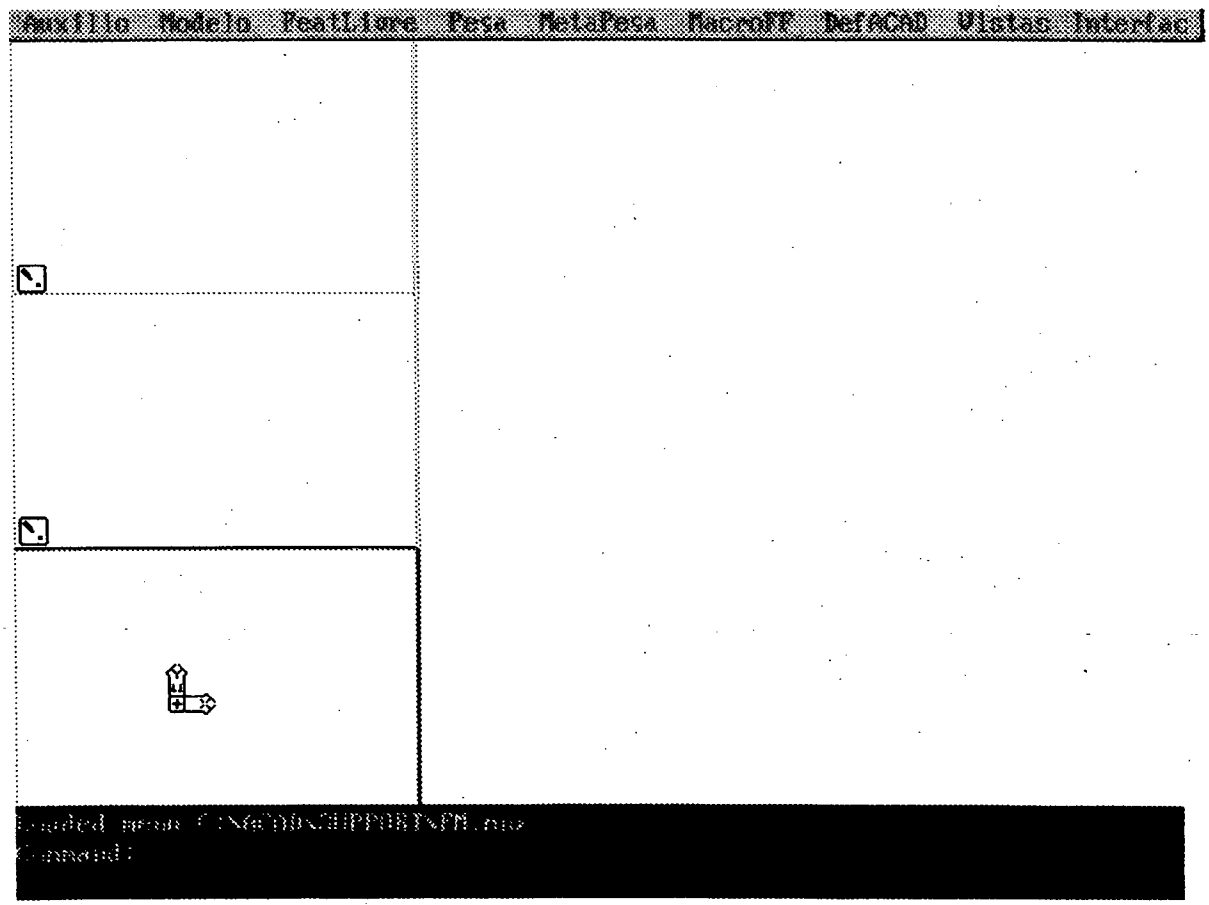


fig. A.1 - Ambiente principal do FM

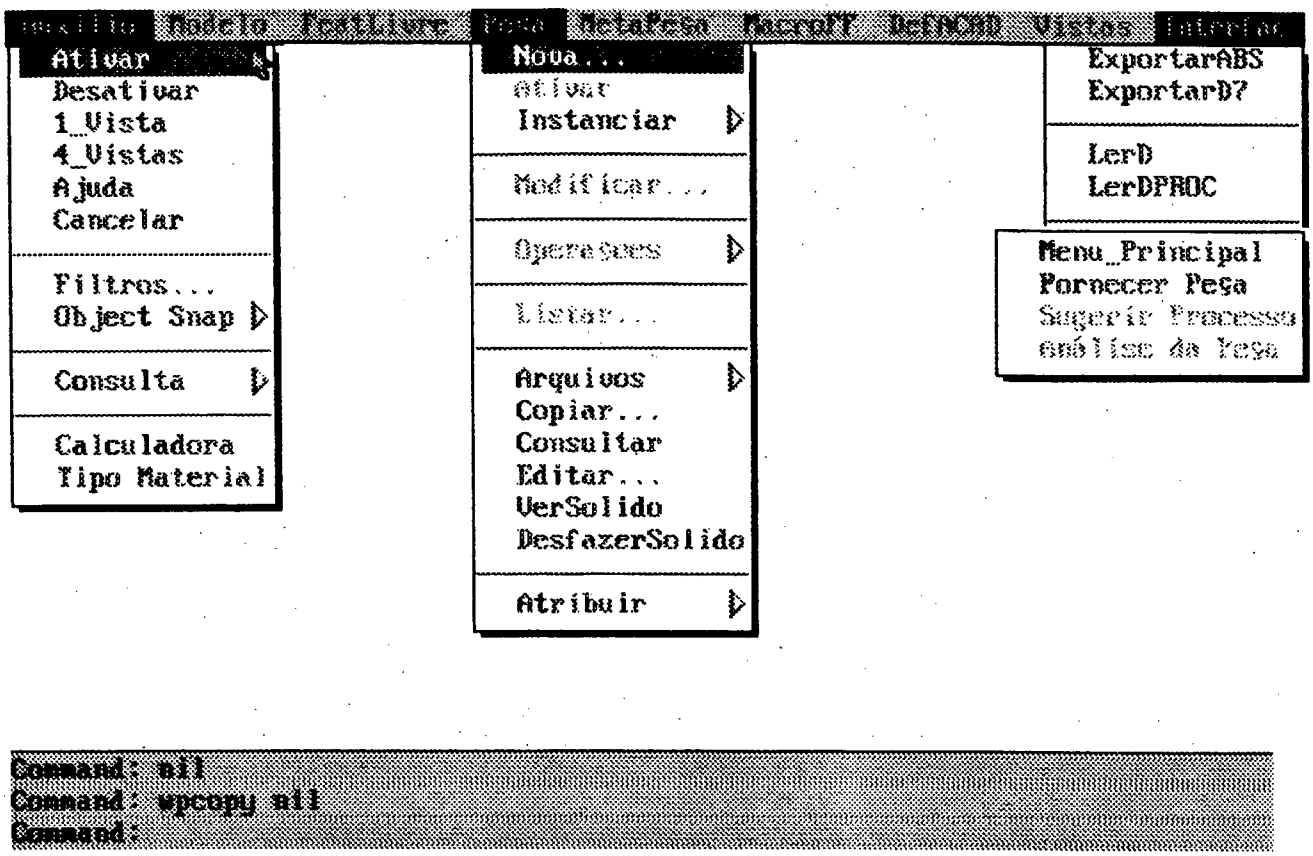


figura A.2 - Opções do menu principal

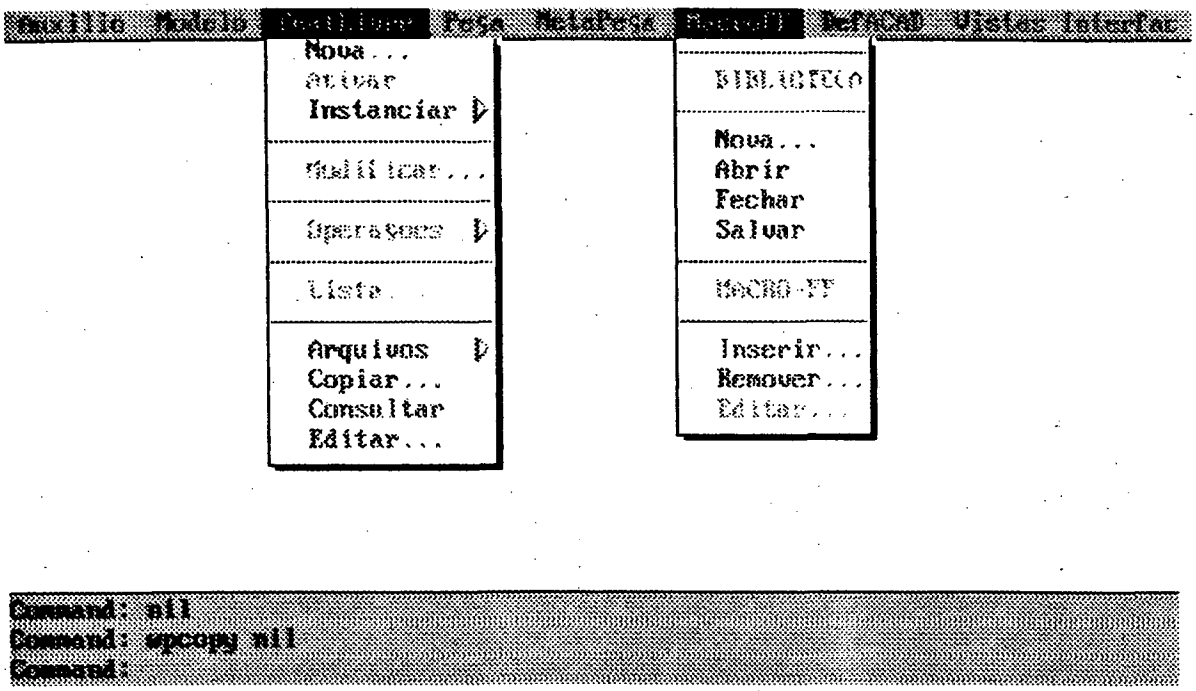


figura A.3 - Opções do menu principal

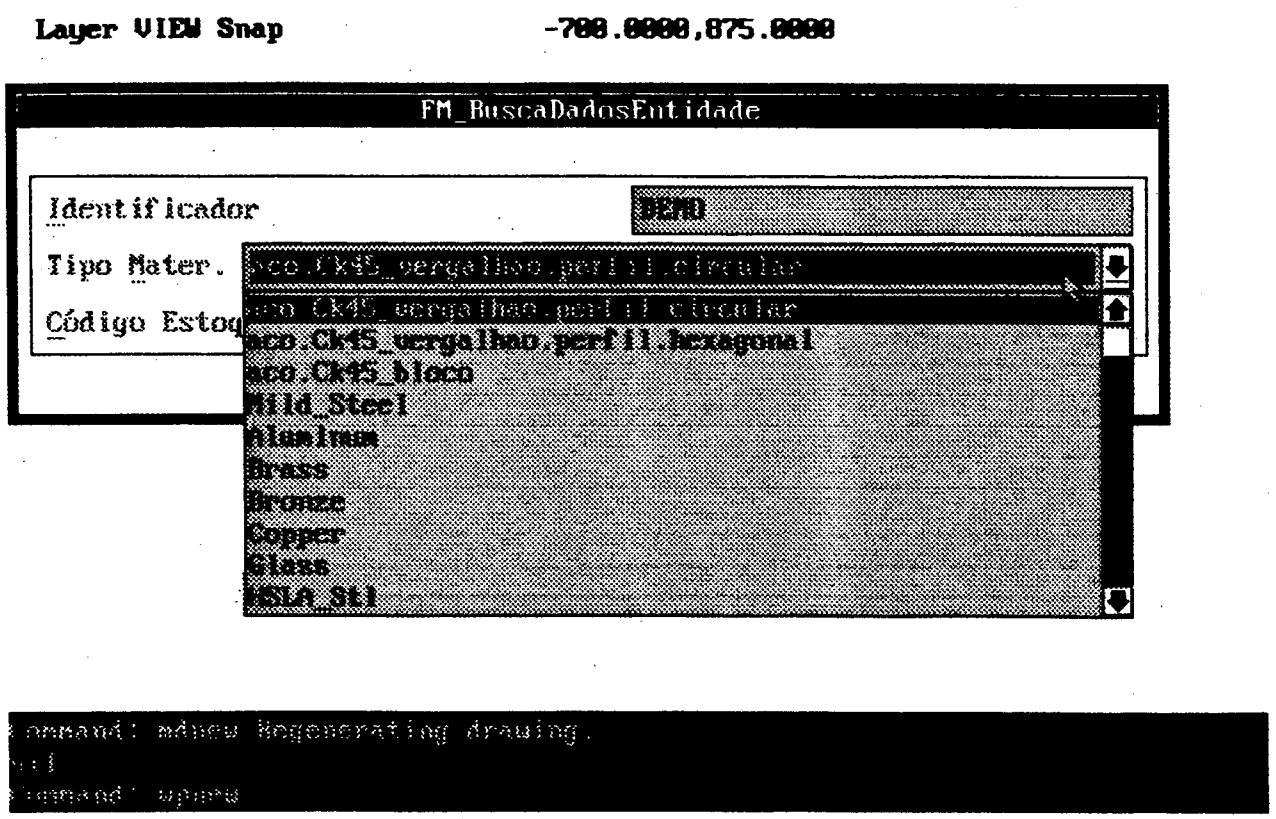


figura A.4 - Criação de uma Peça

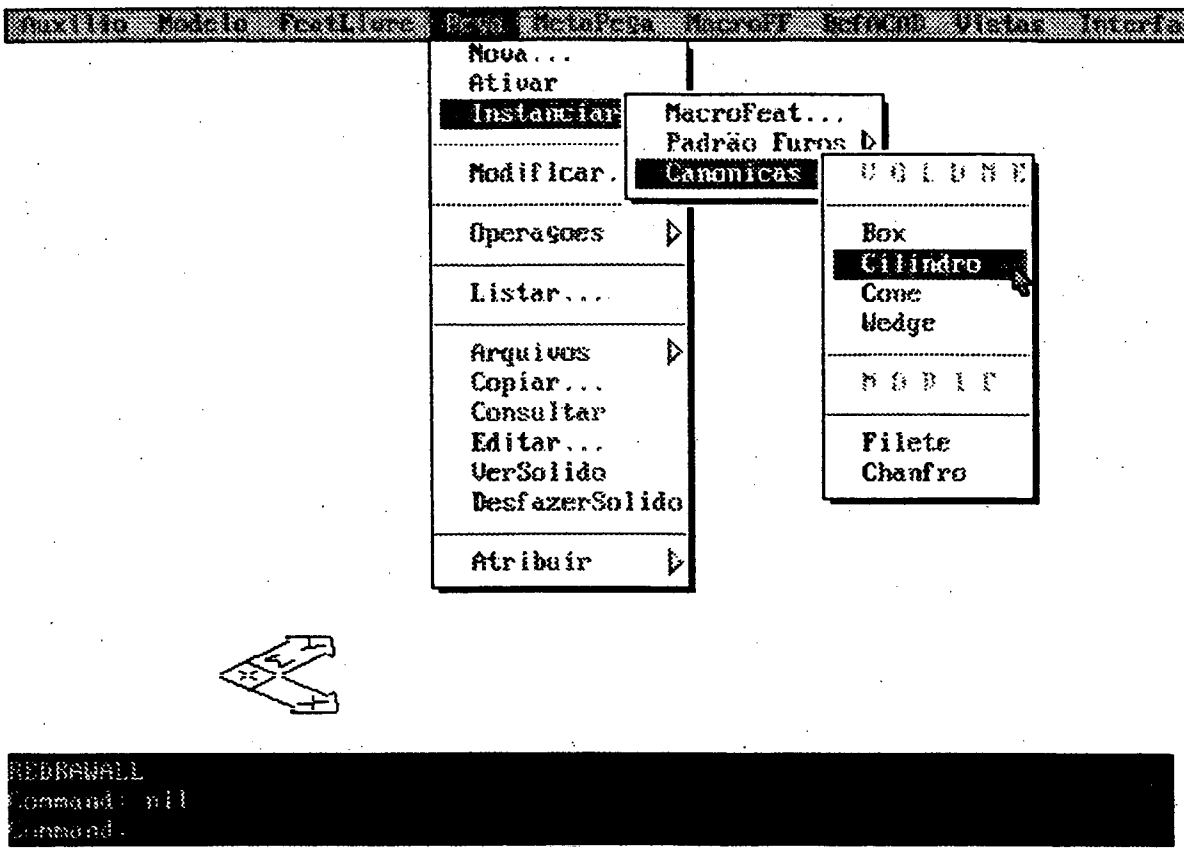


figura A.5 - Opção para Instanciamento de um Cilindro

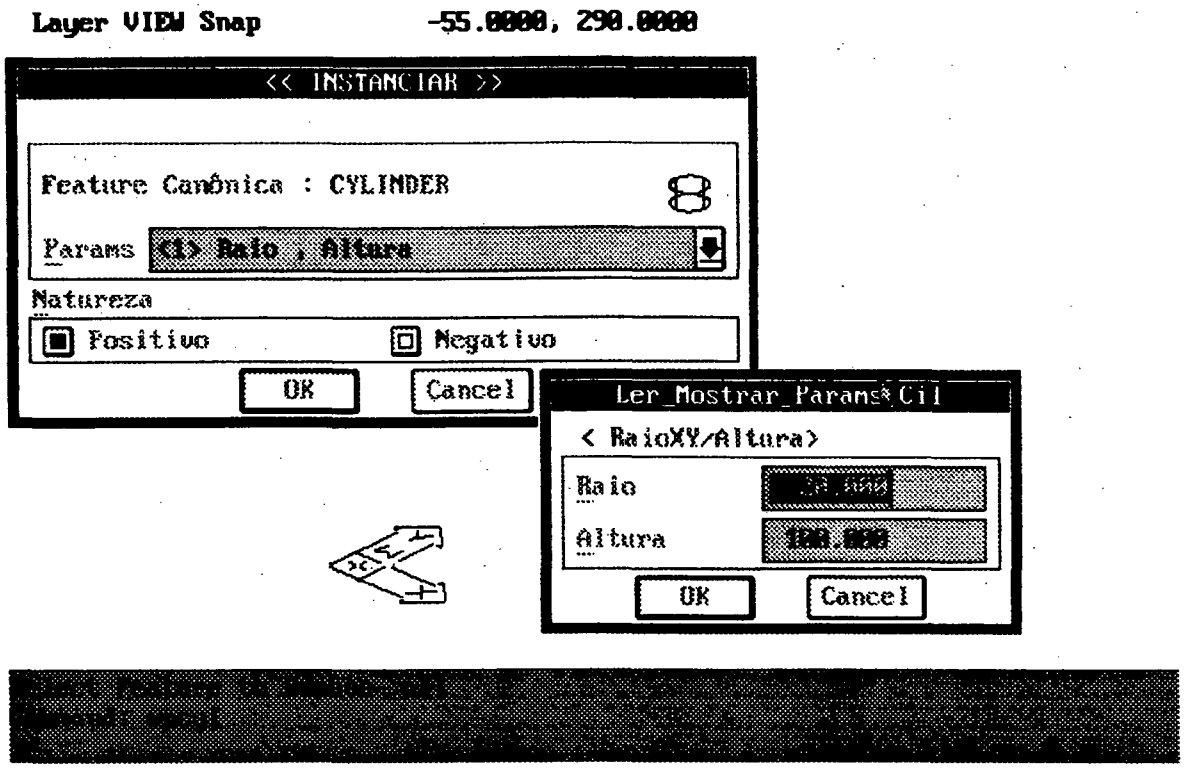


