XIV Conferencia Latinoamericana de Informática
17avas. Jornadas Argentinas de Informática
e Investigación Operativa
Buenos Aires, Setiembre 1988.

GERENCIAMENTO DE ESQUEMAS DE DADOS DINAMICAMENTE MODIFICAVEIS EM MODELO DE BASE DE DADOS ORIENTADOS A OBJETOS.

Caetano Traina Júnior - Jan Frans Willem Slaets
ICMSC - USP - São Carlos IFQSC - USP - São Carlos
Av. Dr. Carlos Botelho, 1465 - Fone (0162) 72-4496
C.E.P. 13.560 - São Carlos - Est. São Paulo - Brasil

Resumo

Este trabalho apresenta um modelo de dados orientado a objetos, desenvolvido para suportar a construção de Sistemas de Gerenciamento de Dados para Aplicações de Projeto, permitindo a representação de grande conteúdo semântico. O modelo favorece a construção de Meta-Sistemas, onde a especificação dos requisitos da propria aplicação deve ser mantida na Base de Dados. Atraves de conceitos que habilitam o modelo a gerenciar a manipulação de versões e alternativas de projeto, cria-se o suporte para que a manutenção de esquemas de dados possa ser feita dinamicamente, simultaneamente com a operação normal do SGBD, sem impacto sobre os dados já armazenados.

1. - Introdução

Os Sistemas de Gerenciamento de Bases de Dados (SGBD) disponiveis atualmente foram em sua grande maioria desenvolvidos tendo por objetivo atender a aplicações comerciais, em que a tônica consiste na manipulação de um volume muito grande de dados estruturalmente idênticos. Por outro lado, aplicações não comerciais dos sistemas de computação (não convencionais para os SGBDs), com freqüência manipulam dados que não tem aquela mesma homogeneidade estrutural. Nesse último caso encontram-se os sistemas de Projeto Apoiado por Computador - PAC (CAD), Fabricação Apoiada por Computador - FAC (CAM), Sistemas de Automação de Escritório, Inteligência Artificial, etc.

Assim, têm sido pesquisados modelos para SGBDs que possam dar o suporte necessário a um amplo espectro de aplicações não convencionais [HAM_81], [FAR_85], [EAR_85] [ABI_87], e portanto que possam ser empregados em sistemas que devam ser integrados em sistemas maiores. Levantamentos de suas necessidades têm sido feitos e apresentados, tal como em [MAT_81], [HAR_85], [BIC_86] e [NAV_86], tendo-se tornado comum o termo **Sistema de Gerenciamento de Dados para Engenharia (SGDE)** para designá-los. Uma das principais necessidades corresponde à possibilidade de se poder alterar o esquema da base de dados mesmo depois desta já ter sido inicializada, uma vez que uma base de dados para projeto tem o seu esquema em constante evolução junto com o proprio projeto.

Pode entender-se um **Meta-Sistema** como um sistema que usa uma estrutura semelhante à sua propria estrutura para definir-se a si mesmo. Por exemplo, uma linguagem que ensine ou especifique uma linguagem será uma Meta-Linguagem; um programa de computador que possa ser definido, feito ou melhorado quando se usa esse mesmo programa será um Meta-Programa. Um Meta-Sistema de Gerenciamento de Dados deve poder aceitar a definição do Esquema de Dados da

mesma maneira que aceita os dados, pois o proprio esquema de dados deve ser encarado como um conjunto de dados que deve ser gerenciado. Tal enfoque implica que o próprio esquema possa ser modificado ou redefinido dinâmicamente, tal como o podem ser os dados extensionais armazenados na Base de Dados.

Neste trabalho apresenta-se um modelo de dados denominado **Modelo de Representação de Objetos - MRO** [TRA_88] que permite a construção de SGDEs que atendem a um amplo espectro de aplicações, incluindo aplicações integradas de projeto e produção de sistemas em geral, e em especial que permite a construção de Meta-sistemas de Gerenciamento de Dados [TRA_86], em que a especificação dos requisitos da propria aplicação devam ser mantidos na Base de Dados. Nas seções seguintes os conceitos principais que caracterizam o MRO são apresentados, com ênfase na maneira como tal modelo permite a operação do SGBD nele apoiado mesmo em face de um esquema de dados dinamicamente em alteração.

2. - A Fundamentação do Modelo

A fundamentação matemática de modelos comerciais mais recentes, tais como o modelo Relacional e o Modelo Entidade Relacionamento (ME-R), é a álgebra de relações [NG*_81], [PAR_85]. Esta é adequada para a manipulação de estruturas de dados que possam ser bem representadas através de tabelas. No caso de um modelo de dados voltado à representação semântica mais intensa, apoiado na modelagem de objetos, tal como o apresentado nesse trabalho, a teoria dos grafos torna-se no entanto mais atraente. A Teoria dos Grafos é mais adequada, uma vez que o reconhecimento dos objetos como nós de um digrafo corresponde exatamente à ideia de que todos os objetos, independentemente de seus tipos, tem a mesma estrutura, tal como todos os nós de um digrafo tem a mesma estrutura funcional.

