NILSON LUIZ MAZIERO

UM SISTEMA COMPUTACIONAL INTELIGENTE DE SUPORTE AO PROJETO, MANUFATURA E MONTAGEM DE PEÇAS BASEADO EM FEATURES: UMA ABORDAGEM COM SISTEMAS ESPECIALISTAS

Tese apresentada como requisito
parcial à obtenção do grau de Doutor.
Curso de Pós-Graduação em Engenharia
Mecânica, Área de Fabricação
Universidade Federal de Santa Catarina
Orientador: Prof. Ph. D. João C. E. Ferreira

Florianópolis, Junho de 1998

NILSON LUIZ MAZIERO

UM SISTEMA COMPUTACIONAL INTELIGENTE DE SUPORTE AO PROJETO, MANUFATURA E MONTAGEM DE PEÇAS BASEADO EM *FEATURES*: UMA ABORDAGEM COM SISTEMAS ESPECIALISTAS.

Dissertação aprovada como requisito parcial para obtenção do grau de Doutor no Curso de Pós-Graduação em Engenharia Mecânica, Área de Fabricação, da Universidade Federal de Santa Catarina.

João Carlos Espíndola Ferreira

Prof. João Carlos Espíndola Ferreira, Ph.D. - Orientador

Júlio César Passos

Prof. Julio Cesar Passos, Dr. - Coordenador do Curso

BANCA EXAMINADORA:

João Carlos Espíndola Ferreira

Prof. João Carlos Espíndola Ferreira, Ph.D. - Presidente

Prof. Dr. Antonio Batocchio

Prof. Dr. Édison da Rosa

Prof. Dr. Gilberto Dias da Cunha

Prof. Nelson Back, Ph. D.

FLORIANÓPOLIS, 16 DE JUNHO DE 1998.

A Vilma, minha companheira
que tanto incentivou essa longa jornada.

A meus filhos, Maurízio e Fabrízio,
que, em algum dia, ao lerem esse trabalho,
possam compreender a minha ausência nesse período.

A meus pais, Orvalino (in memoriam) e Clarice,
que me ensinaram a importancia do saber.

AGRADECIMENTOS

No momento de escrever a página de agradecimentos, as lembraças dessa jornada vêm à tona. É o momento de agradecer a todos que, mesmo por um curto período, de alguma forma contribuíram para que eu pudesse continuar em frente.

Em especial, tenho a agradecer à minha família, à Vilma, minha esposa e companheira, que por esse longo tempo segurou a "barra" e, mesmo distante, sempre dispensou o maior apoio ao meu trabalho. Aos meus filhos, Maurízio e Fabrízio, que, a partir de agora, passam a ter um pai presente. À minha mãe, Clarice; minhas irmãs Nelci e Neuzete, que sempre me incentivaram.

Não poderia esquecer do Norton, o homem das *features*, que, com o seu trabalho, deu o passo inicial no departamento de fabricação para as pesquisas nessa linha.

Como esquecer do companheiro dos primeiros passos, o Adilson U. Butzke, que, com sua camaradagem e capacidade, sempre encontrava uma saída para os problemas. O agradecimento àquele que deu o passo inicial que permitiu que, com a experiência adquirida, esse trabalho fosse desenvolvido nesse estágio.

E o amigo de tantos finais de semana de trabalho (e de cerveja), Darcio de Freitas Rezende, que, com sua capacidade e perseverança, desenvolveu ferramentas de trabalho que me foram muito úteis. Obrigado pelas sugestões para o desenvolvimento do software e pela amizade que compartilhei por tanto tempo.

Ao prof. João Carlos E. Ferreira, que depositou tanta confiança em mim e em meu trabalho; que, com sua boa vontade e paciência, sempre soube conduzir as atividades de forma a permitir que houvesse liberdade nas decisões a serem tomadas; que, com a sua disposição, sempre esteve a postos nos momentos em que eu necessitava da sua colaboração.

Aos amigos Ricardo L. D. Cunha e Claudio R. Grando, pela colaboração na área computacional.

Ao Kleber Bianchi, colega das primeiras horas de curso e de moradia, que tanto me incentivou e com quem compartilhei muitas idéias. Ao Tetsuo Koike e Wagner Corrêa, pela amizade e colaboração.

Aos colegas do Grima, ao Fernando Furlan Neto, que contribuiu na programação do CAD; ao Fernando Santana Pacheco, com o sistema especialista; ao Everton Gubbert, no banco de dados; e ao Marcelo Prim, com o CAD. Todos meus companheiros e que foram meus braços nessa tarefa.

À Universidade de Passo Fundo que, através do seu programa de capacitação docente, permitiu que me ausentasse para o meu aperfeiçoamento.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

CAD Projeto Auxiliado por Computador (Computer-Aided Design)

CAE (Computer-Aided Engineering)

CAM Manufatura Auxiliada por Computador (Computer-Aided Manufacturing)

CAPP Planejamento do Proceso Auxiliado por Computador (Computer-Aided Process Planning)

CIM Manufatura Integrada por Computador (Computer Integrated Manufacturing)

CSG Geometria Contrutiva de Sólidos (Construtive Solid Geometry)

DFA Projeto para Montagem (Design for Assembly)

DFM Projeto para Manufatura (Design for Manufacturing)

NC Comando Numérico (Numeric Control)

STEP Standard for the Exchange of Product Model Data

RESUMO

O objetivo deste trabalho é apresentar uma proposta de um sistema CAD inteligente para a modelagem de informações, visando integrar o projeto, manufatura e montagem e permitindo a adaptação à cultura do ambiente a ser aplicado. A solução proposta aplica-se a produtos compostos por peças rotacionais. O modelo prevê a modelagem das informações baseada na existência de um conjunto, subconjuntos e peças, utilizando-se, para isso, da tecnologia de *features*, juntamente com a análise orientada para objetos.

As *features* são modeladas em termos de *features* básicas e modificadoras. A criação de uma peça é feita a partir de *features* definidas numa biblioteca que está à disposição do usuário, as quais são possíveis combinar em novas *features*, o que pode ser feito diretamente através da programação, ou através da interface gráfica do sistema. Tanto a inserção direta das *features* da biblioteca como a criação de *features* combinadas pela interface gráfica, ou via programação interna do sistema, são validadas a cada operação pela utilização de um sistema especialista que consulta a base de conhecimento com relação a essa atividade, emitindo um parecer quanto à veracidade do fato.

Essa base de conhecimento é modelada em função das combinações permitidas pelas *features* da biblioteca. Com as informações modeladas do produto, faz-se a análise do conjunto para a definição automática de como as peças estão montadas entre si, resultando uma descrição dessa montagem. Isso é feito através da utilização de algoritmos e do uso do sistema especialista, que consulta a base de conhecimento de montagem para verificar as condições em que as peças podem ser definidas como montadas.

Com a descrição da montagem, é possível identificar as dimensões a serem toleradas, bem como as superfícies que devem receber um acabamento superficial adequado. As tolerâncias a serem anexadas às dimensões são obtidas a partir de um banco de dados no qual são descritas em função da aplicação e da precisão. Além disso, com base nas informações que descrevem o conjunto montado, utilizando-se o dimensionamento funcional, pode-se obter a distribuição das dimensões nas peças em função da montagem, as quais vão complementar as informações da peça para a fabricação, inspeção e montagem.

Para verificar a validade do modelo descrito, foi implementado um protótipo de software para microcomputadores da classe PC com sistema operacional "Windows"; para armazenar as informações do modelo, foi construída uma estrutura de dados independente do CAD utilizado, desenvolvida em linguagem C++. A comunicação entre a estrutura de dados e o sistema especialista é feita no momento em que a tarefa é realizada, o que permite a verificação imediata da operação. A modelagem de um conjunto é apresentada como exemplo para a validação do modelo e a visualização do protótipo implementado. Finalmente, discutem-se os resultados e apresentam-se propostas de novos trabalhos que visam complementar e melhorar o modelo proposto.

ABSTRACT

The aim of this work is to propose an intelligent CAD system to model information, aiming at integrating design, manufacture and assembly, and allowing its adaptation to the culture of the environment where it will be applied. The proposed solution is applied to products composed of rotational parts. Its architecture considers that information is modeled based on the existence of an assembly, subassemblies and parts. The technology of features is applied in this system, together with object oriented programming.

Features are modeled in terms of "basic features" and "feature modifiers". A part is created through features that are defined in a library which is available to the user, and these features can be combined into new features, either via internal programming or through the graphical interface of the system. The introduction of features into the system is verified after each operation by an expert system that accesses a knowledge base that contains information about this activity, and the system notifies the user whether the feature meets the requirements.

This knowledge base is modeled as a function of the allowed combinations of the features within the library. With the modeled information about the product, an analysis of the assembly is performed, in order to determine automatically how the parts are assembled, resulting in a description of this assembly. This is done through the utilization of algorithms, and via an expert system, that checks in the assembly knowledge base the conditions in which the parts can be considered as being assembled.

With the description of the assembly, it is possible to identify the dimensions that need tighter tolerances, as well as the surfaces that must receive an adequate surface finish. The tolerances to be associated with the dimensions are obtained from a database in which the tolerances are related to the application and precision. Also, based on the information that describe the assembled product, resulting from functional dimensioning, the distribution of dimensions in the assembled parts can be obtained, which will complement the part's information for manufacturing, inspection and assembly.

In order to verify the validity of the described model, a prototype software for PC-compatible microcomputers was implemented; in order to store the model's information, it was built a data structure independent of the commercial CAD software utilized, developed in C++. The communication between the data structure and the expert system is done at the moment the task is performed, which permits the immediate verification of the operation. The modeling of an assembly is presented as an example for the validation of the model. Finally, results are discussed and future work is presented, aiming at improving the proposed model.

CAPÍTULO 1

INTRODUÇÃO

A produção em massa é o diferencial na competição econômica, permitindo que se obtenham produtos melhores, em maior quantidade e com menor custo. Nas últimas décadas, com o advento de novas tecnologias que foram integradas aos sistemas de manufatura, somente a produção em massa não é suficiente; ainda que seja necessário produzir produtos em quantidade, o mercado exige novas opções de produtos e que esses possam ser diferenciados de modo a satisfazer o consumidor. Dessa forma, o ciclo de vida de um produto, que chegava a ser de décadas em alguns casos, diminuiu drasticamente para meses, sendo necessária a criação de novos produtos e numa velocidade maior, o que influencia todo o ciclo produtivo.

Com a competição pelos mercados de consumo, esse novo diferencial passou a pesar, de forma que a agilidade de cada um em oferecer produtos mais variados e mais personalizados ao cliente, bem como preços competitivos, é vital. Para atingir essas metas, é necessário velocidade e organização do sistema produtivo, o que obriga ao desenvolvimento de novas filosofias, metodologias e ferramentas a serem utilizadas, isso de modo contínuo.

1.1 - O PROJETO COMO ELEMENTO DISSEMINADOR DE INFORMAÇÕES

Para Dong (1993), o desenvolvimento de produtos mecânicos, tradicionalmente, consiste em uma série de projetos e processos de manufatura. Esses processos incluem projeto de concepção, projeto de detalhes, análise de projetos, execução de protótipos e testes, planejamento da produção, usinagem, inspeção e montagem. Segundo o autor, ainda, um projeto mecânico constitui-se de duas fases: o projeto de concepção e o projeto de detalhes. O projeto de concepção é relativo à identificação de mecanismos apropriados para realizar uma função de projeto desejada que preencha os requisitos de funcionalidade, o que vai determinar a configuração básica do produto. O projeto detalhado focaliza a especificação das dimensões das formas e as variações permitidas, incluindo tolerâncias e acabamento das superfícies.

O projeto detalhado, por sua vez, é mais orientado à manufatura, isto é, ele visa a uma solução, mantendo a funcionalidade e procurando facilitar os processos de fabricação. As dimensões e tolerâncias das formas são especificadas para facilitar a produção, a montagem, assim como para diminuir os custos de produção, satisfazendo as restrições de projeto.

Considera-se que o desenvolvimento do projeto de detalhes é a fase que corresponde à concretização do projeto conceitual; é aquela em que o projetista efetua a representação das informações de modo gráfico e literal dos elementos a serem obtidos através da manufatura, o que resulta no produto acabado.

O início da fase do projeto detalhado começa com a representação gráfica do projeto, que pode ser efetuada através de desenhos convencionais realizados numa prancheta ou com a utilização de sistemas CAD ("Computer-Aided Design"). Esses desenhos representam o conjunto que compõe o produto, no qual o objetivo é obter uma forma funcional que está representada graficamente e com dimensões nominais. Para chegar a essa forma, o projetista utiliza-se da sua experiência, de conhecimentos de manuais especializados, de informações de fornecedores, de normas técnicas, etc.

A partir da representação do conjunto, o projetista passa a analisar com maior rigor as dimensões e a introduzir tolerâncias e outros elementos informativos, sempre visando transmitir as informações da funcionalidade do produto, as quais deverão ser respeitadas quando da sua manufatura. O passo seguinte corresponde à execução dos desenhos de detalhes das peças individualmente, quando são explicitadas as informações a respeito do conjunto em termos individuais, isto é, mantendo as características necessárias para que funcionem no conjunto.

Considerado o projeto pronto, essas informações através dos desenhos são remetidas a todos os departamentos envolvidos, para que emitam um parecer a respeito de sua manufaturabilidade. Os departamentos deveriam analisar os respectivos projetos devolvendo-os com as respectivas observações; porém, muitas vezes isso não ocorre, pois é feita uma análise rápida, apenas para cumprimento de uma formalidade, de tal forma que o projeto não é visto como um trabalho de extrema importância. Ao retornarem, os projetos são modificados de acordo com as observações, e aqueles que não apresentam qualquer necessidade de alteração são considerados aptos e liberados para a produção.

Quando da implementação da produção, os departamentos são obrigados a analisar os projetos, e é aí que se descobrem falhas, como tolerâncias impossíveis de serem obtidas ou falta de agilidade do processo para atender à demanda do produto. Nesse ponto, a identificação dos problemas ocorre quando várias fases do projeto já estão em execução, por isso as modificações agora necessárias podem afetar de modo significativo todo o produto. Nesse sentido, Huthwaite (1992) mostra, através de um gráfico, o efeito dos custos em cada fase em que possam ocorrer modificações (Figura 1.1). Por meio dele, pode-se constatar que, em cada fase, multiplica-se por dez o efeito do custo de alteração do produto.

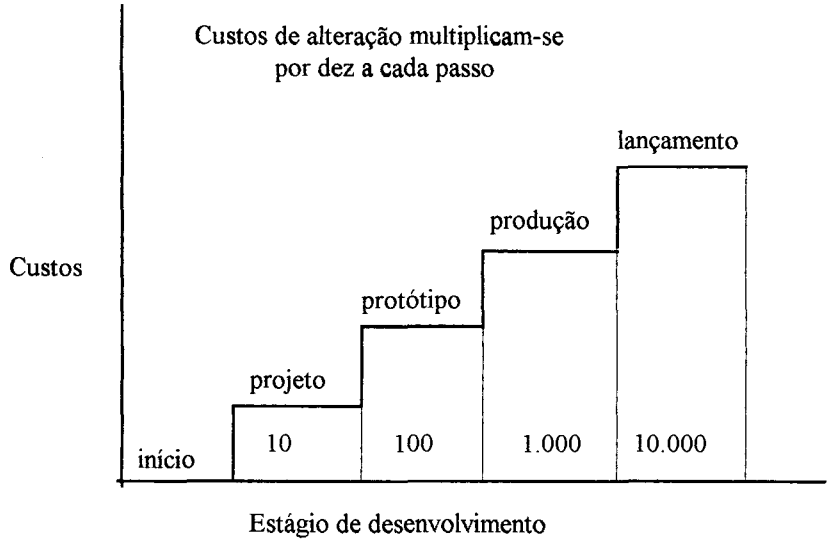


Figura 1.1 - Efeito escada (Huthwaite, 1992).

As informações do projeto, através dos desenhos, seguem para diversos departamentos, como métodos e processos, inspeção e controle, montagem, etc. Há departamentos para quem interessam apenas algumas informações constantes nos desenhos ou informações resultantes de outros departamentos, como produção, compras, etc. Em cada um desses, os projetos são analisados dentro de sua ótica. Assim, ao método e processos interessa o desenho completo da peça para que possa determinar as seqüências de operações, máquinas e ferramentas envolvidas, dispositivos de fixação, etc.

Já, para inspeção e controle, é interessante verificar no desenho o tipo da peça, o que deve ser controlado, o que é verificado através das dimensões, tolerâncias e notas. Na montagem, por sua vez, é de interesse o desenho de conjunto, o qual informa a disposição das peças no produto a partir do qual se pode gerar o plano de montagem e a escolha de ferramentas e dispositivos para as operações. Ao departamento de compras interessam algumas informações constantes no desenho, como tipo de matéria-prima, dimensões e quantidade, componentes de terceiros, etc., o que não envolve o desenho diretamente. Ainda, para o planejamento da produção, interessam os dados do planejamento do processo para que possa, então, organizar a produção, planejar como realizar o transporte de matéria-prima, alocar máquinas, ferramentas, etc. O projeto, assim, é o elemento disseminador de informações na medida em que as transmite para os diversos departamentos.

1.2 - A INFORMÁTICA COMO ELEMENTO DE SUPORTE À INTEGRAÇÃO DO PROJETO E MANUFATURA

Com o advento da informática, surgiram oportunidades de aplicação dessa ferramenta no meio produtivo com o objetivo de auxiliar, simplificar e agilizar as tarefas desde a fase de projeto até a entrega do produto ao cliente, aumentando, desse modo, o desempenho dos sistemas de manufatura. Para isso, desenvolveram-se várias filosofias que visam à implementação de metodologias que auxiliam essa implantação, o que tem sido um tanto difícil. Se inúmeras dificuldades têm surgido nesse caminho, também inúmeras soluções têm sido apresentadas para sanar tais problemas. Na realidade, qual é o problema que se deseja resolver?

Como se sabe, existem meios de representar as informações de manufatura, quais sejam: (i) o tradicional desenho técnico, este realizado tanto manualmente como com a utilização de computadores; (ii) através do modelamento sólido, metodologia relativamente nova que tem sido aplicada para a representação de objetos em sistemas computacionais.

Considera-se, porém, ser fundamental para uma indústria de manufatura um sistema que não somente automatize a tarefa de representar informações através de desenhos ou outra metodologia de representação, mas auxilie o projetista na tomada de decisões e de modo integrado ao processo produtivo. É também importante que tais informações possam ser utilizadas de modo mais racional, em função de outras tecnologias já disponíveis, como os sistemas CAE e CAM.

Nos últimos anos, inúmeras filosofias têm sido desenvolvidas visando à integração dos sistemas de manufatura, buscando obter uma integração adequada e que dê suporte às tarefas de projeto e manufatura. Uma delas é através do CIM ("Computer Integrated Manufacturing"), o qual visa integrar as siglas já conhecidas CAE/CAD/CAM, entre outras, hoje já transformadas em CAE/CAD/CAPP/CAM, associadas à manufatura, como também às atividades administrativas e organizacionais.

Na visão de uma integração adequada, os dados gerados no projeto devem ser utilizados em todo o ciclo de produção do produto, ou seja, através de uma interpretação adequada, cada setor do processo produtivo retira as informações necessárias geradas no projeto e as analisa sob a sua ótica. Conceitualmente, essas idéias são ótimas, mas, na prática, têm-se mostrado de difícil implementação, pois os sistemas de manufatura são muito complexos, e a representação das informações é uma tarefa árdua.

Os problemas iniciam com a representação das informações num sistema computacional, ou seja, tradicionalmente, utilizam-se os desenhos técnicos para transmitir as diversas informações, nas quais a interpretação fica a cargo do indivíduo, que a faz com base no seu conhecimento e experiência. Com a utilização de sistemas computacionais, o problema passa a ser a representação dessa metodologia conhecida numa outra forma de registro de informações,

ou seja, não mais no papel, mas através de *bytes*, de forma que o computador possa registrar e reaver as informações e, ao mesmo tempo, o usuário possa interpretá-las adequadamente. Tal representação deve corresponder a um nível mais elevado, de forma que o próprio sistema seja capaz de registrar e interpretar determinadas informações sem a intervenção humana.

Da mesma forma que as informações são organizadas para o entendimento humano, os computadores precisam de uma forma de organização para elas, para que possam interpretá-las dentro da sua linguagem. Em outras palavras, o conhecimento humano deve ser traduzido para que a máquina possa armazená-lo e manipulá-lo, fazendo a interpretação das informações sob a ótica de um ser humano. Para que isso possa ser realizado, é necessário que o indivíduo que produz esses mecanismos seja capaz de enxergar o mundo como se fosse a máquina, ou seja, conhecer a sua forma de comunicação interna e externa e a forma como o conhecimento e as informações devem ser armazenados, de modo que a máquina possa manipulá-las do modo desejado.

Como exemplo de comunicação interna, tem-se a programação que manipula dados, mas que não interessa diretamente ao usuário; como comunicação externa, têm-se as entradas e saídas de informações, ou seja, o que o usuário deve ver para compreender o que está ocorrendo. Portanto, o indivíduo deve buscar parâmetros que traduzam essas informações de modo satisfatório, para que sejam facilmente convertidas para uma linguagem de programação.

Após a escolha do modelo de representação das informações (forma conceitual de como as informações estão relacionadas), é de fundamental importância a definição da estrutura de dados (forma de representar as informações do modelo num sistema computacional) a ser utilizada, pois é através dela que as informações estarão disponíveis ao usuário. Assim, de posse de uma determinada estrutura de dados, é possível saber onde uma determinada informação está localizada e seu relacionamento com outras. A estrutura de dados permite, então, que as informações sejam armazenadas e manipuladas de modo organizado, de tal forma que tanto o computador como o ser humano as compreendam.

1.3 - O PROBLEMA A SER RESOLVIDO

Tendo em vista o que foi exposto no item anterior, há a necessidade de buscar uma representação computacional para as informações já conhecidas, isto é, uma maneira de capturar, armazenar e disponibilizar informações ao usuário de um modo natural, sem que ele tenha de ser um especialista em computação. Além disso, deve-se também disponibilizar para o usuário aquelas informações de que ele necessita em momento determinado e de modo consistente.

A aplicação da informática no projeto e na manufatura tem sido feito de forma isolada, raramente ocorrendo uma integração adequada das informações, ou seja, as informações de projeto geralmente estão representadas de uma forma particular, sendo necessário transpô-las num formato adequado para a manufatura para que possam ser utilizadas nos respectivos sistemas, trabalho esse muitas vezes executado manualmente pelo usuário. Como simples exemplo, tem-se que os desenhos muitas vezes gerados no projeto não possibilitam a sua reutilização para os processos de manufatura (CAM), sendo necessário refazê-los, pois da forma como foram executados não podem ser interpretados por um sistema CAM. Também em desenhos 2D, na execução do projeto, as linhas não foram executadas no sentido que o sistema CAM interpreta, ou as linhas não possuem ligação entre elas, o que resulta num desenho interrompido e que dificulta interpretações computacionais posteriores.

Como problema mais complexo, há a falta de representação de informações tecnológicas que possam ser extraídas das representações gráficas atualmente utilizadas. Ainda, as informações geradas nos sistemas de CAD, para aplicação em indústrias médias e pequenas, não oferecem mecanismos que possibilitem uma integração adequada para o projeto e manufatura, servindo apenas como uma prancheta eletrônica.

1.4 - OBJETIVOS DO TRABALHO

O objetivo deste trabalho é obter um modelo e um protótipo de sistema computacional que permita a integração do projeto e manufatura, criando mecanismos para a transmissão de informações do projeto para as várias áreas da manufatura.

Assim, este trabalho abrange a representação não somente de uma peça, mas de um grupo de peças de modo que possam ser individualizadas e analisadas num sistema CAPP, ou noutra atividade da manufatura.

Ainda coloca-se como objetivo a utilização de sistemas de CAD comerciais, o que permite que a interface gráfica seja utilizada e anexada a uma estrutura de dados independente para armazenar também informações tecnológicas, já que os sistemas de CAD atuais são deficientes nesse item.

A aplicação tem por objetivo geral atender ao projeto, manufatura e montagem em nível de informação de modo integrado, sendo que a manufatura engloba todas as atividades relativas à obtenção dos componentes.

Como aplicação específica, busca-se gerar uma interface entre o projeto detalhado e a manufatura e montagem, isso através de um conjunto de informações que possam ser mapeadas para essas aplicações.

1.5 - DESENVOLVIMENTO DO TRABALHO

No capítulo 2, faz-se a revisão bibliográfica a respeito dos diversos conteúdos envolvidos, mostrando o estágio em que se encontram outros trabalhos de conhecimento público. No capítulo 3, apresenta-se sistematização das informações para o modelo do sistema proposto, mostrando a sua base conceitual; no 4, a forma como parte do modelo foi implementado, bem como as adaptações necessárias ao ambiente computacional. No capítulo 5, dão-se exemplos de aplicação do sistema implementado, com conjuntos de peças e análises possíveis de serem realizadas com as informações do sistema. No capítulo 6, encontram-se as conclusões extraídas dessa implementação, como também sugestões com relação ao desenvolvimento de futuros trabalhos tomando como base o que é aqui apresentado; finalmente, relacionam-se as referências bibliográficas, bem como a bibliografia geral que serviu de apoio ao trabalho.

CAPÍTULO 2

REVISÃO BIBLIOGRÁFICA

2.1 - MODELO

Um modelo busca representar de algum modo a realidade através de informações. Para Kjellberg (1992), o conteúdo da informação é construído de conceitos e relações com outros conceitos. O conceito é definido como um elemento do pensamento, é uma construção mental dos objetos do mundo real, os quais podem ser materiais ou imateriais. Ainda segundo o autor, quando da construção de um modelo sólido num computador, constrói-se uma estrutura lógica de elementos, faces, arestas, vértices, etc., as quais não existem no mundo real dos objetos, ou seja, são mentalmente construídas pela representação no computador.

Assim, Kjellberg (1992) define *modelo do produto* como sendo o conjunto de modelos que é construído relacionando conceitos de engenharia com conceitos do modelo. As representações gráficas são o modelo de representação da realidade, as quais são relacionadas os conceitos de engenharia, como dimensões, tolerâncias, etc.

2.1.1 - MODELO DO PRODUTO

Para Krause et al. (1993), um modelo de produto deve conter informações que incluam dados, estrutura e algoritmos. Os algoritmos constituem a ponte entre o usuário, dados e estrutura; os dados de um modelo do produto, em geral, são determinados pela estrutura e seu conteúdo, e a estrutura é dependente da natureza do produto e das ferramentas usadas para modelar as informações, bem como para construir o esquema necessário para a base de dados. O conteúdo depende particularmente do produto.

Krause et al. (1993) subdividem o modelo do produto em submodelos ou modelos parciais, em cada um dos quais estão representadas informações a respeito de uma atividade

relacionada ao produto, como o tipo de modelagem, montagem, tolerâncias, planejamento do processo, etc. Pode-se dizer que a estrutura de dados que representa o modelo do produto vem a ser o coração do sistema; assim, uma estrutura inadequada dificulta sobremaneira a implementação de algoritmos que utilizam as informações contidas nessa estrutura.

Juri et al. (1990) consideram que, para que um modelador do produto seja capaz de dar suporte aos requisitos para planejamento do processo, deve ser capaz de modelar:

a) o componente desejado;

b) a peça bruta a ser trabalhada da qual vai resultar o componente feito e

c) um estágio intermediário da peça trabalhada, mostrando como se desenvolve o processo.

Ainda, o modelo proposto e implementado por Juri et al. (1990), para um sistema CAPP, define dois níveis de abstração diferentes: o nível de componente e o nível de *feature* de forma[1].

Suzuki (1996) salienta a importância das informações a serem agregadas ao modelo do produto, classificando tais informações em duas categorias: as principais e as secundárias. As informações principais são aquelas resultantes do processo de projeto, isto é, a descrição do produto, como desenhos de engenharia, etc.; por outro lado, as secundárias registram as várias formas de informações e conhecimentos usados na geração das informações principais, representando o *como* e o *porquê* do projeto ter sido gerado. Nele estão os requisitos de projeto, normas, métodos, etc. Esse autor propõe ainda que não somente as informações principais sejam anexadas ao modelo do produto, mas também as secundárias.

O sistema de CAD Microcadam (CADAM, 1996) já apresenta a modelagem da estrutura do produto, possuindo, inclusive, o registro histórico do desenvolvimento do produto (em nível gráfico). Essa estrutura do produto pode ser acessada em qualquer fase, além de poder construir uma nova solução a partir de uma determinada fase do projeto.

O AutoCAD Designer (1996) permite que um produto seja modelado criando uma estrutura de conjunto, subconjunto e peça, isso feito manualmente pelo usuário; também permite informações a respeito da montagem de uma peça com relação a outra, como restrições de translação e rotação de cada peça. A finalidade dessas informações é obter, através do uso do modelador de sólidos, apenas os desenhos explodidos do conjunto.

Rosa (1993), em sua proposta de desenvolvimento de um ambiente para engenharia simultânea, *New Modeller*, apresenta o modelo do produto subdividido em diversos submodelos,

[1] Detalhes característicos de um objeto de engenharia, como um eixo cilíndrico, um eixo cônico, etc.

como o modelo funcional, modelo de geometria, modelo de análise, modelo de documentação, modelo de fabricação; cada modelo é subdividido em mais quatro submodelos de acordo com o nível interno de representação do produto, como produto, sistema, subsistema e peça.

Wingard (1992) propõe um modelo de produto o qual, inicialmente, divide em duas partes: o modelo geométrico e o modelo tecnológico, e uma ligação através dos dois por uma interface. Para aplicação, o modelo é dividido em três níveis: o nível metaclasse, onde estão descritos o modelador de sólidos e elementos tecnológicos gerais; o nível classe, onde estão descritas as aplicações específicas e, por fim, o nível instância-classe, onde são criadas as entidades e o modelo geométrico.

Jasthi (1994), na implementação do sistema TURBO-MODEL para planejamento de processos, salienta a importância da modelagem de um produto para que dele se possam retirar as informações necessárias para o planejamento do processo e outras atividades necessárias para a fabricação do produto.

Para Van Der Net (1996), o modelo do produto é utilizado para satisfazer três necessidades básicas:

- criar uma descrição de um produto consistente para todos os estágios de projeto e manufatura;

- capturar e registrar as "intenções do projetista";

- permitir análise de manufaturabilidade ao mesmo tempo em que desenvolve o projeto.

Ainda segundo esse autor, um único modelo do produto deveria ser suficiente para descrever as informações em todos os estágios de projeto e processos de manufatura. Estando as informações num único modelo, essas seriam interpretadas conforme as necessidades dos diferentes estágios.

Van Der Net (1996) se utiliza dos conceitos de estados e de estados de transição para descrever o projeto de produtos, considerando que um estado de transição utiliza um determinado estado como entrada e que o resultado é um novo estado (Figura 2.1).

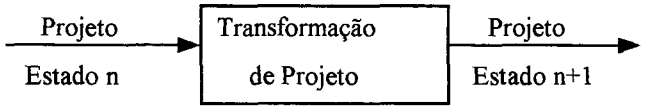


Figura 2.1 - Processo de projeto (Van Der Net, 1996).

2.1.2 - MODELO GEOMÉTRICO

O modelo geométrico corresponde às informações que o usuário terá à disposição para visualizar as representações por ele efetuadas e tem por objetivo registrar a disposição das informações geométricas de modo que mantenham a consistência com o mundo real. Vários sistemas de modelamento têm sido utilizados para essa finalidade. Como a visualização gráfica é fundamental para a interação usuário-computador, serão descritos rapidamente alguns dos sistemas de modelamento mais utilizados.

A modelagem mais elementar utilizada corresponde à representação através de linhas e pontos, a qual tem sido denominado de Modelo de Arame (*wireframe*). Este modelo, em virtude de sua simplicidade de construção, foi muito utilizado no passado; porém, no caso de uma maior complexidade das peças, sua estrutura não é adequada para a representação, portanto, tornou-se obsoleto.

Outro modelador é o baseado no método da Geometria Construtiva de Sólidos (CSG), que opera através de sucessivas operações lógicas efetuadas sobre um conjunto preexistente de formas volumétricas padronizadas. O sistema é representado através de uma estrutura binária.

Os modeladores baseados no método de representação por contorno (B-rep) trabalham com superfícies delimitadas, sendo possível definir neles qual lado é interior e qual é o exterior com relação a uma determinada face. As informações desse sistema são armazenadas em uma estrutura em forma de árvore, na qual os nós correspondem aos vértices, arestas e faces.

Segundo Cunha (1995), a representação de componentes baseada apenas no modelo geométrico não tem sido bem-sucedida em razão de dois fatores preponderantes: primeiro, a representação de diversos elementos fundamentais à descrição do projeto não é adequadamente contemplada nesses modeladores, considerando a necessidade de sua posterior recuperação e utilização pelos sistemas computacionais; o segundo fator diz respeito à representação da forma da peça exclusivamente através da representação das primitivas geométricas, que não explicita a existência de formas de mais alta ordem em termos de abstração quanto à compreensão do projeto.

Nesse ponto, uma saída é a utilização de uma estrutura exclusiva para representar as informações no formato desejado e adotar um modelador de modo que essa estrutura específica possua uma ligação com ele, permitindo, assim, criar uma ponte entre o modelo de informações e o modelador, solução aplicada por Lima (1994).

2.2 - REPRESENTAÇÃO PARA INTEGRAÇÃO

A busca da integração tem levado ao desenvolvimento de novas filosofias de representação das informações, já que os sistemas CAD utilizados somente comportam representações ditas de baixo nível, tipo retas, arcos, círculos, textos. Como exemplo, tem-se a representação de tolerâncias e acabamento superficial, que são representadas na forma gráfica e por isso dificilmente podem ser processadas para a obtenção de informações.

Wingard (1992) menciona que, no uso das tecnologias CAD/CAM atuais, o usuário tem que expressar a operação que ele quer representar usando termos que refletem a representação usada internamente no modelo do sistema, o que não traduz de forma adequada o conceito que se deseja representar.

Para Halevi (1994), os sistemas CAD atuais foram desenvolvidos para atuar somente com o desenho e atendendo a um mínimo das necessidades de projeto. Outros estágios e disciplinas não foram especificados como objetivos para os sistemas CAD, portanto, eles não suprem tais necessidades. Para sanar essas deficiências dos CAD atuais, tem-se lançado mão da tecnologia de *features*, que, de acordo com sua filosofia, possibilitam a representação das informações tão necessárias para uma integração do projeto e manufatura.

Segundo Jasthi (1994), uma *feature* é uma forma geométrica definida por um conjunto de parâmetros, os quais têm um significado especial para projeto ou manufatura, sendo, portanto, capazes de fornecer informações para um alto nível conceitual. Para Shah (1991), por sua vez, dentro do contexto de engenharia, *features* são formas genéricas às quais os engenheiros associam certas propriedades ou atributos e conhecimentos úteis em processos de raciocínio sobre o produto, ou seja, as *features* podem ser vistas como primitivas de engenharia.

Segundo Schulz et al. (1994), para o mapeamento de informações de baixo nível em informações de alto nível, aplicam-se três diferentes tipos de metodologias:

- identificação de *features*: o usuário seleciona iterativamente na tela elementos de um modelo geométrico já representados e os define como fazendo parte de uma determinada *feature*;

- reconhecimento de *features*: já convencionalmente definido o modelo geométrico (isto é, através de operações booleanas entre primitivas sólidas e também de operações de varredura), um programa de computador é executado para identificar automaticamente as *features*;

- projeto por *features*: o modelo da peça é definido diretamente no sistema através de *features*.

Na identificação das *features*, é o projetista que faz o seu reconhecimento, selecionando as entidades que fazem parte dela e informando o sistema, através de *features* predefinidas, quem é o elemento identificado. Esse tipo de abordagem, entretanto, possui o inconveniente de depender muito do usuário para o bom desempenho do sistema e, segundo Shah (1991), tem sido muito utilizado para a entrada de dados para programas de planejamento de processos e geração de trajetórias de ferramentas para programas NC.

Quanto à metodologia de reconhecimento de *features*, Jasthi (1994) e Dong (1993) argumentam a necessidade de se desenvolver algoritmos complexos e específicos para poder identificar simples formas geométricas, o que torna essa metodologia difícil de ser aplicada na prática. Para Bronsvoort (1994), o reconhecimento de *features* é uma atividade um tanto redundante: na fase de projeto, os conceitos de alto nível de informação do produto são traduzidos pelo projetista em informações geométricas de baixo nível, as quais voltam a ser processadas na fase de reconhecimento de *features* com a finalidade de reaver as informações de alto nível do produto novamente.

Fuh et al. (1996) apresentam um sistema integrado CAD/CAPP/CAFP[2] baseado em inteligência artificial que, utilizando a linguagem Prolog e através de regras e fatos, faz o reconhecimento de *features*. Para isso, utiliza uma representação da peça em 3D, buscando extrair a geometria da *feature*, ao passo que as informações de tecnologia são extraídas de uma representação em 2D. Os autores também reconhecem que as definições de *features* podem ser ambíguas, resultando na dificuldade de extração de uma *feature*; por outro lado, também afirmam que a abordagem de projeto por *features* é inconsistente com o que tradicionalmente é seguido pelos projetistas, o que pode forçá-los a trabalhar em detalhamentos e especificações muito maiores bem antes do estágio necessário.

Considera-se que essa inconsistência do caminho seguido e a especificação e detalhamento exagerado podem ser solucionados com a observação de uma metodologia a ser empregada em que as informações sejam passadas de forma mais natural. Por exemplo, pode-se fazer com que, inicialmente, o projetista apenas crie as formas geométricas e, após, seja executada a cotagem com a aplicação das tolerâncias, ao contrário do que alguns sistemas têm apresentado nos quais são anexadas todas as informações no momento da criação da *feature*.

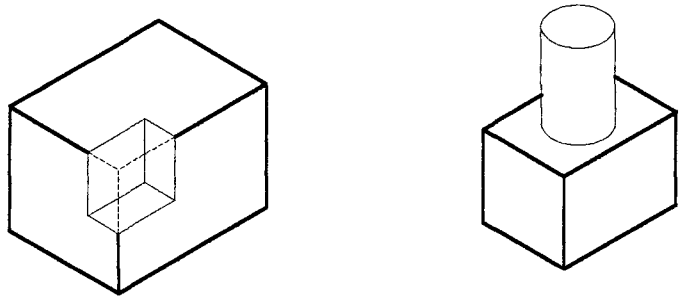
A grande vantagem da metodologia de reconhecimento de *features* estaria na condição de que o sistema pode reconhecer os projetos existentes executados em 2D, convertendo-os em

[2] CAFP ("Computer-Aided Fixture Planning")

features. Entretanto, origina-se daí uma grande dificuldade, pois as representações existentes também não estão completamente padronizadas.

O projeto por *features* baseia-se na existência de uma biblioteca de *features* e de procedimentos de utilização preconcebidos em termos de primitivas de engenharia, procurando obter uma representação de mais alto nível, próxima à necessidade de concepção do projetista, isto é, captar os elementos necessários para a representação completa de uma idéia.

Segundo Cunha (1993), para a abordagem de projeto por *features*, existem dois métodos diferentes: Geometria Destrutiva e Síntese de Elementos Volumétricos. O primeiro baseia-se na subtração volumétrica da peça de um dado volume principal, o qual, usualmente, é um bloco geométrico de matéria-prima. Esse método está intrinsecamente associado ao processo de fabricação, já que se deve conhecer de antemão o bloco que envolve a peça a ser construída (Figura 2.2). O segundo método trabalha com a junção elementar de volumes, os quais são definidos de acordo com o projeto ou significado de manufatura (Cunha, 1993). Nesse método, trabalha-se com formas definidas de elementos de engenharia, representando-se a forma final desejada da peça (Figura 2.2).



Geometria Destrutiva Síntese de elementos Volumétricos

Figura 2.2 - Projeto por *features*.

Schulz (1994) compara essas tecnologias considerando o suporte para o projetista, havendo algumas vantagens para o projeto por *features* em relação aos outros métodos descritos, a saber:

- o projetista interage com o sistema, o qual oferece uma semântica que representa elementos de projeto e manufatura;

- as relações geométricas são definidas para o mais alto nível, o que evita a interação com a geometria de baixo nível e reduz as possibilidades de erro;

- possibilita ao projetista transferir informações da base de dados que está disponível durante o processo de projeto;

- o modelo da peça descrita baseada em *features* contém a descrição geométrica e tecnológica da peça para, mais adiante, integrar os sistemas CAPP e CAM;

- bibliotecas padronizadas de *features* podem ser usadas para construir peças, com a previsão de serem manufaturáveis e a custo compatível;

- as intenções do projetista são resguardadas na representação da peça.

Para Mäntylä citado por Ovtcharova (1992), as principais vantagens no uso direto de *features* no projeto incluem:

- um vocabulário que é mais natural para expressar as peças do produto do que o puro modelamento geométrico;

- a possibilidade de usar *features* como base para o modelamento de informações do produto em diferentes fases do projeto, como análise, planejamento do processo e manufatura;

- um aumento na produtividade do projetista e eficácia nos custos.

A utilização de *features*, porém, leva a uma especialização do sistema , ou seja, quanto mais alto for o nível das *features*, mais elas representam um domínio menor de seu uso. Como vantagem, tem-se a maior rapidez em produzir um projeto e, conseqüentemente, redução do custo final do produto; como desvantagem, o usuário somente pode utilizar aquelas *features* que estão definidas na biblioteca, o que leva a uma limitação quanto ao universo a ser utilizado.

Com esse raciocínio, uma *feature* pode ser um simples eixo, o qual é descrito através de um grupo de informações que auxiliam na sua definição, como um eixo é definido pelo seu diâmetro e comprimento e, também, pode possuir informações sobre a sua localização no espaço, como as coordenadas das duas faces que limitam o seu comprimento, isso para informações de ordem geométrica. Podem ser anexadas informações sobre as tolerâncias das dimensões que definem o comprimento e o diâmetro; também informações de acabamento das superfícies que formam o eixo, isso levando em conta as informações mais elementares que podem ser registradas numa *feature*.

2.3 - MODELOS DE REPRESENTAÇÃO DO PRODUTO PARA MANUFATURA

Várias formas têm sido propostas para representar conhecimentos de engenharia em sistemas computacionais e, igualmente, várias metodologias, algumas infrutíferas e outras com potencial futuro.

Segundo Jasthi (1994), sistemas como o MICRO-CAPP e MICRO-GEPPS (Wang e Wysk, 1986), APPAS (Wysk et al., 1980), TOJICAP (Zhang e Gao, 1984) e AUTOCAP (Mindmany e Davis, 1981), todos aplicados para planejamento de processos, dependem da Tecnologia de Grupo para representar uma peça. Contudo, esse método não é suficiente para representar situações em que há grande quantidade de peças e lotes pequenos, visto que a tarefa de codificação torna-se tediosa e inadequada na forma como é feita nesses trabalhos.

Outro método descrito por Jasthi (1994) é o interativo, no qual o sistema possui uma série de perguntas a serem respondidas com relação à peça. Essa abordagem, no entanto, é limitada e não possui uma interface gráfica. Sistemas, como GCAPPS (Pande e Palsule, 1988), CAPSY (Spur e Anger, 1978) e CMPP (Sack, 1982), seguem essa abordagem.

Jasthi (1994) ainda descreve outro tipo de abordagem, que é definida como uma linguagem de descrição, em que a peça é descrita com informações detalhadas num formato padronizado. O GARI (Descotte e Latombe, 1985), CIMS/PRO (Iwata et al., 1980) e HIMAPP (Berenji e Khoshnevis, 1986) enquadram-se nessa categoria. A descrição, neles, é considerada tediosa e, algumas vezes, pode resultar num modelo ambíguo; além disso, a seqüência na qual é feita a modelagem pode afetar a lógica de planejamento de processos.

A codificação simbólica é outra forma descrita por Jasthi (1994), utilizada para modelar informações para um sistema CAPP, no qual a peça é descrita por um vetor de símbolos primitivos ou códigos alfanuméricos correspondentes, seguidos das dimensões associadas. O SIPPS (Liu e Allen, 1986) e um sistema registrado por Ramachandra et al. (1990) baseiam-se nessa abordagem.

Outro método que tem sido amplamente divulgado e aplicado é a utilização de *features* (Jasthi (1994), Krause (1993), Lima (1994), Rosa (1993), Shah (1988) e outros), o qual tem sido aplicado tanto para o modelamento do produto como para as aplicações que extraem informações do modelo do produto. Nessa abordagem por *features*, têm-se utilizado duas abordagens básicas: sistema de reconhecimento de *features* e sistema baseado em *features*, ambas descritas anteriormente (item 2.2).

2.4 - CONVERSÃO DO MODELO DO PRODUTO PARA MANUFATURA

Para Shah et al. (1994), são essencialmente dois os caminhos pelos quais os dados do produto são preparados para aplicações baseadas em *features* na manufatura:

- reconhecimento de *features* de manufatura de um modelo sólido, em que o modelo sólido é criado primeiro e as *features* de manufatura reconhecidas interativamente ou automaticamente;

- mapeamento de *features* do modelo para *features* de manufatura, sendo que, no mapeamento de *features*, há um benefício, pelo menos na teoria, da extração das informações de manufatura de um banco de dados.

Shah et al. (1994) apresentam alguns métodos para o reconhecimento de *features* de manufatura que têm sido utilizados:

- método de seccionamento;
- decomposição de superfície convexa;
- método baseado em contornos;
- decomposição celular.

O método de seccionamento é muito utilizado para gerar trajetórias de ferramentas para fresamento em 2 1/2D. O problema inicial é que a peça deve poder ser fabricada em máquina 2 1/2D. O caminho é orientado pelas direções dos eixos da máquina; o volume é seccionado em planos paralelos a X-Y, sendo variado o deslocamento dos planos em Z(ΔZ), representando as posições da ferramenta no eixo Z.

O método de superfície convexa, por sua vez, utiliza um algoritmo que decompõe o volume por subtração das superfícies convexas e repete o processo para todos os volumes resultantes. Para reconhecer as *features* de usinagem, os volumes positivos são convertidos em volumes equivalentes negativos. O método é limitado a poliedros sólidos.

Já os métodos baseados em contorno são todos aqueles que operam sobre o modelo primário "B-rep", utilizando a topologia e as relações entre entidades de contorno. Para cada *feature*, as condições geométricas e topológicas que precisam ser satisfeitas são identificadas; por fim, nas *features* de um modelo sólido, a base de dados é investigada para que se constate se as condições correspondem a cada *feature* presente.

Quanto aos métodos da decomposição celular, esses têm sido utilizados para determinar os volumes de usinagem da matéria-prima e do modelo da peça. A subtração booleana entre a matéria-prima e a peça resulta no volume total a ser removido, contudo precisa ser decomposto em partes, as quais correspondem a operações de usinagem específica.

Shah et al. (1994) classificaram o mapeamento em classes como descritas a seguir:

- um-para-um: quando não é requerido o mapeamento (transformação identidade). A *feature* como definida no projeto é utilizada na manufatura;

- reparametrização variante: diferentes conjuntos de dimensões são utilizados para definir a mesma *feature* em diferentes domínios; um furo escalonado no projeto pode ser definido por dois furos cegos de diferentes diâmetros. Para o planejamento do processo, esse furo escalonado cego pode ser feito por uma ferramenta escalonada, sendo redefinido apenas como um furo cego;

- agregação discreta: duas ou mais *features* inteiras combinam-se para produzir uma nova *feature*; uma *feature* furo escalonado cego é resultante da união de dois furos cegos;

- decomposição discreta: uma *feature* é decomposta em duas ou mais *features*; uma *feature* furo escalonado cego definida no projeto pode ser decomposta, em função dos diâmetros, em duas *features* furos cego quando do planejamento do processo;

- conjugação: uma nova *feature* é sintetizada de porções de duas ou mais *features*.

Para cada aplicação que se utiliza de um conjunto fixo de *features* definidas no modelo do produto, é necessário mapear essas *features* de projeto para a aplicação desejada. Para isso, é necessário enumerar todas as situações de mapeamento que podem ocorrer para cada tipo de aplicação, especificando os procedimentos de mapeamento.

2.5 - MODELAMENTO POR *FEATURES*

Shah (1988) descreve requisitos funcionais necessários ao modelamento geométrico para projeto de engenharia, que consiste em:

- deve dar suporte a todos os conjuntos de informações necessários para representar um produto;

- como a definição de cada *feature* depende da aplicação, é necessário que haja um mecanismo que possibilite definir o conjunto de *features* adequadas;

- o sistema definido deve possuir um meio para criar, manipular, modificar e excluir o modelo do produto, bem como definir relações entre as *features*;

- deve possibilitar criar uma interface de mapeamento das *features* do modelador para a aplicação específica.

Bronsvoort (1994) relaciona uma série de requisitos com relação à funcionalidade, que devem ser observados num sistema de modelamento por *features*, e estão descritos a seguir:

- o sistema deve ser interativo e gráfico, pois este é o melhor caminho para dar suporte adequado ao modelamento de processos;

- deve existir um mecanismo para definir as *features* genéricas e armazená-las na biblioteca de *features*; deve-se considerar a parametrização geométrica, na qual somente a geometria , a posição e o dimensionamento do objeto podem variar, bem como a parametrização topológica, na qual a estrutura de um objeto pode variar;

- deve haver um mecanismo para criar instâncias de *features* pela especificação de parâmetros; o sistema deve ser, então, capaz de determinar a forma da *feature* na base de definições genéricas e parâmetros;

- restrições devem ser capazes de garantir a validade das *features*. Se uma modificação é feita no modelo, que implica modificações na base de restrições, então, o sistema deve, automaticamente, assumir essas modificações;

- tanto o projeto *top-down*, no qual inicialmente o projeto conceitual em alto nível é feito e, mais tarde, redefinido para baixo nível no projeto de detalhes, quanto o projeto *bottom-up*, no qual um objeto pode ser construído de simples peças, devem ser suportados; *features* hierárquicas devem estar disponíveis para permitir a composição de outras, consistindo de várias *features* elementares.

O sistema GeoNode (BRONSVOORT, 1994) é um modelador de *features* que utiliza o conceito de conversão de *features* a partir de uma visão centralizada, que, no caso, é a visão de projeto. Essa visão central é denominada de *visão primária*, aplicando-se modelos específicos de conversão para obter, assim, a visão secundária, que é função da aplicação. Isso implica que modificações na visão de projeto produzem, automaticamente, modificações nas visões de aplicação (secundárias); em oposição, modificações nas visões de aplicação não resultam em modificações na visão de projeto e nas outras visões de aplicação.

Para Bronsvoort (1994), numa situação ideal, a alteração nas visões de aplicação deveria produzir alterações na visão de projeto. Porém, isso pode trazer uma série de problemas quanto

ao controle geral, pois, a partir de qualquer visão de aplicação, particularmente à ligada a um domínio muito específico, pode-se produzir alterações que venham a descaracterizar o projeto.

2.6 - *FEATURES* NUM SISTEMA CAD

O uso de *features* para representar informações pressupõe que essas possuam uma representação geométrica para que o usuário possa visualizá-las num formato adequado, permitindo a utilização de um sistema CAD. Para isso, é necessário que alguns itens sejam satisfeitos. Para Roller (1989), os itens para que um sistema CAD possa dar suporte ao projeto por *features* são os seguintes:

- uma estrutura de dados apropriada para capturar *features* no banco de dados do CAD;
- avaliação das bibliotecas de *features* a serem implementadas em função da aplicação, em forma paramétrica;
- mecanismos para manusear restrições de projeto, que se refletem em restrições nas *features*, o que inclui um método para atribuição de regras às *features*;
- comandos de manipulação para *features*, como mover, rotacionar ou excluir.

Shah e Rogers citados por Pratt (1990), propõem que um sistema baseado em *features* deve ser bastante flexível com respeito a dois itens: o primeiro consiste em que diferentes aplicações requerem diferentes classes de *features*, sendo importante que o sistema possa ser configurado de acordo com a aplicação; em segundo lugar, deve-se excluir *features* de baixo nível, que nem sempre correspondem àquelas criadas pelo projetista, devendo, de alguma forma, serem providenciadas a identificação e a aplicação desse tipo de *feature*.

Segundo Roller (1989), as *features* podem ser obtidas por primitivas geométricas, as quais, nesse caso, devem ser criadas pelo método tradicional (p. ex., a peça de um projeto) e, subseqüentemente, providas com atributos das *features*.

Um dos grandes problemas na implementação de sistemas CAD é em relação à manutenção da coerência entre as informações em nível de *feature* e aquelas em nível de geometria; em outras palavras, dar um tratamento gráfico adequado para que a imagem apresentada ao usuário seja fiel às informações.

2.7 - CAD INTELIGENTE

O conceito de "CAD inteligente" tem ganho adeptos à medida que as pesquisas têm progredido em diversas áreas do conhecimento, visando desenvolver formas de armazenar o conhecimento humano para uso computacional (inteligência artificial, sistemas especialistas, etc.).

Para essa tarefa, é necessário que o sistema CAD seja utilizado não somente para representação das informações geométricas e tecnológicas (p. ex., representação com a utilização de *features*), mas possua capacidade de efetuar determinadas tarefas automaticamente, auxiliando, assim, o usuário.

Dentre dessas atividades de auxílio ao usuário, está a capacidade de gerenciar informações de modo a analisá-las e de transmitir ao usuário um parecer. Por exemplo, informar que determinada configuração de uma peça, para uso de um determinado material que envolve um processo específico, pode resultar em problemas na sua execução.

Segundo Ando (1989), um CAD inteligente, incorporando o processo de pensamento humano, deverá ter a habilidade de descrever as intenções do projetista, o qual contribuirá significativamente para a integração entre CAD e CAM. Ainda para o autor, são os principais elementos de um CAD inteligente:

- a assistência ao pensamento humano na tarefa de projeto;

- a realização de geração automática completa das informações de manufatura baseadas no resultado único do projeto.

Um CAD inteligente pressupõe a interligação com um sistema de banco de dados para armazenar informações que servem de auxílio ao usuário durante o processo de projeto, bem como um sistema especialista que é ativado num dado momento e que compara as informações do CAD com a base de conhecimento e a base de dados para, então, dar um veredito (operação aceita ou não) sobre o elemento analisado.

Tomiyama (1993?) enumera uma série de defeitos que os sistemas CAD convencionais possuem, como:

- algumas das atividades de projeto são só parcialmente auxiliadas e deveriam ser capazes de dar suporte ao projeto conceitual;

- a habilidade em resolver problemas é fraca, sendo realizadas apenas aquelas tarefas com soluções predefinidas. Os sistemas deveriam possuir funções para conduzir o projeto criativo cuja solução é desconhecida;

- falta de integração, visto que os modelos de representação não são integrados e os processos de projeto não são unificados, devendo o novo sistema CAD possuir mecanismos que gerem interfaceamento automaticamente, bem como organizar e unificar os processos de projeto.

Dentre os defeitos enumerados por Tomiyama, a questão de os sistemas não serem capazes de conduzir um projeto criativo, ou de resolverem problemas com soluções predefinidas, não pode ser considerada como um defeito, mas como uma limitação dos sistemas computacionais, visto que ainda não se conseguiu construir algo que seja capaz de aprender sozinho.

Para Arbab, citado por Anantha (1996), o raciocínio geométrico é o ponto central para a unificação do projeto e da manufatura. Assim, um sistema de CAD inteligente deve ser capaz de raciocinar no domínio da geometria, de manusear restrições geométricas e de satisfazer essas restrições de uma forma completa e sem ambigüidades.

2.8 - REPRESENTAÇÃO POR *FEATURES*

Em razão das diversas pesquisas dos últimos anos, a linha de pesquisa em *features* tem ganho um grande desenvolvimento no que se refere à representação de objetos e de informações de engenharia. Para a execução de uma representação adequada por *features*, é necessário, primeiramente, que essas estejam classificadas, criando-se, assim, relações de dependência e hierarquia entre elas.

Para Shah, citado por Moreira (1993), a identificação e classificação das *features* são realizadas com os seguintes objetivos :

a) definir um domínio de aplicação e, dessa forma, simplificar a representação dos elementos;

b) definir uma terminologia comum para o uso de *features*;

c) definir uma taxonomia comum, útil na definição de normas para intercâmbio de dados.

Para Shah (1988), as *features* são conjuntos de informações usadas em projetos de engenharia e aplicações de manufatura, dependendo do tipo de produto, da aplicação e do processo envolvido. Propõe, ainda, que se representem *features* através de espaços de *features*, com base em considerações de que elas dependem do tipo de produto, da aplicação e do nível de

abstração. O autor utiliza essa proposta, considerando a transformação de um nível para outro, ou seja, passar do nível de projeto para o da manufatura.

O mapeamento proposto por Shah (1989) e implementado por Bronsvoort (1994), e também por Wosny (1994), é interessante do ponto de vista da conversão de um domínio de aplicação para outro. Porém, para que isso ocorra, é necessário que os domínios em questão sejam amplamente conhecidos a fim de que sejam definidos exatamente os seus correspondentes.

Segundo Gao (1996), recentemente, têm-se disponibilizados sistemas CAD comerciais baseados em *features*, estando entre os mais conhecidos o Pro-Engineer (M. James, 1991) e o CADDS5 (Computervision, 1991). Gao (1996), entretanto, considera que nenhum deles tem implementado uma quantidade suficiente e aceitável de *features* para representar o produto e utilizar as informações para planejamento de processos.

Cunha (1995) cita vários problemas para sistemas baseados em *features* que não apresentam solução, problemas esses levantados da literatura (referências 15,19,23,63,22,29, em CUNHA, 1995) e descritos como sendo os seguintes:

a) o problema da necessidade de serem criadas e utilizadas formas predefinidas canonicamente[3] ;

b) a dificuldade de contar com um conjunto suficientemente completo e significativo de *features* com relação à área de projeto ou a aplicação a ser realizada;

c) o problema de uma mesma peça poder ser gerada por mais de uma combinação possível de formas definidas canonicamente (ambigüidade na composição da forma da peça), o que pode conduzir, posteriormente, a análises diferenciadas da mesma;

d) o problema de ser necessário coexistirem diferentes possibilidades de parametrização de uma mesma forma característica (*features*);

e) o problema das interações geométricas entre as formas características, em que se perde total ou parcialmente o significado associado a elas;

f) o problema da definição de uma maneira adequada de representação da topologia da peça, seja através de grafos que representam as adjacências entre as formas características (*features*), seja através das relações hierárquicas entre essas, em que a existência de uma fica condicionada à existência ou posição relativa de outras.

---

[3] Numa relação ou catálogo

Ainda para Cunha (1995), tanto na abordagem de reconhecimento de *features* como naquela baseada em *features*, a representação das peças é feita de duplo modo, coexistindo um modelo voltado para a representação em modo gráfico com outro dedicado à representação no modo textual (arquivo-texto).

No caso do reconhecimento de *features*, faz-se um mapeamento das formas identificadas no modelador gráfico para uma base de dados que irá conter as informações em modo textual. Por sua vez, no caso da utilização baseada em *features*, as duas possibilidades de representação coexistem: a primeira utiliza a interface gráfica para a construção da peça, atuando sobre o modelador gráfico e, após a validação da construção, transferindo a informação referente a essa construção para uma base de dados (modo textual); a outra hipótese consiste em construir a peça a partir de uma interface textual, na qual a geometria e a topologia da peça são declarativamente descritas, utilizando funções gráficas para a produção da representação gráfica. Quanto a essa última hipótese, é um tanto inadequada a sua utilização, pois a execução da representação textual, inicialmente, fica na contramão da prática utilizada até hoje nos meios de engenharia para representar um projeto.

2.9 - CLASSIFICAÇÃO DAS *FEATURES*

Para Shah (1988), um conjunto de informações adequadamente agrupadas para representar um produto é chamado de *feature*. O autor classifica três formas básicas - *feature* de forma, *feature* de precisão e *feature* material - ressaltando que existem outros conjuntos lógicos de informação, como *features* de montagem, *features* funcionais, *features* de análise, etc.

As *features* de forma são aqueles grupos de entidades geométricas que definem atributos de peças com respeito a tamanho e forma nominal. As *features* de precisão, por sua vez, representam as informações a respeito dos desvios aceitáveis da geometria nominal, incluindo-se as tolerâncias dimensionais e acabamento superficial. Já *feature* material diz respeito às especificações, propriedades, tratamento térmico, tratamento superficial, etc.

Pratt e Wilson, citados por Shah (1991), adotam a classificação das *features* baseada inteiramente na forma antes da aplicação.

Feng et al. (1996) apresentam um trabalho definindo *features* relacionadas à função, ou seja, cada *feature* está relacionada a uma função no produto, apresentando três tipos básicos que são:

- funções relacionadas ao desempenho requeridas para o produto;
- funções correspondentes aos requisitos dos processos de manufatura;
- funções relacionadas com os requisitos de ergonomia.

Para Feng et al. (1996), três razões justificam o estudo das relações das funções das *features*:

- evitar a omissão de certas funções e *features* num projeto, isto é, determinadas condições podem ser requeridas para que duas *features* possam ter a funcionalidade correta;
- contribuir para a análise de tolerâncias; a informação das relações funcionais é usada para determinar as superfícies de referência para as tolerâncias no projeto;
- contribuir para aplicações posteriores, tais como planejamento de processos, em que informações da geometria e funcionalidade facilitam a determinação da seqüência de operações.

Essas relações funcionais são subdivididas em implícitas e explícitas e em outras sub-relações que auxiliam a transmitir a idéia de funcionalidade. Esses dados são, então, representados através de um grafo e uma matriz.

Outro item se refere à cotagem, que tem por base dados obtidos a respeito do produto montado, sendo abordada a cotagem funcional.

Wingard, citado por Lima (1994), propõe uma taxonomia dependente da área de aplicação, sugerindo uma divisão de *features* de forma em duas subclasses: atômicas e compostas. As atômicas são *features* simples que podem ser decompostas como Peça, Modificador e Grupo. A peça é uma *feature* atômica com existência própria; modificador é dependente de outra *feature*, e grupo reúne várias *features* que possuem um significado de engenharia.

2.9.1 - CLASSIFICAÇÃO DAS *FEATURES* DE FORMA

Dixon (1987) apresenta uma classificação das *features* de forma bem ampla e generalizada, iniciando pela definição de *features* estáticas e dinâmicas e particularizando as estáticas. As *features* estáticas são, primeiramente, estruturais na sua exigência funcional (p. ex., uma depressão); as dinâmicas exigem transferência de movimento ou energia para preencher sua função (p. ex., engrenagens). As estáticas são subdivididas em primitivas, intersecções, macros e alguns outros tipos, gerando, assim, uma grande quantidade de *features*.

Juri et al. (1990), em seu modelo implementado para um sistema CAPP, classificam as *features* de forma em primárias e secundárias e, também, em externas e internas. As primárias correspondem a eixos cilíndrico e cônicos, e as secundárias são os furos, roscas, filetes, etc.

Chen, citado por Feng et al. (1996), divide as *features* em primárias e secundárias. As primárias são independentes de qualquer outra *feature*, ao passo que as secundárias dependem de seus pais, sendo usadas para modificar a forma das peças.

Ovtcharova et al. (1992) classificam as *features* de forma com base em sua complexidade, tendo como orientação a classificação STEP. Tal classificação inicia com *features* simples e compostas, numa forma genérica, chegando até a enumeração das *features* mais básicas da estrutura, que são, na realidade, aquelas utilizadas para o instanciamento. Essa classificação pode ser melhor entendida, observando-se a Figura 2.3.

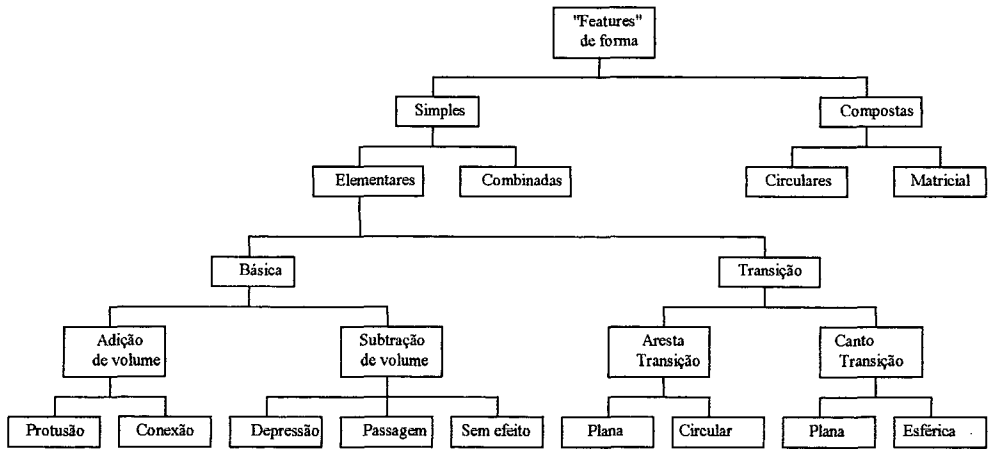


Figura 2.3 - Classificação das *features* de forma de acordo com a complexidade (Ovtcharova, 1992).

Lima (1994) apresenta uma outra classificação das *features* de forma, separando-as, inicialmente, em paramétricas e não paramétricas como grupos principais (Figura 2.4). Dentre as paramétricas, além das *features* simples e compostas, o autor acrescenta o conceito de *features* modificadoras; dentre as não paramétricas, aborda somente as macro-*features*. Vê-se, na Figura 2.4, que as *features* a serem instanciadas são classificadas diretamente em *features* de volume positivo e negativo, ou seja, elementos que possuem massa e elementos que representam o vazio nessas massas.

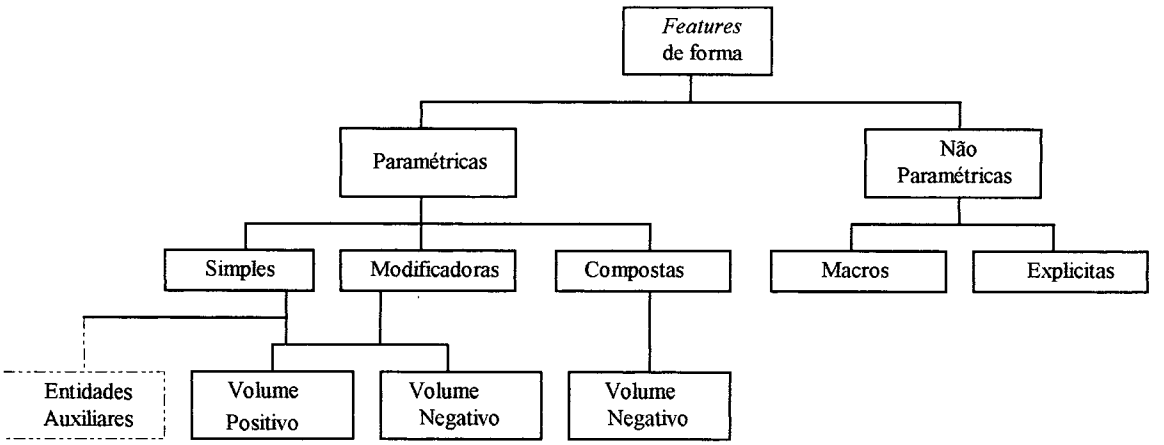


Figura 2.4 - Classificação das *features* de forma (Lima, 1994).

Schulz (1993) classifica as *features* com base no trabalho desenvolvido pelo CAM-I[4], dividindo-as, basicamente, em rotacionais e prismáticas. Nessa classificação, as *features* rotacionais são divididas em quatro grupos: o primeiro corresponde às *features* básicas e, no segundo, têm-se as de diâmetro externo, que estão localizadas no exterior de uma *feature* básica; as *features* de diâmetro interno se encontram no terceiro grupo; no quarto, as rotacionais que estão na face. Quanto às *features* prismáticas, possuem duas subdivisões principais: as prismáticas axiais e as radiais.

2.10 - A REPRESENTAÇÃO DO CONHECIMENTO

Para Dixon (1995), pode-se representar o conhecimento através do conhecimento explícito ou implícito. O explícito é representado separada e distintamente do algoritmo de solução do problema, podendo ser modificado ou substituído sem interrupção ou modificação do código que expressa o algoritmo; por outro lado, o algoritmo que resolve alguns problemas representa o conhecimento implícito.

Várias técnicas para a representação do conhecimento têm sido desenvolvidas, as quais segundo Giarratano (1993), incluem regras, redes semânticas, *frames*, *scripts*, linguagem de representação de conhecimento, assim como o KL-1 (Woods (1983), citado por Giarratano (1993)) e KRYTON (Bachman (1983), citado por Giarratano (1993)) e outras.

[4] "Computer Aided Manufacturing-International"

Para Rich, citado por Juri (1990), existem duas formas válidas de representação que são através de *frames* e de regras. Tipicamente, um *frame* descreve uma classe de objetos, sendo uma coleção de *slots* usados para caracterizar uma variedade de elementos associados a um objeto. Um *frame* liga o conhecimento a ser organizado de forma hierárquica, possuindo a propriedade da herança, o que permite que um *frame* de mais baixo nível herde propriedades de um de nível mais elevado.

Segundo Giarratano (1993), alguns tipos de *shells* para sistemas especialistas, como o CLIPS (*C Language Integrated Production System*), ligam objetos a regras, nos quais o conhecimento pode ser encapsulado. Regras podem modelar, igualmente, objetos como fatos; alternativamente, objetos podem operar independentemente das regras.

Dixon (1995) trabalha com bases de conhecimento com o objetivo de não apenas armazenar o conhecimento sobre um assunto, mas também o conhecimento sobre os métodos de solução para os problemas. Dessa forma, é possível substituir o método, ou métodos de solução sem que o algoritmo básico tenha de ser substituído.

2.10.1 - SISTEMAS ESPECIALISTAS

Um sistema especialista tem por finalidade dotar programas de computador com um tipo de mecanismo de análise que, através do uso de regras, fatos ou *frames*, possibilite modelar informações não algorítmicas (heurísticas), que não possuem uma representação matemática exata.

De acordo com Waterman (1986), o uso de sistemas especialistas considera que, para ajudar na solução do problema, o sistema deve possuir um alto índice de especialização, que é representada pelo conhecimento embutido no sistema para que se obtenha um resultado preciso e eficiente.

Feigenbaum, citado por Giarratano (1993), tem definido um sistema especialista como "...um programa de computador inteligente que usa conhecimento e procedimentos de inferência para resolver problemas que são muito difíceis e que requerem um especialista (ser humano) para resolver o problema". Na realidade, um sistema especialista é um programa computacional que emula (o termo *emulação* é muito mais forte que *simulação*, que, no caso, apenas requer alguns aspectos próximos da realidade) a capacidade de um especialista humano. Para Coyne (1990), o objetivo é representar o conhecimento numa forma que seja compreensível para o

homem e a máquina. Para isso, Araribóia (1988) relaciona duas condições que um sistema especialista deve oferecer ao usuário:

1 - uma linguagem de construção de bases de conhecimento;

2 - um mecanismo de inferência que permita o acesso ao conhecimento implícito que possa ser obtido raciocinando sobre o conteúdo da base de conhecimento.

Segundo esse autor, ainda, os sistemas de representação de conhecimento, para executar as tarefas cognitivas, operam em três estágios:

1 - adquirir conhecimentos;

2 - recuperar as informações da base de conhecimento que estão relacionadas com o problema;

3 - raciocinar sobre as informações para alcançar a solução.

Conforme Waterman (1986), para a constituição de um sistema especialista, são necessários o domínio da especialidade, um engenheiro de conhecimento, uma ferramenta para a construção do sistema especialista e um usuário final (Figura 2.5).

A área de domínio do sistema especialista corresponde ao conhecimento adquirido por pessoas que trabalharam anos no assunto e que adquiriram uma forte experiência na solução dos problemas, estando, portanto, aptas a produzir boas soluções para serem implementadas num sistema. O conhecimento do especialista não está registrado em livros nem em artigos; é um conhecimento implícito ao especialista e deve ser extraído para, então, ser codificado num sistema especialista. Nesse sentido, o engenheiro de conhecimento é o indivíduo que possui conhecimentos em computação e inteligência artificial e que sabe como construir o sistema especialista; é quem organiza o conhecimento dentro do sistema, definindo os métodos de pesquisa.

A ferramenta de construção do sistema especialista é uma linguagem de programação utilizada para construir o sistema. Essas ferramentas diferem das linguagens convencionais em virtude do tipo de conceitos a representar (Waterman, 1986). Quanto ao usuário, pode ser considerado de várias formas, dependendo do enfoque a ser analisado. Pode ser: a) o indivíduo que está testando a própria ferramenta; b) o engenheiro de conhecimento refinando o conhecimento do sistema; c) alguém introduzindo novos dados na base de conhecimento (usuário autorizado) ou, finalmente, d) o usuário que utiliza o sistema especialista em si para a finalidade para a qual foi construído (usuário final).

Na Figura 2.5, a região envolvida pelo retângulo traço ponto corresponde às atividades normalmente desenvolvidas no trabalho de implementação de uma base de conhecimento, cuja descrição detalhada será feita no capítulo 4.

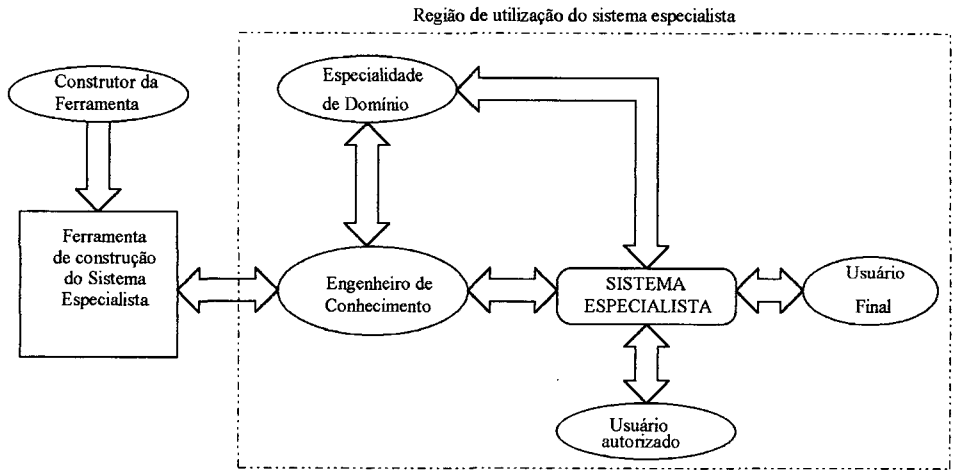


Figura 2.5 - Elementos envolvidos num sistema especialista (Waterman, 1986).

É importante distinguir entre a ferramenta usada para construir o sistema especialista e o sistema em si. A ferramenta de construção fornece ambas as linguagens para representar e consultar o conhecimento contido no sistema e os meios de suporte à programação (programas que auxiliam o usuário a interagir com o sistema) (Waterman, 1986).

Na Figura 2.6, está ilustrado o conceito básico de uso do sistema especialista em um nível de usuário, que pode ser dividido em duas partes principais: a base de conhecimento que contém o conhecimento e o mecanismo de inferência, que retira conclusões a partir do conteúdo da base de conhecimento.

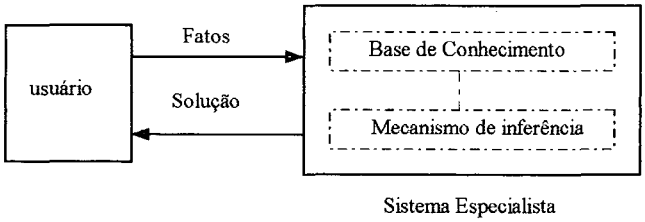


Figura 2.6 - Conceito básico de um sistema especialista (Giarratano, 1993).

Há algumas vantagens em utilizar-se sistemas especialistas. Muitas vezes, por exemplo, a perícia humana, por desatenção, pode falhar ou ter uma habilidade desaparecida se o especialista não a praticar. Num sistema especialista, por sua vez, uma vez adquirido o conhecimento, esse

permanece, a menos que haja uma catástrofe no sistema; por outro lado, esse conhecimento pode ser transferido ou reproduzido. Com o ser humano, essa transferência é lenta e cara, denominando-se educação. Já o conhecimento artificial é mais consistente que o humano, não sendo influenciado por fatores externos, como, por exemplo, fatores emocionais. Certamente, mesmo com várias vantagens, o ser humano não pode ser eliminado, pois, em muitas situações, somente ele pode decidir, sendo superior em criatividade, adaptação, aprendizagem e outras funções bem específicas e que ainda não são possíveis de serem traduzidas para um sistema. Assim, o sistema especialista é utilizado para melhorar e aumentar a habilidade do usuário (waterman, 1986).

Segundo Harmon, citado por Giarratano (1993), o conhecimento de vários especialistas pode ser avaliado simultânea e continuamente num problema, a qualquer hora; além disso, o nível de especialidades combinadas de vários especialistas pode exceder o que um único especialista humano pode ter. Para Giarratano (1993), um sistema especialista pode proporcionar uma explicação em detalhes sobre o raciocínio que levou a uma conclusão; já um ser humano pode estar cansado, relutante ou impossibilitado para isso por um tempo. Isso aumenta a confiança com relação à decisão do sistema especialista.

Deve-se lembrar, entretanto, que, para construir uma base de conhecimento de um sistema especialista, todas as opções que podem ocorrer numa determinada situação devem estar descritas. Isso exige pesquisa mais exaustiva com relação às soluções apresentadas; mesmo aquelas que se deviam à sensibilidade do especialista devem, agora, receber uma explicação melhor para que possam ser mais bem compreendidas e possibilitem a sua codificação.

Uma outra aplicação dos sistemas especialistas se dá em situações de emergência (Giarratano, 1993), quando a rapidez na tomada de decisão é importante, o que muitas vezes não ocorre com o ser humano. O uso de sistemas especialistas integrados a outros tipos de programas deve propiciar uma alta flexibilidade para o sistema, já que, para a alteração da base de conhecimento, não há a necessidade de manuseio de programas-fontes do sistema principal. Com isso, o sistema pode ser adaptado para determinadas variações que possam surgir na implantação de uma nova aplicação.

Para Waterman (1986), um sistema especialista deve ser ágil, isto é, deve aplicar o conhecimento para produzir soluções eficientes e efetivas, usando atalhos ou estratagemas que um perito humano adotaria para eliminar o desperdício ou cálculos desnecessários.

A aplicação de sistemas especialistas para avaliação nas decisões de sistemas CAD e em sistemas aplicados à manufatura traz algumas vantagens, que podem ser descritas como:

- torna-se fácil a adaptação das tomadas de decisões com relação ao ambiente a ser aplicado, pois permite que as regras sejam alteradas sem que seja necessária a compilação do sistema;

- a inserção das regras pode ser feita de modo a não obedecer a uma seqüência; já, na programação tradicional, a seqüência influencia diretamente o resultado;

- uma regra sempre diz uma verdade, ou seja, se existem tais condições somente há uma resposta para aquelas condições;

- a ampliação da base de conhecimento pode ser efetuada facilmente, pois novas regras podem ser inseridas sem que as anteriores sejam alteradas.

Para isso, é necessário que as definições utilizadas para o sistema representem de forma única as informações a fim de que não ocorram contradições.

Waterman (1986) apresenta em seu livro exemplos de várias áreas em que sistemas especialistas foram aplicados, como na química, com o CONPHYDE, que auxilia na seleção de propriedades físicas por métodos estimativos, sistemas computacionais, eletrônica, engenharia, geologia, medicina, etc., mostrando a estrutura de cada aplicação; o DELTA, que auxilia a identificar e corrigir defeitos em locomotivas diesel-elétricas, além de outros.

Lima (1994), por sua vez, utiliza um sistema especialista para realizar as análises quanto às condições de inserção de *features* num sistema CAD e algumas análises de manufatura. O sistema especialista recebe informações do módulo gráfico e executa as referidas análises. Rezende (1996) também utiliza um sistema especialista integrado com um sistema CAD para efetuar análises a respeito de planejamento de processos, o qual possui a entrada de dados via CAD em forma de desenhos, que são traduzidos para o sistema especialista para efetuar as análises, definindo a posição da peça, as ferramentas utilizadas e a seqüência de operações.

Para Gupta, citado por Gao (1996), o uso de inteligência artificial integrado a outros sistemas mostra significativas vantagens sobre o uso da linguagem normal de programação. Como exemplo, Gao (1996) cita as dificuldades no uso de árvores de decisão e de tabelas de decisão em sistemas de planejamento de processos, as quais devem ser codificadas linha a linha. Avalia que, com sistemas especialistas, a forma de armazenar conhecimentos é mais organizada, resultando na facilidade de adicionar ou deletar informações da base de conhecimento.

Quanto à facilidade de adicionar conhecimento, isso se deve ao fato de que o sistema executa a procura das informações, o que é governado pelo mecanismo de inferência, que possui mecanismos especiais de procura, seqüencial ou não; nas linguagens tradicionais, por sua vez, o programa flui seqüencialmente, obrigando a informação a estar numa determinada posição para que o sistema funcione, do contrário pode gerar ambigüidades.

A diferença funcional entre a linguagem de um sistema especialista e a linguagem procedural é a aplicação. Na linguagem procedural, a aplicação se concentra em providenciar técnica robusta e flexível para a representação de dados. Por exemplo, estrutura de dados como matrizes, listas, filas,etc. Já o paradigma de sistemas especialistas liga dois níveis de abstração: a) a abstração de dados e b) a abstração do conhecimento. As linguagens de sistemas especialistas separam os dados dos métodos de manipulação de dados. Como exemplo, tem-se a separação dos fatos (abstração de dados) e regras (abstração do conhecimento) nas linguagens de sistemas especialistas baseados em regras.

Uma diferença fundamental está na metodologia de programação, uma vez que, na programação estruturada, a seqüência de execução deve ser descrita cuidadosamente. Já, em razão da separação dos dados do conhecimento nos sistemas especialistas, esses requerem uma menor rigidez no controle da seqüência de execução. Essa separação entre dados e conhecimento permite um alto grau de modularidade aos sistemas.

Segundo Krishnan (1995), há enormes vantagens em ter-se num sistema especialista a base de conhecimento e o mecanismo de inferência separados. São elas:

1) o construtor do sistema especialista pode ater-se exclusivamente à construção da base de conhecimento;

2) uma troca de informações pode ser realizada numa parte da base de conhecimento sem afetar substancialmente as outras partes;

3) o mesmo mecanismo de inferência pode ser utilizado com diferentes bases de conhecimento;

4) diferentes mecanismos de inferência podem ser experimentados com uma base de regras particular;

5) a melhora no mecanismo de inferência pode ser efetuada sem prejuízo do sistema.

Um sistema especialista é projetado para negociar com as incertezas, pois o raciocínio é a melhor ferramenta existente para tratá-las.

Krishnan (1995) focaliza o projeto e o desenvolvimento de uma metodologia para implementação de uma base de conhecimentos para auxiliar o projeto de montagem de circuitos impresssos (PCB). Para isso, utiliza-se de programação orientada para objeto e de regras para modelar a base de conhecimentos do sistema especialista.

2.10.2 - REPRESENTAÇÃO DO CONHECIMENTO NO SISTEMA ESPECIALISTA

Duas formas mais comuns de representação do conhecimento em sistemas especialistas são: a) o conhecimento dinâmico, que é representado por fatos, e b) o conhecimento estático, que é representado por regras do tipo IF - THEN.

Giarratano (1994) representa o conhecimento através da utilização de classes. Assim os fatos são substituídos pelos atributos dos objetos de cada classe. Isso é mostrado através da organização das classes (ver item 2.11) para o sistema especialista, no qual elas podem ser divididas, por exemplo, em:

FEATURE
EIXO
EIXO CILINDRICO.

Assim, a definição inicial corresponde a uma definição genérica e, à medida que é herdada por uma nova classe, torna-se cada vez mais especializada. Assim têm-se:

FEATURE
Nome
Posição
Direção
Tipo
Face Inicial
Face Final

EIXO *IS-A FEATURE* (herda os atributos de *feature*)
Comprimento

EIXO CILÍNDRICO *IS-A* EIXO (herda os atributos de eixo)
Diâmetro

A classe *FEATURE* representada corresponde a uma classe básica. Uma nova classe chamada EIXO é criada onde *IS-A* indica que EIXO é uma *FEATURE*, ou seja, que o eixo herda todas as informações da *FEATURE*.

A classe mais especializada é EIXO CILÍNDRICO, que é um EIXO, ou seja, herda todas as informações da classe EIXO, que, por sua vez, herda as informações da classe *FEATURE*; logo, EIXO CILÍNDRICO é a união de todos os atributos das classes que herdou.

EIXO CILÍNDRICO

Nome
Posição
Direção
Tipo
Face Inicial
Face Final
Comprimento
Diâmetro

A representação das regras é feita em duas partes: a primeira parte da regra, denominada de antecedente, premissa ou lado esquerdo (LHS), que expressa uma situação; a segunda, chamada de conseqüente ou lado direito (RHS), estabelece a ação ou conclusão. Assim, uma regra pode ser expressa:

IF condição **THEN** ação ou conclusão

O conjunto de regras de conhecimento forma o domínio da base de conhecimento do sistema segundo Brownstoon et al., citado por Juri (1990). Essas regras podem, então, ser expressas em função da estrutura declarada nas classes do sistema.

Segundo Giarratano (1993), um dos mais populares tipos de sistemas hoje é o baseado em regras de produção, visto que é de natureza modular, de fácil interpretação e similar ao processo de pensamento humano. Uma regra de produção pode ter múltiplos antecedentes (lado esquerdo) para uma única conclusão. Assim, tem-se a representação de uma regra de produção:

IF condição1 **AND** condição2 **AND** ..... **THEN** ação ou conclusão

Como exemplo, pode-se ter (Figura 2.7):

IF [(Nome = EIXOCILÍNDRICO) (Posição = EXTERNA) (Direção = AXIAL)
(PtoX > Face finial)]

THEN [(Sentido = 1) (Flag = 0) (o eixo pode ser inserido ao lado direito de um eixo existente (Figura 2.7))]

Nessa regra, o LHS corresponde a uma situação plenamente conhecida e já definida no sistema, que, no caso, corresponde à existência de uma *feature* eixo cilíndrico, e uma outra *feature* que se deseja inserir do lado direito da *feature* existente. Como resultados, têm-se dois atributos que irão disparar funções que executarão a respectiva inserção; os atributos Sentido e Flag passam a ter o valor especificado

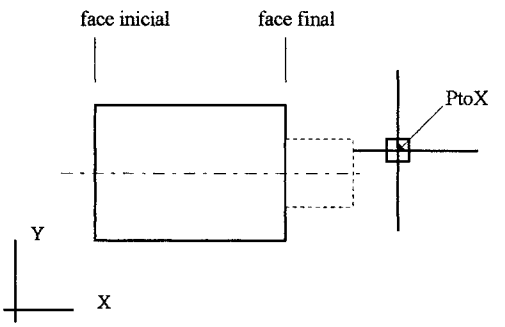


Figura 2.7 - Ponto de Localização de uma nova *feature* eixo.

Outro exemplo de regra pode ser apresentado como:

IF [(Nome = EIXOCILÍNDRICO) (Posição = EXTERNA) (Direção = AXIAL)
(PtoX > Face inicial) (PtoX < Face final)]

THEN [(Sentido = 0) (Flag = 1) (o eixo não pode ser inserido, pois já existe um eixo nessa posição (Figura 2.8))]

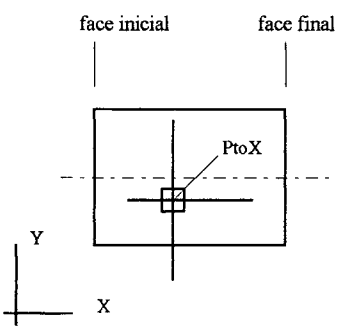


Figura 2.8- Ponto de localização incorreto de uma nova *feature* eixo.

Deve-se observar que as regras sempre devem dizer uma verdade, mesmo que, no segundo exemplo, a operação não possa ser efetuada; a conclusão, contudo, é verdadeira devido à situação.

Um tipo de conhecimento superficial utilizado é o conhecimento heurístico, que vem do grego e significa *descobrir*. A heurística não garante o sucesso, como no caso de um algoritmo; as regras heurísticas ou regras empíricas são resultado do conhecimento adquirido com a experiência, que podem ajudar na solução, mas não são uma garantia para se trabalhar. Porém, em muitas áreas, é o tipo de conhecimento disponível. Através da utilização da heurística, pode-se obter atalhos na resolução de um problema.

Um dos problemas enfrentados com o uso dos sistemas especialistas é que esses estão limitados ao domínio a que foram construídos. Um típico sistema especialista não pode generalizar seus conhecimentos usando analogia para raciocinar sobre uma nova situação, o que uma pessoa pode realizar. Também a transferência do conhecimento humano para um sistema tem sido muito difícil, pois é necessário, muitas vezes, criar uma codificação que represente o conhecimento.

2.10.3 - MECANISMO DE INFERÊNCIA

O mecanismo de inferência é o responsável pela pesquisa na base de conhecimento; é ele que infere sobre quais regras são satisfeitas pelos fatos ou objetos, que prioriza as regras e executa-as de acordo com a prioridade estabelecida. Para Krishnan (1995), o mecanismo de inferência de um sistema especialista possui três funções básicas: a) identificar a regra aplicável; b) resolver conflitos e c) executar a regra.

Dois métodos de inferência são utilizados normalmente: a) encadeamento para frente (*forward chaining*) e b) encadeamento para trás (*backward chaining*), como estratégias para a solução de problemas em sistemas especialistas. A utilização de uma das estratégias é função do tipo de aplicação a ser feita; se utilizado como simulador, recomenda-se o encadeamento para frente, sendo o raciocínio feito sobre os fatos e tendo como resultado uma conclusão a partir desses fatos. Por exemplo, ao desejar-se inserir um chanfro num eixo, o sistema especialista verifica as condições em que o eixo está com relação às outras *features* da peça, experimentando se o chanfro não interfere em nenhuma delas; não havendo interferência, a conclusão é positiva.

Nesse caso, o sistema pega o fato e, através do disparo de várias regras, chega a uma solução; somente então é que outras tarefas podem ser executadas. Isso pode ser observado no caso da inserção de um chanfro num eixo: primeiramente, ele verifica as condições como se estivesse simulando a existência do chanfro verificando o que acontece.

A utilização do encadeamento para trás, por sua vez, permite que, a partir de um fato conhecido, possam ser determinadas as causas ou simplesmente as hipóteses que o sustentam. Ele envolve o raciocínio no sentido inverso da hipótese, ou seja, uma potencial conclusão pode ser provada para o fato que dá suporte à hipótese. Por exemplo, se existe um chanfro aplicado num eixo, o sistema procura verificar as condições e dizer por quais razões o chanfro pode existir.