figura A.5 - Diálogo de Instanciamento de um Cilindro

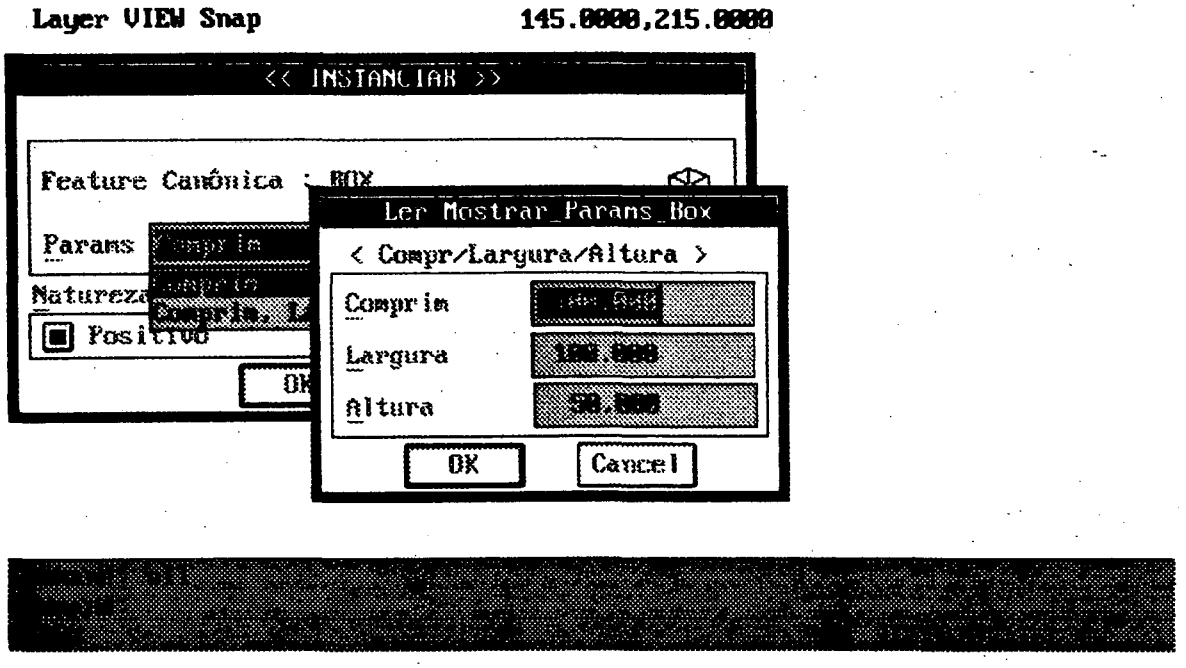


figura A.6 - Diálogos de Instanciamento de um Paralelepípedo

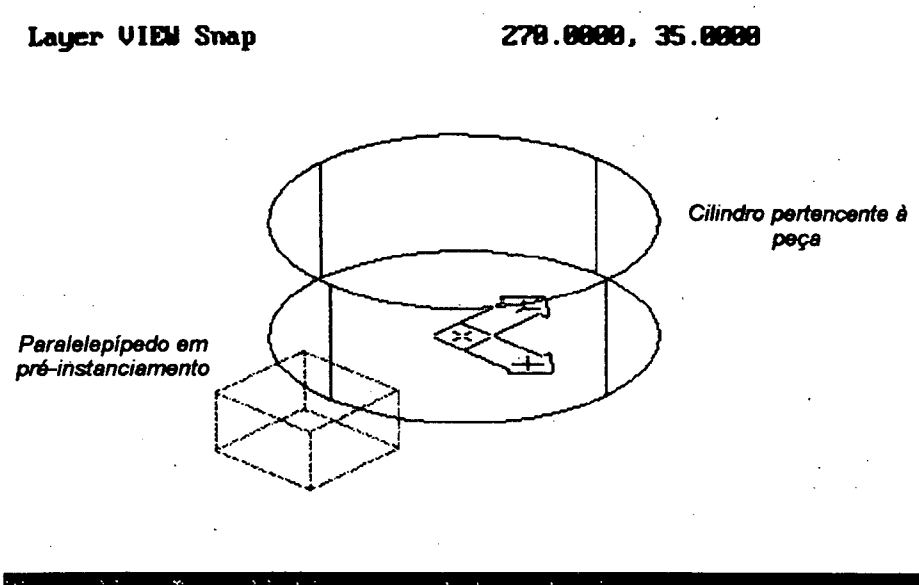


figura A.7 - Pré-instanciamento de um paralelepípedo negativo

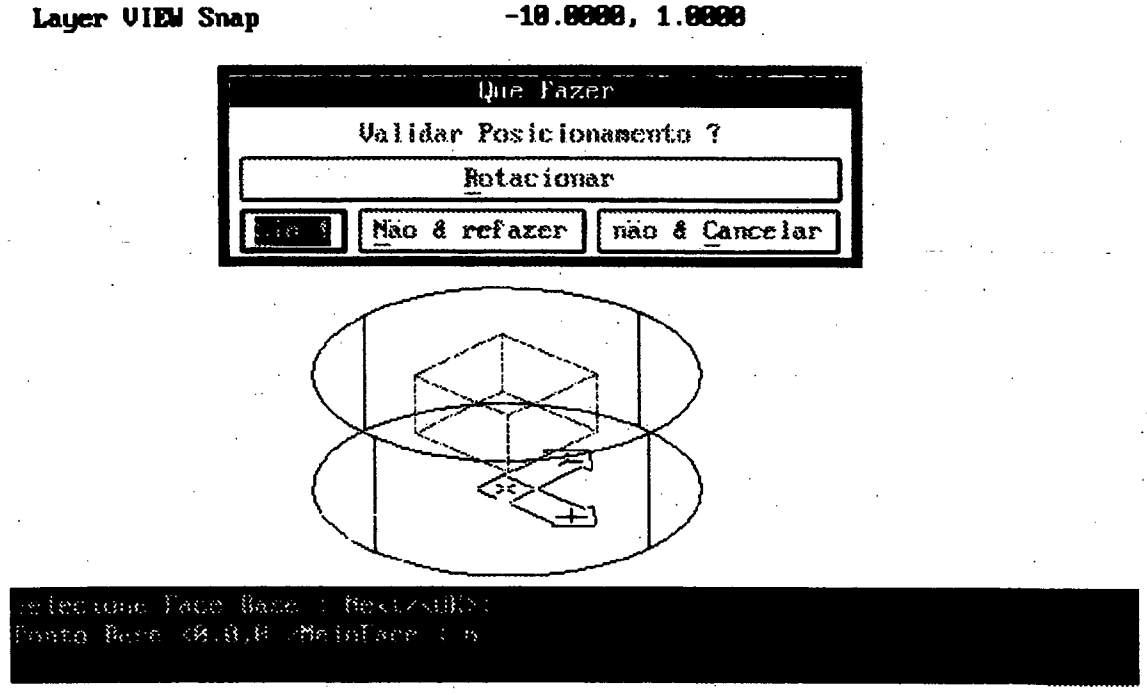


figura A.8 - Justapondo um paralelepípedo negativo a um cilindro

Layer VIEW Snap 0.0000, 1.0000

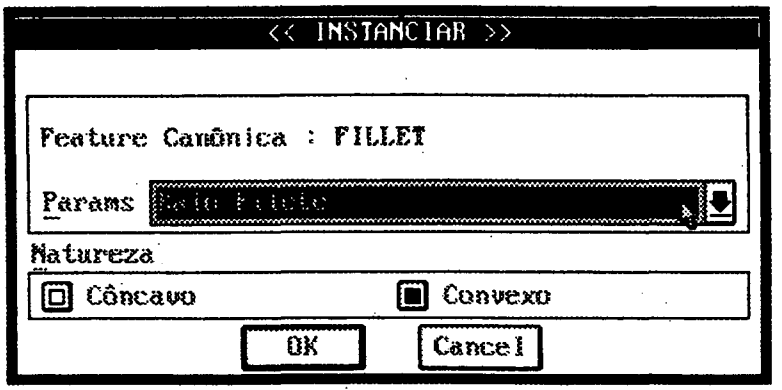


figura A.9 - Diálogo de instanciamento de um filete