O Modelo de Representação de Objetos procura modelar objetos (entes, eventos, coisas, conceitos), tratando cada um individualmente, e mantendo na representação de cada um todas as informações pertinentes sem tentar estabelecer, a priori, padrões para a representação de conjuntos de objetos, tal como no Modelo Entidade-Relacionamento, onde a representação das informações é feita na realidade através das relações de dados das entidades e dos seus relacionamentos, especificados no esquema de dados de forma imutável depois da inicialização da base de dados.

Para que se possa usar um digrafo para a representação de informações, deve-se associar as informações aos elementos que constituem o digrafo. Assim, os nos são a representação dos objetos que existem no mundo real, e os arcos representariam as associações que existem entre eles, as quais no MRO são denominadas relacionamentos. Assim, da mesma maneira que um nó corresponde aos pontos de identificação (e portanto de acesso) de um digrafo, um objeto centraliza todas as referências e informações (inclusive de identificação) que são mantidas no sistema.

A categorização dos nós de um digrafo classifica os objetos atribuindo-se um **Tipo** para cada objeto. Dessa forma, a estrutura de todos os objetos é idêntica (como o são todos os nós de um digrafo), e a classificação segundo sua categoria e um atributo obrigatoriamente associado a todos os objetos. O mesmo e feito quanto ao conceito de ponderação de arcos na teoria dos grafos,

que passa a classificar os relacionamentos entre os objetos atribuindo a todo relacionamento um "**tipo**".

Assim, no MRO os objetos e os relacionamentos são agrupados segundo seus respectivos tipos, e não em conjuntos de entidades e de relacionamentos. Isso permite que na implementação de um SGBD apoiado no MRO todos os relacionamentos e todos os objetos sejam armazenados em apenas um arquivo de relacionamentos e um arquivo de objetos.

Cada relacionamento do MRO indica que os dois objetos estão associados um ao outro, ou seja, no digrafo correspondente, devem existir dois arcos, um em cada direção. Isso não significa que o digrafo seja simétrico, pois cada um dos arcos representa uma forma diferente de associação entre entidades. Efetivamente, cada arco representa um **relacionamento oposto** ao representado pelo seu par. No MRO todo relacionamento tem o seu oposto correspondente.

Os **atributos** que são associados aos objetos e relacionamentos no mundo real são também classificados no MRO segundo seus vários tipos.

Os objetos podem ser diretamente acessados através de seu identificador, porém relacionamentos e atributos somente são acessados através das associações que mantêm com os objetos. Além disso, os atributos de relacionamento somente são acessados através dos relacionamentos aos quais estão associados, os quais por sua vez devem ser acessados através dos objetos que, esses sim, podem ser diretamente acessados (no entanto objetos podem também ser acessados através dos relacionamentos que mantém com outros).

2.1. - Um Diagrama de Representação de Objetos

Para a especificação de um esquema de dados usando um determinado modelo de representação, o uso de convenções gráficas é freqüentemente proveitoso, pois facilita a compreensão das informações. Portanto é interessante que se disponha de uma simbologia adequada, capaz de representar graficamente os recursos e conceitos do MRO. Assim, quando alguma informação é modelada usando-se o MRO, ela pode ser graficamente representada através de um **Diagrama de Representação de Objetos - DRO.**

Sendo apoiado na Teoria dos Grafos, os símbolos usados nos DROs são provenientes das representações gráficas tradicionalmente usadas para grafos, usando-se círculos para representar objetos, e arcos de circunferências para representar relacionamentos.

No entanto, em geral os diagramas elaborados não têm a intenção de representar instâncias de objetos ou de relacionamentos, mas sim de representar como os tipos de objetos e tipos de relacionamentos se associam. Portanto, é interessante que se distinguam dois tipos de diagramas: um que corresponde aos diagramas onde se representam os tipos de objetos e tipos de relacionamentos, os Diagramas de Representação de Objetos (DRO) propriamente ditos; e um outro em que se representam instâncias de objetos e relacionamentos, em geral com o propósito de servirem de exemplos, denominado **Diagrama de Representação de Instâncias - DRI.**

Na figura 1a) mostram-se os símbolos usados para a representação gráfica de Tipos de Objetos e de Tipos de Relacionamentos, e na figura 1b) mostram-se os símbolos usados para a representação de Objetos e de Relacionamentos.