A escolha de como o mecanismo de inferência vai atuar depende do tipo de problema a ser resolvido. Assim, problemas de diagnósticos são mais bem resolvidos com encadeamento para trás, enquanto prognósticos, monitoramento e controle são mais bem feitos pelo encadeamento para frente.

2.11 - ANÁLISE BASEADA EM OBJETOS

Para que seja possível manusear informações num sistema computacional, um dos itens mais importantes é a forma como esses dados estão organizados, ou seja, como está modelada a informação de modo a representar um determinado modelo físico ao qual se pretende armazenar informações e também manipulá-las. Para isso, deve-se conhecer as relações existentes entre as diversas informações a serem modeladas, o que pode ser feito através de várias técnicas de análise, como análise funcional, diagramas de fluxo de dados, análise baseada em objetos, etc.

Segundo Coad (1992), a análise baseada em objetos utiliza os conceitos que se aprendem ainda no jardim de infância, ou seja, são conceitos relacionados com o mundo real, como objetos, atributos, todos e partes, classes e membros. Para Kim (1993), o paradigma de orientação para objeto é uma nova forma de abordagem para a solução de problemas, utilizando-se de modelos organizados em torno de conceitos do mundo real.

A análise baseada em objeto estabelece alguns princípios essenciais para administrar a complexidade do domínio de um problema e das responsabilidades do sistema, os quais são: abstração, encapsulamento, herança, associação, comunicação com mensagens, métodos de

organização, escalas e categorias de comportamento. Assim, o enfoque baseado em objetos pode ser equacionado de forma simples como:

Baseado em objetos = Classes e objetos + Herança + Comunicação com mensagens.

Por meio dessa modelagem, é feita uma ligação entre os dados e as funções que os manipulam, ou seja, combinam-se uma estrutura de dados e o seu comportamento numa única entidade.

Para Usher (1996), o método baseado em objetos requer que se identifique um conjunto de objetos do domínio do problema, expressando-se as operações do sistema como interações entre esses objetos. Ao invés de o sistema incluir algoritmos que cuidam dos passos individuais do processo, o programa é construído em torno de um conjunto de classes de objetos. Cada classe desse modela o comportamento de um objeto do mundo real, onde é definido pelo que o objeto é capaz de fazer.

No conceito de classes, as entidades representadas devem ser identificadas como objetos que pertencem a uma determinada classe (uma classe pode ser comparada a um protótipo, e os objetos, como entidades que obedecem a esse protótipo). Dentro de uma classe, têm-se os atributos e os métodos, os quais dão o comportamento do objeto através da manipulação dos atributos. Para Coad (1992), uma classe é a descrição de um ou mais objetos com um conjunto uniforme de atributos e funções, incluindo uma descrição de como criar novos objetos na classe. E um objeto é uma abstração de alguma coisa num domínio de problemas, exprimindo as capacidades de um sistema manter informações sobre ele, interagir com ele, ou ambos, um encapsulamento de valores de atributos e de suas funções exclusivas. Essa abstração corresponde a ignorar determinados aspectos de um assunto de modo a se concentar apenas nos principais.

Para M. Shaw, citado por Usher (1996), uma abstração pode ser definida como "uma simples descrição, ou especificação de um sistema que enfatiza alguns detalhes do sistema ou propriedades enquanto suprime outras". No projeto baseado em objetos, a abstração surge na definição da classe do objeto de um sistema. Esses sistemas combinam os dados e suas funções relacionadas para formar um objeto da classe. A essa ligação entre membros com um conjunto de procedimentos associados (chamadas funções-membros) usados para manipular os dados denomina-se *encapsulamento* (R.S. Wiener L.J. Pinson, citado por Usher (1996)).

O encapsulamento corresponde a esconder determinadas informações. Assim, as informações podem ser escondidas de uns objetos, mas não de outros. Com isso, pode ser definida a visibilidade, que corresponde a esconder detalhes que não contribuem com características especiais, o que vem reforçar a definição de encapsulamento. Para Usher (1996), o conceito de visibilidade limitada permite o desenvolvimento de um sistema no qual nenhuma parte é dependente de detalhes internos de outras partes do sistema. Para isso, cada objeto deve ter dois componentes: a) uma interface e b) uma implementação.

Segundo Kim (1993), o modelo de programação orientada para objeto oferece quatro formas adicionais na representação das informações: a) encapsulamento; b) generalização; c) inerência e d) polimorfismo. A generalização é uma abstração que permite que uma classe de um objeto seja considerada individualmente; a inerência permite a reutilização do comportamento e do código da classe na definição de uma nova classe, o que equivale à herança. E, por último, o polimorfismo, que significa a habilidade de assumir várias formas, isto é, uma determinada função com o mesmo nome pode existir em diferentes classes, assumindo formas diferentes de aplicação.

Como auxiliar na distinção das classes, pode ser adotada a identificação das estruturas do problema, com a utilização de Estruturas de Generalização-Especialização, nas quais, a partir da perspectiva de especialização, pode-se pensar em uma estrutura "é um" para buscar o próximo elemento menos especializado. Com base nessa estrutura, pode-se identificar a existência do mecanismo de herança.

Outro tipo de estrutura é a Estrutura Todo-Parte, em que a visão é da perspectiva do todo para as partes e pode ser pensada como "tem um", o que vem ao encontro da definição de composição de classes (maiores detalhes ver Coad, 1992 e Usher, 1996). Nessa, os atributos são identificados por substantivos e os métodos, pelos verbos.

Para obter uma modelagem adequada, são necessárias outras classes de modo que as informações sejam modeladas a partir de classes genéricas (como foi descrito anteriormente) chegando até classes mais especializadas (o que vem a compor uma família). Assim, pode-se ter como exemplo uma classe chamada *FEATURE*, que é considerada uma classe genérica:

FEATURE

Atributos:
Nome;
Posição;
Sentido;

Direção;
Ponto de Inserção.

Métodos:

Ler Ponto de Inserção;

Identificar Nome.

Considerando *FEATURE* como uma classe básica (classe genérica que não é instanciada), pode-se dizer que eixo é uma *feature* (ver item 3.5.1); logo, tem-se a classe EIXO que herda todos os atributos e métodos da classe *FEATURE*.

EIXO é uma *FEATURE*

Atributos:

Comprimento.

Como um eixo pode ser cilíndrico ou cônico, obtém-se uma classe mais especializada, que é EIXO CILÍNDRICO em que eixo cilíndrico "é um" EIXO. Assim, tem-se que a classe EIXO CILÍNDRICO (classe que é instanciada, donde se obtém um objeto) herda todas funções e atributos da classe EIXO. A família de classes que foram derivadas da classe-base forma uma hierarquia de herança. Dessa forma, o mecanismo de herança permite que dados que possuam alguma correspondência possam ser relacionados. E, se uma classe básica é modificada, todas as classes dela derivadas sofrem automaticamente as alterações.

EIXO CILÍNDRICO é um EIXO

Atributos:

Diâmetro.

Métodos

Desenhar Eixo Cilíndrico.

A composição é definida através da expressão "tem um" e pode ser mostrada na relação existente entre uma classe-peça e a classe eixo cilíndrico. Uma peça "tem um" eixo cilíndrico,

ou seja, é composta por um eixo cilíndrico; contudo, não se diz que uma peça "é um" eixo cilíndrico (pode ocorrer que uma peça seja composta por um único eixo cilíndrico ou cônico). Assim, pode-se obter uma classe complexa através da composição. Por exemplo, a classe PEÇA torna-se uma classe complexa devido à composição necessária à sua definição como: PEÇA "tem um" EIXO CILÍNDRICO, "tem um" MATERIAL, "tem um" CONJUNTO DE DIMENSÕES.

As classes descritas constituem, então, uma família, a qual inicia numa classe genérica, dita *classe básica*; à medida que são criadas novas classes por herança, essas se tornam mais especializadas na sua definição e mais complexas em função da ocorrência da composição com outras classes.

Para Kim (1993), o paradigma da orientação para objeto também providencia flexibilidade nos dados do modelo, no qual estruturas complexas como a de manufatura podem ser criadas e manipuladas facilmente. A representação por orientação para objeto permite expressar a complexidade dos dados de manufatura através da simples representação de classes, as quais encapsulam os dados e as funções de manufatura. Assim, as informações de manufatura são unificadas e consistentes. A hierarquia das classes providencia, então, uma base para a decomposição do sistema de manufatura.

O mesmo autor apresenta uma proposta de metodologia para modelar informações num sistema de manufatura, chamado OOMIS (*Object-Oriented Modeling Methodology for Manufacturing Information Systems*), que consiste de uma fase de análise e outra de projeto. Na fase de análise, o sistema de manufatura é decomposto em componentes funcionais para a definição do fluxo de informações; as funções de manufatura são descritas através de diagramas funcionais. Na fase de projeto, as informações resultantes da decomposição da fase de análise são transformadas para o modelo de orientação para objeto.

Em Wu (1995), encontra-se uma série de metodologias para análise orientada para objeto, das quais a principal proposta é criar uma ponte entre o sistema de análise e o sistema de projeto, ligando as abordagens funcional e de dados. Dentro das abordagens apresentadas, tem-se a abordagem feita por Booch citado por Wu (1995), GOOD (General Object-Oriented Development) e HOOD (Hierarquical O-O design), ambas as metodologias com muitos pontos em comum. O trabalho realizado por Wu (1995) utiliza-se do conceito de orientação para objetos para modelar um sistema de análise e projeto de operações de manufatura, envolvendo metodologias conhecidas e fazendo algumas adaptações a sua aplicação. A decomposição do

sistema tem sido feita através da abordagem *bottom-up* (das partes para o todo) para identificar as classes e objetos; secundariamente, o sistema é descrito em subsistemas e funções com que se caracteriza mais uma estrutura hierárquica e muito mais uma abordagem *top-down* (do todo para as partes).

Usher (1996) realiza uma revisão de conceito com relação à metodologia de orientação para objeto, citando o desenvolvimento de um modelo do produto baseado em objetos desenvolvido por Zhang et al., no qual classifica as informações do modelo do produto numa hierarquia composta por seis classes de objetos: produto, submontagem, peça, árvore CSG, sólido e *feature*. Cada classe de objeto possui informações gerais das propriedades do objeto, uma descrição dos componentes desse e suas relações com outros objetos.

LeBlanc e Fadel, citados por Usher (1996), utilizam-se da orientação para objeto na modelagem do produto com relação à análise do processo de projeto. Exploram essa metodologia de projeto de sistemas com o objetivo de proporcionar múltiplas visões do projeto para uso pelos vários departamentos de uma empresa.

Kusiak et al., citado por Usher (1996), utilizam a orientação para objetos para desenvolver ferramentas para projeto conceitual. No seu trabalho, procuram simular o caminho que o projetista percorre durante a fase conceitual do projeto, o qual está baseado nas definições de requisitos e funcionalidade, sendo aplicadas regras de produção para guiar a síntese de projeto.

2.11.1 - VANTAGENS DO USO DA ORIENTAÇÃO PARA OBJETO

Várias vantagens são atribuídas, na literatura, ao uso do paradigma de orientação para objeto, entre elas:

- proteção dos dados, visto que somente os atributos de uma determinada classe podem ser acessados por funções da mesma classe;

- modularidade, pois cada função é uma entidade separada e independente, que permite a reutilização de cada entidade sem levar em conta as suas relações com outras;

- a reutilização é melhorada pelo encapsulamento, pela herança e pela capacidade de envio de mensagens. A modularidade também permite o aumento da flexibilidade e da produtividade;

- conseqüentemente, se o sistema é modular, a sua manutenção torna-se muito mais fácil. Ao ser feita a implementação de uma classe, o restante do sistema não sofre alterações.

G. Booch, citado por Usher (1996), cita alguns benefícios do uso do método de orientação para objetos sobre a abordagem tradicional para projeto de sistemas complexos:

- a decomposição de resultados em pequenos sistemas;

- sistemas mais flexíveis em termos de realização de alteração e quanto à evolução do sistema;

- os sistemas são projetados para evoluir incrementalmente a partir de um pequeno sistema, fácil de criar, até um grande e complexo sistema;

- a decomposição dos resultados em módulos que possibilitam o desenvolvimento de outros sistemas, reduzindo os esforços em futuros desenvolvimentos.

2.12 - COTAGEM

Segundo Yeun (1988), na manufatura de peças mecânicas, cotagem e tolerâncias são dois atributos importantes usados para descrever as propriedades geométricas e funcionais de uma peça. A cotagem pressupõe, da parte de quem o faz, o conhecimento da função da peça a ser cotada, para que ela possa cumprir a função adequadamente. Logo, é através do conjunto montado que se pode verificar as dimensões a serem realmente controladas.

As cotas realmente importantes são aquelas que representam posições entre superfícies de contato. Assim, em peças rotacionais, encontram-se dois tipos de contatos característicos: o contato radial e o contato axial. Desse modo, o primeiro passo corresponde a determinar informações de montagem para, posteriormente, realizar a análise de cotagem, que será funcional.

Yeun (1988) mostra que, de acordo com as normas, existem quatro caminhos para se obter a cotagem de entidades numa representação:

- Linear: medida da distância entre planos paralelos e posicionamento de entidades, assim como centros ou pontos de curvas;

- Diametral : medidas de círculos ou esferas;

- Radial : medida de arcos ou esferas;

- Angular : medida de planos inclinados, semi-ângulos de cones em escareados.

Yeun (1988) propõe que a cotagem seja efetuada automaticamente, trabalhando com um modelador de sólidos CSG e utilizando uma única peça. Desse modelo CSG são, então, retiradas as informações dimensionais e é efetuada a respectiva cotagem baseada nas normas. O autor explicita que a cotagem automática é obtida do modelador de sólidos, que pode gerar uma cotagem adequada para um determinado sólido sem nenhuma especificação explícita de referência dimensional por parte do usuário.

Em termos de execução da cotagem de uma peça, basta a simples aplicação das regras básicas de cotagem; porém, do ponto de vista de um projeto, é necessário que a peça seja vista no seu contexto de aplicação, pois é aí que estão as superfícies de referência necessárias para se efetuar uma cotagem que represente a sua função, do contrário, a cotagem torna-se apenas didática, não representando a realidade.

Uma descrição da cotagem Diametral e Longitudinal (linear) será feita em seqüência.

2.12.1 - COTAGEM DIAMETRAL

Pode-se dizer que a realização da cotagem diametral é de extrema simplicidade, ou seja, somente há uma forma de referenciar essa dimensão numa peça. Não há relações funcionais que modifiquem a maneira de efetuar a cotagem, como ocorre com a cotagem longitudinal (linear). A diferença está em que pode haver cotas diametrais controladas ou não, as quais, como foi dito anteriormente, dependem das condições de montagem do conjunto.

2.12.2 - COTAGEM LONGITUDINAL

A necessidade de uma cotagem longitudinal das peças rotacionais é de fundamental importância para que as peças obtidas possam ser montadas adequadamente. Para isso, é necessário que a cotagem represente exatamente o funcionamento do conjunto no qual a peça será montada. Isso corresponde ao que se chama de cotagem funcional, ou seja, a cotagem em razão da funcionalidade do conjunto traduzido para a peça; em outras palavras, a peça é cotada em função do conjunto, não levando em conta o problema de fabricação.

Como na cotagem diametral, é de extrema importância a identificação dos contatos existentes entre as peças na montagem, que, no caso, são os contatos axiais, pois são eles que

ditam as posições das peças no espaço. Dessa forma, tais contatos devem ser traduzidos para a peça. Logo, a cotagem longitudinal pressupõe o conhecimento dos pontos de contato do conjunto, que, para a peça, serão chamados de *coordenadas de referência*, pois as dimensões serão referenciadas a elas. Isso pode ser facilmente observado nos exemplos que seguem.

Nos dois exemplos obtidos de Ropion (1974), é mostrada a importância da análise da montagem para se obtenham as cotas de projeto de cada peça. Nele, uma mesma peça é usada em aplicações diferentes, o que resulta numa cotagem diferente.

Na Figura 2.9, a peça **A** é montada na peça **D**, e a superfície **F2** deve estar em contato com **D**; a posição da superfície **F1** pode ser qualquer uma, mas não deve interferir com a peça **E**; deve existir uma certa folga entre a superfície **F3** e o fundo do alojamento da peça **D**; por último, a superfície **F4** não deve ultrapassar a peça **D**.

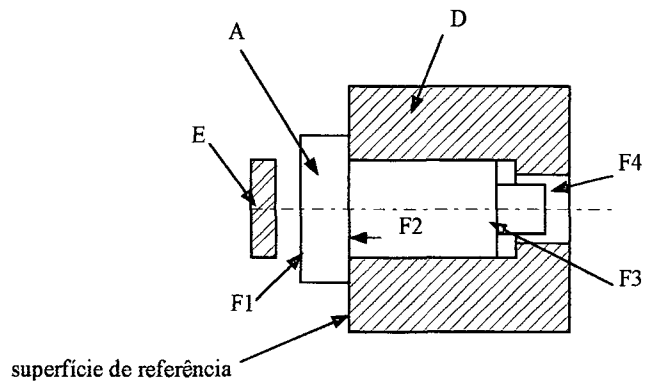


Figura 2.9 - Condições de funcionamento de um conjunto.

Tem-se assim, a descrição do funcionamento do respectivo conjunto, que, através da análise da superfície de contato e tomando-a como *coordenada de referência,* resulta no seguinte (Figura 2.10):

- Cota **2-1**: Estando a face **F2** constantemente apoiada em **D**, basta limitar a posição da superfície **F1** em relação a **F2**, ligando as duas superfícies;

- Cota **2-2**: Esta dimensão determina o comprimento da parte cilíndrica, limitada pela superfície **F3**, em função da profundidade do alojamento da peça **D** e em relação à superfície de contato **F2** que posiciona a peça **A**;

- Cota **2-3**: Da mesma forma, a superfície **F4** deve ser ligada à superfície de contato **F2**, superfície que une a peça **A** a **D**.

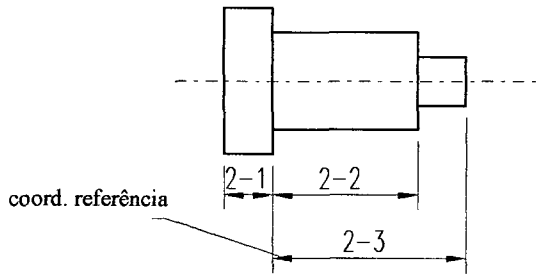


Figura 2.10 - Cotagem linear resultante.

Como no exemplo da Figura 2.10, é utilizada a *coordenada de referência* para a base da cotagem. Nesse caso, há somente uma referência, portanto, a cotagem torna-se simples, tendo-se cotas à esquerda e à direita da referência.

No exemplo a seguir (Figura 2.11), a peça **A** é ajustada no bloco **B** e restringida axialmente pela tampa **C**; a cabeça da peça **A** (parte cilíndrica com o maior diâmetro) deve possuir um ligeiro jogo no seu alojamento. Portanto, o contato é possível em **F1** ou **F2**; a superfície **F3** deve localizar-se sempre internamente ao bloco **B**; a superfície **F4** deve sobressair do bloco **B** de um certo comprimento para o emprego da peça, cujo valor mínimo é fixado.

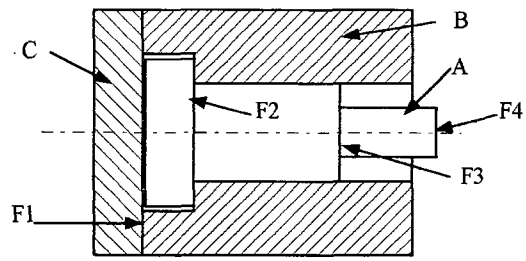


Figura 2.11 - Conjunto montado.

A cotagem resultante é então (Figura 2.12):

- Cota **1-1**: Este é o comprimento da cabeça que deve ser comparado com a profundidade do alojamento do bloco **B** para permitir o ligeiro jogo necessário;

- Cota **1-2**: A superfície **F3** não deve ultrapassar o bloco; é a superfície **F2** no fundo do alojamento que impede a saída da superfície **F3** do bloco. Portanto, limita-se a posição da superfície **F3** por uma cota que a liga à superfície **F2**;

- Cota **1-3**: A superfície **F4** deve ultrapassar o bloco B por uma quantidade fixada, a qual será mínima se a superfície **F1** estiver em contato com a cobertura; portanto, deve-se considerar o caso do contato em **F1**, o que implica ligar a superfície **F4** à superfície **F1**.

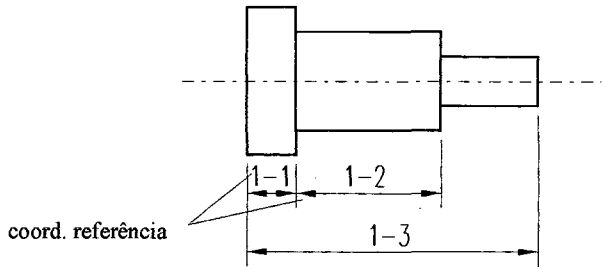


Figura 2.12 - Cotagem funcional.

2.13 - TOLERÂNCIAS

Segundo Zhang (1995), as tolerâncias podem ser divididas em duas categorias: uma é aplicada em relação às dimensões da forma como cilíndrica, esférica, ou a um conjunto de superfícies planas paralelas, cada uma associada com a dimensão, sendo a variação definida pela qual uma dimensão específica pode ser aceita; a outra é a tolerância geométrica, aplicada em geral para controlar a forma como perfil, orientação, localização e rugosidade.

Para Weill et al. (1988), o projetista tem que observar um certo número de normas e regras técnicas relativas às tolerâncias geométricas e dimensionais, as quais são necessárias para produzir um projeto que não deve ser somente consistente com a função da peça, mas também com as restrições de manufatura, limitações de montagem e inspeção. Essas condições, mais tarde, determinam essencialmente a economia da indústria na fabricação das peças.

No sistema TVCAPP (ABDOU, 1993), as tolerâncias são divididas em duas áreas básicas: tolerâncias de projeto, as quais são definidas no estágio de projeto para diminuir os custos de manufatura; e tolerâncias de controle, que representam a garantia de existirem máquinas e ferramentas para produzir a peça dentro das tolerâncias especificadas.

Faingmelermt et al., citados por Zhang (1992), desenvolveram um software para efetuar dimensionamento e incluir tolerâncias, utilizado em microcomputadores e que analisa as categorias de tolerâncias relevantes à manufatura, como tolerâncias de posição e tolerâncias de usinagem, propondo estratégias para otimizar as tolerâncias em relação aos requisitos funcionais e a capacidade dos equipamentos. O programa é utilizado pelo processista quando da escolha do melhor plano de processos.

Weill et al. (1988) propõem a implementação do uso de tolerâncias por função, em que as tolerâncias são o suporte para o projeto, manufatura, inspeção e montagem, abrangendo todo

o processo de manufatura. Por função é generalizada a aptidão para montagem de peças, mas também, em muitos casos, aptidão para o funcionamento de um sistema mecânico, o qual é dependente de pequenas forças de atrito, precisão geométrica, exatidão cinemática, e assim por diante. Contudo, o uso de tolerâncias por função implica o desenvolvimento de um sistema de tolerâncias que necessita dos seguintes dados:

- dimensões da peça;
- geometria da peça;
- requisitos funcionais;
- requisitos de montagem;
- requisitos de manufatura;
- requisitos de inspeção.

O principal problema de tolerâncias por função é, obviamente, a falta de conhecimento referente às relações entre a função e a folga, acabamento e função, forma geométrica e função (Weill, 1988).

Panchal (1992) desenvolveu o sistema CATAP para efetuar a cotagem e a inclusão de tolerâncias no projeto. As informações das tolerâncias estão armazenadas em um banco de dados e o usuário, interativamente, faz a seleção. A localização das cotas a serem controladas é feita pela interpretação de arquivos DXF, nos quais há arquivos que representam cada peça do conjunto e um que representa o conjunto montado. A interpretação das linhas, arcos e círculos propicia a extração das *features*, identificando, assim, o que é um eixo ou furo; com a comparação da posição das linhas, o software identifica as peças montadas e que serão toleradas.

2.13.1 - TOLERÂNCIAS DIMENSIONAIS

Para Zhang (1995), a análise de tolerâncias pode ser dividida em duas categorias: a) a análise de tolerâncias para a montagem e b) a análise de tolerâncias para a fabricação. Vários métodos são utilizados para tratar com as duas categorias de tolerâncias, que se baseiam na teoria de cadeias de tolerâncias desenvolvidas para a montagem das peças e as tolerâncias dimensionais (D&T).

As cadeias e tolerâncias também são utilizadas na fabricação para o redimensionamento da peça em função do processo de fabricação, pois sabe-se que nem sempre a cotagem de projeto (cotagem funcional) corresponde às cotas necessárias à fabricação da peça.

2.13.2 - CADEIA DE TOLERÂNCIAS

Uma cadeia de tolerâncias corresponde a um caminho fechado de cotas, em que a inter-relação entre as várias dimensões pode resultar em erros de montagem, visto que, mesmo que as várias peças envolvidas estejam com as cotas corretas, a soma delas produz um erro.

Ngoi (1996) apresenta o exemplo de uma caixa de engrenagens, onde são analisadas a influência da cadeia de tolerâncias gerada e a equação fundamental da cadeia para que se possa controlar a folga numa montagem. Na Figura 2.13, estão representadas a caixa de engrenagens (a) e a cadeia de tolerâncias resultante (b) a ser analisada com relação à folga "f" especificada.

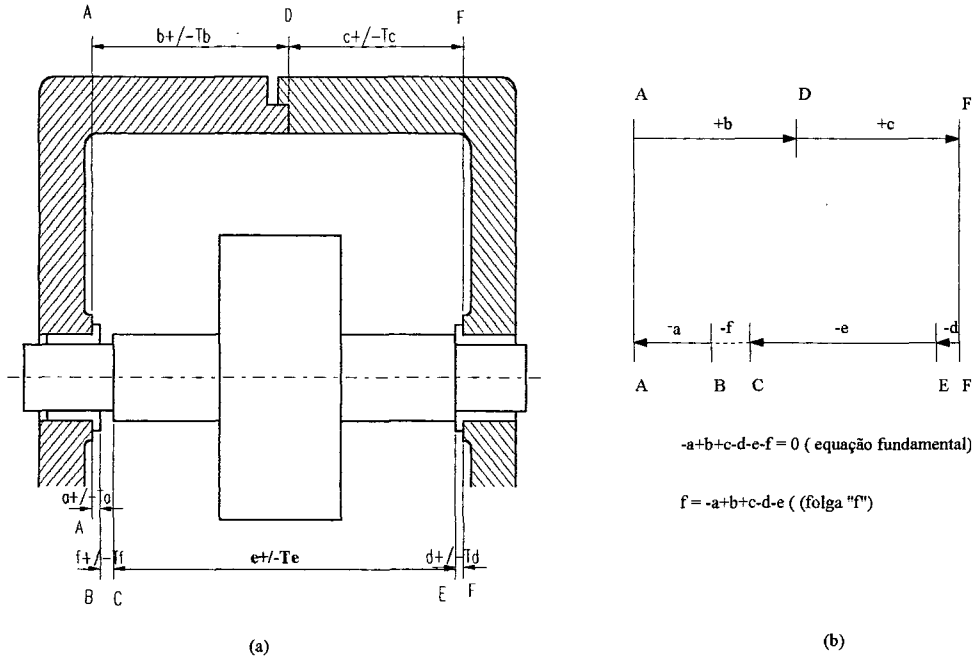


Figura 2.13 - Cadeia de tolerâncias num conjunto (NGOI, 1996).

Scott e Gabrielle, citados por Ngoi (1996), apresentam um programa para análise de tolerâncias em peças e subconjuntos, no qual utilizam um esquema onde os seis graus de liberdade de movimento da peça são restringidos. As informações a respeito das cotas distribuídas na peça são obtidas através do programa. No entanto, uma das limitações do método é que, para cada montagem diferente, devem ser formuladas restrições separadamente.

Ngoi (1996) considera, em seu trabalho de análise de cadeias de tolerâncias, que as superfícies de contato entre as peças sejam conhecidas. A partir dessas informações, ele obtém uma matriz que representa as relações das dimensões na cadeia de tolerâncias. Com o uso da

matriz e de um algoritmo, são obtidas equações lineares em termos das cotas descritas que formam a cadeia, sendo então realizadas as análises.

Segundo Wang (1990), a análise de tolerâncias pode ser classificada em duas categorias: análise de tolerâncias e síntese de tolerâncias. A análise de tolerâncias investiga os efeitos individuais de uma dimensão numa soma com outras; a síntese de tolerâncias determina dimensões individuais de acordo com a soma das cotas especificadas. Para ambos os métodos, é necessário conhecer as relações entre a soma das cotas individuais e a cota individual, a qual é conhecida como equação fundamental e descreve a cadeia de tolerâncias.

Wang (1990) realiza a geração de cadeias de tolerâncias a partir das informações de conjunto, representando, para isso,uma estrutura de dados onde a montagem está representada no topo da estrutura, a qual aponta para instâncias que são nós intermediários, os quais, por sua vez, apontam para peças ou subconjuntos. Cada instância pode apontar para outra instância. O programa desenvolvido funciona interativamente, e o usuário cria peças pela especificação de combinações de *features* e suas inter-relações; esse também define as condições de combinação para cada nova instância criada e estabelece as conexões entre peças e subconjuntos. Após a modelagem da montagem, pode ser obtida a cadeia de tolerâncias especificada em um arquivo.

2.13.3 - REPRESENTAÇÃO DE TOLERÂNCIAS

De acordo com Kulkarni (1996), com o surgimento dos modeladores de sólidos, têm sido criados novos problemas para a representação de tolerâncias, comunicação e o seu uso. O autor classifica duas categorias de abordagens para resolver tais problemas, que são: teorias matemáticas de tolerâncias e estruturas de representação para tolerâncias.

Nas teorias matemáticas que procuram estabelecer bases matemáticas com a intenção de unificar os diferentes tipos de tolerâncias para os modeladores de sólidos, têm-se os trabalhos de Requicha (1983 e 1984), citado por Kulkarni (1996), zona de tolerâncias baseada em classes variacionais, e de Turner e Wozny (1988), citados por Kulkarni (1996), tolerâncias baseadas em espaços vetoriais.

Na linha de estruturas de representação de tolerâncias, Requicha e Chan (1986) citado por Kulkarni (1996), têm procurado implementar o modelo de Requicha, utilizando um modelador "CSG". Johson (1985), citado por Kulkarni (1996), tem se utilizado de um modelador "B-rep" para representar tolerâncias de tamanho e posição.

Kulkarni (1996) utiliza-se de um modelador "B-rep", donde são extraídas as informações através de técnicas de reconhecimento para, então, montar um grafo inter-relacionando tolerâncias e dimensões das entidades que compõem a peça.

Roy (1993) desenvolveu um protótipo para representação de tolerâncias, utilizando-se de de uma entrada de dados para um modelador "CSG" e da avaliação num "B-rep", necessitando, para isso, de um mecanismo de ligação entre ambos para cada fase de desenvolvimento. Utiliza-se do modelador "CSG" para introduzir informações a respeito da criação da *feature* e informar as relações espaciais entre as *features*, como a localização e a orientação entre elas. No modelador "B-rep", são introduzidas as informações a respeito das tolerâncias geométricas, onde é possível relacionar as faces individuais da *feature*.

2.13.4 - TOLERÂNCIAS GEOMÉTRICAS

Gao (1996) apresenta uma tabela oriunda da "Britsh Standard BS 308", a qual relaciona as informações com relação a cada tipo de tolerância, tanto as dimensionais como as geométricas (Figura 2.14).

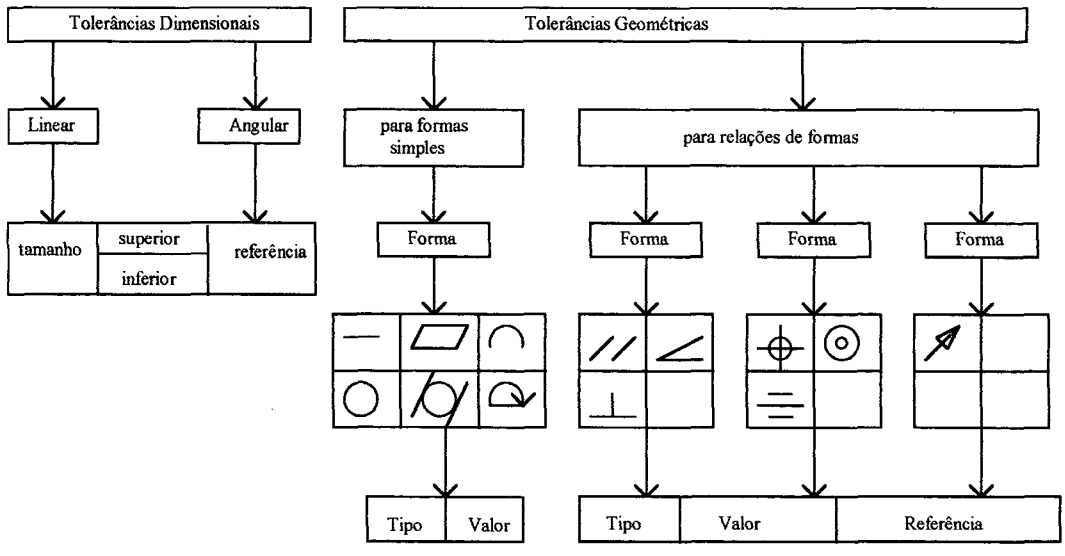


Figura 2.14 - Tabela representativa das tolerâncias geométricas
(Britsh Standard BS 308 em GAO, 1996).

Jasthi (1994) divide a representação das tolerâncias no sistema TURBO-MODEL em tolerâncias intra-*features* e inter-*features*. As tolerâncias intra-*features* são aquelas que

dependem de uma única *feature*, sendo freqüentemente aplicadas numa *feature* simples ou em porções de uma *feature*. Normalmente, são a retilineidade, a circularidade e a cilindricidade.

As tolerâncias inter-*features* controlam uma *feature* em relação à outra *feature*; também podem ser definidas como *features* relacionais. Das duas *features* relacionais, uma pode ser nomeada de *feature* de referência (REF) e a outra, de *feature* tolerada (TOF). Assim REF e TOF são ligadas por um tipo de tolerância denominada de relação (REL) e que pode relatar uma quantidade (QNT) que pode ser especificada por dois valores;:um superior e outro inferior para a mesma tolerância. No trabalho de Jasthi (1994) para componentes rotacionais, são aplicadas a batida radial, concentricidade e perpendicularidade, sendo também incluídos nessa categoria os comprimentos.

Zhang (1995) propõe um algoritmo para realizar a análise de tolerâncias para um sistema CAPP, levando em conta as tolerâncias geométricas e dimensionais para aplicação em tornos NC. Ele discute o problema da peça quando da realização de mais de uma fixação na máquina e a forma de definir as superfícies a serem usinadas em cada fixação em função das tolerâncias geométricas. Para isso, identifica direções das superfícies a serem usinadas, estabelecendo a prioridade de cada uma em função das tolerâncias geométricas, o que gera a seqüência de operações.

2.14 - ANÁLISE DE MONTAGEM

Para Mascle (1994), o uso de computadores na análise de montagem é mais difícil do que com relação à manufatura de peça. Os problemas principais são o baixo nível de padronização das operações de montagem em contraste com a manufatura, na qual operações como torneamento e fresamento são conhecidas há muito tempo.

Segundo Gu (1995), muitos esforços de investigação estão sendo realizados com relação à montagem automática por robôs, porém, na verdade, devido à complexidade da montagem dos produtos mecânicos, esse desenvolvimento tem sido lento se comparado com a montagem de componentes eletrônicos.

Conforme Ko (1987), para gerar procedimentos de montagem automaticamente, há três problemas a serem considerados:

1) como descrever uma montagem num computador;

2) como estruturar os componentes hierarquicamente;

3) como gerar procedimentos de montagem de uma árvore hierárquica de componentes.

De acordo com Shah et al. (1994), o planejamento da montagem tem recebido muita investigação, mas muito menos atenção do que a dada à fabricação de peças. O planejamento de montagem pode ser dividido, aproximadamente, nas seguintes fases:

1 - Seleção de métodos de montagem: em que a intenção é reconhecer o método mais conveniente de montagem para o produto, levando em consideração o tipo de sistema de montagem que será utilizado;

2 - Planejamento da seqüência de montagem: em que a intenção é determinar a seqüência de operações de montagem que podem ser implementadas com a escolha completa do método de montagem;

3 - Planejamento das operações de montagem: em que a ênfase é dada ao detalhamento dos passos de montagem individual, assim como direção de acesso, combinação de movimentos, aplicação de encaixe, etc.

É geralmente reconhecido que o processo de montagem possui pouca qualidade, baixa eficiência e alto custo, os quais estão naturalmente inter-relacionados. Em razão disso, resultou a técnica de projeto para montagem DFA (*Design for Assembly*), na qual o processo de produção mais econômico é selecionado durante o estágio de projeto e adaptado para as características do método de montagem escolhido.

O planejamento de seqüências de montagem na literatura é direcionado, em geral, para problemas de planejamento de montagem por robôs e planejamento da inspeção. Dini (1992), por exemplo, propõe a análise dos subconjuntos e seqüência de montagem baseadas no modelo matemático do produto, obtido através da definição de três matrizes: matriz de interferência, de contato e de conexão. Os possíveis subconjuntos são detectados automaticamente quando satisfazem algumas condições matemáticas aplicadas a essas matrizes. No trabalho, porém, não é revelada a origem dessas matrizes, nem a forma como são obtidas para que as respectivas análises possam ser efetuadas

Para Bronsvoort (1994), a tarefa de planejamento da montagem pode ser dividida em diferentes estágios como:

1) decomposição em subconjuntos, sendo o mais conveniente num processo inicial decompor as montagens em diversos subconjuntos, os quais podem ser determinados com base na decomposição funcional, ou com base nas propriedades de conectividade, que descrevem a interação entre as peças e como estão conectadas umas com as outras. Normalmente, a estrutura

de dados para representar as inter-relações entre peças é um contato, conexão ou grafo de ligação. Elementos adicionais são:

1) a estabilidade geométrica e a uniformidade na direção de montagem e tipo de conexão;

2) geração de seqüências de montagem, na qual, a partir da definição de subconjuntos, possíveis seqüências de montagem podem ser geradas, apesar de somente algumas serem possíveis, visto que a configuração geométrica impõe uma ordenação de montagem;

3) plano, sendo que cada alternativa de seqüência de montagem oferece um plano alternativo; assim, o objetivo de um plano é analisar os diferentes custos envolvidos no que diz respeito a ferramentas, transporte e armazenagem, permitindo verificar as melhores condições;

4) plano de movimento, que é o passo final onde são geradas as instruções para os mecanismos da célula de montagem, como robôs, esteiras, etc.

Bronsvoort (1994) apresenta a proposta de duas classes de *features* a serem utilizadas, as quais são a *feature* conexão e a *feature* manuseio. Uma *feature* conexão descreve o caminho de conexão das peças: direção principal, trajetória de montagem, ferramentas requeridas, operações, graus de liberdade após a montagem inicial, etc.; as *features* de manuseio, por sua vez, fornecem elementos para o plano de fixação, quais sejam as faces da peça que podem ser acessadas pelos fixadores, as faces que devem posicionar a peça nos dispositivos, etc.

Mascle (1994) divide o modelo do produto em três submodelos: o modelo físico de ligações, o de componentes geométricos e o de operações tecnológicas.

Weule (1989) menciona que, para a análise de montagem, desenvolveu o sistema KOMPASS, que utiliza a seqüência de desmontagem visando à análise de montagem, em que todos os detalhes do produto devem ser conhecidos. Isso pressupõe o conhecimento de todas as uniões das superfícies, isto é, das superfícies externas de cada peça, as quais têm contato com outras peças, ou folgas com outras peças, o que pode resultar em restrições para a montagem.

Para Weule (1989), os conhecimentos de dados geométricos na montagem e desmontagem não são suficientes. Como há o problema das folgas e interferências numa montagem, tais informações acabam influenciando o projeto da célula de montagem.

No sistema KOMPASS, é empregado um sistema especialista e técnicas heurísticas para filtrar as informações durante as interações do sistema. Nesse sistema, é possível descrever as características de montagem de uma peça padronizada, como parafusos, anéis, etc.; além disso, o

usuário pode intervir a qualquer momento no sistema para alterar a combinação de peças ou identificar um novo subconjunto, ou uma nova direção para uma peça.

Como Weule (1989) utiliza um modelo "B-rep" para efetuar as análises, a representação da montagem é feita pela análise desse modelo, no qual é difícil representar as folgas e interferências das montagens.

Bourjault, citado por Baldwin (1991), utiliza a ligação entre peças num diagrama de ligações para gerar uma lista de questões sim-não endereçadas ao projetista. O sim ou não correspondem, respectivamente, à capacidade ou incapacidade para montar ou desmontar uma peça, e as respostas a essas questões são processadas para gerar uma lista de possíveis seqüências.

De Fazio e Whitney, citados por Baldwin (1991), mudam as formas das perguntas para o usuário, as quais passam a ser questões que requerem raciocínio geométrico e entendimento por parte dele. Também mostram como representar uma seqüência de montagem através de rede de estados de montagem (nós) e movimentos de montagem (arcos).

Baldwin (1991) gera todas as possibilidades de seqüências de montagem, utilizando o método de conjunto de corte para encontrar e representar todas as restrições geométricas e de montagem como relações precedentes.

Eyada (1991) propõe um sistema de CAD inteligente, capaz de identificar uma montagem e especificar automaticamente tolerâncias. Esse método consiste na interpretação de arquivos DXF, onde as peças são definidas através de entidades-blocos, as quais são obtidas de uma livraria predefinida. A montagem é definida através do reconhecimento dessas entidades; a combinação das peças é identificada com base na comparação de linhas que compõem os diferentes blocos, as quais são analisadas sob diferentes condições. Assim: se a inclinação das linhas é a mesma; se as linhas possuem a mesma trajetória e se existe sobreposição das linhas. O sistema utiliza tais informações para especificar automaticamente as tolerâncias.

Panchal (1992) também se utiliza de método semelhante a Eyada (1992), realizando a extração das *features* a partir de arquivos DXF, havendo um arquivo para cada peça do conjunto e um para a representação do conjunto total. As tolerâncias são escolhidas interativamente.

2.14.1 - REPRESENTAÇÃO DA MONTAGEM

Para Anantha (1996), a representação e manipulação de montagens envolvem relações estruturais e espaciais entre peças individuais num nível mais alto de abstração do que para a representação de uma única peça. Assim, uma representação deve dar suporte a restrições de montagem de todas as peças envolvidas, seleção individual das peças na montagem, escolha da posição relativa das peças e manipulação da montagem como um todo.

Segundo Shah, citado por Anantha (1996), a representação também deve ser capaz de dar suporte à representação de *features* de forma e à combinação das *features* envolvidas nas conexões cinemáticas, determinação dos graus de liberdade das condições das combinações, checagem de interferências e determinação automática das violações das formas da peça.

Shah e Tadepalli, citados por Anantha (1996), propõem o modelamento de montagem por *features*, na qual se capturam as informações de um acoplamento montado entre duas peças combinando as *features*, permitindo, contudo, que a escolha seja feita depois de as peças terem sido montadas. O sistema é limitado somente ao modelamento estático, não suportando múltiplas relações de *features* entre duas peças.

Ko (1987) representa uma montagem através de uma árvore hierárquica e considera que uma montagem é dividida em várias submontagens, cada qual dividida em vários grupos, e cada grupo sendo composto de várias peças. Assim, desenvolve um algoritmo para definir a hierarquia da montagem; inicia a partir do modelo geométrico de todos os componentes primitivos de montagem, utilizando a combinação do grafo de relações de montagem dos componentes com a representação do conceito de *link* virtual (elemento de informação que produz a ligação entre os elementos envolvidos numa montagem).

2.14.2 - ANÁLISE DE SEQÜÊNCIA DE MONTAGEM

A grande ênfase que tem sido dada à montagem refere-se à seqüência das operações para a sua realização, visando principalmente à utilização da montagem automatizada através de robôs, o que pode ser observado em Gu (1995), Weule (1989), Mascle (1994).

Gu (1995) propõe a utilização de um sistema de CAD baseado em *features* como suporte para a extração das informações do produto, para, a seguir, realizar a análise de seqüência de

montagem para robôs. Divide o planejamento da montagem em quatro estágios, considerando a não-intervenção do usuário nas tarefas, os quais são os seguintes:

1) cria um grafo conectivo baseado na representação do produto por *features*;

2) decompõe a montagem em subgrupos usando um grafo conectivo;

3) gera uma seqüência de desmontagem de cada subgrupo formado no estágio 2;

4) combina as seqüências de desmontagem de cada subgrupo na seqüência total da desmontagem.

Para isso, esse autor desenvolve algoritmos para a realização dessa pesquisa, visto que considera que as informações do produto com relação à descrição da montagem estão à disposição, bastando apenas achar a melhor seqüência de montagem.

Mazouz et al. (1991) utilizaram-se de inteligência artificial e de um sistema baseado em conhecimento para gerar uma seqüência de montagem ótima sem a interferência do usuário. Eles definem as peças como externas e internas, considerando que existem ligações entre elas que permitem a montagem; para isso, desenvolveram uma série de regras buscando resolver as questões.

2.14.3 - REPRESENTAÇÃO DE *FEATURES* PARA MONTAGEM

Mascle (1994) descreve dois critérios para a representação de informações para montagem, os quais são denominados de "visões". Num deles adota a visão cinemática, que é mais direcionada para o trabalho com a utilização de robôs. Nesse caso, o tamanho, as tolerâncias e a orientação das superfícies dos objetos devem ser conhecidos precisamente, sendo cada superfície tratada em separado. A definição das características de montagem inclui:

- a geometria da peça (localização e orientação no espaço cartesiano, dimensões e direções de montagem);

- tolerâncias;

- relações entre a peça e o produto;

- reconhecimento de peças similares.

A outra visão de montagem utiliza a representação poliédrica de objetos e componentes, de forma que o módulo inclui quatro tipos de informações:

- informações geométricas (forma e dimensões dos componentes, posição relativa na montagem final);

- informações das peças (*features* de interesse de várias peças representam a montagem e são operadores especificados, como manuseio, base da peça ou prioridades iniciais e finais);

- informações finais da montagem (certas direções de montagem que podem ser eliminadas *a priori* pelo operador);

- informações topológicas (tipos de contatos, restrições de montagem).

Mascle (1994) descreve uma forma de representar e implementar *features* de montagem, utilizando a representação das informações através de uma pseudomatriz, que representa contatos, direções e sentidos de deslocamentos numa montagem. Assume a utilização de um código binário para representar o que chama de *semigraus* de liberdade, em que cada ligação é descrita através dessa matriz e de um conjunto de valores binários.

Mascle (1994) ainda define uma *feature* de montagem como uma interação entre dois elementos conectados por uma ligação orientada, a qual pode ser mecânica (contato físico) ou funcional (restrições geométricas ou folgas). Na descrição do seu modelo, afirma que não se pode pensar em montagem sem levar em conta as tolerâncias, as quais são um dado tecnológico a mais para a seleção de ajustes e, portanto, para a tomada de decisões ante as possibilidades e complexidade das montagens. O autor propõe o desenvolvimento de um modelador que não seja apenas geométrico, mas tecnológico, no qual todos os dados dos tipos de ligações existentes nas peças devam ser avaliados como o estado associado com cada elemento.

2.14.4 - RELAÇÕES DE MONTAGEM

Para descrever uma montagem, Ko e Lee citado por GU (1995) usam uma combinação de condições, como encosto, folga, ajuste e contato, para descrever as relações entre os componentes. A montagem é expressa pela combinação de grafos de componentes, tendo uma estrutura hierárquica gerada pelo desenvolvimento do algoritmo.

Lee e Gossard, citados por GU (1995), projetaram uma estrutura de dados hierárquica para representar a montagem numa base de dados, a qual foi dividida em duas partes: a primeira é a estrutura de dados usada para armazenar informações topológicas e geométricas de cada componente da montagem; na segunda, a estrutura de dados armazena informações sobre a forma como todos os componentes da montagem são conectados. Assim, uma estrutura em forma de árvore utilizando o conceito de *link* virtual foi criada para representar as relações entre componentes da montagem.

Gu (1995) representa uma montagem através de dois tipos de grafos, que são: a) as ligações e b) as relações de contato. Assim, o grafo de ligações fornece as relações de ligações entre as peças, informações essas de forma genérica, sem maiores detalhes sobre a forma como as peças estão relacionadas. O grafo de relações de contato descreve uma série de tipos de relações de montagem, como prensado, ajuste móvel, ajuste de chaveta, ajuste roscado, etc. As relações de montagem são, então, classificadas em duas classes: a) ajustes e b) contatos (Figura 2.15); ao final, todas as informações são convertidas em relações de contato.

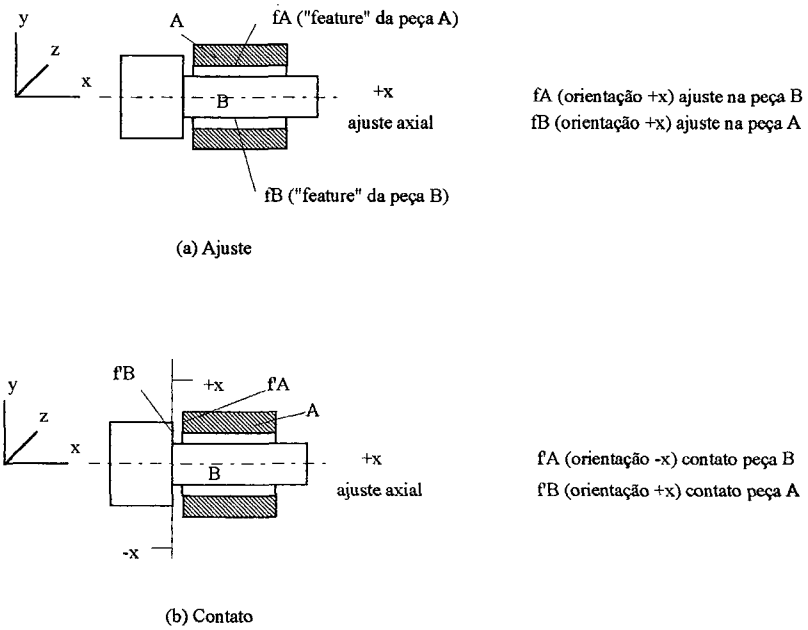


Figura 2.15- Exemplos de relações de contato e ajuste (GU, 1995).

Ko (1987) especifica as relações de montagem em quatro tipos, que são definidos como:

1) encosto, condição de fixação axial entre duas peças que possuem faces planas;

2) ajuste, quando a linha de centro de um cilindro sólido e de um furo estão colineares; essa condição permite o movimento de rotação e translação ao longo da linha de centro;

3) ajuste apertado, componentes que possuem as condições de ajuste, mas que não permitem o movimento de rotação nem de translação;

4) contato, condição de fixação que envolve dois pontos das faces de duas peças (não há o contato de toda a superfície.

2.15 - INSPEÇÃO DIMENSIONAL

O objetivo da inspeção dimensional é manter um controle sobre os resultados do processo empregado para a obtenção de uma forma, que deve estar dentro dos parâmetros previamente determinados no projeto. Essa também deve ser uma atividade integrada ao sistema de manufatura, ou seja, dentro da visão de integrar e automatizar as atividades, é necessário que as informações a serem utilizadas para a inspeção estejam à disposição dentro do modelo que representa o produto.

As informações disponíveis são, então, utilizadas para gerar o plano de inspeção da peça, o que, no caso, pode ocorrer através de procedimentos manuais, quando são informados os instrumentos a serem utilizados e quais os elementos da peça que devem ser verificados; ou, então, pela utilização de máquinas de medição por coordenadas, também executando as medições manualmente ou através da geração de um programa CN que execute a verificação automática.

Para Shah et al. (1994), inspecionar peças manufaturadas é vital para a qualidade, não somente descartando as peças ruins, mas, o mais importante, garantindo o controle total da qualidade das peças produzidas. Desse modo, o planejamento de inspeção deve ser tratado como parte integrante do planejamento da manufatura. Há, segundo os autores, muitos problemas em planejamento da inspeção, o qual deveria concentrar-se nas características do produto que têm maior influência na função correta em relação às outras peças, evitando os custos de inspeção das características menos significativas. Isso implica que o planejamento da inspeção seja profundamente dependente do modelamento do produto, assim como da representação das dimensões e tolerâncias, da representação das relações de montagem e também, da representação de funções do comportamento físico da forma.

Lin (1995) mostra que há dificuldades nessa integração completa, como na conexão das informações do sistema CAD para a máquina de medição, na extração das dimensões dos desenhos e na forma de transferir os dados adquiridos para os intrumentos de medição de modo correto. Utiliza, então, um sistema CAD, buscando desenvolver um conjunto de símbolos, visando introduzir as informações sobre tolerâncias geométricas, as quais podem sofrer manipulações, como alterar, excluir, etc. As informações introduzidas no sistema são retiradas de um desenho existente e construído no sistema CAD; após a extração desses dados de um arquivo no formato DXF, esses são gravados no formato ASCII para que funções auxiliares executem a interpretação e, assim, seja gerado o plano de inspeção da peça.

O trabalho feito por Lin (1995) mostra as dificuldades da introdução de dados para a análise nos processos de manufatura devido à falta de definição de um modelo do produto. O trabalho feito para a extração das informações do sistema CAD pode, em muito, ser facilitado com a utilização de um modelo do produto e das informações em que sejam modeladas como *features* no sistema.

Considera-se esse enfoque muito complexo no sentido da extração das informações e em razão de o arquivo do qual se extraem as informações ser apenas gráfico, não possuindo outros tipos de informações que auxiliem na interpretação da peça.

2.16 - INTERFACES PADRÕES

Para Usher (1996), o STEP (*Standard for the Exchange of Product Model Data*) consiste num esforço internacional para ajudar a definir uma norma de representação de dados do produto, cujo objetivo é criar um mecanismo neutro de troca de informações capaz de descrever um produto. Para isso, é necessária uma variedade de informações. O autor discute a definição e a implementação de uma biblioteca de classes para utilização em planejamento de processos, cuja abordagem é baseada no protocolo de aplicação AP 224, o que pode ser aplicado para representações por *features,* tanto para peças rotacionais como para prismáticas, incluindo dados geométricos e tecnológicos. Salienta, ainda, que o referido protocolo não contempla todas as informações necessárias para dar suporte à referida aplicação.

Segundo Eastman (1994), havia, inicialmente, uma grande expectativa de que o STEP poderia harmonizar e especificar a estrutura semântica utilizada nos sistemas CAD existentes e em aplicações de engenharia, o que, a princípio, ocorreu. Na realidade, o STEP envolveu-se em questões mais difíceis para as quais ainda não havia respostas, tendo sido modificada a orientação para melhorar os resultados deste trabalho, o que tem envolvido um grande número de pessoas no mundo.

2.17 - TÓPICOS ABORDADOS NA INVESTIGAÇÃO

Na bibliografia pesquisada, constatou-se a existência de inúmeros estudos que enfocam o planejamento do processo, planejamento de inspeção e o planejamento da montagem, cada um, no entanto, sendo desenvolvido num ambiente isolado.

Cada um desses desenvolvimentos leva à suposição de que os dados originam-se de algum sistema de projeto que possui os dados definidos na forma desejada, ou possui uma entrada de dados específicos no formato gráfico ou textual. Isso torna a idéia de aplicação de CIM um tanto difícil, já que, para a passagem de um sistema para outro, seria necessário refazer o conteúdo da informação. Com isso, verificou-se a necessidade de construir uma estrutura que permitisse trabalhar com o projeto e na qual os dados obtidos pudessem ser utilizados na manufatura, chegando-se, assim, a uma integração de sistemas. Aqui a manufatura compreende todas as atividades necessárias para a obtenção do produto final.

Outra limitação identificada foi a de que a maioria dos desenvolvimentos efetuados o foram para sistemas de grande porte, nos quais se utilizam estações de trabalho e outros equipamentos de alto custo.

No presente trabalho, aborda-se o desenvolvimento de um sistema CAD para dar suporte a *features* numa representação 2D, isso porque: 1) o domínio de peças utilizadas, no caso rotacionais, corresponde a um grande número de peças industriais; 2) não se dispunha de um modelador adequado disponível para tal tarefa; 3) tal representação pode ser feita facilmente utilizando-se microcomputadores.

A representação será baseada em *features* (*feature-based*), a qual atua em função de uma biblioteca de *features* preconcebida, que será definida em função do ambiente de atuação do sistema. A abordagem de projeto para a manufatura e montagem será feita em função de recomendações advindas da prática, sendo tais informações modeladas no sistema através de um sistema especialista.

Dentre os tópicos sobre os quais tem sido observada falta de investigação, incluem-se:

1) sistemas de auxílio à escolha de tolerâncias, visto que, nas investigações, tem-se feito muito em termos de análise e modelamento das tolerâncias, entretanto tem faltado apoio à escolha inicial dessas tolerâncias para que, posteriormente, possam ser analisadas e otimizadas. Por isso, abordar-se-á uma forma interativa de o usuário obter informações de tipos de acoplamentos utilizados, tanto aqueles amplamente conhecidos na bibliografia como aqueles definidos dentro da empresa. Utilizar-se-á um banco de dados para armazenar tais informações.

2) Outro ponto é com relação ao dimensionamento automático das peças, pois o que se tem encontrado é um dimensionamento das formas que apenas segue a norma, não levando em conta a funcionalidade da peça no conjunto, que é um outro item que deverá estar representado no modelo, ou seja, não apenas uma peça estará representada, mas um conjunto delas. Assim, poder-se-á realizar uma análise sobre esse conjunto, definindo o dimensionamento funcional (cotagem funcional), bem como as superfícies a serem usinadas e o respectivo acabamento.

Isso será feito através de um CAD inteligente, o qual, com base nas informações internas geradas pelos seus processos, pode retirar novas informações, identificando o que está representado. As análises a serem realizadas do ponto de vista de fabricação e montagem serão feitas com a utilização de um sistema especialista, em cuja base de conhecimento estarão representadas as informações em forma de regras, as quais serão resultantes do domínio de peças e de recomendações de prática de projeto, fabricação e montagem.

3) Com relação à montagem, as informações levantadas dão conta de que o principal enfoque tem sido com relação ao seqüenciamento de operações para a montagem com a utilização de robôs. Neste trabalho, o enfoque estará voltado para a análise do conjunto para que possa ser extraído o máximo de informações possíveis a respeito das peças montadas no conjunto.

Dessa análise, deverão resultar informações para a definição das superfícies a serem usinadas e toleradas, seus respectivos acabamentos superficiais e outras informações importantes para a definição completa do projeto.

Para a representação das informações num sistema computacional, será utilizada a programação orientada para objetos, a qual tem-se mostrado mais promissora para a representação de informações no formato de *features*. O modelo utilizado para as *features* será o da síntese de elementos volumétricos e geometria destrutiva, utilizando-se essa combinação com o conceito de *features* básicas e *features* modificadoras.

CAPÍTULO 3

SISTEMATIZAÇÃO DO PROCESSO DE PROJETO DE PEÇAS

3.1 - REQUISITOS DO MODELO

Alguns requisitos básicos foram definidos para que o modelo computacional possa suprir informações necessárias que atendam ao projeto e à manufatura, citando-se os seguintes:

- possibilitar a modelagem de um produto, para que possam ser representados conjuntos, subconjuntos e peças, cada um com as informações para a representação adequada do produto, de forma que esses dados possam ser utilizados para o planejamento do processo, montagem e inspeção;

- a partir da representação do produto, o sistema deve poder identificar as dimensões a serem controladas, bem como as superfícies a serem usinadas e, de forma interativa, escolher ou introduzir as respectivas tolerâncias, ajustes e acabamento superficial;

- a representação das informações deverá ser através de *features*, motivo pelo qual se optou pelo desenvolvimento de um modelo híbrido que utiliza a síntese de elementos volumétricos e geometria destrutiva (ver item 2.2) para representar uma peça de forma completa;

- a partir de uma biblioteca de *features*, devem ser obtidas outras *features* configuráveis, de forma que cada *feature* possa ser identificada pelo sistema independentemente;

- um sistema especialista será o responsável pela análise e interpretação das tarefas a serem executadas tanto no nível do CAD como da análise de parâmetros tecnológicos para manufatura e montagem;

- o sistema deve ter capacidade de análise, de forma que possa auxiliar o usuário na confecção de um projeto adequado à manufatura e montagem;

- a captura das intenções do projetista deve ser do modo mais natural possível, observando as particularidades do meio onde será aplicado o sistema;

- capacidade de auxiliar o usuário no dimensionamento das peças que compõem o conjunto e o subconjunto.

No presente trabalho, foi desenvolvido um sistema, chamado de "FeatCAD-2D", que visa incorporar todos os requisitos descritos.

3.2 - VISÃO GERAL DA ARQUITETURA DO SISTEMA

A escolha de uma arquitetura adequada possibilita que o sistema seja desenvolvido de forma consistente e robusta, porque, estando definidos os respectivos módulos, submódulos e comunicação entre eles, e cada um definido em sua função, é possível ampliar o sistema e introduzir novos elementos sem que a arquitetura do sistema tenha de ser alterada. Isso possibilitará uma flexibilidade que é desejada em qualquer sistema, mas que é de difícil obtenção.

A arquitetura proposta visa à definição de módulos independentes, que possam ser utilizados a partir dos módulos básicos, adicionando novos módulos ou submódulos, ou mesmo eliminando outros, sem que haja interferência nos módulos básicos ou entre módulos já existentes.

Como meio de comunicação entre o usuário e o sistema, ter-se-á uma interface gráfica que se comunica com dois módulos distintos: o módulo do *modelador* e o *módulo analítico* (Figura 3.1). O primeiro, o *modelador*, possui as funções que permitem que sejam instanciadas as *features* elementares e as *features* configuráveis (sejam estas últimas aquelas obtidas a partir da combinação das *features* elementares através do acesso à biblioteca de *features* (item 3.5.1)), como também que seja instanciada a estrutura do produto.

A *biblioteca de features* corresponde ao módulo onde estão descritas as definições das *features* disponíveis no sistema. No modelador, há também um submódulo onde está uma série de funções auxiliares que dão apoio ao modelador, como funções de gerenciamento do sistema, possibilitando a comunicação do sistema com o CAD.

Através do *modelador*, é possível: a) interagir com a *estrutura de dados* do produto com a qual serão instanciadas as *features*; b) através da interface de comunicação, utilizar a base de

conhecimento para modelar as *features*; c) consultar o banco de dados a respeito de manufatura e montagem, através da *interface de consulta.*

No *módulo de análise*, o qual também se comunica com a estrutura de dados, realizam-se diversos tipos de análise, feitas com o auxílio do *sistema especialista* com o qual se comunica através da interface de comunicação. Faz-se também consulta ao *banco de dados* de montagem e manufatura através da *interface de consulta.*

Na *estrutura de dados* do produto, que não corresponde à do CAD comercial, estão representadas as informações do produto, utilizando o formato de *features* como unidades básicas de informação. Essa estrutura de dados é o coração do sistema, pois permite armazenar, identificar e diferenciar os componentes de um produto; é nela que estão armazenadas todas as informações disponíveis no formato de *features* e que podem ser consultadas por qualquer um dos módulos e submódulos (Figura 3.1).

A *interface de comunicação* é a responsável pela comunicação entre o modelador e o módulo de análise com a base de conhecimento, o que é feito através do *sistema especialista.* Na *base de conhecimento,* são descritas as informações através de regras para que se possa realizar a análise das informações, as quais são consultadas pelos módulos e submódulos que das regras se utilizem para análise, como também para instanciação das *features*, identificação das relações de montagem, etc. Na base de dados de manufatura e montagem, são armazenadas informações que auxiliam nas tomadas de decisões, como dimensões de materiais padronizados, tipos de materiais padronizados, etc.

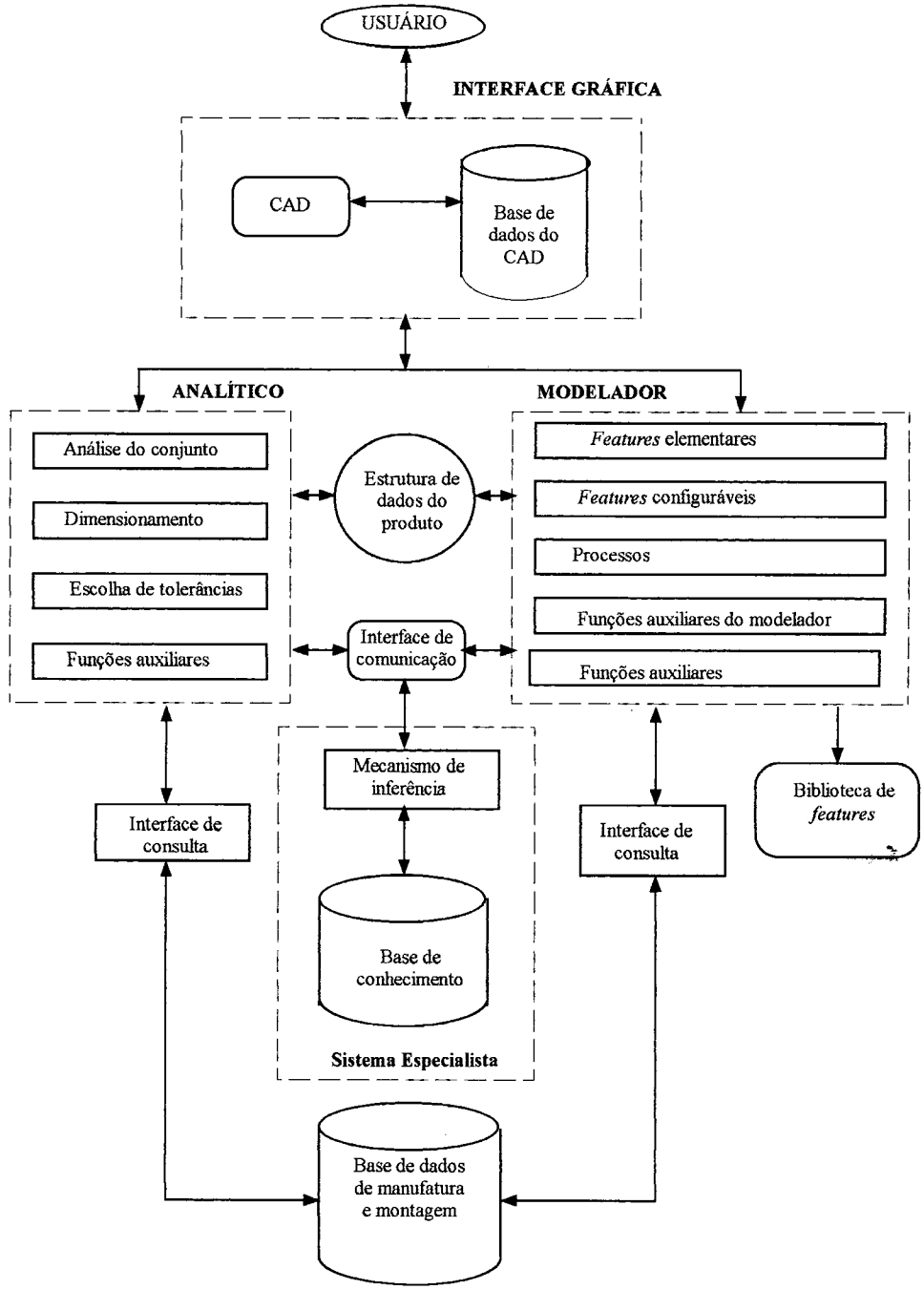


Figura 3.1 - Arquitetura do sistema.

3.2.1 - INTERFACE GRÁFICA

Como interface gráfica, será utilizado um sistema CAD comercial, o que permitirá a utilização de todas as suas funções gráficas, não havendo a necessidade de desenvolvê-las. A interface será responsável pela comunicação do sistema com o mundo exterior. É através dela

que: a) o usuário irá interagir com o sistema; b) se efetuará a passagem das informações para as estruturas internas; c) informações gráficas ou alfanuméricas retornarão ao usuário.

O uso de uma interface gráfica possibilita a visualização das informações introduzidas no sistema, que será no formato gráfico tradicionalmente utilizado nos meios de engenharia, no caso representação em 2D. Além disso, essa representação é considerada suficiente para representar o domínio escolhido, o das peças rotacionais. Dessa forma, torna-se mais amigável a interação usuário-computador.

Como a interface gráfica já possui integrada uma base de dados gráfica, essa servirá de ponte entre as imagens representadas pelo CAD e as informações no formato de *features*. É nessa base de dados que estarão representadas as informações a respeito dos desenhos do sistema FeatCAD-2D e que permitirão a aplicação das operações básicas do sistema CAD utilizado.

A utilização de uma interface gráfica é indispensável para que o sistema se torne amigável, pois a utilização da entrada de dados via texto, o que alguns sistemas experimentais têm utilizado, não é bem vista nos meios industriais onde já é consagrada a representação gráfica. Além do mais, implicaria duplicação do trabalho na representação do projeto.

3.2.2 - MODELADOR

É o módulo responsável pelas operações de instanciação[1] desde uma *feature* elementar até o produto como um todo. Nesse módulo, estão as operações básicas necessárias, como a instanciação, exclusão, alteração, gravação, etc.

O modelador possui a responsabilidade de representar graficamente as informações geradas pelo sistema e introduzidas pelo usuário, passando-as para a estrutura de dados do produto de modo que sejam armazenadas no formato específico, sendo apresentadas ao usuário através da interface gráfica. O modelador deve conectar-se à base de conhecimento e realizar as verificações quanto à consistência de uma determinada *feature* a ser inserida dentro de um contexto antes de efetuar a instanciação.

[1] Operação responsável pela inserção de uma *feature* no sistema

3.2.2.1 - MODELADOR DE *FEATURES*

É o submódulo onde estarão contidas as funções que permitem a instanciação das *features* elementares (item 3.5.1) descritas na biblioteca de *features*, às quais o usuário terá acesso direto para sua instanciação, mas não para a sua manipulação interna. É através desse módulo que o usuário escolherá a *feature* a ser instanciada com seus atributos para a representação.

Este submódulo somente será manuseado internamente quando da criação de uma nova *feature* na biblioteca, o que obrigará à criação de funções para a manipulação da mesma. Pode-se dizer que somente um *superusuário* terá acesso a ela.

3.2.2.2 - PROCESSOS DE EXECUÇÃO

Este submódulo é o responsável pelas operações de atualização das informações quando do acréscimo (inserção) de uma *feature* na biblioteca. Como as operações de inserção para cada *feature* são sempre as mesmas, isto é, uma *feature* eixo, quando inserida, possui um determinado sentido, posição, coordenadas de localização e outras informações, essas devem ser introduzidas ou atualizadas na estrutura de dados, operações que não mudam qualquer que seja a aplicação.

Isso é válido para todas as *features* elementares existentes na biblioteca. Assim, cada *feature* possui um conjunto de procedimentos que não se alteram nunca, independentemente das outras *features* existentes na biblioteca e de outras que venham a ser criadas. Cada vez que for criada uma *feature* na biblioteca (inserção de uma nova *feature*), um conjunto de procedimentos para instanciação e atualização das informações na estrutura de dados deverá ser introduzido no módulo de processos (acrescentar uma nova função para a *feature* específica).

3.2.2.3 - FUNÇÕES AUXILIARES

Dentro deste submódulo estão incluídas as funções que permitem ao usuário utilizar recursos que viabilizam a modelagem das informações, criação, leitura e gravação do modelo do

produto representado. Assim, em nível de conjunto[2], fazem-se necessárias funções para sua criação, exclusão e alteração dos respectivos dados; o mesmo se aplica em nível de subconjunto e peça. Mais especificamente em nível de peça, são necessárias outras funções, como de translação, rotação, escalonamento e cópia.

Em nível de conjunto e subconjunto, as funções de criação e alteração apenas manuseiam dados em nível de informação, não atuando em nível gráfico; já a função de exclusão envolve tanto informações em nível de dados como em nível gráfico, pois envolve excluir tudo o que está abaixo do respectivo nível. Essas são funções que atuam de modo específico de acordo com o tipo de dado a ser manuseado.

Foram implementadas funções auxiliares tipo sistema de navegação, as quais permitem passar de um subconjunto para outro, de uma peça para outra e para outros subníveis de informação existentes em cada um. Essa função contém um procedimento de pesquisa que localiza na estrutura de dados a *feature* procurada, no caso subconjunto ou peça. Esse procedimento de pesquisa também é utilizado por outras funções para a localização das informações na estrutura de dados.

Em nível de representação das *features* em si, são necessárias funções de criação, exclusão, alteração e identificação. A função de criação pode ser subdividida em outras funções em razão de uma aplicação específica, como criar *features* elementares, de alto nível padronizadas ou de alto nível padrão (item 3.5.1). Assim, é necessária uma função de criação de *features* que se comunica diretamente com o usuário, que lhe permite instanciar uma *feature* da biblioteca e informar os parâmetros dessa. Uma *feature* também pode ser criada a partir de parâmetros internos do sistema, como, por exemplo, de dados advindos do banco de dados, função essa utilizada para a instanciação das *features* de alto nível e também para as *features* compostas (item 3.5.1). A última forma corresponde à função que cria uma nova *feature* a partir de uma *feature* criada pelo usuário através da interface gráfica, e essas são chamadas de *features* de alto nível padrão (item 3.5.1). Com relação à exclusão, alteração e identificação, essas funções serão válidas para todas as situações em que tenham sido configuradas em função das *features* descritas na biblioteca.

Existe ainda um grupo de funções que estão num nível mais baixo, que permitem executar operações específicas de apoio às funções auxiliares principais. Entre essas, incluem-se, por exemplo, funções que identificam características específicas de uma determinada *feature*.

[2] Representação que reúne as informações gerais a respeito do produto (ver item 3.4.2).

Outro grupo de funções auxiliares faz a comunicação da estrutura de dados do produto com a estrutura de dados do CAD, o que ocorre de forma transparente ao usuário.

Com relação à inexistência de operações de translação e rotação sobre *features* no sistema desenvolvido, isso ocorre porque, na conceituação do sistema, tais operações somente são permissíveis sobre uma peça, aplicando a todas as *features* envolvidas, respeitando o seu contexto de aplicação.

Ainda em apoio a funções auxiliares, como criar, excluir e alterar, há funções que realizam a consulta ao sistema especialista para validar ou não a respectiva operação.

3.2.2.4 - BIBLIOTECA DE *FEATURES*

A biblioteca de *features* é o módulo que contém todas as definições dos atributos das *features* e funções básicas de manipulação para uma *feature*. É neste módulo que está definida a estrutura do produto, com seus atributos e funções básicas de manipulação que interagem com as *features*. Também estão definidas as ligações que devem existir entre cada *feature* quando da sua instanciação. Cada nova *feature* elementar a ser criada deve ser introduzida neste módulo (ver item 3.5.1), no qual as *features* estão descritas de acordo com as respectivas classes (ver item 3.6).

3.2.3 - MÓDULO ANALÍTICO

O módulo analítico é responsável pela reunião de todos os submódulos, que utilizam apenas a estrutura de dados como meio de obter informações da estrutura, retornando informações a ela em função dessas análises. Este módulo não possibilita operações de modificação da estrutura de dados, mas apenas de preenchimento ou alteração de suas informações.

Dentro deste módulo, podem ser desenvolvidas funções como:

1 - Análise do conjunto;

2 - Cotagem;

3 - Escolha de Tolerâncias;

4 - Identificação do Acabamento Superficial;

5 - Análise de Tolerâncias;

6 - Análise de manufatura específica;

7 - Análise de montagem;

8 - Geração de Planos de Processos;

9 - Geração de Planos de Montagem;

10 - Geração de Planos de Inspeção;

11 - Geração de trajetórias de ferramentas;

12 - Especificação de ferramentas de montagem;

13 - Definição de modelos de fundição das peças

3.2.3.1 - ANÁLISE DO CONJUNTO

Este submódulo analisa a estrutura de dados automaticamente, identificando cada peça e verificando as relações de montagem existentes entre elas, quando houver. Por meio da análise dos dados, extraem-se informações que são agrupadas em atributos adequados na estrutura, informando quais as peças que estão montadas, bem como as *features* que possuem contato quando caracterizam uma montagem.

Esses dados permitem que outros submódulos utilizem tais informações para outros tipos de análise, como, por exemplo, para determinar a cotagem funcional, as superfícies a serem usinadas, bem como o respectivo acabamento superficial, isso de forma automática. Também podem ser utilizados os dados resultantes para a análise de seqüenciamento de montagens manuais ou efetuadas por robôs. Todas as informações geradas neste módulo são armazenadas na estrutura de dados.

3.2.3.2- COTAGEM FUNCIONAL

Este submódulo executa a cotagem das peças, ou seja, analisando as informações a respeito da montagem das peças e com apoio de um sistema especialista e algoritmos, define as cotas funcionais a serem utilizadas e as armazena adequadamente na estrutura de dados.

Também executa a cotagem, introduzindo as cotas no desenho e outras informações que tradicionalmente acompanham um desenho técnico, havendo, ainda, cotagem de forma automática ou manual, de acordo com o desejo do usuário.

As informações de cotagem poderão ser alteradas a qualquer momento através de operações manuais, o que deverá proporcionar a atualização dos respectivos dados na estrutura do produto (inclusive a sua representação gráfica), mantendo a coerência do conceito das *features* e resultando, assim, numa maior flexibilidade para o usuário. A cotagem baseia-se no conceito de cotagem funcional, que considera a função da peça no conjunto para a definição de suas cotas.

3.2.3.3 - ESCOLHA DAS TOLERÂNCIAS

Este submódulo utiliza as informações de quais peças que estão montadas entre sí pelas cotas nominais, dados esses resultantes da análise do conjunto com relação à montagem, e identifica para quais dimensões devem ser indicadas tolerâncias, tanto diametrais como longitudinais (sentido paralelo ao eixo de rotação da peça). As informações a respeito de tolerâncias estão à disposição do usuário através do banco de dados que está interligado ao sistema e que pode ser consultado de duas formas básicas: na primeira, o próprio sistema se encarrega de consultá-lo em função da operação que está sendo executada; na segunda, o usuário pode utilizá-lo livremente para consultas que deseja realizar. Dessa forma, é possível selecionar as tolerâncias de forma interativa, buscando adequar a funcionalidade da peça à montagem do produto.

3.2.3.4 - IDENTIFICAÇÃO DO ACABAMENTO SUPERFICIAL

Com as informações geradas no submódulo de cotagem e de tolerâncias, é possível identificar quais as superfícies a serem usinadas e o respectivo acabamento superficial a ser considerado. A identificação do acabamento superficial está relacionada com as tolerâncias escolhidas para as respectivas dimensões controladas, ou seja, aquelas dimensões que serão usinadas e controladas pela inspeção e que também são as superfícies que possuem contato entre

as peças para que se dê a montagem. Desse modo, a especificação do acabamento superficial depende da análise das tolerâncias especificadas, buscando relacioná-las com o acabamento superficial (rugosidade).

3.2.3.5 - FUNÇÕES AUXILIARES

Neste submódulo, estão as funções auxiliares que permitem aos módulos executarem a tarefa principal. Por exemplo, há funções que efetuam a consulta ao banco de dados, no caso da escolha das tolerâncias; no caso da identificação da montagem, funções que ativam o mecanismo de inferência do sistema especialista para verificar as condições da montagem, etc.

3.2.4 - ESTRUTURA DE DADOS DO PRODUTO

É a estrutura de dados do produto que permite que as informações sejam armazenadas de modo organizado e seja mantida a coerência dos conceitos da representação descritos na biblioteca das *features*. Assim, a estrutura do produto será mantida de forma coerente, e suas ligações, do nível mais elevado ao mais baixo, serão registradas. Desse modo, as informações estarão disponíveis para serem consultadas por qualquer função de qualquer módulo que necessite informações a respeito do produto. Essa estrutura de dados visa armazenar não somente os dados, mas também as relações existentes entre os vários itens da estrutura, de modo que, ao percorrê-la, possa-se conhecer as relações existentes entre os elementos representados.

A estrutura de dados deve permitir que, a partir de um conjunto definido inicialmente, sejam criados vários subconjuntos e que, a partir de cada subconjunto, várias peças possam ser criadas; do mesmo modo, uma peça terá na sua representação várias *features*. Assim, cada elemento criado deve ser individualizado e associado aos elementos anteriores e posteriores. Em nível de representação das *features*, a estrutura deve manter a ligação entre as *features* ditas básicas com as modificadoras (item 3.5.1), sendo possível que uma determinada *feature* modificadora pertença a uma *feature* básica, a qual, por sua vez, pertence a uma peça que é de um subconjunto e, finalmente, a um conjunto (item 4.12).

3.2.5 - INTERFACE DE COMUNICAÇÃO

Este módulo tem a função de servir de ponte de comunicação dos vários módulos e submódulos com o Mecanismo de Inferência que vai atuar sobre a Base de Conhecimento. É nessa interface que serão feitas as conversões de dados para o formato que o sistema especialista requer, passando as informações da estrutura de dados para o sistema especialista, e vice-versa. Deve-se mencionar que os dados que o sistema especialista retorna não correspondem, necessariamente, à mesma quantidade de dados enviados para o sistema especialista.

As informações da estrutura de dados são enviadas para o sistema especialista através do módulo ou submódulo que esteja atuando, isso em função das particularidades de cada módulo ou submódulo (Figura 3.2).

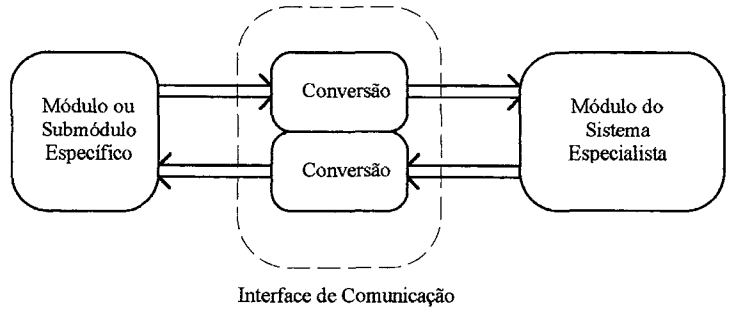


Figura 3.2 - Interface de comunicação com o Sistema Especialista.

3.2.6 - INTERFACE DE CONSULTA

Este módulo tem a função específica de efetuar a comunicação com o banco de dados de manufatura e montagem, com o que, a partir de uma consulta, podem retornar várias informações a respeito do item consultado. Por exemplo, para realizar uma consulta sobre um componente padronizado, como uma arruela, é necessário que se introduza o diâmetro do parafuso, para que no banco de dados obtenha-se o diâmetro interno, o diâmetro externo e a espessura da arruela. A interface de consulta é função de cada tipo de aplicação, ou seja, depende de cada tipo de consulta efetuada no banco de dados.

3.2.7 - SISTEMA ESPECIALISTA

A função do sistema especialista é efetuar a pesquisa na base de conhecimento para a tomada de decisão quanto a um problema em análise, o que é feito através da utilização do mecanismo de inferência do sistema. Para que o sistema possa atuar, é também necessária a definição das informações no sistema especialista; daí a necessidade da existência de uma interface para a passagem de dados e mapeamento no formato do sistema especialista.

3.2.7.1- BASE DE CONHECIMENTO

A base de conhecimento é dividida em vários submódulos, cada um referente a um tipo de aplicação; o conhecimento é representado em forma de regras. Assim, há um submódulo para a base de conhecimento do modelador para avaliar as condições de instanciação de uma nova *feature* na composição de uma peça, para excluir, alterar ou qualquer outra operação do modelador.

Outro submódulo atua na análise do conjunto, onde estão descritas as formas de contato que caracterizam uma montagem de acordo com a disposição das peças. Também estão nesse módulo as informações a respeito da disposição das dimensões que devem ser associadas à peça. Essa base de conhecimento será consultada através da utilização do mecanismo de inferência, o qual recebe os dados do módulo que solicita um parecer, dando em retorno uma mensagem, que poderá ser a autorização à operação ou a sua negação; no último caso, informará o motivo da negação.

3.2.7.2 - BASE DE CONHECIMENTO DO MODELADOR

A base de conhecimento utilizada pelo modelador corresponde a vários tipos de conhecimento, que são representados através de regras e subdivididas em diversos módulos. O primeiro conjunto de regras se refere àquelas restrições relativas às condições de instanciação das *features* básicas, as quais levam em conta condições de manufatura e existência das *features*. Assim, dentre as restrições básicas que devem estar registradas na base de conhecimento, têm-se:

- evitar que duas *features* básicas ditas externas se sobreponham, ocupando a mesma posição no espaço; da mesma forma para as *features* básicas ditas internas;

- evitar que as *features* básicas internas tenham suas faces iniciais obstruídas por outras *features* básicas externas;

- condicionar que as *features* internas estejam totalmente contidas nas *features* externas;

- não permitir que uma *feature* de natureza interna seja inserida antes de uma *feature* de natureza externa.

Essas quatro regras básicas são descritas por Lima (1994) como: Restrição de interpenetrabilidade das *features*; Restrição de inacessibilidade; Restrição existencial 1 e Restrição existencial 2. Isso é perfeitamente válido para a instanciação do conjunto das *features* básicas; com relação às *features* modificadoras, no entanto, surgem restrições novas a serem introduzidas, como :

- Restrição de igualdade : Evitar que duas *features* externas ou internas de faces iguais sejam adjacentes. Nesse caso, pode-se ter a presença de uma *feature* eixo com um determinado diâmetro e comprimento e, posteriormente, de uma nova *feature* eixo de mesmo diâmetro sendo instanciada.

- Restrição de incompatibilidade : Evitar que a inserção ou exclusão de *features* externas tornem incompatível a existência de outras *features*. Isso pode ser observado nos seguintes casos: a) inserção de um eixo, sendo que, na extremidade de um outro eixo existente, há uma *feature* modificadora chanfro, e o eixo a ser inserido é de diâmetro maior que o existente; b) entre dois eixos de uma peça que são de diâmetros diferentes, há uma *feature* modificadora concordância e, quando da exclusão de um dos eixos, torna-se incompatível a existência da concordância.

Essas restrições são descritas em forma de regras em função das diversas *features* descritas na biblioteca e em função das relações que ocorrem entre as diversas *features* da biblioteca. Além dessas regras, outras serão definidas em função do tipo de material da peça, como, por exemplo, a profundidade de um furo roscado, que é definida em função do diâmetro do parafuso e do material com que é executada a rosca.

3.2.7.3- BASE DE CONHECIMENTO DA MONTAGEM

A base de conhecimento da montagem é direcionada no sentido da identificação de quando duas peças estão montadas, isto é, tendo sido construído um determinado conjunto que está representado dentro da metodologia descrita no sistema FeatCAD-2D, essa base de conhecimento servirá para efetuar a verificação das condições em que duas peças são consideradas montadas.

O objetivo dessa base de conhecimento é dotar o sistema de capacidade de identificação dos fatos e gerar informações a respeito do produto representado. Assim, pode-se citar :

- Regra da Igualdade: Para que haja uma montagem entre duas peças, é necessária a existência de uma *feature* básica interna e uma externa em peças distintas e que os referidos diâmetros sejam iguais (diâmetros nominais);

- Regra da Sobreposição: Se há uma *feature* básica interna e uma externa de peças distintas, elas devem estar sobrepostas para que sejam caracterizadas como montagem.

As duas regras descritas, sendo utilizadas na seqüência, caracterizam as condições básicas para a verificação da ocorrência de uma montagem entre duas peças. Além de caracterizar a respectiva montagem, o sistema deverá possibilitar a identificação dos contatos existentes, que foram divididos em contato diametral e contato axial.

O cumprimento das regras descritas para a caracterização da montagem define, ao mesmo tempo, um contato diametral; o contato axial, por sua vez, refere-se à coincidência de duas faces de peças distintas. Assim, pode-se enumerar a seguinte regra:

- Regra do Contato Axial: Se duas peças são consideradas montadas e a face de uma delas é coincidente com a face de qualquer outra peça, há então um contato axial entre essas duas peças (item 3.9.1.2).

As regras descritas são implementadas para abranger o maior número de casos em função de particularidades das *features* envolvidas.

3.2.8 - BASE DE DADOS DE MANUFATURA E MONTAGEM

Este módulo é o responsável pelo registro de informações técnicas que auxiliam e complementam a tomada de decisões do sistema. É nele que estão armazenadas as tabelas de

tolerâncias, que são pesquisadas de acordo com a necessidade dos procedimentos no sistema. Nele também estão registradas as informações a respeito dos produtos normalizados e a padronização da empresa, ou seja, o que existe por norma e o que é utilizado pela respectiva empresa. Outro tipo de registro é quanto a dados técnicos de componentes padronizados, que devem ser consultados quando de sua utilização.

Este módulo está à disposição do usuário do sistema somente para consulta, não possibilitando acesso para a realização de alterações no banco de dados. Quando da necessidade de alterações, isso deve ser feito por um *superusuário*. A consulta ao banco de dados é feita de dois modos. No primeiro, a consulta é realizada pelo próprio sistema quando da identificação da necessidade dessa, o que é feito automaticamente. Por exemplo, quando o sistema identifica que duas peças são montadas, automaticamente consulta o banco de dados para que o usuário selecione as condições de tolerâncias do respectivo acoplamento. No segundo modo, o banco de dados poderá ser chamado de forma independente da aplicação, servindo de meio de consulta para o usuário, para isso possuindo funções independentes.

3.3 - HIERARQUIA NO MODELO DE INFORMAÇÃO

A hierarquia das informações corresponde à ordem em que as informações devem ser dispostas e às ligações necessárias para a obtenção de uma informação correta. Parte desse modelo é baseada em Lima (1994), com as devidas adaptações e ampliações necessárias para o presente trabalho (Figura 3.3).

