METODOLOGIA ORIENTADA A OBJETOS E SUA APLICAÇÃO EM CAD/CAM

Carlos César Tanaka
Maria Alice Grigas Varella Ferreira
Departamento de Engenharia de Computação e Sistemas Digitais
Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, n158 - Cidade Universitária - 05508-970
São Paulo. SP, Brazil
e-mail: catanaka@usp.br, malice@pcs.usp.br

ABSTRACT

Object-oriented models are considered the great promise for the development of software in the future. In this report, the authors investigated the benefits of applying object-oriented methodologies in CAD/CAM systems. They developed a feature recognition system using OMT - Object Modeling Technique - of Rumbaugh. Solids, whose features are recognized by the system, are modeled in a B-rep modeler. They reinforced the idea that features are not only very important, but also valuables keys for CAD/CAM integration. Nowadays, object-oriented methodology is considered the most suitable software methodology for features based representation development in CAD systems. In this report, the authors investigated the advantages, as well as the weakness of OMT methodology in this area, considering the possible results that can be extrapolated to other similar areas in Computer Graphics.

Keywords: Object-oriented methodology, Object Modeling Technique, CAD/CAM, feature recognition, Computer Graphics.

1. Introdução

O conceito de objeto não é novo, tendo aparecido no final da década de 60; no final dos anos 70, disseminou-se pelos vários campos da ciência da computação: sistemas operacionais, linguagens de programação, bancos de dados, e mais recentemente, os conceitos de filosofia, psicologia e representação de conhecimentos enriqueceram o modelo inicial[1]. A orientação a objetos é um paradigma que vem ganhando um espaço cada vez maior no desenvolvimento de sistemas e é considerado por Yourdon[22], uma das "balas de prata" da Engenharia de Software, permitindo melhorar os índices de produtividade na indústria de informática, bem como facilitar a implementação de sistemas complexos.

O paradigma da orientação a objetos constitui-se de métodos, ferramentas e processos, baseados em elementos que abstraem a realidade, conhecidos como objetos. A capacidade de abstração proporcionada é uma das características responsáveis pelas diversas vantagens apresentadas por este paradigma, e permite o modelamento de entidades do mundo real de maneira mais intuitiva, porque possibilita uma fase de análise mais livre e independente do processo de implementação. Uma metodologia deve ser suportada por ferramentas CASE adequadas e, no caso da orientação a objetos, tais ferramentas ainda são poucas e, na maioria das vezes, limitadas. Ao se escolher uma metodologia é importante levar-se em conta a existência destas ferramentas que suportarão o desenvolvimento do software. Assim, a existência de uma ferramenta pode condicionar a escolha da metodologia.

O aparecimento de linguagens orientadas a objetos levou, naturalmente, ao surgimento de um número muito grande de bibliotecas de classes que os implementadores podem utilizar para o desenvolvimento de programas. Algumas empresas tentam usufruir das vantagens oferecidas e migrar para a tecnologia de objetos, forçando a sua equipe de informática a utilizar tais ferramentas, esperando que os conceitos desta filosofia sejam facilmente assimilados. Entretanto, adotar uma tecnologia e compreender os conceitos que estão por trás dela são dois caminhos diferentes, que nem sempre conduzem ao mesmo destino. O sucesso ou insucesso do empreendimento depende, principalmente, da adoção de uma boa metodologia de desenvolvimento de projeto, e um número considerável de metodologias está disponível, atualmente.

Hoje em dia, o paradigma de orientação a objetos vem sendo usado intensamente em certas áreas da computação como, por exemplo, na construção de interfaces homem-máquina e no desenvolvimento de sistemas de tempo real. Estas áreas exigem metodologias poderosas, devido à necessidade de manipulação de eventos e/ou programação concorrente.

Desta forma, uma pergunta que se apresenta e': "Que metodologia deve-se escolher para uma dada classe de problema?".

Neste projeto, as potencialidades das metodologias de orientação a objetos foram investigadas para os sistemas de manufatura integrada ou CAD/CAM. Para isso, selecionou-se uma metodologia, a de Rumbaugh[16], que foi aplicada a um problema de reconhecimento de "features", dentro de um modelador geométrico, já existente, denominado BREULER[5].

OMT foi selecionada por ter sido considerada, no início da pesquisa, uma das metodologias mais completas. O objeto da pesquisa foi determinar os modelos mais importantes para a solução dos problemas desta classe de software.