Os Tipos de Objetos são representados através de um Círculo dividido ao meio. Na metade superior indica-se o Nome do Tipo de Objeto representado, e na metade inferior indica-se o Nome do Tipo de Colônia onde aquele tipo habita (Colônia e Habitação são conceitos descritos na Seção 4.).

Os Tipos de Relacionamentos são representados através de arcos de circunferências ligando os tipos de objetos que são associados por aquele tipo de relacionamento. Além disso, cada tipo de relacionamento é unido por um traço com o seu tipo de relacionamento oposto. Esse traço deve formar um ângulo agudo com cada um dos arcos, e o Nome do Tipo de cada relacionamento será escrito a partir desse ângulo, dando a indicação de qual é o nome de cada um dos dois tipos de relacionamento.

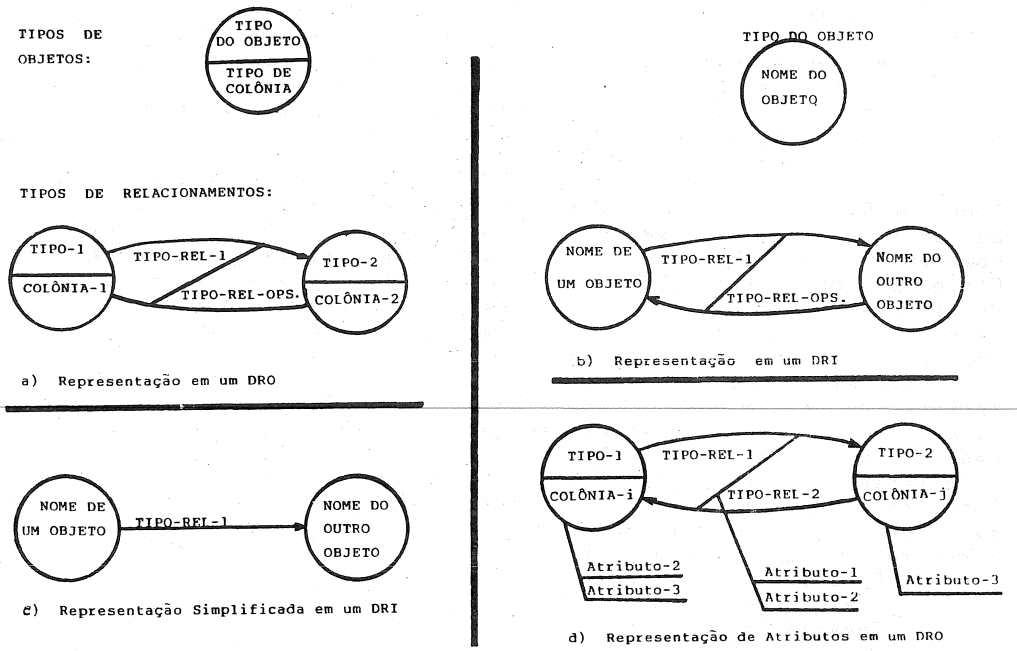

Figura 1 - Representação gráfica de peças do MRO: em a) estão os símbolos das peças em um DRO; em b) os símbolos das peças em um DRI; em c) um DRI simplificado; em d) atributos associados a objetos e relacionamentos em um DRO.

A principal diferença gráfica entre um DRI e um DRO é que os objetos são representados nos DRIs através de círculos não cortados. No interior do círculo escreve-se o nome do objeto. Se necessário, o tipo do objeto é escrito fora, acima do círculo. Em um DRI não se indica o nome da Colônia onde o objeto habita. Os relacionamentos são representados da mesma maneira que nos DROs.

Para efeito de simplificar os DRIs, os relacionamentos opostos podem ser omitidos nesses diagramas, uma vez que os tipos opostos podem ser deduzidos trivialmente através dos DROs e dos tipos dos objetos e do relacionamento envolvido. A figura 1c) mostra o mesmo relacionamento da figura 1b) representado de maneira simplificada.

Os Tipos de Atributos que podem ser associados aos tipos de objetos e de relacionamentos podem também ser expressos nos DRIs e DROs. A figura 1d) mostra um exemplo de como representam-se gráficamente associações de diversos tipos de atributos a tipos de objetos e de relacionamentos.

3. - Características de Tipos de Atributos

Para aumentar o conteúdo semântico do modelo, e facilitar o desenvolvimento dos sistemas que usem SGDEs nele apoiados, identificam-se características comuns em grupos de tipos de atributos, de forma a possibilitar que um SGDE ofereça maiores recursos de tratamento dos tipos de atributos de significado mais usados. Note-se que Tipos de Atributo e Tipo de Dado de um Tipo de Atributo são conceitos distintos.