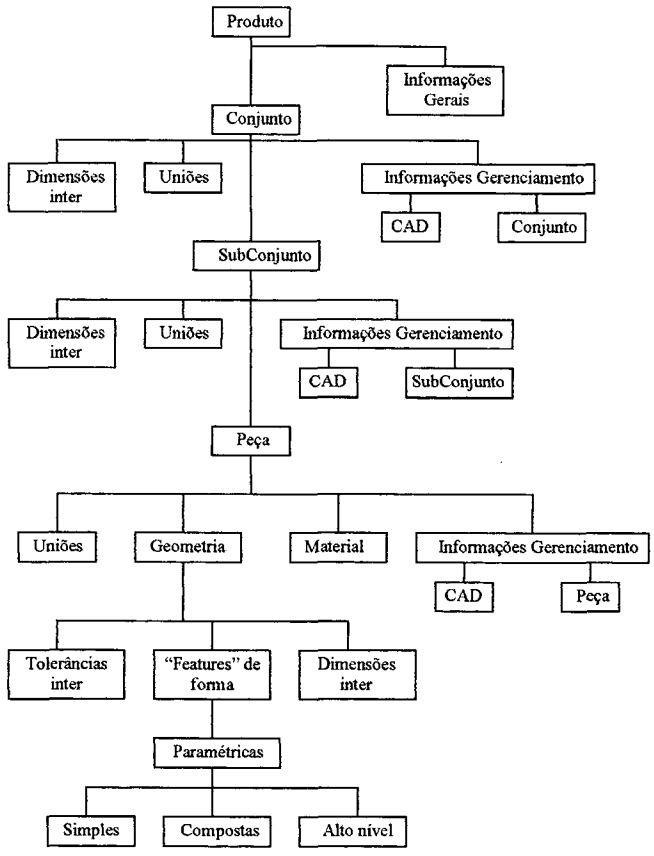


Figura 3.3 - Hierarquia do modelo de informação.

3.4 - O PRODUTO

Segundo Rosa (1993), o produto deve ser considerado como um objeto que é a razão de ser da empresa, ou seja, é o objetivo de todo o esforço despendido para alcançar o objetivo final da empresa. O produto não é um objeto único, mas o resultado da união de vários objetos de forma adequada para que seja obtido um resultado compatível com a finalidade proposta. Para isso, é necessária uma estruturação do produto a fim de que ele possa ser melhor desenvolvido e compreendido. Tal estruturação permite que os diversos níveis possam ser representados de acordo com a fase de desenvolvimento, possibilitando que as informações necessárias possam ser inseridas em cada nível e que suas relações com os outros níveis sejam descritas.

Os níveis que formam a estrutura do produto iniciam com uma visão global, sendo feita, a cada nível subseqüente, uma particularização das informações representadas, as quais descrevem o estado de cada nível e suas respectivas relações com os níveis anterior e posterior, quando for o caso. Com a utilização de uma estrutura adequada, é possível descrever o produto

de modo compatível com as necessidades dos usuários dessas informações, que serão aplicadas na análise do projeto, na fabricação, no controle e em toda a vida desse produto.

Vários autores, como, por exemplo, Krause (1993), Suzuki (1996), Rosa (1993), Wingard (1992), descrevem modelos para a representação do produto.

3.4.1 - O MODELO DO PRODUTO

O modelo do produto representa as informações necessárias para que o produto seja representado adequadamente, o que corresponde à visão dos conceitos necessários para a implementação do produto.

Como o produto é a razão da empresa, todas as atividades a ele relacionadas devem fazer parte do referido modelo, ou seja, tal modelo deve estar organizado de modo que se possa conhecer tudo a respeito do referido produto e de modo natural, sem que seja necessária a duplicação de atividades para a inserção das informações, como vem ocorrendo hoje, quando os desenhos representam apenas informações gráficas (para o computador), sendo colocadas em separado as informações tecnológicas.

Na Figura 3.4, está representado o modelo do produto direcionado para a representação do projeto detalhado, não sendo, entretanto, contemplada a visão total do modelo, como visto na revisão bibliográfica.

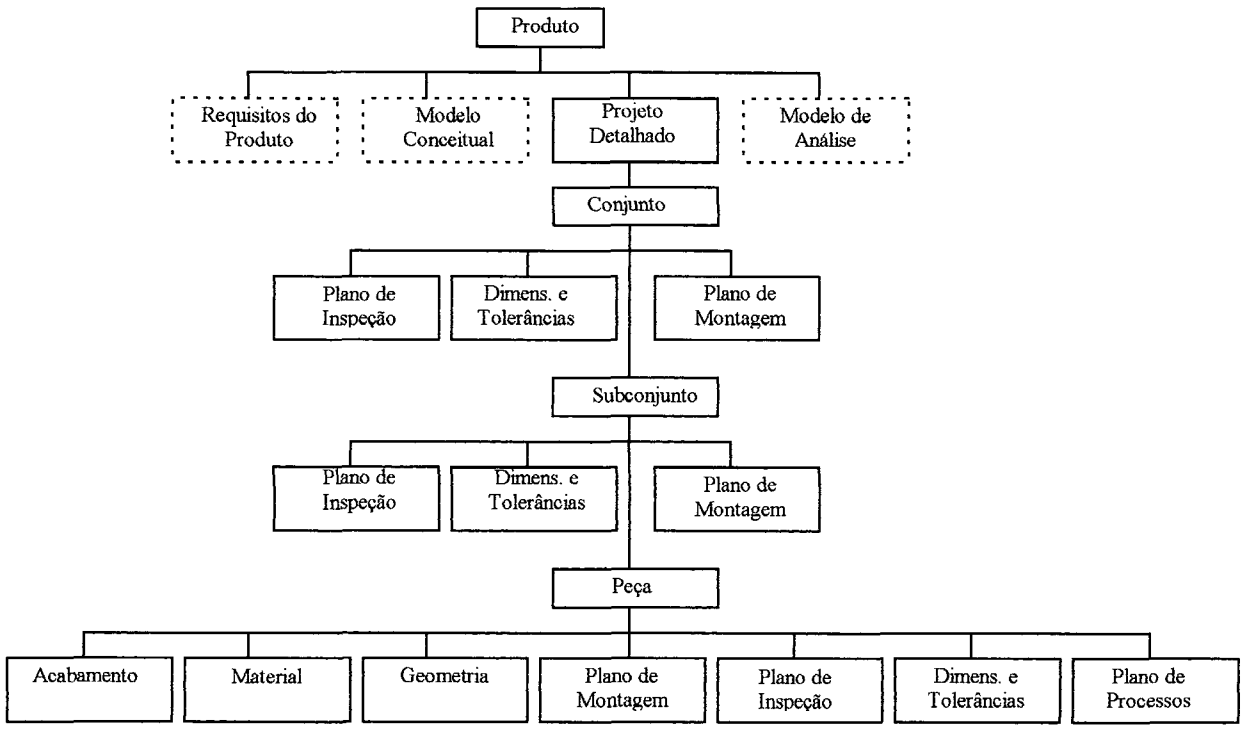


Figura 3.4 - Modelo do produto para o projeto detalhado.

3.4.2 - A ESTRUTURA DO PRODUTO EM NÍVEL DE PROJETO DETALHADO

É plenamente aceito nos dias de hoje que toda tentativa, ou estratégia de integração das diferentes atividades envolvidas num processo de projeto e fabricação, deve considerar como ponto de partida a estrutura de informação (Ahmad citado por Rosa (1993)). A estrutura de um produto pode ser desmembrada em diversas subestruturas (níveis). De uma forma genérica, o produto é composto por uma série de componentes denominados de *subconjuntos*, que, montados, resultam na sua forma final, o conjunto.

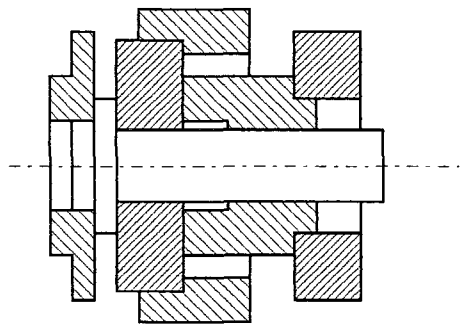


Figura 3.5 - Conjunto montado.

A Figura 3.5 ilustra um exemplo de um conjunto (produto) que pode ser subdividido em diversos subconjuntos, os quais passam a ser componentes do conjunto, como se fossem elementos individuais. A quantidade de subconjuntos é função da complexidade do produto (Figura 3.6). Do ponto de vista do produto, um subconjunto significa diretamente uma submontagem.

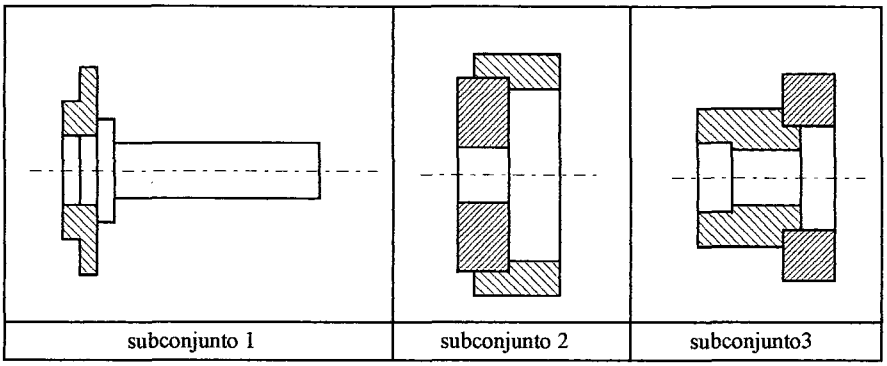


Figura 3.6 - Subconjuntos montados.

Um subconjunto é composto por uma série de componentes, denominados de peças (Figura 3.7). No caso de produtos mais complexos, um subconjunto também pode ser resultante da composição de outros subconjuntos. Do mesmo modo, a quantidade de peças (ou subconjuntos) que compõem um subconjunto vai depender da complexidade do mesmo.

Em nível macro (Figura 3.8), uma peça é a última instância de componente que pode ser manuseado dentro da estrutura do produto. É possível dizer que uma peça é a menor parte do produto que é tratada isoladamente, como um item de compra, um item de estoque, ou um item para a montagem (Rosa, 1993). Na Figura 3.7, podem ser observadas as peças que compõem os respectivos subconjuntos mostrados na Figura 3.6.

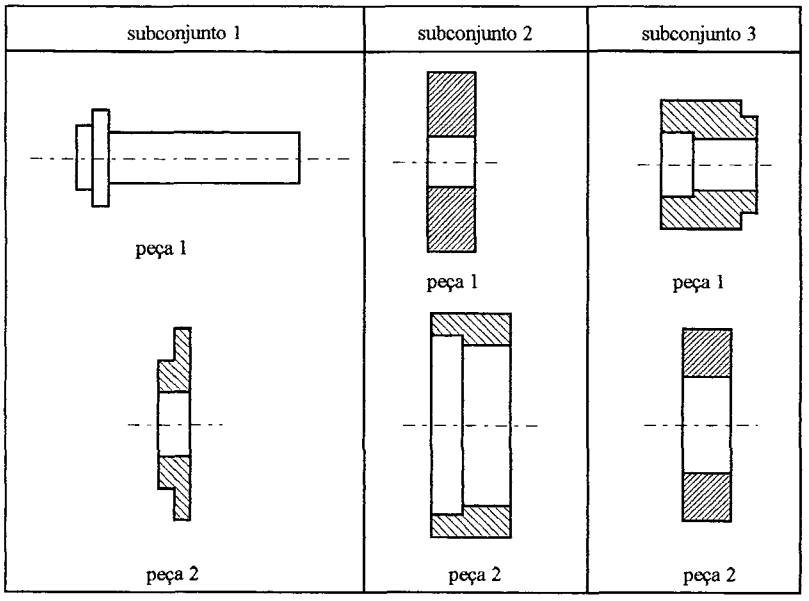


Figura 3.7 - Peças de cada subconjunto.

Observando a Figura 3.8, percebe-se que, em nível macro, estão os elementos que podem ser manuseados no mundo real em nível micro, os elementos que somente são manuseados em nível conceitual, ou seja, em nível de representação de uma idéia.

Em nível conceitual, no presente trabalho, utilizam-se as *features* (Figura 3.8) para a representação das informações que irão constituir uma peça. Com a utilização das *features*, cria-se uma nova estrutura para a representação e utilização da peça.

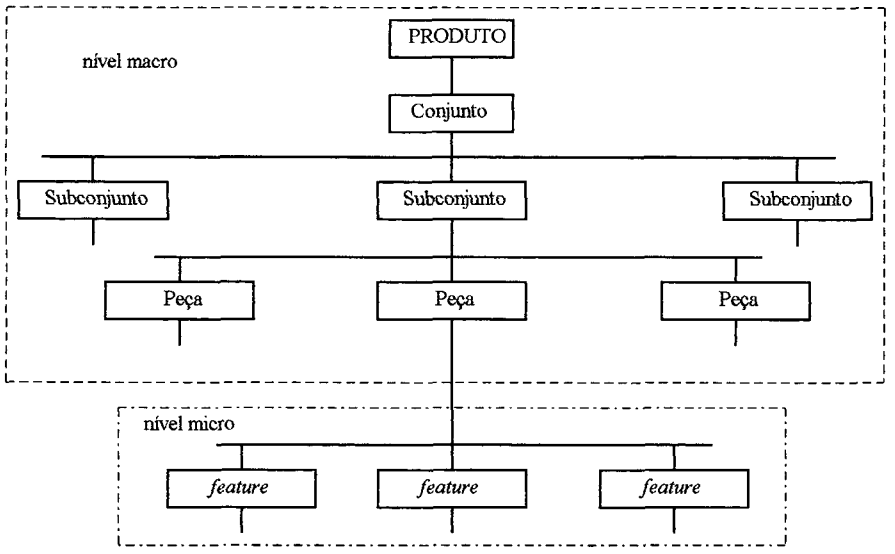


Figura 3.8 - Estrutura do produto.

As *features* são uma representação possível tanto em nível gráfico como em nível tecnológico, possibilitando uma interação completa da representação de informações num ambiente computacional. Neste trabalho, o termo *feature* será utilizado para caracterizar as formas básicas que, isoladas ou em conjunto, irão compor uma peça (Figura 3.9).

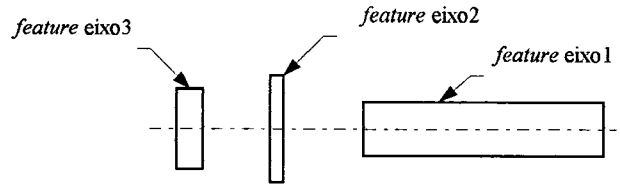


Figura 3.9 - Peça representada através de *features*.

3.4.3 - DEFINIÇÕES DAS ENTIDADES ENVOLVIDAS

É necessário que sejam conhecidos todos os tipos de informações a serem agregadas a cada item que compõe a estrutura do produto. Assim, precisa-se definir cada item de modo a obter-se informações significativas do contexto a representar.

3.4.3.1 - CONJUNTO

Para produtos complexos, um conjunto é definido como a reunião de vários subconjuntos e peças, as quais proporcionam a união dos vários subconjuntos. Para um produto mais simples, a reunião de peças já compõe um conjunto. Ao conjunto deverão estar associadas informações que caracterizam e identificam, o que é importante em nível de conjunto (Tabela 3.1).

Tabela 3.1 - Informações do conjunto

INFORMAÇÕES DO CONJUNTO
Nome do Produto
Data
Código
Responsável Técnico
Projetado por
Lista de Subconjuntos
Dimensões de Montagem

Cada peça criada com seus atributos faz parte da lista de peças do conjunto ou do subconjunto, de acordo com a complexidade; as dimensões de montagem se referem às dimensões a serem obedecidas quando da montagem do produto; a lista de componentes da montagem se refere a elementos que proporcionam a ligação entre subconjuntos para formar o conjunto, como parafusos, pinos,etc., ou subconjuntos que serão unidos uns aos outros.

Um conjunto não possui uma representação gráfica própria, mas é o resultado da união de outras representações, ou seja, de todas as peças que o compõem o conjunto. Essa representação será única em termos de informação, o mesmo sendo válido para o subconjunto.

3.4.3.2 - SUBCONJUNTO

Um subconjunto é composto por um grupo de peças e faz parte de um conjuto maior, do qual, devido à sua complexidade, foi separado (Tabela 3.2).

Tabela 3.2 - Informações de subconjunto

INFORMAÇÕES DO SUBCONJUNTO
Nome Subconjunto
Data
Código
Responsável Técnico
Projetado por
Substitui
Substituído por
Quantidade
Dimensões de montagem
Lista de Peças

O código definido no conjunto se estenderá para os subconjuntos e para as peças à medida que esses forem criados. A lista de peças será resultante dos atributos encontrados nas peças, ao passo que cada peça criada com seus atributos fará parte da lista de peças do subconjunto ou do conjunto, de acordo com a complexidade.

As informações "substitui" e "substituído por" se referem a projetos que sofreram modificações, mantendo assim as referências.

3.4.3.3 - PEÇA

Uma peça é um elemento básico para compor um subconjunto ou um conjunto, e a união de diversas peças leva à formação de um subconjunto ou conjunto de acordo com a complexidade do produto (Tabela 3.3).

Tabela 3.3 - Informações de peça

INFORMAÇÕES DE PEÇA
Nome Peça
Data
Código
Responsável técnico
Projetado por
Material
Quantidade
Substitui
Substituído por
Lista de*features*
Dimensões Lineares
Tolerâncias Dimensionais
Tolerâncias Geométricas

Uma peça é definida não somente em função de sua forma, mas das suas dimensões, bem como das características tecnológicas que a definem como sendo única. Uma peça é representada por uma ou várias *features* de acordo com sua complexidade; poderá ser composta por várias *features*, desde que existam na biblioteca do sistema e satisfaçam as condições de existência (ver itens 3.5.1 e 3.5.2). Numa peça, as *features* externas serão representadas por linhas cheias e as internas, por linhas tracejadas, todas de uma mesma cor, o que identifica uma mesma peça.

As peças aqui representadas correspondem àquelas obtidas a partir de material contínuo, ou seja, peças resultantes de barras ou fundidas. Em nível de informação, uma peça será representada por listas de *features* básicas, que, por sua vez, possuem listas de *features* modificadoras a elas ligadas. Como o sistema de coordenadas utilizado será o absoluto, uma peça não possui um sistema de referência único, podendo ser utilizadas como referências as suas extremidades.

3.4.3.4 - *FEATURE*

Do ponto de vista geral, todas as definições utilizadas na estrutura do produto serão representadas na forma de *feature*, que, de acordo com Shah (1991), são formas genéricas às quais os engenheiros associam certas propriedades ou atributos e conhecimentos úteis em processos de raciocínio sobre o produto, ou seja, as *features* podem ser vistas como primitivas de engenharia.

No caso específico, o termo *feature* sempre será referido a elementos de forma, que, através de composição com outras ou sozinhas, representam uma peça. Para que uma *feature* possa representar adequadamente esse elemento básico, são definidos atributos, divididos em cinco grupos principais:

- atributos de forma;
- atributos de informações geométricas;
- atributos de tolerâncias;
- atributos de acabamento superficial;
- atributos de gerenciamento.

Os dois primeiros grupos correspondem a atributos geométricos de utilização do CAD, enquanto os dois grupos seguintes correspondem a atributos tecnológicos de utilização para os processos de fabricação. O último grupo de atributos é responsável pela representação de elementos que permitem uma melhor identificação e manuseio das informações. Assim, para cada *feature*, esses cinco grupos estarão representados, sendo que os atributos específicos dentro de cada um poderão ser diferentes em função das informações a serem representadas para cada *feature*.

Os atributos de forma correspondem à descrição geométrica da *feature* a qual se deseja representar, constituindo-se nos parâmetros que serão utilizados para a sua representação gráfica; os atributos de informações geométricas são as informações que representam os dados que serão manipulados para efetuar as operações de instanciação e análise; os atributos de tolerâncias serão utilizados para análises de funcionalidade no projeto, na seleção dos processos de fabricação, como também para os processos de controle; os atributos de acabamento superficial terão a mesma aplicação descrita para os atributos de tolerâncias; enfim, os atributos de gerenciamento correspondem a informações que permitem que o sistema possa comunicar e

gerenciar adequadamente a estrutura do produto e o CAD, mantendo a consistência entre ambas as estruturas.

Dentre as várias metodologias que podem ser utilizadas para a representação de *features*, optou-se no presente trabalho pelo projeto por *features*, o qual se baseia na existência de uma biblioteca de *features* e em procedimentos de utilização preconcebidos em termos de primitivas de engenharia, procurando-se obter uma representação do mais alto nível, próxima à necessidade de concepção do projetista, isto é, através dessa representação, procura-se captar os elementos necessários para a representação completa de uma idéia.

3.5 - *FEATURES* DE PROJETO

A idéia da utilização das *features* consiste em captar o primeiro pensamento a que o projetista é levado frente ao problema, desenvolvendo-o, então, a partir daí. Assim, alguns problemas comuns, como escolher um parafuso a ser aplicado num determinado furo roscado, poderão ser traduzidos de forma seqüencial e lógica para uma codificação adequada.

Um exemplo de uma *feature* simples a ser desenvolvida é um eixo. Pergunta-se: qual a idéia inicial que surge a partir desse elemento? Pensa-se logo num cilindro, que possui por característica uma base circular expressa pelo raio e uma altura. Para o projetista, um eixo é uma barra de qualquer tipo de seção e que, de algum modo, está apoiada sobre dois pontos de apoio (p. ex., mancais), possuindo componentes montados sobre ela, e a barra possui um movimento de rotação.

Logo, um eixo não é uma peça isolada como vista pelo processista, mas faz parte de um outro contexto maior, que depende também dos objetos que estejam conectados ao respectivo eixo. No modo mais elementar, um eixo é uma barra cilíndrica, definida por um diâmetro e um comprimento. Assim, o projetista irá pensar num eixo e relacioná-lo diretamente com um diâmetro e comprimento ou, como no caso de um eixo escalonado, por segmentos de eixos.

Utiliza-se o diâmetro e não o raio como elemento de referência porque o primeiro é a dimensão mais fácil de ser obtida na fabricação (p. ex., usinagem). Além disso, os instrumentos de medição mais comuns disponíveis possibilitam uma rápida determinação do diâmetro, e as máquinas operatrizes também se baseiam nessa dimensão para efetuar os movimentos. Desse modo, é importante captar a funcionalidade da *feature* frente ao contexto de aplicação.

Para a utilização das *features*, é necessário que haja uma classificação, a qual servirá de suporte para a representação das informações em todos os níveis do produto.

O desenvolvimento do presente trabalho tem por base a classificação desenvolvida pelos seguintes autores: Ovtcharova (1992), Lima (1994), e Schulz (1993).

3.5.1 - CLASSIFICAÇÃO DAS *FEATURES* DE FORMA

A classificação das *features* é necessária para que sejam conhecidos realmente os tipos de *features* existentes e a sua localização dentro do contexto em análise. As *features* de forma são classificadas, inicialmente (Figura 3.10), quanto ao modo de representação em: (a) *features* paramétricas, as quais podem ser instanciadas através de um ou mais conjuntos de parâmetros geométricos; por exemplo, um furo pode ser definido pelos seguintes parâmetros: diâmetro e profundidade e (b) *features* não-paramétricas, que são definidas ou através de um conjunto de entidades geométricas de baixo nível explicitamente identificadas (p. ex., um furo passante pode ser definido através de suas superfícies, a lateral, a de entrada e a de saída), ou através de conjuntos específicos de *features* paramétricas (*macro*) (Lima, 1994),onde essas são definidas através da descrição da forma. As *features* não-paramétricas não são abordadas no presente trabalho.

As *features* paramétricas são subdivididas em função da forma geral da *feature*, resultando, assim, em duas subdivisões principais, que são: as *features* rotacionais e as *features* prismáticas (Figura 3.10). Neste trabalho, somente serão analisadas as *features* rotacionais, ou seja, peças compostas de *features* em que predominam os corpos de revolução, os quais darão a característica à peça como sendo rotacional (Figuras 3.10 e 3.11).

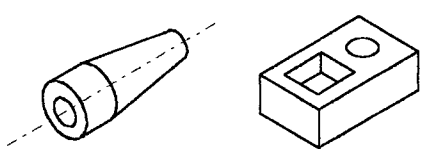


Figura 3.10 - *Features* rotacional e prismática.

Dentre as *features* rotacionais, há três tipos: simples, que correspondem às *features* que são definidas individualmente ou em grupo; compostas e de alto nível (Figura 3.11).

As *features* simples são subdivididas em *features* elementares e *features* combinadas (Figura 3.11), correspondendo as elementares àquelas *features* descritas individualmente. Dentre as *features* elementares, tem-se o nível mais baixo, que são as *features* básicas e as *features* modificadoras (Figura 3.11).

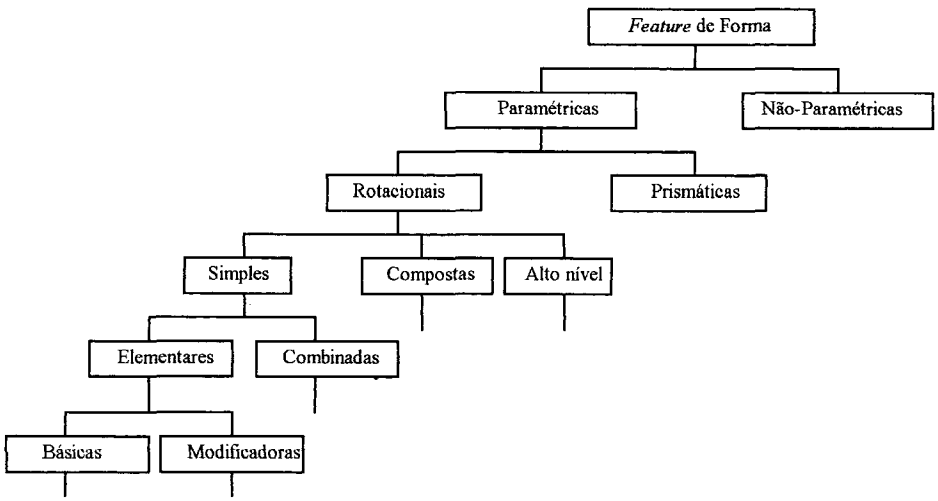


Figura 3.11 - Classificação geral das *features.*

As *features* básicas (Figura 3.12) são classificadas em *features* de volume positivo e de volume negativo, ou seja, aquelas que possuem massa e aquelas que representam o espaço vazio, resultando, então, em elementos práticos como eixo e furo. No presente trabalho, elas se tornam *features* de referência para o desenvolvimento da aplicação do conceito de *features* modificadoras.

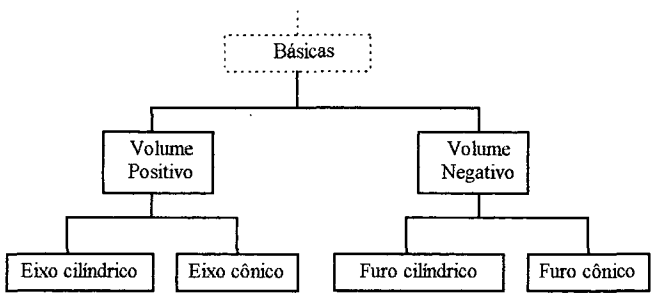


Figura 3.12 - *Features* básicas.

As *features* modificadoras são aquelas que produzem alterações nas *features* básicas, podendo ser rotacionais (Figura 3.13) ou prismáticas (Figura 3.14). São, ainda, divididas em função do seu processo de geração sobre as *features* básicas; as rotacionais em transição

circulares, superfície circular, de face circular e radial (Figura 3.13). Para as *features* prismáticas, tem-se a subdivisão em axiais e radiais (Figura 3.14).

As *features* modificadoras rotacionais de transição circular (p. ex., chanfro ou concordância) são aquelas localizadas sobre uma aresta que separa uma superfície cilíndrica de uma superfície plana (denominada *face*) de um cilindro, ou duas superfícies quaisquer (Figura 3.13). Já as *features* modificadoras rotacionais de superfície circular (p. ex., ranhura) estão localizadas sobre a superfície de uma *feature* básica e produzem modificações em torno do corpo de revolução (Figura 3.13). As *features* modificadoras rotacionais de face circular (p. ex. anel axial de vedação) são aquelas que produzem modificações na face de uma *feature* básica (Figura 3.13).

As *features* modificadoras rotacionais radiais produzem modificações em regiões localizadas da superfície de uma *feature* básica (Figura 3.13), possuindo forma circular e direção do eixo de revolução perpendicular ao eixo de revolução da *feature* básica.

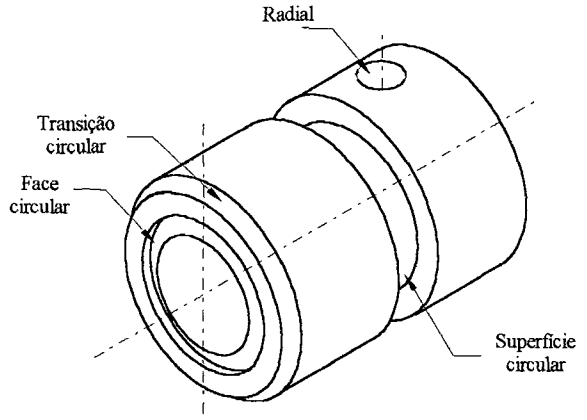


Figura 3.13 - Modificadores rotacionais.

As *features* modificadoras prismáticas axiais (Figuras 3.14 e 3.15) são aquelas que possuem uma seção poligonal ou mista e que, por movimento de translação desta seção paralela ao eixo de rotação da *feature* na qual será inserida, produzem a referida forma. Já uma *feature* modificadora prismática radial é também resultante do movimento de translação de uma seção poligonal, porém no sentido perpendicular ao eixo de revolução da *feature* básica na qual será inserida (Figuras 3.14 e 3.15). Essas definições não estão associadas aos processos de fabricação, mas simplesmente à geometria.

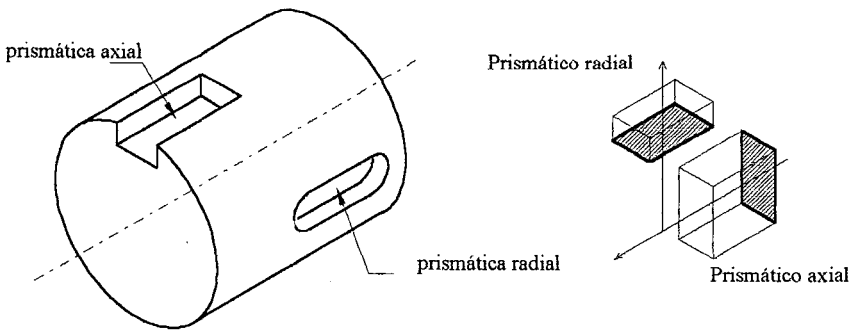


Figura 3.14 - Modificadores prismáticos.

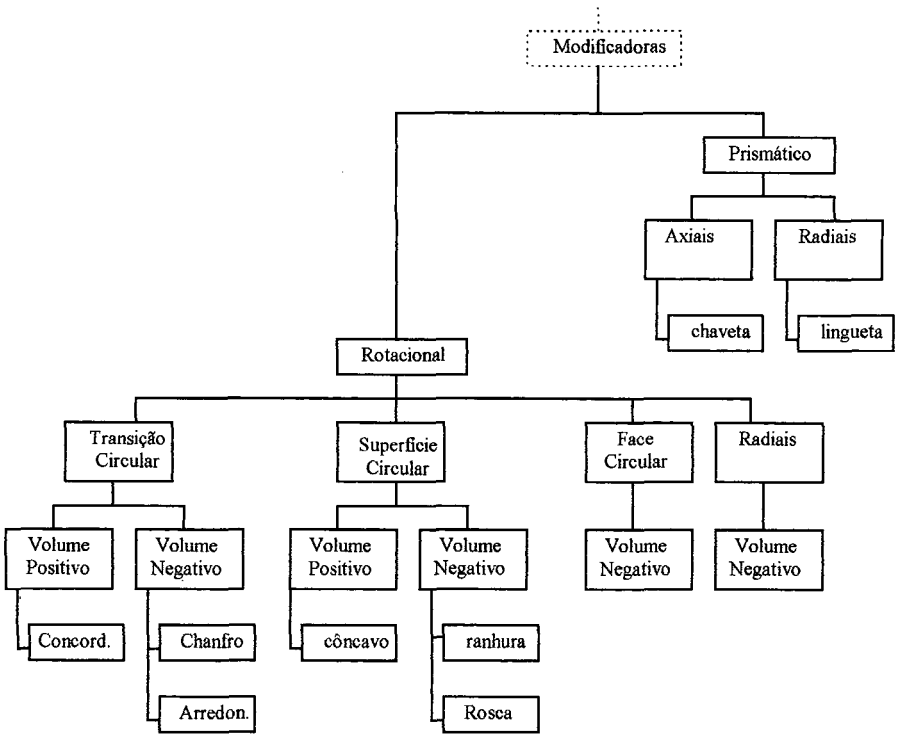


Figura 3.15 - Classificação das *features* modificadoras.

As *features* combinadas (Figura 3.16) são aquelas resultantes da união de duas ou mais *features* elementares que compõem uma forma de nível mais elevado para o projeto (p. ex., furo escalonado, furo escareado). Também podem ser de volume positivo ou negativo.

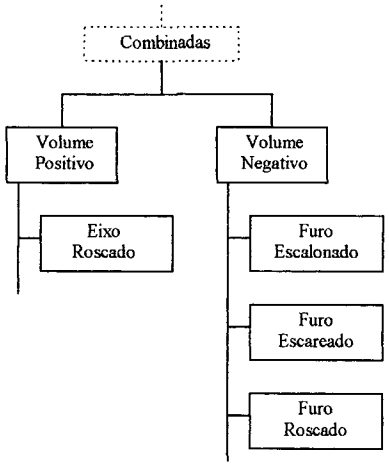


Figura 3.16 - Classificação das *features* combinadas.

As *features* compostas (Figura 3.17) são resultado de uma coleção de *features* simples (Figura 3.11), sendo que essas últimas obedecem a um posicionamento determinado em função de condições preestabelecidas. No caso de peças rotacionais, têm-se por exemplos o padrão circular axial e o padrão circular radial.

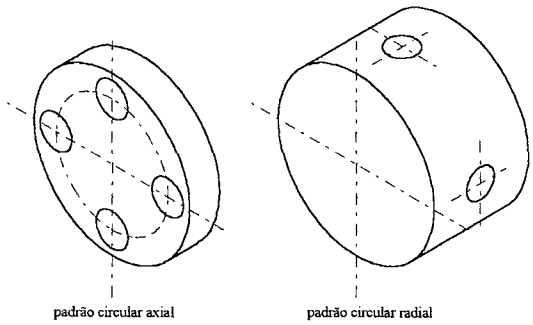
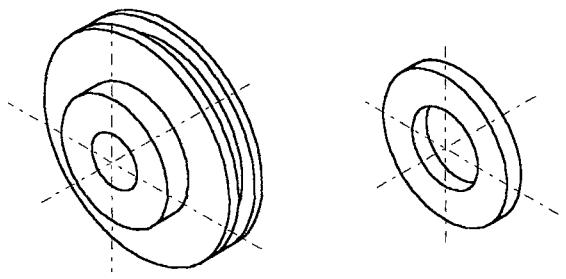


Figura 3.17 - *Features* compostas

Features de alto nível (Figura 3.18) correspondem àquelas representações mais complexas resultantes do uso das *features* simples, compostas e combinadas numa única coleção, resultando numa nova forma totalmente independente.



feature padrão (polia) *feature* padronizada (arruela)

Figura 3.18 - *Features* de alto nível.

As *features* de alto nível ainda podem ser classificadas em *feature* padrão e *feature* normalizada. As *features* padrão são *features* padronizadas internamente pela empresa e que são de seu uso exclusivo, como peças que são utilizadas em vários produtos; já as *features* normalizadas correspondem àquelas peças de fornecimento externo e de padronização feitas através de normas nacionais e internacionais, das quais a empresa passa a ser uma simples usuária (Figuras 3.18 e 3.19).

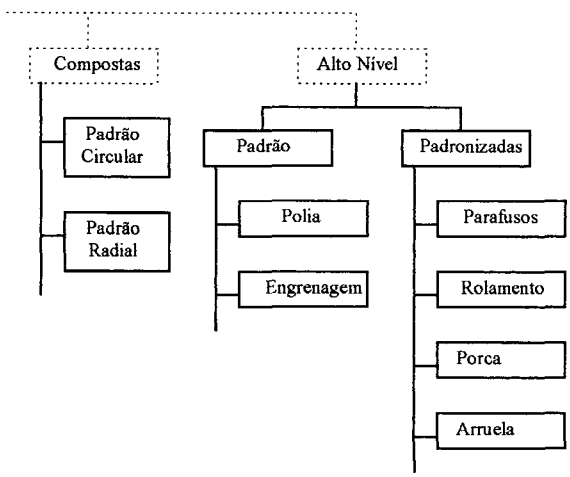


Figura 3.19 - Classificação das *features* compostas e de alto nível.

3.5.2 - HIERARQUIA DAS *FEATURES* E CONDIÇÃO DE DEPENDÊNCIA

A definição da hierarquia das *features* é fundamental para a compreensão da dependência entre cada *feature* durante o procedimento de projeto, conhecendo-se, assim, as restrições necessárias para a criação de cada uma, bem como sua condição de inserção.

Na Figura 3.20, está representada a hierarquia sob a qual as *features* são regidas para a condição de existência de cada uma. Como exemplo, pode-se ver que a *feature* eixo está na parte mais elevada da estrutura, pois a condição para a existência de qualquer outra *feature* é que haja um eixo para que as outras possam ser nela inseridas ou conectadas.

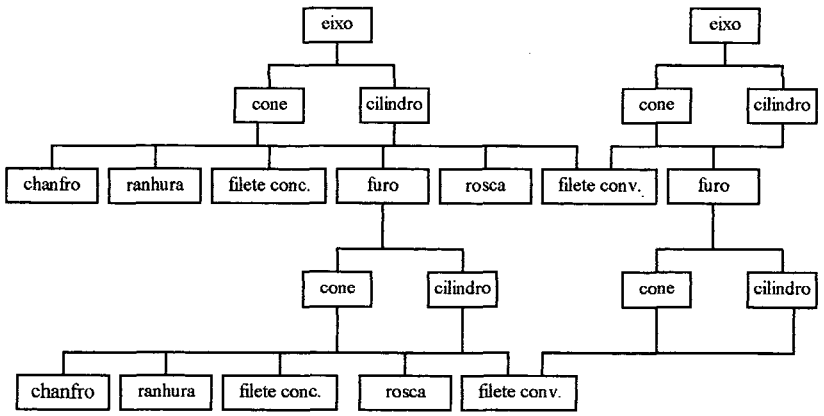


Figura 3.20 - Hierarquia existencial das *features*.

3.5.3 - REPRESENTAÇÃO DAS INFORMAÇÕES NAS *FEATURES* DE FORMA

As *features* básicas representadas pelo eixo e pelo furo guardam informações que são genéricas a ambas, possuindo algumas particularidades que as diferenciam.

Uma *feature* eixo é definida como uma *feature* de volume positivo. Como o domínio escolhido foi o das *features* rotacionais, o eixo cilíndrico e o eixo cônico representam o conjunto de *features* básicas de volume positivo. Desse modo, para representá-las, existe um conjunto de parâmetros necessários para traduzir a sua representação básica em termos de *feature* de forma. Um corpo de revolução como o cilindro possui em sua definição uma superfície de revolução e duas bases que delimitam o comprimento.

Para definir um cone, usa-se uma base e uma superfície cônica. Para o sistema FeatCAD-2D, o cone será também representado por duas bases e uma superfície, no caso cônica. Nesse caso, uma das bases existirá e será definida em função do seu diâmetro; a outra será imaginária, e o diâmetro terá dimensão nula, representando essa base imaginária a posição do vértice. Porém, no caso da representação de um tronco de cone, essa base existirá e terá no atributo diâmetro uma dimensão.

No sistema FeatCAD-2D, a superfície de revolução passou a ser definida pelos respectivos diâmetros que delimitam as bases circulares, as quais foram denominadas de faces e são representadas por coordenadas do sistema cartesiano. O sistema admite a utilização de um sistema de coordenadas absoluto, sendo as *features* posicionadas em relação a um único referencial, independentemente da sua associação com outra *feature*. Assim, o referencial básico a ser utilizado é o centro de rotação da peça, que poderá ser definido a critério do usuário pela inserção da primeira *feature* no sistema.

No domínio considerado, o centro de rotação foi somente considerado na posição horizontal por motivos de simplificação do software protótipo. Como um eixo cilíndrico possui duas bases, que aqui são chamadas de faces, essas foram classificadas como face inicial, que possui a coordenada de posição "CoordI", e face final, que possui a coordenada de posição "CoordF", que serve para delimitar cada *feature* a ser definida. No caso do eixo cilíndrico, a representação pode ser vista na Figura 3.21.

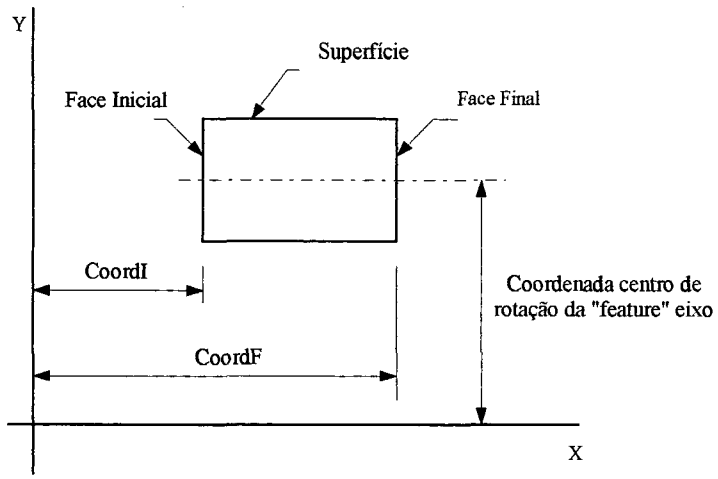


Figura 3.21 - Coordenadas de posição da *feature* eixo cilíndrico.

Desse modo, cada eixo possui coordenadas que representam as suas faces e o centro do seu eixo de rotação, o que permite localizar perfeitamente o eixo no espaço, além dos respectivos atributos de dimensão, que, no caso, são o diâmetro e o comprimento. Deve-se mencionar que toda *feature* terá essas características, mesmo as radiais.

Para uma *feature* furo, é necessário levar em consideração algumas particularidades: no caso dos eixos, as faces inicial e final sempre se encontram na mesma ordem, ou seja, face inicial à esquerda e face final à direita; para um furo, a face inicial é considerada como o início

do furo e a final, o fundo do furo. Numa peça, podem ocorrer furos em condições opostas como representado na Figura 3.22.

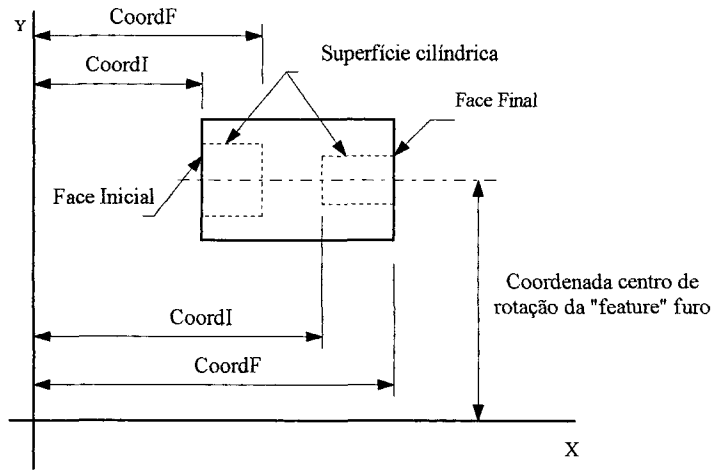


Figura 3.22 - Coordenadas de posição da *feature* furo cilíndrico.

Logo, pode-se observar que a face inicial de uma *feature* furo cilíndrico sempre será coincidente com uma das faces do eixo cilíndrico. Pode ocorrer que a face final de um furo coincida com a face final de outro furo; no caso, um dos furos é passante.

No caso das *features* modificadoras, essas também possuem um sistema de coordenadas absoluto representado pelas coordenadas que delimitam as suas faces e a coordenada de posição do centro de rotação. Pode-se observar na Figura 3.23 que a disposição das faces inicial e final é semelhante à utilizada para os furos, pois essas também dependem da posição em que são inseridas.

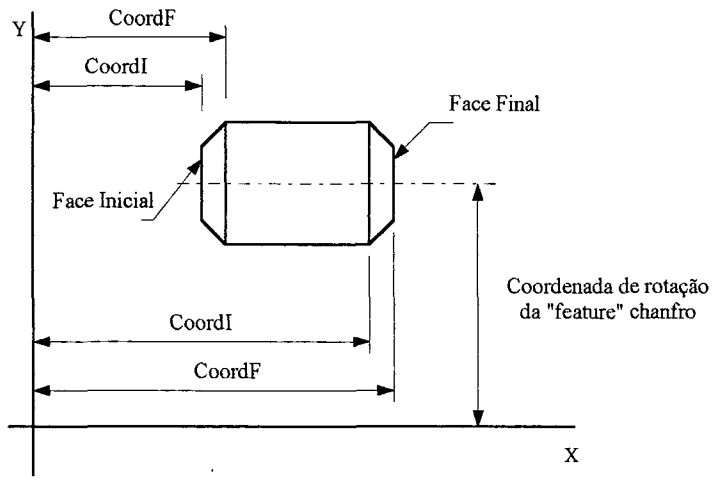


Figura 3.23 - Coordenadas de posição da *feature* chanfro.

Com base nessas considerações, é possível definir atributos básicos a serem representados por qualquer *feature* a ser criada no sistema, que, no caso são (Tabela 3.4):

Tabela 3.4 - Atributo das *features*.

ATRIBUTOS DA *feature*
Nome :
Posição : interno/externo
Direção : axial/radial
Sentido : direito/esquerdo
Centro de Rotação : CoordY
Face Inicial CoordI
Face Final : CoordF
Volume : positivo/negativo

Esses atributos possuem significado especial e podem ser interpretados como:

- o *Nome* corresponde a um *string* que servirá de identificador no sistema, devendo ser único;

- a *Posição* interno/externo para uma *feature* modificadora permite identificar se ela está aplicada sobre uma *feature* de volume positivo ou negativo;

- a *Direção* vai descrever, no caso da *feature* básica eixo cilíndrico, uma característica inerente a ela, que é um elemento axial, pois a mesma não está referenciada a nenhuma outra *feature*. Para o caso das outras *features* descritas no sistema, como furo cilíndrico, este pode ser de direção axial ou radial, dependendo de sua aplicação com relação à *feature* básica eixo. Para as *features* modificadoras, também a direção axial/radial no caso de aplicação sobre furos radiais;

- o *Sentido* é definido com relação à face inicial, se a *feature* se desenvolve à esquerda ou à direita da face inicial;

- com relação aos atributos de posição, *CoordY* corresponde à posição do eixo de rotação da respectiva *feature*, que é paralela ao eixo horizontal do sistema de coordenadas absoluto;

- *CoordI* e *CoordF* correspondem à posição no eixo X das faces da *feature*.

3.5.4 - RELAÇÕES INTER-*FEATURES*

Lima (1994) define formas pelas quais as *features* são interligadas para que resultem na representação de uma peça. Em sua definição, existem basicamente três tipos de relações: 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 à qual essa *feature* pertencerá, como um sistema de coordenadas absoluto; a relação de Adjacência trata da justaposição e contigüidade das *features* que compõem uma peça, tanto em nível geométrico quanto de processos de fabricação; por último, a relação de Dependência trata das intersecções existentes entre *features* negativas e positivas. Além disso, o autor utiliza uma matriz de transformação homogênea para definir o posicionamento e a orientação espacial de cada primitiva sólida

Por ser o presente sistema de representação 2D e não 3D como o apresentado em Lima (1994), algumas dessas características foram simplificadas para a representação, como, por exemplo, o Posicionamento, que, nesse caso, precisa somente do conhecimento de duas coordenadas. Além disso, é utilizado somente o sistema de coordenadas absoluto, sendo as relações de adjacência garantidas em função das coordenadas das respectivas faces das *features,* tanto externas como internas.

As relações de dependência são claramente explicitadas através da Figura 3.20, a qual representa a hierarquia existencial das *features*. Assim, com base nos conceitos desenvolvidos por Lima (1994), as *features* básicas externas são unidas por justaposição através das respectivas faces, tendo o eixo de rotação da *feature* coincidente com a direção da *feature* anterior, não sendo permitida a intersecção de duas *features* básicas externas.

Também para as *features* básicas internas, essas são unidas por justaposição através das respectivas faces; entretanto, uma *feature* básica interna será justaposta à face final de outra *feature*, não sendo permitida a justaposição de duas faces iniciais.

Assim, algumas regras com relação às *features* básicas citadas por Lima (1994) são:

a) Sempre existe uma relação de dependência entre uma *feature* de volume negativo e as *features* de volume positivo que ela intersecta.

b) 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.

c) Relações de adjacência vazias não são permitidas.

d) Não pode existir intersecção entre duas *features* de volume positivo.

e) Não existe relação de dependência entre duas *features* de mesma natureza (positivas ou negativas).

f) O posicionamento de uma *feature* negativa é sempre perpendicular a uma face plana de uma *feature* positiva.

Das regras enunciadas, pode-se dizer que, para o presente sistema, na regra "d", não pode ocorrer a intersecção entre duas *features* básicas de volume positivo ou negativo, ou seja, de mesma natureza. Com relação à regra "f", pode-se estendê-la de forma que o posicionamento de uma *feature* básica de volume negativo seja perpendicular a uma face plana ou a uma superfície curva de uma *feature* positiva, referindo-se ao caso de um furo perpendicular a um eixo.

As relações entre as *features* básicas eixo e furo não são traduzidas explicitamente através de atributos, mas de acordo com a estrutura de dados, através da qual é possível localizar os elementos e obter essas informações das respectivas relações em função das operações a serem realizadas sobre as *features* envolvidas.

3.5.5 - RELAÇÕES INTRA-*FEATURES*

As relações intra-*features* correspondem às condições para que possam ser inseridas as *features* modificadoras, o que também é regido pelas condições de Posicionamento, Adjacência e Dependência descritas anteriormente. Desse modo a condição de Posicionamento de uma *feature* modificadora deve estar relacionada ao Posicionamento da respectiva *feature* básica sobre a qual produz a modificação. Essa *feature* modificadora está referenciada à *feature* básica. Por exemplo: para inserir um chanfro, deve-se efetuar o posicionamento da sua face inicial com relação a uma das faces do eixo ou do furo.

As relações de adjacência estão relacionadas não somente às condições intra-*features*, mas também às condições das relações inter-*features*, o que pode ser observado quando da inserção de uma *feature* modificadora chanfro sobre um eixo. A face inicial de uma *feature* chanfro sempre será coincidente com uma das faces de uma *feature* eixo à qual foi aplicada.

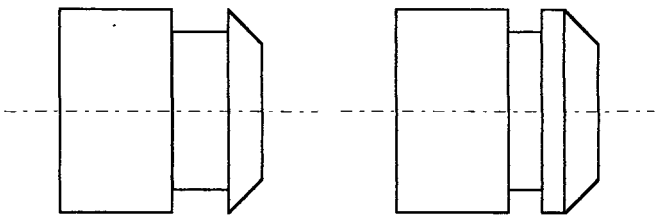
A face desse eixo, porém, pode ter adjacência com outra *feature* eixo (Figura 3.24). Desse modo, quando da inserção de uma *feature* chanfro, além da verificação das condições internas da *feature* básica (i.e., verificação da existência de uma outra *feature* modificadora de

mesma natureza), é necessário verificar as inter-relações com as outras *features* básicas que possam existir. Assim, todas as *features* modificadoras estão sujeitas às relações inter-*features* em maior ou menor grau de envolvimento.

a)Relações intra-*features* b)Relações intra e inter-*features*

Figura 3.24 - Relações intra e inter-*features*.

Uma relação intra-*feature* pode referir-se ao posicionamento de uma *feature* modificadora chanfro com relação a uma *feature* modificadora ranhura normal, cuja interferência entre elas pode não ser admitida como no caso da Figura 3.25 (a), pois, devido à funcionalidade, não é admitida a justaposição dessas duas *features* modificadoras. A Figura 3.25(b) ilustra a condição correta, sendo necessária a existência de uma porção de eixos entre as duas *features* (caso da aplicação de um anel de retenção).



a) disposição errônea b) disposição correta

Figura 3.25 - Disposição de duas *features* modificadoras.

A relação de dependência entre as *features* modificadoras e básicas é bem clara. Uma *feature* modificadora não pode existir sem que exista uma *feature* básica, que pode ser um furo ou um eixo. Assim, uma *feature* modificadora externa depende da existência de uma *feature* básica externa; por outro lado, uma *feature* modificadora interna depende da existência de uma *feature* básica interna, a qual, por sua vez, depende da existência de uma *feature* básica externa.

As *features* modificadoras de mesma natureza não podem se intersectar, nem intersectar *features* de outra natureza e posição. Por exemplo, uma *feature* chanfro externa não pode intersectar uma *feature* furo (Figura 3.26).

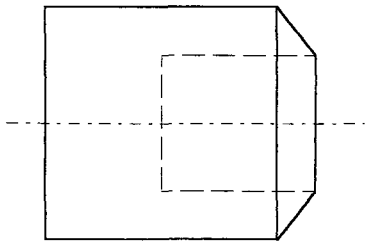


Figura 3.26 - Intersecção entre duas *features*.

Essas questões voltarão a ser analisadas com mais ênfase quando da apresentação das operações do sistema sobre *features*, como excluir e alterar.

Dentre as relações intra-*features*, podem surgir alguns casos a serem definidos em função de sua ocorrência; em outras palavras, em função da localização de uma *feature* modificadora, essa poderá receber uma nova denominação para referir-se ao seu posicionamento com relação às *features*. No presente trabalho, toda *feature* modificadora que é aplicada diretamente sobre uma *feature* básica continua sendo denominada de *feature* modificadora, como, por exemplo, a aplicação de um chanfro na extremidade de um eixo, ou de uma ranhura sobre a superfície cilíndrica de um eixo.

Uma particularidade surge quando da aplicação de uma *feature* modificadora, unindo uma *feature* básica com uma outra modificadora. A essa *feature* que efetua a ligação será denominada de *feature* modificadora de primeira ordem, o que pode ser observado na Figura 3.27, onde se tem a representação de uma ranhura chanfrada sobre um eixo.

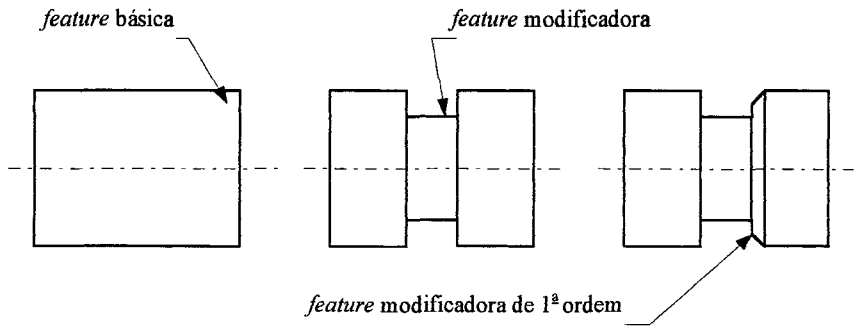


Figura 3.27- Exemplo de *feature* modificadora de primeira ordem.

A aplicação de uma *feature* modificadora na constituição de uma nova *feature* modificadora irá resultar na denominação de uma *feature* modificadora de segunda ordem. Isso pode ser visto na Figura 3.28, onde se tem a inserção de uma concordância internamente a uma *feature* modificadora ranhura.

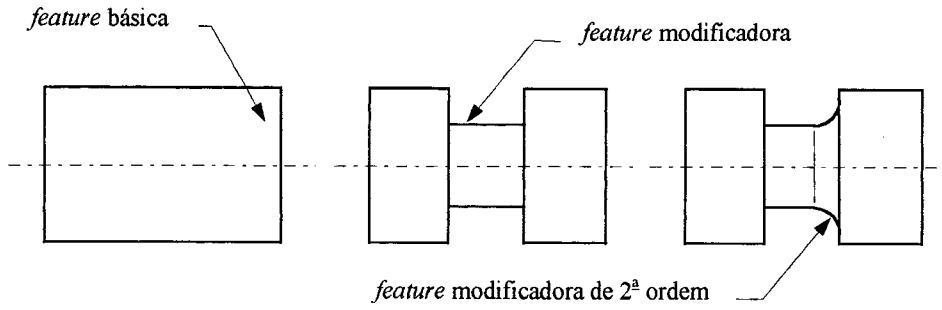


Figura 3.28 - Exemplo de *feature* modificadora de segunda ordem.

3.6 - DEFINIÇÃO DAS CLASSES DO SISTEMA

De acordo com a metodologia baseada em objetos para aplicação sobre sistemas computacionais, as classes foram definidas em função das suas relações entre objetos utilizados dentro de um contexto de projeto. Aplicaram-se as estruturas "generalização-especialização" e "todo-parte" (COAD 1992), com as quais, através de uma simbologia adequada (Figura 3.29), é possível representar a estrutura de dados do sistema desenvolvido.

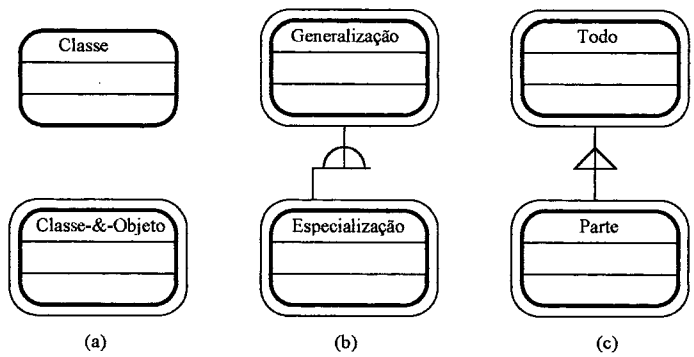


Figura 3.29 - Simbologia para representação para análise baseada em objetos.

Na Figura 3.30, é representada a estrutura baseada em objetos, onde se tem a relação de composição[3] como relação predominante na definição da estrutura do produto.

Dessa forma, tem-se um Produto que é composto (tem um) por um Conjunto, o qual é composto (tem um) por Subconjuntos, que, por sua vez, são compostos (tem um) por Peças, essas compostas (tem um) por *Features*. Uma *feature* é apenas definida como uma classe que serve para definição de outras classes.

[3] É uma estrutura tipo "todo-parte".

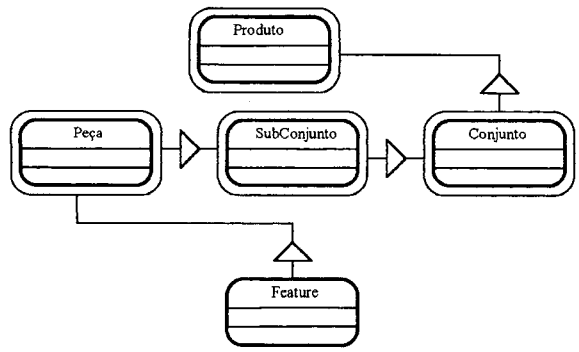


Figura 3.30 - Representação gráfica da estrutura do produto baseada em objetos.

Na representação da Figura 3.30, tem-se uma estrutura do tipo "todo-parte". A definição de *feature* é considerada mais genérica e, por isso, é definida apenas como uma classe, não sendo possível instanciar um objeto da mesma. Assim, a partir de *feature*, tem-se a construção do restante da estrutura do sistema, utilizando-se o tipo "generalização-especialização". As principais classes do modelo estão representadas na Figura 3.31.

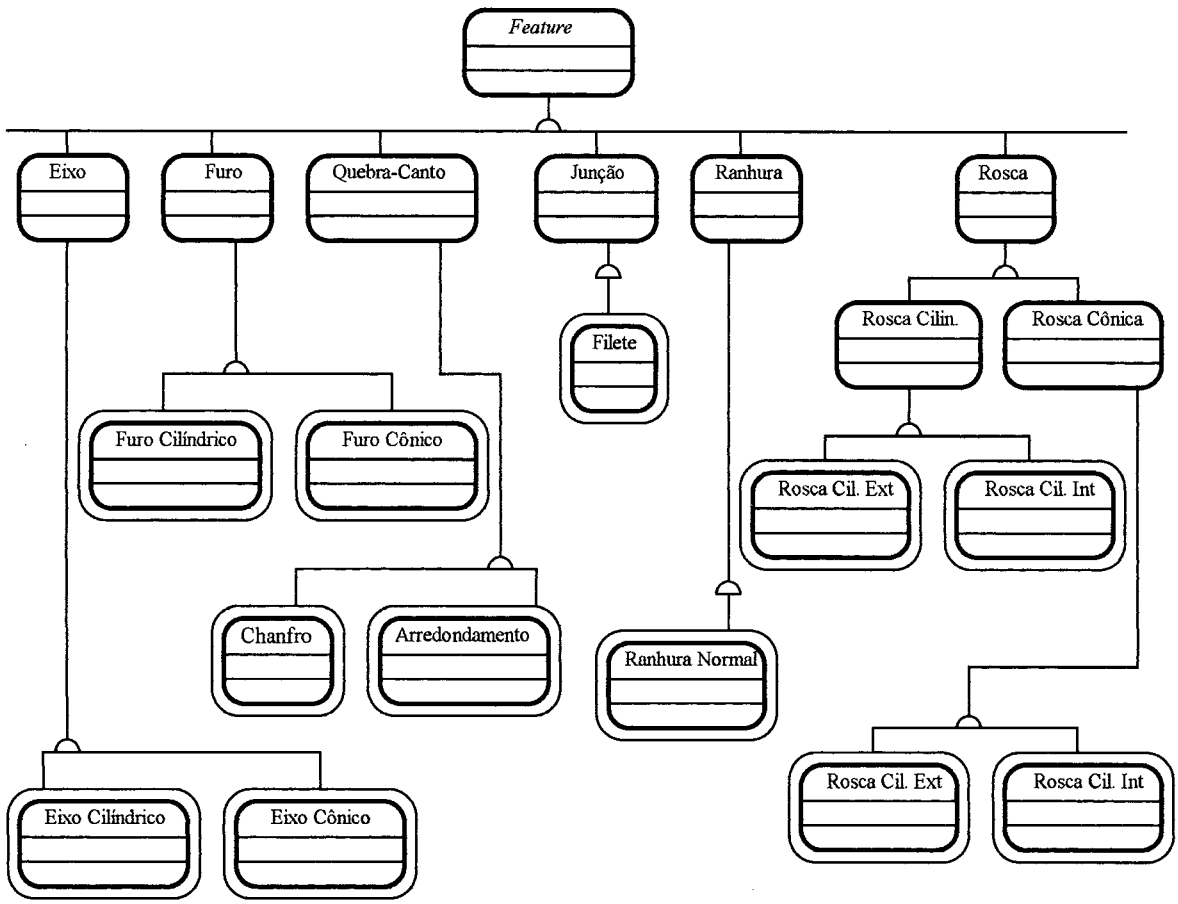


Figura 3.31 - Representação das relações entre as classes e objetos.

Na Figura 3.32, estão descritas as classes que representam as *features* de forma.

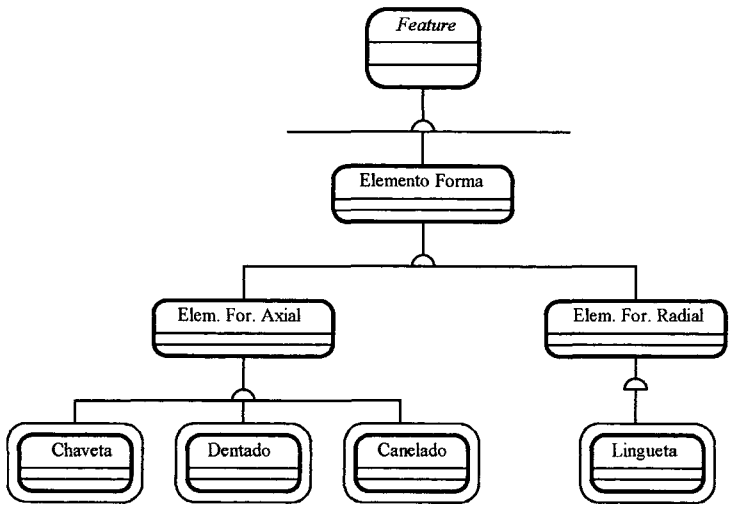


Figura 3.32 - Representação das relações entre as classes.

A seguir, descrevem-se as classes do modelo de modo literal, com sua classificação no caso das *features*, de acordo com a classificação descrita no item 3.5.1.

PRODUTO
CONJUNTO
SUBCONJUNTO
PEÇA
FEATURE
(*features* básicas)
- **EIXO**
  - EIXO_CILÍNDRICO
  - EIXO_CÔNICO
- **FURO**
  - **FURO_CILÍNDRICO**
    - FURO_CILÍNDRICO_CEGO
    - FURO_CILÍNDRICO_PASSANTE
  - **FURO_CÔNICO**
    - FURO_CÔNICO_CEGO
    - FURO_CÔNICO_PASSANTE

(*features* modificadoras de aresta)
- **QUEBRA_CANTO**
  - **CHANFRO**
  - **ARREDONDADO**
- **JUNÇÃO**
  - FILETE

(*features* modificadoras de superfície)
- **RANHURA**
  - RANHURA_NORMAL
- **ROSCA**
  - **ROSCA_CILÍNDRICA**
    - ROSCA_CILÍNDRICA_EXTERNA
    - ROSCA_CILÍNDRICA_INTERNA

ROSCA_CÔNICA
ROSCA_CÔNICA_EXTERNA
(*features* modificadoras prismáticas)
ELEMENTO_FORMA
ELEMENTO_FORMA_AXIAL
CHAVETA
DENTADO
CANELADO
ELEMENTO_FORMA_RADIAL
LINGUETA

3.7 - FUNÇÕES AUXILIARES DO MODELADOR GRÁFICO

As funções auxiliares têm por objetivo dar apoio ao modelador de forma a preencher os requisitos necessários para a manipulação das entidades. Para isso, foram selecionadas algumas funções básicas a serem propostas no sistema FeatCAD-2D, que são: instanciar, excluir, alterar e mover.

3.7.1 - FUNÇÃO INSTANCIAR

A função instanciar tem por objetivo inserir uma *feature* (que pode ser um conjunto, subconjunto, peça ou uma *feature* qualquer) na estrutura do produto. A *feature* a ser instanciada corresponda uma entre aquelas definidas na biblioteca de *features*.

Através da operação instanciar, é inserida a *feature* desejada de acordo com os parâmetros passados via interface gráfica. Essa operação também providencia a atuação do motor de inferência do sistema especialista, a inserção das informações na estrutura de dados, bem como a atualização de informações já existentes na estrutura de dados.

O sistema permite a identificação das *features* já existentes de modo a não haver a duplicidade de informações, que pode levar à inconsistência da peça, subconjunto ou conjunto. Por exemplo, a inserção de uma *feature* modificadora chanfro no local em que já existe uma *feature* de mesma natureza (outra *feature* chanfro) corresponde a uma inconsistência.

3.7.1.1 - FUNÇÃO INSTANCIAR CONJUNTO, SUBCONJUNTO E PEÇA

Conforme a definição do modelo do produto, este é composto por um conjunto, subconjuntos, peças e *features*. De acordo com a ordem de precedência, inicialmente, deve ser criado um conjunto, que, na realidade, corresponde ao produto, sendo preeenchidas as respectivas informações. A seguir, deve ser criado o subconjunto, o qual pertence ao conjunto previamente definido, sendo, do mesmo modo, passadas as respectivas informações. Como um subconjunto é composto por várias peças, essas são criadas em seqüência, informando os respectivos atributos. Assim, o sistema FeatCAD-2D direciona a instanciação do produto obedecendo à hierarquia descrita, o que deve ser obtido através da interface gráfica, que é construída de modo a obrigar o usuário a seguir essa seqüência inicial para preencher os requisitos de um produto, sendo após liberada a criação de qualquer *feature* que desejar.

Dessa forma, a estrutura de dados é inicializada de forma adequada para permitir a sua expansão de acordo com as necessidades de desenvolvimento do produto. Após a criação do conjunto e do primeiro subconjunto, o sistema permite a criação tanto de uma peça para esse subconjunto como de um novo subconjunto, possibilitando a criação de novas entidades aleatoriamente.

3.7.1.2 - FUNÇÃO INSTANCIAR UMA *FEATURE*

A operação instanciar uma *feature* corresponde especificamente à inserção por parte do usuário de uma *feature* para compor uma peça. A escolha da *feature* é feita através da interface gráfica que apresenta as *features* disponíveis na biblioteca. A partir da escolha feita pelo usuário e dos parâmetros informados para a *feature*, ele indica a posição onde deseja inserir a *feature*. De posse dessas informações, o sistema deve acionar o motor de inferência do sistema especialista para pesquisar a base de conhecimento a respeito da viabilidade da respectiva operação (algumas operações de instanciação podem necessitar, também, da consulta ao banco de dados de manufatura).

A partir da resposta do sistema especialista, a *feature* desejada pode ou não ser inserida. Caso a resposta seja positiva, a *feature* é inserida; do contrário, o sistema deve informar ao usuário sobre a impossibilidade, bem como o motivo da negativa da inserção da *feature* naquele

contexto. A análise realizada no sistema especialista corresponde à verificação da situação em que deseja inserir a *feature*.

(a) - INSTANCIAR UMA *FEATURE* BÁSICA

No sistema FeatCAD-2D, a primeira *feature* a ser instanciada deve ser uma *feature* básica de volume positivo, ou seja, deve possuir material para que, a seguir, possam ser instanciadas as *features* básicas de volume negativo e as modificadoras, quando for o caso. A seguir, descreve-se a metodologia de operações de inserção das *features* básicas eixo cilíndrico e furo cilíndrico, bem como suas restrições no sistema FeatCAD-2D.

(a1) - INSTANCIAR UMA *FEATURE* BÁSICA EIXO CILÍNDRICO

Como o domínio do sistema FeatCAD-2D corresponde ao das peças rotacionais, um dos critérios é que os centros de rotação das *features* estejam alinhados na composição de uma peça. Assim, cada *feature* eixo instanciada possui o centro de rotação coincidente com relação ao centro de rotação da peça (Figura 3.33(a)). Na Figura 3.33(b), está representada uma forma de justaposição de *features* eixo, onde o centro de rotação das *features* eixo não está alinhado.

Da mesma forma que dois corpos não podem ocupar a mesma posição no espaço, duas *features* eixo não podem ser sobrepostas, sendo somente permitida a justaposição, isto é, serem colocadas lado a lado, tendo por contato as faces dos respectivos eixos (Figura 3.33(c)).

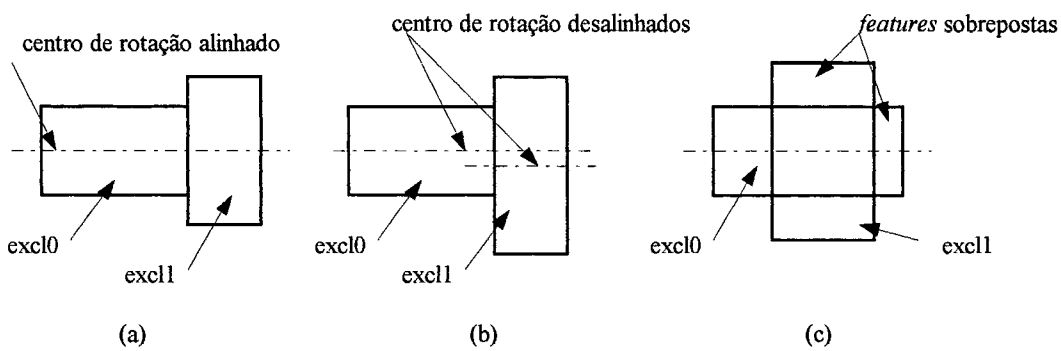


Figura 3.33 - Disposição das *features* eixo.

Como as *features* devem estar alinhadas pelo centro de rotação, não é permitida a inserção de uma *feature* eixo perpendicular a outra *feature* eixo, o que pode ser observado na Figura 3.34.

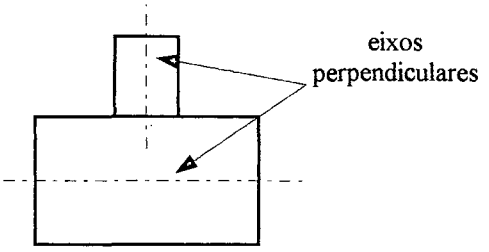
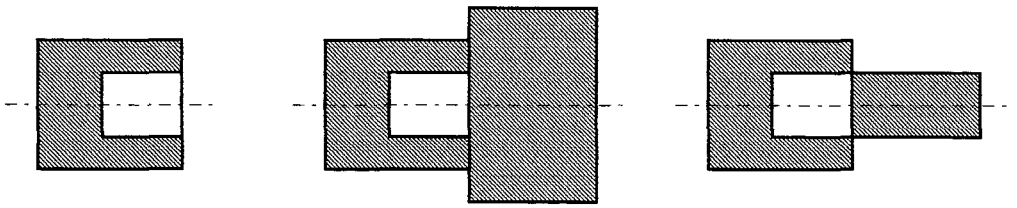


Figura 3.34 - Eixos perpendiculares.

A existência de uma *feature* furo numa das extremidades da peça impede a inserção de um eixo nessa extremidade, pois a situação inviabiliza a existência de um furo, o qual não possui comunicação com o exterior. Na Figura 3.35(a), está representada a situação inicial de uma *feature* eixo com uma *feature* furo cego. Na Figura 3.35(b) e (c), tem-se a inserção de uma *feature* eixo obstruindo a *feature* furo, situação que o sistema não aceita sob hipótese alguma.



(a) Condição inicial (b) Impossível c) Impossível

Figura 3.35- Inserção impossível de um eixo.

A inserção de uma *feature* eixo também pode ter restrições quando da existência de *features* modificadoras do tipo arestas e de face aplicadas a outros eixos. Como exemplo, pode-se observar na Figura 3.36 o caso da inserção de uma *feature* eixo, quando já existe uma *feature* modificadora de aresta do tipo chanfro. Na Figura 3.36(a), o diâmetro da *feature* eixo é menor que o diâmetro da face resultante após a aplicação do chanfro, a inserção é compatível.

Na Figura 3.36(b), o diâmetro da *feature* eixo a ser inserida é maior que o diâmetro resultante após a inserção da *feature* chanfro, não sendo permitida a inserção da *feature* eixo a não ser que a *feature* chanfro seja eliminada.

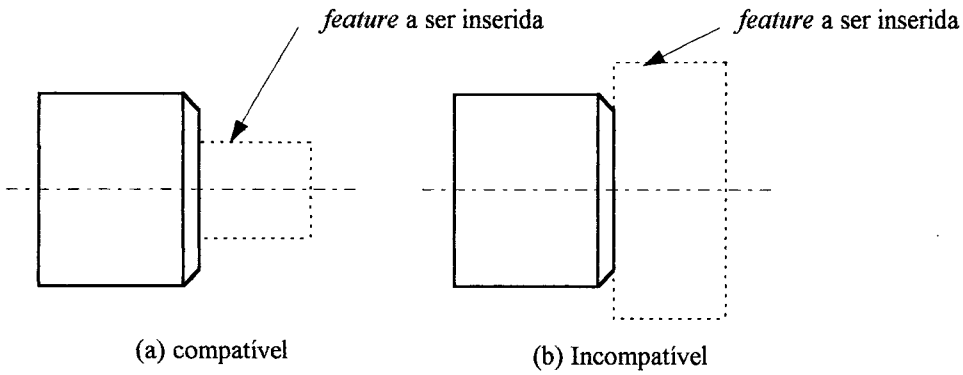


Figura 3.36 - Inserção de *feature* eixo existindo uma *feature* chanfro.

(a2) - INSTANCIAR UMA *FEATURE* BÁSICA FURO CILÍNDRICO

Pela concepção do sistema FeatCAD-2D, uma *feature* furo cilíndrico é uma *feature* de volume negativo que somente pode existir se houver uma *feature* de volume positivo. Assim, fica impossível instanciar uma *feature* furo cilíndrico sem que haja uma *feature* eixo cilíndrico.

Mesmo havendo uma *feature* eixo cilíndrico, a instanciação de uma *feature* furo deve estar localizada sobre a *feature* eixo, pois, do contrário, é impossível inserir um furo onde não existe material. Para localizar uma *feature* furo cilíndrico, deve ser escolhida uma região onde exista um eixo. Na Figura 3.37, pode-se observar o ponto de localização correto, pois o mesmo está sobre a *feature* eixo; e também o ponto de localização incorreto, que está fora da *feature* eixo.

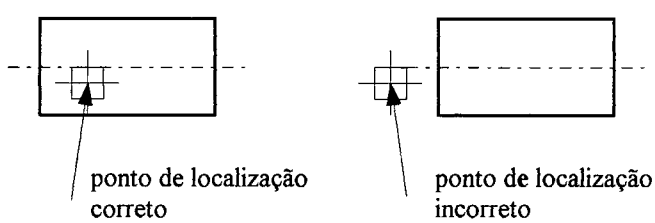


Figura 3.37 - Localização de uma *feature* furo.

A inserção de uma *feature* furo cilíndrico é sempre feita a partir da extremidade da peça, ou seja, no caso de um furo escalonado, não pode ser inserido o segundo furo antes da execução do primeiro (Figura 3.38).

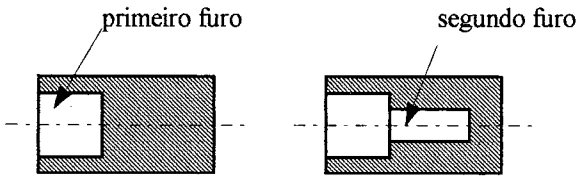


Figura 3.38 - Execução de uma *feature* furo cilíndrico escalonado.

Quando da aplicação de *features* furo, o sistema FeatCAD-2D identifica a região onde um furo pode ser inserido e se há espaço para a sua inserção. Na Figura 3.39(a), tem-se a representação de um eixo com um furo cego, sendo que a região disponível para a inserção de outro furo corresponde à região à direita do furo existente. Se o ponto de localização estiver fora dessa região, o furo não pode ser inserido.

Na Figura 3.39(b), há um furo cego e um furo passante. Nessa representação, o sistema deve identificar a inexistência das condições para a inserção de um novo furo, pois não existe mais espaço para uma nova *feature* .

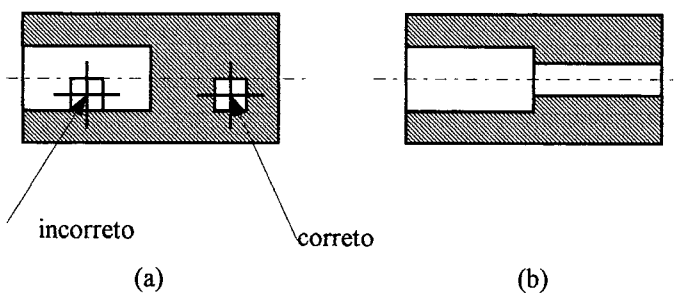


Figura 3.39 - Localização de *features* furo.

Na Figura 3.40, estão representadas as possibilidades de existência de *features* furo com relação a uma única *feature* eixo. Na Figura 3.40(a), tem-se uma *feature* furo cilíndrico cego à direita; em (b), uma *feature* furo cilíndrico passante e, em (c), uma *feature* furo cilíndrico cego à esquerda, situações perfeitamente aceitáveis no mundo real e descritas nas classes das *features*.

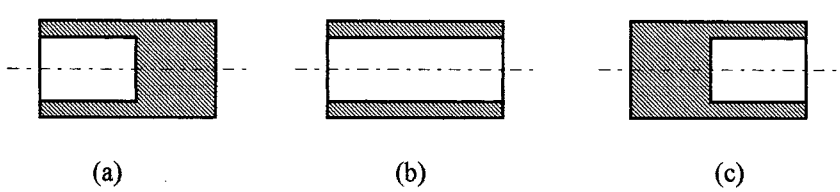
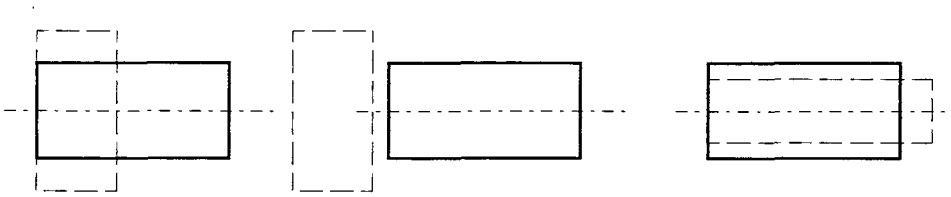


Figura 3.40 - Inserção de furos em eixos.

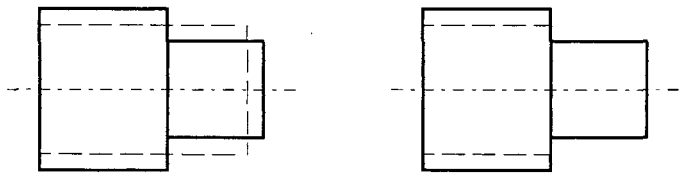
Dentre as situações inaceitáveis pelo sistema FeatCAD-2D, tem-se, na Figura 3.41(a), a representação de uma *feature* furo cego, que possui o diâmetro maior que o diâmetro da *feature* eixo ou, em (b), uma *feature* furo cego que está localizada fora da *feature* eixo, ou seja, da região onde não há material. Ainda na Figura 3.41(c), tem-se o caso de uma *feature* furo passante em que o comprimento é maior que o da *feature* eixo, que, no caso, é a própria peça.



(a) (b) (c)

Figura 3.41 - Situações de furos impossíveis.

Há algumas outras condições que tornam impossível a existência de um furo, por exemplo, quando há mais de uma *feature* eixo cilíndrico na composição de uma peça e uma *feature* furo cria interferências indesejadas. Na Figura 3.42(a), pode ser observado o caso da inserção de uma *feature* furo passante onde ocorre a interferência de diâmetro com uma das *features* eixo que compõem a peça. Na Figura 3.42(b), há interferência de uma *feature* furo cego com uma *feature* eixo, sendo que o fundo do furo provoca a inexistência de material para suportar a *feature* eixo.



(a) (b)

Figura 3.42 - Situações de interferência.

Há um caso em que a inconsistência da operação de inserção pode produzir uma transformação de *feature*. Isso ocorre quando da escolha de uma *feature* furo cego, que possui a profundidade igual ao comprimento da peça, devendo, nesse caso, haver uma transformação de *feature*, ou seja, o furo cego é, na realidade, um furo passante devido à condição encontrada.

No sistema FeatCAD-2D, podem ser inseridos furos que não possuam o eixo de rotação coincidente com o eixo de rotação da peça, que, no caso, correspondem às *features* padrão circular, compostas por furos cilíndricos passantes (também podem ser furos cilíndricos cegos), como está representado na Figura 3.43.

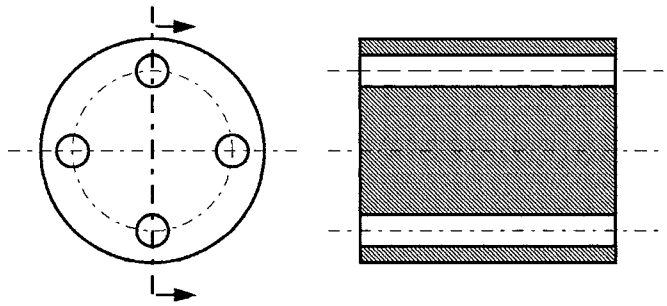
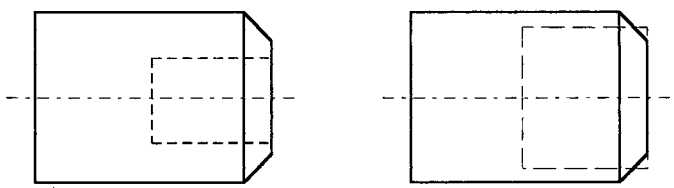


Figura 3.43 - Padrão circular - furos excêntricos.

Quando da inserção de uma *feature* furo, essa pode sofrer uma interferência com relação a uma *feature* modificadora existente. Essas *features* modificadoras podem ser de aresta (chanfro, arredondamento, etc.), de superfície (ranhura, rosca, etc.) ou uma *feature* modificadora de face (rebaixo para anel O'ring). Na Figura 3.44(a), é apresentado um caso em que não ocorre a interferência com uma *feature* chanfro; já, na Figura 3.44(b), a *feature* furo produz uma interferência com uma *feature* chanfro aplicada a um eixo. Nesse caso, a solução consiste em diminuir o diâmetro do furo ou, então, em alterar ou eliminar o chanfro.



(a) (b)

Figura 3.44 - Interferência entre *feature* furo e chanfro.

(b) - INSTANCIAR UMA *FEATURE* MODIFICADORA

A instanciação de uma *feature* modificadora pressupõe a existência de uma *feature* básica, seja ela de volume positivo ou negativo, que podem ser denominadas de *features* básicas

externa e interna. Assim, uma *feature* modificadora é denominada de *modificadora externa ou interna* em função da *feature* básica sobre a qual foi aplicada. Em função do tipo de *feature* modificadora, essa possui um local de aplicação que segue de acordo com a classificação das *features* realizada anteriormente. Assim, têm-se *features* modificadoras de aresta, de superfície e de face.

Para que a instanciação de uma *feature* modificadora possa ser realizada, primeiramente, o usuário deve selecionar a *feature* básica sobre a qual será aplicada a *feature* modificadora. Desse modo, o sistema pode identificar a *feature* desejada, se é de volume positivo ou negativo, além de identificar as *features* vizinhas.

(b1) - *FEATURE* MODIFICADORA DE ARESTA - CHANFRO

A *feature* modificadora de aresta, denominada de *chanfro*, pode ser caracterizada por ser uma *feature* modificadora de volume negativo, pois retira material das *features* básicas para a obtenção da sua forma.

Uma *feature* modificadora de aresta - chanfro - pode ser aplicada sobre uma *feature* eixo ou furo. No caso da aplicação sobre uma *feature* eixo, pode-se identificar que o eixo possui somente dois locais que podem receber um chanfro, ou seja, nas suas extremidades, conforme a Figura 3.45.

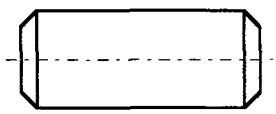


Figura 3.45 - Aplicação de uma *feature* modificadora chanfro.

As *features* modificadoras de aresta tipo chanfro, quando aplicadas sobre uma peça composta por vários eixos, podem resultar em algumas restrições que devem ser observadas para que uma *feature* chanfro seja compatível. Assim, na Figura 3.46, estão representadas algumas situações de inconsistência de um chanfro frente aos eixos. Na Figura 3.46(a), está o caso da aplicação de um chanfro onde o eixo vizinho (eixo à direita) possui um diâmetro maior, o que torna inconsistente a aplicação do chanfro nessa situação.

Na Figura 3.46(b), tem-se o caso do chanfro que produz uma interferência com o eixo vizinho (eixo à direita). Nesse caso, o chanfro também não pode ser inserido a menos que o ângulo ou o comprimento, ou ambos, sejam alterados para que seja obtida a peça representada na Figura 3.46(c), que é compatível.

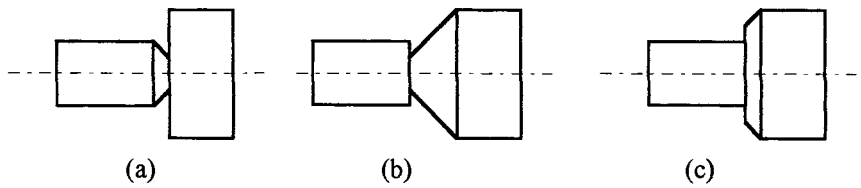


Figura 3.46 - Restrições à aplicação de chanfros em eixos.

A aplicação correta de chanfros sobre eixos pode ser observada na Figura 3.47, onde se observa que esses não produzem inconsistências nem interferências com as *features* vizinhas.

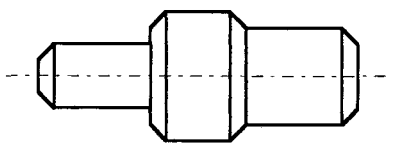


Figura 3.47 - Aplicação de chanfros a uma peça.

As *features* modificadoras de aresta também podem ser aplicadas sobre *features* básicas furos cilíndricos. Para os furos, ocorrem dois casos distintos de aplicação de chanfros: 1) sobre um furo cego e 2) sobre um furo passante.

Para um furo cego, o chanfro somente pode ser aplicado na face de entrada do furo, ou seja, na face inicial (Figura 3.48(a)). No caso de um furo passante, esse permite a aplicação do chanfro em ambas as extremidades, ou seja, na face inicial e na face final do furo (Figura 3.48(b)).

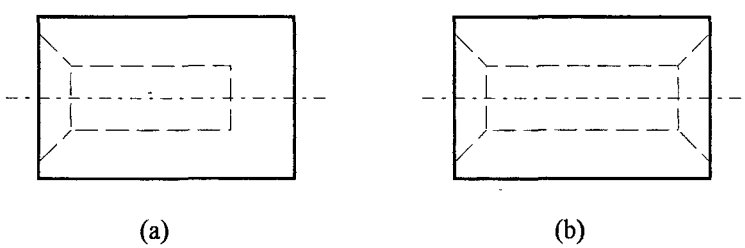


Figura 3.48 - Aplicação do chanfro em furo cego e furo passante.

No caso em que o ângulo do chanfro é 45° e em que esse tem outras funções que não apenas quebrar os cantos (eliminar as arestas vivas da peça), o chanfro passa a se chamar de *escareado*. Nesse caso, a sua função pode ser de alojar a cabeça de um parafuso de cabeça escareada (Figura 3.49).

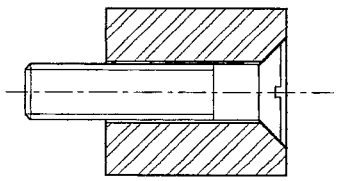
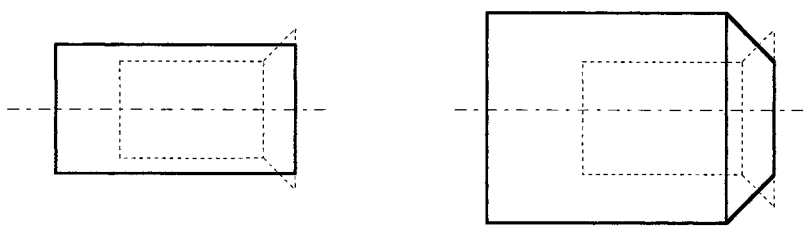


Figura 3.49 - Chanfro e escareado.

As dimensões dos chanfros aplicados sobre furos possuem como restrição as dimensões dos respectivos eixos. Essa restrição passa a ser a interferência produzida pelo chanfro sobre o eixo ou sobre uma outra *feature* modificadora, que também pode ser um chanfro.

Na Figura 3.50(a), tem-se a inserção de uma *feature* modificadora chanfro que possui um ângulo e comprimento tal que produz uma interferência com o eixo, o que leva à inconsistência da *feature* eixo (altera as dimensões da *feature*).