2. CAD/CAM e "features"

O grande progresso industrial ocorrido a partir da Segunda Guerra Mundial teve como algumas de suas conseqüências a introdução dos computadores e das máquinas de controle numérico. Através da evolução e união desses dois elementos surgiu um ambiente de manufatura integrado conhecido como CAD/CAM ou Projeto e Manufatura Auxiliados por Computador.

O principal objetivo de sistemas CAD/CAM é o aumento da produtividade a baixo custo. Nesse sentido, esforços têm sido aplicados para se obter a integração do planejamento, projeto, manufatura, montagem e inspeção de produtos[10]. Entretanto, a ligação entre algumas dessas etapas é deficiente; é o que acontece durante a transmissão de informações de projeto, que são de natureza tipicamente geométrica, para as etapas posteriores, que envolvem informações de caráter contextual, intimamente relacionadas com as características da manufatura.

Wolf[21] afirma que os sistemas de projeto auxiliado por computador, CAD, tornaram-se tão grandes e complicados que passaram a exigir novas tecnologias que pudessem ser empregadas no seu desenvolvimento. Para superar os obstáculos de integração anteriormente citados, é necessário o desenvolvimento de uma tecnologia multidisciplinar que englobe gerenciamento de banco de dados, algoritmos e representações geométricas, interfaces homem-máquina e representação de conhecimentos, entre outras[11].

Em relação à integração entre o projeto e a manufatura, os *modelos baseados em "features"* começam a ser amplamente utilizados. "Feature" pode ser definida como uma região de interesse de um objeto, que contém informações geométricas e não geométricas, constituindo uma forma com significado em Engenharia como, por exemplo, furos e chanfros em Engenharia

Mecânica. O termo *"feature"* tem sido largamente utilizado e tem muitos significados. Este artigo usa a palavra "feature" como um sinônimo de *"shape-feature"*[7].

As pesquisas sobre "features" em sistemas de CAD estão divididas em três grandes áreas: 1) *Projeto baseado em "features"*, onde "features" pré-definidas são utilizadas na fase de Projeto para modelar diretamente o sólido; este método tem uma desvantagem: limita a criatividade do projetista; 2) *Identificação de "features"*, onde, após a modelagem do objeto, o projetista determina quais são as "features"; 3) *Reconhecimento automático de "features"*, que é o problema mais difícil, mas que apresenta as melhores perspectivas dentro da manufatura[3, 8, 9, 17, 18].

O reconhecimento automático de "features" baseia-se no fato das informações referentes a formato poderem ser deduzidas a partir da descrição geométrica. Isso permite que os modeladores geométricos presentes nos sistemas de CAD sejam usados. Este trabalho realizou o projeto e a implementação de um sistema de reconhecimento de "features", que foi acoplado a um modelador geométrico, como um *estudo de caso* para análise da metodologia de Rumbaugh no desenvolvimento de sistemas de CAD[19]. O sistema de reconhecimento de "features" foi escolhido porque o uso de metodologias orientadas a objeto parecia se adequar muito bem à solução desse tipo de problema[3,6,9,10,18,21].

3. Uma breve visão da metodologia de Rumbaugh

A metodologia OMT - Object Modeling Technique - tem como objetivo identificar e organizar os conceitos num domínio de aplicação, e para isso utiliza três modelos, que são detalhados durante as fases de Análise, Projeto e Implementação. Estes modelos são criados durante a Análise e refinados durante o Projeto e a Implementação.

O *Modelo de Objetos*[15] mostra os objetos da aplicação, as relações entre as classes, atributos e métodos, de uma forma estática. O diagrama gráfico - *Diagrama de Objetos* - é constituído pelo *Diagrama de Classes* e pelo *Diagrama de Instâncias*; este último diagrama é utilizado durante a fase de testes do sistema.

No Diagrama de Classes, cada classe é representada por um retângulo, dividido em três partes: identidade ou nome, estado e comportamento, similar aos que existem em muitas outras metodologias. O diagrama também apresenta as relações entre as classes: as *ligações* - representadas por linhas - são conexões entre as instâncias, e as *associações* - representadas por losangos - denotam grupos de ligações com estrutura e semântica semelhantes.