Layer VIEW Snap -10.0000, 1.0000

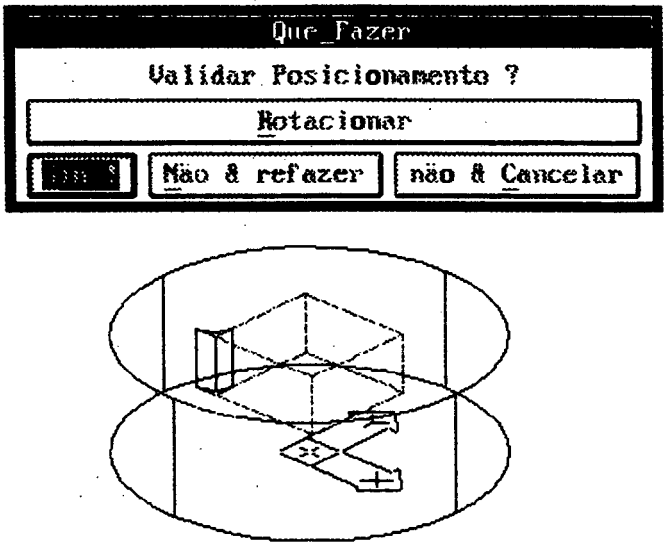


figura A.10a -Operação de rotação pós-instanciamento de um filete

Layer VIEW Snap 0.0000, 1.0000

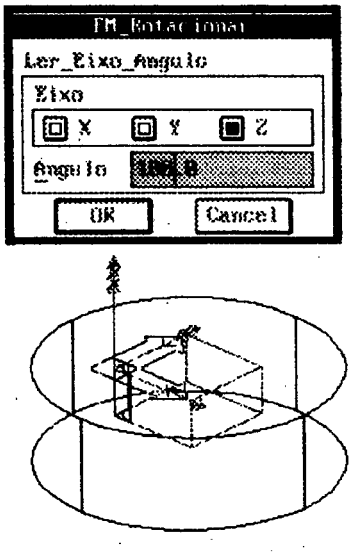


figura A.10b -Operação de rotação pós-instanciamento de um filete

Layer VIEW Snap -545.0000,665.0000

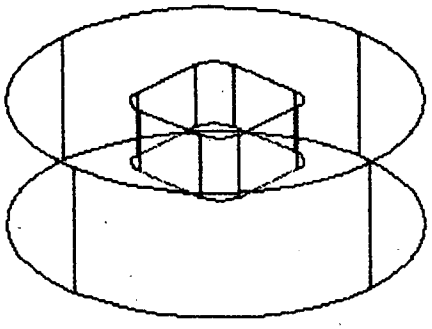
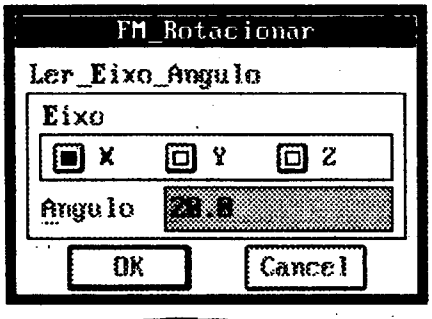
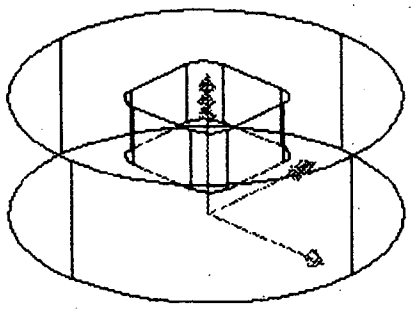
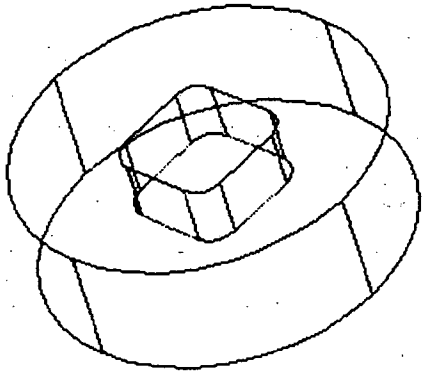