Caso semelhante ocorre quando o chanfro aplicado ao furo produz uma interferência com um chanfro existente sobre o eixo (Figura 3.50(b)), levando à inconsistência da *feature* modificadora chanfro, o que também é incompatível com a lógica do que representa um chanfro.



(a) (b)

Figura 3.50 - Interferência produzida pelo chanfro.

(b2) - *FEATURE* MODIFICADORA DE SUPERFÍCIE - RANHURA

A *feature* modificadora de superfície, denominada de *ranhura*, é caracterizada por ser uma *feature* modificadora de volume negativo, pois retira material das *features* básicas.

A aplicação de uma *feature* modificadora de superfície - ranhura - pode ser feita sobre uma *feature* eixo ou furo. No caso da aplicação sobre uma *feature* eixo, tem-se a sua localização numa região internamente à superfície da *feature* (Figura 3.51), devendo essa *feature* sempre estar localizada a uma determinada distância da face do eixo (que é função da aplicação).

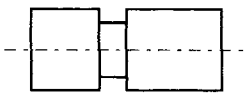


Figura 3.51 - Aplicação de uma *feature* modificadora - ranhura.

Na Figura 3.52(a), tem-se um eixo onde a colocação da ranhura coincidente com a face do eixo produz uma inconsistência, não permitindo a existência da face da ranhura do lado esquerdo. A Figura 3.52(b) mostra o caso anterior representado do lado direito da *feature* eixo. Em (c), tem-se a representação correta, pois, entre a face da *feature* eixo e a face da *feature* ranhura, deve haver uma porção mínima de material para caracterizar a *feature* modificadora. A dimensão exigida (Figura 3.52(c)) é função do tipo de aplicação.

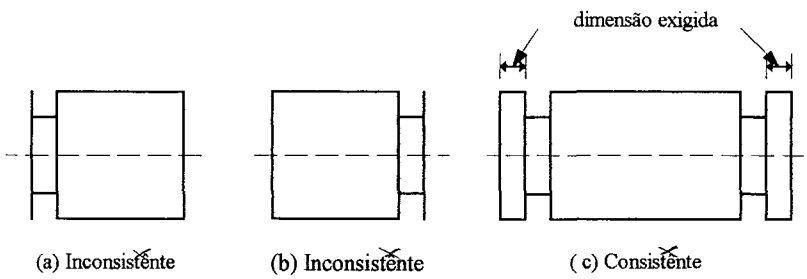


Figura 3.52 - Condição de existência de *features* ranhura.

As interferências que podem ocorrer com uma *feature* ranhura podem ser com relação à *feature* furo, ou seja, quando da inserção de uma *feature* ranhura, o seu diâmetro interno não pode ser menor que uma determinada dimensão para que seja mantida uma quantidade mínima de material. Na Figura 3.53, pode-se observar o caso da interferência com um furo, o que deve ser evitado através do sistema especialista, que providencia a verificação dos dados.

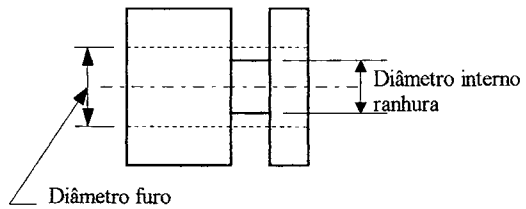


Figura 3.53 - Interferência da *feature* ranhura com uma *feature* furo.

A aplicação de uma ranhura em uma *feature* eixo que já possua uma *feature* modificadora de aresta, como um chanfro, pode produzir inconsistências na sua representação, o que é evitado com a aplicação do conceito representado na Figura 3.52, onde se observa que a face do chanfro mantém uma determinada distância da face da *feature* eixo mais próxima. No caso de um chanfro, essa distância deve ser mantida considerando a face do chanfro (ver item 3.5.3), que não é aquela que coincide com a face da *feature* eixo. Na Figura 3.54(a), está representada a opção incorreta e, em (b), a opção a ser utilizada.

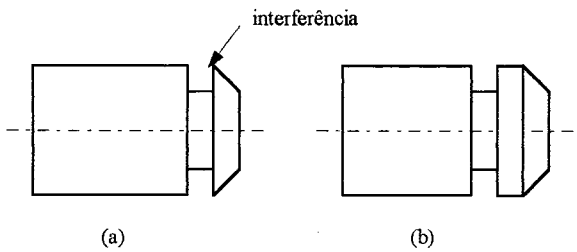


Figura 3.54 - Interferência entre ranhura e chanfro.

Quando aplicada a *feature* ranhura numa *feature* furo, a interferência a ocorrer é com relação ao diâmetro da *feature* eixo que as contém (Figura 3.55(a)), como também com relação ao seu posicionamento, o qual não pode estar junto às faces da *feature* furo (Figura 3.55(b) e (c)) nem de forma a interferir em outra *feature* que esteja aplicada na *feature* furo, ou na *feature* eixo, como acontece com a ranhura e o chanfro aplicados à *feature* furo (Figura 3.55(c)).

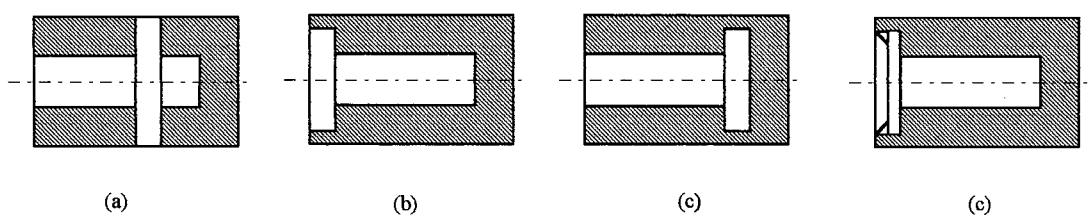


Figura 3.55 - Interferência entre *feature* ranhura e *features* furo e eixo.

3.7.2- FUNÇÃO EXCLUIR

Excluir corresponde a eliminar uma informação, que pode ser um conjunto, um subconjunto, uma peça ou uma *feature*. A operação consiste em eliminar as informações da estrutura de dados do sistema FeatCAD-2D e da base de dados do CAD (AutoCAD), eliminando, assim, a representação gráfica da referida informação.

Para que a operação seja coerente, é necessário definir o que deve ocorrer em cada caso de exclusão de uma entidade. A seguir, são especificadas as bases conceituais para o funcionamento dessa operação.

3.7.2.1 - FUNÇÃO EXCLUIR CONJUNTO, SUBCONJUNTO E PEÇA

Excluir um conjunto significa excluir toda a estrutura do produto, eliminando todo tipo de informação a respeito desse item. A operação de excluir um conjunto inicia-se pelas *features* modificadoras de cada peça, passando, então, às *features* básicas. Quando todas as *features* que compõem as peças dos subconjuntos estiverem excluídas, passa-se a excluir as peças; em seguida, os subconjuntos e, finalmente, o conjunto. Essa operação consiste em excluir todas as listas existentes no modelo do produto que se quer eliminar, inclusive as informações gráficas da base de dados do CAD.

Para excluir um subconjunto, o procedimento é semelhante ao descrito anteriormente para o conjunto, com a exceção de que, dentre os subconjuntos a serem excluídos, somente aquele especificado é que será eliminado da lista de subconjuntos, bem como toda estrutura abaixo dele. Excluir uma peça corresponde a eliminar uma determinada peça da lista de peças que pertence a um determinado subconjunto, abaixo do qual toda a estrutura de peças é eliminada, ou seja, todas as *features* e, inclusive, a própria referência da peça na lista do subconjunto.

3.7.2.2 - FUNÇÃO EXCLUIR *FEATURE*

A operação excluir uma *feature* corresponde a eliminar da estrutura de dados uma *feature* específica escolhida pelo usuário. Esse seleciona a *feature* desejada, e o sistema identifica o tipo de *feature*, escolhendo os procedimentos adequados para que complete a referida operação. Para isso, também é utilizado o sistema especialista, que possui uma base de conhecimentos para decidir sobre as operações.

Para excluir uma *feature*, é necessário identificar especificamente a *feature* a ser excluída. Há dois procedimentos básicos a serem tomados para executar a exclusão: a) excluir *features* modificadoras e b) excluir *features* básicas. Para cada caso, há procedimentos específicos.

Como há dois tipos de *features* distintas a serem excluídas, é utilizada uma função auxiliar que permite identificar se a *feature* selecionada é uma *feature* básica ou modificadora. Além disso, em cada uma delas, deve-se identificar especificamente, por exemplo, se é uma *feature* básica e se é uma *feature* furo ou eixo; no caso de modificadora, se a *feature* é um chanfro, uma ranhura ou um arredondamento, etc.

3.7.2.2.1 - FUNÇÃO EXCLUIR *FEATURE* MODIFICADORA

Excluir uma *feature* modificadora é eliminar somente a *feature* selecionada, não afetando as *features* modificadoras associadas à *feature* básica à qual a *feature* modificadora a excluir está associada.

Pratt (1990) relata o problema da implementação de algumas *features*, notavelmente com relação à exclusão, pois tal função apresenta dificuldades a não ser que o sistema possa lembrar o estado antes da criação da *feature*. Utiliza, para isso, um exemplo de Miner citado por Pratt (1990), visando mostrar as opções criadas para excluir a referida *feature*. Na Figura 3.56, Miner apresenta três opções possíveis de resultado da exclusão da *feature* ranhura em função do modelador "CSG" utilizado, e Pratt (1990) reforça a existência desse problema.

Com a aplicação do conceito de *features* modificadoras, uma *feature* ranhura é uma modificadora que é aplicada sobre uma *feature* eixo, a qual permanece intacta, ou seja, é mantida a memória da *feature* anterior. Assim, ao ser excluída a *feature* ranhura, a representação

anterior retorna de modo natural, ou seja, o eixo, que, no caso, é a representação (b) da Figura 3.56.

As outras duas alternativas propostas na Figura 3.56 (a) e (c) produzem, na realidade, uma alteração indesejada da *feature*, visto que, em (a), a *feature* eixo à direita muda o atributo comprimento para se adequar, enquanto, em (c), é realizada a transformação de uma *feature* em outra que não possui nenhuma relação com o que se deseja, que é somente eliminar a *feature* ranhura.

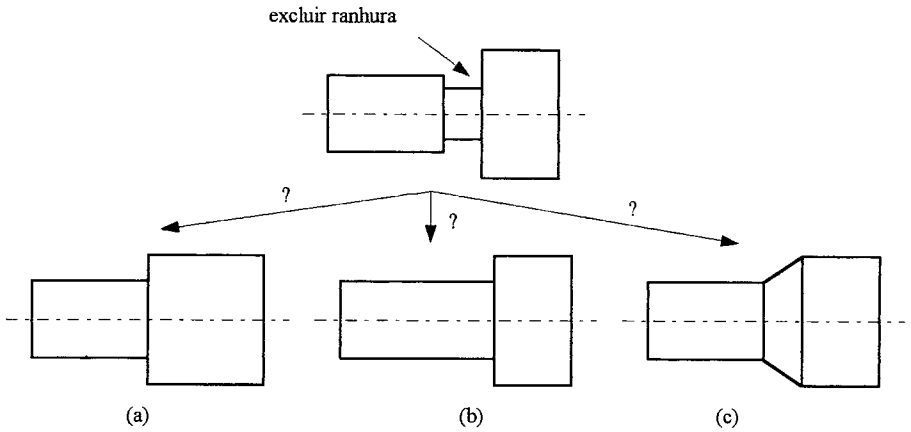


Figura 3.56 - Problema de ambigüidade na exclusão de *features* (Pratt, 1990)).

Como exemplo, ao excluir uma *feature* chanfro, somente ela deve ser eliminada, devendo o eixo ou o furo ao qual estava associada ser redesenhado sem a existência da *feature* eliminada (Figura 3.57).

A seqüência a ser seguida para a exclusão de uma *feature* chanfro é semelhante para qualquer outra *feature* modificadora, ilustrando-se, a seguir, essa operação:

a) escolher a função excluir *feature*;

b) selecionar a *feature* a ser excluída;

c) se é uma *feature* modificadora, então :

c1) apagar a *feature* selecionada do CAD;

c2) eliminar a entidade do CAD à qual está associada;

c3) reconstruir a *feature* básica à qual a *feature* modificadora estava associada;

c4) modificar os atributos da lista de dados da estrutura do FeatCAD-2D.

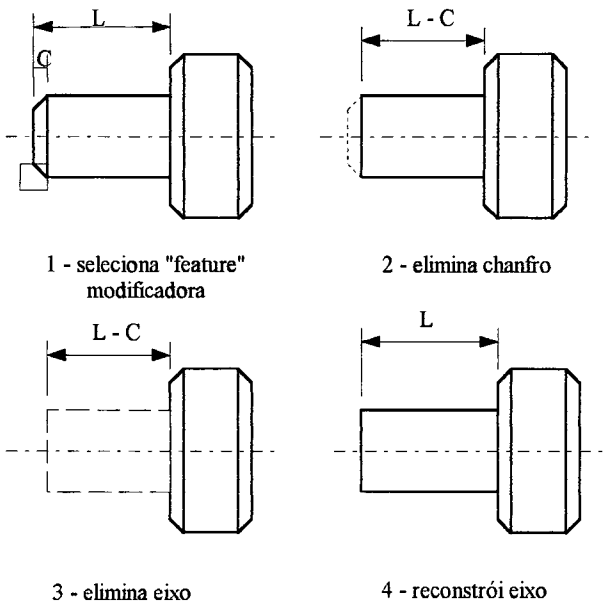


Figura 3.57 - Seqüência da exclusão de um chanfro.

3.7.2.2.2 - FUNÇÃO EXCLUIR *FEATURE* BÁSICA

No sistema FeatCAD-2D, excluir uma *feature* básica corresponde a eliminar uma *feature* eixo ou uma *feature* furo, bem como toda a estrutura das *features* modificadoras relacionadas a elas e outras *features* básicas envolvidas (p. ex., um furo interno a um eixo).

Para executar a operação de excluir uma *feature* básica, não basta somente identificar e eliminar as informações da estrutura de dados ou as informações gráficas da base de dados do CAD; é necessário também efetuar a verificação da consistência do resultado da eliminação dessa *feature*, isto é, se é excluída uma *feature* eixo que está no meio de uma peça, deve-se considerar o que acontecerá com o restante das *features* envolvidas na peça. Para resolver essas e outras questões, é necessário definir os procedimentos a serem adotados. Assim, nos próximos itens, especificam-se tais procedimentos.

(a) - EXCLUIR UMA *FEATURE* BÁSICA EIXO

Para excluir uma *feature* eixo, inicialmente, pode-se simplificar o problema abordando somente a existência de eixos numa peça. Desse modo, o problema se resume a excluir uma

feature eixo que está nas extremidades da peça ou no interior da mesma, ou seja, a peça possui duas *features* eixo que são suas vizinhas, uma de cada lado (Figura 3.58).

Para resolver o problema com relação à posição da *feature* a ser excluída com as suas vizinhas, é introduzido o conceito de plano de referência, o qual especifica qual das faces da *feature* a ser excluída deve permanecer na mesma posição e quais devem ser reposicionadas. Isso resulta em dois modos: a) excluir à direita e b) excluir à esquerda.

Uma exclusão é dita à direita quando, na *feature* a ser excluída, a face direita é considerada como superfície de referência. Após excluir-se a *feature*, aquela localizada à esquerda (podem ser várias *features*) da *feature* excluída sofre uma translação para a direita (Figura 3.58(a)). Excluir à esquerda, por sua vez, significa que a superfície de referência é a face esquerda da *feature*. As *features* localizadas à direita da selecionada sofrem uma translação para a esquerda (Figura 3.58(b)).

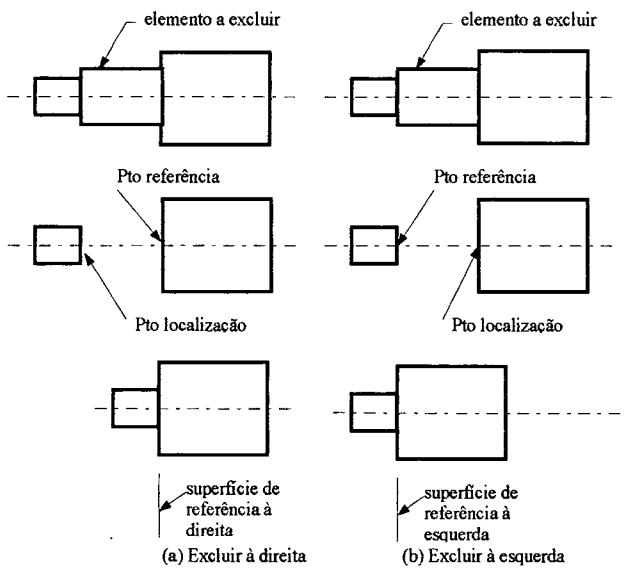


Figura 3.58 - Excluir *feature*.

As operações de exclusão descritas eliminam as *features* selecionadas e produzem a translação das *features* restantes. Ao mesmo tempo, devem ser realizadas algumas análises frente às possíveis combinações existentes, que devem ser verificadas pelo sistema especialista antes da regeneração dos desenhos. Assim, cada *feature* a ser excluída dispara uma série de regras através do sistema especialista, que realiza a verificação das relações dessa *feature* com as outras da peça. No caso de exclusão de uma *feature* básica, todas as outras *features* modificadoras a ela ligadas também são excluídas (Figura 3.59).

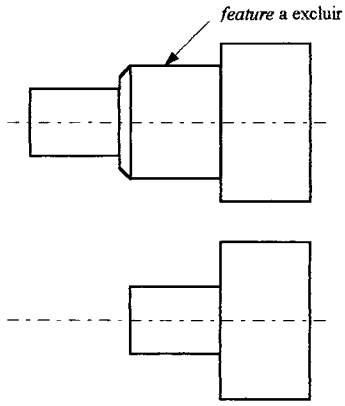


Figura 3.59 - Excluir à direita uma *feature* básica com uma *feature* modificadora associada.

(b) - EXCLUIR UMA *FEATURE* BÁSICA FURO

A exclusão de uma *feature* básica furo é um pouco mais simples que a exclusão de uma *feature* eixo. Como uma *feature* furo está só ou associada a outra *feature* furo, excluir um furo implica excluir um único furo se for o caso, ou na análise da combinação existente entre os diversos furos que possam existir numa peça.

Dessa forma, na Figura 3.60, está representado como se faz a exclusão de uma *feature* furo em função da sua disposição. As operações são válidas independentemente do sentido do furo, pois, conforme as definições (ver item 3.5.3), um furo pode ter sentido à direita ou à esquerda. Assim, um único furo cego pode ser simplesmente excluído como está representado na Figura 3.60.

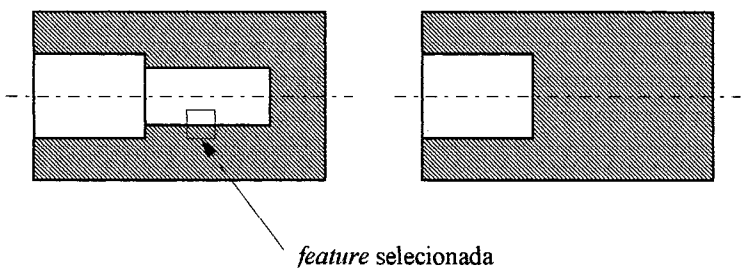


Figura 3.60 - Exclusão de um furo cego final.

Se o furo a ser excluído possui um outro furo na sua seqüência (como um furo escalonado), ao ser deletado o primeiro furo, o seguinte deve ser deslocado de modo que a face inicial do furo que restou seja coincidente com a face inicial do excluído, ou seja, o furo que

permaneceu deve ser deslocado da profundidade do que foi deletado e no sentido contrário, o que pode ser observado na Figura 3.61.

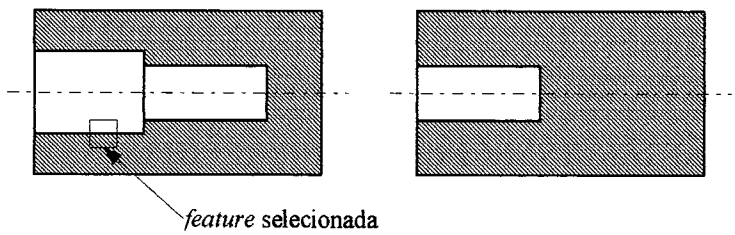


Figura 3.61 - Excluir uma *feature* furo.

Se uma peça tem um furo passante, a operação de exclui-lo consiste simplesmente em efetuar sua eliminação da estrutura de dados e da base de dados do CAD (Figura 3.62).

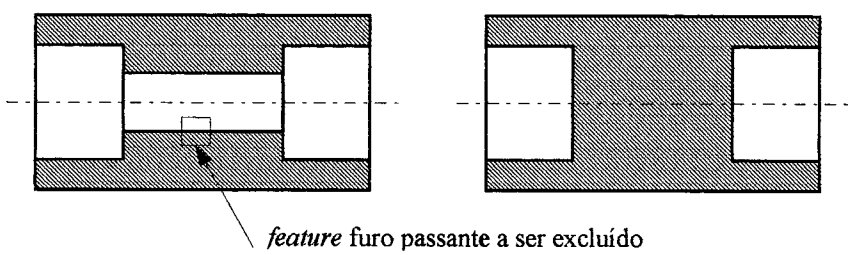


Figura 3.62 - Exclusão de um furo passante associado a furos cegos.

Se, porém, há um furo passante unindo dois furos cegos ou se há um furo passante e um furo cego, quando excluído o furo cego, o passante deve ser alterado no comprimento para que continue sendo um furo passante, ou seja, deve ocupar o lugar do furo cego excluído (Figura 3.63). Essas modificações são realizadas de modo automático.

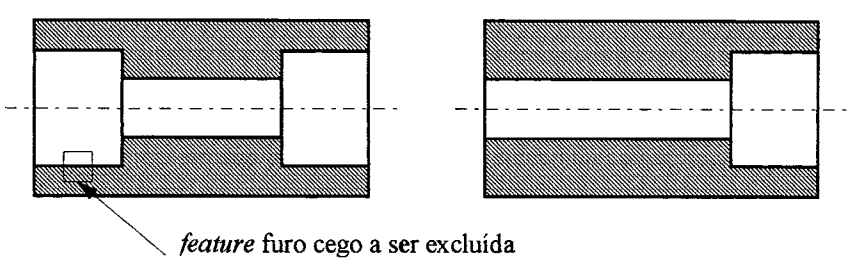


Figura 3.63 - Exclusão de um furo cego associado a um furo passante.

(c) - EXCLUIR *FEATURES* BÁSICAS INTER-RELACIONADAS

Duas *features* básicas são consideradas inter-relacionadas quando da execução de uma operação sobre uma *feature* básica, a qual interfere com uma outra *feature* básica existente. Um exemplo característico pode ser observado ao excluir-se uma *feature* básica eixo onde há uma *feature* furo interna ao eixo, visto que, quando for eliminado o eixo, esse deve produzir um resultado sobre a *feature* furo em razão da eliminação do eixo.

O princípio a ser utilizado para excluir essas *features* inter-relacionadas é o da eliminação total da secção envolvida, ou seja, ao ser excluído um eixo e havendo um furo envolvido, o que estiver coincidente com o eixo será eliminado, sendo a reconstituição das *features* realizadas automaticamente pelo sistema. Na Figura 3.64, são apresentadas algumas possibilidades e resultados em função dos conceitos utilizados. Na figura, a peça é composta por várias *features* eixo e uma *feature* furo passante, sendo apresentados três casos de exclusão dos eixos que compõem a peça e os seus respectivos resultados. Pode-se observar nele que qualquer que seja a *feature* eixo excluída, a *feature* furo passante continua sendo válida, ou seja, um furo passante continua sendo um furo passante.

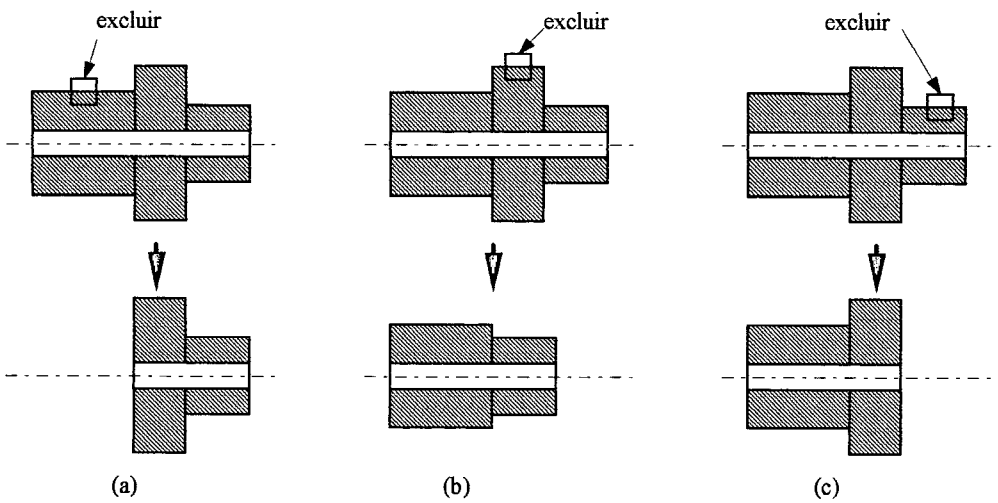


Figura 3.64 - *Features* eixo excluídas e *features* furo passantes inter-relacionadas.

Fato semelhante ocorre quando há um furo cego relacionado com uma *feature* eixo a ser excluída. Na Figura 3.65, é expressa a forma do comportamento das *features* básicas inter-relacionadas quando da exclusão de uma *feature* eixo. Pode-se observar que, no caso (a), ao ser

excluída a *feature* eixo, parte da *feature* furo é eliminada, permanecendo a parte restante na peça. A *feature* furo também é alterada para os casos (b) e (c).

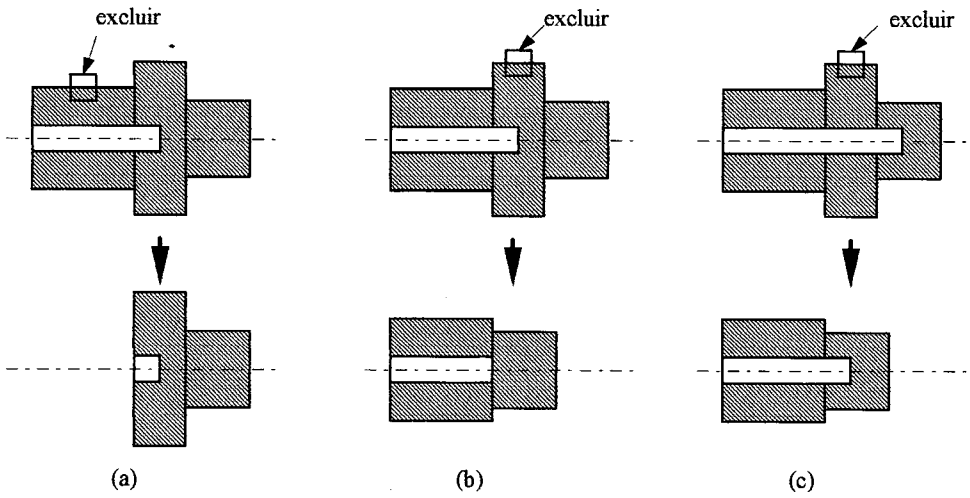


Figura 3.65 - *Features* eixo excluídas e *features* furo cego inter-relacionados.

Ao excluir algumas *features* eixo e dependendo da suas inter-relações, profundas alterações nas *features* podem ocorrer. Na Figura 3.66(a), uma *feature* eixo é excluída, passando a *feature* furo cego inter-relacionada a ser definida como uma *feature* furo passante (transformação de *feature*).

No caso (b), as *features* não sofrem nenhuma transformação, apenas passam a ter posições diferentes para suas faces; no caso (c), a *feature* eixo a ser excluída contém completamente uma *feature* furo e, após a exclusão da *feature* eixo, a *feature* furo também é eliminada.

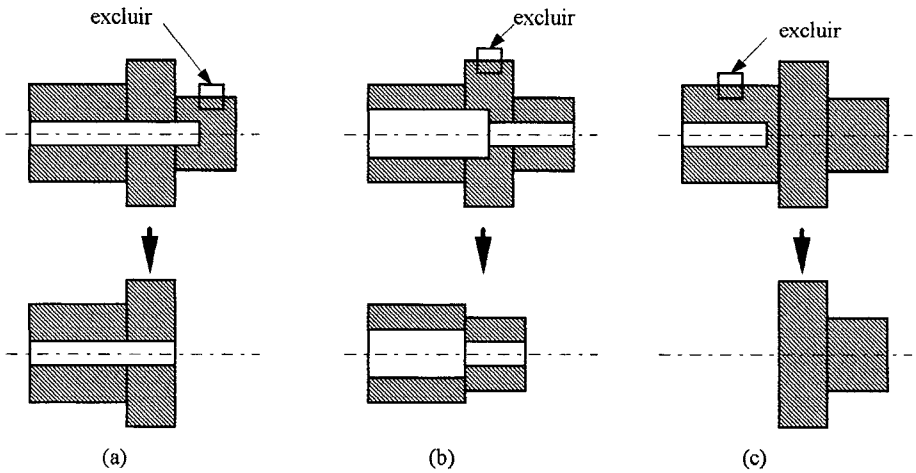


Figura 3.66 - *Features* eixo excluídas e *features* furos cegos inter-relacionadas.

3.7.3 - FUNÇÃO ALTERAR

Esta função efetua a alteração dos atributos de uma determinada *feature*, permitindo a mudança tanto de atributos geométricos como de atributos tecnológicos. A alteração dos atributos é acompanhada das alterações da representação gráfica, ou seja, o desenho é regenerado com os novos atributos.

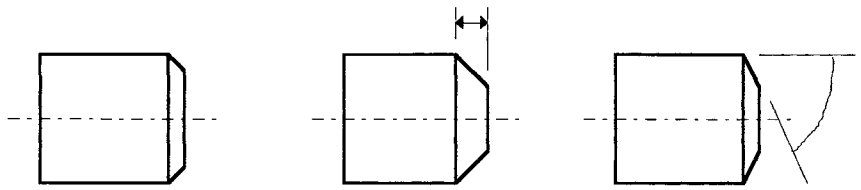
Para efetuar-se a alteração de uma *feature*, deve-se identificar a *feature* desejada; após a identificação, o sistema apresenta a opção de quais os parâmetros que se deseja modificar, parâmetros esses característicos de cada *feature*. Quando da alteração de uma *feature*, análises devem ser realizadas, através do sistema especialista, com relação às *features* vizinhas que podem estar relacionadas. A alteração pode ser feita numa *feature* modificadora ou numa *feature* básica.

3.7.3.1 - FUNÇÃO ALTERAR *FEATURE* MODIFICADORA

A alteração de uma *feature* modificadora é caracterizada pela alteração da própria *feature* e, em alguns casos, da *feature* básica à qual está aplicada. Assim, somente a representação gráfica da *feature* básica que possui a *feature* modificadora é alterada.

No caso de efetuar-se alteração de uma *feature* chanfro, dois parâmetros podem ser alterados, que são o comprimento do chanfro (Figura 3.67 (b)) e o ângulo do chanfro (Figura 3.68 (c)). Identificada a *feature* a ser alterada, o sistema solicita a escolha de qual dos parâmetros deseja-se alterar; feita a escolha, procede-se às operações necessárias para efetuar as modificações gráficas e à atualização dos dados da *feature* na estrutura. Na Figura 3.68, são mostradas as alterações possíveis de um chanfro aplicado a um eixo, as quais são válidas quando aplicadas a uma *feature* furo.

A alteração produzida no comprimento da *feature* chanfro reflete sobre a representação gráfica da *feature* eixo que deve ser modificada. Quando da alteração do ângulo da *feature* chanfro, somente a representação gráfica da *feature* chanfro é modificada (Figura 3.67).



(a) peça (b) altera comprimento (c) altera ângulo

Figura 3.67 - Alteração dos parâmetros de uma *feature* chanfro.

3.7.3.2 - FUNÇÃO ALTERAR *FEATURE* BÁSICA

A alteração de uma *feature* básica é caracterizada pela alteração da própria *feature* e das *features* a ela associadas, que podem ser outras *features* básicas ou outras *features* modificadoras. Nesse caso, não somente a representação gráfica da *feature* básica é alterada, mas também os seus atributos. Quanto às *features* modificadoras, elas podem ou não ter seus parâmetros tecnológicos alterados, o que depende dos parâmetros alterados da *feature* básica.

Entre as *features* básicas que sofrem a operação de alteração, têm-se a *feature* eixo e a *feature* furo, cada uma com as suas particularidades e inter-relações.

3.7.3.2.1 - ALTERAR *FEATURE* BÁSICA EIXO

A alteração de uma *feature* básica eixo consiste, inicialmente, em selecionar a peça e, em seguida, a *feature* desejada. Automaticamente, o sistema FeatCAD-2D identifica o tipo da *feature* que se deseja alterar e apresenta os parâmetros que podem ser alterados, os quais, para uma *feature* eixo, são o diâmetro e o comprimento; um deles deve ser escolhido.

(a) - ALTERAR O COMPRIMENTO DE UMA *FEATURE* EIXO

A alteração do comprimento de um eixo, considerando que a peça não possui nenhum outro tipo de *feature* básica ou modificadora, pode ser feita de duas formas: a) à direita e b) à esquerda. Uma alteração à direita consiste em definir um plano de referência que está à direita da *feature* a qual se deseja alterar, de forma que as *features* que se localizam à esquerda da

feature a ser alterada sofrem uma translação do valor a ser alterado, do comprimento (ΔL) da *feature* que sofre a alteração(Figura 3.68).

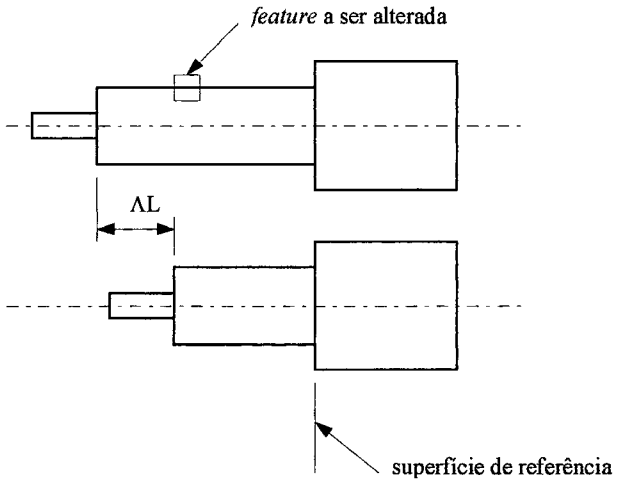


Figura 3.68 - Alterar *feature* eixo à direita.

Uma alteração à esquerda consiste em definir um plano de referência que está à esquerda da *feature* que se deseja alterar, sendo que as *features* que se localizam à direita da *feature* a ser alterada sofrem uma translação do valor a ser alterado, do comprimento (ΔL) da *feature* que sofre a alteração (Figura 3.69).

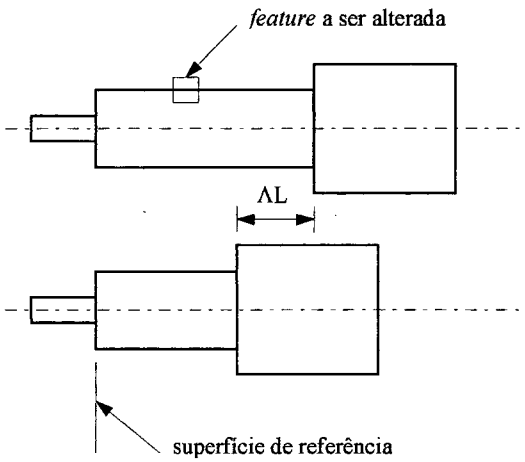
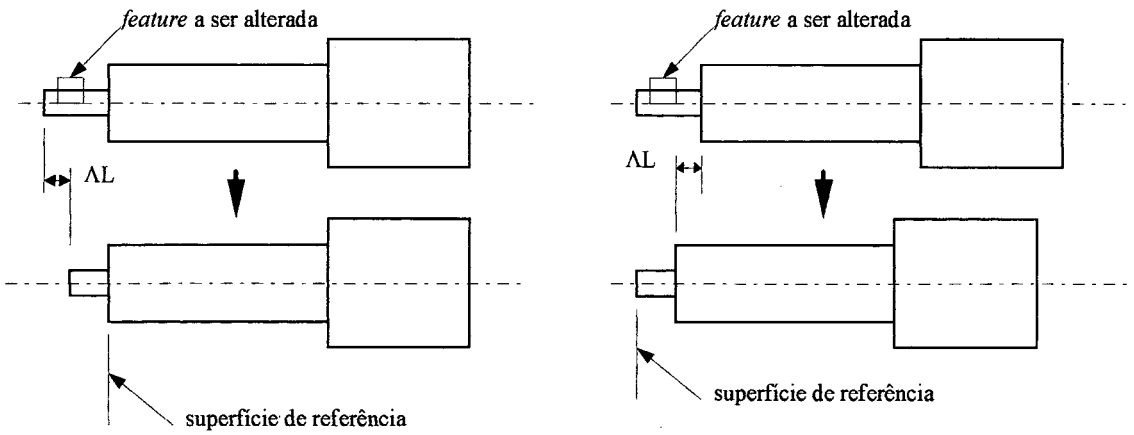


Figura 3.69 - Alterar *feature* eixo à esquerda.

No caso de a *feature* a ser alterada se localizar numa das extremidades da peça e for alterada no comprimento, nenhuma outra *feature* sofrerá translação se o plano de referência corresponder ao plano entre duas *features* eixo (Figura 3.70(a)). Se o plano de referência for a

face da *feature* na extremidade da peça (Figura 3.70 (b)), as *features* restantes sofrerão o deslocamento de acordo com a direção da alteração (ΔL).



(a) Alteração à direita (b) Alteração à esquerda

Figura 3.70 - Alteração de *feature* eixo na extremidade de uma peça.

Observando a Figura 3.70, pode-se supor que o usuário deseje efetuar uma alteração no comprimento da *feature* eixo de modo que a variação ΔL seja igual ao comprimento da própria *feature*, o que resultaria no desaparecimento da mesma. Esse caso não é executado pelo sistema, pois consiste na operação de excluir uma *feature* e não de alterar.

(b) - ALTERAR O DIÂMETRO DE UMA *FEATURE* EIXO

A alteração do diâmetro de uma *feature* eixo é um processo simples quando da existência de apenas *features* eixo numa peça, ou seja, a alteração do diâmetro não influencia nenhuma outra *feature* de modo significativo, como pode ser observado na Figura 3.71.

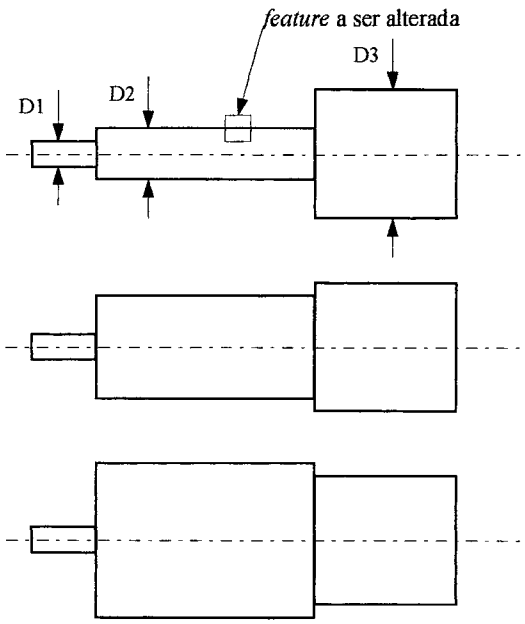


Figura 3.71 - Alteração do diâmetro de uma *feature* eixo.

Se a alteração do diâmetro da *feature* eixo for tal que resulte em duas *features* eixo vizinhas de mesmo diâmetro, uma união dessas *features* deve ser realizada para que resulte numa nova *feature* eixo que terá o mesmo diâmetro, mas com comprimento igual à soma dos comprimentos das duas *features* envolvidas (Figura 3.72). Assim, na Figura 3.72, os diâmetros D2 e D3 resultaram iguais após a alteração da *feature* eixo de diâmetro D2, ocorrendo, assim, a união das duas *features* envolvidas.

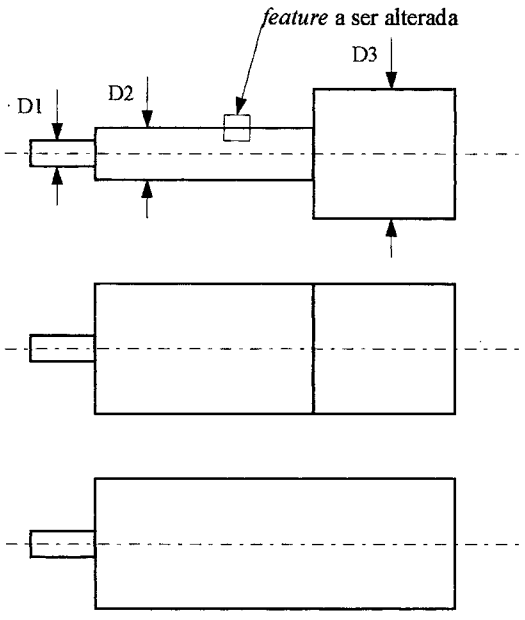


Figura 3.72 - Alteração do diâmetro de uma *feature* eixo resulta na união de duas *features* eixo.

Um outro caso pode vir a ocorrer, como se pode observar na Figura 3.72, quando se pode supor o diâmetro D2=0. Nesse caso, a *feature* eixo deixa de existir, e o sistema não admite tal operação, pois se o diâmetro é zero, a *feature* é eliminada, o que deve ser feito através da operação excluir.

3.7.3.2.2 - ALTERAR *FEATURE* BÁSICA FURO

Assim como para a *feature* eixo, a alteração de uma *feature* básica furo pode ser feita na profundidade e no diâmetro. Além do mais, as alterações produzidas numa *feature* básica furo podem refletir sobre *features* modificadoras ou sobre outras *features* básicas furo existentes.

A *feature* básica eixo atua como elemento de restrição nas alterações que podem ser executadas numa *feature* furo. Assim, tem-se uma restrição quanto ao diâmetro da *feature* eixo que limita o diâmetro máximo da *feature* furo. A alteração da profundidade de uma *feature* furo tem como restrição a profundidade de uma outra *feature* furo (no caso dessa ser uma *feature* furo tipo cego), o comprimento da peça e, algumas vezes, o diâmetro da *feature* eixo que contém a *feature* furo.

Para efetuar a alteração da profundidade de uma *feature* furo, há uma particularidade a ser observada: não é possível alterar a profundidade de um furo de modo que a sua face inicial seja mudada de posição (Figura 3.73).

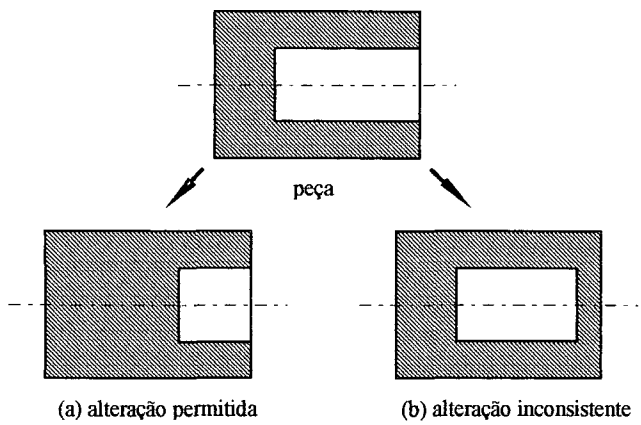


Figura 3.73 - Alteração de uma *feature* furo.

Ao definir a *feature* furo a ser alterada, automaticamente é tomada como superfície de referência a sua face inicial; logo, a alteração da profundidade altera a posição da face final da

feature. Se a *feature* a ser alterada possui na sua seqüência uma outra *feature* furo (p. ex., um furo escalonado cego), a *feature* seguinte que não sofre alteração é deslocada de modo que sua face inicial permaneça em contato com a face final da *feature* furo alterada (Figura 3.74).

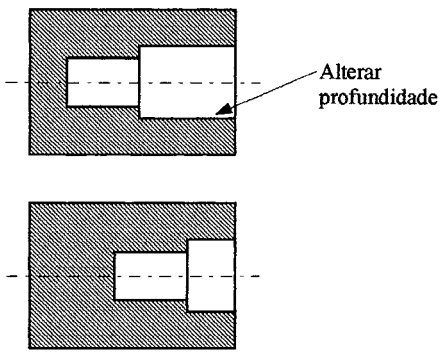


Figura 3.74 - Alteração de uma *feature* furo escalonado cego.

A alteração da profundidade de um furo tem como limite máximo a profundidade de material disponível na peça ou o comprimento total da peça. Dessa forma, dois fatos podem ocorrer em função da alteração da profundidade do furo: a) se a alteração da profundidade do furo permanece dentro do limite disponível, esse furo continua sendo uma *feature* furo cego e b) se a alteração do comprimento desse furo cego atinge o limite disponível, isso o torna uma *feature* furo passante. Deve-se mencionar que uma *feature* furo passante, nesses casos, é considerada sempre a de diâmetro menor (Figura 3.75).

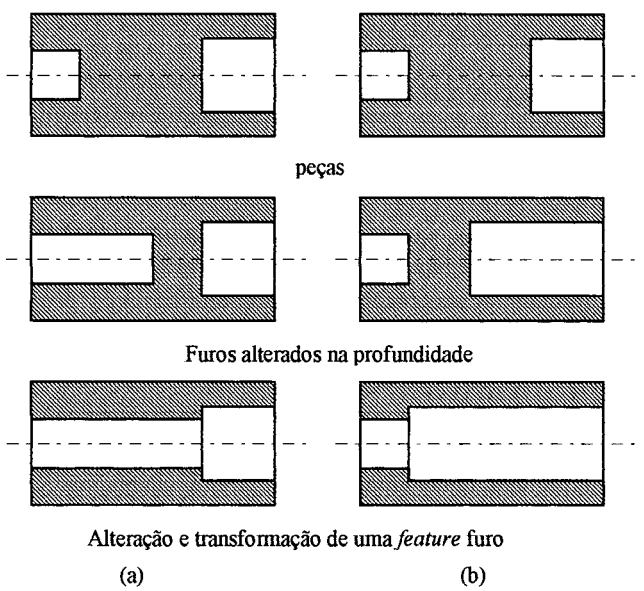


Figura 3.75 - Alteração e transformação das *features*.

No caso da presença de uma *feature* furo cego ligada a outra *feature* furo passante, deve-se alterar a primeira, uma vez que, na concepção, não se pode alterar a profundidade de um furo passante, que possui como limites o espaço disponível no qual foi inserido; portanto, a mudança desse espaço provoca a sua variação.

Logo, os seus limites podem ser o limite da peça, pois, no caso, somente com a variação do comprimento da peça pode ser variada a profundidade de uma *feature* furo passante. Por outro lado, se a *feature* furo passante estiver ligada a uma *feature* furo cego, o limite é dado pela peça e pelo furo cego. Ao variar-se o furo cego, conseqüentemente, será feita a variação do furo passante (Figura 3.76).

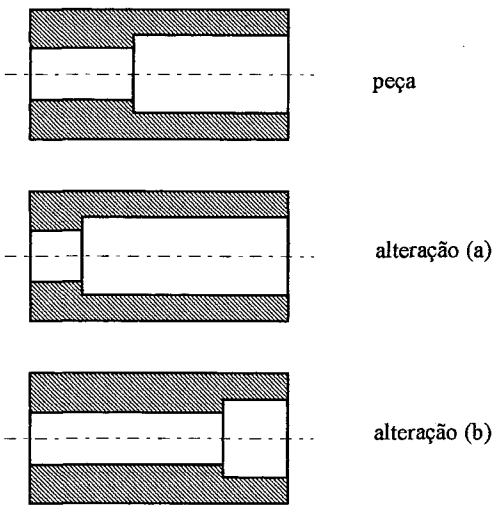


Figura 3.76 - Alteração de um furo cego e um furo passante.

A alteração da profundidade de um furo, algumas vezes, exige uma verificação do diâmetro das *features* furo e eixo envolvidas na alteração. Isso ocorre se a peça na qual o furo está inserido é constituída de várias *features* eixo. Desse modo, a verificação não se limita apenas à profundidade, mas também à relação aos diâmetros envolvidos (Figura 3.77).

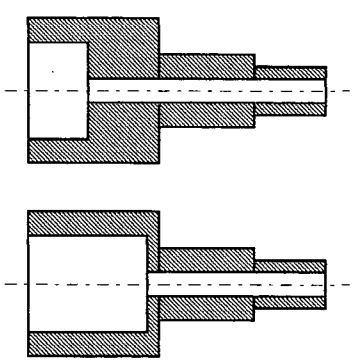


Figura 3.77 - Alteração da profundidade se é limitada pelo diâmetro.

(a) - ALTERAR DIÂMETRO DOS FUROS

Alterar o diâmetro de uma *feature* furo, como foi dito anteriormente, possui como restrição as dimensões das *features* eixo nas quais ela está inserida. Assim, pode ocorrer que a *feature* furo selecionada para alteração de diâmetro esteja totalmente contida numa única *feature* eixo, onde a análise se retringe à verificação do diâmetro da *feature* eixo.

Se a *feature* furo selecionada está contida em mais de uma *feature* eixo, a verificação deve ser realizada nas *features* eixo envolvidas, prevalecendo como limite a de menor diâmetro (Figura 3.78)

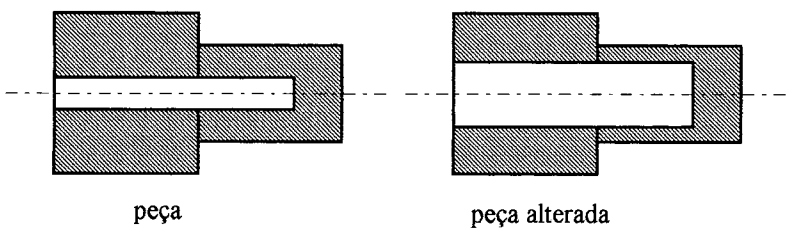


Figura 3.78 - Alteração do diâmetro de uma *feature* furo.

Transformações de *features* podem ocorrer em função da alteração de diâmetro. A transformação de uma *feature* ocorre quando as alterações ocorridas na peça provocam a mudança na interpretação daquela *feature*. Isso pode ser observado quando duas *features* furo estão ligadas entre si, como no caso de uma *feature* furo cego e uma *feature* furo passante.

Ao alterar-se o diâmetro da *feature* furo passante para um diâmetro maior do que o do furo cego, a *feature* furo cego passa a ser uma *feature* furo passante e vice-versa (Figura 3.79).

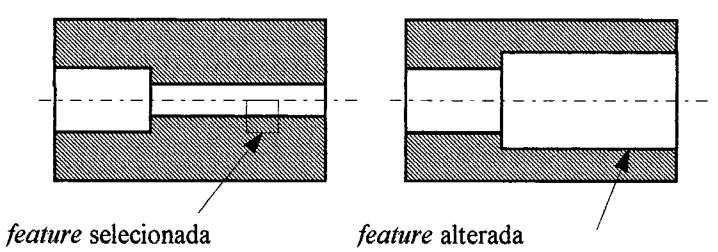


Figura 3.79 - Alteração de diâmetro com transformação de *features*.

3.7.3.2.3 - ALTERAR *FEATURES* BÁSICA E MODIFICADORA INTER-RELACIONADAS

Nos itens anteriores, foi descrita a forma de alteração das *features* básicas e das modificadoras independentemente. Agora, é necessária a verificação das inter-relações existentes entre as *features* de natureza diferente, ou seja, as relações entre as básicas e as modificadoras.

No presente trabalho, serão analisadas as inter-relações entre as *features* básicas eixo e furo com a *feature* modificadora chanfro, do mesmo modo que, na função excluir, há na base de conhecimento algumas regras para a tomada de decisão com relação a essa função.

A inter-relação mais simples consiste na alteração do diâmetro de uma *feature* eixo que possui numa extremidade livre aplicada uma *feature* chanfro (Figura 3.80). A alteração apenas provoca o redimensionamento da *feature* chanfro sobre a *feature* eixo

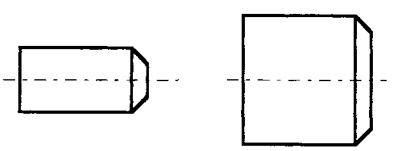


Figura 3.80 - Alteração de uma *feature* eixo com uma *feature* chanfro.

Já a existência de uma *feature* chanfro aplicada a uma *feature* eixo com outra *feature* eixo vizinha pode provocar interferências quando da alteração do eixo ou do chanfro. Na Figura 3.81, tem-se o caso em que o diâmetro do eixo é alterado de modo que o chanfro que está aplicado a esse eixo produz uma inconsistência com o eixo vizinho. Logo, tal operação deve ser vetada pelo sistema.

O sistema especialista deve retornar uma mensagem informando ao usuário que essa operação não pode ser executada devido à inconsistência. Para que a operação tenha sucesso, o chanfro deve ser excluído antes de ser efetuada a alteração. A decisão de excluir o chanfro pode ser deixada a cargo do sistema, ou seja, quando for detectado esse fato, automaticamente, o chanfro será eliminado. Isso depende do grau de automação desejado para o sistema e do poder de decisão que o usuário deverá ter a seu dispor. No presente trabalho, o poder de decisão fica a cargo do usuário, o que significa que o chanfro deve ser eliminado manualmente.

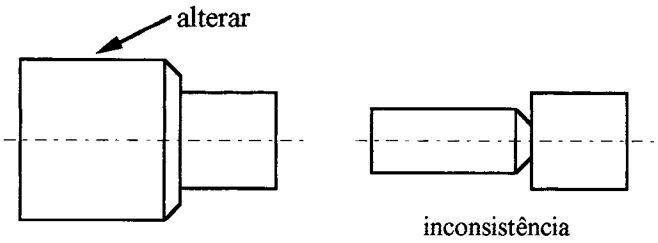
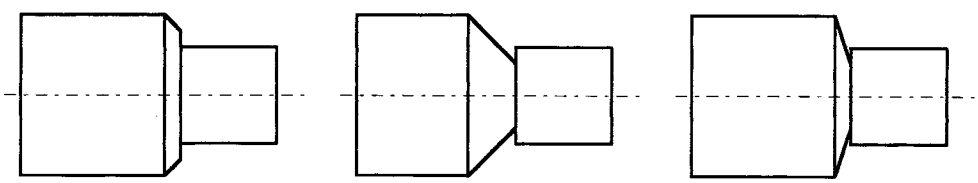


Figura 3.81 - Alterar o diâmetro do eixo produzindo uma inconsistência.

Pode surgir um outro tipo de interferência resultante da alteração do comprimento do chanfro ou da alteração do ângulo, o que é ilustrado na Figura 3.82, onde, em (a), tem-se a peça a qual se deseja alterar os parâmetros do chanfro; em (b), o comprimento do chanfro foi alterado e, em (c), somente foi alterado o ângulo do chanfro. Esses últimos correspondem a dois casos de inconsistência de alteração. Conseqüentemente, o sistema rejeita a operação informando ao usuário o que ocorre.

Um chanfro é limitado a um determinado comprimento, acima do qual ele deixa de se caracterizar como um chanfro. Na Figura 3.82(b), observa-se que a alteração provocada no chanfro pode ocasionar uma transformação de *feature*. Se o comprimento do chanfro alterado for maior que o comprimento limite especificado na base de conhecimento, o sistema rejeita a alteração. Nesse caso, ao invés de um chanfro, seria ideal a existência de uma *feature* eixo cônico no local.



(a) Consistente (b) Inconsistênte (c) Inconsistênte

Figura 3.82 - Alterar chanfro.

A alteração do comprimento de uma *feature* eixo não provoca nenhuma alteração dos parâmetros dimensionais da *feature* modificadora, apenas o seu deslocamento da posição.

Com relação à existência de um furo com um chanfro, a modificação do diâmetro do furo pode provocar interferência com o eixo que contém o furo. Na Figura 3.83, tem-se um eixo com um furo e um chanfro, sendo que a alteração do diâmetro do furo provoca uma inconsistência para o chanfro com relação ao eixo.

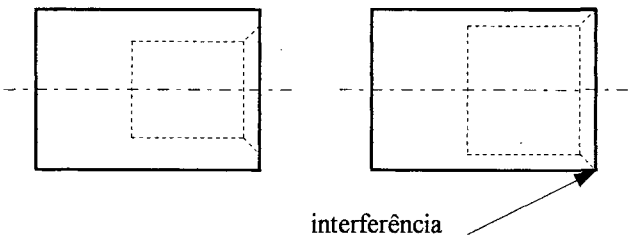


Figura 3.83 - Alteração do diâmetro de um furo com um chanfro interno.

De modo semelhante a um chanfro sobre um eixo para um furo, a alteração dos parâmetros do chanfro (i.e., comprimento e ângulo) também pode provocar inconsistência com relação ao eixo. Na Figura 3.84, estão representados esses casos; em (a), tem-se a peça a ser alterada; em (b), foi feita a alteração do comprimento do chanfro, o que provoca uma interferência com o eixo; do mesmo modo, em (c), onde foi alterado o ângulo do chanfro, que também provoca uma interferência com o eixo. São todos casos que o sistema não aceita, cancelando a operação de alteração.

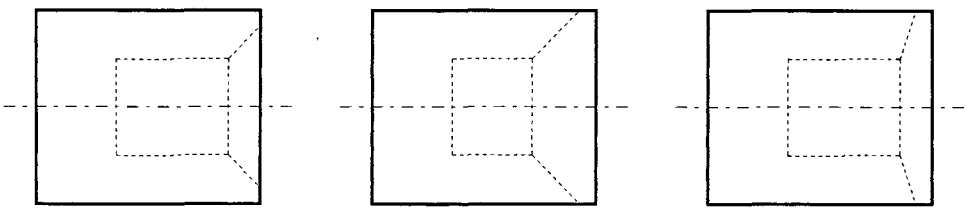


Figura 3.84 - Alteração de um chanfro interno.

No caso de haver um chanfro aplicado num eixo e num furo, estando os mesmos numa mesma face do eixo, a interferência pode ocorrer entre duas *features* modificadoras, que, no caso, são os dois chanfros. Isso pode ser observado através da Figura 3.85(a), onde se tem a peça a ser alterada. Em (b), é realizada a alteração do comprimento do chanfro externo o qual interfere com o chanfro interno; em (c), a alteração é efetuada no ângulo do chanfro interno, o que produz uma interferência com o chanfro externo. Pode-se observar que resultaram disso duas inconsistências que o sistema não aprova.

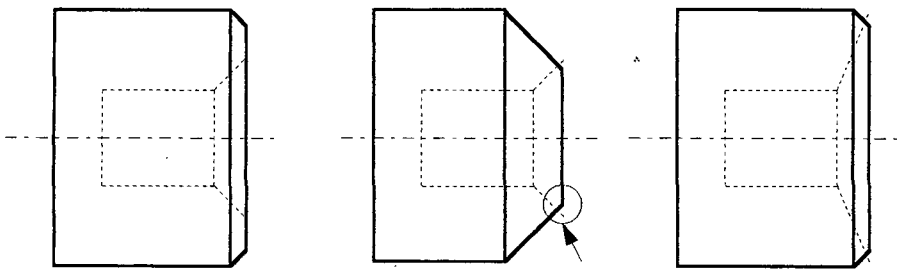


Figura 3.85 - Interferência na alteração de um chanfro interno ou externo.

3.7.4 - FUNÇÃO MOVER

No sistema FeatCAD-2D, a operação mover é definida somente em termos de mover uma peça, ou seja, não é possível mover uma *feature*, a qual somente é permitido excluir ou alterar. Considera-se que mover uma *feature* é equivalente a excluir a *feature* daquela posição, criando-a noutra posição.

3.7.4.1 - MOVER PEÇA

Para mover uma peça, deve-se, inicialmente, selecioná-la; em seguida, escolher um ponto de referência que deve estar sobre a peça e, após, um ponto de referência final, que é o ponto para onde se deseja que a peça seja movida. O ponto de referência da peça coincidirá com o ponto de referência final (Figura 3.86). Com a operação mover peça, todas as *features* que compõem a peça sofrem deslocamento para a nova posição.

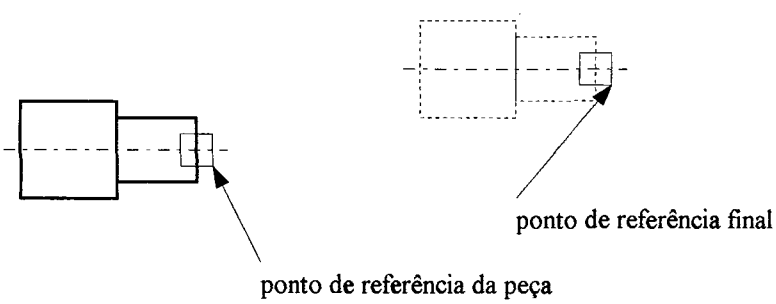


Figura 3.86 - Operação mover peça.

3.8 - ANÁLISE DO PRODUTO REPRESENTADO

As análises de manufatura, montagem, inspeção e outras atividades que dependem das informações advindas do projeto consideram que essas informações sejam consistentes e estejam disponíveis. Assim, a decisão de ter-se uma superfície usinada e a utilização de uma determinada tolerância para a uma dimensão, bem como a disposição das dimensões, são funções de uma análise rigorosa do projeto representado e não do gosto do indivíduo que as projetou.

Logo, é de fundamental importância a especificação de atributos corretos a cada peça, o que é resultado não da análise da peça isoladamente, mas da consideração também do conjunto em que ela vai atuar. Para a decisão quanto aos aspectos de tolerâncias, acabamento superficial e identificação das respectivas superfícies, bem como a cotagem funcional, que é um requisito importante, é fundamental que se tenha a visão global do conjunto.

Para isso, é proposta para o sistema FeatCAD-2D, como descrito anteriormente, uma estrutura de dados que comporta a representação de um produto com sua respectiva hierarquia. De posse dessas informações, é possível o computador efetuar análises que são feitas manualmente pelo indivíduo, automatizando-as a partir dos critérios utilizados por aquele. Como por exemplo, o indivíduo identifica que uma cota deve ser controlada em função de que ela possui uma importância com relação à montagem, do mesmo modo o computador deve analisar.

A primeira análise a ser feita é com relação à montagem que o conjunto representa, ou seja, identificar quais as peças que estão montadas entre si, quais as superfícies cilíndricas (eixos e furos) que estão em contato, bem como os contatos axiais que se realizam (contatos pelas faces das *features*). Conhecendo-se quais as superfícies cilíndricas que possuem contato, é possível identificar as dimensões que representam os diâmetros que devem receber tolerâncias.

Determinadas as dimensões do eixo e do furo a serem controladas, compete ao usuário efetuar a escolha das tolerâncias que, a partir da função que o par de acoplamento possui no conjunto, ele escolherá interativamente, com o auxílio do sistema, o par de ajuste que melhor se adapte à função. O acabamento superficial que deve ser associado a essas superfícies é determinado em função da tolerância escolhida.

Com os contatos axiais determinados, também é possível definir as tolerâncias das cotas lineares, assim como o respectivo acabamento superficial das faces em contato. As faces de contato passam a funcionar como referências para a cotagem linear das peças, o que traduz as

condições da montagem para a peça, isso com a chamada cotagem funcional (ver item 2.13.2). Essa cotagem pode ser automático ou manual.

Como conseqüência, se uma superfície possui um contato, essa deve ser usinada e, como resultado, um bom acabamento superficial deve ser associado a ela; portanto, todas essas superfícies em contato podem ser encontradas automaticamente para a especificação do acabamento.

Com o conhecimento das tolerâncias e ajustes, é possível verificar qual o ferramental necessário para efetuar determinadas montagens, como, por exemplo: para um ajuste prensado, deverá ser escolhida uma prensa para a tarefa.

Vários tipos de análise do produto podem ser efetuadas, as quais, aqui descritas, procuram extrair novas informações a partir de outras já representadas na estrutura de dados.

3.8.1 - IDENTIFICAÇÃO DE UMA MONTAGEM

Quando da elaboração do projeto detalhado, a primeira fase corresponde à execução da representação gráfica com a utilização de cotas nominais, ou seja, não é considerado o problema de desvios de fabricação, bem como problemas de funcionalidade referentes a folgas que devem existir entre os componentes na montagem.

Quando duas peças estão montadas, as linhas que representam as suas superfícies de contato estão sobrepostas, daí possuírem as mesmas coordenadas de posicionamento num sistema de coordenadas absoluto. Assim, para que o sistema FeatCAD-2D possa interpretar como as peças estão montadas, a partir da representação gráfica efetuada pelo usuário, sendo fundamental a execução gráfica dentro de determinadas regras básicas a fim de que possam ser transmitidas ao sistema essas características.

3.8.1.1 - IDENTIFICAÇÃO DA MONTAGEM EM PEÇAS CILÍNDRICAS

Se duas peças estão montadas, elas possuem determinadas cotas e coordenadas de posicionamento em comum. Quando se verifica que duas peças estão em contato pelo diâmetro, a condição é que a cota nominal do furo e do eixo sejam iguais. Se estão ocupando a mesma

posição no espaço, isto é, o posicionamento axial das duas peças é coincidente, significa que estão montadas.

Na Figura 3.87, estão representados quatro casos: no primeiro (a), está ocorrendo contato diametral, sendo identificada uma condição de montagem; em (b), (c) e (d), vê-se que a condição de contato diametral não ocorre.

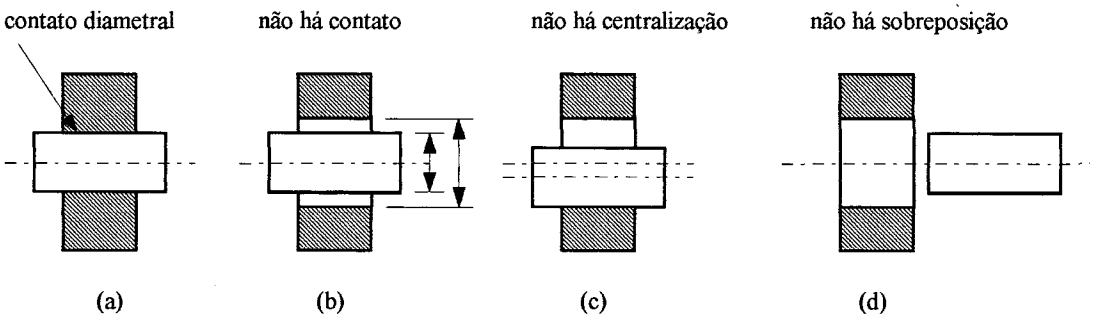


Figura 3.87 - Condições de contato diametral.

Numa montagem de peças rotacionais, além do contato diametral, é usual o contato axial para efetuar o posicionamento adequado da peça do conjunto. Se a posição no espaço das superfícies terminais de duas peças é a mesma, elas possuem contato axial. A Figura 3.88 (a) ilustra um contato axial entre as duas peças; em (b), esse contato não ocorre pela falta de coincidência das superfícies terminais.

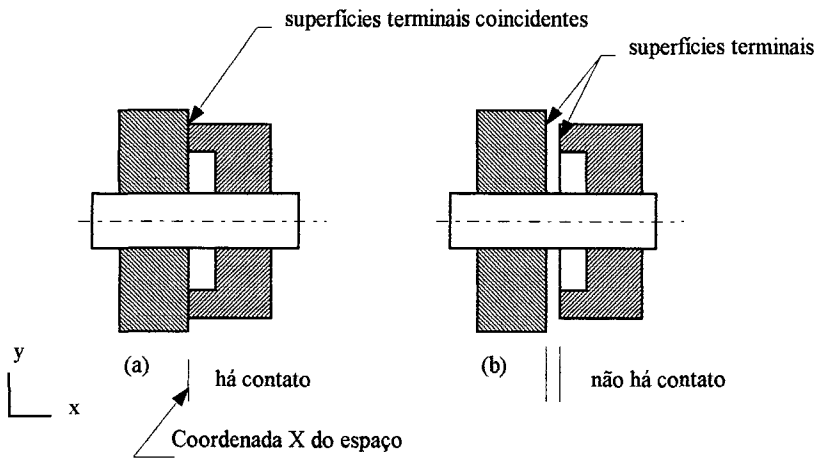


Figura 3.88 - Condições de contato axial.

Logo, pode-se observar que, através da leitura adequada da representação de um produto, é possível criar critérios que possibilitem uma análise da montagem. Realiza-se essa análise

através do uso de algoritmos, que executam um processo de busca e identificação e de regras (sistema especialista), que possibilitam a modelagem das informações para a tomada de decisão em cada situação.

Como se pode observar, peças rotacionais possuem dois tipos básicos de contatos que são os diametrais e os axiais. Esses contatos podem, normalmente, ocorrer entre duas peças, não sendo conveniente a ocorrência de dois contatos de mesmo tipo ao mesmo tempo, o que pode ser observado na Figura 3.89 (a), onde o contato diametral ocorre duas vezes entre as mesmas peças; em (b), tem-se uma montagem correta visto que ocorre um contato de cada tipo entre as peças.

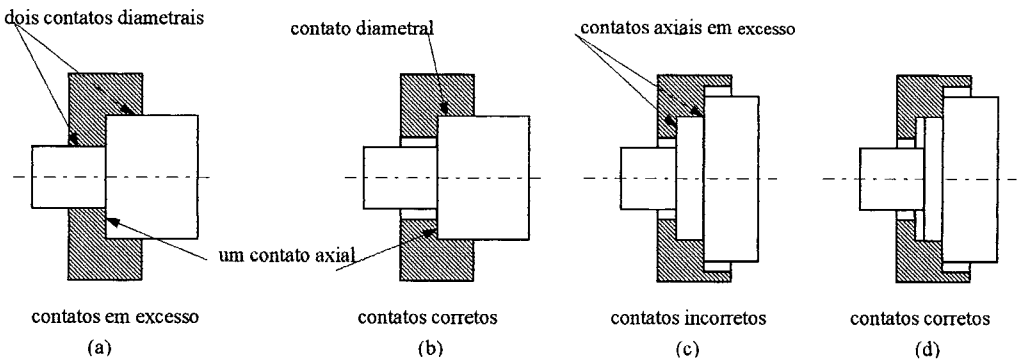


Figura 3.89 - Ocorrência de contatos.

A ocorrência de contatos adequados permite a obtenção de um bom produto, reduzindo a precisão necessária e resultando numa fácil fabricação, fácil montagem, verificação, etc. Desse modo, é importante verificar as condições de contato entre as peças, as quais vão ditar boas normas de projeto para o produto.

O sistema FeatCAD-2D toma por base a análise dos contatos para identificar a disposição das peças na montagem; de posse desse levantamento, efetua a análise e o retorno de informações quanto à consistência do projeto do produto, visando à fabricação, inspeção e montagem. Essas informações são condições que se referem ao conjunto ou subconjunto conforme o caso; nelas devem estar registradas todas as características necessárias para identificar a montagem e dela retirar outras informações para a análise não somente da montagem, mas para outros processos.

Para uma análise adequada, é necessário que sejam definidos os tipos de contatos tanto diametrais como axiais. Assim, os contatos diametrais serão considerados como aptos a uma montagem adequada se ocorrerem dentro de determinadas condições que estão ilustradas na Figura 3.90, onde estão representadas peças montadas que possuem somente contato diametral.

A ocorrência do contato diametral já caracteriza indícios de uma montagem, porém a montagem é considerada boa quando a superfície de apoio é a mínima suficiente. Na Figura 3.90(d), está representada a condição que deve ser encontrada para que a montagem seja aceita como adequada.

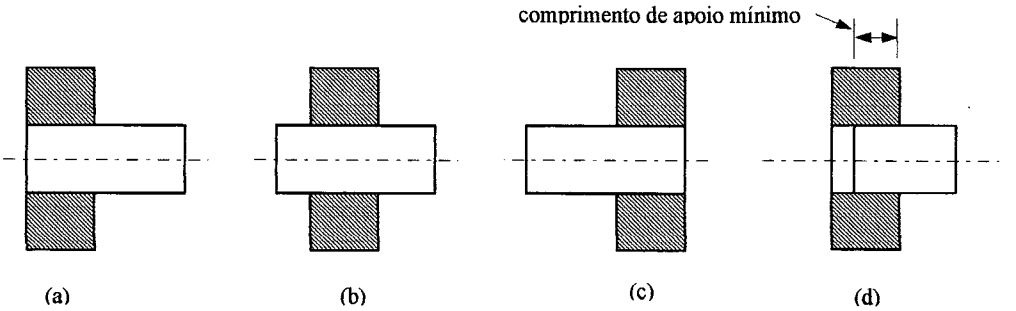


Figura 3.90 - Posições que um eixo pode ocupar com relação a um furo.

Na Figura 3.91, estão representadas as condições de montagem entre furos cegos e eixos, condições essas vistas pela ótica do contato diametral, que é o primeiro julgamento realizado a partir das condições apresentadas, ou seja, uma peça somente é analisada axialmente se for identificado um contato diametral.

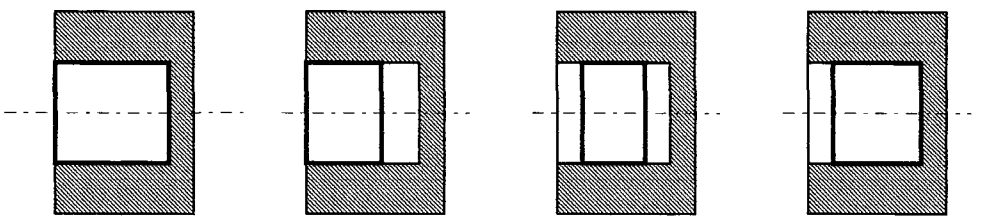


Figura 3.91 - Disposição eixo / furo para contatos diametrais.

A definição dos contatos axiais é feita através das coordenadas que correspondem às faces planas das *features*, pois, através delas, é possível determinar as coordenadas de referência das peças e saber quem está em contato com quem. A seguir, são apresentados exemplos de contatos axiais aceitos como adequados.

Na Figura 3.92, estão representados os contatos axiais externos básicos, ou seja, que ocorrem pelas faces planas de *features* ditas externas, as quais, no caso, são eixos. Em (a), tem-se que, na mesma peça, ocorre o contato diametral e o contato axial (peça0); assim, logicamente, as *features* que possuem o contato devem ser diferentes. Já em (b), o contato axial

entre as peças 1 e 2 é entre peças que não possuem contato diametral entre si, caracterizando-se, então, uma outra forma de montagem.

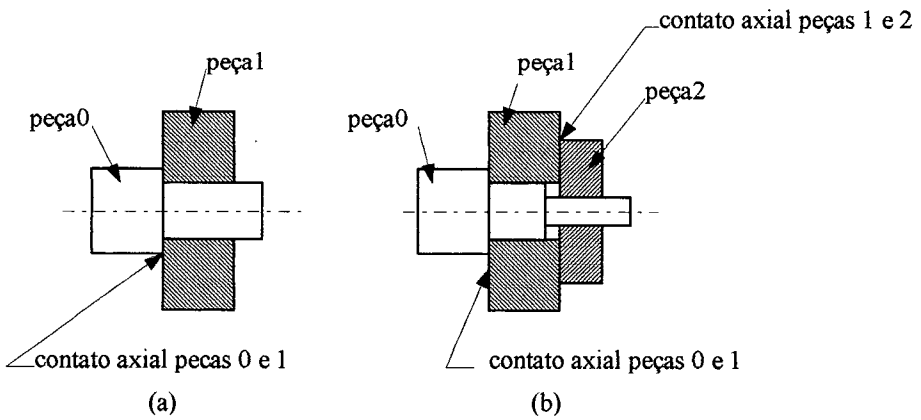


Figura 3.92 - Contatos axiais básicos eixo/furo.

Na Figura 3.93, estão representados dois casos de contatos axiais internos, onde as faces de contato das *features* se dão entre uma face de uma *feature* interna e a face de outra *feature* que deve ser externa.

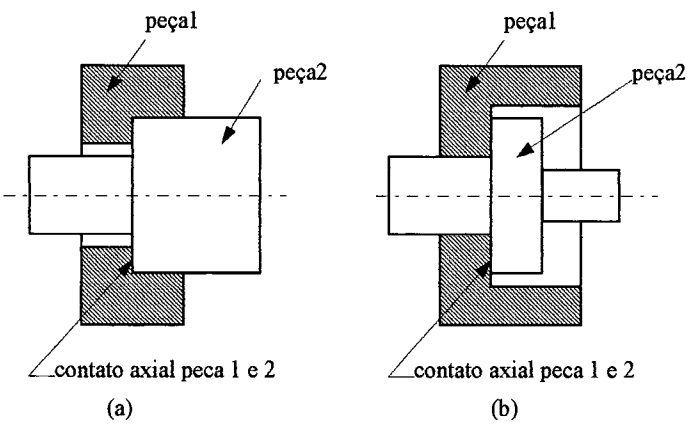


Figura 3.93 - Contato axial interno.

A identificação desses contatos, diametral e axial, deve levar à formação de um conjunto de informações que traduzam a situação de cada peça na respectiva montagem. Assim, é possível identificar qual a peça que possui a *feature* furo, bem como a que possui a *feature* eixo e, ainda, o respectivo diâmetro de contato, isso para traduzir o contato diametral se somente esse ocorrer.

Para o contato axial, é fundamental saber qual o nome das peças que possuem o contato axial comum e também as respectivas *features* envolvidas de cada peça, assim como a coordenada comum das faces em contato.

Outros exemplos de ocorrência de contato axial numa montagem são descritos na Figura 3.94(a), onde está representado o contato axial entre duas peças que possuem contato diametral e o contato axial; naturalmente, as *features* não são as mesmas.

Na Figura 3.94(b), o contato axial se dá entre duas peças que não possuem relação de contato diametral, apenas a ocorrência de contato axial.

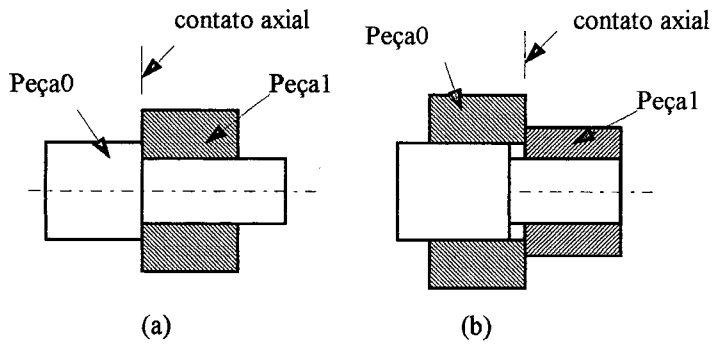


Figura 3.94 - Exemplos de contatos axiais.

Há casos em que o contato axial não ocorre entre duas peças devido às cotas do diâmetro. Na Figura 3.95(a), ocorre o contato diametral entre a Peça0 e a Peça1, mas não há contato axial. Na Figura 3.95(b), estão duas peças que não possuem relação de contato diametral, e o contato axial não ocorre devido ao fato de a Peça0 possuir um diâmetro interno maior que o diâmetro externo da Peça1. Na Figura 3.95(c), têm-se duas peças montadas onde apenas ocorre o contato diametral, não existindo nenhuma outra peça para promover o contato axial.

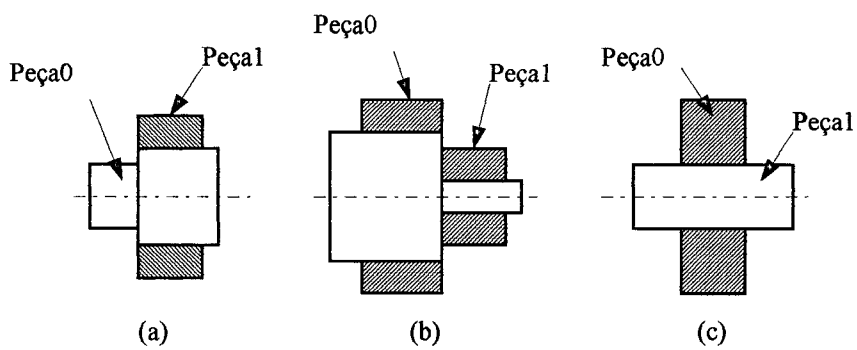


Figura 3.95 - Exemplos da não-ocorrência de contato axial.

3.8.1.2 - IDENTIFICAÇÃO DA MONTAGEM DE PEÇAS CÔNICAS

A caracterização dos contatos realizados através de um par de acoplamento furo/eixo cônico exige novas definições da caracterização das superfícies que produzem o contato diametral. Dois casos em que um acoplamento furo/eixo cônico são permitidos são ilustrados na Figura 3.96, a partir dos quais podem ser realizadas as respectivas identificações.

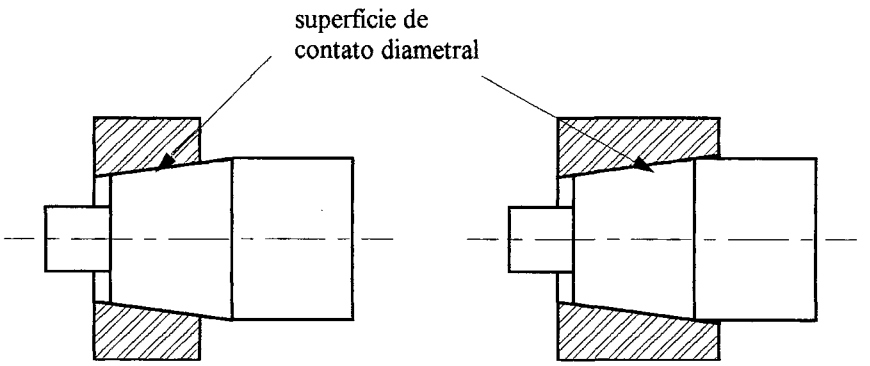


Figura 3.96 - Exemplos de montagem de elementos cônicos.

3.8.2 - ESPECIFICAÇÃO DOS CONTATOS

Os contatos identificados num conjunto devem ser representados numa forma em que se possa identificar quais as peças envolvidas na montagem, como também as *features* envolvidas em cada peça e o tipo de contato que está ocorrendo. Para isso, os tipos de contato foram divididos em: a) contatos diametrais e b) contatos axiais. Como mostrado anteriormente, os contatos diametrais são aqueles que possuem como característica o mesmo diâmetro nominal para o eixo e o furo, e essas duas *features* possuem sobreposição de imagem; já o contato axial presume que as faces de duas *features* de duas peças diferentes estejam em contato.

Desse modo, foi definido um conjunto de atributos que visam especificar quando está ocorrendo uma montagem e quais as peças e *features* que estão nela envolvidas (Figura 3.97).

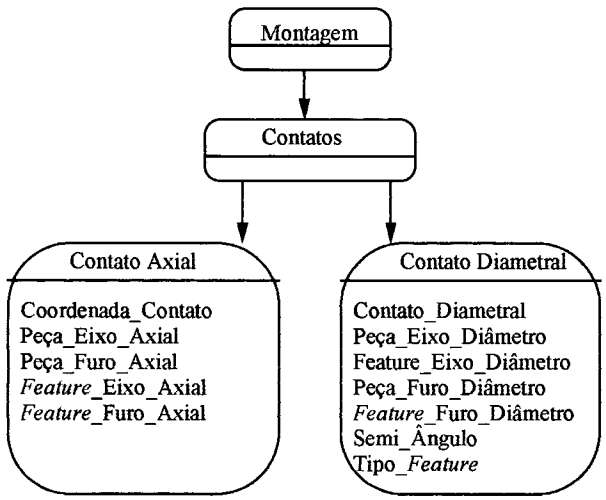


Figura 3.97 - Atributos de contato da classe conexão.

Com relação ao contato axial, pode-se descrever :

- Coordenada_Contato: corresponde à coordenada axial onde as faces de duas peças em contato são coincidentes (Figura 3.98);

- Peça_Eixo_Axial: é a peça que possui o eixo o qual está em contato com a face do furo (Figura 3.98);

- Peça_Furo_Axial: é a peça que possui o furo que está sendo analisado (Figura 3.98);

- *Feature*_Eixo_Axial: corresponde à *feature* eixo, que possui o contato axial com a peça que possui o furo (Figura 3.98);

- *Feature*_Furo_Axial: corresponde à *feature* furo, que possui a face em contato com a *feature* eixo de outra peça (Figura 3.98).

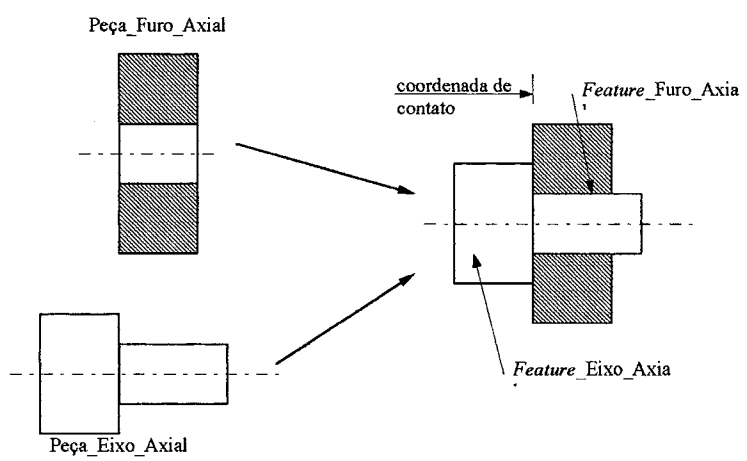


Figura 3.98 - Definições das peças e *features* envolvidas no contato axial.

Com relação ao contato diametral, pode-se definir:

- Diâmetro_Contato: cota nominal que é igual para o eixo e furo que estão montados (Figura 3.99);

- Peça_Eixo_Diâmetro: corresponde à peça que possui a *feature* eixo sobre o qual está montada a peça que possui a *feature* furo (Figura 3.99);

Peça_Furo_Diâmetro: corresponde à peça que possui a *feature* furo sob o qual está montada a peça que possui a *feature* eixo (Figura 3.99);

- Feature_Eixo_Diâmetro: corresponde à *feature* eixo que possui uma *feature* furo de outra peça montada sobre este eixo (Figura 3.99);

- Feature_Furo_Diâmetro: corresponde à *feature* furo que possui uma *feature* eixo de outra peça montada dentro desse furo (Figura 3.99).

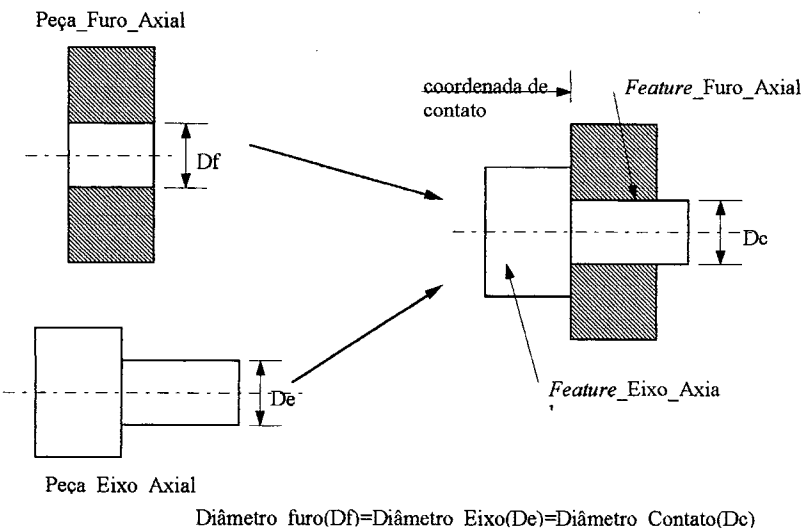


Figura 3.99 - Definições das peças e *features* que possuem contato diametral.

- Semi_Ângulo: corresponde ao ângulo de inclinação da reta que representa a superfície cilíndrica ou cônica. Se a *feature* é cilíndrica, o Semi_Ângulo é zero; se a *feature* é cônica, o Semi_Ângulo é diferente de zero;

- Tipo_*Feature*: caracteriza o tipo de *features* que estão em contato diametral, as quais, no caso, podem ser cilíndricas ou cônicas.

3.9 - COTAS E TOLERÂNCIAS

Um dos problemas a serem resolvidos, quando da cotagem da peça, é a escolha de quais das cotas devem receber tolerâncias, ou seja, quais as dimensões que vão influir no produto montado. Essa decisão somente é possível com uma análise do conjunto montado (produto), verificando a importância de cada superfície e a cota envolvida. Dessa forma, as superfícies a serem controladas, para que suas cotas permitam a montagem com seus pares e um funcionamento adequado, correspondem àquelas que possuem contato com outras superfícies de outras peças.

O caminho mais simples consiste em procurar os acoplamentos eixo/furo que o conjunto apresenta, pois são essas as partes críticas quando da usinagem e montagem, ou seja, obter uma cota na peça que permita a montagem. Outras superfícies que não possuem acoplamentos são dispensáveis ao controle dimensional rigoroso.

Os contatos axiais revelam as superfícies que devem ter o controle na direção axial do centro de rotação da peça, as quais são perpendiculares ao centro de rotação de peças rotacionais. Se uma peça possui contatos em duas superfícies pertencentes a ela com outras duas peças distintas, as duas superfícies devem possuir uma cota controlada (tolerância) entre elas para que, quando montadas, possam ocupar a posição adequada no conjunto, servindo de superfícies de referência (Figura 3.100).

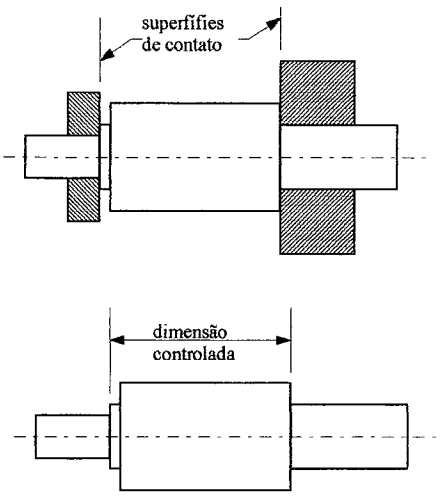


Figura 3.100 - Conjunto montado e cota a ser controlada na peça.

As cotas e tolerâncias na representação de uma peça são importantes, pois, através dessas informações, é possível interpretar as suas relações de montagem com as outras peças do conjunto.