Existem alguns tipos especiais de associações, como no modelo de Entidade-Relacionamento para banco de dados[2]: a) *agregação*, que permite a composição de um objeto a partir de suas partes, descritas por outras classes; b) *generalização/especialização*, que são relações entre classes e superclasses[13], as classe herdam atributos e métodos das superclasses. Essa herança ocorre sob o controle do programador, que pode também adicionar novos atributos e métodos às classes. A Figura 1 mostra o Diagrama de Classe simplificado para o Sistema de Reconhecimento de "Features", onde os retângulos apresentam apenas o nome das classes. Os triângulos na figura indicam herança.

O *Modelo Dinâmico* mostra como os estados dos objetos são modificados quando ocorrem eventos no sistema; o objeto que recebe o estímulo pode alterar o seu estado e atuar em outros objetos. O comportamento dinâmico é descrito através do Diagrama de Estados. Há um diagrama de estados para cada classe que apresenta um comportamento dinâmico. Alguns eventos possuem

atributos, como os do "mouse" que necessitam indicar qual botão foi pressionado. A Figura 2 mostra um Diagrama de Estados para o Sistema de Reconhecimento de "Features".

Uma seqüência de eventos é conhecida como *Cenário*, e o *Diagrama de Cenários* é usado para mostrar estas transições. Este diagrama é muito importante para a realização dos testes no sistema. A Figura 3 mostra um Diagrama de Cenários para o Sistema de Reconhecimento de "Features".

O *Modelo Funcional*[14] é baseado no tradicional Diagrama de Fluxo de Dados - DFD[12] - que mostra como as informações de entrada são transformadas através do sistema, para gerar as informações de saída ou dados que devem ser armazenados.

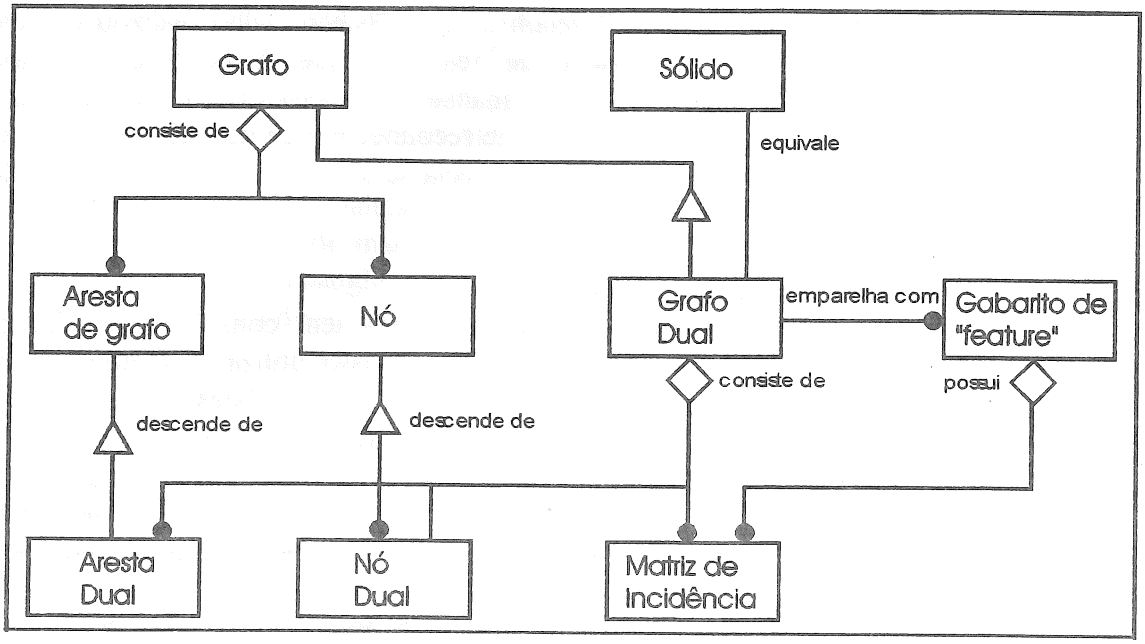

Fig. 1 Modelo de Objetos para o Sistema de Reconhecimento de "Features"