figura A.11 - Peça após instanciamento de 4 filetes na cavidade

Layer VIEW Snap -510.0000,655.0000







Peça após a rotação

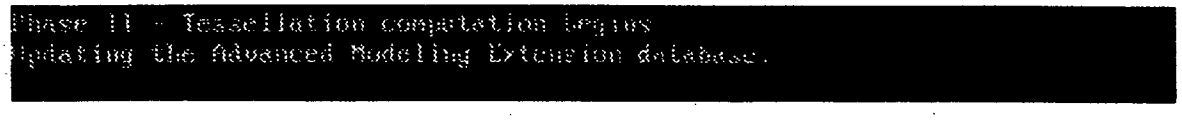


figura A.12 - Rotação de uma Peça

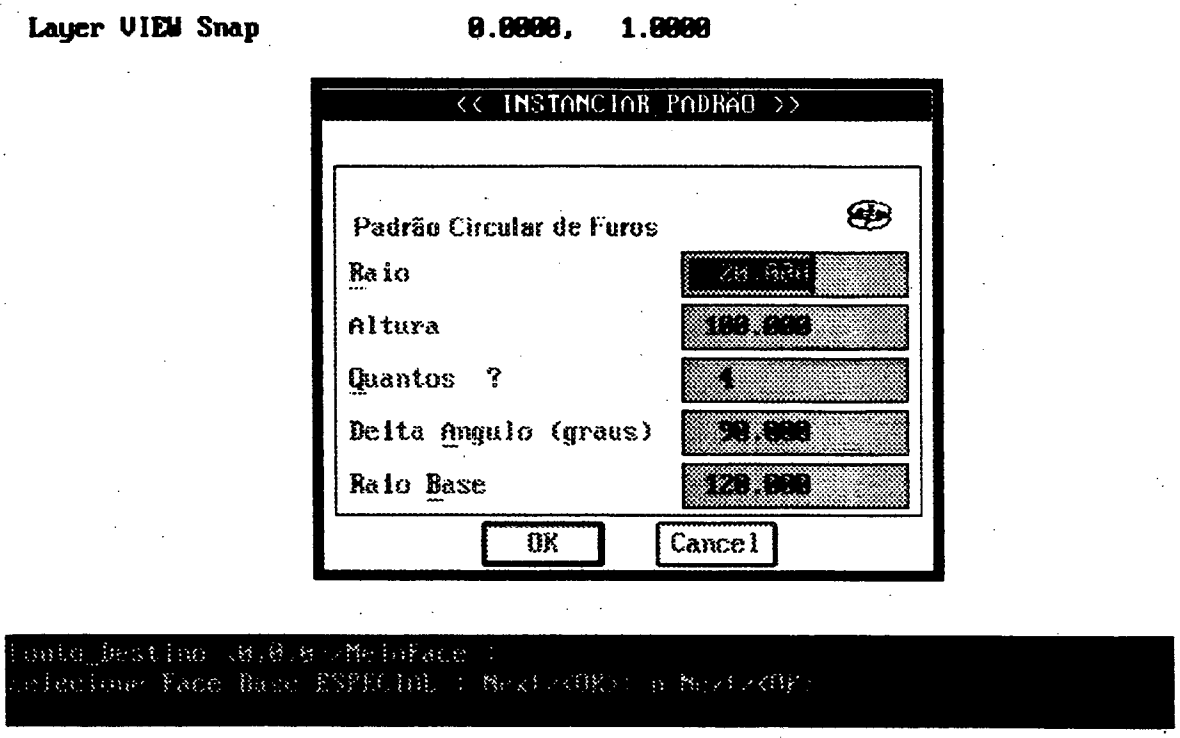


fig. A.13 - Diálogo de instanciamento de um padrão Circular de Furos

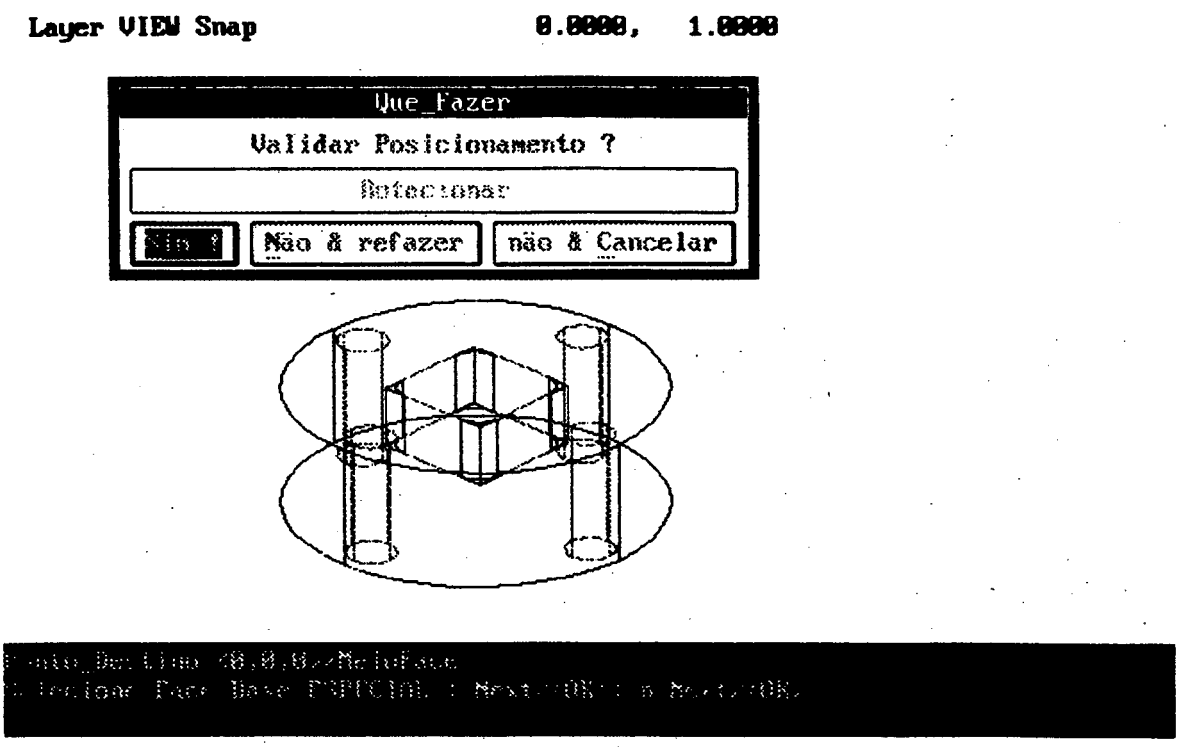