3.9.1 - IDENTIFICAÇÃO DAS COTAS A SEREM CONTROLADAS

Quando da elaboração do projeto detalhado, a primeira fase corresponde à execução da representação gráfica com a utilização de cotas nominais, ou seja, não são considerados os problemas de desvios e funcionalidade dos acoplamentos. Para introduzir as tolerâncias, deve-se identificar, primeiramente, quais os elementos de maior importância no conjunto e que devem receber uma cota controlada.

Na análise de peças rotacionais, dois tipos de cotas se tornam primordiais: as cotas diametrais e as axiais. Dentre as cotas a serem controladas, pode-se identificar as cotas intra-*features* e as cotas inter-*features*. As cotas ditas intra-*features* correspondem àquelas referentes a uma única *feature* numa peça, não levando em conta as outras existentes e vizinhas. O diâmetro de uma *feature* está nessa categoria, o que pode ser observado na Figura 3.101, pelas cotas D1, D2 e D3. Cota inter-*feature*, por sua vez, corresponde àquelas cotas que relacionam mais de uma *feature* numa mesma cota estabelecida. Por exemplo, as cotas axiais L1, L2 e L3 na Figura 3.101 são consideradas cotas inter-*feature*; mesmo a cota L1 que representa exatamente o comprimento de uma *feature* eixo.

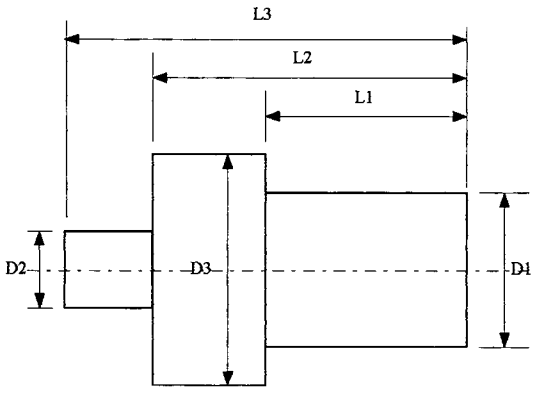


Figura 3.101 - Cotas intra e inter-*features*.

As cotas intra-*features* são consideradas atributos de uma *feature*, ao passo que as cotas inter-*features* são atributos da peça. O motivo dessa consideração são os exemplos do item 2.10.2, que mostram que acotagem axial é função exclusiva da peça e de sua aplicação.

3.9.1.1 - COTAS DIAMETRAIS A SEREM CONTROLADAS

A primeira análise é realizada com relação aos diâmetros, onde procuram as peças que possuem um acoplamento eixo/furo de mesmo diâmetro nominal e que estão montadas, conforme a Figura 3.102.

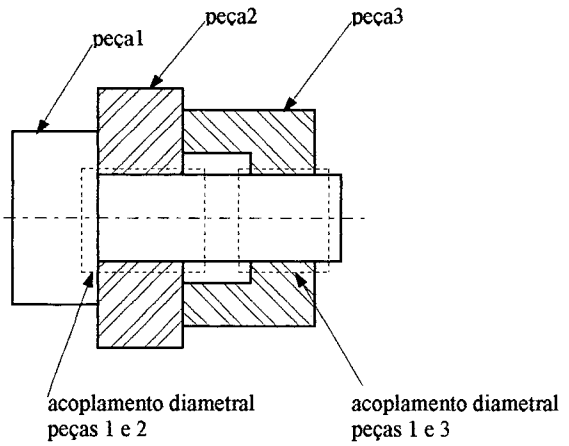


Figura 3.102 - Montagem diametral.

O sistema FeatCAD-2D inicia a pesquisa por uma peça que tenha um furo e procura uma outra que possua um eixo de mesmo diâmetro nominal para ser candidata à montagem. Se o eixo e o furo são sobrepostos como mostra a Figura 3.102, para o caso da montagem entre as peças 1 e 2 e entre as peças 1 e 3, essa é considerada uma montagem. Logo, as cotas a serem controladas são mostradas na Figura 3.103, sendo dadas por D12, d21 e d32, que representam as dimensões diametrais identificadas a serem controladas (Figura 3.103).

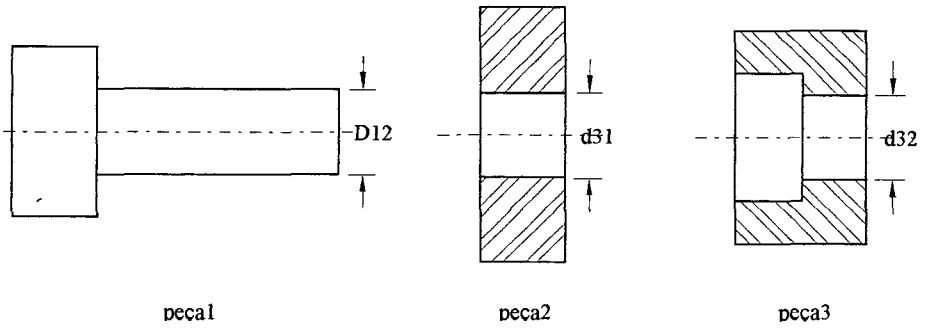


Figura 3.103 - Cotas diametrais a serem controladas.

As condições de montagem já foram estabelecidas no item 3.8, onde são identificados os contatos tanto diametrais como axiais. Como esses dados já estão à disposição, basta analisá-los e identificar em quais peças e *features* as cotas diametrais devem ser controladas.

O próximo passo corresponde a escolher as tolerâncias que melhor se adaptem ao referido acoplamento, quando a escolha é feita interativamente com o sistema, através do qual se buscam, num banco de dados, informações a respeito dos tipos de acoplamentos existentes e as tolerâncias e ajustes recomendados para tais aplicações.

3.9.1.2 - COTAS LONGITUDINAIS A SEREM CONTROLADAS

As cotas longitudinais são identificadas em função dos contatos axiais realizados entre as peças. Desse modo, o sistema FeatCad-2D procura pelas faces das *features* coincidentes de peças diferentes, as quais são consideradas como aptas a representar uma montagem (possuem o contato diametral já definido como um acoplamento aceito).

Assim, o sistema inicia a pesquisa a partir de uma peça do conjunto, procurando uma outra peça que possua uma *feature* de face coincidente com uma das *features* da primeira; se existe uma face coincidente, essa passará a ser uma superfície de referência de contato. Na Figura 3.104, está representado um conjunto onde estão identificadas duas superfícies de contato, uma entre a **peça1** e a **peça2** e a outra entre a **peça2** e a **peça3**, superfícies essas também já conhecidas através da análise do conjunto realizada no item 3.9 e identificado através dos atributos da Tabela 1 (Atributos dos tipos de contato). Isso é feito através do plano de contato, definido como *Coordenada_Contato*, e das respectivas peças e *features* envolvidas.

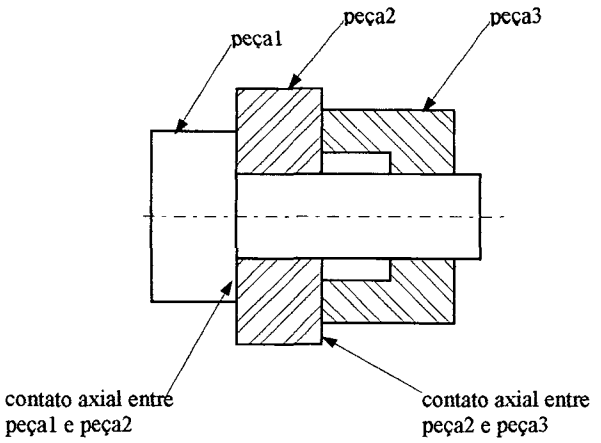


Figura 3.104 - Contatos axiais

Na Figura 3.105, está o resultado da identificação das superfícies de contato que ocorrem em cada peça, as quais passam a ser denominadas de superfícies de referência para que se possa executar a cotagem. Essas superfícies de referência são: **F12** para a **peça1**, **F21** e **F22**, para a **peça 2**, e **F31**, para a **peça3** (Figura 3.105).

Como se pode observar nas Figuras 3.99 e 3.100, as superfícies **F12** e **F21** são coincidentes, bem como **F22** e **F31**. Na Figura 3.105, estão representadas as peças individualmente, com a identificação das superfícies de referência.

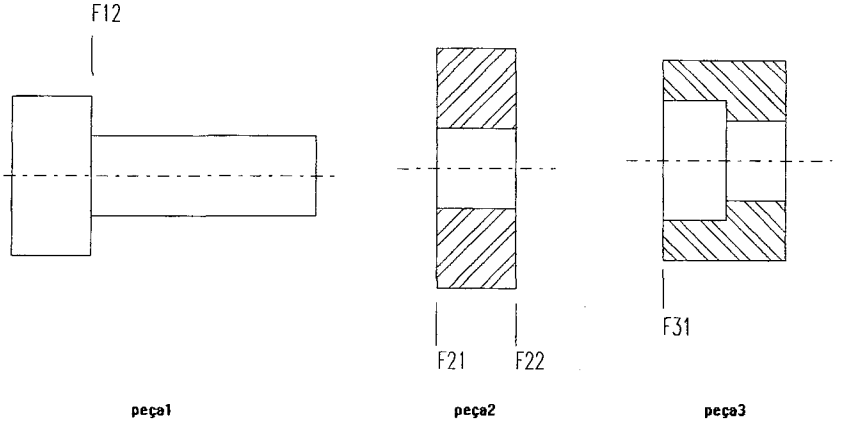


Figura 3.105 - Superfícies de referência de montagem e cotagem.

A identificação das cotas longitudinais não é, porém, tão simples como a realizada com as cotas diametrais. No caso dessas, havendo o contato entre duas superfícies de duas *features* (eixo/furo) de peças distintas e sobreposição, isso é suficiente para caracterizar que a cota do

diâmetro deve ser controlada. No caso das cotas longitudinais, essas são resultantes de cada conjunto montado e do seu funcionamento.

O que há até o momento são informações das superfícies de referência que foram identificadas na análise do conjunto como superfícies de contato, as quais, quando individualizadas em cada peça, passam a ser denominadas de superfícies de referência para a cotagem da peça. Antes, porém, de se efetuar a cotagem, é necessário escolher as cotas de maior importância para a peça com relação ao conjunto no qual está inserida.

3.9.1.2.1 - DEFINIÇÃO DAS COTAS LONGITUDINAIS

No caso das cotas longitudinais a serem controladas, uma análise mais rigorosa deve ser efetuada, ou seja, definir quais são, na realidade, as cotas longitudinais, sua disposição e o que essas cotas representam. Para iniciar a análise, são conhecidas as superfícies de referência que foram identificadas na análise do conjunto; logo, essas superfícies servem como ponto de partida para a definição da posição das cotas. São essas referências a base de todas as cotas longitudinais numa peça.

Como se pôde observar na revisão bibliográfica (item 2.10.2), essa superfície de referência pode estar localizada em qualquer parte da peça em função de como ela está montada (superfícies de contato axial). Como a análise se refere à peça isolada, alguns conceitos são introduzidos de forma simples para obter-se uma cotagem que leve em conta as superfícies de referência.

Uma cota é definida para o sistema como possuindo um valor nominal (Figura 3.106), um ponto inicial, um ponto final, de forma que a diferença positiva desses dois pontos corresponde ao valor da cota. A essa cota podem ser associadas as tolerâncias. Se essa cota for colocada numa representação gráfica, deverá ser definida a sua posição com relação à peça (posição da linha de cota no desenho); para isso, é necessário organizar a distribuição das demais linhas de cotas no respectivo desenho.

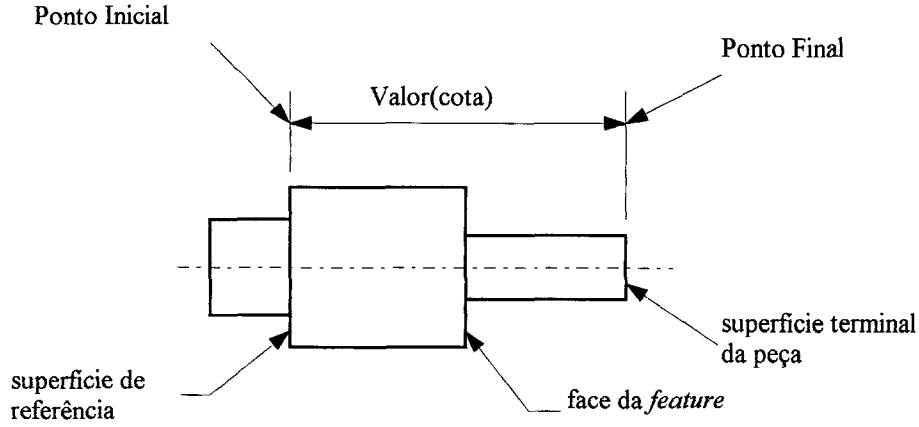


Figura 3.106 - Definição de uma cota.

Uma superfície de referência é tomada como ponto inicial de uma cota (F12) (Figura 3.107), e a superfície terminal que recebe o ponto final da dimensão pode ser a face de uma *feature* ou de uma peça, isso para o caso de a peça possuir somente uma superfície de referência (Figuras 3.107 (a) e Figura 3.106). Assim, L1 e L2 partem do ponto inicial em F12, e o ponto final de L1 corresponde à face final do eixo, que coincide com a superfície terminal da peça à direita (Figura 3.107(a)). O ponto final de L2 está situado na face inicial da *feature* eixo e na superfície terminal à esquerda da peça.

No caso de uma peça que possui suas duas superfícies terminais também de referência (F21 e F22), como na Figura 3.107 (b), uma será o ponto inicial e a outra, o ponto final da cota, no que resulta a cota L3. Na Figura 3.107 (c), tem-se o caso de uma peça com uma única superfície de referência (F31) coincidente com a superfície terminal da peça, sendo a mesma o ponto inicial para as cotas L4 (externa) e L5 (interna).

Pode-se dizer que, numa cota que possui um ponto inicial sobre uma superfície de referência, o ponto final dessa cota pode estar localizado sobre a face da *feature* mais próxima que não seja coincidente ou, então, numa das extremidades da peça.

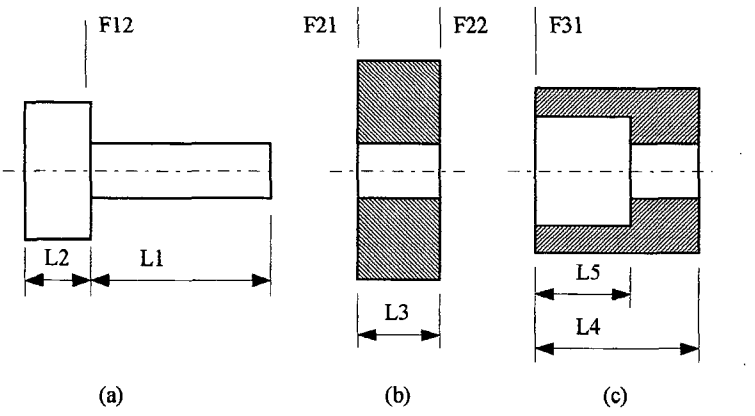


Figura 3.107 - Cotagem com superfícies de referência.

Caso exista mais de uma superfície de referência, e essas não se localizam nas extremidades da peça, elas podem assumir, ao mesmo tempo, ponto inicial de uma cota e ponto final de outra cota.

Na Figura 3.108, têm-se as superfícies de referência F1 e F2, que não correspondem às extremidades da peça; logo, as cotas L1, L2 e L3 possuem o ponto inicial em F1. A cota L4 está entre duas superfícies de referência, podendo qualquer uma delas receber o ponto inicial ou final. Finalmente, a cota L5 tem o ponto inicial sobre F2.

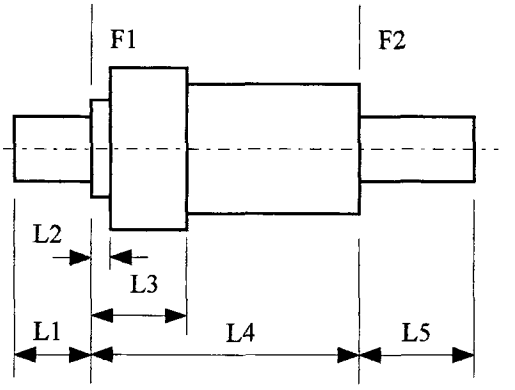


Figura 3.108 - Cotas logitudinais com duas superfícies de referência.

Se, numa mesma peça, há duas superfícies de referência, essas devem possuir uma cota de controle para o seu posicionamento entre si. Para as outras cotas, utiliza-se o critério de que a posição da face da *feature* que estiver mais próxima da superfície de referência corresponderá ao ponto final dessa cota, e o ponto inicial estará na superfície de referência (caso das cotas L2 e L3 na Figura 3.108 e, também, das cotas L1 e L5).

3.9.2 - ESCOLHA DAS TOLERÂNCIAS

Com a análise do conjunto, definem-se as informações de contato diametral e axial, bem como a especificação das cotas mais relevantes a serem controladas, com o que, então, as tolerâncias podem ser especificadas. Com a identificação dos pares eixo/furo montados, é possível especificar as tolerâncias dos respectivos diâmetros, o que é feito interativamente pelo usuário, ou seja, identificado o acoplamento no qual se deseja introduzir as tolerâncias, é apresentada uma série de questões a serem respondidas; a partir das respostas dadas, o sistema seleciona o par adequado, tolerâncias e ajustes.

Esse trabalho é feito através de um banco de dados no qual estão armazenadas as especificações conhecidas e que servem de base para a escolha. Assim, as cotas que fazem parte da montagem recebem especificação dimensional adequada à sua função, enquanto as outras seguem uma especificação considerada normal para aquele tipo de aplicação, que poderá também ser selecionada.

Com a especificação dos contatos axiais, é possível identificar as faces planas das *features* que servem de apoio na montagem e estabelecer um critério para o cotagem dessas e a necessidade de tais cotas serem controladas ou não, além de, no caso de serem controladas, qual a especificação que devem receber.

3.9.3 - COTAS LONGITUDINAIS E O CONJUNTO

Após a especificação das cotas e tolerâncias de uma peça, é necessária a realização de uma análise para a verificação das variações de cada peça sobre o conjunto montado. De acordo com as variações de cada peça quando essas são montadas, o somatório das variações pode resultar em folgas excessivas ou cotas que não permitem a montagem. Assim, é necessário que o somatório das cotas envolvidas de cada peça seja verificado dentro do conjunto. Para isso, é necessário que se tenham descritas todas as cotas e tolerâncias de cada peça que foram definidas no conjunto. Com esses dados, é possível realizar a análise de dois modos : a) manualmente e b) automaticamente. Se realizada manualmente, o usuário escolhe cada cota a ser inserida na cadeia dimensional para a análise; a definição das cotas será feita pela escolha da *feature* a ser

analisada. Se a opção for para análise automática, o sistema FeatCAD-2D deve pesquisar e organizar a cadeia dimensional para executar a análise.

3.9.3.1 - REPRESENTAÇÃO DAS COTAS E TOLERÂNCIAS DIMENSIONAIS

Sabendo-se o que deve ser representado em termos de informações dimensionais, é necessário um modelo que possibilite que essas informações sejam codificadas da forma mais próxima possível da realidade a qual se deseja representar. Para a representação de cotas e tolerâncias, não é diferente.

Como foi definido anteriormente, há informações de cotas ditas intra e inter-*features* e, de acordo com o tipo, há um conjunto de informações. No caso das peças rotacionais, as cotas intra-*feature* se referem somente àquelas que cotam uma *feature* específica, não a relacionando com qualquer outra *feature* onde tal informação passa a ser um atributo específico da *feature*. Por exemplo, no caso de uma *feature* eixo cilíndrico, uma cota intra-*feature* é o diâmetro. Para sua completa definição, pode-se ter a posição em que essa cota diametral está relacionada, ou seja, à qual das faces ela está se referindo, pois, quando da aplicação de uma *feature* eixo ou furo cônico, há dois diâmetros diferentes numa mesma *feature*.

Como o diâmetro está sujeito à aplicação de tolerâncias para controlar o valor da cota, as tolerâncias, nesse caso, passam a ser definidas como intra-*features*. Uma tolerância passa, então, a ser um atributo da cota. Para quantificar a tolerância de uma cota, alguns atributos são necessários, os quais são descritos na tabela 3.5.

Tabela 3.5 - Atributos de tolerância.

TOLERÂNCIA
Qualidade_de_Trabalho
Ajuste
Afastamento_Superior
Afastamento_Inferior

No caso dos atributos de tolerâncias de uma cota diametral, pode-se ter associadas a cota nominal e a indicação da *feature*.

As cotas longitudinais necessitam, além do valor de controle como descrito anteriormente, a localização da referida cota na peça, sendo consideradas neste trabalho como inter-*features*, pois relacionam mais de uma *feature*.

A cotagem longitudinal é função da peça; logo, uma cota longitudinal é definida com alguns atributos que permitem a sua identificação e localização (Tabela 3.6).

Tabela 3.6 - Atributos da cota longitudinal.

COTA LONGITUDINAL
Valor_Nominal
Ponto_Inicial
Ponto_Final
Referência
Sentido
Tolerância

O atributo Tolerância é agregado à Cota Longitudinal com os atributos definidos anteriormente.

3.9.3.2 - CADEIA DE TOLERÂNCIAS

Conhecendo-se as cotas funcionais longitudinais e as referências de contato entre as peças, é possível identificar automaticamente a cadeia de tolerâncias ou várias cadeias que possam existir numa montagem. Como cada cota longitudinal é definida com um ponto inicial e um final (Tabela 3.6), torna-se possível identificar as cotas que possuem ligação para formar uma cadeia dimensional entre as peças na montagem.

3.9.3.3 - REPRESENTAÇÃO DAS TOLERÂNCIAS GEOMÉTRICAS

As tolerâncias geométricas também podem ser divididas em intra e inter-*features*, o que se deve ao fato de haver tolerâncias que controlam a forma da *feature* em questão, como também aquelas tolerâncias que controlam a relação de posição entre as diversas *features* de uma mesma peça.

Assim, as tolerâncias geométricas intra-*features* são as seguintes: retilineidade, planicidade, circularidade, forma de linha qualquer e forma de superfície qualquer. Já, as tolerâncias geométrica inter-*features* podem ser assim relacionadas: paralelismo, perpendicularidade, inclinação, posição, concentricidade e coaxialidade, simetria e batimento (maiores detalhes sobre tolerâncias geométricas podem ser vistas em Foster (1992) e Gooldy (1995)).

Lin (1995) ressalta a importância da utilização dos símbolos gráficos na interface gráfica em vez de palavras para a descrição, o que apresenta como vantagens o seguinte:

1) símbolos possuem significado universal; as palavras, por sua vez, podem ser interpretadas diferentemente, o que resulta em erros;

2) os símbolos são fáceis de escrever e podem ser localizados facilmente em qualquer lugar;

3) os símbolos podem ser utilizados em desenhos de modo uniforme e claro.

A representação gráfica das tolerâncias geométricas é feita através de um quadro retangular que é dividido em dois ou três espaços de acordo com o tipo de tolerância a ser representada. Na Figura 3.109, estão representados os quadros, bem como as informações que devem ser descritas em cada célula do símbolo.

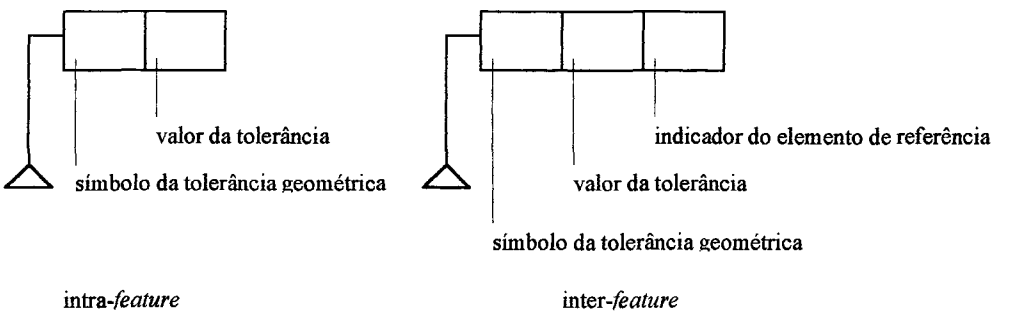


Figura 3.109 - Representação gráfica das tolerâncias geométricas nas *features*.

A representação das informações de tolerâncias geométricas na forma de classes e atributos pode ser feita como mostrado na Tabela 3.7.

Tabela 3.7 - Atributos das tolerâncias geométrica nas *features*.

Tolerâncias Geométricas	
Intra-*feature*	Inter-*feature*
Tipo	Tipo
Valor	Valor
Condição1	Condição1
---	Referência
---	Condição2

Aqui também se considera que esse tipo de tolerância está relacionada com a aplicação da peça, de forma que uma peça pode ter diferentes tolerâncias geométricas em função da sua montagem em diferentes produtos.

3.9.4 - ACABAMENTO DAS SUPERFÍCIES

O processo de usinagem para obter acabamento superficial rigoroso somente é aplicado àquelas superfícies que têm contato com outras superfícies, ou seja, àquelas superfícies que trabalham em contato com outras, as quais, em função da aplicação, necessitam de um tipo de acabamento específico ou, em função do seu posicionamento, necessitam de dimensões muito precisas. As outras superfícies que não possuem contato terão uma especificação definida pelo usuário de acordo com o tipo de peça.

Dessa forma, é possível relacionar o acabamento superficial às superfícies que possuem contato, ou seja, são também aquelas que recebem o controle dimensional mais rigoroso e que são controladas. Superfícies que não possuem contato receberão um acabamento que pode ser função do processo de obtenção da peça (p. ex., fundição, usinagem, etc.), mas, em geral, podem ter um acabamento superficial muito inferior.

A escolha do acabamento superficial está relacionada com a função da superfície, isto é, àquelas que possuem cotas controladas e que correspondem às cotas diametrais e às longitudinais, que, no caso, já estão definidas.

Sabendo qual é a cota controlada, identificam-se as superfícies que devem possuir o acabamento superficial, o que é feito através da relação existente entre qualidade de trabalho e rugosidade, que são descritas na Tabela 3.8. De posse dos dados dessa tabela, é possível

relacionar o acabamento em função da qualidade de trabalho IT e da cota que está sendo analisada, resultando na rugosidade superficial (Ra).

Tabela 3.8 - Relação entre tolerâncias ISO e a rugosidade superficial (AGOSTINHO, 1977).

Ra (μm)					
ISO	Dimensão (mm)				
	3	3-18	18-80	80-250	250
IT6	0,2	0,3	0,5	0,8	1,2
IT7	0,3	0,5	0,8	1,2	2
IT8	0,5	0,8	1,2	2	3
IT9	0,8	1,2	2	3	5
IT10	1,2	2	3	5	8
IT11	2	3	5	8	11
IT12	3	5	8	12	20
IT13	5	8	12	20	----
IT14	8	12	20	----	----

Logo, o acabamento superficial é referenciado com relação à rugosidade, que, para o planejamento de processos, a rugosidade pode ser relacionada com o tipo de processo que resulta o acabamento desejado relacionado a forma da peça.

Na realidade, a Tabela 3.8 serve de orientação para o usuário, devendo haver outras alternativas para que o mesmo possa escolher adequadamente o acabamento superficial, com outro conjunto de informações, como, por exemplo, uma tabela de exemplos de aplicações de rugosidade em determinados tipos de peças. O objetivo é garantir que a escolha de uma tolerância seja compatível com o acabamento desejado da superfície, bem como não se deve exigir de uma tolerância muito larga um acabamento muito superior, o que somente seria compatível com tolerâncias mais apertadas.

Sendo conhecida a superfície a ser usinada e o acabamento a ser obtido, é possível, no planejamento de processos, definir as superfícies a serem usinadas. Essa identificação é muito útil quando do planejamento do processo de usinagem de peças fundidas, as quais somente devem sofrer o acabamento nas superfícies estritamente necessárias. Isso não ocorre quando a peça é resultante da usinagem de barras, da qual a maior parte das superfícies são resultantes.

3.95 - ESCOLHA DAS TOLERÂNCIAS DIMENSIONAIS

Após a identificação das cotas diametrais a serem controladas, é necessário escolher as tolerâncias a serem utilizadas no par de montagem identificado, o que é feito em função da aplicação do par de montagem e fica a cargo do usuário. Esse deve possuir informações a respeito de como o referido par de acoplamento funciona, como, por exemplo, conhecer as folgas necessárias a serem utilizadas ou um tipo de aplicação semelhante à que ele deseja.

Quando da elaboração de um projeto, essas informações devem estar disponíveis para o usuário de uma forma organizada, para que também as novas informações possam ser colocadas à disposição em função da experiência adquirida. Para Weill et al. (1988), escolha das tolerâncias é baseada na função. Por função é generalizada a aptidão para montagem de peças, mas também, em muitos casos, aptidão para o funcionamento de um sistema mecânico, o qual é dependente de pequenas forças de atrito, precisão geométrica, exatidão cinemática, e assim por diante. Assim, um banco de dados de tolerâncias, ajustes e aplicações características pode ser disponibilizado ao usuário, de forma a organizar as informações existentes na literatura como também aquelas do uso da empresa. Dessa forma, tem-se um ponto de partida para a inserção de tolerâncias num projeto, a partir do que se pode buscar a otimização das mesmas. As informações, inicialmente, devem representar a funcionalidade do produto, para, após, verificar-se quanto à manufatura, o que não será abordado no presente trabalho.

Para especificar as tolerâncias de projeto, devem estar disponíveis nesse banco de dados os afastamentos superior e inferior em função do tipo de ajuste e qualidade de trabalho (dados constantes das normas), classificados de acordo com os diâmetros. Tais informações devem, então, ser transferidas para a estrutura do produto, sendo anexadas aos atributos que representam essas informações numa peça ou numa *feature*.

Para determinar os ajustes e tolerâncias diametrais de um par acoplado, pode-se trabalhar a partir de três opções:

1) A partir do conhecimento do diâmetro nominal de montagem, buscam-se exemplos de aplicação do acoplamento semelhantes ao caso em análise, como, por exemplo, o acoplamento eixo/engrenagem. De posse dessas informações, o sistema deve fornecer o par de ajustes e tolerâncias recomendados, os respectivos afastamentos para o eixo e o furo, informando a qualidade de trabalho (IT), bem como a folga ou interferência.

2) Também a partir do conhecimento do diâmetro nominal de montagem, o usuário especifica a precisão desejada e obtém o par de montagem, os respectivos afastamentos e tolerâncias relacionados ao eixo e ao furo; identifica, ainda, o tipo de aplicação conhecida para o par de montagem, a folga ou interferência. A precisão é classificada em extrapreciso, preciso, média precisão e comum.

3) A última opção corresponde à obtenção de informações de elementos padronizados, como, por exemplo, rolamentos, quando uma série de perguntas devem ser respondidas para a obtenção das informações. Tais dados podem ser fornecidos pelo fabricante do componente e cadastrados no sistema.

Assim, os dois primeiros tipos de pesquisa são de natureza genérica, a partir de informações subjetivas, procurando moldar o problema a um exemplo já conhecido, ao passo que, no último, a pesquisa é diretamente em função do produto. Esse tipo de consulta deve estar disponível não somente para ser acionada automaticamente pelo sistema quando da identificação de um par de peças acopladas, mas também de forma manual, de modo que o usuário possa efetuar consultas em qualquer tempo e independentemente da aplicação.

CAPÍTULO 4

IMPLEMENTAÇÃO DO MODELO

4.1 - PLATAFORMA E FERRAMENTAS UTILIZADAS

Com base na estrutura do sistema proposto, foi desenvolvido um protótipo baseado em softwares comerciais, os quais sofreram as adaptações necessárias para que fosse obtido o desempenho esperado. Os módulos principais são uma interface gráfica, um sistema especialista e um banco de dados, todos funcionando de modo integrado.

Como plataforma gráfica, foi utilizado o software AutoCad na sua versão R12 "for WINDOWS" 3.11, do qual foram utilizadas apenas as suas primitivas gráficas e a interface para a comunicação com o usuário. Uma das razões dessa escolha foi que ele permite programação em linguagem C/C++; além disso, é um software popular, permitindo que futuros desenvolvimentos possam ser implementados em empresas usuárias.

Para o sistema especialista, foi utilizado um *shell*, o CLIPS (*C Language Integrated Production System*) na sua versão 6.0, o qual se utiliza de regras e fatos para modelar as informações, software este também construído em linguagem C, do que decorreu a escolha pela sua possibilidade de comunicação com o software de CAD.

Como banco de dados, foi escolhido o DELPHI da Borland na sua versão 1.0, o qual é programável em linguagem Pascal, contudo, com os recursos da programação "for WINDOWS" e com o uso de DLL[1], possibilita o interfaceamento com a linguagem C/C++.

Finalmente, foi utilizado o software de programação em linguagem C/C++ da Borland, na sua versão 4.5 "for WINDOWS", como meio de integração e comunicação entre os diversos softwares especializados. Deve-se lembrar que todo o sistema FeatCAD-2D trabalha sob o ambiente operacional "WINDOWS" e que todos os softwares utilizados permitem programação orientada para objetos.

[1] *Dynamic Library Linguge*

Desse modo, obteve-se a integração desses softwares os quais rodam por trás do sistema de CAD, tornando-se transparentes ao usuário.

O protótipo implementado roda em computadores da classe PC linha 486, com configuração mínima de 66MHz e 8MB de memória RAM, trabalhando de forma isolada (monousuário).

4.2 - REPRESENTAÇÃO DAS INFORMAÇÕES

A representação das informações no sistema pode ser dividida em dois formatos básicos: a) formato interno, que corresponde às informações propriamente ditas, representadas internamente ao sistema, dentro de uma estrutura de dados no formato computacional e b) formato externo, que é a representação das mesmas informações em nível de usuário, ou seja, no formato que o usuário é capaz de compreender.

Para que essas duas representações possam ser coerentes, deve haver uma correspondência entre informações no formato computacional e no formato gráfico. Do mesmo modo, cada manipulação desejada pelo usuário e representada graficamente deverá ter a sua correspondente no formato interno. Desse modo, a estrutura do produto, em nível de projeto detalhado e que é representada internamente, deve possuir, simultaneamente, um formato externo para o usuário onde todos os níveis do produto estarão representados.

4.3 - VISUALIZAÇÃO EXTERNA DAS INFORMAÇÕES

Na visão do usuário, o sistema deve apresentar as informações de modo organizado e em separado, possuindo elementos que o ajudem a identificar e separar rapidamente cada informação a ser utilizada. Assim, a estrutura do produto descrita no modelo proposto deve ser de alguma forma visualizada pelo usuário através da interface gráfica, de modo a permitir-lhe identificar corretamente em que parte da estrutura do produto está trabalhando.

Uma das maneiras encontradas para efetuar-se este trabalho é através da utilização do gerenciador do produto (ver item 4.5), que tem a função de permitir ao usuário percorrer a estrutura, informando-o graficamente sobre o que está ocorrendo. Assim, o usuário fica sabendo

onde está situado na estrutura de dados, podendo manusear as informações tanto de natureza gráfica quanto tecnológica: a primeira é necessária para que ele visualize as informações; a segunda, para possibilitar-lhe as análises técnicas.

A visualização da estrutura do produto pelo usuário é possível com a utilização das camadas (*layers*) que os CAD comerciais já fornecem no sistema, por meio das quais é feita a individualização das informações a serem transmitidas ao usuário.

4.3.1 -VISUALIZAÇÃO DO CONJUNTO E SUBCONJUNTO

Uma camada corresponde ao produto, outra ao conjunto (Figura 4.1); uma para o subconjunto e outra para a peça, sendo essa última relacionada com o seu respectivo subconjunto, e cada subconjunto relacionado ao conjunto principal. Uma descrição da estrutura das camadas é dada na Figura 4.3.

A utilização de camadas de representação possibilita a separação de cada um dos itens que compõem um produto quando da apresentação no CAD, de modo a serem ativadas apenas no momento de sua utilização.

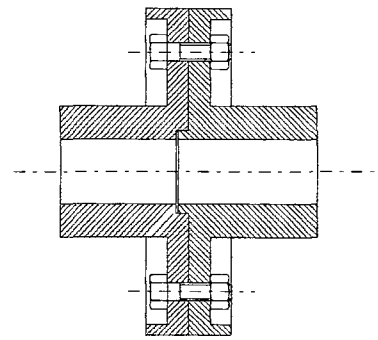


Figura 4.1 - Visão de todas as camadas - Conjunto.

Desse modo, cada camada está associada a informações na estrutura de dados onde está representado o produto através de um grupo de atributos. Assim, a primeira camada a ser criada corresponde à camada-conjunto, onde, nas representações gráfica e tecnológica, podem ser vistos todos os componentes da montagem (Figura 4.1).

Em seguida, é criada de forma automática a camada-conjunto-dimensão, que somente permite a representação de dimensões, tolerâncias e notas a respeito do conjunto. Também

podem ser visualizados nessa camada todos os subconjuntos e as respectivas peças que o compõem (Figuras 4.1 e 4.2). Essa camada recebe o mesmo nome da camada inicial, mais o sufixo "DD" para diferenciação.

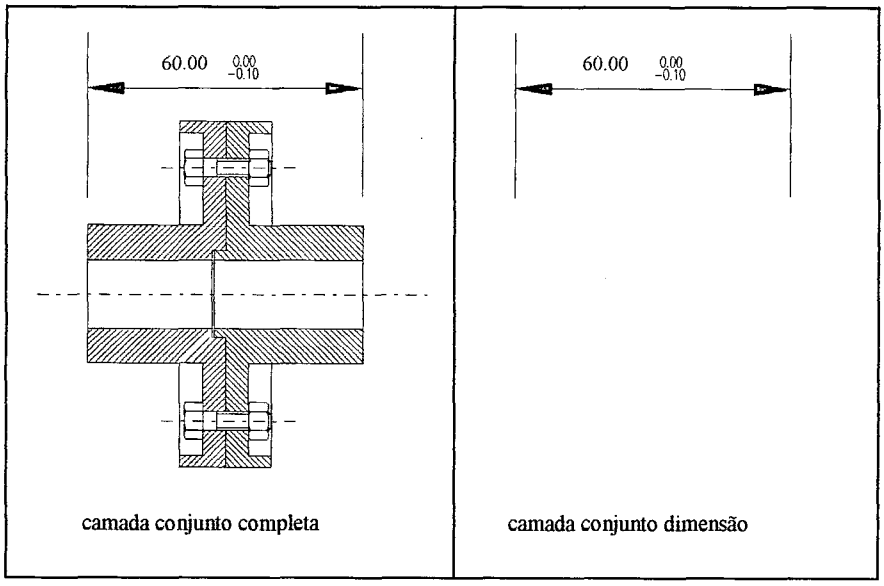


Figura 4.2 - Camadas do conjunto.

A sobreposição das diversas camadas de subconjuntos resulta na camada-conjunto. A sobreposição da camada-peça com a camada-peça-dimensão resulta na camada-projeto da peça, onde estão representados o desenho da peça, as dimensões, tolerâncias e notas individualizadas do conjunto, e as informações representadas no formato gráfico. Desse modo, cada camada pode possuir diversas subcamadas de acordo com a necessidade de representação de informações.

Igualmente, para cada subconjunto criado, é feita uma camada para representar o subconjunto e uma camada para representar dimensões e notas referentes a esse subconjunto. A camada-subconjunto comporta-se de modo semelhante ao conjunto. Em cada camada-subconjunto, estão representadas as peças que o compõem, não sendo permitido o acesso às peças para efetuar modificações, tanto da camada-subconjunto como da subconjunto dimensão.

Em cada camada-peça, são criadas também duas camadas, que correspondem à camada-peça, onde estão representadas as *features*, e à camada-peça-dimensão, onde estarão representadas as dimensões e notas com relação à peça. Na Figura 4.3, está representada a estrutura de camadas para a representação do produto.

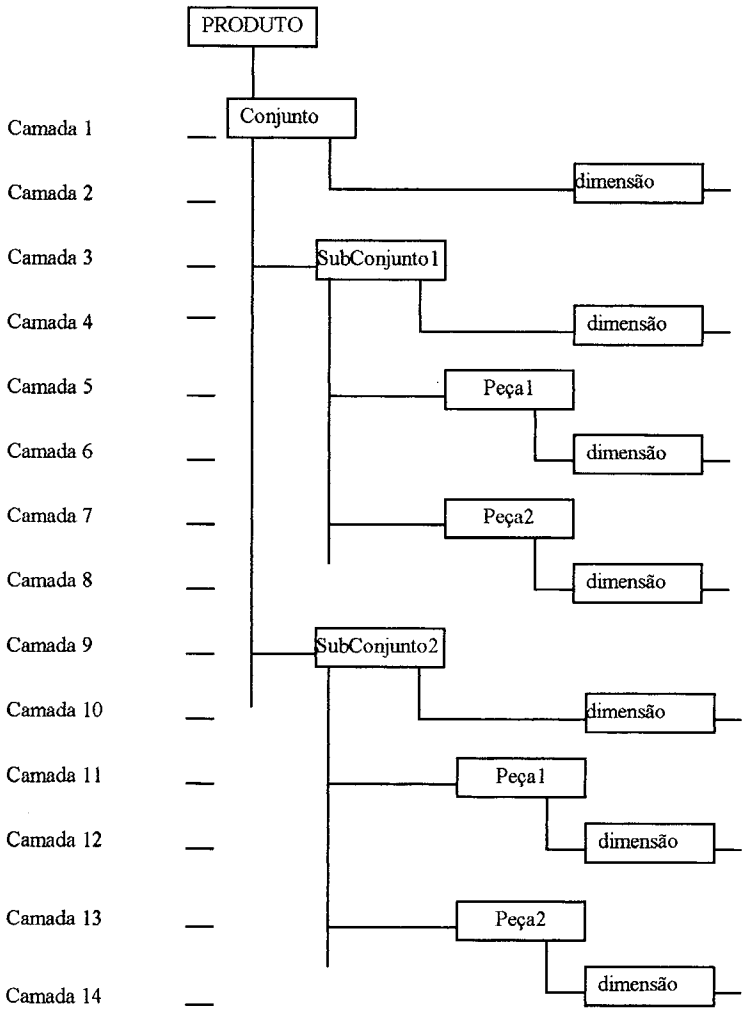


Figura 4.3 - Estrutura de camadas.

Assim, cada camada representada na Figura 4.3 refere-se à camada-dimensão de uma peça de um subconjunto (p. ex., a camada 12 refere-se à dimensão da peça1 do subconjunto2). É lógico que, em lugar de um número, estará o nome da peça, do subconjunto ou conjunto com o nome da camada para facilitar a sua visualização pelo usuário.

4.3.2 - VISUALIZAÇÃO DE UMA PEÇA

Como cada peça é representada por uma camada, no momento em que o sistema é informado de que está sendo criada uma nova peça, a qual possui um nome, automaticamente esse nome será o mesmo da respectiva camada. Desse modo, uma peça somente pode ser manuseada (criar, excluir, mover) se a referida camada estiver ativa; através do sistema de

navegação, a camada será ativada permitindo que o usuário efetue, então, as tarefas que desejar. Durante a fase de trabalho, todas as peças que compõem um subconjunto estarão visíveis ao usuário.

Estando o usuário num subconjunto específico, ele somente terá acesso às peças que compõem esse subconjunto; para acessar as de outro subconjunto, deverá utilizar o mecanismo de navegação, procurando o subconjunto desejado e, então, a peça.

A representação gráfica de uma peça é feita através da utilização das *features* disponíveis na biblioteca já configurada. Através das operações de instanciação, é possível selecionar uma *feature* da biblioteca e inseri-la numa peça, obedecendo às regras impostas à mesma. Assim, numa mesma camada que representa uma peça, estarão representadas várias *features*.

4.3.3 - VISUALIZAÇÃO DAS DIMENSÕES

No sistema FeatCAD-2D, convencionou-se representar separadamente a geometria que representa as formas das peças da geometria que representa o dimensionamento das mesmas. Para isso, foi criada a camada-dimensão, onde somente pode ser representada a geometria que corresponde à representação de elementos dimensionais, não sendo permitido, a partir dessa camada, o acesso a partes do desenho e à sua manipulação.

Assim, para cada peça será, automaticamente, criada uma camada-dimensão, onde estarão representados os atributos referentes ao dimensionamento, notas e tolerâncias, possuindo essas entidades gráficas uma representação correspondente na *feature* à qual estão associadas.

4.4 - GERENCIAMENTO DO PRODUTO

Para que a estrutura do produto possa ser criada e utilizada computacionalmente, é necessário um sistema gerenciador do produto, cuja função é servir de mecanismo para que a estrutura de dados possa ser criada e, ao mesmo tempo, possa ser percorrida para que se efetuem as funções necessárias ao desenvolvimento do produto. A primeira tarefa reservada ao gerenciador do produto é prover um mecanismo de criação da estrutura de dados, ou seja, ao ser definido o conjunto que representa o produto, esse deve inicializar a estrutura e armazenar as

informações referentes ao conjunto, bem como preparar a estrutura para que possam ser criados os respectivos subconjuntos.

Ao introduzir as informações de cada subconjunto, os quais são criados em qualquer seqüência em função do desenvolvimento do produto, o gerenciador deve preparar a estrutura para a introdução das respectivas peças que compõem cada subconjunto. Como as peças são criadas e desenvolvidas aleatoriamente, é necessário que o sistema de gerenciamento possua mais um mecanismo para essa tarefa, dito de navegação, o qual permite que o usuário possa se deslocar dentro da estrutura e preencher a estrutura de um subconjunto, uma peça ou uma *feature* a qualquer momento em que haja a necessidade.

A necessidade desse gerente do produto é conseqüência de os CAD comerciais não possuírem uma estrutura de dados adequada para tratar informações utilizando *features*, nem uma estrutura de produto, sendo, então, necessário criar uma estrutura específica para armazenar essas informações. Logo, como há uma estrutura para as *features* e uma estrutura que é própria do CAD, é necessário realizar uma comunicação adequada entre as duas de modo que as *features* possam ser representadas adequadamente no CAD. É possível, então, armazenar as informações tanto gráficas como tecnológicas, que serão apresentadas pelo CAD.

4.5 - ESTRUTURA DE CLASSES

A estrutura de classes é baseada no modelo do produto, ou seja, cada elemento significativo possui uma classe a representá-lo. Assim, tem-se:

PRODUTO;
  CONJUNTO;
    SUBCONJUNTO;
      PEÇA;
        FEATURE.

O produto resulta da condição de composição dessas classes, condição essa pertinente às linguagens de programação orientadas para objeto. Assim, as entidades manipuladas pelo sistema estão representadas pelas referidas classes.

4.6 - *FEATURES*

No sistema proposto, as entidades básicas a serem manipuladas são as *features*, que, no caso, são representadas por entidades geométricas bidimensionais. As *features* possuem uma representação gráfica e uma representação em nível de informação (representação interna ao sistema, descrita através de classes e atributos). Desse modo, elas seguem a classificação descrita no modelo proposto, onde são apresentadas as *features* básicas, as modificadoras, as combinadas e as de alto nível.

Assim, constroem-se algumas *features* que compõem as referidas primitivas (básicas) das quais, a partir da modificação (modificadoras), combinação e composição, podem ser obtidas outras mais complexas que mantêm a mesma característica das primitivas. Porém, essa flexibilidade é dada parcialmente ao usuário, visto que a customização pode ser efetuada de duas formas: via programação do sistema para criação das primitivas, a que o usuário comum não possui acesso, ou diretamente pela interface gráfica, onde o usuário, a partir das *features* básicas, pode obter novas *features*, ditas configuráveis, das quais, durante a construção, o sistema especialista verifica a consistência. Na verdade, no primeiro caso, é utilizada uma espécie de sublinguagem para a obtenção de outras *features* a partir das elementares (item 3.51).

4.7 - ASPECTOS DE IMPLEMENTAÇÃO DAS *FEATURES*

Conforme as descrições feitas anteriormente, a implementação das *features* propriamente ditas começa com as *features* básicas, ou seja, eixo e furo cilíndricos, o que obedece à concepção da utilização de *features* rotacionais. As *features* começam a ser definidas a partir da criação de uma peça, ou seja, sem a existência de uma peça, não é possível instanciar uma *feature* no sistema. Isso é controlado, inicialmente, a partir do menu do sistema CAD.

As duas *features* básicas, eixo e furo, mantêm uma relação de precedência que é sempre obedecida, isto é, uma *feature* furo não pode ser instanciada se antes não for instanciada uma *feature* eixo. Como as entidades gráficas que representam as *features* são as primitivas do CAD (para o sistema 2D, as primitivas geométricas utilizadas são a reta, o círculo e o arco de círculo), essas são construídas pelo sistema e identificadas internamente através do identificador próprio

do CAD, que é copiado para o sistema. O sistema de coordenadas utilizado para localizar cada *feature* é o absoluto, o que facilita sobremaneira as análises em peças rotacionais.

Como as *features* são representadas num sistema 2D, o sistema de coordenadas de cada *feature* possui somente duas dimensões; por simplificação, foram utilizadas a coordenada do centro de rotação da peça e coordenadas de cada face de cada *feature*.

A primeira *feature* eixo pode ser instanciada (localizada) em qualquer posição, sendo que, através dessa instanciação, serão passadas informações para a peça, que manterá as informações necessárias para as *features* posteriores. Cada *feature* instanciada é representada graficamente, porém permanece mantendo a sua independência, o que possibilita a identificação e a manipulação em nível gráfico. Em nível de informação, a ligação da *feature* com a sua representação gráfica é mantida através da estrutura de dados. Para a identificação de cada *feature* criada, é utilizado um identificador fornecido pelo AutoCAD, o qual serve de ponte entre as *features* instanciadas e as suas representações gráficas.

A descrição das *features* deve tomar como base a decomposição de cada uma em termos de *features* eixo e *features* furo, sempre descrevendo as informações que são realmente significativas para sua interação no referido contexto.

4.7.1 - ATRIBUTOS APLICADOS ÀS *FEATURES*

Os atributos aplicados às *features* são descritos na capítulo 3, item 3.5.3, onde são explicitadas as *features* de forma.

4.7.2 - *FEATURES* BÁSICAS

Pela classificação das *features*, foram consideradas como básicos o eixo e o furo, os quais, na implementação, adquirem importância igual frente à existência de outras *features*. Assim, a existência de *features* modificadoras internas e externas depende das básicas.

Na implementação, mesmo sendo consideradas ambas como básicas, há uma relação de precedência natural que deve ser obedecida, o que pode ser facilmente visto na medida em que um furo somente pode existir se há uma massa a seu redor. Portanto, é necessária a existência de

um eixo para que um furo possa existir, criando-se, assim, a estrutura para a existência de uma peça.

4.7.3 - *FEATURES* MODIFICADORAS

As *features* modificadoras somente podem ser implementadas via programação do sistema, sendo necessário, para isso, construir a classe, funções internas da classe, funções externas para instanciação e manuseio de informações.

O domínio dessas *features* deve ser definido quando da construção do sistema, pois considera-se extremamente difícil desenvolver um sistema que permita ao usuário executar essas alterações e inclusões de novas *features* modificadoras no sistema.

4.7.4 - *FEATURES* COMBINADAS

As *features* combinadas, como foram definidas na classificação das *features*, são resultantes da combinação das *features* básicas com modificadoras. Assim, a constituição de uma *feature* combinada necessita da existência de, no mínimo, uma *feature* básica, no caso o eixo.

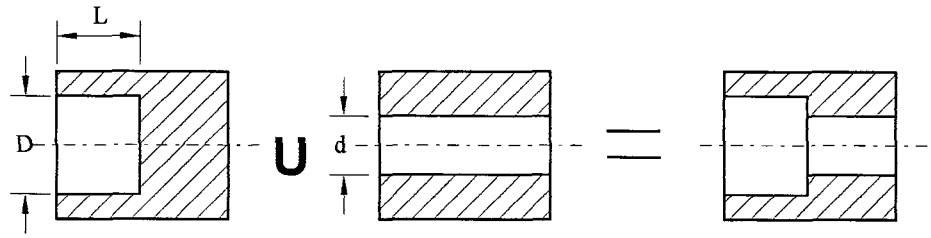
A implementação das *features* combinadas é resultante da utilização de funções semelhantes às funções de inserção das *features* básicas e modificadoras. Essas funções foram adaptadas para uma fácil utilização sem que haja a necessidade de serem construídas novamente, mas apenas utilizadas como se fossem uma linguagem de programação.

Com a implementação das *features* básicas e modificadoras, foi possível criar algumas *features* combinadas, descritas em seqüência.

4.7.4.1 - *FEATURE* COMBINADA FURO ESCALONADO PASSANTE

Uma *feature* furo escalonado passante (FFEP) é resultado da união de duas *features* básicas, que são: uma *feature* furo cilíndrico cego (FFCC) e uma *feature* furo cilíndrico passante (FFCP) (Figura 4.4).

Assim, FFEP = (FFCC) U (FFCP)



Atributos : D - diâmetro do furo cego
d - diâmetro do furo passante
L - profundidade do furo cego
D > d

Figura 4.4 - *Feature* combinada furo escalonado passante.

Como exemplo de programação de uma *feature* combinada furo escalonado passante, tem-se a aplicação das funções responsáveis pela sua criação:

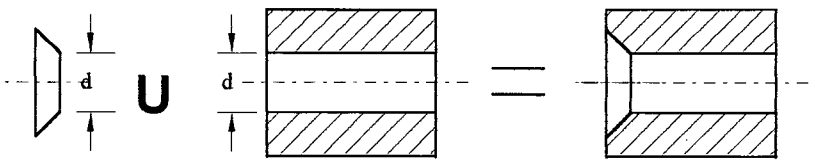
- FuroCego (diâmetro, profundidade, sentido);
- FuroPassante (diâmetro, profundidade, sentido),

onde os atributos da referida *feature*, que são os atributos das *features* furo cego e passante, são passados através de uma entrada adequada de informações. Os testes feitos pelo sistema especialista para a inserção de uma *feature* individualmente são válidos para a inserção de cada *feature* que compõe a *feature* combinada.

4.7.4.2 - *FEATURE* COMBINADA FURO ESCAREADO PASSANTE

Uma *feature* furo escareado passante (FFEsP) é resultado da união de uma *feature* básica mais uma modificadora, que são: uma *feature* furo cilíndrico passante (FFCP) e uma *feature* modificadora chanfro interno (FMCI), a qual possui o ângulo definido em 45° (Figura 4.5).

Assim, FFEP = (FFCP) U (FMCI).



Atributo : d - diâmetro do furo passante

Figura 4.5 - *Feature* combinada furo escareado.

4.7.5 - *FEATURES* COMPOSTAS

As *features* compostas, conforme a definição, são resultantes da reunião de *features* básicas, modificadoras, ou combinadas numa nova composição a ser definida. No presente trabalho, tais *features* não são implementadas. Dentre essas, pode-se ter uma composição de furos passantes no formato matricial ou no formato polar. Ao invés de furos passantes, pode-se ter furos escalonados cegos obedecendo a uma das disposições citadas ou a outra qualquer que venha a ser definida.

4.7.6 - *FEATURES* DE ALTO NÍVEL

As *features* de alto nível são construídas de modo semelhante às *features* combinadas, ou seja, utilizam-se as mesmas funções para a sua definição. Ao invés de ter-se novas *features* que irão complementar a descrição das peças, as *features* de alto nível, na realidade, representam peças, que, no caso, são peças de fornecimento de terceiros, e que podem ser padrão apenas da empresa ou padronizadas por normas. Como exemplo de *features* de alto nível normalizada, foi implementado o rolamento rígido de uma carreira de esferas.

4.7.6.1 - *FEATURES* DE ALTO NÍVEL PADRONIZADAS

Estas *features* correspondem àquelas que representam peças oriundas de terceiros e que possuem uma padronização já consagrada através de normas de âmbito nacional ou internacional. As *features* de alto nível padronizadas somente podem ser implementadas via

programação, isto é, somente podem ser inseridas no sistema a partir da execução de funções específicas, pois necessitam de mecanismos de interligação que, por enquanto, somente são obtidos através da programação.

Neste trabalho, foram implementadas algumas *features* padronizadas, sendo descrito a seguir um exemplo delas.

Feature de alto nível normalizada rolamento

A *feature* desse nível implementada foi um rolamento rígido de uma carreira de esferas, o qual foi representado através de *features* eixo e furo, como é a base do sistema de representação da estrutura de dados. A decomposição do rolamento é feita em função dos atributos que são relevantes para a tarefa de projeto, ou seja, o diâmetro externo, o diâmetro interno e a largura do rolamento, conforme Figura 4.6.

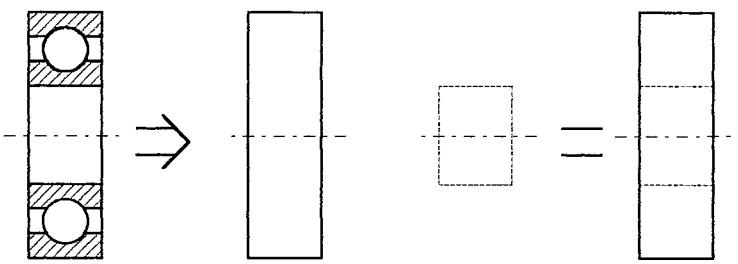


Figura 4.6 - Representação de uma *feature* de alto nível normalizada rolamento.

Em termos de representação gráfica, este trabalho não desenvolve a representação gráfica completa, como tradicionalmente é feito nas representações técnicas, que se encontra na Figura 4.6, à esquerda. Essa decisão foi tomada em razão dos objetivos do trabalho, que não têm por primordial o desenvolvimento das representações gráficas das *features*. Como a representação interna do rolamento possui apenas função de apresentação, a sua representação gráfica foi feita apenas em termos das *features* eixo e furo.

A *feature* rolamento é representada como uma peça qualquer, sendo decomposta na estrutura de dados como uma *feature* eixo e uma *feature* furo, possuindo uma classificação especial nos atributos, dos quais o atributo "Origem" foi definido como "normalizada". Uma informação importante que deve ser preservada é que essa peça deve ser identificada como um rolamento, o que é feito através do nome da *feature* quando da sua criação.

4.7.6.2 - *FEATURE* DE ALTO NÍVEL PADRÃO

A *feature* de alto nível padrão é semelhante a uma *feature* de alto nível normalizada; o que as diferencia é que a padrão é função da decisão interna da empresa. Desse modo, qualquer peça de uso comum da empresa que seja considerada como padrão pode ser transformada numa *feature*, facilitando, assim, o trabalho de representação.

Uma *feature* de alto nível padrão pode ser obtida de dois modos: o primeiro, através de funções de programação do sistema, quando é necessário manusear os arquivos-fonte, compilar os mesmos e obter o resultado; o segundo modo é diretamente da interface gráfica, ou seja, pode-se construir uma *feature* de alto nível padrão na interface gráfica e, após, obter uma *feature* à disposição do usuário, a qual pode, então, ser inserida a qualquer momento. Esse segundo modo é mais rápido e amigável, visto que qualquer usuário pode construir uma nova *feature* e utilizá-la livremente. A criação dessa nova *feature* é checada pelas regras do sistema especialista, portanto, é válida para a utilização.

A seguir, representa-se um exemplo de *feature* de alto nível padrão que foi implementada, sempre com base no conceito de decomposição da peça naquelas *features* que estão representadas na biblioteca.

Feature de alto nível padrão bucha

Esta *feature* é composta pelas *features* disponíveis na biblioteca e pode ser implementada via interface gráfica ou programação do sistema, de modo que passe a fazer parte integrante do mesmo. De acordo com a Figura 4.7, a representação desta *feature* de alto nível pode ser obtida conforme a seqüência descrita, sendo que, ao ser criada via interface gráfica, a validação é feita automaticamente.

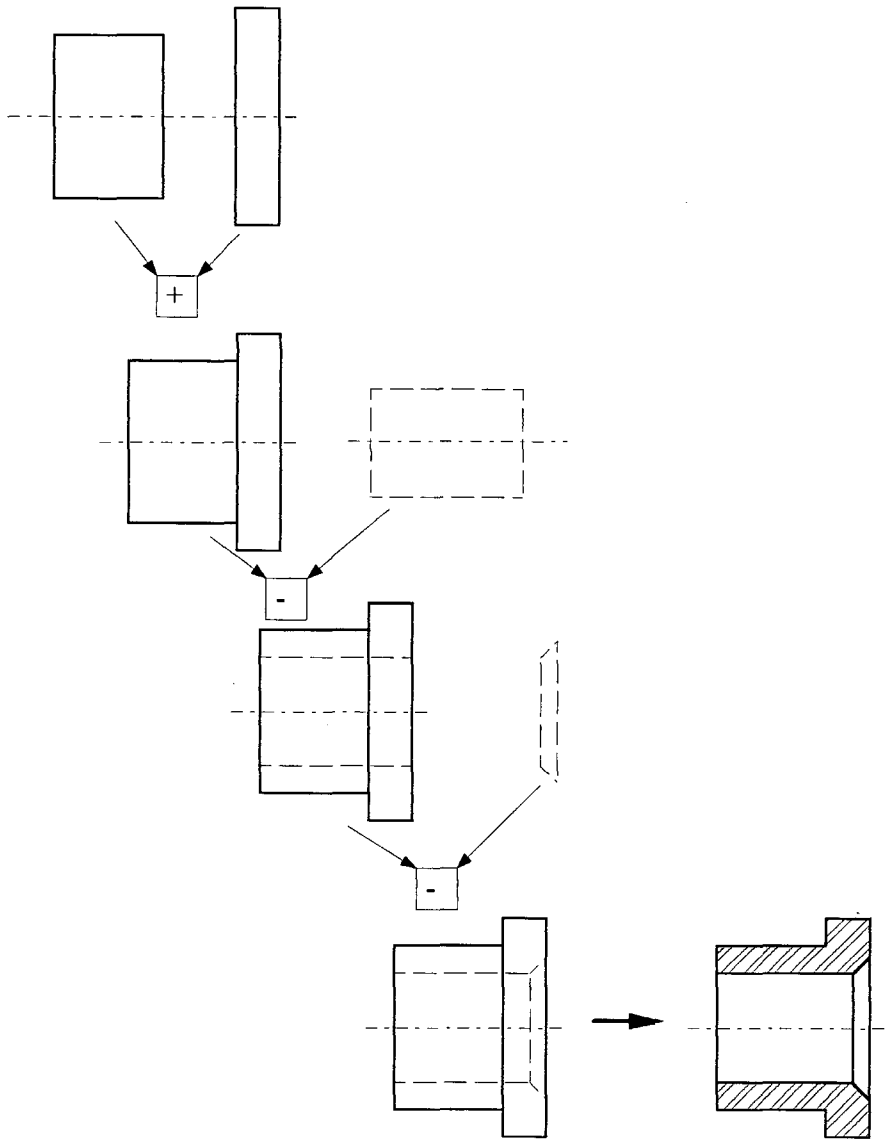


Figura 4.7 - Representação de uma *feature* de alto nível padrão bucha.

4.7.6.3 - *FEATURES* DE ALTO NÍVEL PADRÃO CONFIGURÁVEIS

Como dito anteriormente, estas *features* podem ser configuradas diretamente da interface gráfica. Desse modo, o usuário pode construí-las a partir das *features* disponíveis na biblioteca.

Durante a construção da referida *feature*, todas as regras estarão sendo checadas normalmente, como se estivesse sendo feita a construção de uma peça a partir das *features* disponíveis. Após a criação dessa nova *feature*, pode-se gravá-la em disco; assim, ela estará

disponível ao usuário quando ele dela necessitar. A gravação desta *feature* é feita num arquivo texto, o qual contém as classes descritas no formato para o sistema especialista.

Para extrair uma *feature* de alto nível padrão configurável, é utilizada uma função própria que possui a capacidade de interpretar o referido arquivo e, então, de mapeá-lo para a respectiva estrutura de dados do sistema. Deve-se observar que essa *feature* não é um simples bloco de informações, contudo as *features* que a compõem podem ser identificadas individualmente, podendo o usuário executar operações sobre as mesmas (alterar, excluir, etc.).

Uma *feature* dessa natureza somente poderá ser instanciada com os atributos que foram gravados, ou seja, com as respectivas dimensões, mas não quanto à posição. Assim, a *feature* obtida será sempre a mesma. Para efetuar modificações, é, então, necessário proceder a operações de exclusão e alteração, buscando-se obter o resultado desejado.

4.7.7 - RELAÇÕES DE DEPENDÊNCIA ENTRE AS *FEATURES*

As relações de dependência entre as *features*, como descritas no item 3.5.2, são implementadas no programa-fonte. Assim, como exemplo, tem-se que, ao se desejar inserir uma *feature* furo, o sistema verifica se na referida peça existe alguma *feature* eixo que conste da lista dessa peça. Se não há, a *feature* furo não pode ser inserida, e a condição de dependência entre uma *feature* de volume positivo (eixo) e uma de volume negativo (furo) é mantida. O mesmo se aplica às *features* modificadoras, que não podem ser inseridas sem a existência das *features* básicas, onde essas relações são garantidas pela verificação da existência de cada uma via programa-fonte.

4.7.8 - RELAÇÃO DE DEPENDÊNCIA ENTRE CONJUNTO, SUBCONJUNTO, PEÇA E *FEATURE*

A dependência referente ao fato de que uma *feature* somente pode existir se há uma peça; uma peça, se há um subconjunto, e um subconjunto, se existe um conjunto, é necessária para que a criação das informações seja feita, inicialmente, de modo organizado.

Essas relações de dependência são garantidas através da programação da interface gráfica (menu do AutoCAD), na qual somente é possível criar um subconjunto depois de o conjunto ter sido inicializado, sendo, então, liberada a função no menu para essa operação. O mesmo se aplica à criação de uma peça e ao manuseio das suas informações, que somente estarão disponíveis depois que a peça for criada. Conseqüentemente, as funções para o manuseio das *features* somente estarão ativas a partir do momento em que for criada uma peça.

4.8 - DEFINIÇÃO DAS CLASSES DAS *FEATURES*

A implementação das classes das *features* foi realizada com uma alteração dos conceitos apresentados no modelo conceitual. Isso se deve a observações feitas durante a implementação, quando, em virtude da forma da estrutura de dados que foi adotada, optou-se por criar classes genéricas em que se utiliza o atributo "tipo" para diferenciar as particularidades existentes entre *features*. Como exemplo dessa abordagem, tem-se o caso da *feature* furo cilíndrico passante e furo cilíndrico cego, as quais se diferenciam apenas pelo conceito aplicado. Desse modo, com a aplicação do atributo "tipo" (i.e., "passante" ou "cego"), elas podem ser diferenciadas e, quando da ocorrência da transformação de *features* .(item 3.7.2.2.2), apenas esse atributo é alterado para obter-se uma nova *feature*.

Essa forma também proporciona algumas vantagens no momento em que ocorre uma transformação de *features*, como apresentado nos itens 3.7.2.2.2(c) e 3.7.3.2.2. Assim, basta apenas alterar a informação do atributo "tipo" (outros atributos também podem ser alterados), para para que se tenha a transformação da *feature*, não sendo necessário destruir um objeto de uma classe e instanciar um novo objeto de outra, o que simplifica sobremaneira o trabalho de implementação.

A seguir são descritas as classes como devem ser implementadas, ressaltando-se que, neste trabalho, isso foi feito somente com as *features* eixo, furo e chanfro. Também estão representadas as outras classes que não foram implementadas, mas da forma como deverão ser feitas em função da modificação. Essa classificação segue a abordagem apresentada por Coad (1992).

FEATURE

(*features* básicas)

EIXO

EIXO_CILÍNDRICO

EIXO_CÔNICO

FURO

FURO_CILÍNDRICO

tipo - furo_cilíndrico_cego

tipo - furo_cilindrico-passante

FURO_CÔNICO

tipo - furo_cônico_cego

tipo - furo_cônico-passante

(*features* modificadoras de aresta)

QUEBRA_CANTO

CHANFRO

tipo - chanfro_externo

tipo - chanfro_interno

ARREDONDADO

JUNÇÃO

FILETE

(*features* modificadoras de superfície)

RANHURA

RANHURA_NORMAL

tipo - ranhura_externa

tipo - ranhura_interna

ROSCA

ROSCA_CILÍNDRICA

tipo - rosca_cilíndrica_externa

tipo - rosca_cilíndrica_interna

ROSCA_CÔNICA

ROSCA_CÔNICA_EXTERNA

(*features* modificadoras prismáticas)

ELEMENTO_FORMA

ELEMENTO_FORMA_AXIAL

CHAVETA

DENTADO

CANELADO

ELEMENTO_FORMA_RADIAL

LINGUETA

4.9 - DEFINIÇÃO DAS *FEATURES* BÁSICAS QUANTO À GEOMETRIA

Conforme a classificação das *features* feita no item 3.5.1, têm-se como básicas o eixo e o furo, uma dita de volume positivo e a outra de volume negativo.

4.9.1 - DEFINIÇÕES GENÉRICAS

As definições feitas neste item procuram generalizar um grupo de conceitos que serão utilizados para descrever as *features* dentro das classes.

Ponto de referência: é o ponto de onde são referenciadas as coordenadas para executar a construção gráfica da *feature*.

Ponto de inserção: é o ponto definido para inserir a nova *feature*. O ponto de inserção deverá ser obrigatoriamente sobre uma outra *feature* (com exceção da primeira *feature* eixo).

Ponto de localização: é o ponto de uma coordenada do espaço utilizado para realizar a seleção de uma *feature*, que auxilia na definição do sentido das *features* e do ponto de inserção da *feature*.

Superfície cilíndrica: representada por um par de retas paralelas ao centro de rotação da *feature*.

Superfície cônica: representada por um par de retas concorrentes ao centro de rotação da *feature*, com ângulo $0° < \alpha < 90°$.

Face: região delimitada por um plano perpendicular ao centro de rotação e à interseção da superfície cilíndrica ou cônica.

Face Inicial (FI): é denominada a coordenada "X" de uma face de uma *feature* inserida que pode ser função do seu sentido e função da *feature*, tendo por base a posição dessa face com relação à reta X=0 de um sistema de coordenadas absoluto.

Face Final (FF): é denominada a coordenada "X" de uma face de uma *feature* inserida que pode ser função do seu sentido e função da *feature*, tendo por base a posição dessa face com relação à reta X=0 de um sistema de coordenadas absoluto.

Posição: a posição da *feature* pode ser externa ou interna.

Posição externa: uma *feature* é dita externa, além de um eixo, se a *feature* é aplicada sobre um eixo (aplicado um modificador externo).

Posição interna: uma *feature* é dita interna, além de um furo, se a *feature* é aplicada sobre um furo (aplicado um modificador interno).

Volume positivo: aquelas *features* que, quando aplicadas, acrescentam material.

Volume negativo: aquelas *features* que, quando aplicadas, retiram material.

Coordenada: é o ponto de localização do centro da face de uma *feature*.

Direção: a direção da *feature* pode ser radial ou axial.

Direção axial: corresponde à direção do centro de rotação da forma geométrica em questão, neste caso perpendicular à face da *feature*.

Direção radial: corresponde à direção perpendicular ao centro de rotação da forma geométrica em questão, neste caso perpendicular à superfície da *feature*.

4.9.2 - *FEATURE* BÁSICA DE VOLUME POSITIVO - EIXO CILÍNDRICO

Um eixo é resultante da existência de uma superfície cilíndrica, a qual produz a superfície do eixo, e de dois planos paralelos afastados entre si de uma distância "L" e perpendiculares à superfície cilíndrica (Figura 4.8).

A superfície de um cilindro é representada por inúmeras geratrizes, sendo, com base no sistema de projeção ortogonal, representada pelas duas geratrizes mais externas e opostas com relação ao eixo de revolução.

A face do cilindro é o círculo resultante da intersecção da superfície cilíndrica e de um plano perpendicular ao centro de rotação, sendo representada por retas de acordo com o sistema de projeção ortogonal.

A distância entre as duas geratrizes externas é denominada de *diâmetro*. Um eixo é considerado de volume positivo, sendo, nesse caso, admitida somente a direção axial.

ATRIBUTOS :

Nome (s):

Diâmetro (d) :

Comprimento (d) :

Face Inicial (d) : FI

Face Final (d) :FF

Direção (s) : axial

Sentido (i): à direita (+1), à esquerda (-1)

Posição (s) : externo

Volume (i) : positivo (+1)

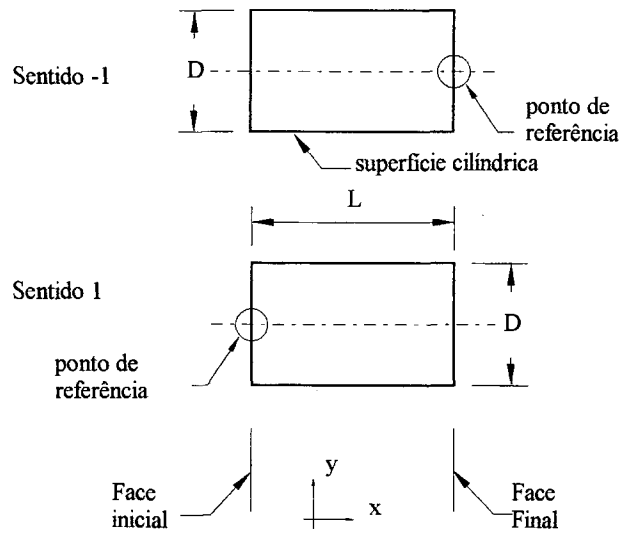


Figura 4.8 - Elementos da *feature* eixo cilíndrico.

A representação de *features* eixo cilíndrico é mostrada na Figura 4.9. Deve-se observar que a face inicial (FI) é sempre considerada a face mais próxima do eixo "Y", e a mais afastada a face final (FF), independentemente do sentido de construção.

Na Figura 4.9 (b), tem-se o caso em que a peça está localizada no lado negativo do eixo dos "X". Neste trabalho, somente é considerado o lado positivo do sistema de coordenadas, ou seja, "X" e "Y" positivos.

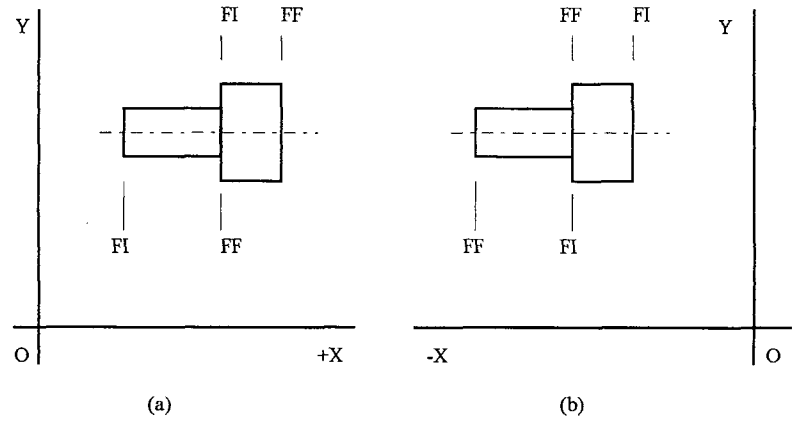


Figura 4.9 - Disposição das *features* eixo no sistema.

4.9.3 - *FEATURE* BÁSICA DE VOLUME NEGATIVO - FURO CILÍNDRICO

Uma *feature* furo cilíndrico é considerada um volume negativo, sendo composta por uma superfície cilíndrica interna e limitada por dois planos paralelos, ambos perpendiculares ao eixo de revolução do cilindro.

A superfície resultante da intersecção da superfície interna cilíndrica com os planos paralelos resulta nas faces do furo cilíndrico. As geratrizes externas da superfície cilíndrica resultarão na representação da superfície interna.

A face inicial (FI) pode ser coincidente com uma das faces de um eixo (FI ou FF), com uma superfície (no caso de direção radial) ou na face final (FF) ou um outro furo. A face final (FF) estará em qualquer posição internamente a um eixo, podendo até ser coincidente com uma das faces de um eixo ou superfície (Figura 4.10).

Numa *feature* furo, sempre é dada como referência a face inicial (FI). O sentido é dado em termos de à direita ou à esquerda para furos axiais. As *features* furos cilíndricos podem ser de direção axial ou radial.

ATRIBUTOS:

Nome (s):

Tipo (s): cego, passante

Diâmetro (d):

Profundidade (d):

Face Inicial (d): FI

Face Final (d):FF

Direção (s): axial

Sentido (i): à direita (+1), à esquerda (-1)

Posição (s): interno

Volume (i): positivo (-1)

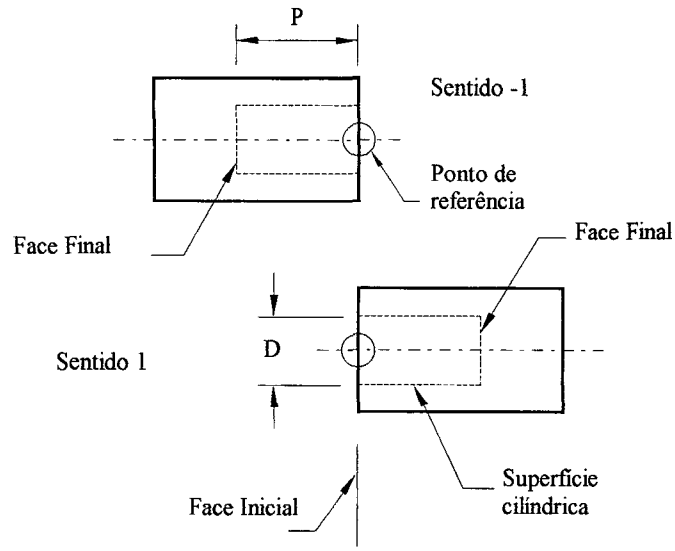


Figura 4.10 - Elementos de uma *feature* furo cilíndrico.

4.9.4 - *FEATURE* MODIFICADORA DE ARESTA DE VOLUME NEGATIVO - CHANFRO

Um chanfro por definição corresponde à retirada de uma aresta ou, em linguagem mais comum, a quebrar um canto. Sendo o chanfro considerado pela classificação como uma *feature* modificadora, portanto, que não existe sozinha, necessita de uma *feature* básica que representa volume positivo ou negativo para que possa existir (Figuras 4.11 e 4.12).

Um chanfro é dito externo quando aplicado como modificador de uma *feature* externa; é interno quando aplicado como modificador de uma *feature* interna. Um chanfro pode ter direção axial ou radial, dependendo da *feature* que produzirá a modificação.

O ponto de referência de uma *feature* chanfro está localizado sobre uma face que será coincidente com uma das faces da *feature* na qual atuará como elemento modificador.

ATRIBUTOS:

Nome (s):

Comprimento (d): C

Face Inicial (d): FI

Face Final (d):FF

Direção (s): axial

Sentido (i): à direita (+1), à esquerda (-1)

Posição (s): interno / externo

Volume (i): negativo (-1)

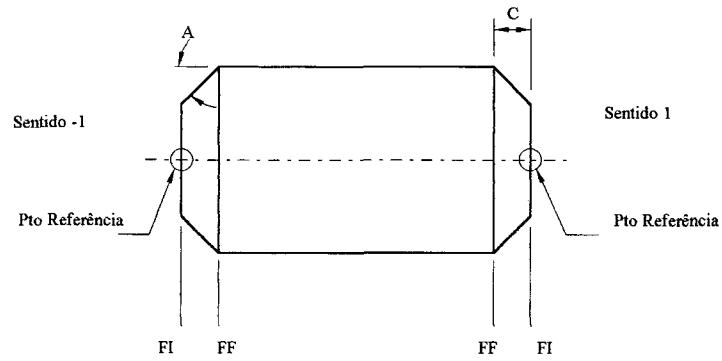


Figura 411 - Elementos de uma *feature* chanfro externo.

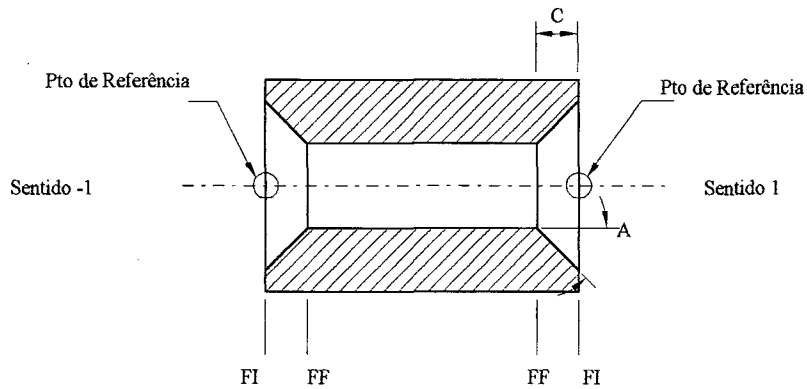


Figura 4.12 - Elementos de uma *feature* chanfro interno.

4.9.5 - *FEATURE* MODIFICADORA DE SUPERFÍCIE DE VOLUME NEGATIVO - RANHURA

Da mesma forma que um chanfro, a ranhura é considerada pela classificação como uma *feature* modificadora, portanto, que não existe sozinha necessitando que uma outra *feature* básica que representa volume positivo ou negativo esteja presente para que possa existir (Figura 4.13).

Uma ranhura é dita externa quando aplicada como modificador de uma *feature* externa; é interna quando aplicada como modificador de uma *feature* interna. Uma ranhura pode ter direção axial ou radial, dependendo da *feature* sobre a qual produzirá a modificação. O ponto de referência de uma *feature* ranhura está localizado sobre uma face que limita a largura da ranhura.

ATRIBUTOS:

Nome (s):
Largura (d): L
Diâmetro Interno (d): DI
Diâmetro Externo (d): DE
Face Inicial (d): FI
Face Final (d):FF
Direção (s): axial
Sentido (i): à direita (+1), à esquerda (-1)
Posição (s): interno / externo
Volume (i): negativo (-1)

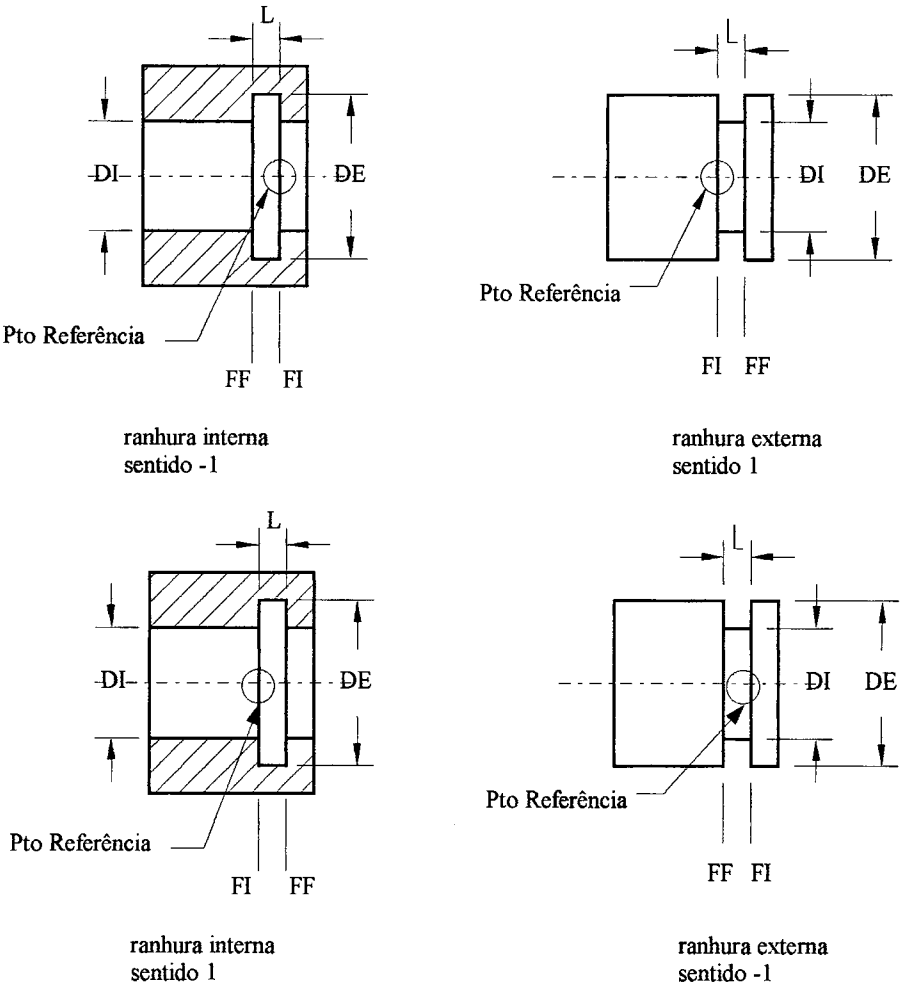


Figura 4.13 - Elementos de uma *feature* ranhura interna e externa.

Numa *feature* ranhura, o ponto de referência está sempre mais próximo de uma outra *feature* com a qual vai realizar contato (montagem), bem como a face inicial é coincidente com o ponto de referência.

4.10 - DEFINIÇÕES GEOMÉTRICAS PARA A PEÇA

As informações de geometria da peça auxiliam para uma melhor interpretação das informações e de modo rápido e eficiente (Figura 4.14).

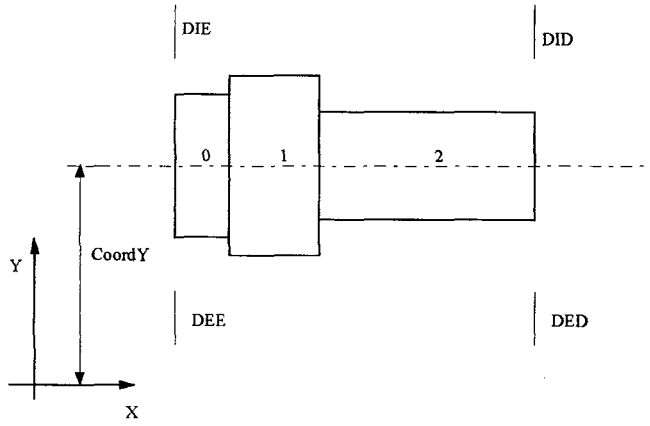


Figura 4.14 - Elementos externos de uma peça.

CoordY: coordenada "Y" da posição do centro de rotação com relação ao sistema de referência. Esta coordenada será sempre a mesma para uma peça e será aquela introduzida pela primeira *feature* (Figuras 4.14 e 4.15).

DED: dimensão Externa Direita, posição da coordenada "X" máxima da peça do lado direito (Figura 4.14).

DEE: dimensão Externa Esquerda, posição da coordenada "X" mínima da peça do lado esquerdo (Figura 4.14).

As dimensões externas servirão de controle do comprimento da peça e de base para a análise da inserção de novas *features* eixo ou furos axiais.

DID: dimensão Interna Direita, posição de coordenada máxima "X" da peça do lado direito, para consideração de elementos internos (Figura 4.15).

DIE: dimensão Interna Esquerda, posição de coordenada mínima "X" da peça do lado esquerda, para consideração de elementos internos (Figura 4.15).

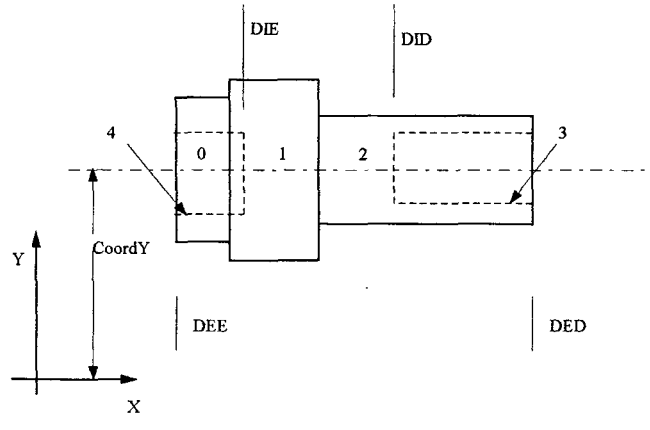


Figura 4.15 - Elementos internos de uma peça.

4.11 - MODELO GEOMÉTRICO DE REPRESENTAÇÃO DE UMA PEÇA POR *FEATURES*

Dos conceitos aplicados para projeto por *features*, pode ser obtido um modelo híbrido que utiliza tanto a síntese dos elementos volumétricos como a geometria destrutiva, os quais são aplicados em várias fases de acordo com as definições do sistema. Para isso, é utilizado o conceito de modificadores (LIMA, 1994), existindo no sistema as chamadas *features* básicas (ver item 3.5.1) sobre as quais são aplicadas determinadas *features* que produzem modificações sobre as primeiras. Também é possível aplicar esse conceito de forma que ocorra a aplicação de uma *feature* modificadora sobre outra.

O processo de modelagem de informações de geometria inicia com a criação de uma *feature* de volume positivo, que, no caso, vem a ser um eixo. A união das *features* básicas de volume positivo é feita pela síntese de elementos volumétricos, e a obtenção da aplicação das *features* modificadoras se dá através da geometria destrutiva. Isso pode ser observado quando da obtenção de um eixo escalonado, obtido pela união de dois elementos volumétricos que correspondem à *feature* eixo (Figura 4.16).

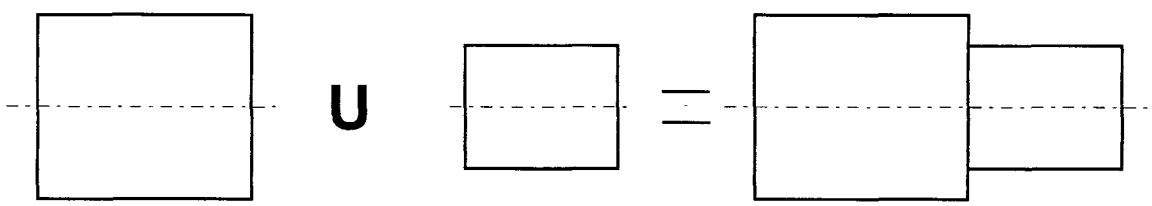


Figura 4.16 - Síntese de elementos volumétricos.

A união de dois eixos é feita sempre face a face, levando em conta os centros de rotação das *features*, que devem estar alinhados. Nesta proposta, não está incluída a junção de eixos ortogonais, nem de eixos excêntricos. Eixos podem ser adicionados tanto à esquerda como à direita da peça; uma única *feature* eixo pressupõe a existência de uma peça.

A outra *feature* básica a ser inserida corresponde à *feature* furo, a qual é de volume negativo. Com a criação de uma *feature* furo, a estrutura para a existência de *features* internas é inicializada. Assim, um furo pode ser adicionado a outro, por exemplo, um furo escalonado, subtraindo-se o volume do eixo (Figura 4.17).