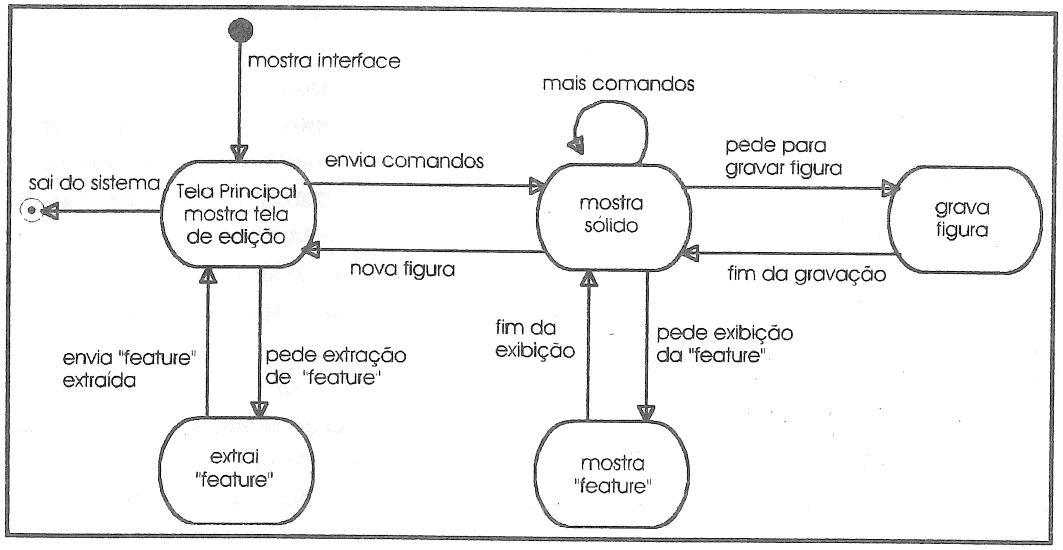

Fig. 2 Diagrama de Estados para o Sistema de Reconhecimento de "Features"

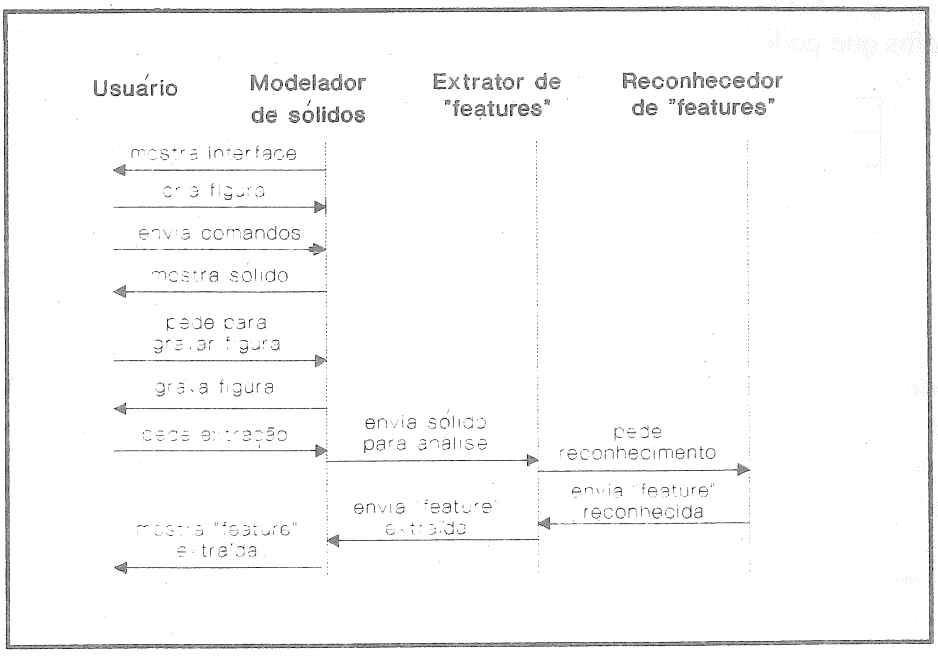

Fig. 3 Diagrama de Cenários para o Sistema de Reconhecimento de "features"

4. Aplicação da OMT no desenvolvimento de CAD

O processo de reconhecimento de "features" requer: 1) uma classificação hierárquica de "features", onde as "features" usadas como gabaritos serão armazenadas; 2) uma representação do objeto adequada para o processo de reconhecimento, e 3) um mecanismo de inferência que consiga reconhecer as "features" de maneira consistente. A OMT permite que esses três elementos sejam modelados independentemente, porque há pouco acoplamento entre eles.