Para isso deve-se caracterizar o significado de determinados Tipos de Atributos, que possam ser reconhecidos e individualizados para o SGDE. Por exemplo, em uma aplicação de PAC um atributo comumente usado é o que se refere às coordenadas de posição. Atributos desse tipo podem ser representados através de um tipo de dado Vetor_Real_de_Três_Posições, porém apesar de ser descrita por três valores reais, coordenada_de_posição é mais significativo do que indicar esses três valores como apenas números.

Existem vários tipos de atributos cujo significado pode ser reconhecido por um SGDE. Cada significado que possa ser reconhecido terá uma **Característica de Tipo de Atributo**. Dessa forma, a cada Tipo de atributo estará associado um Tipo de Dado e uma Característica, os dois constituindo o Significado associado ao Tipo de Atributo. Entre as Características de Tipo de Atributo mais comuns a um Sistema de Gerenciamento de Dados de Engenharia podem ser citadas as seguintes:

- **Sinônimo.** Corresponde a um identificador alternativo para um objeto;
- **Comentário.** Corresponde a informações para o usuário, que não são analisadas pelo sistema;
- **Posição.** Corresponde à informação de localização no espaço, podendo indicar um ponto, uma orientação, ou ambos.
- **Tempo.** Indicação de tempo relativo ou absoluto associado aos objetos e relacionamentos.
- **Conjunto de Dados.** São agrupamentos de informações semelhantes, que normalmente são representados em estruturas de arranjos de dados (vetores e matrizes).
- **Procedimento.** São fórmulas ou parâmetros associados a atividades que devem ser exercidas de uma forma pré-determinada pelo sistema.
- **Regra.** São atributos que possuem uma condição e uma ação associados.

Tipos de Atributos que não tenham alguma característica especial que os distingua dos demais são considerados tipos de

atributos normais, referenciados simplesmente como **Propriedades** dos objetos ou relacionamentos a que estão associados. As características dos tipos de atributos representados em um DRO (ou dos atributos representados em um DRI) podem ser indicadas precedendo o nome do tipo do atributo por um sinal que o caracterize, tal como indicado na lista de características de atributo acima.

4. - Escopo de Abrangência dos Identificadores de Objetos

Em Modelos de dados fundamentados na Álgebra das Relações o atributo (ou os atributos) usado para identificar uma tupla deve ser capaz de identificá-la univocamente no conjunto a que pertence. Tuplas pertencentes a conjuntos distintos podem ter um identificador comum sem que isso represente um problema. O mesmo vale para os identificadores de relacionamentos. Se um atributo não é suficiente para identificar univocamente uma determinada tupla, combinam-se mais atributos, para chegar a um conjunto de atributos que constitua uma chave de acesso que permita identificar univocamente cada tupla do conjunto.

Já no MRO obrigatoriamente apenas um campo deve identificar cada objeto, e como todos os objetos são mantidos em uma única estrutura de dados, conclui-se que cada objeto deveria ter um Nome (e tantos Sinônimos quanto necessário) que fosse único em toda a Base de Dados.

Para evitar essa restrição, o modelo permite restringir o escopo de abrangência de um identificador usado para localizar um objeto.

Quando visto de maneira global, um sistema apresenta um pequeno grau de detalhe. Quando o interesse recai em uma de suas componentes, o grau de detalhe aumenta, porém toda a informação que então se apresenta refere-se diretamente àquela componente que está sendo observada.

Quando se manipula a base de dados de um ponto de vista global, são poucos os objetos acessíveis. Indicando-se ao sistema que se pretende considerar a seguir um desses objetos, indica-se uma região da Base de Dados em que se pretende obter ou manipular informações. Dessa forma tornam-se acessíveis novos objetos. Esse procedimento pode ser repetido, indicando-se para consideração um dos novos objetos tornados disponíveis, dessa forma aumentando-se sucessivamente o nível de detalhe, e correspondentemente a quantidade de objetos a que se tem acesso diretamente através de seus Nomes (ou Sinônimos).

Cada vez que um objeto é colocado em consideração, aumenta-se um nível de detalhe, o que torna disponíveis os objetos que detalham mais o objeto considerado. Note-se que com o refinamento de um determinado aspecto, os demais não deixam de estar acessíveis, apenas não são por sua vez refinados.

O MRO conta com essa capacidade devido aos conceitos de Colônia de Objetos, Objeto Considerado, Objetos Habitantes de uma Colônia, e Contexto.

Quando se procura acessar um objeto armazenado numa base de dados apoiada no MRO, se for conhecido o seu código (a indicação univoca de um nó de um digrafo) então o acesso é imediato, uma vez que o código identifica univocamente um objeto dentre todos os objetos da base de dados. No entanto, quando a identificação é

feita pelo nome (ou por um sinônimo), essa identificação não é unívoca, uma vez que pode (a princípio) existir mais de um objeto com um mesmo nome. Para que se possa identificar um objeto pelo seu nome univocamente, é necessário que se especifique o **Contexto** onde esse nome está definido. A caracterização da existência de contextos é feita através da partição dos identificadores existentes na base de dados em **Colônias**, de maneira que um nome esteja definido somente no domínio de uma Colônia.