fig. A14 - Instanciamento do Padrão de Furos

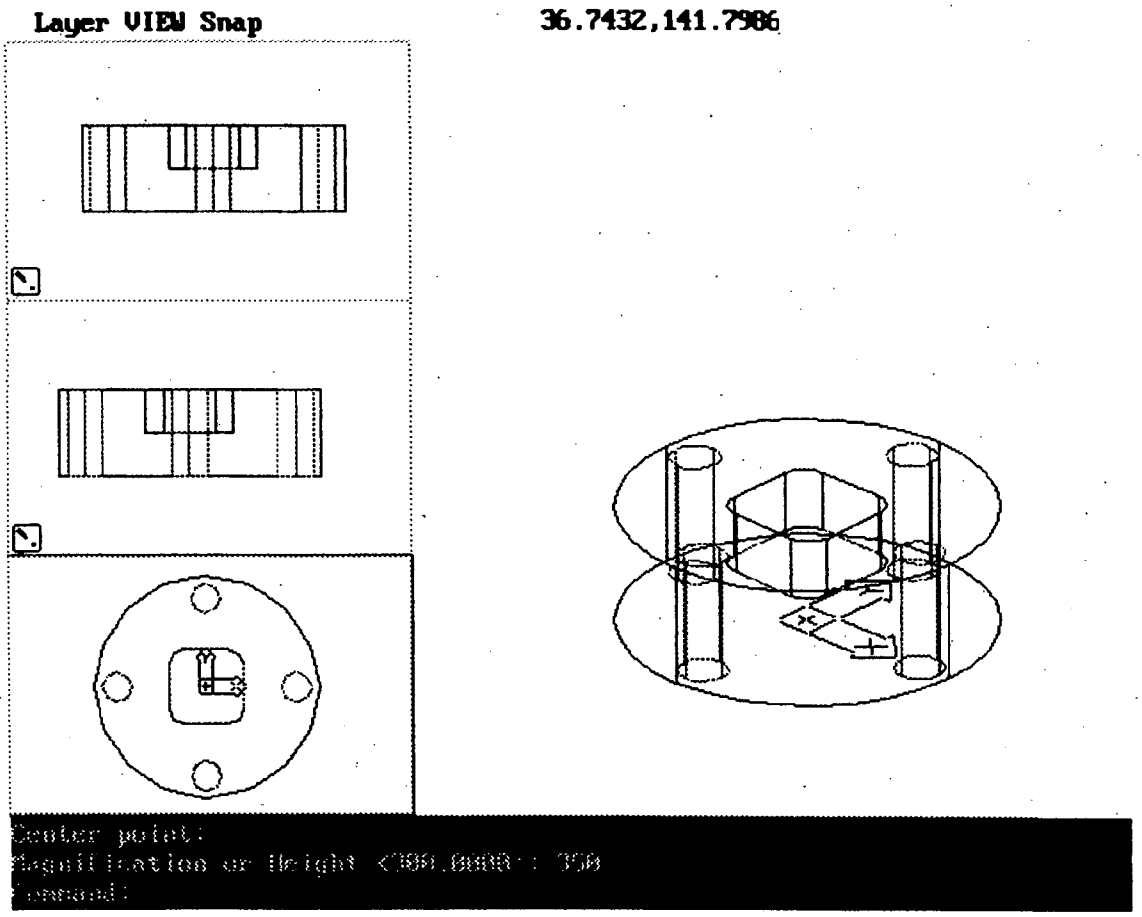


fig. A15 - Ambiente de Trabalho do FM em várias Vistas

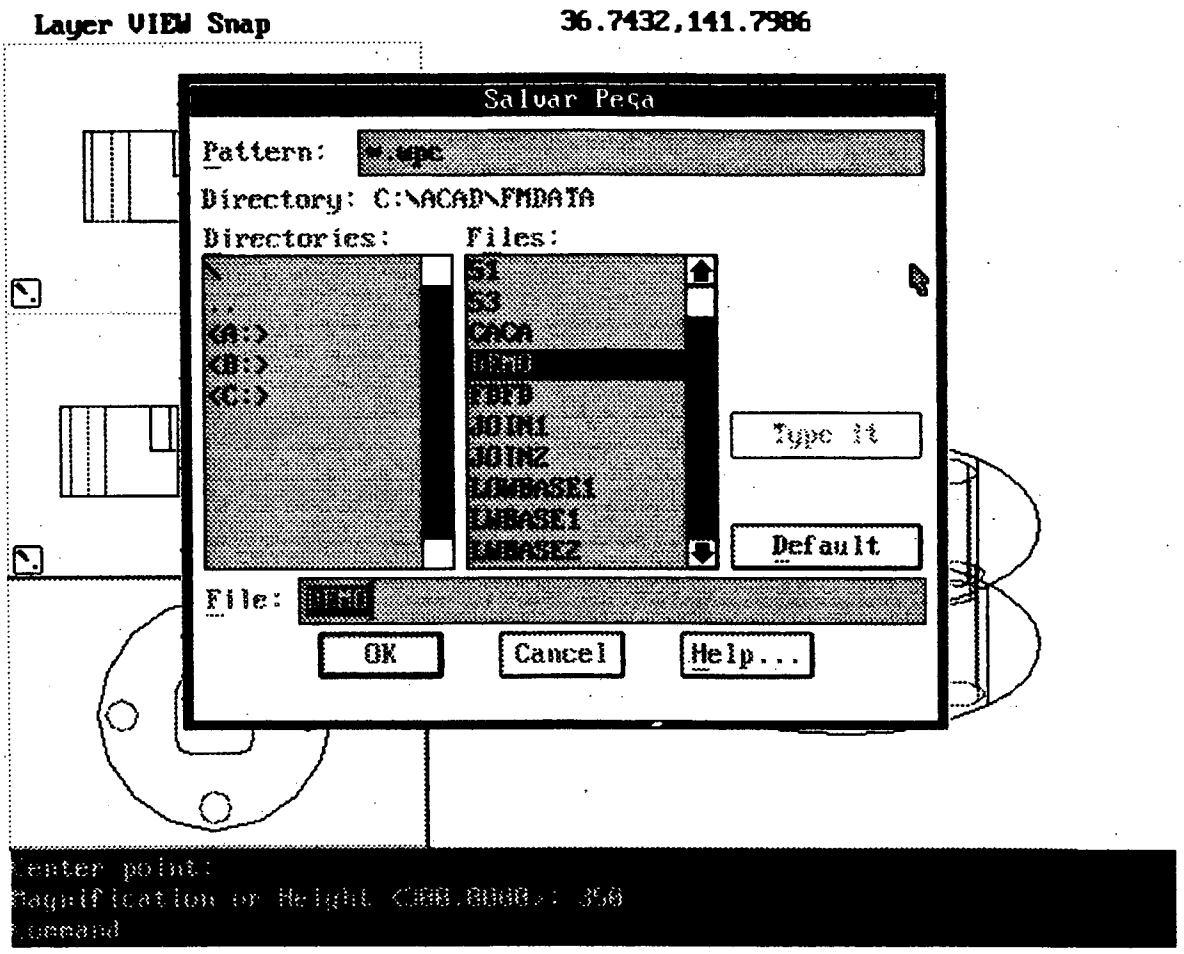


fig. A16 - Operação de Armazenamento de uma Peça

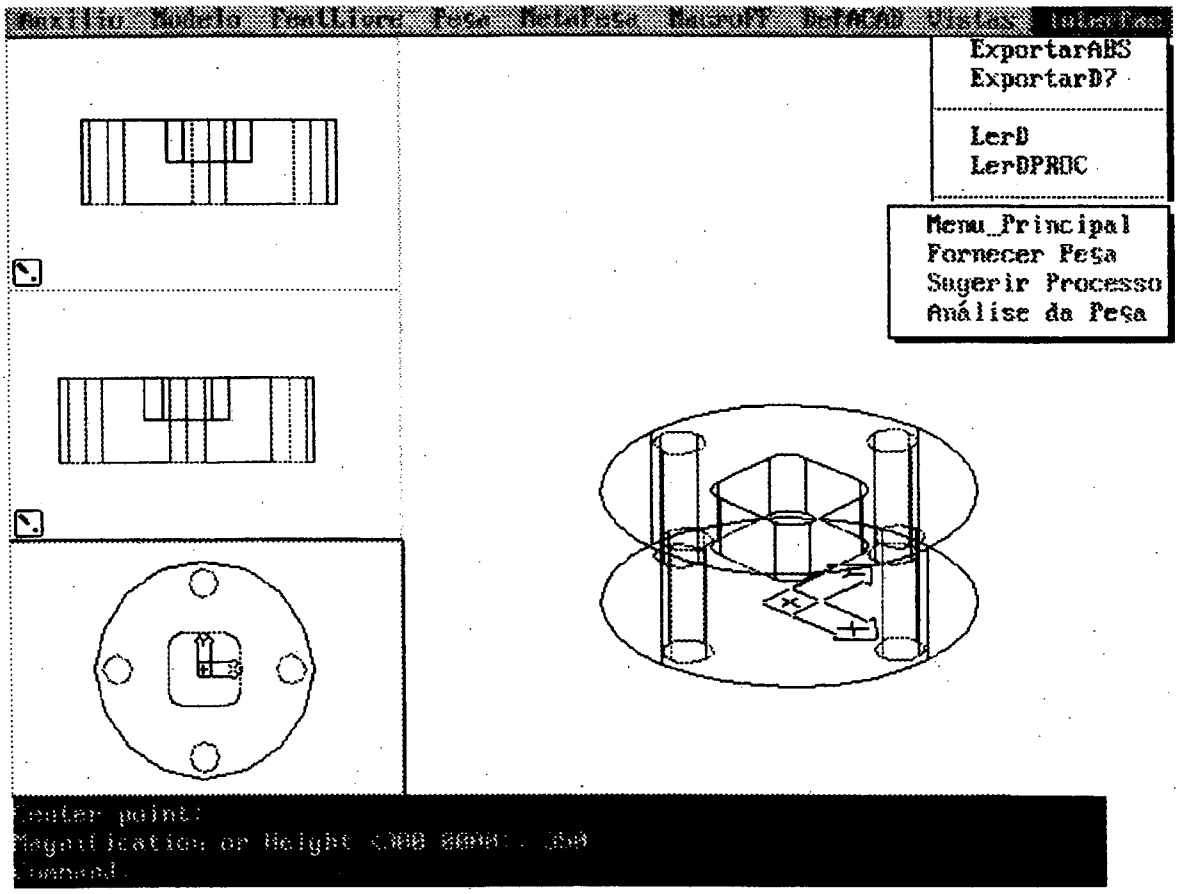


fig. A17 - Interface para comunicação com o Módulo Abstrato

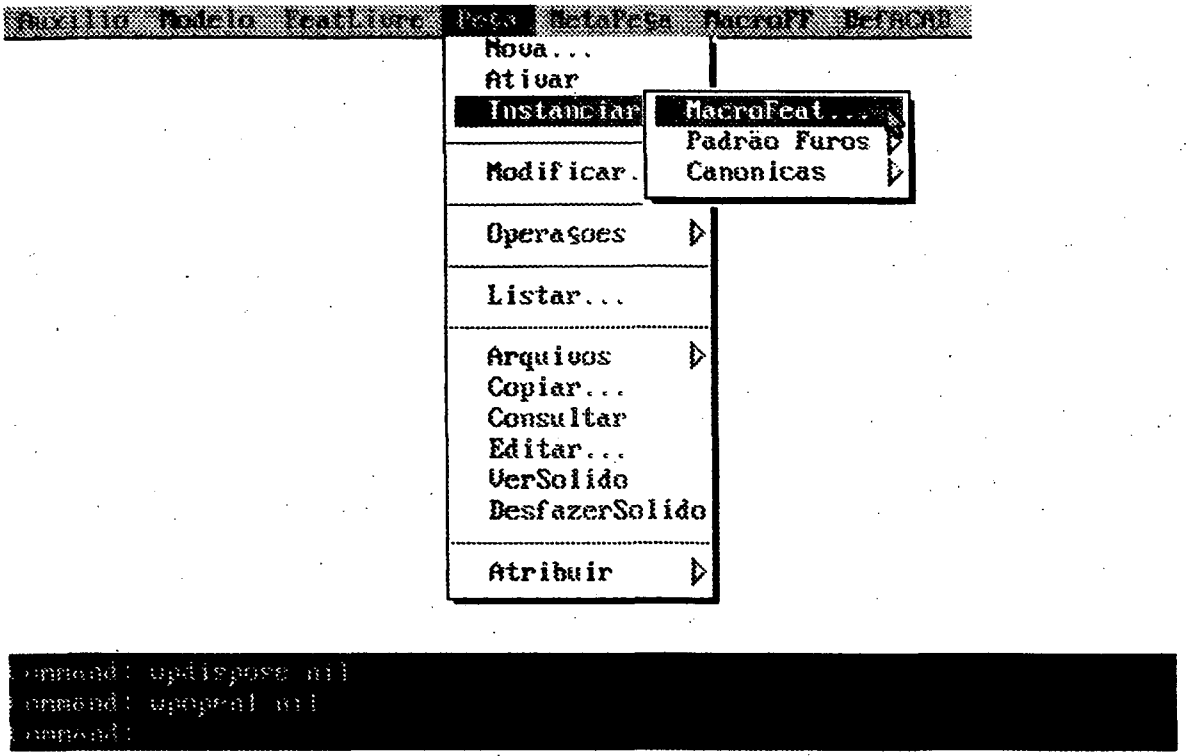