A aplicação da OMT ao problema proposto é apresentada a seguir; a exposição é realizada de acordo com as três fases do ciclo de vida de desenvolvimento do software: Análise, Projeto e Implementação.

Análise

Para efeito de estudo, dividiremos o problema em três passos:

Passo 1 - Classificação de "features" - nesta fase, é necessário criar uma estrutura de dados para armazenar os atributos das "features" como nome, posição e orientação; esta estrutura de dados deve ser capaz de incluir novos dados e realizar associações. Essa classificação, é na verdade, um banco de dados, por isso o Modelo de Objetos é importante nessa fase. Como as "features" têm propriedades hierárquicas, elas podem ser classificadas através da propriedade de hierarquia dos objetos. Para isso, serão criadas classes de objetos genéricos e depois serão feitas especializações, de acordo com as necessidades; as classes resultantes serão chamadas de subclasses ou descendentes; estas classes herdam todas as características de suas ancestrais. A Tabela 1 mostra uma dessas hierarquias.

Passo 2 - Representação do Sólido a ser Reconhecido - Nesta etapa é abordada a questão dos modeladores geométricos; a Representação por Superfícies ou modelo B-rep tem sido amplamente utilizada no reconhecimento de "features", mas a Geometria Construtiva de Sólidos - CSG - também tem sido usada; a preferência pelo modelo B-rep é justificada pela existência de

muitos sistemas de CAD baseados nesse modelo e porque existe uma representação dual do sólido através de grafos que pode ser usada pelo algoritmo de reconhecimento.

"FEATURE"			
Cilíndrica		Poliédrica	
Passante	Não Passante	Passante	Não Passante
Canal cilíndrico	Cone	Furo	Fenda
Funil	Anel	Canal	Chanfro
Furo cilíndrico	Furo Cego	Degrau	Cavidade

Tabela 1 Uma hierarquia de "features"

No modelo B-rep, um sólido é descrito usando-se entidades de mais baixo nível tais como faces, arestas e vértices, vinculadas às relações topológicas. Estes elementos possuem entidades equivalentes num modelo de grafo: as faces são equivalentes aos nós e as arestas equivalentes às arestas do grafo. Assim, para o processo de reconhecimento e extração de "features", a representação B-rep de um sólido é transformada no seu grafo dual.

O modelamento dinâmico nessa fase é muito simples, uma vez que não há eventos externos. O Modelo Funcional descreverá o processo algorítmico de conversão do modelo B-rep para a representação por grafos.

Nessa fase, o Modelo de Objetos é o principal elemento de um sistema de CAD pelas seguintes razões: o modelo de objetos mostra todas as relações entre os elementos do problema, determina explicitamente todas as funções e suas localizações nas classes; estabelece o modelo básico que será refinado na fase de Projeto e define os subsistemas.

Algumas características importantes da orientação a objeto, como encapsulamento, modularidade e hierarquia, são apresentadas no Modelo de Objetos obtido nesta fase, e que se encontra na Figura 1.

Passo 3 - Mecanismo de Inferência - esta fase é a responsável pelo reconhecimento e extração de "features". Este passo também apresenta um modelo dinâmico simples, pelas mesmas razões do passo anterior. Por outro lado, o modelo funcional é extenso, uma vez que o processo de reconhecimento de "Features" é algorítmico. Muitas funções nesse processo devem ser incorporadas ao Modelo de Objetos apresentado na Figura 1. Essas funções constituirão o comportamento dos objetos. Durante o modelamento funcional, serão definidas algumas estruturas de dados que serão usadas pelos algoritmos de reconhecimento; elas serão incorporadas aos objetos e definirão seus estados internos. Esses estados internos não são acessíveis às outras classes.

A Tabela 2 apresenta as principais tarefas a serem desenvolvidas durante a fase de Análise.

Projeto

Os modelos definidos na fase de Análise serão refinados na fase de Projeto; a fase de Projeto é dividida em *Projeto do Sistema e Projeto de Objetos*.

O Projeto de Sistema é o estágio onde se define a arquitetura do sistema; para isso, devem ser consideradas as restrições e classes de objetos obtidas durante a Análise. Em sistemas de CAD, o Projeto de sistemas é muito importante para a definição de restrições e prioridades, de acordo com o ambiente de produção específico. Os passos básicos do Projeto de sistema são mostrados na Tabela 3.