Cada **Colônia de Objetos** é constituída por um conjunto de objetos, e se refere não somente a um nível específico de detalhe de informação, mas também a um determinado aspecto de interesse, associado ao objeto do qual se deseja maiores detalhes.

Para isso, os objetos de uma Base de Dados devem ser classificados segundo diferentes aspectos, sendo essa classificação feita usando os diferentes Tipos de Objetos do Esquema de dados. Dessa forma, cada objeto pertence a apenas uma Colônia de Objetos, a qual é determinada através do seu tipo de objeto, e é constrita pelo objeto que caracteriza o aspecto particular do qual esse objeto é um detalhe.

O conceito de Colônias de Objetos divide os objetos de uma Base de Dados em vários aspectos de interesse, que são definidos pelos Tipos de Objetos que compõem o Esquema de Dados daquela base. O conceito estabelece portanto uma partição também no conjunto de Tipos de Objetos, sendo cada conjunto de Tipos de Objetos denominado **Tipo de Colônia**. Os objetos da base de dados estarão também particionados segundo os seus tipos, em correspondência com a partição que os Tipos de Colônias estabelecem sobre o conjunto de todos os Tipos de Objetos definidos na base. No entanto, além de estarem particionados segundo seus tipos (através dos Tipos de Colônias), os objetos da base de dados estarão separados também pelas várias Colônias de Objetos, uma vez que podem existir várias Colônias de Objetos correspondendo a um mesmo Tipo de Colônia. Os objetos que pertencem a uma colônia são ditos **habitantes** dessa colônia. Cada objeto habita exatamente uma colônia.

Os objetos que habitam uma colônia, tornam-se acessíveis quando a colônia a que pertencem é **aberta para acesso**. Num dado instante somente uma colônia de cada tipo pode estar acessível de cada vez. Assim, das várias colônias de objetos que correspondem a cada tipo de colônia apenas uma pode estar aberta num dado instante. Pode-se abrir uma colônia para permitir que os objetos se tornem acessíveis para consulta e modificação ou apenas para consulta.

Uma das colônias terá que estar permanentemente disponível, e será denominada **Colônia Global**. Os objetos da Colônia Global são os que estão disponíveis quando se está no nível menos detalhado de acesso ao sistema, e permanecem sempre disponíveis. Correspondentemente existe um **Tipo de Colônia Global**, o qual contém os tipos de objetos que existem na Colônia Global. Dessa forma, todos os objetos dos tipos que pertencem à Colônia Global estarão permanentemente disponíveis.

Objeto Considerado é um objeto que se tem presentemente disponível, e que indica-se para consideração, o que faz com que os objetos que o especificam com maior grau de detalhe se tornem também disponíveis. Um objeto está disponível quando a Colônia de Objetos em que ele habita está aberta.

Indicar um objeto para consideração caracteriza uma operação

de Abrir uma Colônia de Objetos para acesso, tornando disponíveis os objetos que habitam aquela colônia. Os objetos que habitam essa colônia constituem-se na **Colônia de Objetos Constrita pelo Objeto Considerado**. Quando considera-se um objeto, indica-se implicitamente uma ou mais colônias de objetos. A indicação de um objeto para consideração tem por objetivo abrir novas colônias de objetos para acesso, e a indicação de um objeto pode abrir mais de uma colônia. Dessa forma, o Conjunto de Objetos Constritos por um objeto pode envolver uma ou mais Colônias de Objetos.

Quando indica-se para consideração um objeto que constringe uma Colônia de Objetos correspondente à um Tipo de Colônia já aberto, este é **fechado**, o que faz com que se feche a Colônia de Objetos correspondente que estava aberta; a seguir o mesmo Tipo de Colônia é novamente aberto, agora tornando disponíveis os objetos da Colônia de Objetos constrita pelo Objeto Considerado agora. Outros objetos, pertencentes à colônias que não aquela focalizada implicitamente pela indicação de um novo objeto para consideração, e a correspondente operação de abertura de colônia, não são afetados por essa operação.

Se o usuário desejar referir-se a qualquer dos objetos que habitam qualquer uma das colônias abertas, o Nome do Objeto e a indicação de seu Tipo será suficiente para identificá-lo. No entanto, se for necessário referir-se a algum objeto que habita uma colônia não aberta, ou a um objeto que habita uma colônia de objetos constrita por um objeto que não o presentemente considerado, então será necessário que se identifique todo o caminho de acesso, desde o ponto de interesse atual na base de dados até o ponto onde está o objeto a que se quer se referir.