fig. A18 - Opção para instanciamento de uma Macro Feature

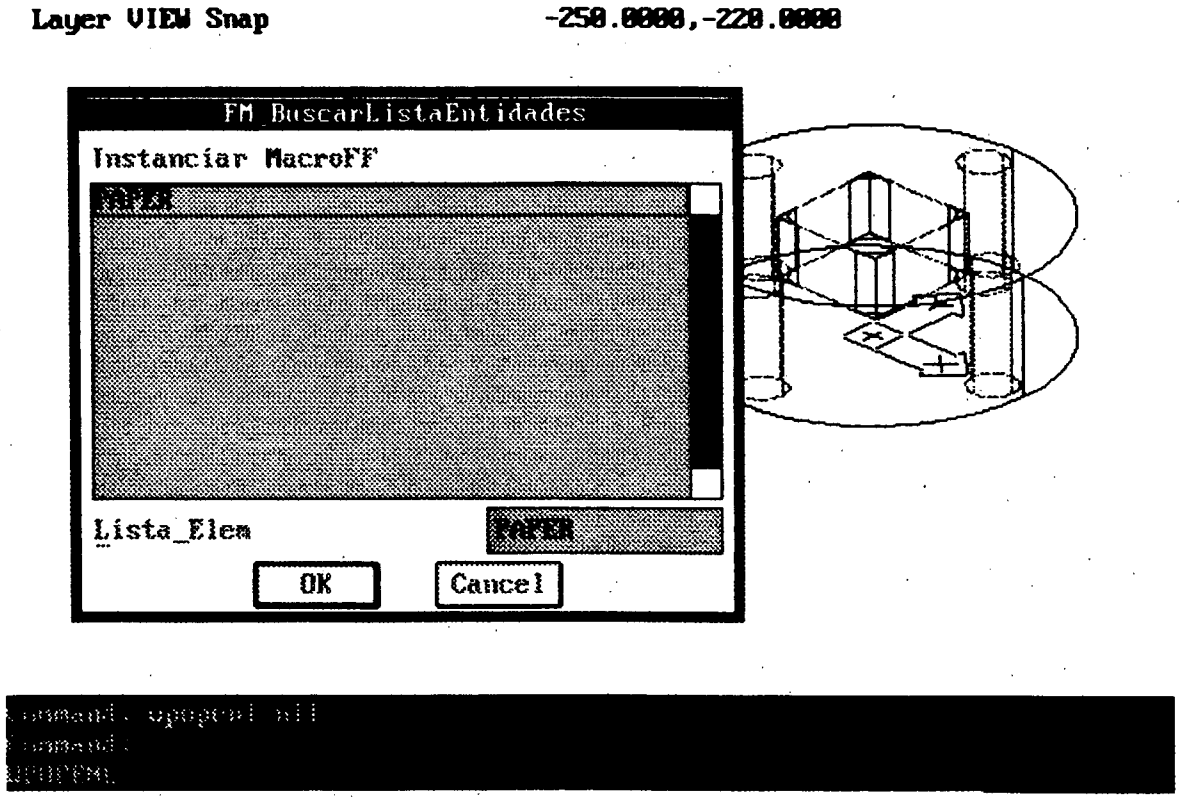


fig. A.19 - Diálogo de Instanciamento de uma Macro-Feature

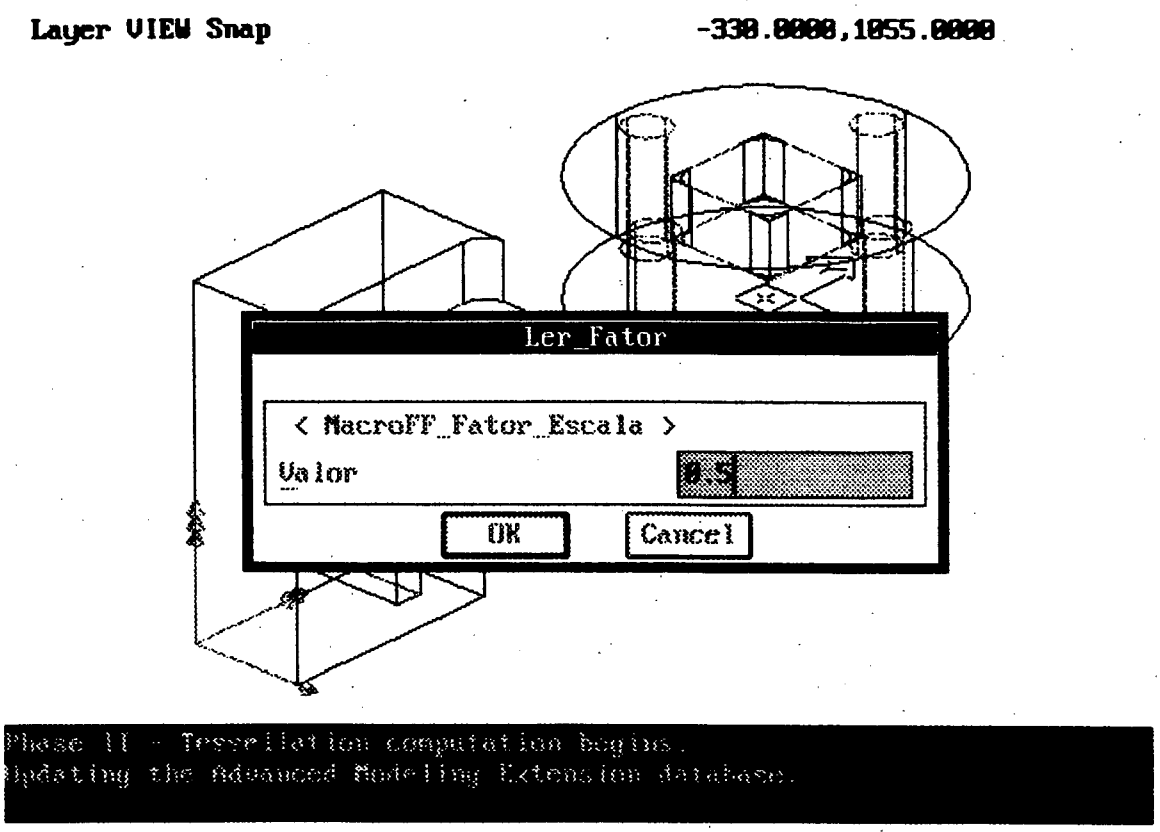


fig. A.20 - Escalonamento de uma Macro-Feature durante o instanciamento

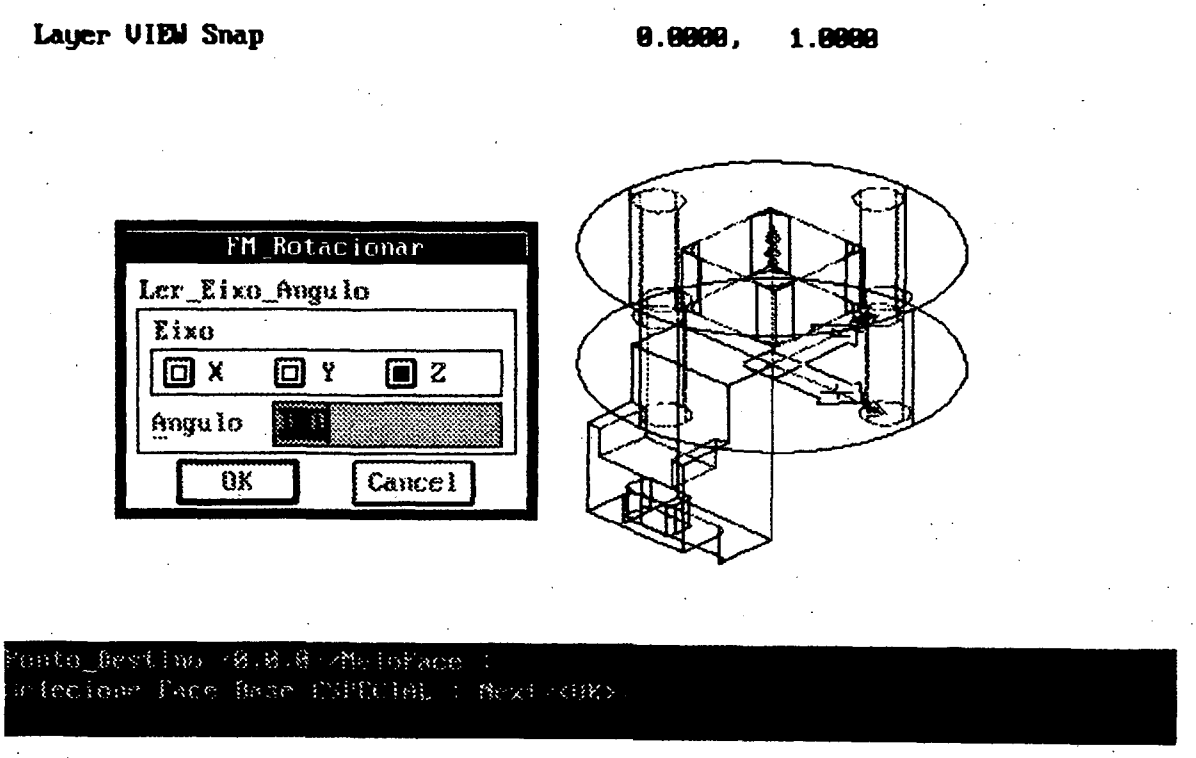