MODELO	TAREFA
Objetos	Identificar classes e objetos
	Preparar um dicionário de dados
	Identificar associações entre objetos
	Identificar atributos e ligações de objetos
	Organizar e simplificar as classes de objetos usando herança
	Verificar as ligações e associações do diagrama
	Refinar o modelo se necessário
	Agrupar as classes em módulos
Dinâmico	Preparar cenários para seqüências de interação gerais
	Identificar eventos entre objetos
	Rastrear a ocorrência dos eventos em cada cenário
	Criar diagramas de estado para cada classe com comportamento dinâmico
	Verificar a consistência e ausência de eventos entre estados.
Funcional	Identificar os valores de entrada e saída
	Construir um diagrama de estados mostrando as dependências funcionais
	Descrever as funções
	Identificar as restrições
	Especificar critérios de otimização

Tabela 2 Os modelos da fase de Análise e suas principais etapas

Passo	Etapas do Projeto do Sistema
1	Organizar o sistema em subsistemas
2	Identificar as concorrências
3	Distribuir os subsistemas entre os processadores
4	Escolher métodos para gerenciamento das estruturas de dados
5	Identificar os recursos globais e determinar mecanismos de acesso a eles
6	Escolher um método para implementar o controle de software
7	Considerar as condições limites
8	Estabelecer prioridades para a distribuição de recursos

Tabela 3 Projeto de Sistema e suas principais fases

O segundo estágio é o Projeto de Objetos; nessa fase os objetos descobertos durante a Análise serão refinados; alguns deles serão rearranjados em novas classes e outros podem deixar de existir. As características dos objetos, como modularidade e encapsulamento, auxiliam a aplicação do paradigma "dividir para conquistar" no projeto de sistemas de CAD/CAM. As atividades desenvolvidas nessa etapa estão enumeradas na Tabela 4.

Implementação

Durante essa fase é realizada a tradução do projeto em código. A OMT não dá muita ênfase a este processo porque . considera que a codificação é um processo direto e quase automático, sem nenhuma decisão importante a ser realizada. Se algumas mudanças são necessárias nessa fase, seus efeitos terão influência local, afetando sempre uma pequena parte do programa.

5. Conclusões

As metodologias orientadas a objeto estão sendo intensivamente aplicadas devido à promessa de construção de software expansível e modificável. Apesar da grande contribuição da

Análise e Projeto Estruturados no desenvolvimento de software, essas metodologias não permitem expansões e modificações de programas facilmente. Nosso projeto utilizou um modelador geométrico existente para testar, também, estas características, constituindo o pacote de "features" uma extensão do mesmo sistema. O modelador utilizado havia sido desenvolvido pela metodologia estruturada tradicional.

Passo	Etapas do Projeto de Objetos
1	Determinar as operações das classes
2	Desenvolver algoritmos para implementar os métodos dos objetos
3	Otimizar o acesso aos dados
4	Implementar o controle de software
5	Ajustar a estrutura das classe para aumentar a herança
6	Projetar a implementação das associações
7	Determinar representações adequadas para os atributos dos objetos
8	Agrupar classes e associações em módulos

Tabela 4 Projeto de Objetos e suas principais fases

Com base no projeto de CAD baseado em "features" desenvolvido com a OMT, serão enunciadas a seguir, algumas conclusões sobre o uso desta metodologia em sistemas de CAD.

Abstração, encapsulamento e herança são algumas das mais importantes qualidades dos objetos quando uma metodologia deste tipo é aplicada no desenvolvimento de sistemas com "features". A dificuldade de transição entre a fase de Análise e a fase de Projeto é a grande deficiência dos métodos orientados a objetos. A característica de abstração garante aos objetos grande flexibilidade durante as fases de Análise e de Projeto, pois retarda a abordagem dos aspectos de implementação. A reutilização de objetos já existentes é fruto do encapsulamento e da herança, naturais nos objetos. O encapsulamento foi responsável, em grande parte, pelo sucesso da extensão feita ao modelador, pois evitou o conflito de nomes e os efeitos colaterais entre os procedimentos antigos e as "features" incorporadas.

Os métodos orientados a objetos se encaixam ainda melhor quando são utilizados no projeto de sistemas de CAD que manipulam "features". As "features" possuem uma classificação naturalmente hierárquica e bem definida; além disso a constituição interna de uma "feature" também pode ser descrita através de hierarquia, por exemplo, usando-se grafos. Estes aspectos das "features" garantem um bom mapeamento por objetos.