Com esses conceitos, o Modelo de Representação de Objetos passa a permitir que objetos que pertencem a Colônias de objetos Constritas por objetos distintos tenham nomes iguais, e mesmo assim possa manter-se a sua identificação inequívoca.

5. - O Controle de Versões e Alternativas de Projeto

O projeto de um produto de engenharia é um processo de refinamento sucessivo, em que os conceitos imaginados são transformados em objetos reais, as especificações genéricas são detalhadas, o todo é analisado em partes até os menores detalhes. No processo de projeto, o engenheiro parte de especificações genéricas ou de produtos realizados anteriormente, e vai sucessivamente incorporando novos dados, ou modificando partes já definidas anteriormente. Uma ferramenta de apoio à projetos deve verificar a consistência das novas informações e das mudanças efetuadas, comparar com as restrições impostas ao projeto, usando todas as informações presentes na Base de Dados que sejam correlatas a cada passo do processo.

Muitas versões de um projeto podem existir simultaneamente, e essa característica impõe ao SGDE a capacidade de poder armazenar e gerenciar múltiplas versões e as alternativas propostas para um projeto. Katz [KAT_84] [KAT_86] caracteriza "Versão" e "Alternativa" de um projeto considerando que um projeto está inicialmente em progresso, e que sobre esse projeto os projetistas efetuam mudanças e incorporação de dados. As alterações feitas para experimentar outras formas de implementação são **Alternativas**, e as alterações que são oficialmente aprovadas para

fazer parte do produto final são chamadas **versões**. Note-se que um projeto pode ter muitas versões e muitas alternativas, sendo que as versões produzidas são armazenadas para sempre.

O problema de controle das versões produzidas por um processo de desenvolvimento de software tem sido estudado sob a denominação de Gerenciamento de Configuração de Software [BER_84]. Sob esse enfoque, a menos de não se cuidar do conceito de "Alternativa", trata-se do mesmo problema, restrito ao desenvolvimento de software. O Gerenciamento de Configuração de Software, ataca problemas semelhantes aos de gerenciamento de configurações em geral, os quais foram re-estudados sob o enfoque das necessidades de configurações (e versões) de projeto de software. No entanto, os conceitos desenvolvidos podem ser aplicados ao Gerenciamento de Configurações de Projetos em geral, e serão aplicados neste trabalho como um recurso de auxílio ao gerenciamento e controle de desenvolvimento de projetos em geral, armazenados em uma base de Dados suportada pelo MRO.

Note-se que o Gerenciamento de configuração de projetos é muito mais semelhante ao Gerenciamento de Configuração de Software do que ao Gerenciamento de Configuração em geral, uma disciplina bastante estruturada que se aplica ao estudo do gerenciamento de bens em geral que possam ser produzidos. Isso porque um projeto - quando pensado em termos de sua concepção, motivos, decisões, etc, e em especial quando é constituído dos dados armazenados em uma base de dados - é muito mais semelhante ao software, no que diz respeito a maneira como deve ser gerenciado. Assim, as alterações nas regras de Gerenciamento de Configuração em geral, que tem sido estudadas para adequá-las ao Gerenciamento de Configuração de Software, continuam válidas para o Gerenciamento de Configuração de Projetos, mesmo que o bem final produzido por um dado projeto seja mais adequadamente controlado pelas regras de Gerenciamento de Configuração em geral.

As diversas perspectivas dos usuários são contempladas pelo MRO, implementado-se o esquema de proteção e/ou ocultamento de informações pelo controle de acesso às Colônias de Objetos. O acesso a Colônias de Objeto pode ser controlado de forma a que quando uma colônia é aberta, permite-se abrí-la apenas para consulta ou também para atualização, permitindo assim que a proteção seja feita a nível de colônias. Para obter acesso aos dados de interesse, os programas de aplicação precisam informar quais colônias devem ser abertas e quando devem ser fechadas.

Esse esquema de proteção tem uma correspondência direta com as operações de alto nível dos programas de aplicação, uma vez que se reflete nas operações solicitadas pelos usuários do programa de aplicação. Por exemplo, se um usuário estiver usando um editor para modificar o conteúdo da base de dados, ele provavelmente estará trabalhando com apenas parte da base toda, restrita a alguma colônia de objetos local ao seu ponto de interesse num dado instante. Com a proteção atuando a nível de colônias, a colônia onde ele estiver trabalhando estará protegida do acesso por qualquer outro usuário, ao mesmo tempo em que não poderá ter acesso a uma área de interesse sendo acessada por outro usuário.

Em operação normal, qualquer usuário tem acesso para leitura na colônia global, e indicando sucessivamente vários objetos para consideração, várias colônias vão se tornando acessíveis. Ao se indicar cada objeto para consideração, indica-se também se a colônia que o objeto considerado abre deve ser aberta para apenas

leitura ou para leitura e escrita.