fig. A21 - Posicionamento da Macro-Feature

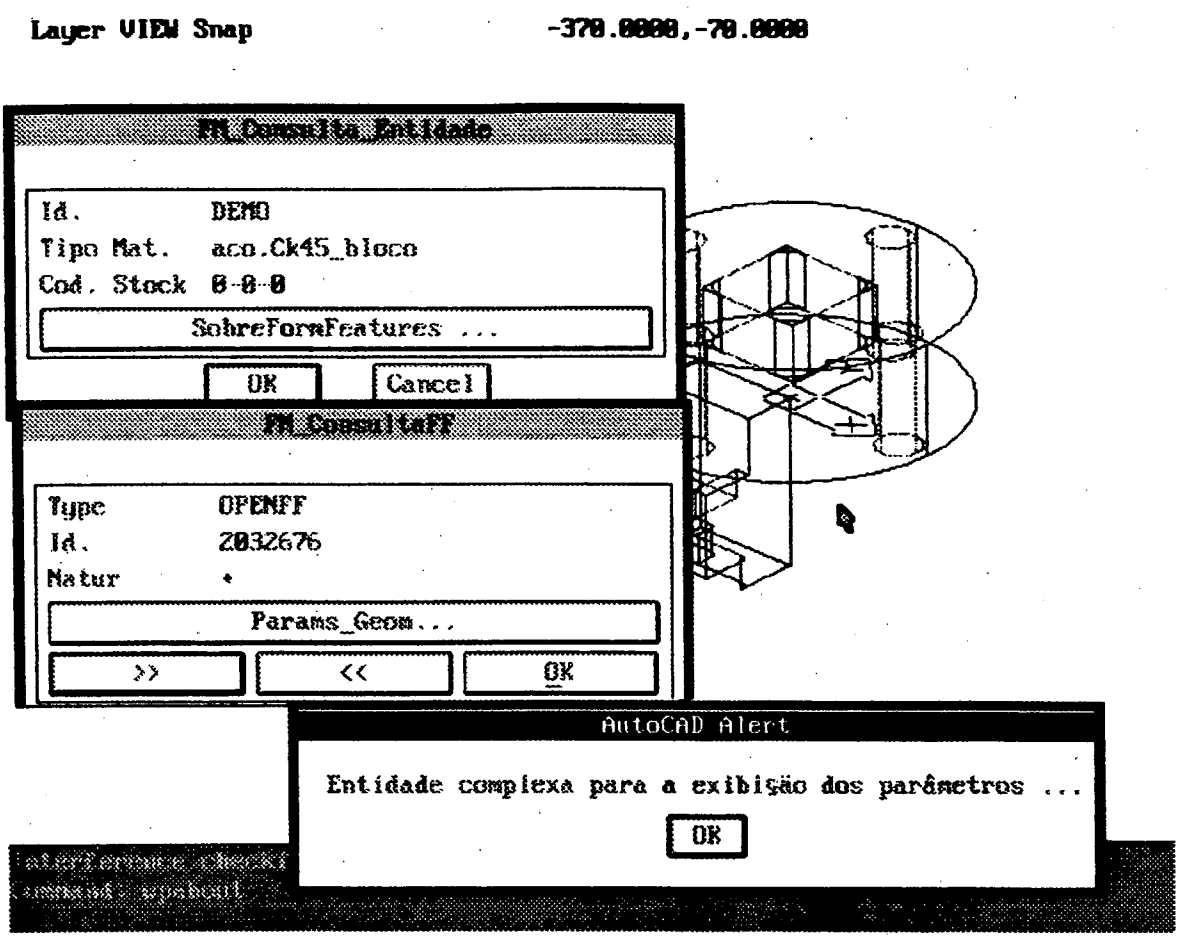


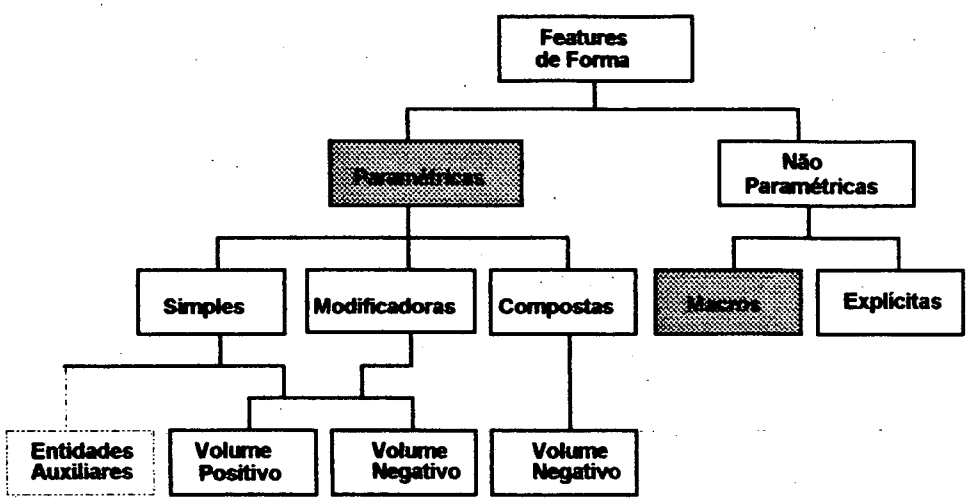
fig. A.22 - Consulta sobre uma peça

ANEXO 2

Diagrama Geral dasFeatures

Esse anexo contém um diagrama mostrando a classificação geral das features, bem como os conceitos contextuais criados dentro da aplicação do modelo, durante a fase de implementação.