Os próprios sistemas de CAD são idealizados para serem modulares, e seus elementos são bem definidos. Isso também facilita o modelamento através de objetos. Pode-se induzir que os sistemas que puderem ser completamente descritos através de componentes e seus relacionamentos, serão ótimos candidatos a serem modelados através de objetos.

Entre outras vantagens do paradigma de objetos, em especial da OMT, estão :

- Adiamento das decisões relativas à implementação do sistema, permitindo maior flexibilidade ao analista e maior facilidade de alteração do projeto, pois muitas alterações estão em um nível de projeto mais elevado;
- A OMT permite a especificação detalhada de um sistema, já que possui um conjunto de ferramentas - Diagrama de Objetos, de Estado, de Fluxo de dados - bastante minuncioso. Estes modelos permitem a elaboração de documentação detalhada relativa a cada etapa

envolvida na metodologia, permitindo que alterações de projeto sejam sistematicamente incorporadas;

- A OMT permite um tratamento pormenorizado de sistemas que trabalham em tempo real, pois tem uma boa abordagem dinâmica. Esse, porém, não é o caso de sistemas de CAD que normalmente são sistemas mais estáticos;
- O modelamento por objetos facilita a divisão do sistema em subsistemas e módulos. A análise dos objetos e de seus relacionamentos fornecem indicações sobre as divisões a serem feitas;
- A determinação das operações - métodos - fica muito facilitada, pois elas são automaticamente deduzidas a partir dos Diagramas de Objetos, de Estados e de Fluxo de dados;
- A reutilização de classes e de objetos também deve ser evidenciada. O protótipo desenvolvido utilizou classes para a manipulação de listas, matrizes, pilhas, conjuntos, etc., fornecidas com o compilador, possibilitando dessa forma redução significativa de trabalho.

Entretanto, os métodos orientados a objetos possuem deficiências. As principais deficiências detectadas através deste trabalho, bem como algumas diretrizes para tentar resolver esses problemas estão listadas a seguir:

- Os métodos orientados a objetos atuais não apresentam boas soluções para realizar a compreensão do domínio do problema e, por conseguinte, atingir uma identificação adequada das classes. Embora criticado por autores puristas em relação ao paradigma de objetos, enquanto não há uma solução consagradamente melhor, pode-se utilizar a descrição textual do problema e a extração das informações importantes através de regras gramaticais, como é realizado em métodos estruturados e em algumas metodologias de Análise baseadas em objetos[4];
- Há uma grande diversidade de notações e nomenclaturas entre as metodologias orientadas a objeto existentes. Isso dificulta uma análise comparativa e, portanto, retarda o aproveitamento dos benefícios de cada método; a unificação em curso entre várias das principais metodologias poderá minimizar este problema a curto prazo;
- O modelo dinâmico da OMT é bastante vasto e dificulta a conferência de dados e a consistência. Como já foi dito, este problema não foi crítico no tratamento de CAD abordado, no qual a interface de comunicação homem-máquina já estava implementada. Em um sistema de CAD de produção intensiva e, eventualmente distribuído, esse ponto pode ser crítico;
- Durante a fase de levantamento das operações dos objetos, a OMT fornece pouca consistência com a manipulação de dados já determinados; seria interessante tentar indicar a composição dos parâmetros e resultados e fazer uma referência cruzada com os objetos e seus atributos já conhecidos. O ideal seria dispor-se de algum método ou linguagem que permitisse, durante a fase de Projeto, especificar algoritmos e estruturas de dados, que através de conversores convenientes pudessem ser traduzidos para código;
- O fato da OMT utilizar o diagrama de fluxo de dados pode induzir analistas que já usavam outros métodos não voltados a objetos a realizarem projetos sem características de objetos[20]. A solução nesse caso é fornecer um treinamento de transição envolvendo conceitos sobre objetos, bem como os fundamentos da metodologia que será empregada;
- Todos os métodos orientados a objetos deveriam possuir ferramentas CASE "inteligentes" que auxiliassem a restruturação das classes durante os sucessivos refinamentos. Isso porque

uma descrição inicial por objetos pode diferir muito da versão final. A maioria das ferramentas de mercado destinam-se apenas à automação gráfica específica sugerida pela metodologia, realizando às vezes algum tipo de verificação de consistência entre os vários modelos utilizados. O seu grau de assistência deixa muito a desejar.