Necessariamente, todos os usuários deverão ter acesso à colônia global para que qualquer outra colônia possa ser acessada. Porém, se for considerado que em uma base correspondente a um projeto já adiantado, envolvendo vários usuários, a colônia global já deverá estar definida e razoavelmente estática, então qualquer usuário necessitará ter acesso a ela apenas para consulta, permitindo que muitos usuários compartilhem a consulta à colônia global, e apenas em colônias em nível hierárquico mais baixo cada usuário obterá permissão individual para atualizações. A atualização da colônia global será permitida apenas a usuários privilegiados, tal como o gerente do projeto, o qual para atualizá-la poderá requerer que naquele instante ninguém mais tenha acesso à qualquer porção da Base de Dados.

O MRO Suporta a armazenagem das várias versões e alternativas de um projeto permitindo a geração de **Instâncias de Colônias de Objetos** dentro da mesma base de dados, de maneira que uma colônia constrita por um objeto possa ter várias versões e alternativas de projeto. Cada instância é chamada de uma **Variante**, e pode corresponder à uma Versão ou à uma Alternativa, e terá atributos de acesso e proteção independentes das demais instâncias do mesmo objeto, de maneira que cada uma poderá ser acessível por um usuário de maneira independente das demais.

Considere-se o desenvolvimento de um determinado aspecto de um projeto. Isso normalmente é efetuado por um projetista usando algum tipo de editor interativo, que lhe permite ir colocando novos dados, alterando os já existentes, até que o projetista fica satisfeito, e aquele aspecto do projeto é considerado terminado. O fato de aquele aspecto do projeto requerer atenção especial por parte de um projetista caracteriza que esse aspecto deve ser um **ítem de Configuração de Projeto (ICP)**, que deve ter atenção por parte do **Gerenciador de Configuração do Projeto**. Esse aspecto irá então constituir um ICP, e poderá ou não envolver outros ICPs.

Para ser um ICP, aquele determinado aspecto foi identificado como tal, e lhe foi atribuído identificação e um **registro de histórico** para acompanhamento, o qual indica continuamente seu estado para acesso. A identificação corresponde no MRO à existência de um objeto, que constringe a colônia de objetos que o descreve. Para que pudesse ser acessado pelo projetista, seu registro de histórico deveria estar indicando que ele estava em desenvolvimento, e a partir do instante em que o projetista passasse à edição indicando aquele objeto para consideração, seu registro de histórico indicaria o fato, evitando que outros projetistas pudessem acessá-lo ao mesmo tempo.

Cada ICP é armazenado no MRO como uma colônia. As instâncias de colônias que devem ser consideradas versões de um ICP será "Congelada", significando que qualquer usuário somente poderá ter acesso para leitura a essa colônia. Uma versão não poderá nunca ser modificada, e se alguma alteração for necessária àquele ICP, então será necessário que se crie uma nova alternativa que, essa sim será modificada, e uma vez completada a alteração, poderá vir a ser uma nova versão, se assim desejado. Em termos de operações na base de dados, isso será efetuado com a instanciação da colônia constrita pelo ICP a ser alterado, criando-se assim duas instâncias para aquela colônia. A instância que corresponde à versão original permanecerá protegida contra escrita, e a outra

instância, que corresponde à nova alternativa criada (inicialmente idêntica à versão original) estará liberada para alterações, e será então editada. Se a alteração deve levar à uma nova versão, ela será por sua vez a seu tempo também congelada, e protegida contra escrita.

Cada usuário pode ter apenas uma das instâncias de cada colônia disponível num dado instante, a qual é chamada de **Variante Corrente** da colônia. Dessa maneira cada usuário terá sempre disponível uma versão - uma Configuração Básica segundo Bersoff - do projeto, composto pela coleção de versões ou alternativas correntes de cada ICP do projeto.

6. - Meta-Sistemas de Gerenciamento de Dados

A armazenagem dos dados em uma base Base que utiliza o MRO pode ser logicamente feita utilizando-se vários arquivos. Além dos arquivos lógicos de objetos e de relacionamentos já mencionados, devem existir os vários arquivos de atributos, para armazenar atributos das várias características e os arquivos de históricos das colônias.

Deve ser notado que, ao contrário dos demais modelos de dados, cada arquivo lógico do MRO não armazena os dados de um mesmo item de especificação do esquema, e sim armazena dados que são estruturalmente semelhantes, mesmo que correspondam logicamente a ítens diferentes, e até de estruturas diferentes. Isso deve-se ao fato de que cada item de dado tem existência própria e individual, seja esse item um objeto, um relacionamento ou um atributo. Isso torna possível modelar o próprio MRO segundo um modelo de base de dados.