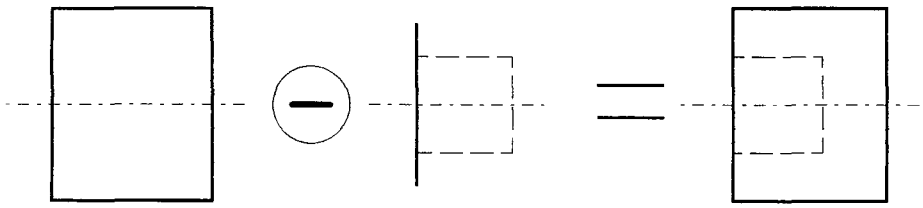
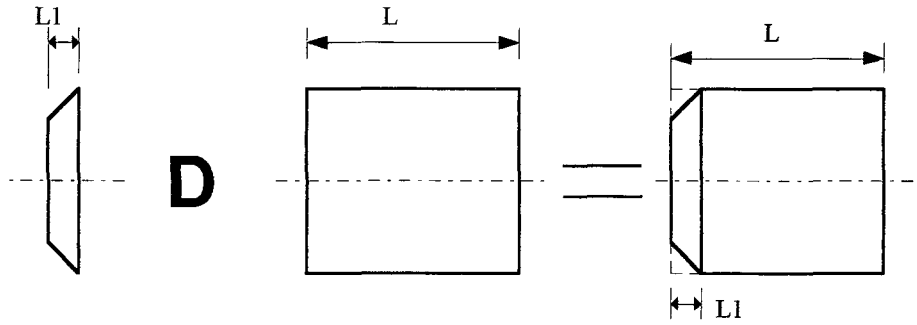


Figura 4.17 - Inserção de uma *feature* furo.

A introdução de um chanfro corresponde à inserção de uma *feature* modificadora de aresta, que, através da geometria destrutiva, retira o material de um eixo para essa representação. O resultado é a aplicação de dois conceitos, resultando numa aplicação híbrida. Desse modo, a inserção de um chanfro faz com que o eixo mantenha o comprimento inicial "L"; contudo graficamente, a representação da parte cilíndrica é diminuída de um comprimento "L1", espaço ocupado pelo chanfro (Figura 4.18).

A aplicação do conceito descrito serve para todas as *features* modificadoras de aresta, cuja inserção tanto pode ser feita sobre eixos como sobre furos.



onde "D" é o operador que aplica a geometria destrutiva entre as *features* consideradas.

Figura 4.18 - Geometria destrutiva para inserção de um chanfro.

A inserção de uma *feature* modificadora de superfície, como uma ranhura, produz uma modificação na superfície de um eixo, fazendo com que, na representação gráfica, o eixo seja dividido em dois e seja interpretado como um único eixo em termos de informação (Figura 4.19).

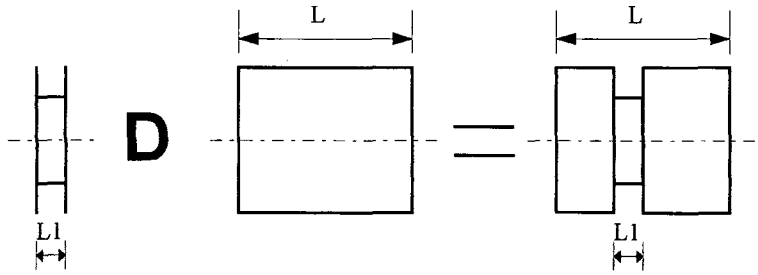
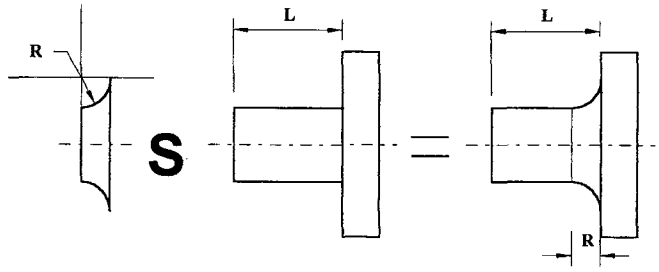


Figura 4.19 - Inserção de uma *feature* ranhura externa.

A inserção de uma *feature* modificadora de aresta de volume positivo corresponde à utilização de um raio de concordância, o qual produz uma modificação no eixo a ser aplicado, provocando uma redução no comprimento da parte cilíndrica do eixo na representação gráfica e mantendo internamente a informação de um eixo como um todo na estrutura de dados (Figura 4.20).



onde "S" é o operador que aplica a síntese de elementos volumétricos entre as "features" consideradas.

Figura 4.20 - Inserção de uma *feature* modificadora filete.

Desse modo, fica clara a ligação que deve haver entre uma *feature* básica e uma modificadora. No domínio do presente trabalho, as básicas definidas são o eixo e o furo, e as modificadoras são todas aquelas que proporcionam uma alteração a essas duas. Assim, têm-se o chanfro, a concordância, o arredondamento, a ranhura, rasgos de chaveta, canais, etc.

4.12 - REPRESENTAÇÃO DA ESTRUTURA DE DADOS

Uma das grandes preocupações que se tem observado em vários trabalhos (Schulz, 1993 - Schulz, 1994) é com respeito à representação das *features* e de seus atributos. Nos casos em que é abordado o problema de estrutura de dados, utiliza-se a estrutura em árvore binária como

utilizada na estrutura dos modeladores "CSG" e "B-rep". Porém, é de grande importância a estrutura de dados, que é utilizada para a representação no sistema computacional. Isso, muitas vezes, é deixado para o analista de sistemas decidir, ou seja, escolher qual tipo de estrutura quer utilizar.

Neste trabalho, a estrutura de dados é de fundamental importância para que o modelo conceitual possua uma representação computacional próxima à representação física. Conhecendo-se a estrutura de dados, é possível verificar como os dados estão dispostos no sistema e como podem ser manuseados, criando uma visão de interpretação do sistema computacional.

A importância da estrutura de dados é decorrente da forma como se imagina que eles estejam associados entre si, o que representa uma forma de apresentação e armazenamento, em outras palavras, saber a forma como os dados estão associados, o local onde estão armazenados e a forma de navegar pela estrutura para poder localizá-los e manipulá-los de forma racional.

Conforme a estrutura do produto descrita (ver item 3.4.2), observa-se que a estrutura pode ser expandida de acordo com a necessidade. A representação a ser usada computacionalmente deve possuir a flexibilidade que a estrutura conceitual exige. Assim, um produto é composto por um *conjunto;* um *conjunto* é composto por *subconjuntos*; um *subconjunto* é composto por inúmeras *peças*.

A estrutura de dados computacional a ser utilizada para representar o produto descrito é mista, e sua forma geral é de uma estrutura em árvore, onde em cada nó há uma lista e, de cada célula da lista, partem novas listas.

O *produto* é considerado único, ou seja, é representado um produto de cada vez. O *produto* possui um único *conjunto*, que, na representação da estrutura de dados, é implementado com a utilização de lista ligada, ocupando uma célula da lista denominada de *lista de conjuntos*. Essa lista é limitada a uma única célula porque o produto possui um único conjunto (Figura 4.21).

Figura 4.21 - Lista de conjuntos.

De acordo com a estrutura do produto, um *conjunto* pode ser composto por vários *subconjuntos* ou por um único, dependendo da complexidade do produto a ser representado, o

que será definido em tempo de execução do produto. Assim, de um *conjunto*, que é uma célula da lista de conjuntos, sai uma lista de *subconjuntos* (Figura 4.22).

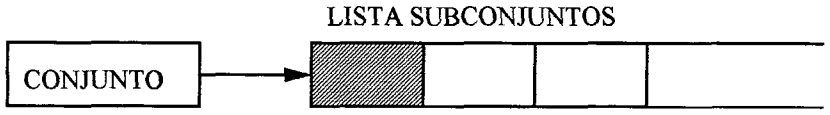


Figura 4.22- Lista de subconjuntos.

Um *subconjunto* é representado numa célula da lista de subconjuntos, sendo composto por várias *peças*; para cada *subconjunto*, uma lista de *peças* deve estar a ele ligada, resultando numa lista de peças que sai de uma célula da lista de *subconjuntos* (Figura 4.23) :

Figura 4.23- Lista de peças.

Uma *peça* é composta por várias *features* e, de acordo com a classificação das *features* apresentadas no modelo do sistema, as *features* básicas são divididas em *features* de volume positivo e de volume negativo, que podem ser chamadas de *feature* básica externa e *feature* básica interna.

A estrutura nesse ponto é subdividida de acordo com a classificação das *features*, surgindo, então, duas novas listas a partir de uma célula da lista de peças. Assim, de uma célula da lista de peças saem duas listas que são a lista de *features* externas e a de *features* internas, que correspondem às *features* básicas eixo e furo (Figura 4.24).

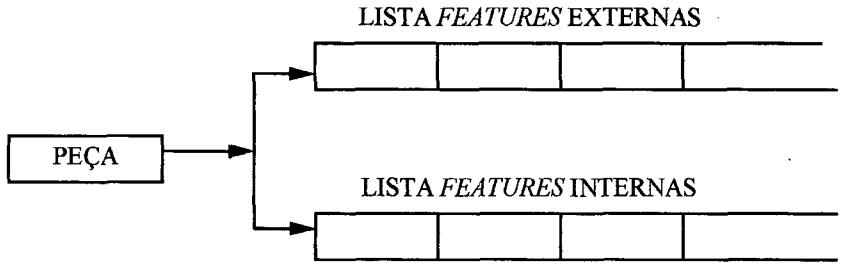


Figura 4.24- Listas de *features* externas e internas.

Às *features* internas e externas podem ser associadas às *features* modificadoras. De acordo com a definição dessas últimas, a lista foi concebida de modo que as *features* modificadoras estejam ligadas às básicas, sobre as quais produzem as modificações. Assim, as *features* modificadoras podem ser aplicadas tanto a *features* básicas internas (volume negativo) como a *features* básicas externas (volume positivo).

Da célula de uma *feature* interna ou externa podem sair tantas listas quantos forem os modificadores que possam ser associados a essas *features* básicas (Figura 4.25).

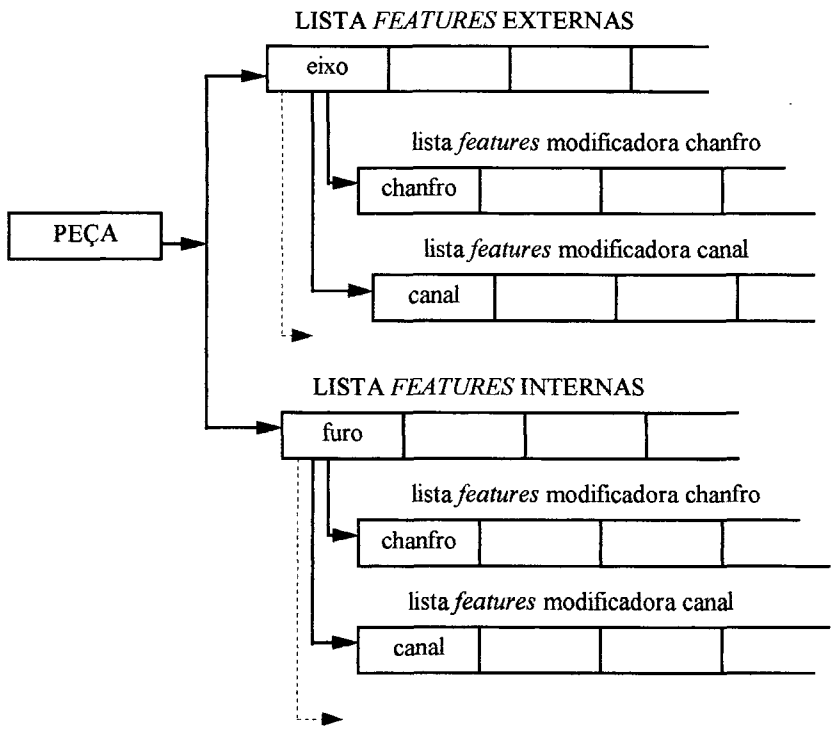


Figura 4.25 - Listas de *features* modificadoras aplicadas sobre *features* básicas.

Após a definição de cada ligação entre as subestruturas formadas pelas listas, onde se utilizam listas ligadas e listas duplamente ligadas, a representação geral da estrutura pode ser observada na Figura 4.26. Observando-a, vê-se que cada tipo de elemento é armazenado numa lista independente. Assim, as informações de conjunto estão armazenadas numa célula da lista de conjuntos, a qual será de tamanho um, ou seja, somente uma célula será ocupada e um único conjunto que representa um produto.

Esse conjunto possui uma série de subconjuntos que são representados pelas células ocupadas na respectiva lista. Cada célula, além dos próprios atributos de subconjunto, possui uma lista de peças que compõem o respectivo subconjunto. Cada peça é representada por

elementos externos e internos que estão na estrutura em forma de duas listas independentes, as quais estão contidas em cada célula da lista de peças. Além disso, cada elemento externo ou interno é capaz de ter uma série de elementos modificadores, que, no presente trabalho, são dois, mas que podem ser tantos quantos forem necessários, originando-se de cada célula da lista de elementos externos e internos.

Também desse modo se cria uma hierarquia de existência já descrita no item 3.5.2, obrigando-se à criação das *features* básicas *a priori* para que possam existir, em seguida, as modificadoras. Para cada tipo de modificador, deve existir uma lista disponível para o seu armazenamento. Assim, cada célula criada numa lista inicializa automaticamente as listas que partem dela, cujo preenchimento poderá ser feito ou não, dependendo da necessidade da respectiva *feature* (Figura 4.26).

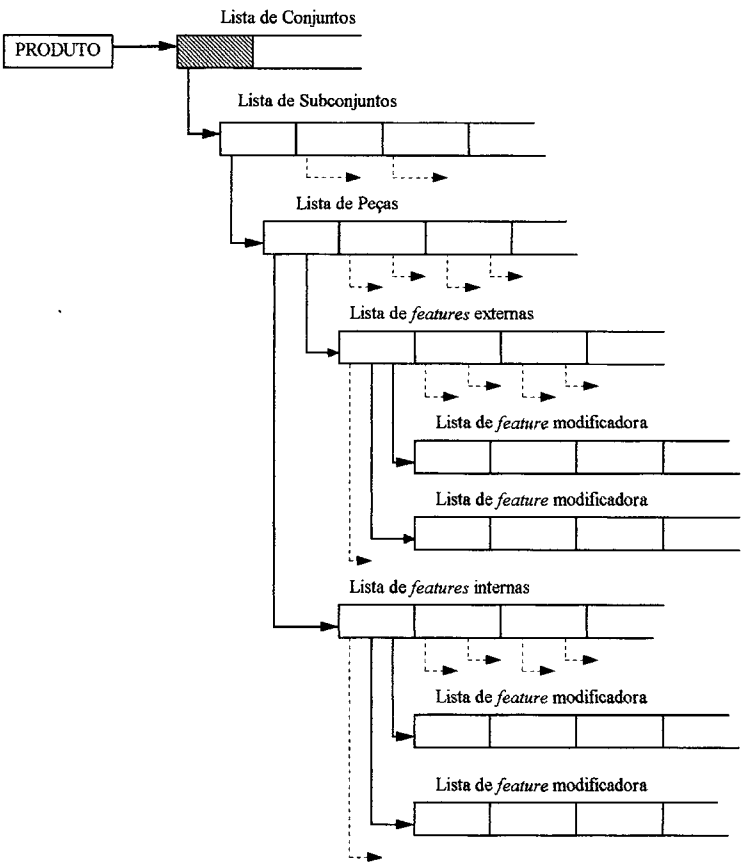


Figura 4.26- Visão geral da estrutura de dados.

Conforme definido no modelo conceitual do sistema FeatCAD-2D, as *features* modificadoras podem ser classificadas em *features* modificadoras de primeira e segunda ordem.

A representação dessas *features* em nível computacional corresponde à criação de listas que partem de uma célula de uma lista de *features* modificadoras (Figura 4.27).

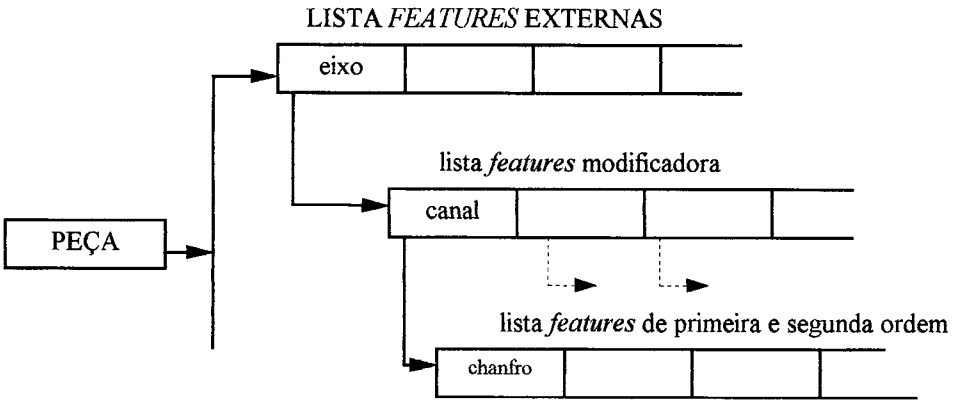


Figura 4.27- Representação das *features* modificadoras de primeira e segunda ordem.

Deve-se mencionar que as *features* modificadoras de primeira e segunda ordem não foram implementadas neste trabalho.

4.12.1 - NAVEGANDO NA ESTRUTURA DE DADOS

Navegar na estrutura de dados significa percorrer a estrutura de modo a encontrar uma informação desejada. Para isso, é necessário que a estrutura forneça elementos de identificação e que haja ferramentas adequadas para a manipulação dessas informações.

O elemento procurado deve ser de fácil visualização e possibilitar ser identificado de forma única. Esse elemento característico pode ser um atributo gráfico, tecnológico ou um atributo próprio criado para esse fim. Logo, para cada tipo de análise, existem atributos que caracterizam de forma única o elemento procurado, devendo, então, haver a necessidade de uma análise cuidadosa para a sua identificação.

Desse modo, considerando uma estrutura de dados representando um produto, para navegar na estrutura, são necessárias informações, como, por exemplo, identificar o conjunto, o que é feito através do seu nome (um *string*), de forma que a busca se efetuará pelo nome dado ao conjunto. Como o conjunto é armazenado numa lista que possui uma única célula, a busca torna-se simples, sendo suficiente identificar a lista de conjuntos e a primeira célula que possua o atributo-nome do conjunto igual ao desejado. Encontrado o conjunto, este possui um ponteiro

(atributo de ligação entre as listas) para a lista dos subconjuntos. De posse da lista de subconjuntos, que são identificados através do nome (pode ser feito através de seu código), o usuário escolhe o subconjunto desejado. Encontrando o nome do subconjunto, o sistema verifica se esse possui uma lista de peças. Como cada peça também possui um nome, essa é identificada de forma semelhante ao conjunto e subconjunto.

Essas listas descritas são uma estrutura de dados independentes no caso do software AutoCAD, sendo construídas de forma a representar o conceito desenvolvido para o produto e suas subestruturas, com a utilização das *features*. Essas listas são invisíveis ao usuário na sua forma de representação, sendo a sua visualização obtida através da identificação das camadas (*layers*) (ver item 4.4.1), que estão associadas a cada elemento descrito anteriormente, o que possibilita, assim, identificar peça, subconjunto e conjunto.

A peça é a última camada a ser representada na estrutura do produto, o que é feito pela reunião das diversas *features* que a compõem. Uma peça pode ser identificada através do seu nome, ou através do índice da camada (*layer*) que está representada, através dos quais o sistema procura na lista que contém todas as peças daquele conjunto uma que possua o nome ou índice especificado. Identificada a célula da lista, a peça aponta para duas listas, uma de *features* externas e a outra de *features* internas. As *features* também possuem um nome de identificação (*string*), atribuído automaticamente pelo sistema durante a sua criação; além disso, cada *feature* recebe um identificador interno do banco de dados do CAD (AutoCAD), que serve de elo de ligação entre a estrutura do produto e as informações gráficas modeladas no CAD.

Desse modo, para identificar, por exemplo, um eixo que está representado graficamente no CAD, basta clicar com o *mouse* sobre as linhas que o definem que as funções internas do sistema identificam a entidade, procurando na estrutura uma *feature* representada que possua essa mesma entidade e encontrando, assim, os atributos que estão representados na estrutura de dados do sistema FeatCAD-2D para aquela *feature* procurada.

Identificando-se a *feature* externa ou interna, essa também possui apontadores (um apontador é um ponteiro, ou seja, um elemento computacional que armazena endereços de memória em vez do valor de variáveis e que serve de elo de ligação entre as diversas listas) para todas as listas de modificadores possíveis de serem associados a cada *feature* externa ou interna.

Através da estrutura descrita, torna-se possível navegar e identificar cada elemento existente de forma única. Pode-se também aplicar outras operações sobre as listas de modo a manipular as informações equivalentes às representações gráficas ocorridas na tela; assim, é

possível excluir e incluir elementos. Entretanto, para isso, é necessário que se tenham operações gráficas equivalentes que representem as operações realizadas sobre as listas.

4.12.2 - REPRESENTAÇÃO DE UM SUBCONJUNTO NA ESTRUTURA DE DADOS

Como exemplo de representação de um subconjunto na estrutura de dados, tem-se um eixo de transmissão, conforme ilustrado na Figura 4.28. A representação de um subconjunto é a primeira célula da lista da estrutura do produto a ser preenchida para organizar as informações. Dessa forma, a partir do conjunto, é inicializada uma lista de subconjuntos em cuja primeira célula está o subconjunto **eixo de transmissão**. Por sua vez, esse subconjunto possui uma lista de peças que são **mancal**, **rolamentos** e o **eixo acionamento** (Figura 4.28).

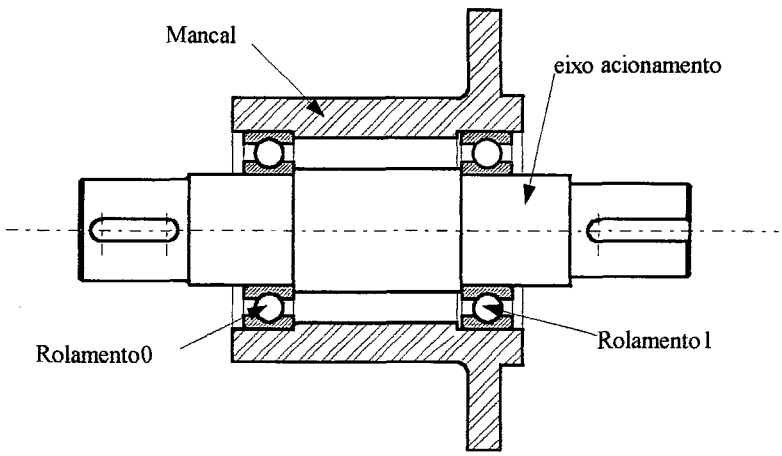


Figura 4.28 - Conjunto exemplo - Mancal.

A representação da estrutura de dados do produto está na Figura 4.29, onde é mostrado desde o conjunto até a especificação das peças.

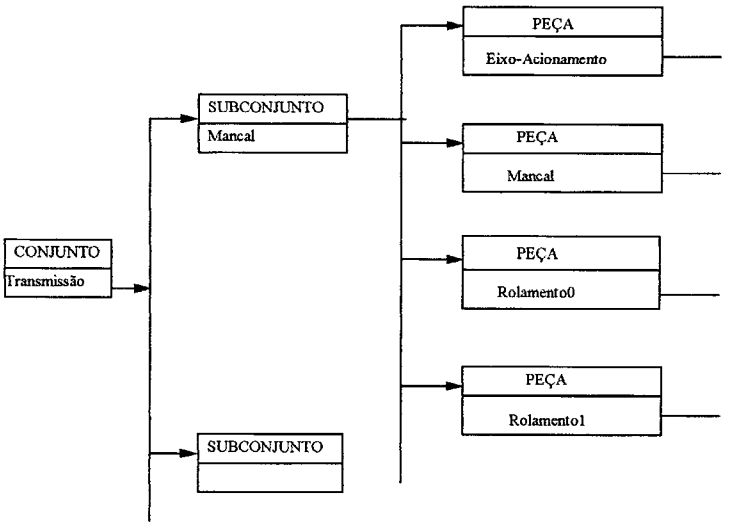


Figura 4.29- Estrutura de dados do produto.

4.12.3 - REPRESENTAÇÃO DE UMA PEÇA NA ESTRUTURA DE DADOS

A representação de um produto pode ser realizada como foi descrito na Figura 4.29, o que é feito através do seguinte exemplo (Figura 4.30).

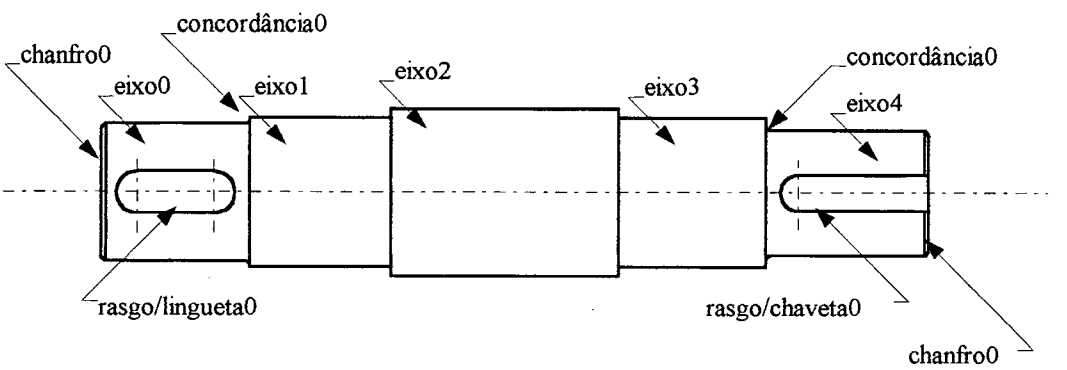


Figura 4.30 Peça-exemplo - Eixo acionamento.

Considerou-se essa peça (Figura 4.30) como pertencente a um *conjunto* e *subconjunto* determinados, criando-se a respectiva estrutura a partir da definição de peça. Dessa forma, foi criado um *produto* com o nome que passa para o *conjunto*, chamado de *Transmissão,* que possui um *subconjunto*, o qual foi designado de *Mancal*, sendo o primeiro elemento da *lista de subconjuntos*. Como cada subconjunto é composto por uma determinada quantidade de peças, foi criada a *lista de peças,* das quais uma corresponde ao *Eixo-Acionamento*.

Como cada peça é composta, basicamente, por eixos e furos, nesse caso, tem-se somente uma *lista de Eixos* que é composta por cinco células (0,1,2,3, e 4), as quais são representadas na Figura 4.31 (ver também Figura 4.30). Cada eixo pode possuir determinadas *features* modificadoras que são representadas através das listas, sendo que as listas de *features* furo e *features* modificadoras que não são aplicadas permanecem vazias. Isso pode ser observado no presente exemplo na *feature* furo, na qual quando da criação de uma peça, é criada uma lista de *features* furo que permanece vazia pela inexistência dessas *features* (Figura 4.31).

Desse modo, a partir do *eixo0* (Figura 4.31), pode-se observar que essa *feature* possui várias modificadoras a ela associadas, como chanfro, rasgo de lingueta e concordância. As *features* eixo de 1 a 3 (Figuras 4.30 e 4.31) não apresentam nenhuma *feature* modificadora. Na *feature eixo4* (Figuras 4.31), observa-se que estão a ela associadas *features* modificadoras chanfro, rasgo de chaveta e concordância.

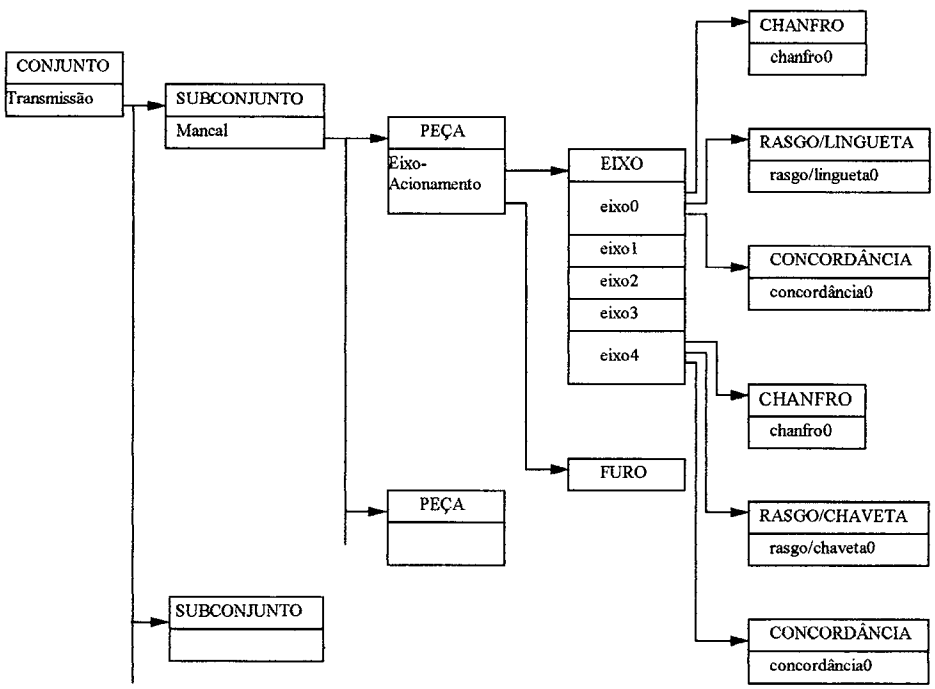


Figura 4.31- Representação de um produto na estrutura de dados.

Desse modo, a cada nova informação criada, a estrutura é expandida para acomodá-la. No caso de haver somente a alteração na informação (valor do atributo), essa, então, é substituída.

4.13 - MODELAGEM DAS *FEATURES*

Na implementação do modelador, a modelagem das *features* foi subdividida em vários submódulos, de acordo com o tipo de *feature* a ser modelada, obedecendo a suas características e à forma de entrada dos atributos.

A cada operação de inserção, exclusão e alteração de uma *feature*, a base de conhecimento referente a cada procedimento é consultada e, frente ao resultado da consulta as operações, são completadas ou abortadas. No caso de serem abortadas, o motivo é apresentado ao usuário, o que lhe permite tomar providências para atuar de modo correto. Assim, a modelagem das *features* foi subdividida conforme segue: a) modelagem das *features* elementares e b) modelagem das *features* configuráveis.

4.13.1 - MODELAGEM DAS *FEATURES* ELEMENTARES

É o submódulo onde estão contidas as funções que permitem a instanciação das *features* elementares descritas na biblioteca de *features*, à qual o usuário tem acesso direto para sua instanciação, mas não para a sua manipulação interna. É através desse submódulo que ele escolhe a *feature* a ser instanciada e seus respectivos atributos para a representação. Esse submódulo é manuseado somente internamente quando da criação de uma nova *feature* na biblioteca, o que obriga à criação de funções para a sua manipulação. Pode-se dizer que somente um *superusuário* tem acesso ao submódulo.

O submódulo permite a instanciação de *features*, como eixo cilíndrico, furo cilíndrico e *features*, denominadas de modificadoras, como chanfro, ranhura (eixo e furo cônicos, concordância, arredondamento e outros que não foram implementados neste trabalho).

4.13.1.1 - MODELAGEM DAS *FEATURES* BÁSICAS

Como proposto no modelo do sistema, há uma hierarquia entre as *features* básicas, de forma que a primeira a ser introduzida deve ser uma *feature* básica de volume positivo, que, no caso, é um eixo cilíndrico; é inserida em qualquer posição do espaço 2D, pois, inicialmente, não

há nenhum referencial a ser seguido. A partir da inserção da segunda *feature*, o sistema especialista passa a verificar as condições do produto; a partir daí, uma *feature* eixo cilíndrico será chamada de *feature* eixo , e uma *feature* furo cilíndrico, *feature* furo, ou simplesmente eixo e furo.

Assim, se já existe uma *feature* eixo, a inserção de uma outra *feature* eixo é feita através da escolha do ponto de localização (ver item 4.10.1). A posição desse ponto é analisada em relação à *feature* eixo existente: se o ponto de localização está à direita dela, o novo eixo será inserido à direita; se está à esquerda dela, a nova *feature* eixo será inserida à esquerda. Se, porém, o ponto de localização estiver situado sobre a *feature* eixo existente, o sistema não poderá inserir uma nova *feature* eixo.

A nova *feature* eixo, quando inserida, possui uma das faces coincidente com uma das faces da *feature* eixo existente, o que é condição para garantir a continuidade da peça. A inserção de eixos perpendiculares não foi implementada neste trabalho em razão dos processos de fabricação considerados (no caso de eixos perpendiculares e sendo uma peça única, deve ser executada pelo processo de fundição). A inserção de uma *feature* básica furo somente é possível quando da existência de uma *feature* eixo, o que é garantido através da pesquisa na estrutura de dados, onde é feita a verificação sobre se existe, ao menos, uma *feature* eixo na peça.

Como um furo somente pode existir onde há material, um furo será inserido através da escolha do ponto de localização, que deve estar sobre a peça (que, no caso, é composta por *features* eixos). Se o ponto de localização está mais à esquerda na peça, o furo será dito à direita e inicia na face esquerda da peça; se o ponto de localização está mais à direita na peça, será inserido um furo que inicia do lado direito e possui sentido à esquerda. Se, porém, o ponto de localização estiver fora da peça, não será possível inserir um furo na peça.

Os furos inseridos são axiais e o seu centro coincide com o centro de rotação da peça. Se a peça já possui um furo e se deseja inserir um novo furo em continuação ao primeiro, o ponto de localização deve estar localizado logo após o final do primeiro furo (local onde há material na peça).

Na presente implementação, foram considerados dois tipos de furos axiais: a) furo cilíndrico cego e b) furo cilíndrico passante. Considera-se um furo cilíndrico cego todo aquele que possui somente uma superfície de entrada, que é a face inicial; já um furo cilíndrico passante é aquele que possui tanto as faces inicial e final posicionadas de modo a permitirem a

passagem, ligando uma extremidade da peça à outra. Um furo passante pode ser inserido para efetuar a comunicação entre dois furos cegos opostos.

Quando da existência de um furo passante, torna-se impossível a inserção de um novo furo, qualquer que seja a posição. Nesse caso, as únicas operações possíveis serão excluir ou alterar a *feature* existente. A existência de uma *feature* furo na peça impede a inserção de uma *feature* eixo na extremidade da peça onde está o furo, pois isso impede o acesso à *feature* furo, tornando inconsistente a existência de uma *feature* furo sem uma entrada.

4.13.1.2 - MODELAGEM DAS *FEATURES* MODIFICADORAS

As *features* modificadoras consideradas nesta implementação foram o chanfro e a ranhura e podem ser aplicadas tanto a *features* básicas externas como a internas (volume positivo ou negativo). Como a condição de aplicação de uma *feature* modificadora é a existência de uma *feature* básica, é necessário verificar se há na estrutura de dados ao menos uma *feature* eixo.

Para a inserção de uma *feature* modificadora, é necessário que seja identificado em que tipo de *feature* básica se deseja inseri-la. Tal identificação é realizada de modo automático, determinando-se os procedimentos para a referida inserção em função da *feature* básica.

Assim, uma *feature* modificadora chanfro aplicada a um eixo passa a ser denominada de *feature* chanfro externo, caracterizando o tipo de *feature* básica à qual foi aplicada. Para inserir essa *feature* chanfro, uma função específica é solicitada, na qual o usuário informa os parâmetros, que são: a) o comprimento do chanfro e b) o ângulo do chanfro.

A seguir, o sistema solicita o ponto de localização, o que é feito sobre uma reta que representa a *feature* eixo mais próxima possível da face em que será inserida a *feature* chanfro. O sistema, então, identifica a respectiva *feature* como sendo um eixo, passando a informação para a análise. Se a operação é validada pelo sistema especialista, o chanfro é inserido sobre o eixo conforme representação feita no item 3.8.3.

Com a utilização do sistema especialista, é possível realizar uma análise da situação, como no caso em que se deseja inserir um chanfro num eixo onde há um eixo vizinho de diâmetro menor, mas de tal forma que a dimensão escolhida para o chanfro provoque uma interferência com o respectivo eixo (Figura 4.32). Nesse caso, o sistema especialista deverá

recusar a operação informando o fato. A inserção de uma *feature* modificadora chanfro sobre uma *feature* básica furo ocorre de modo semelhante ao descrito anteriormente.

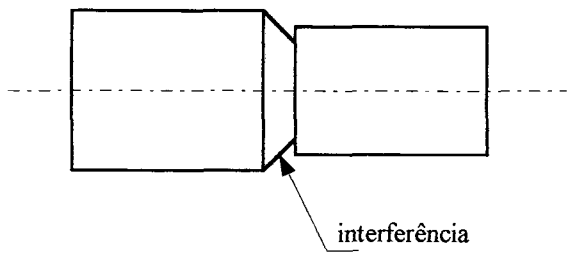


Figura 4.32- Interferência entre eixo e chanfro.

Nesta implementação, a inserção de uma *feature* modificadora circular de superfície, no caso uma ranhura, produz um fato novo no modelador, ou seja, o eixo/furo sobre o qual é aplicada passa por uma modificação em nível gráfico. Em nível de informação, o eixo/furo continua sendo um só, porém em nível gráfico passam a ser dois eixos/furos, que são interpretados como uma única *feature* para manter a consistência das informações.

Para a inserção de uma *feature* modificadora ranhura, devem ser informados pelo usuário os seguintes atributos: a) diâmetro interno da ranhura; b) largura da ranhura; c) um ponto de referência, que pode ser uma outra *feature* e d) a distância desse ponto de referência. As informações, então, são passadas para o sistema especialista, que verifica se essa *feature* não produz nenhuma interferência em qualquer outra *feature* descrita na peça. Como, exemplo, tem-se a situação em que o diâmetro interno da ranhura interfere no furo existente na peça (Figura 4.33); nesse caso, o sistema especialista rejeita a operação e informa o fato ao usuário.

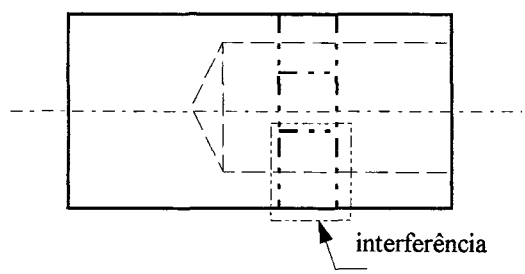


Figura 4.33- Interferência entre ranhura e furo.

4.13.2 - MODELAGEM DAS *FEATURES* COMBINADAS

A modelagem das *features* combinadas é resultante da combinação de *features* básicas e *features* modificadoras. A razão da utilização das *features* combinadas é a obtenção de *features* de nível mais elevado, de modo que guardem as características daquelas que a compõem. Assim, foi criado um outro submódulo responsável pela instanciação dessas *features*.

Para isso criaram-se novas funções de instanciação, as quais permitem que essas *features* sejam criadas levando em conta a combinação efetuada. As *features* combinadas possuem, então, uma forma geral que estão representando, podendo o usuário alterar os seus parâmetros na hora de serem instanciadas. Daí ser necessário o desenvolvimento de funções especiais que possibilitem descrever rapidamente em forma de linguagem a combinação desejada.

Para manter a consistência, as funções de validação do sistema especialista continuam ativas nessas funções, ou seja, a cada nova *feature* combinada que é inserida, o sistema especialista procede à verificação de todas as *features* envolvidas em função da situação de inserção encontrada, bem como dos parâmetros da *feature* combinada a ser inserida. A descrição desse tipo de *feature* deve ser feita ainda em nível de programação do sistema, o que não permite, ainda, ao usuário comum criar as suas *features* combinadas.

O uso das *features* combinadas se adequa muito bem a *features* do tipo furo escalonado passante, furo escareado passante, as quais serão inseridas numa *feature* eixo existente. A criação de uma *feature* combinada do tipo furo escalonado pode ser descrita em linguagem comum como a criação de um furo cego seguida de um furo passante. Em linguagem de programação pode-se ter:

- FUROCEGO (diâmetro, profundidade, sentido);
- FUROPASSANTE (diâmetro),

onde os parâmetros passados pelas fuções podem ser manipulados da forma que se deseja quando da implementação da função.

Uma das vantagens das *features* combinadas é que pode ser realizada uma combinação de *features* internas sem que seja definida uma *feature* externa, ou seja, a *feature* externa é definida quando da construção da peça. Quando da especificação de uma *feature* combinada, a condição de hierarquia das *features* deve ser obedecida, ou seja, se há um furo e sobre este furo é aplicado um chanfro, a seqüência de construção deve ser essa. É pertinente lembrar, ainda, que as *features* que compõem uma *feature* combinada, quando instanciadas, são passadas para a

estrutura de dados, obedecendo ao formato especificado e sendo decompostas nas respectivas *features* elementares.

A representação na estrutura de dados do produto de uma *feature* combinada é feita pela decomposição dessa *feature* em *features* básicas e modificadoras que a compõem. Assim, a *feature* combinada furo escalonado passante é representada na estrutura de dados como um furo cego e por um furo passante com os seus respectivos atributos. O sistema FeatCAD-2D não representa a combinação explicitamente, a qual é utilizada apenas para criar uma interface mais amigável com o usuário, existindo somente no momento da criação.

4.13.3 - MODELAGEM DAS *FEATURES* CONFIGURÁVEIS

Este submódulo é composto por funções que permitem a instanciação de *features* como os anteriores; a diferença está em que essas funções permitem que novas *features* sejam construídas a partir das *features* elementares descritas na biblioteca de *features*. Assim, é possível criar novas *features* a partir das *features* básicas e modificadoras definidas na biblioteca, as quais possuem todas as características e testes para sua validação. Também neste submódulo são definidas as *features* de alto nível (ver item 3.5.1), o que torna possível modelar parafusos, arruelas, porcas, rolamentos e outras formas que possam ser representadas dentro dos conceitos do modelador.

Este submódulo apresenta duas formas de efetuar a construção das *features* configuráveis. A primeira opção corresponde à definição das novas *features* via programação, sendo utilizado para a criação das *features* configuráveis de alto nível padronizadas; a segunda opção é através da interface gráfica, que está à disposição do usuário para a construção das *features* de alto nível padrão (ver item 3.5.1) através da manipulação gráfica das *features* elementares.

4.13.3.1 - MODELAGEM DAS *FEATURES* CONFIGURÁVEIS DE ALTO NÍVEL NORMALIZADAS

Estas *features* são construídas via programação em função da sua forma de atuação; representam componentes (peças normalizadas fornecidas por terceiros) normalizados oferecidos comercialmente, isto é, produtos que obedecem a normas internacionais. A construção dessas *features* é feita de modo semelhante às *features* combinadas, sendo descritas por funções. Assim, o componente deve ser descrito pelas *features* existentes na biblioteca, ou seja, as *features* elementares. Para a utilização das *features* de alto nível normalizadas, também é necessário uso intensivo de banco de dados, onde estão armazenadas as informações sobre os componentes de fornecimento. Assim, cada *feature,* ao ser construída, é descrita através de funções, sendo feitas consultas ao banco de dados durante a instanciação da mesma.

Isso pode ser observado através do seguinte exemplo: a escolha de um rolamento é feita pelo tipo, como, por exemplo, rolamento rígido de esferas, e pelo seu diâmetro. Quando da sua escolha, se é selecionada uma *feature* eixo sobre a qual será montado o rolamento, é fornecido o diâmetro da *feature* eixo e realizada a consulta no banco de dados à procura do rolamento com aquele diâmetro, que, no caso, é o diâmetro interno do rolamento. Caso esse seja encontrado, a inserção dessa *feature* de alto nível padronizada prossegue; em caso contrário, a operação é abortada. Se for selecionado um furo para ser colocado um rolamento, funções específicas construídas dentro da *feature* de alto nível identificam o tipo de *feature* e direcionam a pesquisa no banco de dados para que seja feita pelo diâmetro externo do rolamento

4.13.3.2 - MODELAGEM DAS *FEATURES* CONFIGURÁVEIS DE ALTO NÍVEL PADRÃO

As *features* denominadas de configuráveis de alto nível padrão possuem uma característica especial com relação à sua modelagem. Sua diferença com relação às *features* anteriormente descritas é que todas foram definidas internamente ao sistema, não sendo possível ao usuário comum (aquele que apenas utiliza o sistema via interface gráfica) criar novas *features.* Através desse submódulo, o usuário comum pode criar novas *features* que, na realidade, são peças padronizadas através da interface gráfica e recuperadas através da mesma,

mantendo todas as características das *features* elementares e sua instanciação na estrutura de dados.

A criação de uma *feature* de alto nível padrão segue os mesmos passos utilizados para criar uma peça através da utilização de *features* elementares e combinadas. Desse modo, quando da criação dessa peça, todos os testes de validação das *features* são efetuados normalmente pelo sistema especialista. O próximo passo corresponde a gravar a peça num formato que possa ser recuperado, de forma que, quando se deseja chamar essa *feature*, ela possa ser introduzida no sistema e instanciada na estrutura de dados, mantendo o conceito da ligação entre as *features* e todos os seus atributos.

Quando uma *feature* configurável de alto nível padrão é criada, seus parâmetros são especificados via interface gráfica. Logo, quando essa *feature* é gravada, ela o é com esses parâmetros, os quais serão utilizados quando da sua recuperação. Daí resulta que uma *feature* configurável de alto nível padrão não pode ser alterada quando da sua instanciação; as alterações somente podem ser feitas após a instanciação, o que é realizado através das operações de excluir, alterar e mover.

Quando da sua instanciação na estrutura de dados, a respectiva *feature* é decomposta nas suas *features* elementares (básicas e modificadoras), podendo-se, então, manipular tais informações de acordo com a necessidade.

4.14 - IMPLEMENTAÇÃO DAS FUNÇÕES DE ANÁLISE

As funções de análise se caracterizam por submódulos de aplicação específica, que utilizam os dados armazenados na estrutura do sistema, verificam a base de conhecimento para a tomada de decisão e consultam o banco de dados para obter informações a respeito de componentes, manufatura e montagem. Dessa forma, cada submódulo é implementado de modo a atuar independentemente ou em conjunto de acordo com o tipo de aplicação.

No modelo do produto (item 3.2.3), são listados vários submódulos de análise que podem ser implementados, dos quais apenas alguns o foram em função da sua importância frente ao contexto a ser pesquisado. Em seqüência, descrevem-se os submódulos de análise do conjunto, escolha de tolerâncias e cotas.

4.14.1 - ANÁLISE DO CONJUNTO

A análise do conjunto tem por objetivo o levantamento de informações a respeito da forma como as peças estão montadas, a descrição dos contatos existentes entre peças e entre *features*. Considera-se como ponto de partida para a análise do conjunto a estrutura de dados que representa o modelo do produto, com a qual, através da verificação das informações, o sistema realiza as análises necessárias.

A presente implementação considera apenas a análise de um único subconjunto, não realizando a dos subconjuntos montados uns em relação aos outros. Como definido no modelo do produto, considera-se nessa implementação um produto simples, composto por um único conjunto, que é o mesmo de um subconjunto, possuindo na sua composição peças homogêneas, resultantes de um único material (barras).

A identificação das informações a serem repassadas ao sistema especialista é feita em razão da verificação das *features* envolvidas na montagem, o que é realizado pela verificação das informações na estrutura de dados, observando-se as *features* relacionadas em cada caso.

4.14.1.1 - IDENTIFICAÇÃO DO CONTATO DIAMETRAL

Para realizar a identificação das peças e *features* que estão em contato e descrever as relações do conjunto representando uma montagem, foi desenvolvido um algoritmo procedural combinado com um sistema especialista; o algoritmo executa a procura dos possíveis pares de montagem, enquanto o sistema especialista procede à verificação da ocorrência da montagem. O algoritmo permite a identificação das peças que estão montadas e as respectivas *features* que possuem contato, partindo do princípio de que, em peças diferentes, para cada *feature* furo, pode existir uma *feature* eixo que possui o mesmo diâmetro nominal, e acontecer que ambos ocupam a mesma posição no espaço (há sobreposição de *features*). A análise que permite verificar se há uma montagem adequada ou não é realizada pelo sistema especialista através de regras que estão na base de conhecimento.

Na Figura 4.33, está representado o fluxograma do algoritmo para a análise dos contatos na montagem com relação ao diâmetro e ao posicionamento da *feature* no espaço de modo a caracterizar realmente uma montagem.

O processo de identificação de um conjunto montado inicia com a procura da primeira peça do conjunto. Se não existe uma peça (Peça(n)), o algoritmo termina; ao contrário, se há uma peça, o próximo passo é verificar se ela possui uma *feature* furo. Caso não exista uma *feature* furo, é providenciada a próxima peça do conjunto (Peça(n+1)); se há uma *feature* furo, o próximo passo é verificar se ela está à direita ou à esquerda. Existindo um furo, obtém-se a primeira peça do conjunto (Peça(m)) e verifica-se a sua existência, ou seja, se o conjunto possui a peça. Caso não possua a peça de índice (m), o algoritmo retorna ao início e obtém a peça (Peça(n+1)); se não há nenhuma outra peça, o algoritmo termina.

Se há uma peça Peça(m), verifica-se se as duas peças (n e m) são iguais, isto é, de mesmo nome. Sendo a mesma peça, obtém-se a peça (Peça(m+1)) que será diferente e também o primeiro eixo dessa peça, se ele existe; caso não haja um eixo, busca-se a peça seguinte.

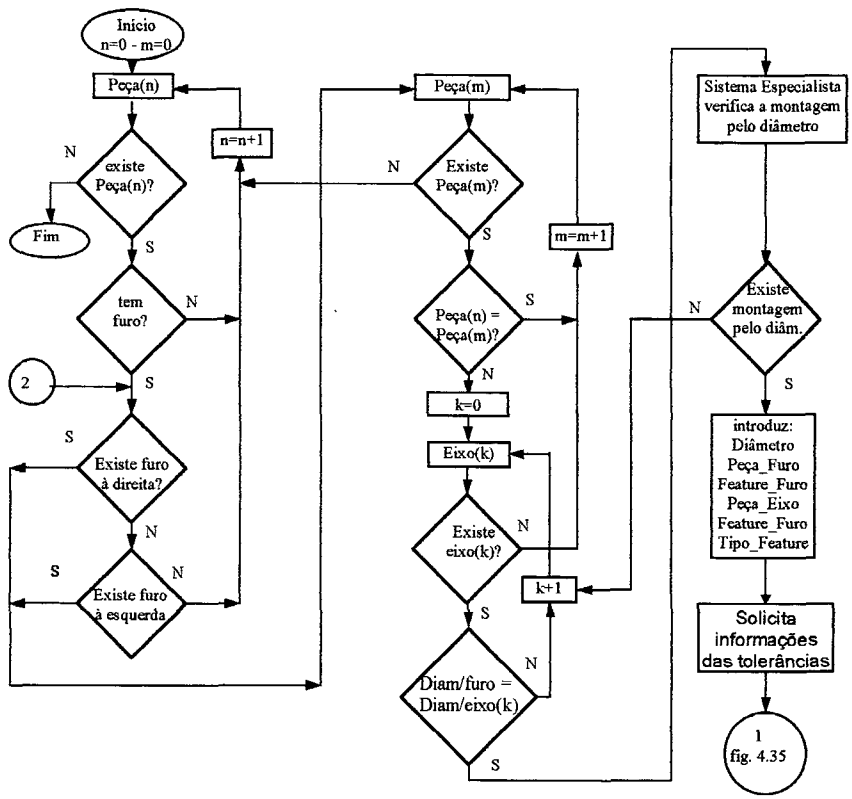


Figura 4.34- Fluxograma de análise da montagem - identificação do contato diametral.

Se existe um eixo na peça, verifica-se se os diâmetros nominais do furo e do eixo são iguais; sendo diferentes, busca-se o eixo seguinte e repete-se a operação até que seja encontrado um eixo de mesmo diâmetro nominal do furo. Se tal não for encontrado, passa-se à peça seguinte.

Se há um eixo e um furo de diâmetros nominais iguais, as informações sobre eles são passadas para o sistema especialista, o qual, a partir dessas informações, verifica a ocorrência da montagem ou não. Se, pelas condições, não ocorre uma montagem, procura-se pelo próximo eixo da peça ou, se for o caso, a peça seguinte. Se a resposta do sistema especialista confirma a montagem, as informações a respeito da montagem pelo contato diametral são armazenadas na estrutura de dados (descrição da dimensão do diâmetro e das *features* e peças envolvidas, ver no item 3.8).

Como uma montagem é uma característica de um conjunto ou subconjunto, essas informações são seus atributos. No caso da presente implementação, somente se usa um subconjunto para análise; logo, as respectivas informações são atributos desse subconjunto. Tais dados são armazenados numa estrutura em lista, onde estão representados objetos da classe Ligações, na qual estão descritos os atributos que representam os contatos que caracterizam a montagem. Esses atributos estão descritos no item 3.8.2.

4.14.1.2 - IDENTIFICAÇÃO DO CONTATO AXIAL

Se o par de acoplamento (contato diametral) furo/eixo é aceito (saída 1 do fluxograma da Figura 4.34), o próximo passo corresponde à verificação dos contatos axiais da peça (Figura 4.35) que a *feature* furo possui com relação às outras peças do conjunto e do contato axial entre *features*. Essas informações também são anexadas aos objetos descritos da classe Ligações. Tendo sido identificadas as peças e as *features* envolvidas na descrição do contato diametral, a identificação do contato axial toma como ponto de partida a Peça_Furo e a *Feature*_Furo.

De posse dessas informações, o conjunto é pesquisado obtendo-se as peças do conjunto, diferentes da Peça_Furo. Procura-se um eixo da peça do conjunto que possua a face inicial ou a face final coincidentes com a face inicial da Feature_Furo, que foi definida como possuindo contato diametral. Se existe uma *feature* eixo nessas condições, são enviadas as informações para o sistema especialista, que procede à análise e dá um parecer como retorno. Se existem condições que caracterizam, na montagem, o contato axial (ver item 3.8), as informações de sua caracterização são registradas na estrutura de dados como atributos de subconjunto, na classe Ligações; se, porém, não ocorre a condição, é procurado o próximo eixo da peça escolhida para

análise ou, se for o caso, a peça seguinte. No final, o sistema retorna à localização 2, no fluxograma da Figura 4.34, e continua a procura na montagem.

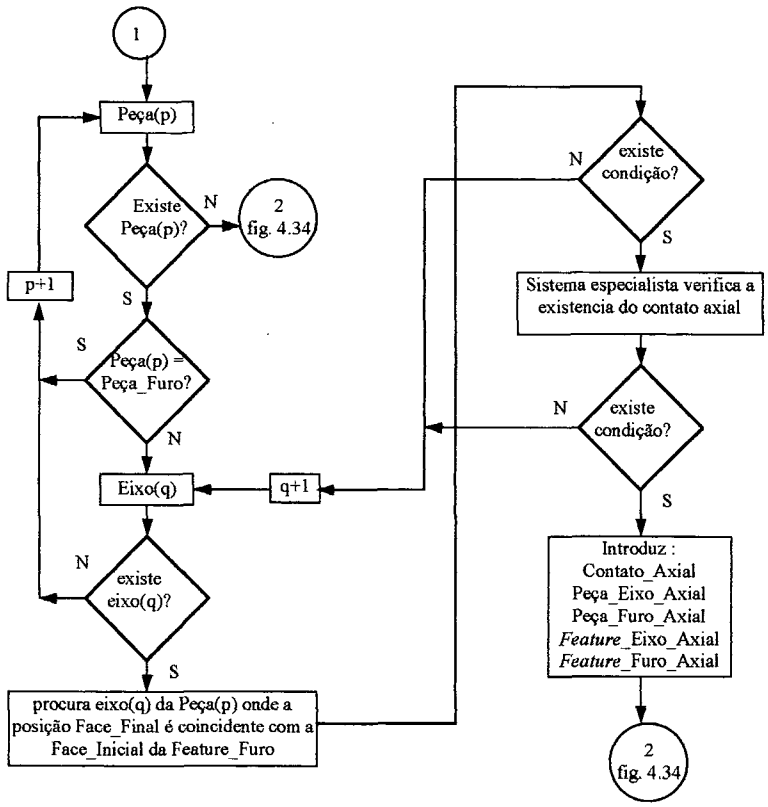


Figura 4.35- Fluxograma de análise da montagem - identificação contato axial.

4.14.2 - ESCOLHA DAS TOLERÂNCIAS

A escolha das tolerâncias toma como base a interatividade entre o sistema e o usuário, de forma que o sistema detecta as dimensões a serem controladas, e o usuário, com o seu conhecimento e as informações disponíveis no banco de dados do sistema, complementa-as escolhendo as tolerâncias mais adequadas à sua aplicação. A escolha das tolerância também pode ser feita manualmente, ou seja, o usuário seleciona a *feature* a ser dimensionada e as tolerâncias através do banco de dados.

4.14.2.1 - TOLERÂNCIAS DIAMETRAIS

A identificação das dimensões diametrais a serem controladas é feita a partir das informações da montagem do produto. Como as cotas a serem toleradas são as que exigem um grau maior de controle e somente essas dimensões possuem uma aplicação específica numa montagem, são as dimensões dessas *features* que proporcionam a montagem. Assim, têm-se as cotas diametrais dos eixos/furos que são facilmente identificadas pelo sistema.

Como, na descrição das relações de montagem têm-se as peças que estão montadas pelo diâmetro, as *features* envolvidas (eixo/furo) e a respectiva cota do diâmetro, o sistema identifica-as e informa ao usuário; ao mesmo tempo, apresenta a interface com o banco de dados, que possibilita a escolha da tolerância para o acoplamento em questão. Essa escolha baseia-se nas relações entre precisão, ajuste e tipos de aplicações características conhecidas através da literatura ou da experiência do usuário, que podem ser cadastradas no banco de dados (ver item 4.16.2).

Essas informações são relacionadas de tal forma que permitem pesquisa no banco de dados de duas formas: i) a partir da precisão de montagem (extrapreciso, preciso, média precisão e ordinária) e do tipo de ajuste (Folgado Rotativo Livre, Prensado, etc.), identificando onde esses pares podem ser aplicados (tipos de equipamentos), bem como os pares de tolerâncias para o eixo e para o furo, juntamente com o ajuste e a qualidade de trabalho (H7/f7); ii) escolhendo uma aplicação que é semelhante ao problema do usuário, e o banco de dados informando os pares de ajuste (ajuste e qualidade de trabalho) aplicada ao exemplo.

Feita a escolha do par de ajuste do acoplamento - e lembrando que a cota a ser controlada é de conhecimento do sistema - ao retornar ao sistema CAD, é realizada automaticamente uma consulta ao banco de dados que possui as tabelas ISO de tolerâncias; com o valor do diâmetro e dos ajustes e qualidade de trabalho para o eixo e o furo, são obtidos os afastamentos inferior e superior para cada dimensão de cada *feature*. Esses dados são, então, colocados na estrutura de dados do produto do sistema FeatCAD-2D, estando cada informação relacionada com a *feature* em questão.

4.14.2.2 - TOLERÂNCIAS LONGITUDINAIS

A especificação automática das tolerâncias das cotas longitudinais de uma peça rotacional é mais complexa do que as tolerâncias para as cotas diametrais. Nesse caso, as tolerâncias longitudinais estão relacionadas com as respectivas cotas, que, por sua, vez são resultantes da montagem que está representada. Desse modo, é necessário, primeiramente, definir as cotas longitudinais e quais as que devem ser controladas para, depois, proceder à inserção das mesmas, Tal definição é feita no item 4.14.3. Após a definição das cotas longitudinais, é possível que o sistema identifique automaticamente as cotas a serem controladas, o que é realizado com a ajuda do sistema especialista, que, então, apresenta ao usuário opções de escolha da precisão (extrapreciso, preciso,etc.) da referida cota.

A identificação das cotas a serem controladas é feita a partir das superfícies de referência identificadas na montagem (ver item 4.10). As cotas que não recebem tolerâncias específicas devem receber uma tolerância de qualidade inferior a ser indicada pelo usuário, isso para as cotas longitudinais e para as diametrais.

4.14.3 - DEFINIÇÕES DA COTAGEM LONGITUDINAL

A cotagem longitudinal de uma peça refere-se às cotas inter-*features*, ou seja, não se refere especificamente à dimensão de uma *feature* (pode haver uma peça que tenha uma única *feature*); logo, não pode ser atributo de uma *feature* específica. Nesse caso, as cotas longitudinais são atributos da peça, pois a cotagem longitudinal (como também o diametral) é resultante da funcionalidade da peça no conjunto.

As cotas longitudinais são definidas como um objeto que possui vários atributos para que possam ser representadas adequadamente no contexto. Assim, os atributos de uma cota longitudinal são (Figura 4.36):

PtoI: coordenada inicial da cota a ser representada, que é a coordenada de referência;

PtoF: coordenada final da cota a ser representada;

PtoMédio: é a coordenada resultante da média de dois pontos de referência mais próximos da cota em análise. Quando há apenas uma referência na peça, o ponto médio é a

própria referência, ou quando à direita ou à esquerda de uma referência há elementos a serem cotados;

CotaNominal: valor da cota nominal a ser representada e que servirá de base para as tolerâncias;

Afs: afastamento superior da CotaNominal;

Afi: afastamento inferior da CotaNominal;

Sentido: sentido em que a cota é representada, sempre partindo do PtoI e terminando no PtoF. Representa o sentido do vetor cota, que pode ser negativo se o deslocamento for à esquerda, e positivo se o deslocamento for à direita.

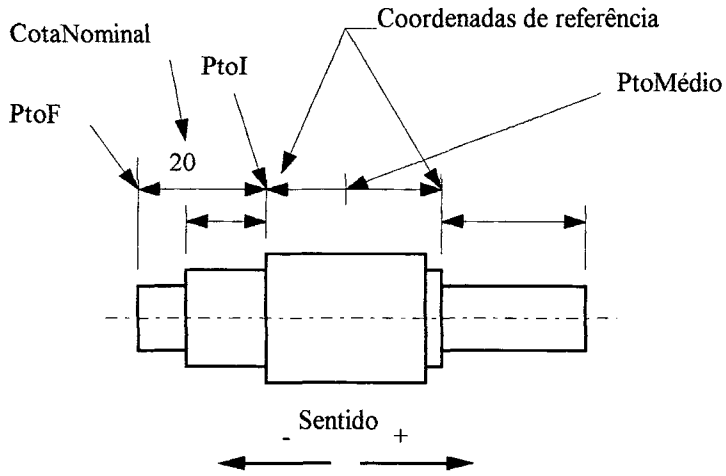


Figura 4.36 - Definição das cotas longitudinais.

Com as informações a respeito da distribuição das cotas na peça (cotagem funcional), é possível executar análises das cotas para a fabricação, o que pode ser feito através de um sistema CAPP, o qual dará uma resposta quanto à precisão exigida, se é possível ser obtida devido ao acúmulo das tolerâncias (análise de cadeia de tolerâncias).

Para que a cotagem longitudinal possa ser executada, é necessário que o sistema tenha conhecimento das referências de montagem de cada peça, o que já foi definido na análise do conjunto. Desse modo, desenvolve-se a seguir um algoritmo que identifica as cotas longitudinais a serem controladas na peça.

4.14.3.1 - ALGORITMO PARA IDENTIFICAÇÃO DAS COTAS LONGITUDINAIS

Quando da aplicação do algoritmo que define os contatos existentes entre as peças em um conjunto, os contatos axiais identificados passam a ser a referência para a cotagem de cada peça (ver item 3.9.1.2.1), procedimento que se baseia no conceito de cotagem funcional. Essa análise é realizada por um algoritmo procedural que localiza as referências de montagem e as *features* a serem cotadas e, com a ajuda do sistema especialista, realiza a análise final quanto à identificação das cotas a serem utilizadas e aquelas que devem receber uma tolerância específica na cotagem (Figura 4.37).

A cotagem longitudinal é aplicada à peça que está ativa (*layer* ligada); assim, todas as atividades estarão direcionadas a essa peça, utilizando-se a parte da estrutura de dados correspondente à mesma. O algoritmo combinado com o sistema especialista funciona como descrito a seguir e representado na Figura 4.37. O algoritmo obtém a primeira referência de montagem da peça (superfície de contato axial da peça) mais próxima da origem do sistema de coordenadas absoluto e procura uma *feature* eixo que esteja à esquerda dessa referência (CoordF). Se existe essa *feature* eixo, é utilizada a face inicial desta *feature* (CoordI) como o PtoF dessa cota, dados que são colocados na estrutura do produto.

O algoritmo, então, verifica a existência de um outro eixo que esteja à esquerda da referência e do eixo identificado anteriormente. Se houver, uma nova cota é criada a partir da referência até a face inicial deste eixo; não sendo encontrado um eixo nessas condições, a pesquisa é, então, realizada à direita dessa referência, de modo semelhante. Se o eixo procurado nessa posição existe, as informações são passadas para o sistema especialista, que realiza a análise dos dados e emite um parecer quanto à existência ou não dessa cota (ver item 4.15.4).

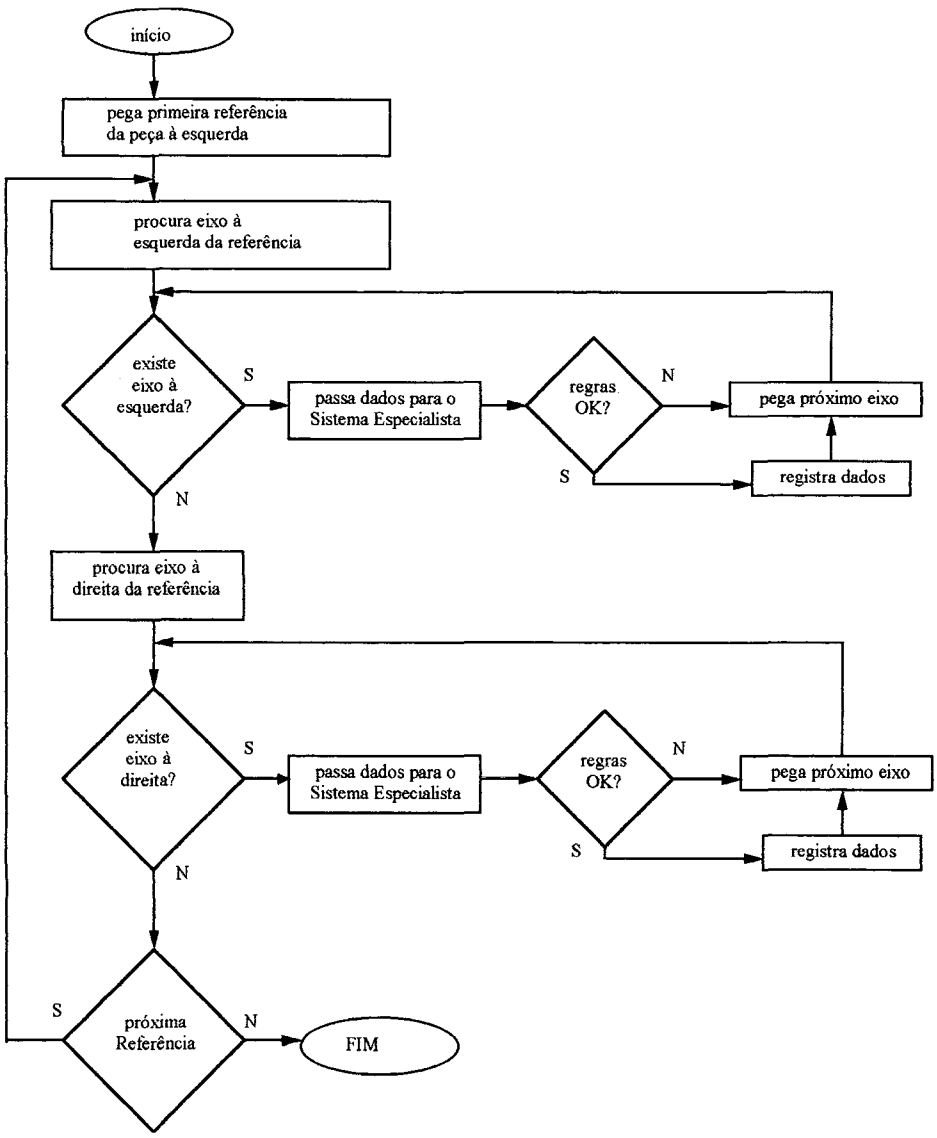


Figura 4.37 - Fluxograma para a identificação das cotas longitudinais.

Terminada essa análise, sendo definidas as cotas, são, então, identificadas aquelas cotas que possuem PtoI e PtoF coincidentes com as referências de montagem; logo, essas cotas é que devem ser controladas; são elas é que irão influenciar a montagem final no sentido longitudinal do conjunto de peças rotacionais.

Após essas identificações, o sistema FeatCAD-2D providencia a colocação das cotas (dimensões e tolerâncias) nas representações gráficas das *features*, obedecendo às regras de desenho técnico.

4.15 - IMPLEMENTAÇÃO DO SISTEMA ESPECIALISTA

A implementação do sistema especialista consiste em modelar as possíveis situações na forma de regras. Assim, cada caso a ser analisado deve ser descrito com relação às condições que proporcionam sua aprovação e às condições que desaprovam.

Lembra-se que todas as regras descritas no sistema para formar a base de conhecimento devem conduzir a uma verdade, ou seja, devem modelar a situação que é válida ou inválida de forma única, não deixando dúvidas. As dúvidas geradas num sistema especialista são resultantes da falta de descrição de informações ou, então, da ambigüidade com que é descrita uma regra.

Para implementar uma base de conhecimentos no sistema especialista, é necessário implementar um modelo das informações, tendo sido utilizado, no caso, o software CLIPS[2]. O *shell* para sistema especialista CLIPS, além de permitir a descrição em forma de classes, também possui o mecanismo de herança, o que permite simplificar em muito as definições.

Para que o sistema especialista possa realizar as análises, é necessário que essas informações conceituais sejam traduzidas numa forma que possa ser analisada pelo sistema especialista, uma vez que, na realidade, é um computador que as está analisando. E como se sabe que o computador não possui o dom dos seres humanos com relação à interpretação das informações de forma visual, as situações devem ser traduzidas em forma de parâmetros que representem condições da realidade que se quer analisar.

Nessa implementação, são construídas três bases de conhecimento em função das aplicações a serem feitas; assim, tem-se uma base de conhecimento para o modelador de *features*, uma para análise de montagem e uma para a cotagem longitudinal.

São essas bases de conhecimento que proporcionam a existência do CAD "inteligente". Porém, não se pode esperar de um CAD dito "inteligente" que seja capaz de decidir situações; ele apenas possui a capacidade de escolher opções em função das alternativas disponíveis, bem como aplicativos mais sofisticados que se comunicam mais facilmente. É o projetista, contudo, que possui o poder de tomar a decisão final, pois é ele quem detém o conhecimento no todo do objeto que deve ser representado. Na realidade, um CAD dito "inteligente" depende de vários fatores que, juntos, criam as condições necessárias para a sua existência e bom desempenho. Pode-se enumerar alguns desses fatores:

- a existência das informações de modo adequado para a aplicação a ser solicitada;

[2] *C Language Integrated Production System*

- programas auxiliares que, a partir do modelo do produto, podem extrair as informações e analisá-las;

- um modelador de *features* e também um modelador geométrico adequado para o tipo de informação a ser representada;

- sistemas baseados em conhecimento que possam ajudar a efetuar análises que envolvem heurísticas;

- definição do domínio em que o referido CAD vai atuar, refletindo a metodologia de projeto a ser aplicada.

4.15.1 - IMPLEMENTAÇÃO DA BASE DE CONHECIMENTO PARA O MODELADOR DE *FEATURES*

No modelo das *features* do sistema FeatCAD-2D, no item 3.5.1, foram descritas as situações em que o modelador de *features* deve validar ou invalidar a situação da criação de uma *feature*.

Para realizar a implementação da base de conhecimento, é necessária a criação de parâmetros que representem determinadas situações, de modo a permitir que o sistema computacional possa manipular as informações num formato adequado a ele. Assim, são utilizadas variáveis que possuem um significado tanto geométrico como tecnológico para traduzir ao sistema as informações. Cada variável descrita no sistema especialista possui um suporte conceitual que traduz uma informação a respeito dos elementos envolvidos.

A seguir, apresentam-se algumas regras implementadas no modelador de *features* e explicações a respeito de como são interpretadas as informações descritas nas regras.

REGRA 1- Execução de um eixo com sentido à direita (Figura 4.39).

IF

Peça: (DID = DED)

Eixo0: (ptLocalizaçãoX >= DED)

Eixo1: (EREF1 = Diâmetro Eixo1)

==========>>>>>

THEN

SentidoEixo0=1

Flag= 0 (o eixo é compatível)

A regra descrita possui uma representação gráfica na Figura 4.39, e a sua interpretação pode ser feita da forma como é explicitado em seqüência.

Para a criação de uma nova *feature* eixo, com exceção da primeira que compõe a peça, a análise do sistema especialista envolve informações a respeito da peça, da *feature* eixo (Eixo1) que já existe e da *feature* eixo (Eixo0) que se deseja criar e inserir no sistema (Figura 4.39).

Com relação à peça, são comparados dois parâmetros (DID == DED), onde DID corresponde a uma coordenada que controla a posição da face final do furo pelo lado direito da peça (DID descreve a localização da porção da peça que possui material disponível para a inserção de um furo); DED controla a coordenada da face externa da peça, correspondendo à posição da face final do último eixo do lado direito. Se os dois parâmetros são iguais, significa que não existe um furo no lado direito da peça, podendo então ser considerada a inserção de um novo eixo.

Com relação ao eixo a ser inserido, é verificada a posição do ponto de localização e se ele está à direita da peça (ptLocalizaçãoX >= DED), o que pode ser observado na Figura 4.39.

Com relação à *feature* eixo já existente (Eixo1), é feita a verificação se há alguma *feature* modificadora de aresta (chanfro) (EREF1 == Diâmetro Eixo1), pois, se existe, a variável EREF1 é diferente do diâmetro do eixo; se não existe, a nova *feature* pode ser inserida.

O EREI (Espaço Radial Externo Inicial) corresponde ao diâmetro de uma *feature* eixo na face inicial (Figura 4.38(a)). Se não foi introduzida nenhuma *feature* modificadora de aresta, EREI é igual ao diâmetro da *feature* eixo; contudo, ao ser inserida uma *feature* modificadora de aresta, como chanfro, essa variável passa a ser diferente do valor do diâmetro da *feature* eixo, sendo que EREI é com referência à face inicial do eixo. De modo semelhante foi definido EREF, que corresponde à variável de controle da face final de uma *feature* eixo.

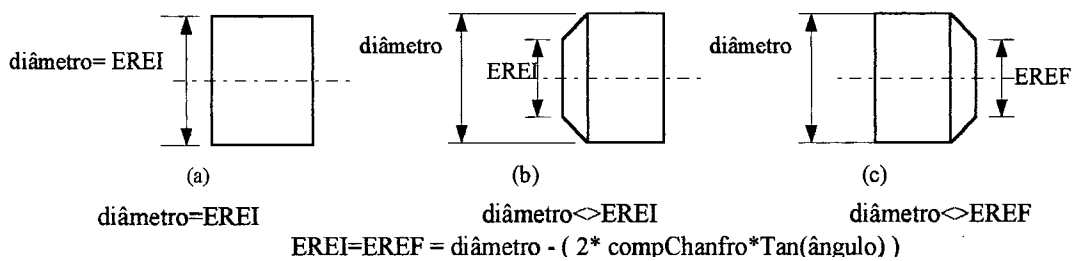


Figura 4.38 - Definição do atributo EREI e EREF.

Como resultado, o sistema especialista fornece em retorno duas informações para o sistema FeatCAD-2D, o sentido do eixo a ser desenhado (Sentido Eixo0 = 1) e a Flag (Flag = 0), que corresponde a uma permissão para a inserção da nova *feature* eixo.

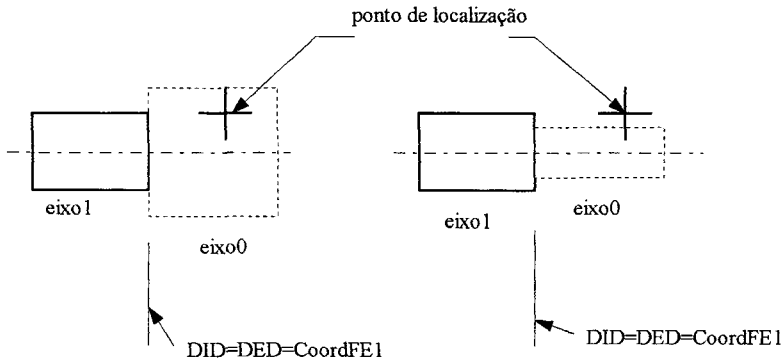


Figura 4.39 - Inserção de uma *feature* eixo à direita.

Outras variáveis são necessárias para desenhar a representação do eixo, bem como atualizar, na estrutura de dados, as informações da nova *feature* e também para atualizar as informações da peça, visto que DID e DED mudam de valor, passando a ser a coordenada da face final da nova *feature* eixo, ou seja, DID=DED=CoordFEixo0. Isso é realizado através da função auxiliar denominada de *Processos*, em que estão descritos todos esses procedimentos com relação à estrutura de dados.

REGRA 2 - Não executa um eixo à direita se existe um furo axial na face da peça (Figura 4.40).

IF

Peça: (DID <> DED) (se essa condição ocorrer, sobrepõe-se às outras regras)

Eixo0: (ptInserçãoX >= DED)

=========>>>>>

THEN

SentidoEixo0=0

Flag=2 (O furo deve ser retirado para a inserção do eixo)

Nesse caso, são analisadas informações a respeito da peça e do eixo que se deseja inserir. Para a peça, se DID é diferente de DED (Figura 4.40), significa que foi inserido um furo nessa

posição, de onde foi subtraída a profundidade do furo da posição DID. Havendo essa condição, nenhuma outra regra necessitará ser analisada, pois descarta-se a possibilidade de inserir um eixo de modo a obstruir um furo existente.

Como o ponto de localização (ptLocalizacãoX) possui coordenada maior que a coordenada de DED, o usuário deseja inserir o eixo à direita, contudo, nessas condições, não é possível a inserção da nova *feature* eixo pretendida. O sistema especialista retorna, então, o sentido do eixo que se pretendia inserir, que, no caso, é zero (SentidoEixo0=0), e a Flag, que possui um valor diferente de zero, o qual corresponde a uma mensagem de erro para o usuário, que, no caso, é "O furo deve ser retirado para a inserção do eixo".

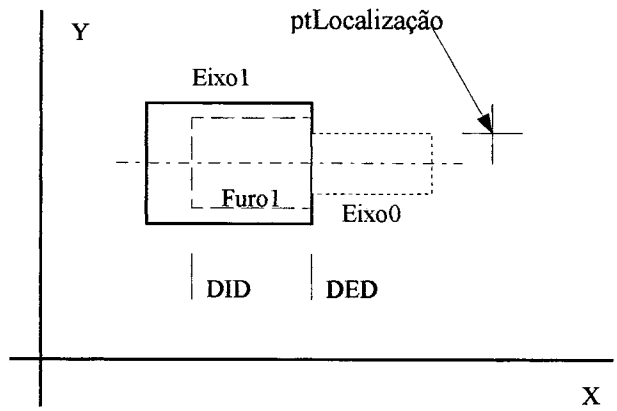


Figura 4.40 - Não permite a inserção de um eixo se há um furo.

REGRA 3 - Execução de um chanfro externo na extremidade esquerda de um eixo que está na extremidade esquerda da peça (Figura 4.41).

IF

(CoordIEixo = DEE) (DiâmetroEixo = EREI)

(PtInsercãoX < (CoordIEixo + CoordFEixo) /2) (Posição = externo)

=====>>>>>

THEN

SentidoChanfro = -1

Flag = 0 (O chanfro pode ser inserido)

Quando da inserção de uma *feature* modificadora, o usuário seleciona a *feature* básica sobre a qual deseja aplicá-la; assim, é definido se é uma *feature* eixo ou furo. No caso

específico, é uma *feature* eixo, sendo, então, a posição da *feature* chanfro externa (Posição = externo).

O ponto de localização no caso é utilizado para verificar sobre qual das faces o usuário deseja inserir a *feature* chanfro, o que é feito pela expressão (PtInsercãoX < (CoordIEixo + CoordFEixo) /2), sendo o ponto de localização analisado com relação à coordenada média das faces da *feature* eixo, a qual define o sentido da *feature* chanfro.

Se o diâmetro do eixo é igual ao EREI (DiametroEixo = EREI), significa que não há nenhuma outra *feature* modificadora de aresta nessa posição, podendo ser inserida uma *feature* chanfro. Com a expressão (CoordIEixo = DEE), deduz-se que não há nenhum outro eixo além do selecionado, não sendo necessário realizar a verificação de interferência do chanfro com um outro eixo.

Sendo preenchidas essas condições, uma *feature* chanfro pode ser inserida, sendo retornados dois parâmetros para o sistema, que são o SentidoChanfro, indicando que o chanfro é à esquerda (SentidoChanfro = -1), e a Flag (Flag=0), sinalizando que a operação pode ser efetuada.

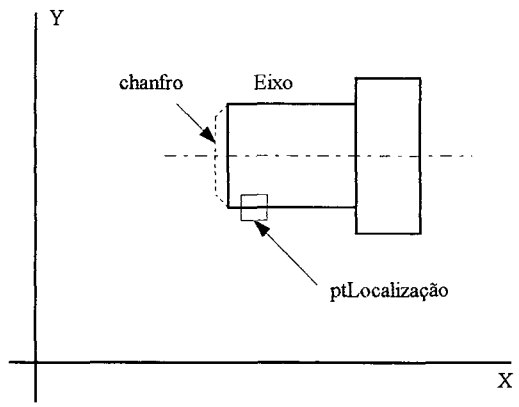


Figura 4.41 - Inserção de uma *feature* chanfro esquerdo.

REGRA 4 - Executa um chanfro num furo à direita (Figura 4.43).

IF

(CoordIFuro == DEE) (SentidoF =1) (Tipo== furocilíndricocego)

(DiâmetroFuro==ERIIFuro) (Posição==interno)

(PtInsercaoX < (CoordIFuro + CoordFFuro)/2)

(EREIEixo > (ERIIFuro + (2 * ComprimentoChanfro * Tan (Ângulo))))

=====>>>>>

THEN

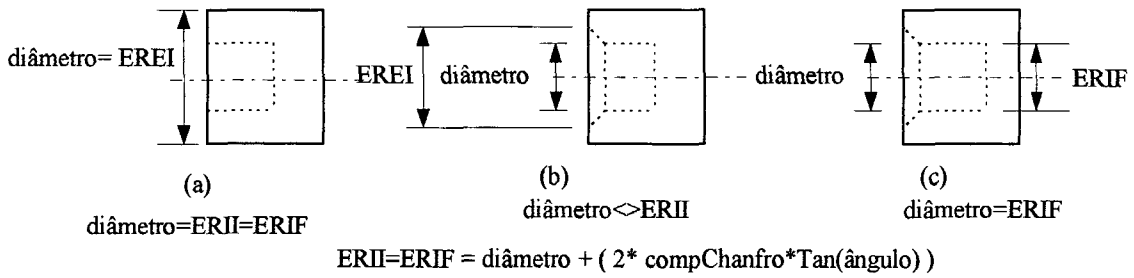
SentidoChanfro = -1

Flag = 0 (inserir chanfro)

Nessa regra, Figura 4.43, tem-se a inserção de uma *feature* modificadora chanfro em um furo. Para que um furo seja à direita, sua face inicial deve estar do lado esquerdo da peça, o que é descrito através da sentença (CoordIFuro = DEE). Através do sentido do furo, essa condição é duplamente confirmada (SentidoF =1).

Como uma *feature* furo pode ser do tipo cilíndrico cego ou passante, pois a inserção de chanfro possui algumas peculiaridades a cada uma, torna-se necessário definir o tipo que, no caso, é furo cilíndrico cego (Tipo= furocilíndricocego). Outra condição verificada é quanto à existência no mesmo local de uma outra *feature* modificadora de aresta, o que é feito através do espaço radial do furo (DiâmetroFuro=ERIIFuro).

A variável espaço radial do furo é associada à face inicial e final do furo, resultando em ERII e ERIF. Inicialmente, essa variável possui o mesmo valor do diâmetro do furo e, quando há um modificador de aresta aplicado, o diâmetro do furo é aumentado de um valor tornado diferente do diâmetro, o que possibilita identificar a existência de uma *feature* modificadora na referida posição. Na Figura 4.32, estão representadas as duas variáveis para o caso do furo cego.



ERII=ERIF = diâmetro + (2* compChanfro*Tan(ângulo))

Figura 4.42 - Definição do atributo ERII e ERIF.

A posição da *feature* básica sobre a qual se deseja inserir a *feature* modificadora é confirmada através da posição (Posição=interno).

A localização em que se deseja inserir a *feature* chanfro é verificada através do ponto de localização, o qual deve estar mais próximo da face inicial do respectivo furo (PtInsercaoX < (CoordIFuro + CoordFFuro)/2).

Por último, é feita a verificação quanto à possível interferência do chanfro a ser inserido no furo com o eixo ou alguma *feature* modificadora de aresta, análises que são realizadas comparando-se os espaços radiais das duas *features* básicas (EREIEixo > (ERIIFuro + (2 * ComprimentoChanfro * Tan (Ângulo)))).

Como resultado, o SentidoChanfro é à esquerda (SentidoChanfro= -1), e a flag (Flag=0) sinaliza para o sistema a autorização para completar a operação de inserção da *feature* chanfro.

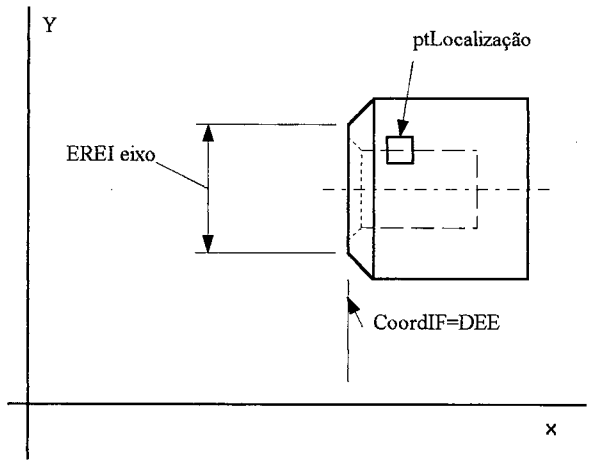


Figura 4.43 - Inserção de um chanfro num furo à direita.

4.15.2 - IMPLEMENTAÇÃO DA BASE DE CONHECIMENTO PARA AS FUNÇÕES AUXILIARES DO MODELADOR GRÁFICO

Como descrito no item 3.7, as funções auxiliares básicas necessárias ao modelador gráfico são excluir, alterar e mover. Dentre essas, apenas excluir e alterar é que necessitam de um sistema especialista para funcionarem adequadamente.

Para se efetuar a implementação das duas funções, o método será semelhante ao que está sendo descrito para a implementação da base de conhecimento para o modelador gráfico, para a montagem e para a cotagem. Para isso, deve-se seguir o modelo conceitual descrito no capítulo 3, itens 3.7.2 e 3.7.3.

4.15.3 - IMPLEMENTAÇÃO DA BASE DE CONHECIMENTO PARA MONTAGEM

A implementação da base de conhecimento para a montagem segue o princípio adotado no modelador gráfico, ou seja, as informações são enviadas ao sistema especialista, o qual dá como retorno uma mensagem autorizando ou não a concretização da operação.

Para as análises, são utilizadas as informações gráficas que estão representadas na estrutura de dados, que, através dos algoritmos descritos nos itens 4.14.1.1 e 4.14.1.2, verificam as condições mínimas que caracterizam uma montagem; o sistema especialista, por sua vez, realiza uma análise mais detalhada. As classes definidas no modelo do produto são as mesmas utilizadas para efetuar a análise no sistema especialista.

Como descrito nos itens 3.81 e 3.82, são realizados dois tipos de análise em função do contato diametral e do contato axial, constituindo-se de dois módulos independentes numa mesma base de conhecimento.

4.15.3.1 - REGRAS DO SISTEMA ESPECIALISTA PARA IDENTIFICAÇÃO DO CONTATO DIAMETRAL

Como descrito no modelo do sistema, capítulo 3, o sistema especialista é que possui a função de realizar a avaliação quanto à tomada de decisões do sistema. A cada momento em que uma análise é realizada através do sistema especialista, as informações a respeito das peças e *features* envolvidas são passadas para o sistema especialista através de uma interface de comunicação, a qual realiza a conversão do formato de dados utilizado no CAD para o formato do sistema especialista.

Estando o sistema especialista de posse dos dados e conhecendo-se o formato de representação das informações no sistema especialista, regras podem ser escritas para modelar as condições de montagem.

Em vista das considerações feitas com relação ao contato diametral no item 3.8.1, as regras são construídas visando representar essas condições que identificam uma montagem. Os mesmos atributos descritos na definição das *features* são utilizados para a construção das regras, sendo que, conforme descrito no item 2.10.2, uma regra é composta de duas partes: o lado

esquerdo da regra, "IF" - a condição, e o lado direito, "THEN" a ação ou conclusão relativa à condição descrita. Assim, pode-se descrever uma primeira regra com relação à montagem:

REGRA 1: Não há montagem de duas peças se o furo e o eixo de cada peça não estão centrados axialmente (Figura 4.44).

IF

((Tipo = furocilíndricopassante) V (Tipo = furocilíndricocego))

(DiâmetroF = DiâmetroE) ((SentidoF= 1) V (SentidoF= -1)) (CoordYF <> CoordYE)

=====>>>>>

THEN

Flag =1 (Não há montagem)

Esta regra condiciona que, para qualquer tipo de furo cilíndrico (passante ou cego), mesmo sendo o diâmetro do furo igual ao do eixo (DiametroF == DiâmetroE) e o sentido do furo qualquer um (SentidoF= 1 ou -1), mas as coordenadas do centro de rotação do eixo diferentes do centro de rotação do furo (CoordYF <> CoordYE), a montagem não ocorre entre as duas peças em análise através das respectivas *features* eixo e furo (ver Figura 4.44).

Como conclusão da regra (Flag = 1), a montagem não ocorre e, através do valor da "Flag", é informada a função de erro que produz a mensagem "Não há montagem" para o usuário através da interface gráfica.

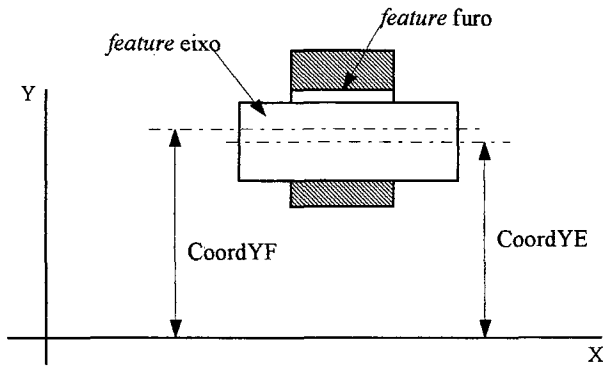


Figura 4.44 - Condições gráficas da regra.