AGRADECIMENTOS

Este trabalho foi desenvolvido com o apoio do CNPq - Brasil, entidade do Governo Brasileiro voltada ao desenvolvimento científico e tecnológico.

BIBLIOGRAFIA

1. BOOCH, G. **Object-oriented design with applications**. Redwood City, The Benjamin/Cummings, 1991.
2. CHEN, P. **A abordagem Entidade-Relacionamento para Projeto Lógico**. McGraw-Hill, 1990.
3. CHUNG, J.C.H.; PATEL, D.R.; COOK, R.L. SIMMONS, M.K. Feature-based modeling for mechanical design. **Computers & Graphics**, v.14, n.2, p.189-99, 1990.
4. COAD, P.; YOURDON, E. **Análise baseada em objetos**. Trad. da CTI Informática. 2. ed., Rio de Janeiro, Campus, 1992.
5. FONSECA, G. L. M. **Modelamento Geométrico usando os Operadores Topológicos de Euler**. São Paulo, 1993. Dissertação (Mestrado) - Escola Politécnica, Universidade de São Paulo.
6. JACOBSON, I. Is object technology software's industrial platform? **IEEE Software**, v.10, n.1, p.24-30, Jan. 1993.
7. KANG, TZONG-SHYAN; NNAJI, B.O. Feature representation and classification for automatic process planning systems. **Journal of Manufacturing Systems**, v.12, n.2, p.133-44, 1993.
8. MANTYLA, M. A modeling system for top-down design of assembled products. **IBM Journal Resource Development**, v.34, n.5, p.636-58, Sept. 1990.
9. MAREFAT, M.; KASHYAP, R.L. Automatic construction of process plans from solid model representations. **IEEE Transactions on Systems, Man and Cybernetics**, v.22, n.5, p.1097-115, Sept./Oct. 1992.
10. MAREFAT, M.; MALHOTRA, S.; KASHYAP, R.L. Object-oriented intelligent computer-integrated design, process planning and inspection. **IEEE Computer**, v.26, n.3, p.54-65, Mar. 1993.
11. MILLER, J.R. Architectural issues in solid modelers. **IEEE Computer Graphics and Applications**, v.9, n.5, p.72-87, Sept. 1989.
12. PRESSMAN, R.S. **Software Engineering:** A practitioner's approach. New York, McGraw-Hill, 1987.
13. RUMBAUGH, J. A matter of intent: how to define subclasses. **Journal of Object-Oriented programming**, v.9, n.9, p.5-9, Sept. 1996.
14. RUMBAUGH, J. OMT: the functional model. **Journal of Object-Oriented programming**, v.8, n.3, p.10-4, Mar./Apr. 1995.
15. RUMBAUGH, J. OMT: the object model. **Journal of Object-Oriented programming**, v.8, n.1, p.21-7, Jan. 1995.
16. RUMBAUGH, J.; BLAHA, M.; PREMERLANI, W.; EDDY, F.; LORENSEN, W. **Object-Oriented Modeling and Design.** New Jersey, 1991.
17. SAKURAI, H.; GOSSARD, D.C. Recognizing shape features in solid models. **IEEE Computer Graphics and Applications**, v.10, n.5, p.22-32, Sept. 1990.
18. SALOMONS, O.W.; van HOUTEN, F.J.A.M.; KALS, H.J.J. Review of research in feature-based design. **Journal of Manufacturing Systems**, v.12, n.2, p.113-32, 1993.
19. TANAKA, C.C.; Metodologia orientada a objetos e sua aplicação em sistemas de CAD baseado em features. São Paulo, 1997. Dissertação (Mestrado) - Escola Politécnica, Universidade de São Paulo.
20. TRAVASSOS, G.H. Desenvolvimento orientado a objetos. In: II Workshop Orientação a Objetos, Rio de Janeiro, 1996. Minicurso. Rio Janeiro, Engenharia de Sistemas de Computação, COPPE/UFRJ.
21. WOLF, W. Object-oriented programming for CAD. **IEEE Design and Test of Computers**, v.8, n.1, p.35-42, Mar. 1991.
22. YOURDON, E. **Decline & Fall of the American Programmer**. Prentice-Hall, New Jersey, 1993.