A figura abaixo mostra a classificação geral das features de forma, tal como ela se apresenta no modelo conceitual do FM, o que significa dizer que essa classificação encontra-se descrita no modelo. Um conjunto virtual, chamado de ***features Implícitas***, engloba as features *Paramétricas* e as *Não-Paramétricas Macros*, uma vez que a definição de ambas se dá através de parâmetros implícitos.



Na implementação dos conceitos contemplados no modelo apresentado neste trabalho, surgiram algumas classificações contextualizadas especificamente, que são as seguintes :

Features Canônicas e Abertas

Denominações usadas para identificar a origem das features, isto é, se uma feature faz parte do conjunto das features oferecidas pelo FM (*feature canônica*), ou se faz parte de uma biblioteca (**aberta**, no sentido de não oferecer restrições maiores quanto a definição das features nela contidas) de features construída pelo próprio usuário (*feature aberta ou Macro-Feature*).

Feature-mãe e feature-filhas

Denominações aplicadas as features envolvidas em uma relação de dependência, onde *feature-filha* é sempre uma feature negativa que depende existencialmente de 1 ou mais features positivas (*features-mãe*),

Feature-folha e feature-de-referência

Denominações aplicadas as features pertencentes a uma peça, onde *feature de referência* é qualquer feature que possua uma ou mais features posicionadas em relação a ela. *Feature-folha* é qualquer feature que ***NÃO*** possua features posicionadas em relação a ela. Em uma peça podem existir ***n*** features-folha e ***m*** features-de-referência.

As operações de modificação e remoção não são aplicáveis às features-de-referência, uma vez que a propagação de uma dessas operações pode ter efeitos imprevisíveis e altamente complexos (ver figura 3.13, pag. 49).

ANEXO 3

Exemplo de um Arquivo de Interface

Esse anexo mostra um arquivo contendo informações disponibilizadas pelo FM, ao Módulo Abstrato.

peca
peca.discreta lowbase1
formfeatures "FF_1659796,FF_1681168,FF_1664212,FF_1713568"
tipo.de.material.de.construcao aco.Ck45
forma.do.material.de.construcao vergalhao.perfil.circular
volume.peca 0.035399
peso 278.239504
identificador.forma.comercial vergalhao.perfil.circular.lowbase1
dureza.Rockwell.B.peca.acabada 210
projetista Carlos.Nascimento
processista Gabriele.Vitali
d.1 0
d.2 0
d.3 0
d.4 0
d.5 0
d.6 0
d.7 0
d.8 0
dproc.1 1
dproc.2 1
dproc.3 1
dproc.4 1
dproc.5 1
dproc.6 1
dproc.7 1
dproc.8 1
dproc.9 1
dproc.10 1
dproc.11 1
dproc.12 1
@
prim.geometrica.de.volume
paralelepipedo.retang FF_1659796
tipo.de.caract.forma ext
fc.ref.pos.geom 0
x.pto.A 0.000000
y.pto.A 200.000000
z.pto.A 100.000000
x.pto.B 530.000000
y.pto.B 200.000000
z.pto.B 100.000000
x.pto.O 0.000000
y.pto.O 0.000000
z.pto.O 0.000000

l1 1.000000
m1 0.000000
n1 0.000000
l2 0.000000
m2 1.000000
n2 0.000000
l3 0.000000
m3 0.000000
n3 1.000000
tol.s.compr 0.500000
tol.i.compr 0.500000
compr 530.000000
tol.s.lado 0.500000
tol.i.lado 0.500000
lado 400.000000
tol.s.lado 0.500000
tol.i.lado 0.500000
lado.2 200.000000
Sfc.1 NO-VALUE
Sfc.2 NO-VALUE
Sfc.3 NO-VALUE
Sfc.4 NO-VALUE
Sfc.5 NO-VALUE
Sfc.6 NO-VALUE
rug.Sfc.1 NO-VALUE
rug.Sfc.2 NO-VALUE
rug.Sfc.3 NO-VALUE
rug.Sfc.4 NO-VALUE
rug.Sfc.5 NO-VALUE
rug.Sfc.6 NO-VALUE
classe.de.proc.de.fabr.1 usinagem
@
prim.geometrica.de.volume
paraleiepipedo.retang FF_1681168
tipo.de.caract.forma int
fc.ref.pos.geom 0
x.pto.A 115.000000
y.pto.A 200.000000
z.pto.A 150.000000
x.pto.B 415.000000
y.pto.B 200.000000
z.pto.B 150.000000
x.pto.O 0.000000
y.pto.O 0.000000
z.pto.O 0.000000
l1 1.000000
m1 0.000000
n1 0.000000
l2 -0.000000
m2 1.000000
n2 0.000000
l3 0.000000
m3 0.000000
n3 1.000000
tol.s.compr 0.500000
tol.i.compr 0.500000

compr 300.000000
tol.s.lado 0.500000
tol.i.lado 0.500000
lado 200.000000
tol.s.lado 0.500000
tol.i.lado 0.500000
lado.2 100.000000
Sfc.1 NO-VALUE
Sfc.2 NO-VALUE
Sfc.3 NO-VALUE
Sfc.4 NO-VALUE
Sfc.5 NO-VALUE
Sfc.6 NO-VALUE
rug.Sfc.1 NO-VALUE
rug.Sfc.2 NO-VALUE
rug.Sfc.3 NO-VALUE
rug.Sfc.4 NO-VALUE
rug.Sfc.5 NO-VALUE
rug.Sfc.6 NO-VALUE
classe.de.proc.de.fabr.1 usinagem
@
prim.geometrica.de.volume
cilindro.circular.reto FF_1664212
tipo.de.caract.forma int
fc.ref.pos.geom 0
x.pto.A 510.000000
y.pto.A 380.000000
z.pto.A 0.000000
x.pto.B 510.000000
y.pto.B 380.000000
z.pto.B 200.000000
x.pto.O 0.000000
y.pto.O 0.000000
z.pto.O 0.000000
l1 1.000000
m1 0.000000
n1 -0.000000
l2 0.000000
m2 1.000000
n2 0.000000
l3 0.000000
m3 0.000000
n3 1.000000
tol.s.r 0.500000
tol.i.r 0.500000
r 15.000000
tol.s.compr 0.500000
tol.i.compr 0.500000
compr 200.000000
Sfc.1 existente
Sfc.2 existente
Sfc.3 existente
rug.Sfc.1 NO-VALUE
rug.Sfc.2 NO-VALUE
rug.Sfc.3 NO-VALUE
classe.de.proc.de.fabr.1 usinagem

@
prim.geometrica.de.volume
cilindro.circular.reto FF_1713568
tipo.de.caract.forma int
fc.ref.pos.geom 0
x.pto.A 20.000000
y.pto.A 20.000000
z.pto.A 0.000000
x.pto.B 20.000000
y.pto.B 20.000000
z.pto.B 200.000000
x.pto.O 0.000000
y.pto.O 0.000000
z.pto.O 0.000000
l1 1.000000
m1 0.000000
n1 -0.000000
l2 0.000000
m2 1.000000
n2 0.000000
l3 0.000000
m3 0.000000
n3 1.000000
tol.s.r 0.500000
tol.i.r 0.500000
r 15.000000
tol.s.compr 0.500000
tol.i.compr 0.500000
compr 200.000000
Sfc.1 existente
Sfc.2 existente
Sfc.3 existente
rug.Sfc.1 NO-VALUE
rug.Sfc.2 NO-VALUE
rug.Sfc.3 NO-VALUE
classe.de.proc.de.fabr.1 usinagem
@
elemento.de.uniao.de.fcs
elemento.de.uniao.de.fcs U_1659796_1681168
tipo.uniao meio.continuo
peca.discreta.ref.1 lowbase1
fc.ref.1 FF_1659796
Sfc.fc.ref.1 Sfc.4
peca.discreta.ref.2 lowbase1
fc.ref.2 FF_1681168
Sfc.fc.ref.2 Sfc.4
@
elemento.de.uniao.de.fcs
elemento.de.uniao.de.fcs U_1659796_1664212
tipo.uniao meio.continuo
peca.discreta.ref.1 lowbase1
fc.ref.1 FF_1659796
Sfc.fc.ref.1 Sfc.4
peca.discreta.ref.2 lowbase1
fc.ref.2 FF_1664212
Sfc.fc.ref.2 Sfc.3

@
elemento.de.uniao.de.fcs
elemento.de.uniao.de.fcs U_1659796_1713568
tipo.uniao meio.continuo
peca.discreta.ref.1 lowbase1
fc.ref.1 FF_1659796
Sfc.fc.ref.1 Sfc.4
peca.discreta.ref.2 lowbase1
fc.ref.2 FF_1713568
Sfc.fc.ref.2 Sfc.3
@