REGRA 2: Há montagem se eixo está com Coordf fora do furo e CoordiE=CoordiF (Figura 4.45).

IF

(CoordYF = CoordYE) (Tipo =furocilíndricopassante) (DiâmetroF = DiâmetroE)
(Sentido= 1) (CoordIF = CoordIE) (CoordFF < CoordFE)
((CoordFF - CoordIF) >= DiâmetroF / 2)
=====>>>>>

THEN

Flag = 0 (Há montagem)

Como condição inicial, tem-se que os centros de rotação da *feature* furo e da *feature* eixo devem ser coincidentes (CoordYF = CoordYE) (Figura 4.45). No caso, está sendo analisado para uma *feature* furo cilíndrico passante (Tipo= furocilíndricopassante); outra condição de montagem é que os diâmetros das *features* furo e eixo envolvidas devem possuir a mesma dimensão nominal (DiâmetroF = DiâmetroE); o furo deve possuir sentido à direita (Sentido= 1).

A condição de posicionamento axial (verificação da sobreposição) entre o furo e o eixo é analisada com relação ao posicionamento das suas respectivas faces inicial e final. Uma das condições considera que as faces iniciais da *feature* furo e eixo sejam coincidentes (CoordIF = CoordIE) (Figura 4.45); que a face final do furo não seja coincidente com a face final do eixo e que esteja localizada mais próxima da face inicial do furo (CoordFF < CoordFE) (Figura 4.45).

Finalmente, é verificada a condição da dimensão de apoio entre a *feature* furo e eixo, tendo-se considerado, no caso, que a mínima deve ser igual à metade do diâmetro da *feature* furo ((CoordFF - CoordIF) >= DiâmetroF / 2) (Figura 4.45).

Sendo a regra disparada, o sistema especialista envia uma mensagem (Flag=0), autorizando que o referido par de acoplamento das *features* furo e eixo seja considerado uma montagem.

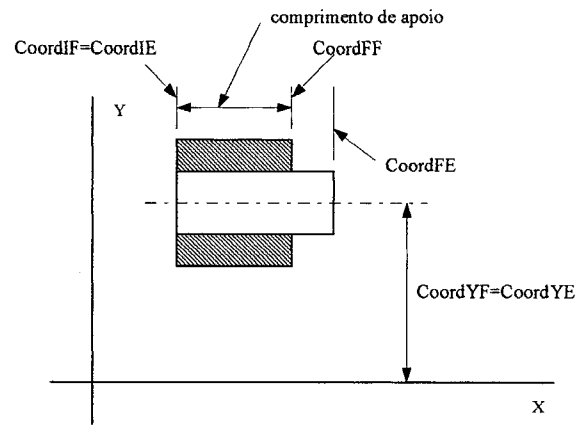


Figura 4.45 - Representação da regra.

4.15.3.2 - REGRAS DO SISTEMA ESPECIALISTA PARA IDENTIFICAÇÃO DO CONTATO AXIAL

O contato axial pressupõe a existência de duas faces de duas *features* de peças distintas em contato, ou seja, que suas coordenadas de posição (coordenadas X) possuam o mesmo valor nominal. Somente é feita a verificação do contato axial para aquelas *features* que possuam contato diametral definido e a condição de montagem aceita.

REGRA 1: Contato axial de uma peça com furo à direita e um eixo (Figura 4.46).

IF

(CoordYP0 = CoordYP1) (DEDP0 <> CoordFE0) (SentidoF1= 1)

(DiametroF1 < DiametroE0) (CoordFE0 = CoordIF1)

=====>>>>>

THEN

Flag = 0 (há contato axial entre as peças)

A análise considera a peça que possui a *feature* furo como a Peça1, e a peça que possui a *feature* eixo como a Peça0. Para que possa haver o contato axial entre um furo e um eixo, é necessário que os centros de rotação das peças envolvidas sejam coincidentes (CoordYP0 =

CoordYP1). Além disso, a face final da *feature* eixo que coincide com a face inicial da *feature* furo não deve ser a extremidade direita da peça (DEDP0 <> CoordFE0).

O sentido da *feature* furo da Peça1 deve ser à direita (SentidoF1= 1); o diâmetro da *feature* furo deve ser menor que o diâmetro da *feature* eixo vizinha para que haja o contato axial (DiâmetroF1 < DiâmetroE0). Finalmente, a face final da *feature* eixo deve ser coincidente com a face inicial da *feature* furo (CoordFE0 == CoordIF1).

Essa regra analisa o caso em que as peças que possuem contato diametral são as mesmas que possuem o contato axial.

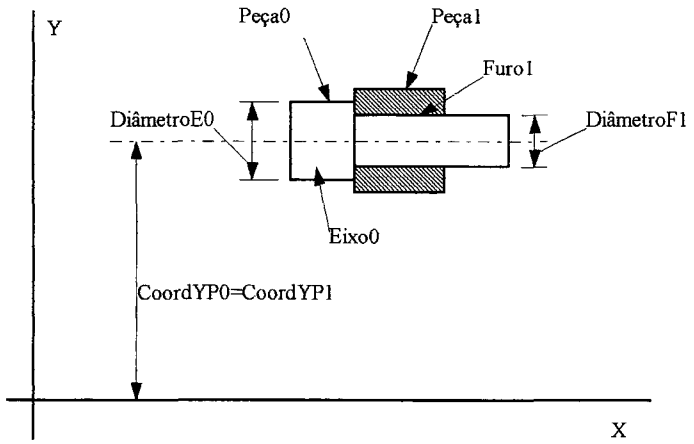


Figura 4.46 - Representação da regra de contato axial.

REGRA 2: Contato axial de uma peça com um furo à direita com outra peça qualquer (Figura 4.47).

IF

(CoordYP0 = CoordYP1) (DEDP0 = CoordFE0) (SentidoF1= 1)

(DiâmetroF0 < DiâmetroE1) (CoordFE0 = CoordIF1)

=====>>>>>

THEN

Flag = 0 (Há contato axial entre as peças)

Esta regra analisa duas peças que possuem somente contato axial entre si, sendo o contato diametral realizado através de uma terceira peça. Pela regra, a primeira consideração é com relação ao centro de rotação das peças envolvidas, que deve ser coincidente (CoordYP0 == CoordYP1). Como, nesse caso, as peças envolvidas não possuem contato diametral, a montagem

ilustrada na Figura 4.47 considera que o eixo da Peça0 é o último do lado direito da peça (DEDP0 = CoordFE0).

O sentido da *feature* furo é à direita (SentidoF1= 1). O diâmetro do furo da Peça0 deve ser menor que o diâmetro do eixo da Peça1, para que ocorra realmente o contato axial (DiâmetroF0 < DiâmetroE1), do contrário a Peça0 e a Peça1 teriam apenas coincidência de posicionamento. Para confirmar o contato axial, a face final do eixo da Peça0 deve ser coincidente com a face inicial da *feature* furo da Peça1 (CoordFE0 == CoordIF1).

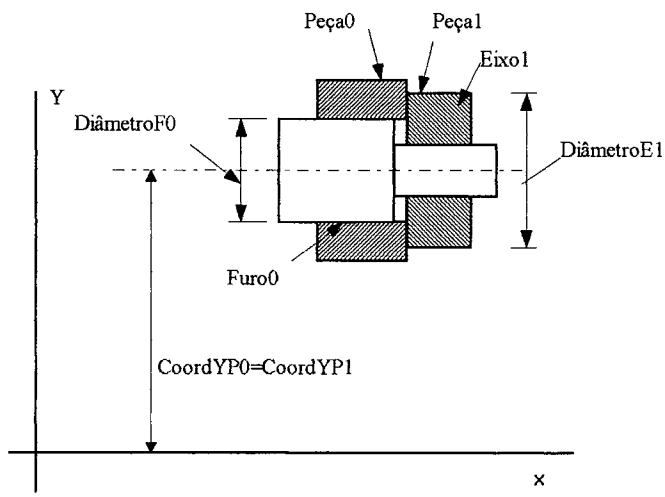


Figura 4.47 - Regra de contato axial.

REGRA 3: Não há contato axial de uma peça com um furo à direita com outra peça qualquer (Figura 4.48).

IF

(CoorYP0 = CoordYP1) (DEDP0 = CoordFE0) (SentidoF1= 1)

(DiâmetroF0 > DiâmetroE1) (CoordFE0 = CoordIF1)

=====>>>>>

THEN

Flag = 1 (Não há contato axial entre as peças)

Esta regra é semelhante à anterior, com a diferença de que o diâmetro da *feature* furo da Peça0 é maior que o diâmetro da *feature* eixo da Peça1, o que não produz um contato entre as faces das duas peças envolvidas. Na regra, essa condição está especificada através da seguinte sentença: (DiâmetroF0 > DiâmetroE1).

Se as condições desta regra forem satisfeitas, uma mensagem é passada pelo sistema especialista não considerando a existência do contato axial (Flag = 1). O número recebido pela variável "Flag" corresponde a um código de erros que pode ser utilizado para enviar mensagens ao usuário.

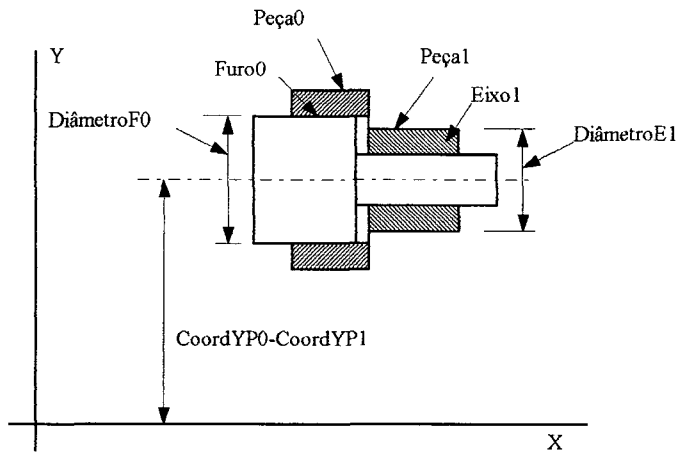


Figura 4.48 - Representação da regra de não-ocorrência de contato axial.

4.15.4 - IMPLEMENTAÇÃO DA BASE DE CONHECIMENTO PARA A COTAGEM

A base de conhecimento para o dimensionamento contempla a análise das referências de montagem da peça, que servem de ponto inicial da dimensão e das faces das *features* eixo que não são coincidentes com a coordenada de referência e que serão os pontos finais das dimensões (Figura 4.49).

Para a análise das dimensões da peça, a cotagem foi subdividida em dois tipos: i) quando a peça possui uma única superfície de referência de montagem (Figura 4.49(a)) e ii) quando há mais de uma superfície de referência de montagem (Figura 4.49(b)). Assim, as considerações para a distribuição das dimensões na peça, levando em conta a funcionalidade, podem ser realizadas. No presente trabalho, a análise das dimensões longitudinais leva em conta apenas as dimensões externas, ou seja, relativas às *features* eixos.

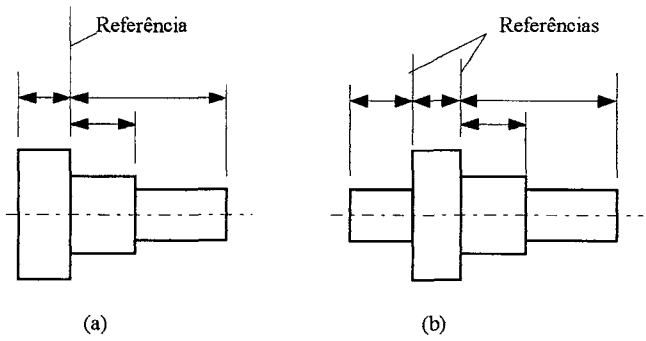


Figura 4.49 - Uso das referências de montagem para a cotagem.

a) Dimensionamento com uma coordenada de referência

REGRA 1: Cotagem à direita e uma coordenada de referência (Figura 4.50(a)).

IF

(numeroReferências = 1)

(PtoMédio = PtoI) (PtoF > PtoI) (TipoReferência = externa)

=====>>>>>

THEN

Flag = 0

Sentido = 1

Se a peça possui apenas uma superfície de contato axial, ela possui uma única referência para o dimensionamento (númeroReferências= 1). Assim, o PtoMédio, que é a média das distâncias entre duas referências numa mesma peça, é considerado igual ao ponto inicial (PtoI) da dimensão. A condição (TipoReferência= externa) assegura que o dimensionamento é para uma *feature* externa (nesse caso, um eixo). Ainda, há o requisito de que essa dimensão possua o seu ponto final (PtoF) à direita da referência (PtoF > PtoI). Nessas condições, a dimensão é aceita (Flag=0), possuindo o sentido para a direita (Sentido= 1) (Figura 4.50(a)).

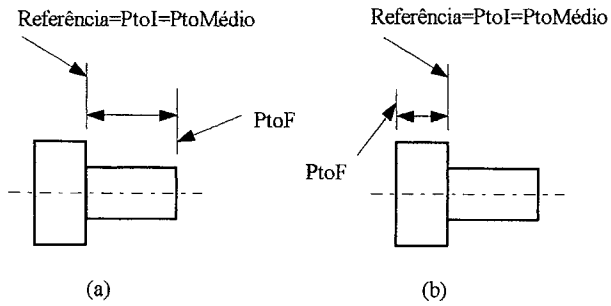


Figura 4.50 - Cotagem com uma única referência.

REGRA 2: Cotagem à esquerda e uma coordenada de referência (Figura 4.50(b)).

IF

(PtoMédio = PtoI) (PtoF < PtoI) (TipoReferência = externa)

(númeroReferências = 1)

=====>>>>>

THEN

Flag = 0

Sentido = -1

A análise feita pela regra 1 é semelhante para a regra 2, apenas mudando o sentido da dimensão, que passa a ser à esquerda (Sentido= -1).

b) Dimensionamento com mais de uma coordenada de referência

REGRA 1: Cotagem à esquerda se há mais de uma referência e essa cota refere-se à primeira referência (Figura 4.51).

IF

(PtoMédio = PtoI) (PtoF < PtoI) (TipoReferência = externa) (númeroReferências > 1)

=====>>>>>

THEN

Flag = 0

Sentido = -1

Para a primeira referência, nesse caso, o ponto médio é considerado igual ao ponto inicial da dimensão (PtoMédio = PtoI), e o número de superfícies de referência é maior que um

(númeroReferência > 1). Isso satisfaz para a colocação de dimensões à esquerda da primeira referência de uma peça que possui mais de uma referência (Figura 4.51).

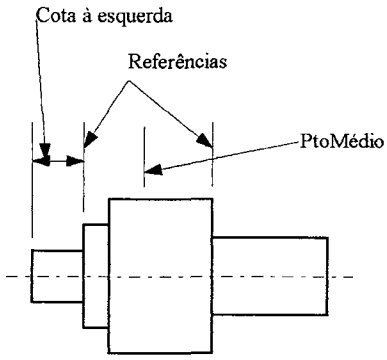


Figura 4.51 - Cota à esquerda para duas referências.

REGRA 2: Cotagem à direita se existe mais de uma referência (Figura 4.52).

IF

(PtoMédio > PtoI) (PtoF > PtoI) (TipoReferência = externa) (PtoF < PtoMédio)

(númeroReferências > 1)

=====>>>>>

THEN

Flag = 0

Sentido = 1

Nesse caso, está localizando uma dimensão do lado direito da referência, sendo que a segunda referência está à direita dessa. Logo, o elemento a ser dimensionado deve ter o comprimento menor que o ponto médio entre as duas referências ((PtoMédio>PtoI) $\wedge$ (PtoF > PtoI) $\wedge$ (PtoF < PtoMédio)). Assim, a dimensão resultante é à direita (Sentido=1) (Figura 4.52).

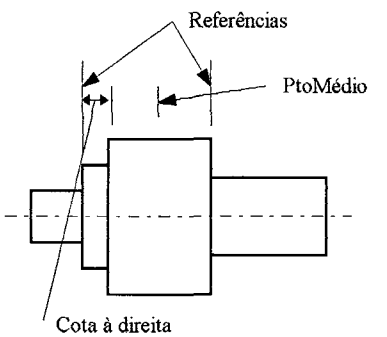


Figura 4.52 - Cota à direita para duas referências.

De modo semelhante, há uma regra que analisa o dimensionamento à esquerda quando da existência de mais de uma referência na peça.

4.15.5 - ASPECTOS COMPUTACIONAIS DA IMPLEMENTAÇÃO DO SISTEMA ESPECIALISTA

O sistema especialista foi implementado utilizando-se o *shell* CLIPS na sua versão 6.0; a base de conhecimento é escrita na forma de arquivo-texto, o qual é interpretado mas não compilado com o programa principal. Isso permite uma flexibilidade quanto à alteração das regras escritas na base de conhecimento, pois podem ser alteradas a qualquer momento e testadas sem que os arquivos-fontes do programa sejam compilados.

Como o *shell* de sistema especialista CLIPS é feito em linguagem C, a comunicação com o sistema FeatCAD-2D é praticamente direta. O respectivo *shell* fornece funções de manipulação em linguagem C, que permitem que o sistema FeatCAD-2D possa criar instâncias as quais são manipuladas no sistema especialista.

A utilização dessas funções permite criar uma interface de comunicação de forma que as informações da estrutura de dados sejam fornecidas ao sistema especialista no momento em que determinadas regras vão utilizá-las.

Várias interfaces de comunicação foram adaptadas para as aplicações específicas como: i) criar *features*; ii) análise do conjunto; iii) cotagem, etc. As interfaces desenvolvidas fazem parte do programa principal, não sendo alteradas facilmente, para isso sendo necessário manusear os arquivos-fontes do sistema.

A interface de comunicação fornece informações para o sistema especialista (uma grande quantidade de informações), cujo retorno também é feito através de uma interface de comunicação; na maioria das vezes, apenas dois parâmetros retornam.

4.16 - IMPLEMENTAÇÃO DE BANCO DE DADOS

A construção de bancos de dados no sistema visa disponibilizar, de forma organizada e rápida, informações necessárias ao usuário para que ele possa desempenhar adequadamente suas funções. Para isso, podem ser construídos vários bancos de dados de acordo com a aplicação, como, por exemplo, para materiais, tolerâncias, componentes padronizados e componentes normalizados, ferramentas, etc.

No presente trabalho, a consulta aos bancos de dados pode ser feita tanto pelo usuário como pelo próprio sistema FeatCAD-2D, que é construído para realizar automaticamente a consulta das informações necessárias à execução das tarefas.

Quando a análise de montagem é realizada, um par de peças acopladas é identificado e o diâmetro nominal é conhecido, a consulta ao banco de dados de tolerâncias é ativada automaticamente; assim, de modo interativo, o usuário completa a pesquisa, fornecendo informações solicitadas pelo banco de dados. Por exemplo, quando da inserção de uma *feature* rolamento rígido de esferas, o usuário identifica o diâmetro do eixo no qual o rolamento será montado e, automaticamente, o sistema consulta o banco de dados, identificando um rolamento que possua o diâmetro desejado; do contrário, informa ao usuário sobre a indisponibilidade da informação.

4.16.1 - BANCO DE DADOS DE MATERIAIS

O objetivo do banco de dados de materiais é ter informações a respeito dos materiais que estejam à disposição do usuário, quer na empresa ou não. Dessa forma, os materiais são cadastrados no banco de dados em função do seu nome, como aço SAE 1020; são então, disponibilizadas informações a respeito do material para que o usuário possa conhecer as suas características de uso, como tensão de tração, ruptura, etc.

O acesso ao banco de dados de materiais é feito automaticamente pelo sistema no momento em que é criada uma nova peça. Também é possível ao usuário o acesso a qualquer momento para que possa realizar consultas sobre materiais. Nesse banco de dados, estão à disposição informações a respeito das características do material de modo a fornecer subsídios ao projetista na hora da escolha.

4.16.2 - BANCO DE DADOS DE TOLERÂNCIAS

Baseado no conceito de escolher tolerâncias pela função (Weill, 1988), é organizado o banco de dados para tolerâncias, ajustes e aplicações características relacionadas aos pares acoplados. A organização das informações a serem pesquisadas no banco de dados baseia-se na literatura, donde se obteve uma classificação de ajustes e tipos de precisão, bem como de exemplos de aplicações (Tabela 4.1, baseada em Pugliesi, 1986 e Novaski, 1994).

Com base na tabela 4.1, o banco de dados é estruturado de modo a efetuar a pesquisa conforme descrito no item 3.95, para a escolha das tolerâncias para as cotas diametrais. Assim, na Figura 4.53, estão representadas as opções de consulta que podem ser efetuadas no banco de dados. A opção 1 representa a consulta da Tabela 4.1 pela coluna que define a "Caracterização e Exemplos de Aplicação"; nela, os exemplos são apresentados ao usuário com o respectivo par de acoplamento e outras informações constantes da tabela. Por exemplo, se for escolhido exemplo de aplicação "embuchamento de roda", o banco de dados mostra ao usuário que o ajuste é do tipo "incerto forte" e que há dois tipos de precisão para esse tipo de montagem, que são a "extraprecisa" (H6k5) e a "precisa" (H7k6) (Tabela 4.1).

Escolhendo um dos pares de acoplamento, o banco de dados fornece os afastamentos superior e inferior relacionados ao diâmetro nominal do acoplamento, ao ajuste e à qualidade de trabalho, tanto para o furo como para o eixo, passando-os para a estrutura de dados do produto. As informações referentes aos afastamentos são obtidas de tabelas definidas em normas (ABNT NB-86, Ajustes recomendados ISO) e cadastradas no sistema, ficando à disposição para a consulta sobre qualquer tipo de aplicação.

Pela opção 2 (Figura 4.53), a pesquisa ao banco de dados é feita escolhendo-se como parâmetros de entrada o Ajuste e a Precisão para o acoplamento (Tabela 4.1). O banco de dados fornece, então, informações a respeito dos pares de acoplamentos cadastrados, bem como as respectivas aplicações para os mesmos. Feita a escolha, as informações sobre os afastamentos superior e inferior relativos ao diâmetro nominal do acoplamento, ao ajuste e à qualidade de trabalho são repassadas ao sistema.

Tabela 4.1 - Tabela básica contendo os tipos de acoplamentos e as diferentes exigências de precisão.

ajuste	exemplo de ajuste	extra preciso	preciso	media precisao	comum	CARACTERIZACAO E EXEMPLOS DE APLICACAO
folga rotativo forte		H6g5	H7d9	H8d10	H11a11	utilizam-se em pecas que devam ter uma ampla folga. Mancais de turbo-geradores, casos especiais
folga rotativo livre			H7e8			Pecas cujo funcionamento necessitam de folga por motivo de dilatacao termica ou mau alinhamento.
folga rotativo			H7f7			Pecas que giram ou deslizam com pouca lubrificacao.
folga semi-rotativo			H7g6	H8f8 H8f9	H11d11	Pecas que deslizam e giram com grande precisao. engrenagens deslizantes em caixas de combio
folga leve		H6h5	H7h6	H8h8 H8h9		pecas que bem lubrificadas pode-se monto-los e des-monta-las com a mao. aneis distanciadores, colunas moveis de furadeiras
incerto leve		H6f5(i.l.)	H7j6(i.l.)			pecas a acoplar e desacoplar a mao ou golpe com martelo de borracha. Aneis inernos de rolamentos de esfera paro pequenos cargas e aneis externos de rolamentos fixados nas carcacas
incerto forte		H6k5(i.f.)	H7k6(i.f.)			acoplamento fixo e desmontagem pouco frequentes, podendo desacoplar-se a golpe de martelo comum. o movimento de rotacao e assegurado por chaveta. Embuchamento de rodas, rotores de turbinas e bombas centrifugas.
interfe-rencia leve		H6m5				Montagem e desmontagem com martelo sem danificar o ajuste. Aneis internos de rolamentos montados em eixos para cargas normais.
interfe-rencia forte		H6n5	H7n6			pecas que devam ficar solidamente acopladas, podendo acoplar ou desacoplar mediante pressao. Movimento de rotacao garantido por chaveta. Eixos de motores eletricos
interfe-rencia prensado			H7n6			pecas de ajuste permanente unidas com muito pressao. Eixo de saida de redutor de ponte rolante de empresa side-rurgica, acoplado a engrenagem.

Se, por exemplo, o ajuste escolhido for "Interferência Forte" e o grau de precisão for "Preciso", o par de acoplamento é H7n6, sendo uma aplicação típica para eixos de motores elétricos. Nas opções 1 e 2, a dimensão do diâmetro nominal das peças que estão acopladas (eixo/furo) é identificada quando da análise do conjunto montado e automaticamente passada pelo sistema FeatCAD-2D para o banco de dados, para a consulta das informações.

Na opção 3 (Figura 4.53), a escolha das tolerâncias e ajustes é realizada em função do produto, ou seja, tolerâncias e ajustes recomendados estão cadastrados em função do produto. Dessa forma, foi implementado no presente trabalho um cadastro específico para rolamentos (o mesmo poderia ter sido feito para arruelas, parafusos, etc.), cujo banco de dados é construído para satisfazer as exigências de escolha conforme a aplicação do rolamento de acordo com a orientação do fabricante. Os afastamentos superior e inferior são retirados das tabelas de ajustes e tolerâncias já cadastradas no sistema.

Pode-se observar uma particularidade para o caso da implementação do banco de dados para rolamentos, que deve ser implementado tanto em função do diâmetro interno como do diâmetro externo, pois o produto oferece montagem nas duas formas.

A terceira opção mostrou-se trabalhosa para a implementação, pois cada produto possui características próprias, devendo ser adaptada a interface em função de cada um.

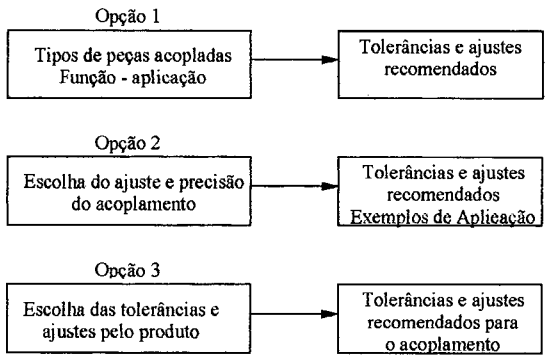


Figura 4.53 - Opções do banco de dados para tolerâncias.

A caracterização da escolha das tolerâncias por função depende do conhecimento da aplicação, à qual estão relacionados os pares de acoplamento conhecidos e cadastrados. Essas informações cadastradas no banco de dados são fruto da experiência de outros (literatura) e também da própria empresa (cultura da empresa).

4.16.3 - BANCO DE DADOS PARA COMPONENTES PADRONIZADOS E NORMALIZADOS

A implementação de um banco de dados para componentes padronizados é função exclusiva do produto a ser cadastrado, ou seja, das características necessárias para realizar a seleção das cotas do componente a ser utilizado.

No presente trabalho, realiza-se a implementação de um banco de dados de rolamentos rígidos de esfera, de modo que a escolha é em função das dimensões do diâmetro ou do furo, se montado num eixo ou numa caixa.

Cabe ao usuário escolher o tipo de rolamento - rolamento de rolos, rolamento axial de esfera, rolamento rígido de esferas - escolhendo então, onde será montado, o que é feito selecionando a *feature* eixo ou furo. Com essa seleção, é identificado o diâmetro que é passado

ao banco de dados. Com o valor do diâmetro interno/externo, é feita a procura no banco de dados; se encontrado um rolamento com esse diâmetro, as informações a respeito das dimensões do rolamento são passadas para a estrutura de dados do produto, sendo executada a representação gráfica do componente.

4.16.4 - ASPECTOS COMPUTACIONAIS DA IMPLEMENTAÇÃO DE BANCO DE DADOS

A implementação do banco de dados foi realizada com a utilização do software de banco de dados DELPHI na versão 2.0, na qual foram construídas a base de dados e as funções de manipulação em linguagem de programação PASCAL. Como o sistema FeatCAD-2D é construído em linguagem C++, foi necessário realizar a comunicação entre os softwares, o que foi conseguido com a utilização de DLL feitas em PASCAL.

Como o ambiente operacional do sistema FeatCAD-2D é o "Windows" e as DLL geradas em qualquer software "Windows" podem ser intrepretadas por qualquer outro software também "Windows", as funções descritas nas DLL feitas em PASCAL no banco de dados podem ser chamadas dentro do sistema FeatCAD-2D em tempo de execução (essas DLL não fazem parte do programa executável principal). Desse modo, os dados do sistema FeatCAD-2D podem ser passados para o banco de dados e as informações do banco de dados, para o sistema de origem.

4.17 - LIMITAÇÕES DO MODELO IMPLEMENTADO

As análises realizadas neste modelo abordam a representação das informações de peças rotacionais através de um sistema CAD, sendo posteriormente realizada a análise do conjunto, cotagem dos componentes do conjunto e a escolha iterativa das tolerâncias pelo usuário. Esses itens correspondem aos três primeiros descritos no item 3.2.3.

Com as informações obtidas a respeito da análise da montagem, é possível obter a seqüência de montagem, o que não é contemplado neste modelo.

Os aspectos de representação das informações no sistema CAD correspondem à representação do projeto detalhado (item 3.4.1), considerando que o projeto conceitual já esteja definido.

As *features* representadas no sistema correspondem àquelas que se adaptam a configurar peças rotacionais.

Considera-se as peças representadas no modelo como oriundas de barras como matéria-prima, não sendo considerado o caso de peças fundidas ou obtidas através de operações de soldagem.

As *features* compostas descritas no item 3.5.1 não são implementadas no modelo, permanecendo apenas em nível conceitual.

As *features* modificadoras de primeira e segunda ordem não foram implementadas, permanecendo também em nível conceitual.

Com relação à cotagem automática, essa se limita a identificar as cotas de controle e a dispor sobre a representação gráfica, sendo que a presente implementação não abordou a distribuição das mesmas de forma racional sobre o desenho (Figuras 5.41, 5.42 e 5.43).

Com relação à escolha de tolerâncias, o sistema limita-se a identificar as *features* que devem ser controladas a apresentar informações (banco de dados de tolerâncias) para que o usuário proceda à escolha de acordo com o seu conhecimento.

CAPÍTULO 5

EXEMPLOS DE APLICAÇÃO E RESULTADOS DA IMPLEMENTAÇÃO

Um conjunto montado de peças rotacionais é apresentado na Figura 5.1. O conjunto é composto por um *corpo* que é fechado por uma *tampa*, em cujo interior está alojado um *eixo* que tem montado sobre ele um *rotor* e uma *bucha*. O *eixo* se apóia em furos localizados no *corpo* e na *tampa*.

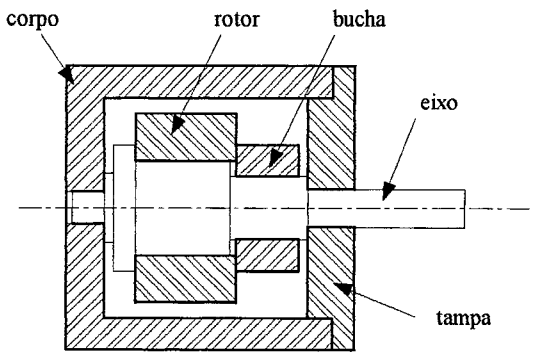


Figura 5.1 - Conjunto montado, baseado em DINI (1992).

A seguir, apresenta-se uma rápida seqüência para a definição desse conjunto no sistema FeatCAD-2D, bem como as análises que podem ser realizadas conforme descrito no modelo do sistema (capítulo 3) e na implementação (capítulo 4).

5.1 - O CONJUNTO

Um conjunto é definido quando da inicialização do sistema. Desse modo, é necessário inicializar o CAD e solicitar a criação com a seqüência **Criar - Conjunto - Iniciar** (Figura 5.2). Com a inicialização, todos os arquivos do sistema são carregados para a memória do computador, sendo inicializada a estrutura básica do produto.

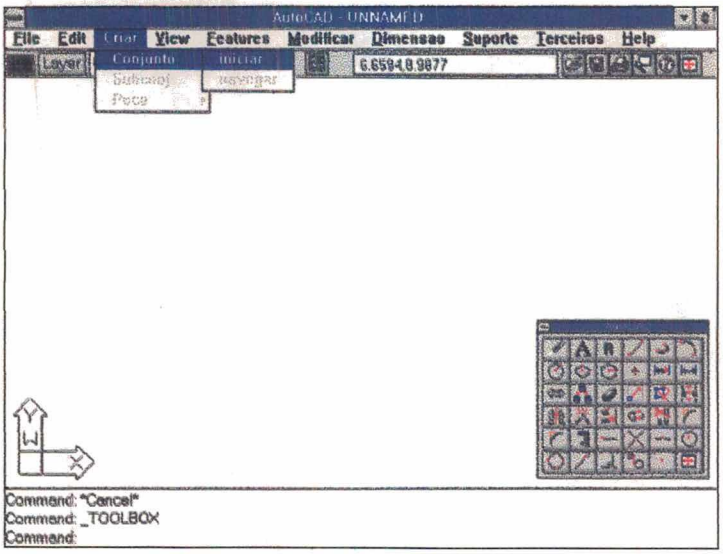


Figura 5.2 - Inicialização do sistema.

5.2 - DEFINIÇÃO DE UM SUBCONJUNTO

O próximo passo para a estruturação do produto é a definição do primeiro subconjunto para que possam ser criadas as peças. Isso pode ser visto na Figura 5.3, onde a opção no menu para criar subconjunto está ativa, pois o comando somente é ativado após a inicialização do sistema com o conjunto.

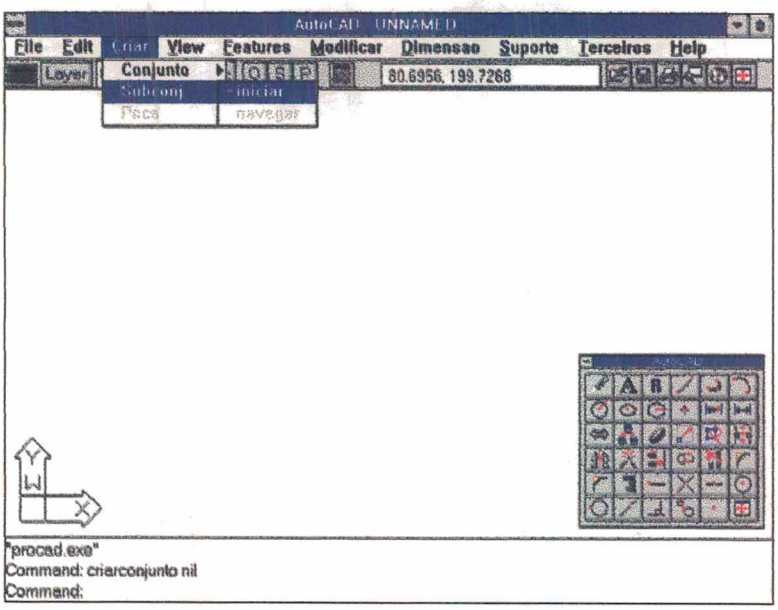


Figura 5.3 - Iniciar um subconjunto.

Na Figura 5.4, está aberto o quadro de diálogo para definir o primeiro subconjunto do produto, sendo, para isso, necessário especificar o nome do subconjunto e a data de sua criação para fins de registro.

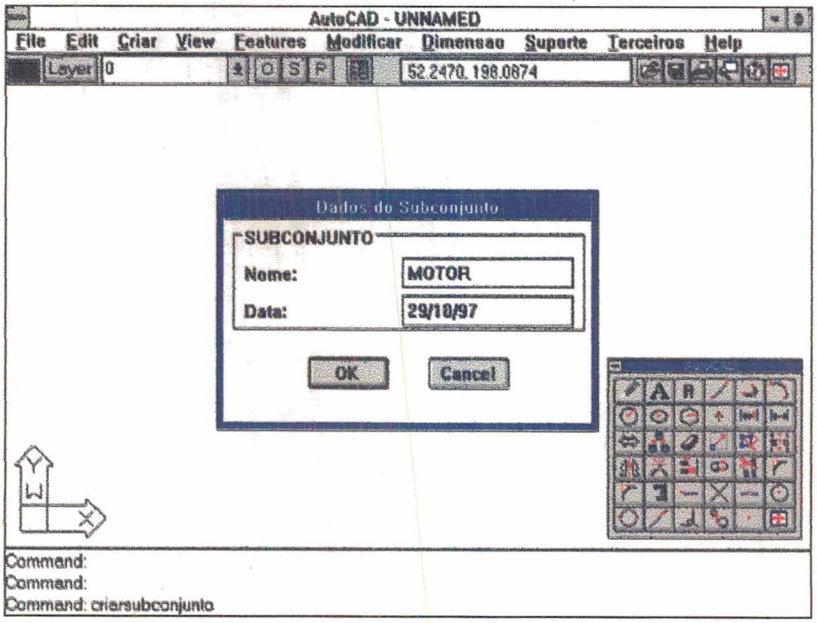


Figura 5.4 - Definição do subconjunto.

Com a definição do subconjunto, o sistema FeatCAD-2D está apto para que se possa criar outros subconjuntos ou mesmo peças. A criação das *features* propriamente ditas que vão formar as peças somente estará à disposição após a definição da primeira peça. É uma condição do sistema para direcionar a inicialização de forma a organizar as informações, criando a estrutura interna do produto.

5.3 - CRIAÇÃO DE UMA PEÇA

A criação de uma peça é feita a partir do nome que lhe é dado, o qual passa a ser o da camada (*layer*) na qual serão representadas as *features* que compõem a peça (a cor da camada é selecionada pelo sistema). Na Figura 5.5, está o quadro de diálogo para a definição da primeira peça do subconjunto MOTOR, chamada de CORPO, em função de ser o componente que dá suporte ao restante das peças que compõem o conjunto MOTOR.

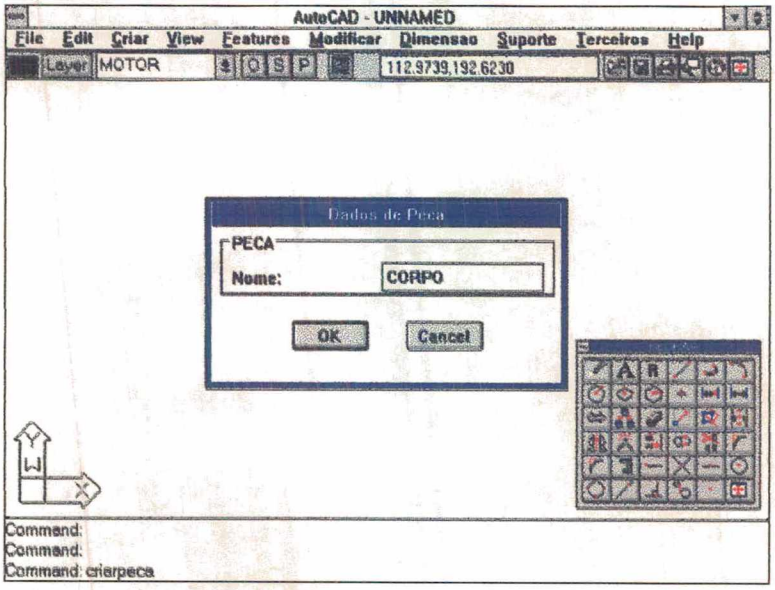


Figura 5.5 - Criação de uma peça.

Após a definição do nome da peça, automaticamente o banco de dados é consultado e apresenta as opções de escolha dos materiais disponíveis. Na Figura 5.6, está representado o quadro de diálogo do banco de dados de materiais, através do qual é possível verificar os materiais existentes por meio da tecla "caracterização", propriedades do material selecionado.

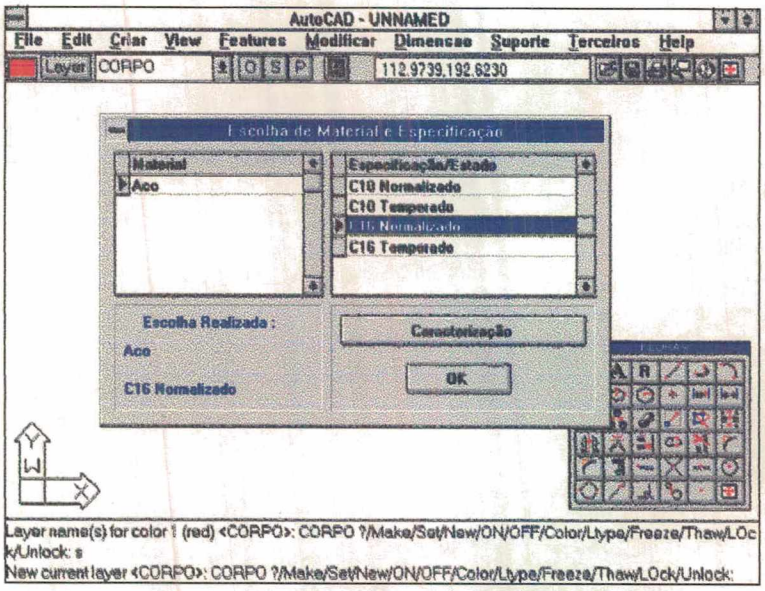


Figura 5.6 - Definição do material da peça.

A consulta às propriedades do material escolhido pode ser observada na Figura 5.7. No quadro de diálogo, são vistas as aplicações usuais para o material escolhido.

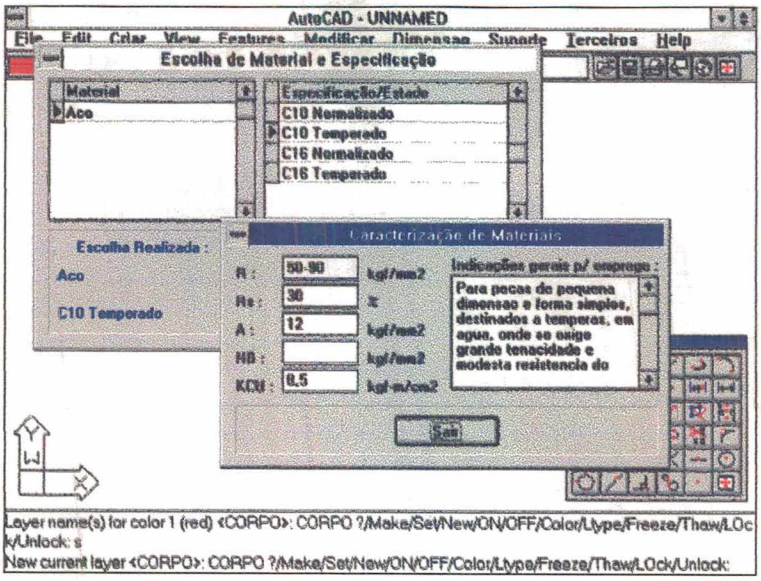


Figura 5.7 - Propriedades do material.

5.4 - INSERÇÃO DAS *FEATURES*

A concretização da criação de uma peça passa pela inserção das *features* que vão compor a peça. Através da biblioteca de *features* disponível no sistema FeatCAD-2D, o usuário pode compor uma peça escolhendo as mais adequadas. Na Figura 5.8, é apresentada a forma de acesso à biblioteca de *features* através do menu principal do sistema, ***Features*** - **Biblioteca**.

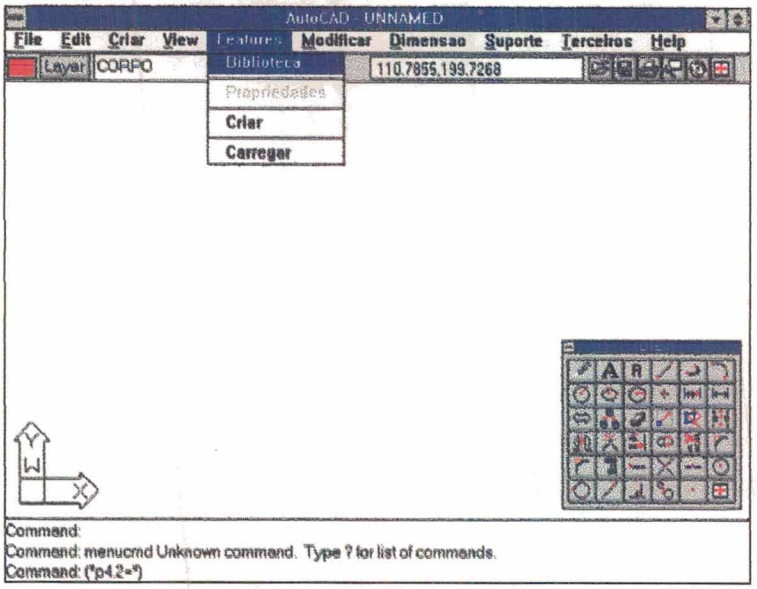


Figura 5.8 - Acesso à biblioteca de *features*.

O quadro de diálogo da biblioteca de *features* à disposição do usuário mostra a *feature* eixo cilíndrico selecionada para a inserção (Figura 5.9).

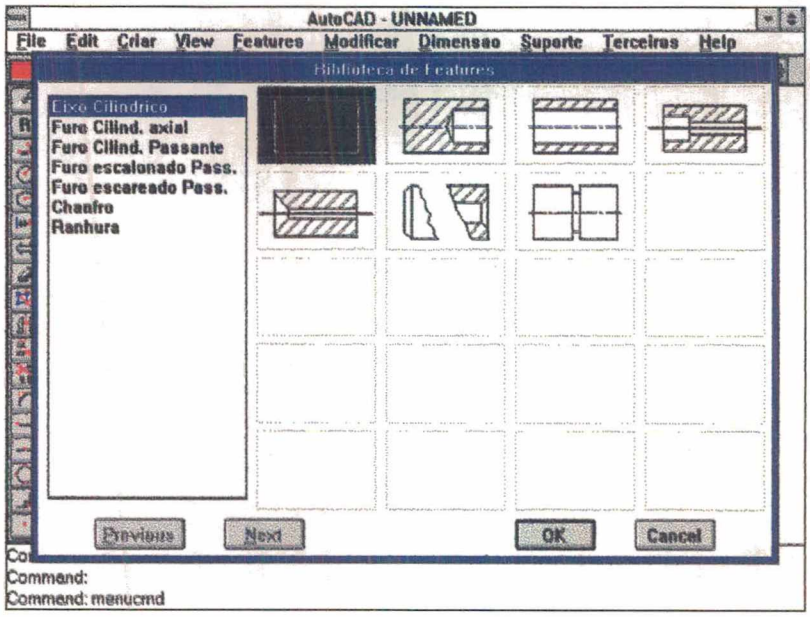


Figura 5.9 - Biblioteca de *features*.

Feita a escolha da *feature*, surge o quadro de diálogo (Figura 5.10) com os parâmetros para a *feature* eixo cilíndrico, no qual se deve informar o Diâmetro, o Comprimento e o Sentido da *feature* quando essa for a primeira da peça.

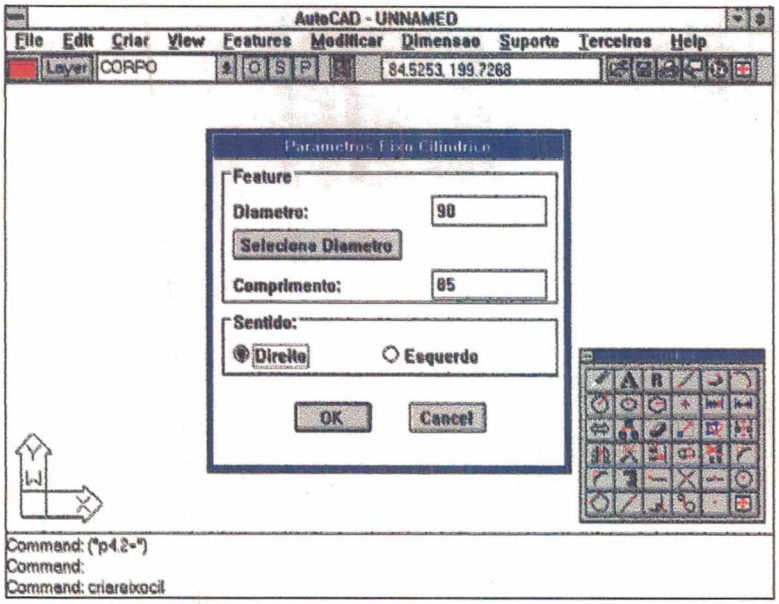


Figura - 5.10 - Definição dos parâmetros da *feature* eixo cilíndrico.

Especificados os parâmetros e acionada a tecla "OK", o sistema solicita o ponto de inserção da *feature* e efetua a representação gráfica, anexando as informações à estrutura de dados do produto. Na Figura 5.11, está a *feature* eixo cilíndrico que foi inserida.

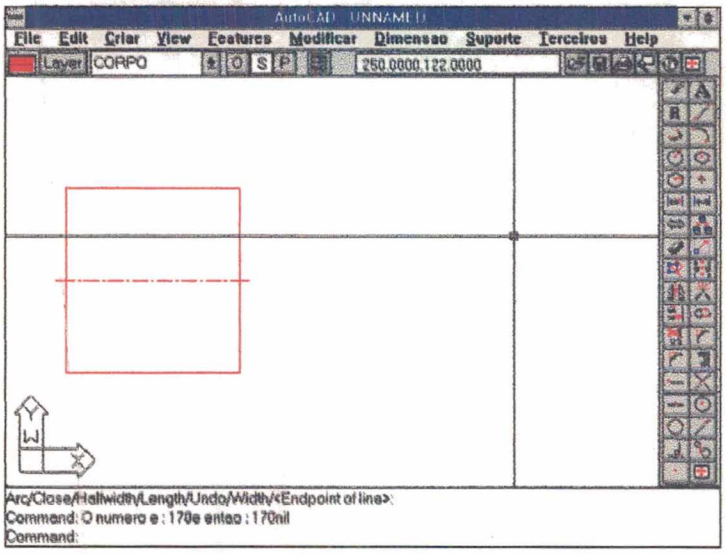


Figura 5.11 - Inserção da *feature* eixo.

A inserção de uma *feature* furo cilíndrico cego à esquerda é realizada na Figura 5.12, onde está o quadro de diálogo solicitando o Diâmetro e a Profundidade. Após a introdução desses parâmetros e sua aceitação, o sistema solicita a localização da *feature*, o que é feito pelo usuário através do *mouse*. Deve-se notar, na Figura 5.13, que a *feature* interna é representada por linhas tracejadas.

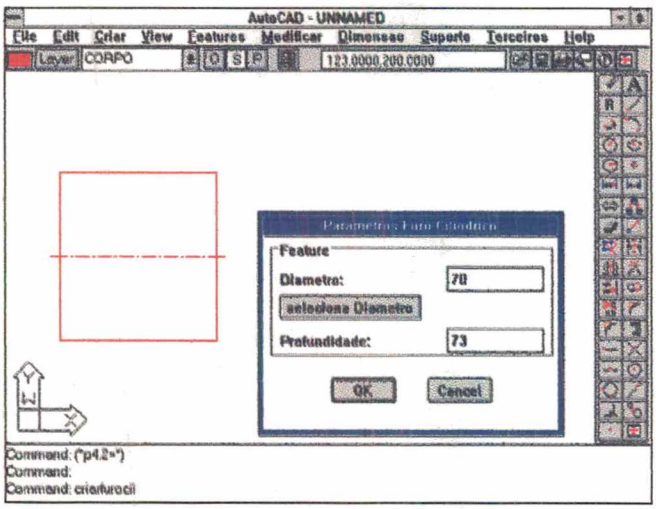


Figura 5.12 - Parâmetros de uma *feature* furo cego.

Para a introdução de uma *feature* furo cilíndrico passante, observa-se a Figura 5.13, em que está o quadro de diálogo solicitando o diâmetro da *feature,* pois é o próprio sistema quem define a profundidade em função do local escolhido para a inserção.

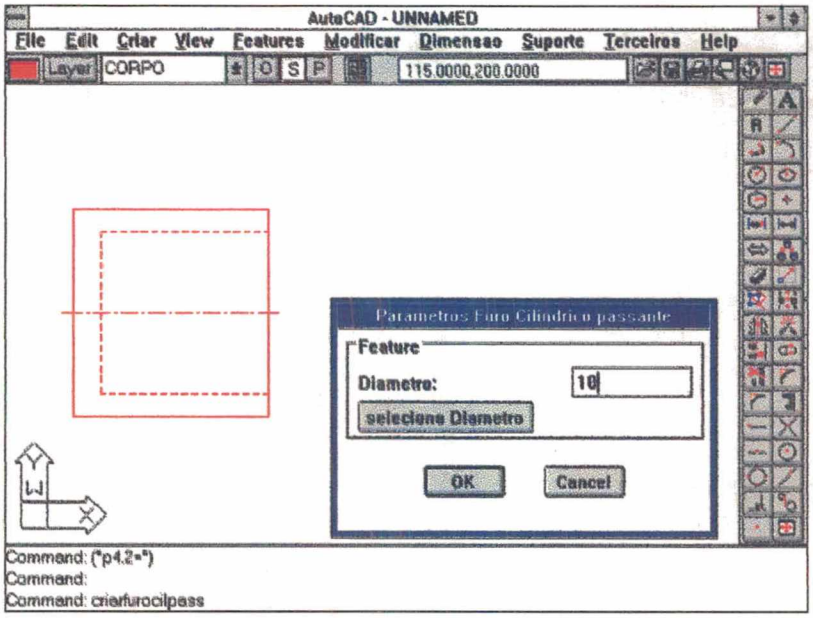


Figura 5.13 - Parâmetros da *feature* furo passante.

Já a partir da introdução da segunda *feature* no sistema (que, no caso, é a *feature* furo cilíndrico cego), o Sistema Especialista passa a atuar de modo a executar a verificação da consistência das informações quando da inserção das *features.*

No caso da inserção da *feature* furo cilíndrico passante e do ponto de localização estar fora da região delimitada pelo eixo cilíndrico (região de material), o Sistema Especialista não permite a inserção desta *feature*, enviando uma mensagem ao usuário e informando o erro (Figura 5.14). O ponto de localização corresponde à cruz à esquerda da peça-CORPO (Figura 5.14).

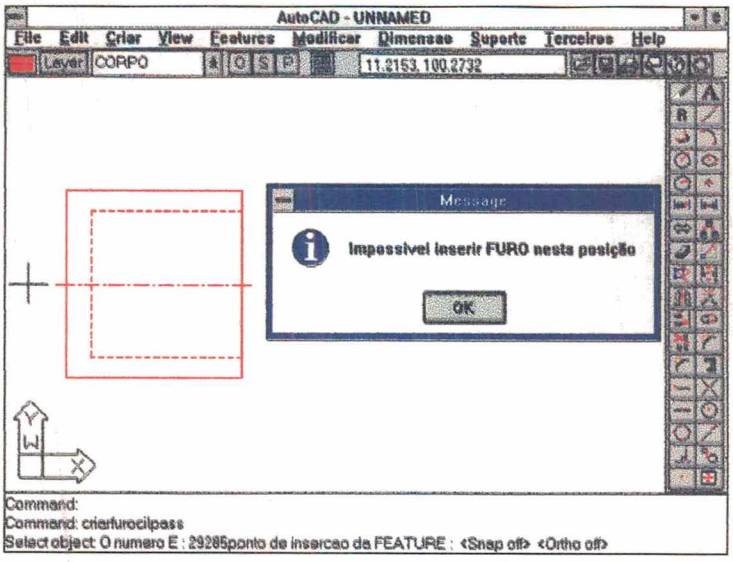


Figura 5.14 - Erro do ponto de localização.

Com a escolha correta do ponto de localização na peça, é possível, então, inserir a *feature* furo cilíndrico passante (diâmetro=10). A peça-CORPO está completa na sua representação gráfica (Figura 5.15), como também na estrutura de dados.

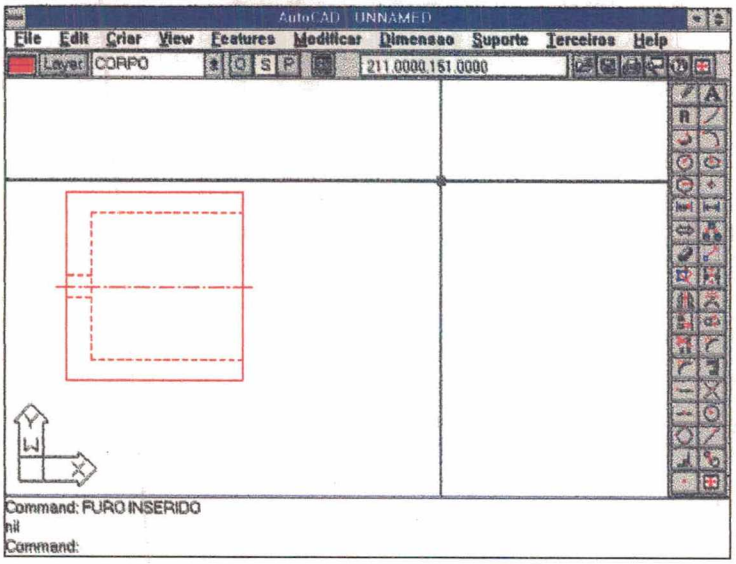


Figura 5.15 - Peça-corpo completa.

Estando a peça-CORPO concluída, é iniciada a peça-TAMPA, composta por duas *features* eixo cilíndrico e uma *feature* furo cilíndrico passante (Figura 5.1).

Na Figura 5.16, já está inserida uma *feature* eixo cilíndrico que compõe a peça-TAMPA, sendo definidos os parâmetros da outra *feature* eixo cilíndrico, que correspondem à feature" que

tem contato com o furo cego da peça- CORPO. A especificação do diâmetro dessa *feature* eixo pode ser feita através da tecla "Seleciona Diâmetro", a qual permite que o diâmetro da *feature* eixo seja definido através da seleção de uma *feature* que já esteja representada, a qual, no caso, é a *feature* furo cilíndrico cego da peça-CORPO, que possui o diâmetro de 70mm. Dessa forma, é definida a parte do encaixe da tampa com o corpo, quando ocorre um contato diametral de montagem e as peças envolvidas devem ter o mesmo diâmetro nominal.

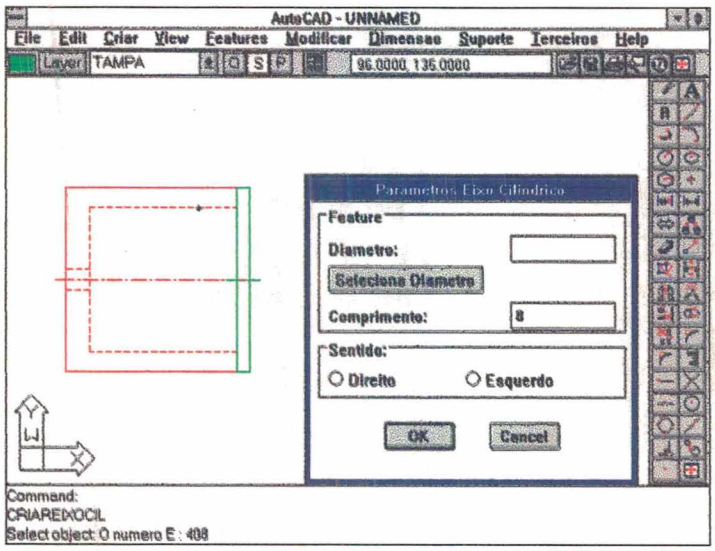


Figura 5.16 - Inserção de *feature* eixo de mesmo diâmetro da *feature* furo cego.

Na Figura 5.17, tem-se o resultado da inserção desta *feature* eixo descrita na Figura 5.16, bem como da *feature* furo cilíndrico passante que será um dos apoios da peça EIXO.

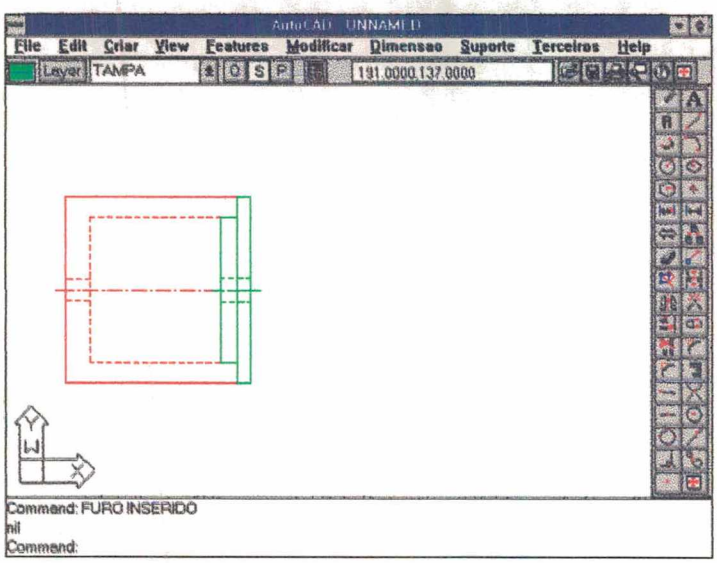


Figura 5.17 - Tampa completa.

A próxima peça a ser inserida é EIXO, o que pode ser visto na Figura 5.18, onde já está inserida uma *feature* eixo cilíndrico à esquerda, a qual foi colocado dentro do furo cilíndrico passante que está no fundo da peça-CORPO.

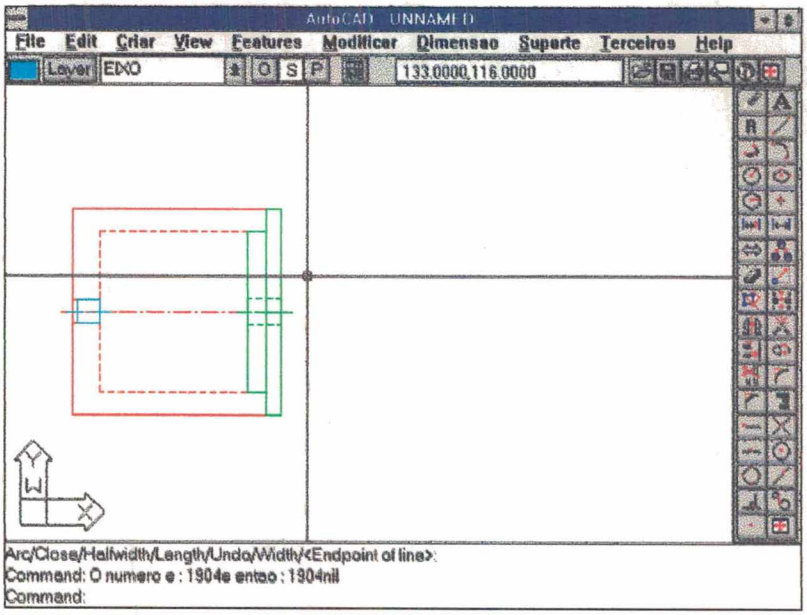


Figura 5.18 - Inserção da primeira *feature* eixo à esquerda da peça eixo.

Na Figura 5.19, está representada a peça-EIXO completa dentro da peça-CORPO.

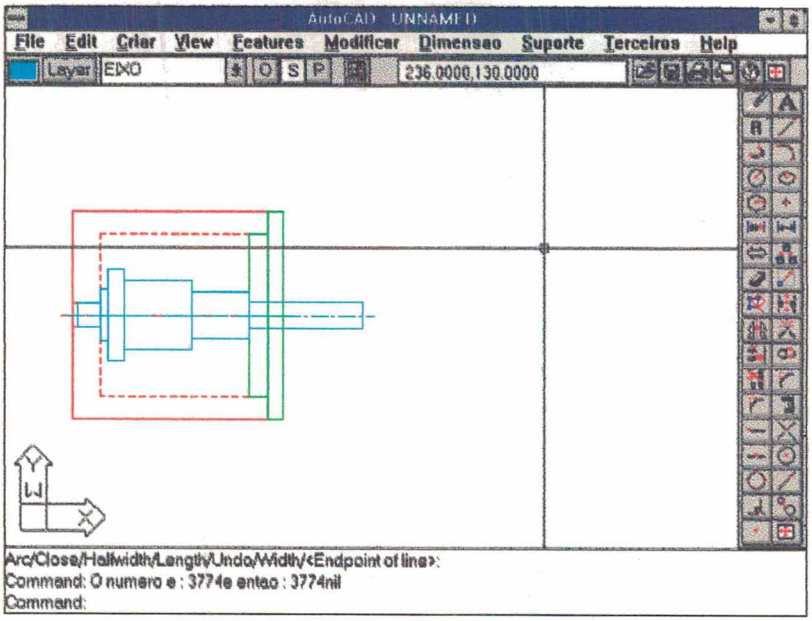


Figura 5.19 - Peça-EIXO completa.

Para inserir a *feature* furo cilíndrico passante que compõe a peça ROTOR, o seu diâmetro pode ser identificado com o uso da tecla "Seleciona Diâmetro" (Figura 5.20), que permite ao usuário identificar a *feature* eixo que tem o mesmo diâmetro que terá a *feature* furo cilíndrico passante, lendo este valor e preenchendo o atributo do quadro de diálogo para a inserção da *feature*.

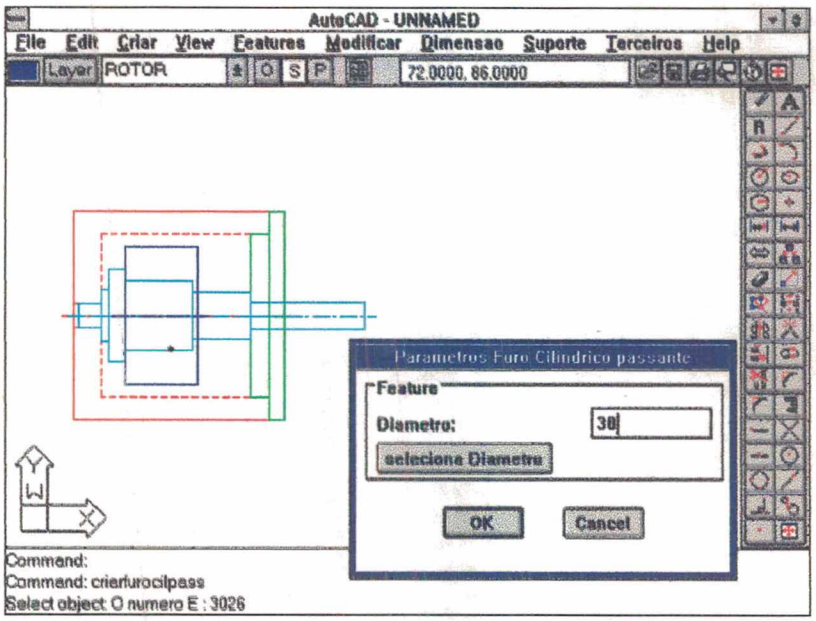


Figura 5.20 - Inserção de uma *feature* furo passante com a seleção do diâmetro.

Na Figura 5.21, está representado o conjunto e a peça-ROTOR completa.

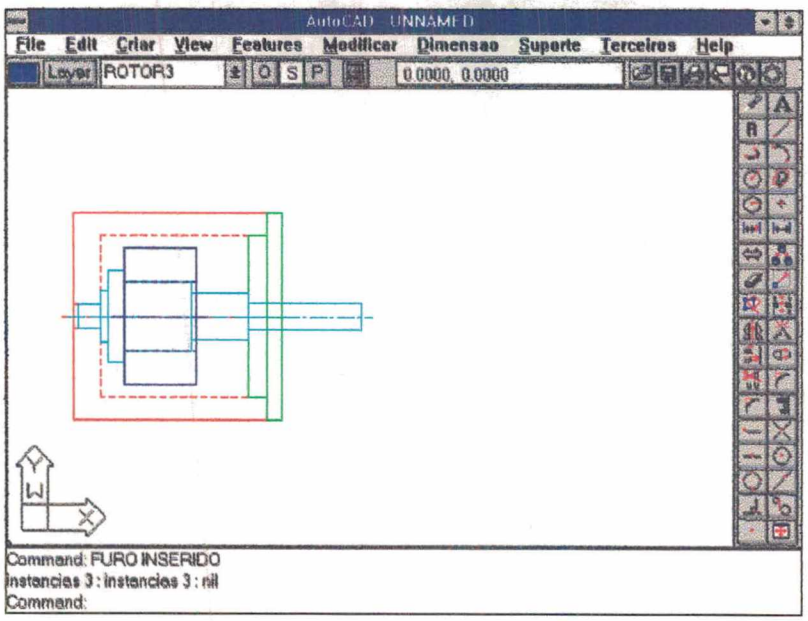


Figura 5.21 - *Feature* furo passante inserida na peça-ROTOR.

Para concluir o conjunto, é inserida a peça-BUCHA (Figura 5.22) com as suas *features*, que são semelhantes às utilizadas na peça-ROTOR.

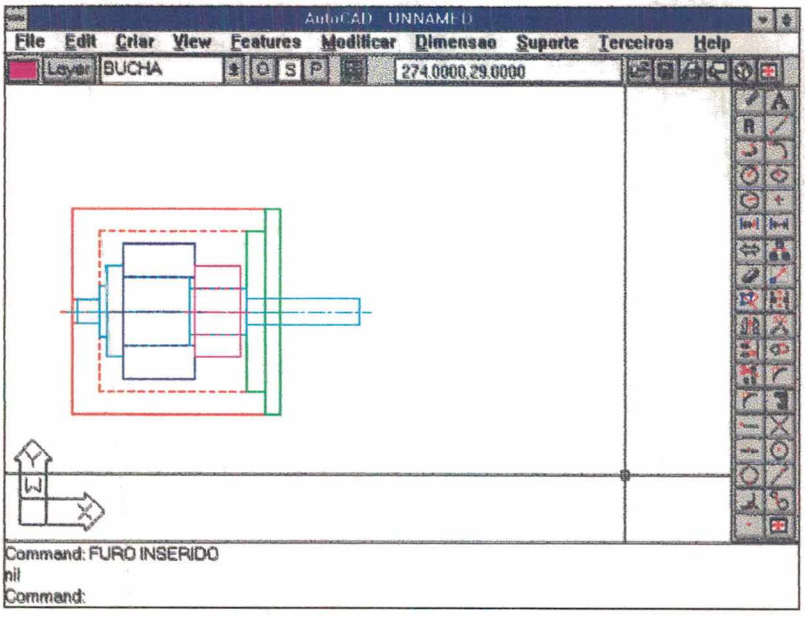
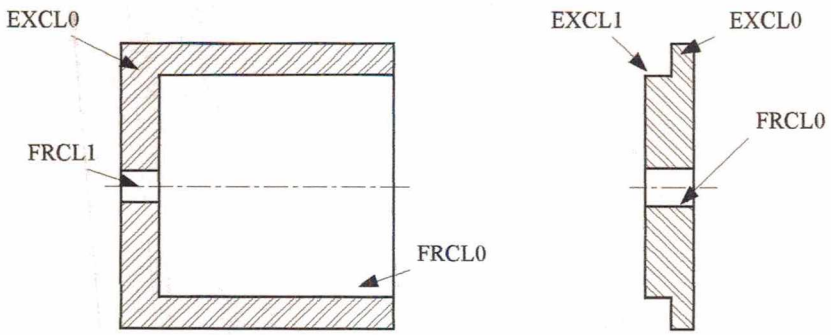


Figura 5.22 - Inserção da peça-BUCHA.

5.5 - DEFINIÇÃO DAS *FEATURES* REPRESENTADAS

As *features* que representam as peças podem ser mais bem visualizadas através de tabelas com os seus atributos. Assim, é feita uma apresentação das informações do conjunto que foram modeladas através do sistema FeatCAD-2D.

A peça-CORPO é composta por uma *feature* eixo cilíndrico EXCL0, uma *feature* furo cilíndrico cego FRCL0 e uma *feature* furo cilíndrico passante FRCL1 (Figura 5.23(a)). Na Figura 5.23(b), são apresentadas as *features* que compõem a peça-TAMPA: uma *feature* eixo cilíndrico EXCL0, uma EXCL1 e uma *feature* furo cilíndrico passante FRCL0.



(a) *features* que compõem o corpo

(b) *features* que compõem a tampa

Figura 5.23 - Descrição das *features* que compõem as peças CORPO e TAMPA.

Nas Tabelas 5.1 e 5.2, são apresentados os atributos das *features* que compõem as peças da Figura 5.23.

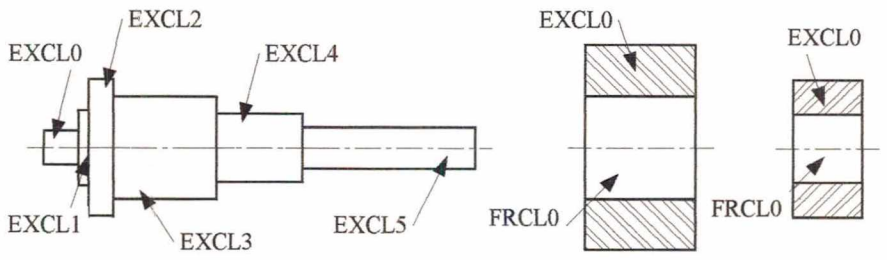
Tabela 5.1 - Atributos das *features* da peça-CORPO.

Peça : CORPO			
	features		
Nome	EXCL0	FRCL0	FRCL1
Tipo	eixocilíndrico	furocilíndricocego	furocilíndricopassante
Posição	externo	interno	interno
Direção	axial	axial	axial
Sentido	1	-1	-1
CoordY	100	100	100
CoordI	30	115	42
CoordF	115	42	30
Comp/Prof	85	73	12
Diâmetro	90	70	10
Volume	1	-1	-1

Tabela 5.2 - Atributos das *features* da peça-TAMPA.

Peça : TAMPA			
	features		
Nome	EXCL0	EXCL1	FRCL1
Tipo	eixocilíndrico	eixocilíndrico	furocilíndricopassante
Posição	externo	externo	interno
Direção	axial	axial	axial
Sentido	1	-1	1
CoordY	100	100	100
CoordI	115	115	107
CoordF	122	107	122
Comp/Prof	7	8	15
Diâmetro	90	70	12
Volume	1	1	-1

Na Figura 5.24, estão representadas as peças *eixo*, *rotor* e *bucha*, com as respectivas *features* indicadas.



(a) eixo (b) rotor (c) bucha

Figura 5.24 - Indicação das *features* que compõem as peças eixo, rotor e bucha.

Nas tabelas 5.3 e 5.4, estão representados os atributos das peças *eixo*, *rotor* e *bucha*, respectivamente.

Tabela 5.3 - Atributos das *features* da peça-EIXO.

Peça : EIXO						
	features					
Nome	EXCL0	EXCL1	EXCL2	EXCL3	EXCL4	EXCL5
Tipo	eixocil	eixocil	eixocil	eixocil	eixocil	eixocil
Posição	externo	externo	externo	externo	externo	externo
Direção	axial	axial	axial	axial	axial	axial
Sentido	-1	1	1	1	1	1
CoordY	100	100	100	100	100	100
CoordI	32	42	45	52	82	107
CoordF	42	45	52	82	107	157
Comp/Prof	10	3	7	30	25	50
Diâmetro	10	22	40	30	20	12
Volume	1	1	1	1	1	1

Tabela 5.4 - Atributos das *features* das peças ROTOR e BUCHA.

Peças : ROTOR e BUCHA				
	*features*no rotor		*features*na bucha	
Nome	EXCL0	FRCL0	EXCL0	FRCL0
Tipo	eixocilíndrico	furocilíndricopas	eixocilíndrico	furocilíndricopas
Posição	externo	interno	externo	interno
Direção	axial	axial	axial	axial
Sentido	1	1	1	1
CoordY	100	100	100	100
CoordI	52	52	84	84
CoordF	844	84	104	104
Comp/Prof	32	32	20	20
Diâmetro	60	30	40	20
Volume	1	-1	1	-1

5.6 - PROPRIEDADES

Com a utilização do comando **propriedades**, é possível visualizar informações a respeito das *features* representadas, que tanto podem corresponder a uma peça como a uma *feature* que compõe uma peça. Para verificar as propriedades de uma determinada peça, basta solicitar através da seqüência **Criar - Peça - Propriedades**, do menu principal.

Abre-se, então, um quadro de diálogo correspondente à peça que possui a camada (*layer*) ativa, que, no caso da Figura 5.25, é a peça-EIXO onde o quadro de diálogo apresenta o subconjunto a que pertence, o nome da peça, tipo de fabricação, material, especificação do material e comprimento total da peça.

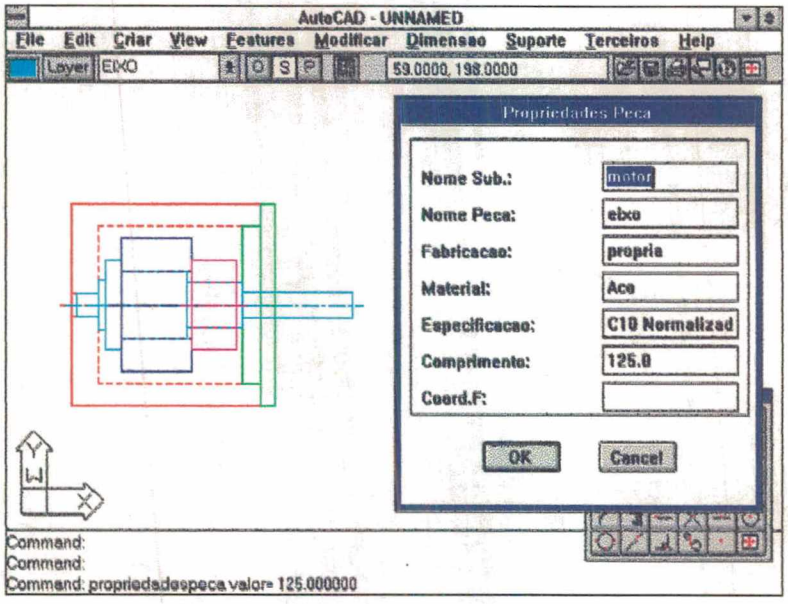


Figura 5.25 - Propriedades da peça-EIXO.

Na Figura 5.26, estão as características de uma *feature* que pertence à peça-TAMPA. Para visualizar as características de uma *feature*, é necessário solicitar os comandos **Features - Propriedades**, selecionando a *feature* desejada, independentemente da camada da peça que está ativa.

No quadro de diálogo (Figura 5.26), as informações apresentadas são o Ajuste, Afastamento Inferior e Superior, que se encontram em branco ou com valores nulos porque ainda não foi realizada a análise com respeito à montagem e à escolha das tolerâncias. As outras

informações dizem respeito à modelagem das informações na estrutura de dados com relação à geometria.

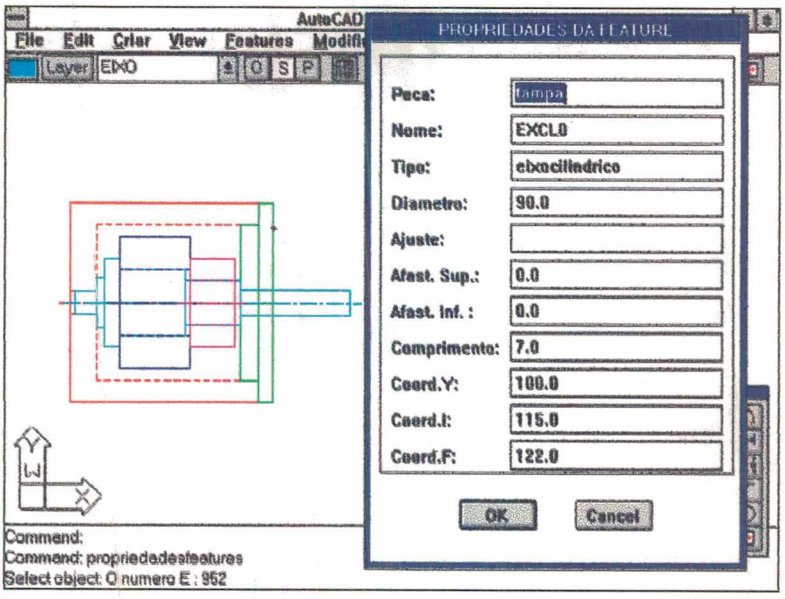


Figura 5.26- Propriedades de uma *feature*.

5.7 - ANÁLISE DO CONJUNTO MONTADO

A análise do conjunto somente pode ser realizada a partir do momento em que o mesmo está pronto. Assim, pode-se solicitar a função no menu principal **Suporte - Montagem**, que inicia a análise das peças que compõem o conjunto.

No momento em que o sistema identifica um acoplamento eixo/furo que possui diâmetros nominais iguais, uma linha de cota é colocada para identificar a dimensão que está sendo analisada (Figura 5.27), apresentando um quadro de diálogo que permite a escolha da forma como serão determinadas as tolerâncias, ou seja, pelo Ajuste ou pela Aplicação (Figura 5.28).

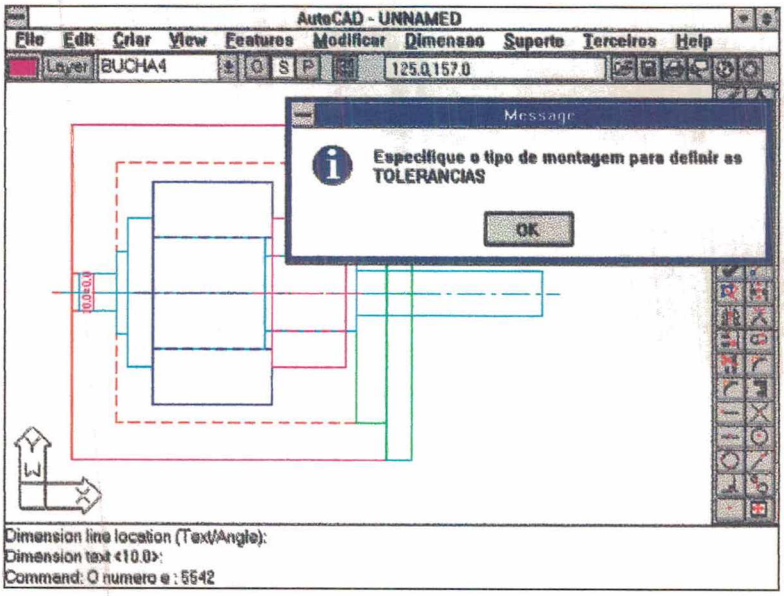


Figura 5.27 - Identificação de um acoplamento eixo/furo.

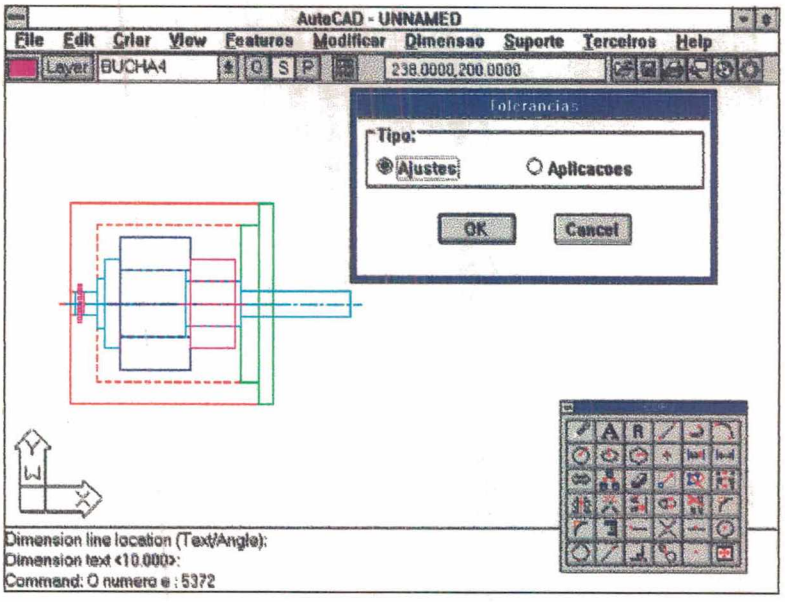


Figura 5.28 - Escolha do modo de especificar as tolerâncias.

O usuário escolhe uma das opções de tolerâncias, e o sistema abre um quadro de diálogo que serve de interface do banco de dados para a escolha da tolerância mais adequada, o que é feito interativamente pelo usuário. Após a escolha das tolerâncias, o sistema providencia a retirada da linha de cota.

5.8 - CONSULTA AO BANCO DE DADOS DE TOLERÂNCIAS

A implementação do banco de dados de tolerâncias foi dividida em módulos de acordo com o tipo de abordagem como definido anteriormente (item 4.16.2). Assim, tem-se mais explicitamente:

Caso1: Quando a procura se faz pelos ajustes, além da escolha desse, também é necessário especificar a precisão (Figura 5.29), do que resultam os pares de ajuste recomendados, bem como exemplos de aplicação para reforçar a decisão.

Na Figura 5.29, está representada a caracterização do ajuste escolhido, que serve de auxílio ao projetista para definir melhor o problema. No caso, o ajuste escolhido é Folga Rotativo Forte. Do mesmo modo, na Figura 5.30, pode-se ver a caracterização da precisão escolhida, que, no caso, é Extrapreciso.

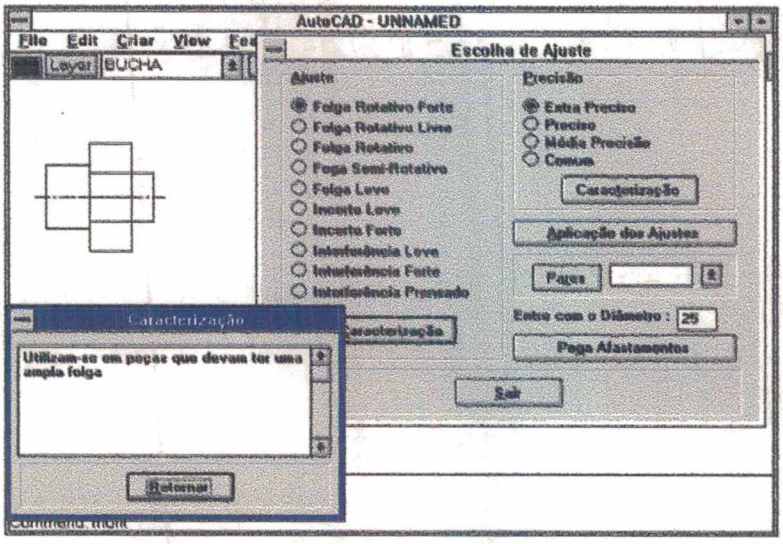


Figura 5.29 - Caracterização do ajuste.

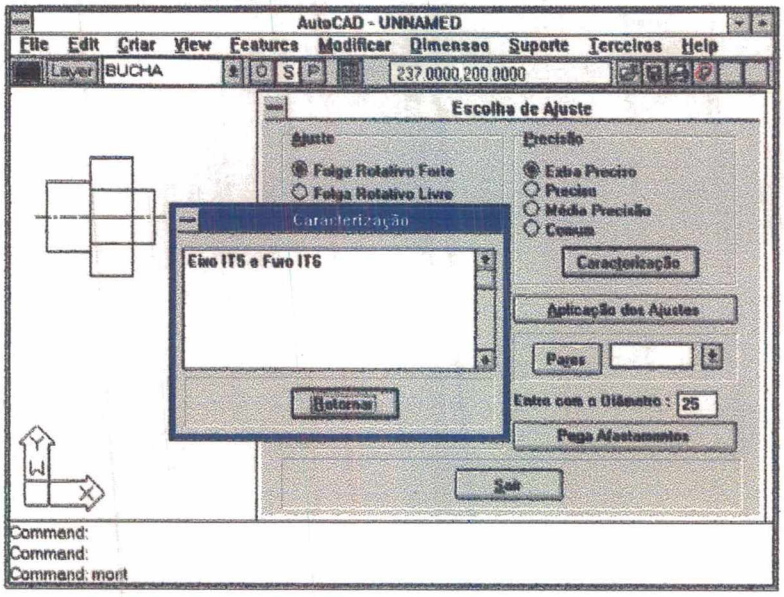


Figura 5.30 - Caracterização da precisão.

Com a escolha do ajuste e precisão, pode-se verificar os tipos de aplicações em que são usados (Figura 5.31).

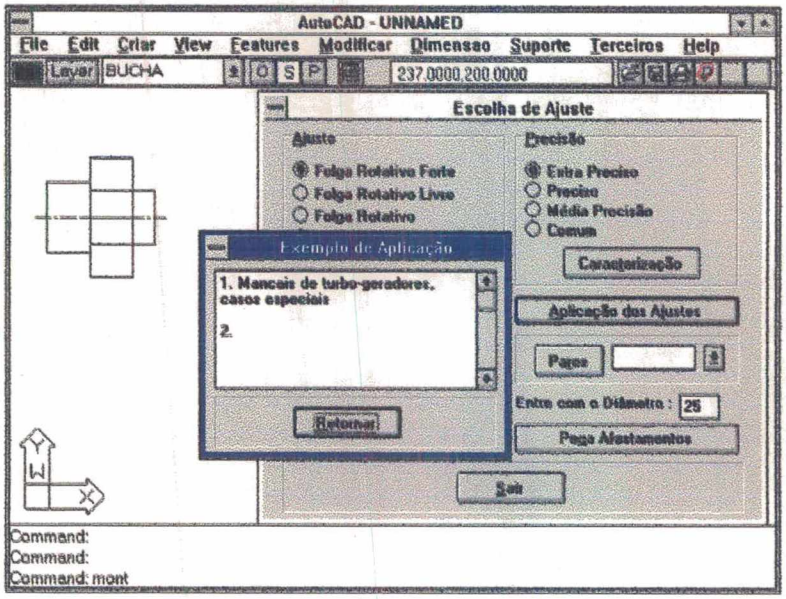


Figura 5.31 - Exemplos de aplicação.

Na Figura 5.32, pode-se observar que o par de acoplamento já está escolhido, clicando a tecla Pares, o que resulta numa série de pares que podem ser utilizados. Como o diâmetro está especificado, basta utilizar a tecla Pega Afastamentos para ter as informações finais. Quando da utilização da função de análise da montagem, em que o sistema identifica automaticamente os

acoplamentos, o valor do diâmetro é passado para o banco de dados, pois ele serve para identificar os valores dos afastamentos. O diâmetro nominal a ser tolerado deve ser especificado, para que o sistema possa retornar os afastamentos.

Ainda na Figura 5.32, pode-se observar na janela à esquerda o resultado da consulta, os ajustes, bem como os valores dos afastamentos para o furo e o eixo.

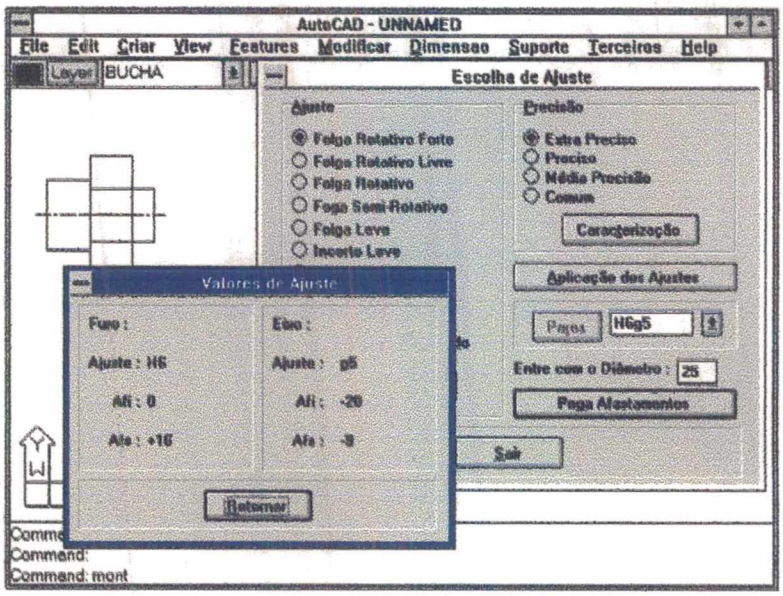


Figura 5.32 - Informações resultantes do banco de dados.

Caso 2: Através de exemplos de aplicações conhecidas de pares de acoplamento, são escolhidos aqueles pares que mais se assemelham ao caso em estudo. Escolhe-se a aplicação e informa-se o diâmetro nominal (que, dependendo do tipo de consulta é automaticamente fornecido pelo sistema CAD). O banco de dados fornece o par de ajustes, a qualidade de trabalho, as tolerâncias e os afastamentos para a dimensão nominal; informa também quanto à folga ou interferência. Os afastamentos são pesquisados no banco de dados específico.

Na Figura 5.33, está a interface com o banco de dados pelo CAD. Na janela superior, está o exemplo escolhido e, abaixo, estão representadas as características do referido acoplamento. Ao sair da aplicação, nesse caso, as informações serão transferidas para a estrutura de dados do sistema FeatCAD-2D.

As informações representadas na janela devem servir de base para a decisão do projetista, que deve escolher com base em seu conhecimento.

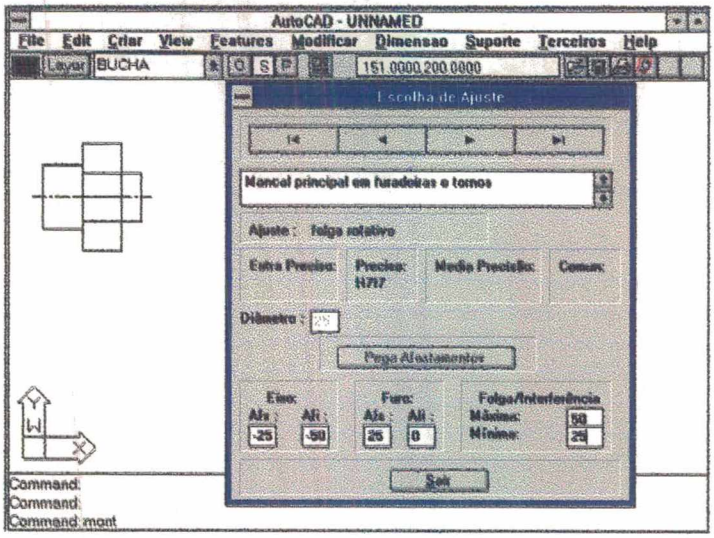


Figura 5.33 - Tolerâncias por aplicação.

Caso 3: Considera aquelas peças fornecidas por terceiros, as quais já possuem dimensões toleradas, e o sistema deve identificar a peça, buscando automaticamente, na tabela referente ao produto, qual das tolerâncias deverá ser utilizada em função de determinadas informações específicas que o sistema deve conhecer.

Esse caso é de simples uso, entretanto sua modelagem se torna mais complexa, pois, para cada componente (p. ex., rolamentos, ver Figura 5.34), deve ser gerado um banco de dados específico, inclusive com interface adequada para receber as informações específicas do componente.

Quando da especificação automática das tolerâncias e ajustes, o sistema segue a seqüência de conexão entre as peças a serem montadas, facilitando, assim, o entendimento do conjunto. Tal especificação deve ser iniciada pelas peças nas quais as tolerâncias e ajustes são conhecidos, como é o caso das peças padronizadas.

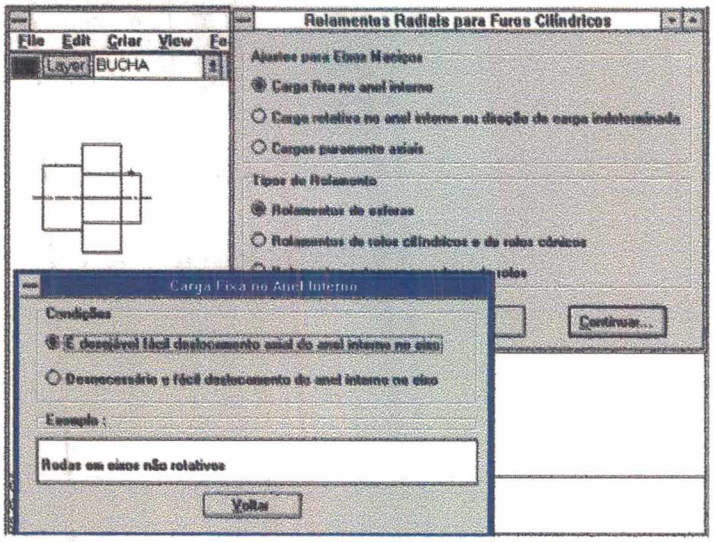


Figura 5.34 - Escolha de tolerâncias pelo produto.

5.9 - IDENTIFICAÇÃO DOS CONTATOS DIAMETRAL E AXIAL

As ligações correspondem à identificação dos contatos que ocorrem no conjunto de peças montadas que representam o conjunto. A identificação desses contatos, diametral/axial, é resultante da análise do conjunto.

As informações sobre os contatos são apresentadas ao usuário como mostra a Figura 5.35. Através da seqüência de comandos **Suporte-Ligações** no menu principal, têm-se os atributos que representam um contato axial do conjunto desenhado ao lado. Na Figura 5.36, é apresentado um quadro de diálogo com os atributos do contato diametral do mesmo conjunto.

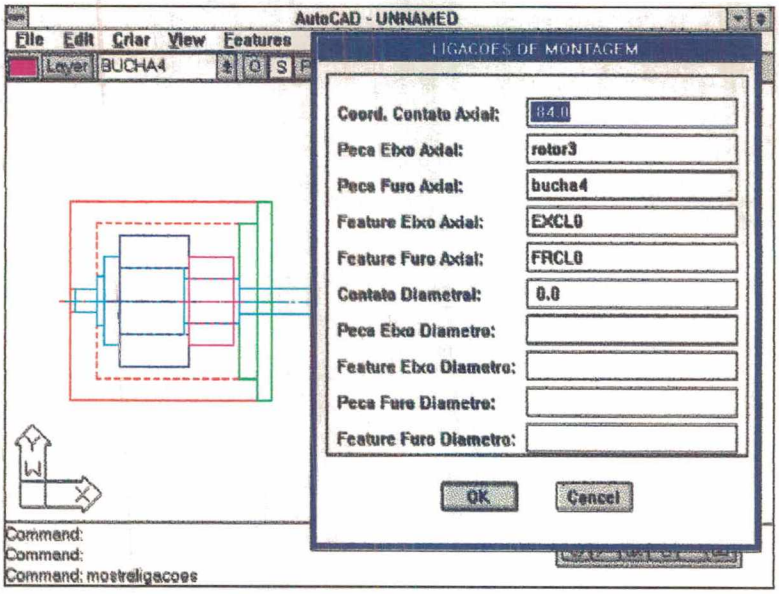


Figura 5.35 - Ligações de contato axial.

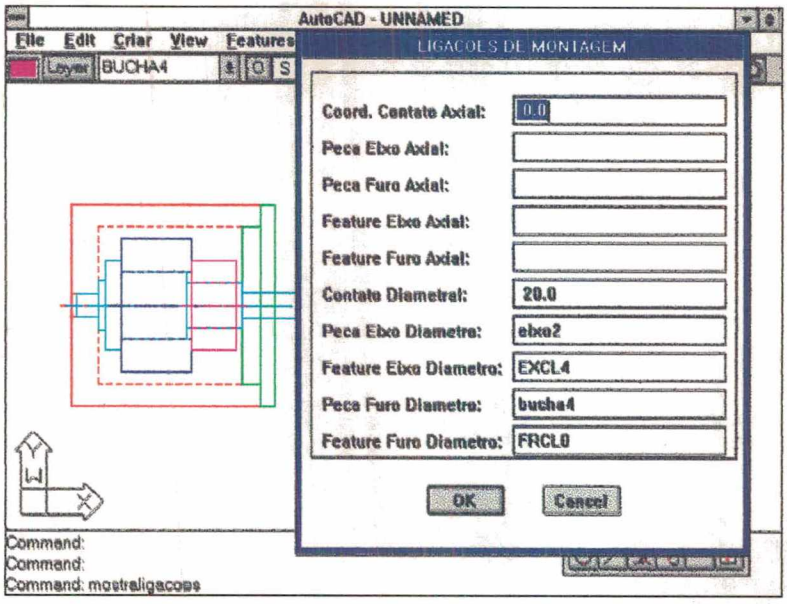


Figura 5.36 - Ligações de contato diametral.

5.10 - DESCRIÇÃO DOS CONTATOS

Na Figura 5.37, são indicados os diâmetros que possuem contato diametral para contagem, dados que estão registrados na Tabela 5.5 em forma de atributos da classe Ligações e que são visualizados pelo usuário no formato apresentado nas Figuras 5.35 e 5.36.

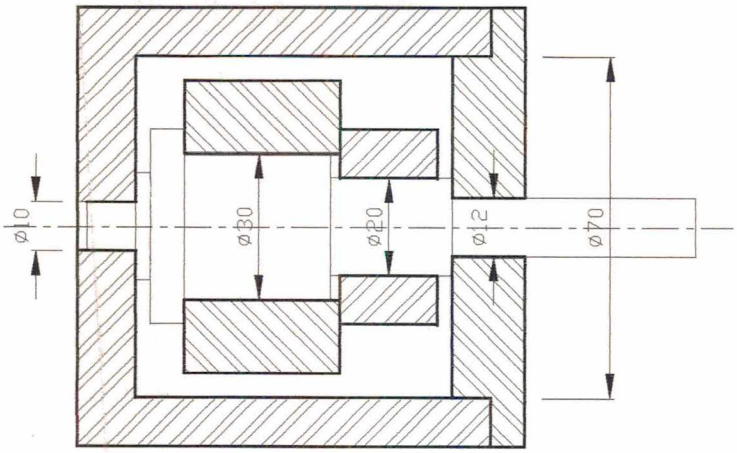


Figura 5.37 - Contatos diametrais.

Tabela 5.5 - Contatos diametrais.

Contatos diametrais				
Diâmetro_ Contato	Peça_Eixo_ Diâmetro	Feature_Eixo_ Diâmetro	Peça_Furo_ Diâmetro	Feature_Furo_ Diâmetro
10	eixo	EXCL0	corpo	FRCL1
30	eixo	EXCL3	rotor	FRCL0
20	eixo	EXCL4	bucha	FRCL0
12	eixo	EXCL5	tampa	FRCL0
70	tampa	EXCL1	corpo	FRCL0

A partir da observação do desenho do conjunto (Figuras 5.1, e 5.22) e das Tabelas 5.1, 5.2, 5.3 e 5.4, pode-se observar que as faces de contato indicadas na Figura 5.38 correspondem às coordenadas CoordI ou CoordF das peças que possuem valores iguais, caracterizando o contato axial entre as peças. Tais contatos axiais são ilustrados na Tabela 5.6.

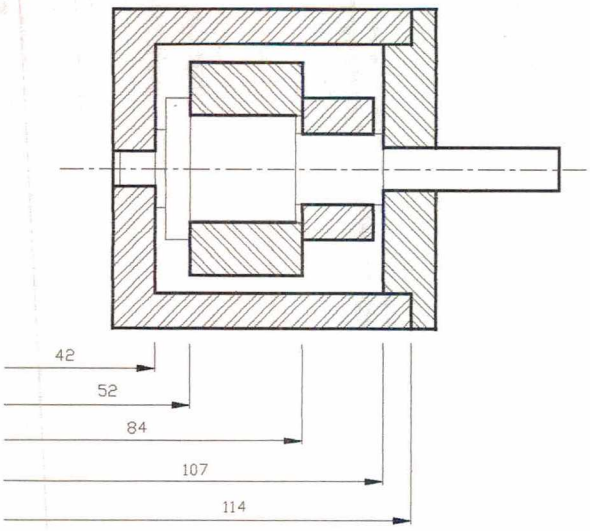


Figura 5.38 - Coordenadas dos contatos axiais das peças no conjunto.

Tabela 5.6 - Contatos axiais.

Contatos axiais				
Coordenada_ Contato	Peça_Eixo_ Axial	Feature_Eixo_ Axial	Peça_Furo_ Axial	Feature_Furo_ Axial
42	eixo	EXCL1	corpo	FRCL1
52	eixo	EXCL2	rotor	FRCL0
84	bucha	EXCL0	rotor	FRCL0
107	eixo	EXCL4	tampa	FRCL0
114	tampa	EXCL1	corpo	FRCL0

5.11 - COTAGEM

A cotagem segue os conceitos descritos no item 3.9 e implementados no item 4.14. A cotagem é feita tanto automaticamente como manualmente através de funções disponíveis no menu principal. Neste trabalho, somente foi feita a cotagem de *features* básicas externas (eixos).

Através do comando **Dimensão**, surgem as opções de cotar um conjunto, um subconjunto ou peça. Dentro da opção dimensionar peça, é possível navegar entre as camadas de dimensão das diversas peças em que se deseja efetuar alguma alteração.

A cotagem automática pode ser feita tanto para as cotas diametrais como para as logitudinais; o mesmo para a opção de dimensionamento manual, o que possibilita que as cotas sejam introduzidas diretamente pelo usuário através de função específica do sistema FeatCAD-2D (Figura 5.39).

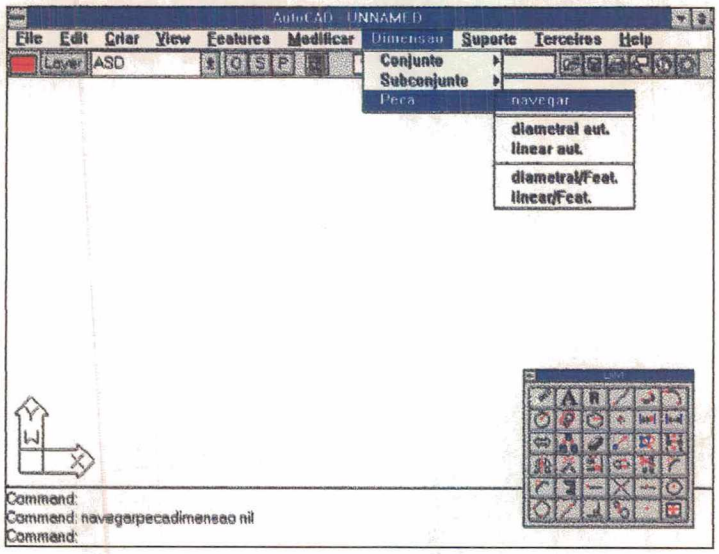


Figura 5.39 - Opções para a cotagem .

Com a escolha da peça a ser cotada, a camada à qual essa pertence fica ativa, e o sistema, automaticamente, busca a camada dimensão, efetuando o cotagem e a distribuição das cotas de acordo com regras existentes no módulo de cotagem do sistema especialista.

Na Figura 5.40, está representado o conjunto-MOTOR, sendo que a peça-EIXO está selecionada para executar a cotagem automática conforme as regras representadas na base de conhecimento.

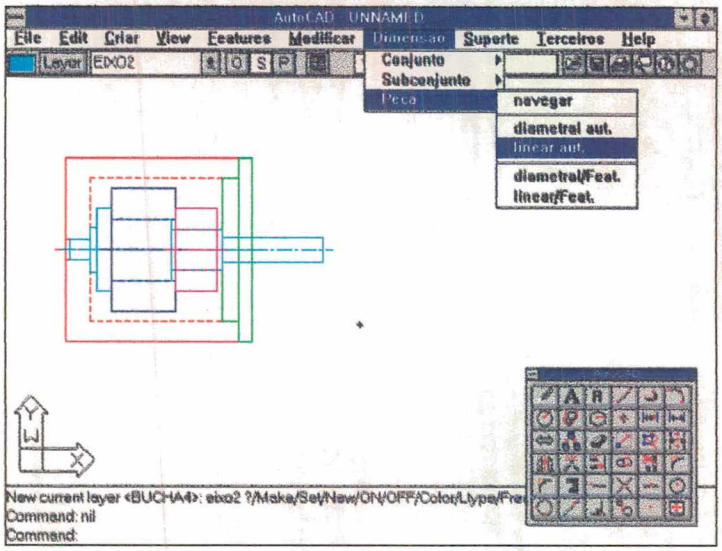


Figura 5.40 - Menu principal para a cotagem.

Na Figura 5.41, está representada a peça-EIXO com a cotagem longitudinal automática representada através das cotas.

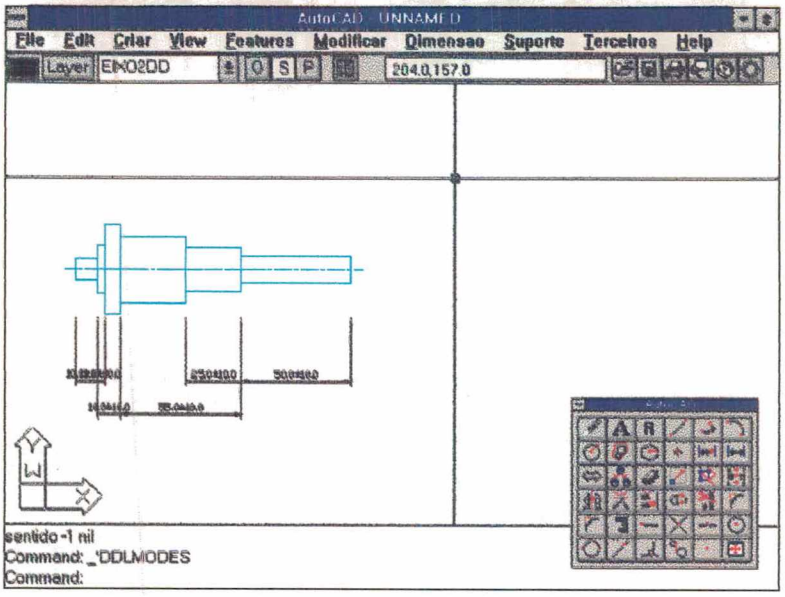


Figura 5.41 - Peça-EIXO cotada.

Na Figura 5.42, está sendo escolhida a cotagem diametral automática.

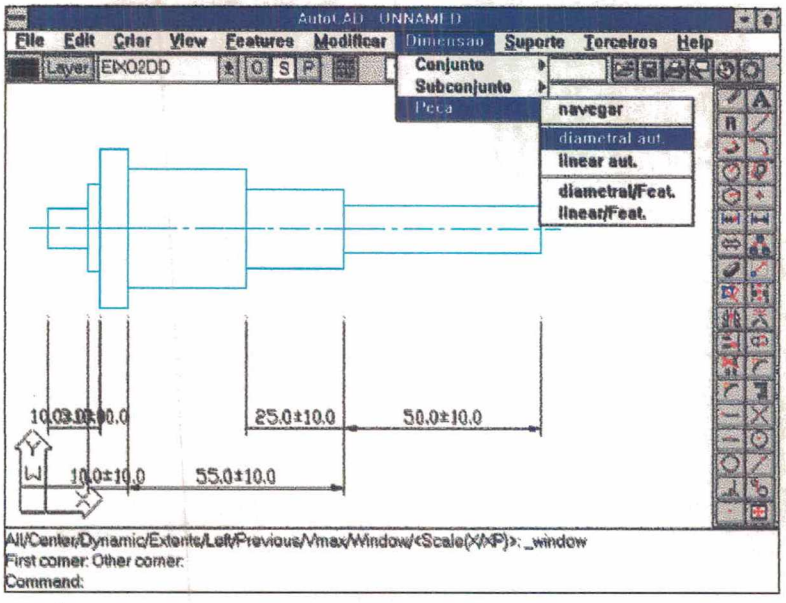


Figura 5.42 - Escolha da opção de cotagem diametral automática.

A cotagem diametral é representada através das cotas que podem ser observadas na Figura 5.43, bem como as tolerâncias escolhidas quando da identificação automática da montagem, as quais também foram escolhidas pelo usuário.

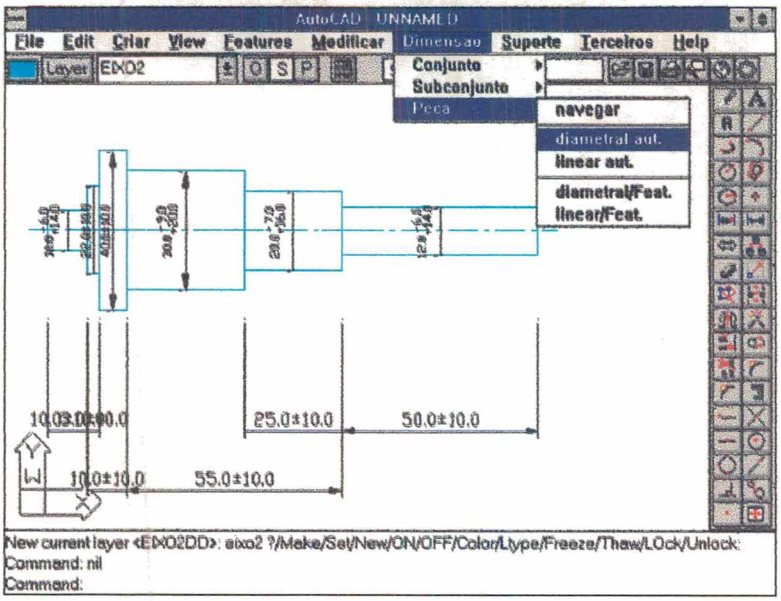


Figura 5.43 - Representação da cotagem longitudinal e diametral.

As coordenadas de referência utilizadas para a cotagem representada na Figura 5.42 são especificadas na Figura 5.44, referências que são específicas para a peça-EIXO.

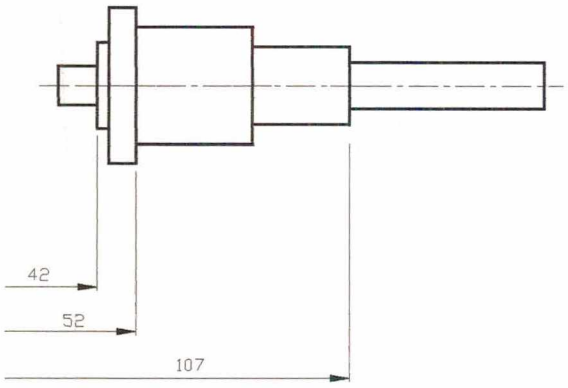


Figura 5.44 - Coordenadas de referência para a cotagem.

5.12 - *FEATURES* CONFIGURÁVEIS

As *features* configuráveis, como foi explicitado no capítulo 3, podem ser definidas pelo usuário através da programação ou pela interface gráfica. Através da interface gráfica, basta ter uma determinada peça desenhada e solicitar a função **Criar** (Figura 5.45), que está dentro da função **Biblioteca**.

Uma *feature* configurável será aquela que está com a camada ativa, a qual receberá este nome ao ser criada. Quando da criação, é necessário definir um ponto de inserção, o qual será utilizado quando da operação de **Carregar** uma *feature* (Figura 5.45).

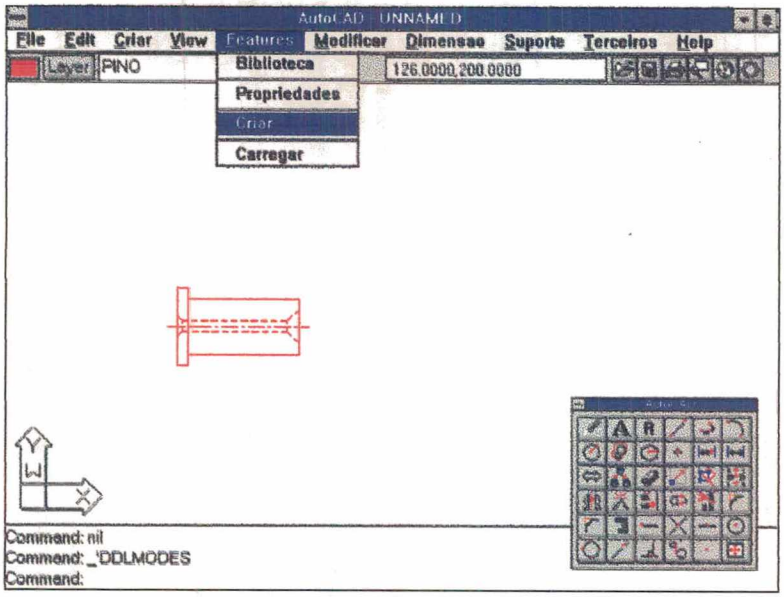


Figura 5.45 - Criando uma *feature* configurável PINO.

Na Figura 5.46, mostra-se o quadro de diálogo responsável pela chamada da *feature* configurável criada anteriormente.

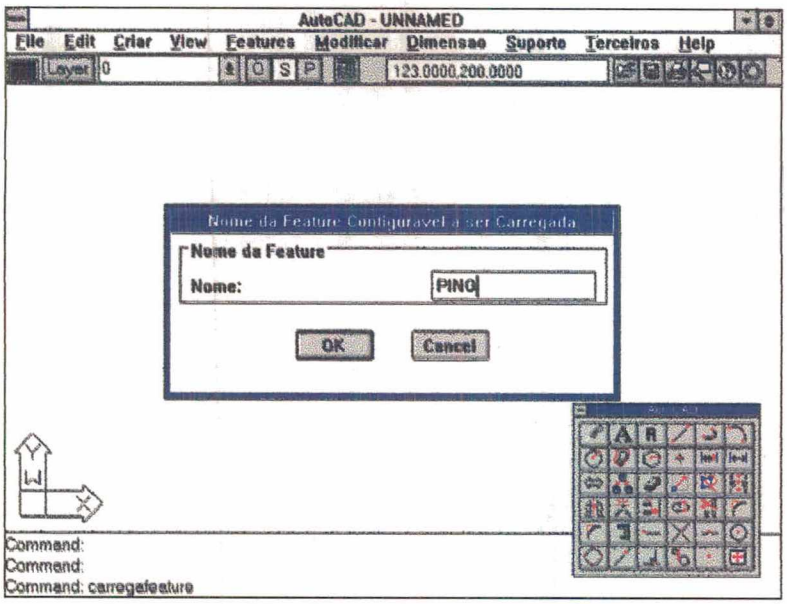


Figura 5.46 - Carregando uma *feature* configurável.

CAPÍTULO 6

DISCUSSÃO, CONCLUSÕES E FUTUROS TRABALHOS

6.1 - DISCUSSÃO

No presente trabalho, implementou-se uma abordagem de modelagem do projeto, focalizando a introdução das informações na fase do projeto detalhado, a partir do qual o usuário efetua de forma interativa a análise e obtém informações para a modelagem do produto, utilizando-as na manufatura e montagem.

Para isso, foi necessário o desenvolvimento de uma interface gráfica, que é baseada em *features* e tem como plataforma um CAD comercial (AutoCAD), o qual, a partir de suas funções básicas e algumas adaptações, permite efetuar a modelagem das informações.

Foi definida uma biblioteca de *features* básicas e modificadoras que permitiu a realização da modelagem de peças. A partir dessas *features* definidas na biblioteca, também foi possível modelar as chamadas *features* combinadas, resultantes da combinação das *features* disponíveis na biblioteca, como, por exemplo, criar uma *feature* furo escalonado passante, o que é realizado através do uso de funções de programação específicas que foram desenvolvidas. Com essas funções, foi possível modelar as chamadas *features* padronizadas e padrão; as padronizadas podem ser utilizadas para a representação de peças, como parafusos, arruelas, rolamentos, etc.; já as *features* padrão são para o uso interno da empresa, para suas peças padronizadas.

A modelagem de *features* padrão foi desenvolvida de modo a permitir a sua criação de duas formas: na primeira, através da programação, como a utilizada nas *features* combinadas, é necessário que o usuário entre no sistema e as defina, para o que deve compilar todo o programa, necessitando de alguém que entenda de programação; na segunda forma, o usuário pode implementar essas *features* através da interface gráfica e, depois, simplesmente introduzi-las no projeto.

A necessidade de representar um produto obrigou à construção de uma estrutura de dados independente do sistema CAD, com a definição de um conjunto, subconjunto, peça e *features*. Com a utilização dessa estrutura de dados independente, foi possível representar todas as informações em termos de *features* básicas e modificadoras, o que permite que uma *feature* padrão possa ser alterada depois que foi inserida, pois o sistema FeatCAD-2D permite a identificação de cada *feature* representada no sistema.

Como se desejava que o sistema fosse um CAD "inteligente", foi utilizado um sistema especialista com a finalidade de executar a análise dos passos feitos pelo usuário, confrontando-os com uma base de conhecimento de modo a alertar e, algumas vezes, corrigir o usuário na sua tarefa.

Essa base de conhecimento permite, por exemplo, que qualquer tarefa que seja realizada para a inserção de uma *feature* seja analisada, buscando-se verificar a sua validade perante o contexto do projeto. Essa base de conhecimentos também pode conter informações a respeito das funções, como excluir e alterar, as quais envolvem transformações do contexto de projeto quando aplicadas. Com as informações organizadas na estrutura de dados e sabendo que elas são válidas para as situações ali representadas, é possível realizar a análise dessas informações e obter conclusões que geram novas informações sobre o produto representado.

Assim, foi desenvolvida a análise do conjunto a partir das informações das *features* na estrutura de dados, sendo possível identificar as peças que estão montadas, suas relações de montagem, superfícies de contato, gerando informações a respeito de como está representada a montagem do conjunto. Essa análise conta com a ajuda do sistema especialista, que atua sobre uma base de conhecimento de montagem, descrevendo as características de montagem entre as diversas *features* modeladas no sistema

De posse das informações do conjunto montado, também foi possível identificar as superfícies de referência que são utilizadas para o chamado dimensionamento funcional e, conseqüentemente, especificar automaticamente as dimensões a serem controladas no conjunto, isso com o uso de algoritmos e do sistema especialista, que se apóia na devida base de conhecimento. Além disso, foi possível identificar as cotas que se deve indicar, tolerâncias específicas em razão de sua condição de montagem de forma que o usuário, interativamente, proceda à escolha das tolerâncias mais adequadas à situação. Para isso, foi desenvolvido um banco de dados para tolerâncias a fim de fornecer as informações ao usuário, estando o mesmo integrado ao sistema CAD. Conhecendo as dimensões que possuem tolerâncias, é possível

identificar as superfícies que elas controlam, o que torna possível identificar as superfícies que devem ser usinadas e o respectivo acabamento superficial em função das tolerâncias já especificadas.

Finalmente, o sistema representa essas informações dimensionais através das cotas no desenho representado em 2D, sendo a sua distribuição definida através dos conceitos da cotagem funcional das *features* externas, o que é realizado automaticamente ou de forma manual. Tais informações estão armazenadas na estrutura de dados. Um banco de dados contendo informações a respeito de componentes fornecidos por terceiros (p.ex., rolamento) foi implementado e conectado ao sistema de modo que o usuário possua acesso a essas informações quando necessário, informações essas que permitem selecionar o rolamento em função das características de funcionamento e da dimensão do eixo onde será montado.

6.2 - PROBLEMAS ENCONTRADOS NA IMPLEMENTAÇÃO

A maior parte das dificuldades encontradas no presente trabalho foram de ordem computacional, ou seja, como representar uma informação de um objeto real num formato virtual utilizado pelo computador e, ao mesmo tempo, transmitir ao usuário a informação num formato que esse possa entender rapidamente.

Essa dificuldade pode ser caracterizada pelas representações geométricas apresentadas ao usuário, como no caso da inserção de uma *feature* modificadora ranhura. Sua inserção na estrutura de dados consiste apenas em preencher um espaço numa lista, contudo a representação gráfica envolve a eliminação da representação gráfica de uma *feature* eixo e a reconstrução deste eixo segmentado e da própria ranhura, mantendo a unicidade de duas representações gráficas que formam um único eixo e a individualidade da nova *feature* ranhura.

Em razão do grande volume de trabalho necessário para a representação das *features* de forma detalhada, algumas foram feitas de modo simplicado (representação gráfica dos elementos principais), visando apenas permitir a validação do modelo.

Na construção das regras, pode-se dizer que há dupla dificuldade: primeiro, em definir parâmetros que serão utilizados para representar as informações na base de conhecimento; segundo, a confiabilidade de que as novas regras introduzidas na base de conhecimento não entrem em conflito com as regras existentes, criando inconsistências. Há uma grande dificuldade

em manter a consistência de todo o sistema, já que, à medida que são introduzidas novas *features*, as inter-relações existentes em todos os aspectos tornam-se críticas, tanto no que se refere à manipulação das *features* como com relação à base de conhecimento das diversas funções que as utilizam.

6.3 - CONCLUSÕES

Os testes realizados no sistema foram feitos dentro do ambiente acadêmico, com a modelagem de alguns produtos que a biblioteca de *features* permitiu, na qual se procurou validar os conceitos do sistema, como, por exemplo, mancal de rolamentos, como visto na figura 4.28, sem a representação do rasgo de lingueta e chaveta, e outros conjuntos de peças semelhantes.

Através de testes de modelagem como o apresentado no capítulo 5, pôde-se verificar que a estrutura de dados concebida é robusta e que permite a modelagem de um conjunto de peças, cada uma possuindo várias *features* na sua composição, sendo que todas são representadas na estrutura do produto de forma única, permitindo prontamente a sua identificação.

Pôde-se observar a facilidade de criar *features* padrão e de chamá-las para dentro do sistema novamente, o que confirma a individualidade de cada *feature* que compõe o referido padrão, possibilitando a execução de alterações, bem como a inserção de novas *features* sobre as já existentes. A criação de *features* básicas e modificadoras para a biblioteca mostrou-se um tanto trabalhosa por exigir do usuário um bom conhecimento não só de programação, mas da filosofia do sistema para saber não somente onde introduzir novas informações, mas a forma de modelá-las. Dessa forma, o usuário do sistema (p. ex., um projetista) terá dificuldades de interagir em nível de programação, necessitando do auxílio de alguém treinado para a tarefa.

Com relação a customizar a base de conhecimentos utilizada tanto para a modelagem das *features* como para as bases de conhecimento utilizadas para a análise, como montagem, cotagem, etc., a mesma dificuldade descrita no parágrafo anterior pôde ser observada quando da execução dessa tarefa por um usuário com pouco conhecimento interno do sistema. Essa experiência foi realizada com estagiários do departamento quando da ampliação do sistema com novas *features*.

Quanto à utilização do sistema para um simples usuário (projetista), esse se mostrou satisfatório, uma vez que permitiu que um usuário comum conseguisse facilmente criar um produto e manipular as *features* da biblioteca, criar novas *features* através das *features* padrão.

Quanto à adaptação do sistema, chega-se à conclusão de que, para que o usuário possa efetivamente adaptar o sistema, o ideal seria que todas essas tarefas pudessem ser realizadas via interface gráfica, com base no conhecimento do meio onde o mesmo atua; por exemplo, o usuário poderia criar novas *features* para a biblioteca através da construção gráfica dessa, bem como incluir novas regras.

Mesmo com a utilização de meios mais amigáveis para realizar a adaptação, esse sistema exige um grupo de especialistas que auxiliem na descrição das relações existentes entre as diversas *features* do sistema com a nova *feature* criada. Com isso, gera-se uma grande quantidade de análises para verificar essas relações entre as *features*, o que afeta todos os módulos do sistema.

Para introduzir uma nova *feature*, é necessário que se criem as funções de manipulação (inserção, exclusão e alteração), bem como que se descrevam na base de conhecimento as novas regras para cada função de manipulação, levando em conta as inter-relações. Desse modo, à medida que o sistema cresce, a quantidade de dados a ser verificada para cada nova *feature* cresce em quantidade e complexidade.

Em razão desses fatores, o sistema protótipo foi implementado dentro da filosofia de que pudesse evoluir para se observar melhor as dificuldades do desenvolvimento de um sistema maior.

6.4 - POTENCIALIDADES DO SISTEMA

Durante a implementação do trabalho, foi desenvolvida uma interface que permitiu a comunicação dos dados gerados no sistema FeatCAD-2D com o sistema de planejamento de processos (CAPP) implementado por Rezende (1996), o que confirmou plenamente a expectativa de intercâmbio de dados entre os dois sistemas e a possibilidade de trabalharem integrados, já que os softwares de desenvolvimento utilizados foram praticamente os mesmos.

Outra experiência realizada foi a implementação de uma função para gerar o perfil de uma peça gerada por *features* para ser utilizada num software comercial de CAM, exportando o

perfil para o mesmo. Isso visou solucionar o problema de uma empresa quanto à geração de perfis para o torneamento de peças em softwares já existentes.

6.5 - CONTRIBUIÇÕES

Como principais contribuições deste trabalho, citam-se as seguintes:

- uso de uma estrutura de dados para representar as *features* independente do modelador gráfico, havendo entre eles um elo de ligação para manter a consistência entre as representações gráficas e as informações tecnológicas;

- um modelo de representação computacional para o produto, sendo sua implementação baseada em *features*;

- utilização de um Sistema Especialista juntamente com um sistema CAD, baseado em *features*, com o objetivo de validar a consistência das operações sobre as *features* em tempo real (operações como instanciar, excluir e alterar);

- construção de novas *features* (*feature* padrão) a partir das *features* disponíveis na biblioteca, sendo que essas novas *features* podem ser instanciadas mantendo a consistência. Sua criação se dá através da interface gráfica, e sua validação é feita pelo Sistema Especialista;

- identificação automática das peças montadas e suas relações de montagem a partir das *features* representadas na estrutura de dados (produto), não sendo utilizada para isso a base de dados gráfica do sistema CAD;

- definição de conceitos e parâmetros para a construção de uma base de conhecimento a respeito da identificação das peças montadas e de suas relações;

- identificação automática das superfícies que devem receber cotas com tolerâncias, análise essa baseada nas informações do conjunto montado;

- definição de uma base conceitual para a manipulação das *features* no CAD em relação às operações de instanciar, excluir e alterar;

- cotagem automática das peças, tendo como base as informações com relação ao conjunto montado;

- base de conhecimento para a identificação das cotas a serem atribuídas a uma peça.

6.6 - FUTUROS TRABALHOS

As propostas de trabalhos citadas nesse item têm como objetivo resolver alguns dos problemas encontrados no desenvolvimento do protótipo, como também ampliar esse sistema de modo que possa ser efetivamente utilizado na indústria.

- Com relação à ampliação das *features* da biblioteca, seria importante criar uma interface gráfica para a criação das *features* de modo que não seja necessário que o usuário se envolva com a linguagem de programação do sistema.

- Um trabalho importante a ser realizado, que também já foi citado por Rezende (1996), é um módulo que possibilite a verificação automática das regras quanto à consistência, o que permitirá que a adaptação da base de conhecimento possa ser realizada de modo mais fácil.

- Também com relação às regras, sugere-se a criação de uma interface gráfica que permita ao usuário a criação das regras sem que ele tome contato com a linguagem de programação, sistema semelhante feito para a criação de *features* padrão no sistema FeatCAD-2D.

- Desenvolvimento de um sistema com representações gráficas em 3D, utilizando-se também peças com formas prismáticas, baseado na estrutura de dados desenvolvida para esse sistema, com a utilização do conceito de *features* básicas e modificadoras como ponto de referência para a modelagem das informações.

- Elaboração de um módulo capaz de executar a distribuição das cotas nos desenhos de modo racional, obedecendo às normas técnicas.

- Refinamento do algoritmo para cotagem automática, levando em conta as cotas de outras *features* não desenvolvidas no presente trabalho e a sua funcionalidade no conjunto analisado.

- Construção de um módulo para a introdução de tolerâncias geométricas (p. ex., concentricidade), de modo a relacionar as *features* envolvidas para possibilitar que essas informações possam ser utilizadas para análise de projeto ou, então, por um sistema CAPP.

- De posse das cotas de cada peça, como mostra a implementação, desenvolver uma análise de cadeias de tolerâncias para o conjunto representado.

REFERÊNCIAS BIBLIOGRÁFICAS

ABDOU (1993), G., CHANG, R. TVCAPP, Tolerence verification in Computer-Aided Process Planning. **International Journal of Production Research**, v. 31, n. 2, p. 393-411.

ANANTHA (1996), Ram; KRAMER, Glenn A. and CRAWFORD Richard H. Assembly modelling by geometric constraint satisfaction. **Computer-Aided Design**, v. 28, n. 9, p. 707-722.

ANDO (1989), Koichi; YOSHIKAWA, Hiroyuki. Generation of manufacturing information in Intelligent CAD. **Annals of the CIRP**, v. 38, p.133-136.

ARARIBÓIA (1988), G. **Inteligência artificial.** Rio de Janeiro : Livros Técnicos e Científicos Editora Ltda..

AUTODESK (1996) INC. **AutoCAD Designer**. Assembly Modeling. AutoDesk, Inc..

BALDWIN (1991), Daniel F., ABELL, Thomas E., LUI, Man-Cheung Max, DE FAZIO, Thomas L., WHITNEY, Daniel E. An integrated computer aid for generating and evaluating assembly sequences for mechanical products. **IEEE Transactions on Robotics and Automation**. v. 7, no. 1, pp. 78-94.

BRONSVOORT (1994), Willem F.; JANSEN, Frederik W. Multiview feature modelling for design and assembly. **Advances in Feature Based Manufacturing**, Shah, J.J., Mäntylä, M., and Nau, D.S., Elsevier Science B. V., p. 313-330.

CADAM (1996), Helix Design System, CD-ROM, Versão Demo, EUA.

COAD (1992), Peter; YOURDON, Edward. **Análise baseada em objetos.** 2. ed. Rio de Janeiro : Campus.

COYNE (1990), R.D.; Rosenman, M.A.; Radford, A.D.; Balachandran, M.; Gero, J.S. **Knowledge-based design system**. Addison-Wesley Publishing Company, inc..

CUNHA (1993), G. D. & TEIXEIRA, J.J.P. A Feature-based model for design and manufacturing knowledge representation. **Proceedings of 12th Brazilian Congress of Mechanical Engineering**, Brasília, Brazil, December.

CUNHA (1995), G.D. O **Conceito de projecto orientado à fabricação aplicado ao projecto assistido por computador**. Lisboa.Tese (Doutorado em Engenharia), Universidade de Nova Lisboa.

DINI (1992), G.; SANTOCHI, M. Automated sequencing and subassembly detection in assembly planning. **Annals of the CIRP**, v. 41/1, p. 1-4.

DIXON (1987), John R., CUNNINGHAM, John J., SIMMONS Melvin K. RESEARCH IN DESIGNING WITH FEATURES. **Proceedings of the IFIP TC5/WG Workshop on Intelligent CAD**, Boston: Mass., pp. 137-148. oct..

DIXON (1995), J.R. Knowledge-based systems for design. **Transactions of the ASME**, v. 117, p. 11-16, june.

DONG (1993), Zuomin. Design for automated manufacturing. **Concurrent Engineering : Automation, Tools, and Techniques**, Edited by Andrew Kusiak. John & Sons, inc., p. 207-233.

EASTMAN (1994), Charles. Out of STEP? **Computer-Aided Design**, v. 26, n. 5, p. 338-340.

EYEDA (1991), O.E. e ONG, J.B. An Assembly recognition algorithm for automatic tolerancing. **Transactions of NAMRI/SME**, p. 309-314.

FENG (1996), Chan-Xue(Jack); HUANG, Chun-Che; KUSIAK, Andrew e LI Pei-Gen. Representation of functions and features in detail design. **Computer-Aided Design**, v. 28, n. 12, p. 961-971.

FUH (1996), Jerry Y.H.; CHANG, Chao-Hwa e MELKANOFF, Michel A. The development of an integrated and intelligent CAD/CAPP/CAFP environment using logic-based reasoning. **Computer-aided Design**, v. 28, n. 3, p. 217-232.

GAO (1996), J.X.; HUANG, X.X. Product and manufacturing capability modelling in an integrated CAD/process planning environment. **Int. J. Manuf. Technol.**, v. 11, p. 43-51.

GIARRATANO (1994), Joseph; RILEY, Gary. **Expert Systems :** principles and programming. 2. ed. Boston : PWS Publishing Company.

GU (1995), P. e YAN, X. CAD-directed automatic assembly sequence planning. **International Journal of Production Research.**, v. 33, n. 11, p. 3069-3100.

HALEVI (1994), Gideon. CAD for Manufacturing Support. **Feature modelling and recognition in advanced CAD/CAM system, Proceddings of the IFIP International Conference**, v. 1, p. 737-390.

HUTHWAITE (1992), Bart with SHNEBERGER, David. **Design for competitiveness** - The teamwork approach to product development.

JASTHI (1994), S.R.K.; PRASAD, A.V.S.R.K.; MANIDHAR, G.; RAO, P.N.; RAO, U.R.K. e TEWARI, N.K. A Feature-based part description system for computer-aided process planning. **Journal of Design and manufacturing**, p. 67-80.

JURI (1990), A.H.; SAIA, A.; DE PENNINGTON, A. Reasoning about machining operations using feature-based models. **International Journal of Production Research**, v. 28, n. 1, p. 153-171.

KIM (1993), Cheolhan; KIM, Kwangsoo e CHOI, Injun. AN OBJECT-ORIENTED INFORMATION MODELING METHODOLOGY FOR MANUFACTURING INFORMATION SYSTEMS. **Computers ind. Engng.**, v. 24, n. 3, p. 337-353.

KJELLBERG (1992), Torsten; SCHMEKEL, Hans. Product modelling and 'information-integrated' engineering system. **Annals of the CIRP**, v. 41/1, p. 201-204.

KO (1987), Heedong e LEE, Kunwoo. Automatic assembling procedure generation from mating conditions. **Computer-Aided Design**, v. 19, n 1, p. 3-10.

KRAUSE (1993), F. -L.; KIMURA, F.; KJELLBERG, T.; LU, S. C. -Y. Product modelling. **Annals of the CIRP**, v.42/2, p. 695-706.

KRISHNAN (1995), S. e SHRIHARI, K. A knowledge-based object oriented DFM advisor for surface mount PCB assembly. **Int. J. Adv. Manuf. Technol.**, v. 10, p. 317-319.

KULKARNI (1996), V.S. e PANDE, S.S. Representation of fetaure relationship tolerances in solid models. **International Journal of Production Research**, v. 34, n. 7, p. 1975-1994.

LIMA (1994), Celson P. **Modelamento baseado em *features* em um conceito de projeto para fabricação e montagem**. Florianópolis.Dissertação (Mestrado em Engenharia Mecânica)-Universidade Federal de Santa Catarina.

LIN (1995), Zone-Ching e CHEN, Shyh-Chang. The study of design data extraction from design drawing with application in measurement. **Int J. Adv. Manuf.**, v. 10, p. 99-109.

MASCLE (1994), C.; DUPINET, E.; MARANZANA, R. Feature modeling in assembly planning. **Proceedings of the IFIP International Conference of Feature_Modeling and Recognition in Advanced CAD/CAM System**. v. 2, p. 605-627.

MAZOUZ (1991), A.K.; SOUILAH, A.e TALBI, M. Design of an expert system for generating optimal assembly sequences. **Computer-Aided Engineering Journal**, p. 255-259.

MOREIRA (1993), N.P. Uma proposta de modelagem de informações para integração na manufatura e engenharia concorrente. Florianópolis.Dissertação (mestrado em Engenharia Mecânica) - Universidade Federal de Santa Catarina.

NGOI (1996), B.K.A. e ONG. C.T. Optimum assembly using a component dimensioning method. **The International Journal of Advanced Manufacturing Technology**. v. 11, p. 172-178.

NOVASKI (1994), Olívio. **Introdução à engenharia de fabricação mecânica**. Editora Edgard Blücher Ltda.

OVTCHAROVA (1992), Jiika; PAHL, Gerhard e RIX, Joachim. A proposal for feature classification in feature-based design. **Comput. & Graphics**, v. 16, n 2, p. 187-195.

PANCHAL (1992), Kaushai; RAMAN, Shivakumar; PULAT, P. Simin. Computer-aided tolerance assignment procedure (CATAP) for design dimensioning. **International Journal of Production Research**, v. 30, n. 3, p. 599-610.

PRATT (1990) Michael J. A hybrid feature-based modelling system. **Advanced Geometric Modelling for Engineering Applications**. F.-L. Krause and H. Jansen. Editor Elsevier Science Publisher B.V., p.189-201.

PUGLIESI (1986), Marcio. **Desenho mecânico e de máquinas**. por Marcio Pugliesi [e] Diamantino F. Trindade. São Paulo: Ícone Editora Ltda..

REZENDE (1996), Darcio de Freitas. Planejamento de processos de fabricação assistido por computador através de um sistema especialista baseado na tecnologia de *features*: um modelo de desenvolvimento voltado para a realidade industrial. Florianópolis.Dissertação (mestrado em Engenharia Mecânica) - Universidade Federal de Santa Catarina.

ROLLER (1989), Dieter. Design by Feature : An approach to high level shape manipulations. **Computer in Industry**, v. 12, p. 185-191.

ROPION (1994), R. **Cotação funcional dos desenhos técnicos.** Tradução João M. Csillag, São Paulo: McGRAW-HILL.

ROSA (1993), Edison da. **Banco de dados em sistemas integrados de manufatura**.Florianópolis : Universidade Federal de Santa Catarina. (Apostila).

ROSA (1994), Edison da, SILVA, J.C., SILVA, C.A., RAMINELLI, L.F., MOREIRA, N.P., FARIA, P.O., POSTAL, R., VIEIRA, R.S. Uma base para a implantação dos conceitos de engenharia simultânea em um ambiente computacional, Atas do CICONGRAF'94 - Congresso Internacional de Computação Gráfica, São Paulo.

ROY (1993), Utpal e LIU, C.R. Integrated cad frameworks: tolerance representation scheme in a solid model. **Computers ind. Engng**. v. 24, n. 3, p. 495-509.

SCHULZ (1993), H, & SCHÜTZER, K. Integração de projeto e planejamento baseado em *feature*. **Máquinas e Metais**. p.28-36, set..

SCHULZ (1994), Herbert, SCHÜTZER, Klaus. FINDES - Integration design and manufacturing. **Feature Modelling and Recognition in Advanced CAD/CAM System, Proceddings of the IFIP international conference**, v 1, p. 43-58.

SHAH (1988), J.J., ROGERS, M.T. Feature based modelling shell : design and implementation. **Proceedings of ASME Computers in Engineering Conference**, July.

SHAH (1991), J.J. Assessment of features technology. **Computer-aided Design**. v. 23, n. 5, p-331-343.

SHAH (1994), Jami J.; MÄNTYLÄ, Martti; NAU, Dana. Introduction to feature based manufacturing. **Advanced in Feature Based Manufacturing**, Shah, J.J., Mäntylä, M., and Nau, D.S., Elsevier Science B.V., p. 1-11.

SUZUKI (1996), K.; KIMURA, F.; MOSER, B.; YAMADA, T. Modelling information in design background for product development support. **Annals of the CIRP**, v. 45/1, p.141-144.

TOMIYAMA (1993), Tetsuo; XUE, Deyi, e YOSHIKAWA, Hiroyuki. **Developing an intelligent CAD system**. p. 11-39.

USHER (1996), John M. A tutorial and review of objetct-oriented design of manufacturing software systens. **Computers ind. Engng**., v. 30, n. 4, p. 781-798.

USHER (1996), John M. A STEP-based object-oriented product model for process planning. **Computers ind. Engng**., v. 31, n. 1/2, p. 185-188.

VAN DER NET (1996), A.J. A relation-based product model suited for integrating design and manufacturing. **Annals of the CIRP**, v. 45/1, p. 161-164.

WANG (1996), Nanxin e OZSOY, Tulga M. Representation of assemblies for automatic tolerance chain generation. **Engineering with Computers**. v. 6, p. 121-126.

WATERMAN (1996), Donald A. **A guide to expert system**. Addison-Wesley P.C..

WEILL (1988), R.; CLÉMENT, A.; HOCKEN, R.; FARMER, L.E.; GLADMAN, C.A.; WIRTZ, A.; BOURDET, P.; FRECKLETON, J.E.; KUZMANN, H.; HAM, I.; TRUMPOLD, H.; MATHIAS, E. Tolerancing for function, **Annals of the CIRP**, v. 37/2, p. 603-610.

WEULE (1989), H., FRIEDMAN, Th. Computer-aided product analysis in assembly-planning. **Annals of the CIRP**, v 38/1, p.1-4.

WINGARD (1992), Lars; CARLEBERG, Per; KJELLBERG, Torsten. Enabling use of angineering in product models and user interfaces of CAD/CAM systems. **Annals of the CIRP**, v. 41/1, p.205-208.

WOSNY (1994), M.J., PRATT, M.J., POLI, C. Topics in feature-based design and manufacturing. **Advances in Feature Based Manufacturing**. Shah, J.J., Mäntylä, M, and Nau, D.S., Elsevier Science B.V., pp. 481-410.

WU (1995), B. Object-oriented systems analysis and definition of manufacturing operations. **International Journal of Production Research**, v. 33, n. 4, p. 955-974.

YEUN (1988), M.M.F; TAN, S.T.; YU, K.M. Schema for automatic dimensioning of CSG defined parts. **Computer-aided Design**, v. 20, n. 3, p.151-159.

ZHANG (1992), H.C.; HUQ, M.E. Tolerancing techniques : the state-of-art. **International Journal of Production Research**, v. 30, n. 9, p. 2111-2135.

ZHANG (1995), Hong C. e MEI J. Automated tolerance analysis for CAPP system. **Int. J. Adv. Manuf. Technol**. v. 10, p. 219-224.

BIBLIOGRAFIA

AGOSTINHO, Osvaldo Luís.**Tolerâncias, ajustes, desvios e análise de dimensões**. por Osvaldo Luís Agostinho, Antonio Carlos dos Santos Rodrigues [e] João Lirani. São Paulo: edgard Blücher, 1977.

ARAKAKI, Reginaldo; ARAKAKI, Julio; ANGERAMI, Paulo Mattos et al. **Fundamentos de programação C: técnicas e aplicações**. 2. ed, Rio de Janeiro: Livros Técnicos e Científicos, 1990.

AUTODESK INC. **AutoCAD Development System:** programmer's reference. Manual. 1. ed. USA : AutoDESK, 1992.

BACK, Nelson. **Metodologia de projeto de produtos industriais**. Rio de Janeiro, Guanabara Dois, 1983.

BOOTHROYD, Geoffrey. Product design for manufacture and assembly. **Computer-Aided Design**, v. 26, n. 7, p. 505-520, 1994.

BORLAND INTERNATIONAL INC. **Borland C++ Version 4.5:** Class Libraries Guide. 1. ed. USA : Borland International, 1994.

BORLAND INTERNATIONAL INC. **Borland C++ Version 4.5:** Libraries Reference. 1. ed. USA : Borland International, 1994.

BORLAND INTERNATIONAL INC. **Borland C++ Version 4.5:** Programer's Guide. 1. ed. USA : Borland International, 1994.

BORLAND INTERNATIONAL INC. **Borland C++ Version 4.5:** User's Guide. 1. ed. USA : Borland International, 1994.

BROWN, Curtis W. **Feature-Based tolerancing for intelligent process definition**. International Forum on Dimensional Tolerancing and Metrology, ASME 1993, CRTD-v. 27, pp. 249-258.

CALVERT, Charlie. **Programando aplicações em Windows com C & C++**. 1. ed. Rio de Janeiro : Berkeley, 1994.

CHANG, Tien-Chien, WYSK, Richard A. **An introduction to automated process planning system**. Printice-Hall, INC., 1985.

CUTKOSKY, M.R.; TENENBAUM, J.M., CAD/CAM INTEGRATION THROUGH CONCURRENT PROCESS AND PRODUCT DESIGN, [?].

DELBRESSINE, F.L.M.; VAN DER WOLF, A.C.H. Integrating design and manufacturing. **Annals of the CIRP**, v. 39, p.149-152, 1990.

ELMARAGHY, Hoda A. e ElMaraghy Waguih H. **Computer-Aided Inspection Planning (CAIP), Advanced in Feature Based Manufacturing**. Shah, J.J., Mäntylä, M., and Nau, D.S., Elsevier Science B.V., p. 363-396, 1994.

FERREIRA, João C.E.; BUTZKE, A.U.; FURLAN NETO, F. Um sistema de projeto de peças usinadas baseado em *features* aplicado à realidade industrial. **Revista Brasileira de Ciências Mecânicas**, v. xvii, n. 2, 1995.

GUI, Jin-Kang e MÄNTILÄ, Martti. Assembly modeling on the basis of a mechanical design prototype. **Geometric modeling for product realization**. P.R Wilson, M.J. Wozny and M.J. Pratt. Elsevier Science Publisher B.V., p. 109-128, 1993.

GUPTA, Satyandra K. e NAU, Dana S. Systematic approach to analysing the manufacturability of machined parts. **Computer-Aided Design**, v. 5, n. 27, p. 323-342, 1995.

HEEMSKERK, Ir. C.J.M. The Use of heuristics in assembly sequence planning. **Annals of the CIRP**, v. 38, p. 37-40, 1989.

IRANI, S.A.; KOO, H.-Y. e RAMAN, S. Feature-based operation sequence generation in CAPP. **International Journal of Production Research**, v. 33, n. 1, p. 17-39, 1995.

KIM, Joo-Yong; MITTAL, Ravio; O'GRAYDY, Peter e YOUNG Robert E. Process selection for concurrent engineering in the domain of rotational parts. **Journal of Design and manufacturing**, p. 199-209, 1992.

KIMURA, Fumihiko; SUZUKI, Hirosama. Representing background information for product description to support product development process. **Annals of the CIRP**, v. 44, p. 113-116, 1995.

KUSIAK, Andrew. **Intelligent manufacturing system**. Printice-Hall, Inc., 1990.

LI, J.K.; ZHANG, C. Operational dimensions and tolerances calculation in CAPP systems for precision manufacturing. **Annals of the CIRP**, v. 38, p. 403-406, 1989.

LORINI, Flávio J. **Tecnologia de grupo e organização da manufatura**. Florianópolis, ed. da UFSC, 1993.

MÄNTYLÄ, M. A modeling system for top-down design of assembled products. **IBM J. RES. DEVELOP.**, v. 34, n. 5, p. 636-659, 1990.

MOURÃO, Antonio J.F. A engenharia simultanea como metodologia organizativa de suporte à aplicação do projecto para fabrico e montagem [?] Universidade de Nova Lisboa.

MULLINS S.H. and ANDERSON D.C. Feature-based tolerance representation for design and analysis. **Journal of Design and Manufacturing**, v. 1, p. 107-118, 1991.

NASA. **CLIPS Version 6.0:** C Language Integrated Production System. 1. ed. Georgia: NASA, 1993.

NASA. **CLIPS Version 6.0 : Advanced Programming Guide**. 1. ed. Georgia, 1993.

NASA. **CLIPS Version 6.0:** Basic Programming Guide. 1. ed. Georgia, 1993.

NASA. **CLIPS Version 6.0:** Interfaces Guide. 1. ed. Georgia, 1993.

NASA. **CLIPS Version 6.0:** User's Guide. 1. ed. Georgia, 1993.

PANDE, S.S., DESAI, V.S. Expert CAPP system for single apindle automats. **International Journal of Production Research,** v 33, n. 3, p. 819,833, 1995.

PERRY, Greg. **Programação orientada para objeto com turbo C++**. 1. ed. Rio de Janeiro : Berkeley, 1994.

PLUMMER, J.C.S., HANNAM, R.G. Design for manufacture using a CAD/CAM system - a metodology for turned parts. **Proc Inst. Mech. Engrs**, v.197B, p. 187-195, 1983.

POLI, Corrado and FENOGLIO, F. Designing parts for automatic assembly. **Machine Design**, p. 140-145,december 10, 1987.

SHPITALNI, M., ELBER, G., LENZ, E. Automatic assembly of three-dimensional structures via connectivity graphs. **Annals of the CIRP**, v. 38, p. 25-28, 1989.

SINGH, Rajiv and RAMAN, Shivakumar. METEX - An expert system for machining planning. **International Journal of Production Research**, v. 30, n.7, p. 1501-1516, 1992.

SPYRIDI Antonia J. e REQUICHA, Aristides A.G. Automatic planning for dimensional inspection. **International Forum on Dimensional Tolerancing and Metrology**, ASME 1993, CRTD-VOL-27, p. 219-228.

VAN HOUTEN, F.J.A.M. **PART:** a computer aided process planning system. Tese (Doutoramento em Engenharia) Universiteit Twente, 1991.

APÊNDICE I
HIERARQUIA DE CLASSES DO SISTEMA ESPECIALISTA

Aqui são apresentadas as classes do sistema especialista que definem o modelo de informação utilizado para efetuar as análises do sistema FeatCAD-2D.

```
(defmodule MAIN
 (export ?ALL)
)

;* Modulo Regras *
(defmodule PECA
 (export ?ALL)
)

;*******************
(defclass PECA::PECA
                (is-a USER)
                (role concrete)
                (pattern-match reactive)
        (slot CoordY
                (type FLOAT)
                (create-accessor read-write)
                (visibility public))
        (slot DED
                (type FLOAT)
                (create-accessor read-write)
                (visibility public))
        (slot DEE
                (type FLOAT)
                (create-accessor read-write)
                (visibility public))
        (slot DID
                (type FLOAT)
                (create-accessor read-write)
                (visibility public))
        (slot DIE
                (type FLOAT)
                (create-accessor read-write)
                (visibility public))
        (slot NumeroRefExt
                (type INTEGER)
                (create-accessor read-write)
                (visibility public))
        (multislot Features
                (type INSTANCE-NAME)
                (create-accessor read-write)
                                        (visibility public))
)

;**********************
(defclass PECA::FEATURE
                (is-a USER)
                (role concrete)
                (pattern-match reactive)
        (slot PontoX
```

```
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot PontoY
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot Direcao
        (type STRING)
        (create-accessor read-write)
        (visibility public))
    (slot Posicao
        (type STRING)
        (create-accessor read-write)
        (visibility public))
    (slot Tipo
        (type STRING)
        (create-accessor read-write)
        (visibility public))
    (slot Sentido
        (type INTEGER)
        (create-accessor read-write)
        (visibility public))
    (slot ptInsercaoX
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot ptInsercaoY
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot Flag
        (type INTEGER)
        (create-accessor read-write)
        (visibility public))
    (slot nomeentidade
        (create-accessor read-write)
        (visibility public))

    (slot CoordI
        (type FLOAT)
        (default 0.0)
        (create-accessor read-write)
        (visibility public))
    (slot CoordF
        (type FLOAT)
        (default 0.0)
        (create-accessor read-write)
        (visibility public))
    (slot CoordY
        (type FLOAT)
        (default 0.0)
        (create-accessor read-write)
        (visibility public))

)
;**********
;
(defclass PECA::EIXO
```

```
                (is-a FEATURE)
                (role concrete)
                (pattern-match reactive)
        (slot Comprimento
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))
        (slot EREI
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))
        (slot EREF
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))
        (slot ERII
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))
        (slot ERIF
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))
)

;**********************
(defclass PECA::EIXOCIL
                (is-a EIXO)
                (role concrete)
                (pattern-match reactive)
        (slot Diametro
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write))
)

;*******************
(defclass PECA::FURO
                (is-a FEATURE)
                (role concrete)
                (pattern-match reactive)
        (slot Profundidade
                (type FLOAT)
                (create-accessor read-write)
                (visibility public))
        (slot EREI
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))
        (slot EREF
                (type FLOAT)
                (default 0.0)
```

```
                (create-accessor read-write)
                (visibility public))
        (slot ERII
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))
        (slot ERIF
                (type FLOAT)
                (default 0.0)
                (create-accessor read-write)
                (visibility public))

)

;*********************
(defclass PECA::FUROCIL
                (is-a FURO)
                (role concrete)
                (pattern-match reactive)
        (slot Diametro
                (type FLOAT)
                (create-accessor read-write))
)
;*********************
(defclass PECA::CANTO
                (is-a FEATURE)
                (role concrete)
                (pattern-match reactive)
)
;*********************
(defclass PECA::CHANFRO
                (is-a CANTO)
                (role concrete)
                (pattern-match reactive)
        (slot Angulo
                (type FLOAT)
                (create-accessor read-write))
        (slot Diametro
                (type FLOAT)
                (create-accessor read-write))
        (slot Comprimento
                (type FLOAT)
                (create-accessor read-write))
)

;*********************
(defclass PECA::CONCORDANCIA
                (is-a CANTO)
                (role concrete)
                (pattern-match reactive)
        (slot RaioConcordancia
                (type FLOAT)
                (create-accessor read-write))
        (slot Diametro
                (type FLOAT)
                (create-accessor read-write))
)
```

```
;**********************
(defclass PECA::ARREDONDAMENTO
        (is-a CANTO)
        (role concrete)
        (pattern-match reactive)
    (slot Raio
        (type FLOAT)
        (create-accessor read-write))
    (slot Diametro
        (type FLOAT)
        (create-accessor read-write))
)

;*******************
(defclass PECA::RANHURA
        (is-a FEATURE)
        (role concrete)
        (pattern-match reactive)
    (slot DiametroExt
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot DiametroInt
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
)

;*******************
(defclass PECA::RANHURANORMAL
        (is-a RANHURA)
        (role concrete)
        (pattern-match reactive)
    (slot Largura
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
)
;*******************
(defclass PECA::DIMENSAO
        (is-a USER)
        (role concrete)
        (pattern-match reactive)
    (slot PtoI
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot PtoF
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot Afs
        (type FLOAT)
        (create-accessor read-write)
        (visibility public))
    (slot Afi
        (type FLOAT)
```

```
                (create-accessor read-write)
                (visibility public))
        (slot PtoMedio
                (type FLOAT)
                (create-accessor read-write)
                (visibility public))
        (slot TipoCota
                (type STRING)
                (create-accessor read-write)
                (visibility public))
        (slot Flag
                (type INTEGER)
                (create-accessor read-write)
                (visibility public))
        (slot Sentido
                (type INTEGER)
                (create-accessor read-write)
                (visibility public))
)
;*****************************************
;* Fim do Arquivo PECA.CLP               *
;*****************************************
```

APÊNDICE II
REGRAS DE INSERÇÃO DE *FEATURES*

Aqui são apresentadas algumas regras para ilustrar a base de conhecimento implementada com relação à inserção de *features* pelo sistema FeatCAD-2D.

```
;**************************
;* Modulo Regras para Eixo *
;**************************
(defmodule REGRASEIXO
 (import PECA defclass ?ALL)
)

;*******
;DESENHA UM EIXO COM SENTIDO A DIREITA 1
(defrule REGRASEIXO::EixoDireita
 ?Peca <- (object (is-a PECA) (DED ?Ded)(DID ?Did&:(= ?Ded ?Did)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(> ?PtX ?Ded)) (Diametro ?Diam0))
 ?ECi1 <- (object (name [EXCL1]) (EREF ?Eref1) (Diametro ?Diam1&:(= ?Eref1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido 1)
 (send ?ECi0 put-Flag 0)
)

;*******
;DESENHA UM EIXO COM SENTIDO A DIREITA SE HA UM MODIFICADOR CHANFRO
(defrule REGRASEIXO::EixoDireita1
 ?Peca <- (object (is-a PECA) (DED ?Ded)(DID ?Did&:(= ?Ded ?Did)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(>= ?PtX ?Ded)) (Diametro ?Diam0))
?ECi1 <- (object (name [EXCL1]) (EREF ?Eref1&:(>= ?Eref1 ?Diam0)) (Diametro ?Diam1&:(<> ?Eref1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido 1)
 (send ?ECi0 put-Flag 0)
)

;*******
```

```
;NAO DESENHA EIXO A DIREITA SE DIAMETRO DO EIXO INTERFERE COM CHANFRO EXISTENTE
(defrule REGRASEIXO::EixoChanfroDireita
 ?Peca <- (object (is-a PECA) (DED ?Ded)(DID ?Did&:(= ?Ded ?Did)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(>= ?PtX ?Ded)) (Diametro ?Diam0))
 ?ECi1 <- (object (name [EXCL1]) (EREF ?Eref1&:(< ?Eref1 ?Diam0)) (Diametro ?Diam1&:(<> ?Eref1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido 0)
 (send ?ECi0 put-Flag 3)
)

;*******
;NAO DESENHA UM EIXO COM SENTIDO A DIREITA SE EXISTE UM FURO AXIAL
(defrule REGRASEIXO::NaoEixoDireita
 (declare (salience 20))
 ?Peca <- (object (is-a PECA) (DED ?Ded)(DID ?Did&:(<> ?Ded ?Did)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(>= ?PtX ?Ded)) (Diametro ?Diam0))
=>
 (send ?ECi0 put-Sentido 0)
 (send ?ECi0 put-Flag 2)
 (return)
)

;*******
;NAO DESENHA EIXO A DIREITA SE DIAMETRO DO EIXO E' IGUAL AO DO EIXO VIZINHO
(defrule REGRASEIXO::EixoDiametroDireita
 ?Peca <- (object (is-a PECA) (DED ?Ded)(DID ?Did&:(= ?Ded ?Did)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(>= ?PtX ?Ded)) (Diametro ?Diam0))
 ?ECi1 <- (object (name [EXCL1]) (Diametro ?Diam1&:(= ?Diam1 ?Diam0))
                                 (EREF ?Eref1&:(= ?Eref1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido 0)
 (send ?ECi0 put-Flag 4)
)

;*******
;DESENHA UM EIXO COM SENTIDO A ESQUERDA -1
(defrule REGRASEIXO::EixoEsquerda
 ?Peca <- (object (is-a PECA) (DEE ?Dee)(DIE ?Die&:(= ?Dee ?Die)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(< ?PtX ?Dee)) (Diametro ?Diam0))
```

```
 ?ECi1 <- (object (name [EXCL1]) (EREI ?Erei1) (Diametro ?Diam1&:(= ?Erei1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido -1)
 (send ?ECi0 put-Flag 0)
)

;*******
;DESENHA UM EIXO COM SENTIDO A ESQUERDA SE HA UM MODIFICADOR CHANFRO
(defrule REGRASEIXO::EixoEsquerda1
 ?Peca <- (object (is-a PECA) (DEE ?Dee)(DIE ?Die&:(= ?Dee ?Die)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(<= ?PtX ?Dee)) (Diametro ?Diam0))
 ?ECi1 <- (object (name [EXCL1]) (EREI ?Erei1&:(>= ?Erei1 ?Diam0)) (Diametro ?Diam1&:(<> ?Erei1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido -1)
 (send ?ECi0 put-Flag 0)
)

;*******
;NAO DESENHA EIXO A ESQUERDA SE DIAMETRO DO EIXO INTERFERE COM CHANFRO EXISTENTE
(defrule REGRASEIXO::EixoChanfroEsquerda
 ?Peca <- (object (is-a PECA) (DEE ?Dee)(DIE ?Die&:(= ?Dee ?Die)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(<= ?PtX ?Dee)) (Diametro ?Diam0))
 ?ECi1 <- (object (name [EXCL1]) (EREI ?Erei1&:(< ?Erei1 ?Diam0)) (Diametro ?Diam1&:(<> ?Erei1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido 0)
 (send ?ECi0 put-Flag 3)
)

;*******
;NAO DESENHA UM EIXO COM SENTIDO A ESQUERDA SE EXISTIR UM FURO AXIAL
(defrule REGRASEIXO::NaoEixoEsquerda
 (declare (salience 10))
 ?Peca <- (object (is-a PECA) (DEE ?Dee) (DIE ?Die&:(<> ?Dee ?Die)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(<= ?PtX ?Dee)))
=>
 (send ?ECi0 put-Sentido 0)
 (send ?ECi0 put-Flag 2)
 (return)
)
```

```
;*******
;NAO DESENHA EIXO SE O PONTO DE INSERCAO ESTA SOBRE UM EIXO EXISTENTE
(defrule REGRASEIXO::NaoEixo
 ?Peca <- (object (is-a PECA) (DEE ?Dee) (DED ?Ded))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(and (>= ?PtX ?Dee)
                                                    (<= ?PtX ?Ded))))
=>
 (send ?ECi0 put-Sentido 0)
 (send ?ECi0 put-Flag 1)
)

;*******
;NAO DESENHA EIXO A ESQUERDA SE DIAMETRO DO EIXO E' IGUAL AO DO EIXO VIZINHO
(defrule REGRASEIXO::EixoDiametroEsquerda
 ?Peca <- (object (is-a PECA) (DEE ?Dee)(DIE ?Die&:(= ?Dee ?Die)))
 ?ECi0 <- (object (name [EXCL0]) (PontoX ?PtX&:(<= ?PtX ?Dee)) (Diametro ?Diam0))
 ?ECi1 <- (object (name [EXCL1]) (Diametro ?Diam1&:(= ?Diam1 ?Diam0)) (EREI ?Erei1&:(= ?Erei1 ?Diam1)))
=>
 (send ?ECi0 put-Sentido 0)
 (send ?ECi0 put-Flag 4)
)

;**************************
;* Modulo Regras para Furo*
;**************************
(defmodule REGRASFURO
 (import PECA defclass ?ALL)
)
;*******
;DESENHA UM FURO COM SENTIDO À DIREITA 1
(defrule REGRASFURO::Furo1
 ?Peca <- (object (is-a PECA) (DID ?Did) (DIE ?Die) (DED ?Ded))
 ?ECil <- (object (is-a EIXOCIL) (CoordF ?CdF) (EREI ?EreiE))
 ?Furo <- (object (is-a FUROCIL) (PontoX ?PtX&:(< ?PtX (/ (+ ?Did ?Die) 2)))
          (Profundidade ?Prof&:(and (<= ?Prof (- ?CdF ?Die)) (<= ?Prof (- ?Did ?Die))))
               (Tipo "furocilindricocego")(Diametro ?DiamF&:(< ?DiamF ?EreiE)))
=>
 (send ?Furo put-Sentido 1)
```

```
 (send ?Furo put-Flag 0)
)
;*******
;DESENHA UM FURO COM SENTIDO A ESQUERDA
(defrule REGRASFURO::Furo2
 ?Peca <- (object (is-a PECA) (DID ?Did) (DIE ?Die))
 ?ECil <- (object (is-a EIXOCIL) (CoordI ?CdI) (EREF ?ErefE))
 ?Furo <- (object (is-a FUROCIL) (PontoX ?PtX&:(> ?PtX (/ (+ ?Did ?Die) 2)))
          (Profundidade ?Prof&:(and (<= ?Prof (- ?Did ?CdI))(<= ?Prof (- ?Did ?Die))))
                    (Diametro ?DiamF&:(< ?DiamF ?ErefE)))
=>
 (send ?Furo put-Sentido -1)
 (send ?Furo put-Flag 0)
)

;*******
; NAO DESENHA UM FURO SE PONTO DE INSERCAO ESTIVER FORA DO EIXO
(defrule REGRASFURO::Furo3
 (declare (salience 10))
 ?Peca <- (object (is-a PECA) (DID ?Did) (DIE ?Die))
 ?Furo <- (object (is-a FUROCIL) (PontoX ?PtX&:(or (> ?PtX ?Did) (< ?PtX ?Die))))
=>
 (send ?Furo put-Sentido 0)
 (send ?Furo put-Flag 1)
 (return)
)

;*******
;NAO DESENHA FURO A ESQUERDA SE DIAMETRO DO FURO E MAIOR QUE DIAMETRO DO EIXO
(defrule REGRASFURO::Furo4
 ?Peca <- (object (is-a PECA) (DID ?Did) (DIE ?Die))
 ?ECil <- (object (is-a EIXOCIL) (EREF ?ErefE))
 ?Furo <- (object (is-a FUROCIL) ;(PontoX ?PtX&:(or(> ?PtX (/ (+ ?Did ?Die) 2)) (> ?PtX (/ (+ ?Did ?Die) 2))))
(Diametro ?DiamF&:(>= ?DiamF ?ErefE)))
=>
 (send ?Furo put-Sentido 0)
 (send ?Furo put-Flag 2)
)
```

```
;*******
;NAO DESENHA FURO A DIREITA SE DIAMETRO DO FURO E MAIOR QUE DIAMETRO DO EIXO
(defrule REGRASFURO::Furo5
 ?Peca <- (object (is-a PECA) (DID ?Did) (DIE ?Die))
 ?ECil <- (object (is-a EIXOCIL) (EREI ?EreiE))
 ?Furo <- (object (is-a FUROCIL) ;(PontoX ?PtX&:(or(< ?PtX (/ (+ ?Did ?Die) 2)) (> ?PtX (/ (+ ?Did ?Die) 2))))
(Diametro ?DiamF&:(>= ?DiamF ?EreiE)))
=>
 (send ?Furo put-Sentido 0)
 (send ?Furo put-Flag 2)
)

;*******
;COMPRIMENTO DO FURO A DIREITA E MAIOR QUE COMPRIMENTO DO EIXO
(defrule REGRASFURO::Furo6
 ?Peca <- (object (is-a PECA) (DID ?Did) (DIE ?Die))
 ?ECil <- (object (is-a EIXOCIL) (Diametro ?DiamE) (CoordF ?CdF))
 ?Furo <- (object (is-a FUROCIL) (PontoX ?PtX&:(< ?PtX (/ (+ ?Did ?Die) 2)))
                  (Profundidade ?Prof&:(> ?Prof (- ?CdF ?Die)))
                  (Diametro ?DiamF&:(< ?DiamF ?DiamE)))
=>
 (send ?Furo put-Sentido 1)
 (send ?Furo put-Flag 1000)
)

;*******
;COMPRIMENTO DO FURO A ESQUERDA MAIOR QUE COMPRIMENTO DO EIXO
(defrule REGRASFURO::Furo7
 ?Peca <- (object (is-a PECA) (DID ?Did) (DIE ?Die))
 ?ECil <- (object (is-a EIXOCIL) (Diametro ?DiamE) (CoordI ?CdI))
 ?Furo <- (object (is-a FUROCIL) (PontoX ?PtX&:(> ?PtX (/ (+ ?Did ?Die) 2)))
          (Profundidade ?Prof&:(> ?Prof (- ?Did ?CdI)))
          (Diametro ?DiamF&:(< ?DiamF ?DiamE)))
=>
 (send ?Furo put-Sentido -1)
 (send ?Furo put-Flag 1000)
)

;*************************************
```

```
;* Modulo Regras para Chanfros em Eixos*
;*************************************
(defmodule REGRASCHANFROEIXO
 (import PECA defclass ?ALL)
)

;*******
;INTRODUCAO DE UM CHANFRO EXTERNO NA EXTREMIDADE ESQUERDA DA PECA
(defrule REGRASCHANFROEIXO::ChanfE1
 ?Peca <- (object (is-a PECA) (DEE ?Dee))
 ?ECil <- (object (is-a EIXOCIL) (Diametro ?DiamE) (CoordI ?CdI&:(= ?CdI ?Dee))
          (CoordF ?CdF) (EREI ?Erei&:(= ?DiamE ?Erei)) (ERII ?Erii))
 ?Chan <- (object (is-a CHANFRO) (PontoX ?PtX&:(< ?PtX (/ (+ ?CdI ?CdF) 2)))
                  (Posicao "Externo") (Comprimento ?CompCh) (Angulo ?AngCh&:
                  (> ?Erei (+ ?Erii (* 2 (* ?CompCh (tan ?AngCh)))))))
=>
 (send ?Chan put-Sentido -1)
 (send ?Chan put-Flag 0)
)

;*******
;NAO INTRODUCAO DE UM CHANFRO EXTERNO NA EXTREMIDADE ESQUERDA DA PECA SE
;JA EXISTE CHANFRO
(defrule REGRASCHANFROEIXO::ChanfE2
 ?Peca <- (object (is-a PECA) (DEE ?Dee))
 ?ECil <- (object (is-a EIXOCIL) (Diametro ?DiamE) (CoordI ?CdI&:(= ?CdI ?Dee))
                  (CoordF ?CdF) (EREI ?Erei&:(<> ?DiamE ?Erei)))
 ?Chan <- (object (is-a CHANFRO) (PontoX ?PtX&:(< ?PtX (/ (+ ?CdI ?CdF) 2)))
                                                  (Posicao "Externo"))
=>
 (send ?Chan put-Sentido 0)
 (send ?Chan put-Flag 3)
)

;*******
;NAO INTRODUCAO DE UM CHANFRO EXTERNO NA EXTREMIDADE ESQUERDA DA PECA SE HA
;INTERFERENCIA COM DIAMETRO DO FURO
(defrule REGRASCHANFROEIXO::ChanfE12
 ?Peca <- (object (is-a PECA) (DEE ?Dee))
```

```
 ?ECil <- (object (is-a EIXOCIL) (Diametro ?DiamE) (CoordI ?CdI&:(= ?CdI ?Dee))
                    (CoordF ?CdF) (EREI ?Erei&:(= ?DiamE ?Erei)) (ERII ?Erii))
 ?Chan <- (object (is-a CHANFRO) (PontoX ?PtX&:(< ?PtX (/ (+ ?CdI ?CdF) 2)))
          (Posicao "Externo") (Comprimento ?CompCh) (Angulo ?AngCh&:
                    (< ?Erei (+ ?Erii (* 2 (* ?CompCh (tan ?AngCh)))))))
=>
 (send ?Chan put-Sentido 0)
 (send ?Chan put-Flag 7)
)
;*******
;INTRODUCAO DE UM CHANFRO EXTERNO NA EXTREMIDADE DIREITA DA PECA
(defrule REGRASCHANFROEIXO::ChanfE3
 ?Peca <- (object (is-a PECA) (DED ?Ded))
 ?ECil <- (object (is-a EIXOCIL) (Diametro ?DiamE) (CoordI ?CdI)
          (CoordF ?CdF&:(= ?CdF ?Ded)) (EREF ?Eref&:(= ?DiamE ?Eref)) (ERIF ?Erif))
 ?Chan <- (object (is-a CHANFRO) (PontoX ?PtX&:(> ?PtX (/ (+ ?CdI ?CdF) 2)))
                    (Posicao "Externo") (Comprimento ?CompCh) (Angulo ?AngCh&:
                    (> ?Eref (+ ?Erif (* 2 (* ?CompCh (tan ?AngCh)))))))
=>
 (send ?Chan put-Sentido 1)
 (send ?Chan put-Flag 0)
)

;*******
;NAO INTRODUCAO DE UM CHANFRO EXTERNO NA EXTREMIDADE DIREITA DA PECA SE
; JA EXISTE CHANFRO
(defrule REGRASCHANFROEIXO::ChanfE4
 ?Peca <- (object (is-a PECA) (DED ?Ded))
 ?ECil <- (object (is-a EIXOCIL) (Diametro ?DiamE) (CoordI ?CdI)
                    (CoordF ?CdF&:(= ?CdF ?Ded)) (EREF ?Eref&:(<> ?DiamE ?Eref)))
 ?Chan <- (object (is-a CHANFRO) (PontoX ?PtX&:(> ?PtX (/ (+ ?CdI ?CdF) 2)))
                    (Posicao "Externo"))
=>
 (send ?Chan put-Sentido 0)
 (send ?Chan put-Flag 3)
)
```

APÊNDICE III
REGRAS DE MONTAGEM

Aqui são apresentados alguns exemplos das regras implementadas no sistema especialista com relação à base de conhecimento de montagem.

```
;*****************************
;* Modulo Regras para Montagem*
;*****************************
(defmodule REGRASMONTAGEM
 (import PECA defclass ?ALL)
)

;*********************************
;                  FUROS PASSANTES
;*********************************
;Analise diametral para Furo Cilindrico Passante
;*****
;NAO HA MONTAGEM SE FURO E EIXO NAO SAO CENTRADOS AXIALMENTE
(defrule REGRASMONTAGEM::Montag1
 (declare (salience 10))
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (<> ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (Diametro ?DiamE))
 ?Furo <- (object (name [FRCL0]) (Diametro ?DiamF&:(= ?DiamF ?DiamE)))
=>
 (send ?Furo put-Flag 1)
 (return)
)

;*******
;NAO EFETUA A MONTAGEM SE O EIXO ESTA TOTALMENTE FORA DO FURO - EIXO
;LOCALIZADO A ESQUERDA DO FURO
(defrule REGRASMONTAGEM::Montag2
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
```

```
                (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(> ?CdIF ?CdFE)) (CoordF ?CdFF&:(> ?CdFF ?CdFE))
        (Tipo "furocilindricopassante") (Sentido 1)  (Diametro ?DiamF&:(= ?DiamF ?DiamE)))
=>
 (send ?Furo put-Flag 1)
)

;*******
;NAO EFETUA A MONTAGEM SE O EIXO ESTA TOTALMENTE FORA DO FURO - EIXO
;LOCALIZADO A DIREITA DO FURO
(defrule REGRASMONTAGEM::Montag3
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
                (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordF ?CdFF&:(and (< ?CdFF ?CdIE)(< ?CdFF ?CdFE))); (CoordI
?CdIF&:(< ?CdIF ?CdIE))        (Tipo "furocilindricopassante") (Sentido 1)
                (Diametro ?DiamF&:(= ?DiamF ?DiamE)))
=>
 (send ?Furo put-Flag 1)
)

;*******
;HA MONTAGEM SE EIXO ESTA TOTALMENTE DENTRO DO FURO - FURO A DIREITA
(defrule REGRASMONTAGEM::Montag4
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
                (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(= ?CdIF ?CdIE)) (CoordF ?CdFF&:(=      ?CdFF ?CdFE))
        (Tipo "furocilindricopassante") (Sentido 1)  (Diametro ?DiamF&:(and (= ?DiamF ?DiamE)
        (>= (- ?CdFE ?CdIE) 2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 0)
)

;*******
;NAO HA MONTAGEM SE EIXO ESTA TOTALMENTE DENTRO DO FURO
;COMPRIMENTO DE APOIO INSUFICIENTE
```

```
(defrule REGRASMONTAGEM::Montag5
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
                  (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(= ?CdIF ?CdIE)) (CoordF ?CdFF&:(= ?CdFF ?CdFE))
          (Tipo "furocilindricopassante") ;(Sentido 1) (Diametro ?DiamF&:(and (= ?DiamF ?DiamE)
          (< (- ?CdFE ?CdIE)2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 2)
)

;*******
;MONTAGEM SE EIXO ESTA TOTALMENTE DENTRO DO FURO - COORDIF=COORDIE
(defrule REGRASMONTAGEM::Montag6
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
          (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(= ?CdIF ?CdIE))
          (CoordF ?CdFF&:(> ?CdFF ?CdFE))        (Tipo "furocilindricopassante") (Sentido 1)
          (Diametro ?DiamF&:(and (= ?DiamF ?DiamE)(>= (- ?CdFE ?CdIF) 2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 0)
)

;*******
; NAO MONTAGEM SE EIXO ESTA TOTALMENTE DENTRO DO FURO - COORDIF=COORDIE E
; COMPRIMENTO DE MONTAGEM INSUFUCIENTE
(defrule REGRASMONTAGEM::Montag7
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
          (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(= ?CdIF ?CdIE))
          (CoordF ?CdFF&:(> ?CdFF ?CdFE))        (Tipo "furocilindricopassante") (Sentido 1)
          (Diametro ?DiamF&:(and (= ?DiamF ?DiamE)(< (- ?CdFE ?CdIF) 2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 2) ;comprimento de montagem insuficiente
```

```
)

;*******
; MONTAGEM SE EIXO ESTA COM COORDF FORA DO FURO E COORDIE=COORDIF
(defrule REGRASMONTAGEM::Montag8
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
        (Comprimento ?CompE))
?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(= ?CdIF ?CdIE))(CoordF ?CdFF&:(< ?CdFF ?CdFE))
        (Tipo "furocilindricopassante") (Sentido 1)(Diametro ?DiamF&:(and (= ?DiamF ?DiamE)
        (>= (- ?CdFF ?CdIE) 2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 0)
)
;*******
;NAO MONTAGEM SE EIXO ESTA COM COORDF FORA DO FURO, COORDIE=COORDIF E COMPRI-
; MENTO DE MONTAGEM INSUFICIENTE
(defrule REGRASMONTAGEM::Montag9
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
        (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(= ?CdIF ?CdIE))
                (CoordF ?CdFF&:(< ?CdFF ?CdFE))        (Tipo "furocilindricopassante") (Sentido 1)
        (Diametro ?DiamF&:(and (= ?DiamF ?DiamE)(< (- ?CdFF ?CdIF) 2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 2) ;comprimento de montagem insuficiente
)

;*******
; MONTAGEM SE EIXO ESTA COM COORDI FORA DO FURO E COORDIF=COORDIE
(defrule REGRASMONTAGEM::Montag10
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
        (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(> ?CdIF ?CdIE))
        (CoordF ?CdFF&:(= ?CdFF ?CdFE))        (Tipo "furocilindricopassante") (Sentido 1)
```

```
        (Diametro ?DiamF&:(and (= ?DiamF ?DiamE)
        (>= (- ?CdFE ?CdIF) 2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 0)
)

;*******
; NAO MONTAGEM SE EIXO ESTA COM COORDI FORA DO FURO E COORDFF=COORDFE
(defrule REGRASMONTAGEM::Montag11
 ?Pec1 <- (object (is-a PECA) (CoordY ?CdY1))
 ?Pec2 <- (object (is-a PECA) (CoordY ?CdY2&:(and (neq ?Pec1 ?Pec2) (= ?CdY1 ?CdY2))))
 ?ECil <- (object (name [EXCL0]) (CoordI ?CdIE) (CoordF ?CdFE) (Diametro ?DiamE)
        (Comprimento ?CompE))
 ?Furo <- (object (name [FRCL0]) (CoordI ?CdIF&:(> ?CdIF ?CdIE))
        (CoordF ?CdFF&:(= ?CdFF ?CdFE))        (Tipo "furocilindricopassante") (Sentido 1)
        (Diametro ?DiamF&:(and (= ?DiamF ?DiamE)
        (< (- ?CdFE ?CdIF) 2))));(/ ?DiamF 2)))))
=>
 (send ?Furo put-Flag 2)
)
```