Considerando que cada objeto, relacionamento, colônia, atributo, etc, são por sua vez objetos de tipo respectivamente OBJETO, RELACIONAMENTO, COLÔNIA, ATRIBUTO, etc, pode-se descrever um Esquema genérico do MRO através de um **Meta-Esquema** de dados para o modelo. Isso pode ser feito considerando-se que cada arquivo lógico pode ser encarado como um Meta-Tipo de objeto. Existirão os seguintes Meta-Tipos de Objetos:

- Identificadores: correspondem a todos os identificadores de objetos da Base de Dados, sejam eles os identificadores principais (nomes) ou sinônimos;
- Objetos: correspondem a todas as informações de cada objeto da base de dados;
- Colônias: armazenam os conjuntos de objetos constritos em cada classe;
- Relacionamentos: correspondem a todos os relacionamentos existentes na base de dados;
- Atributos: armazenam o conjunto de todos os atributos de cada objeto ou relacionamento da base de dados;
- Atributos das várias características: existirá um Tipo de Atributo específico para cada Característica de Atributo (Propriedade, Regra, Posição, Comentário, etc).

As várias Características de Atributos são tratadas sempre da mesma maneira, permitindo que qualquer característica de atributo seja incorporada a um SGDE. Note-se que os atributos com característica de Identificadores, sejam eles os identificadores principais ou qualquer tipo de sinônimo, são tratados de forma diferen-

ciada dos demais atributos.

Uma conseqüência imediata da existência de um Meta-Esquema de Dados que permite definir o esquema de dados de uma dada aplicação, é que o Sistema de Gerenciamento de Dados para Engenharia permitirá a existência de um Meta-Sistema de Gerenciamento de Base de Dados (Meta-SGBD), o qual por sua vez manterá uma Meta-Base de Dados.

Um SGDE que abrange um Meta-SGBD é denominado neste trabalho **Meta-Sistema de Gerenciamento de Dados**, ou **Meta-SGD**. Num sistema assim, co-existem uma Meta-Base de Dados e uma Base de Dados. A Meta-Base de Dados armazena as informações **léxicas**, **sintáticas** e **semânticas** que constituem o Esquema de dados das informações manipuladas pela Base de Dados, ou seja, as informações que especificam como os dados se relacionam e qual o seu significado na Base de Dados.

O Meta-Sistema de Gerenciamento de Dados atua como um Gerenciador do Esquema de Dados armazenado na Meta-Base de Dados. Esse Esquema de Dados incorpora as informações de Sintaxe da Linguagem de Definição de Dados da Base de Dados, e nesse sentido é um Esquema de Dados semelhante aos usados nos demais modelos. No entanto, sendo definido pelo Meta-Esquema, o esquema de uma aplicação armazena também as informações sobre os próprios dados, e permite que essas informações sejam representadas em uma estrutura sintática idêntica à que é usada para representar os próprios dados. Isso permite que o Meta-Esquema do modelo MRO seja representado usando o próprio MRO.

A Meta-Base de Dados é usada pelo SGDE para manipular os dados da Base de Dados, e tendo a mesma estrutura de uma Base de Dados, a Meta-Base de Dados pode ser manipulada usando as mesmas operações que permitem a Manipulação de uma Base de Dados. Com esse enfoque, uma aplicação pode tanto manipular uma Base de Dados quanto manipular a definição dessa Base. Um ponto a ser analisado é o quanto uma alteração na Meta-Base de Dados afeta o conteúdo da Base de Dados. Isso será feito na seção 7.

Sem considerar-se efeitos de alterações da Meta-Base de Dados e as conseqüências semânticas de operações de inicialização de Bases de Dados e de Meta-Bases de Dados, pode-se usar a dualidade de estruturas e operações da Base e da Meta-Base de Dados para incorporar ambas em apenas uma **Base Unificada.** Nessa base são então armazenados o próprio esquema de dados e os dados da base. Ou seja, usando-se a terminologia de bases de dados relacionais, a mesma base armazena tanto a base extencional quanto a base intencional.

Isso pode ser feito aproveitando-se a disponibilidade permanente dos objetos da Colônia Global. Sempre que uma Base Unificada é criada, inicializa-se automaticamente a Colônia Global com um objeto de tipo controlado pelo sistema, que também deve ser único, o qual pode ser denominado de tipo ESQUEMA (esse nome não precisa estar entre os nomes de tipos de objetos definidos pelo usuário). Esse objeto passa a constringir a colônia "Esquema", a qual irá armazenar os objetos de tipos (também controlados pelo sistema - na realidade Meta-tipos) OBJETOS, IDENTIFICADORES, COLÔNIAS, RELACIONAMENTOS e ATRIBUTOS. A figura 2 corresponde ao DRO de um esboço de como essa colônia é esquematizada.

O usuário define e manipula o esquema de sua aplicação através da definição e manipulação de objetos dessa colônia. Atributos com características de Regra ou Procedimento são espe-

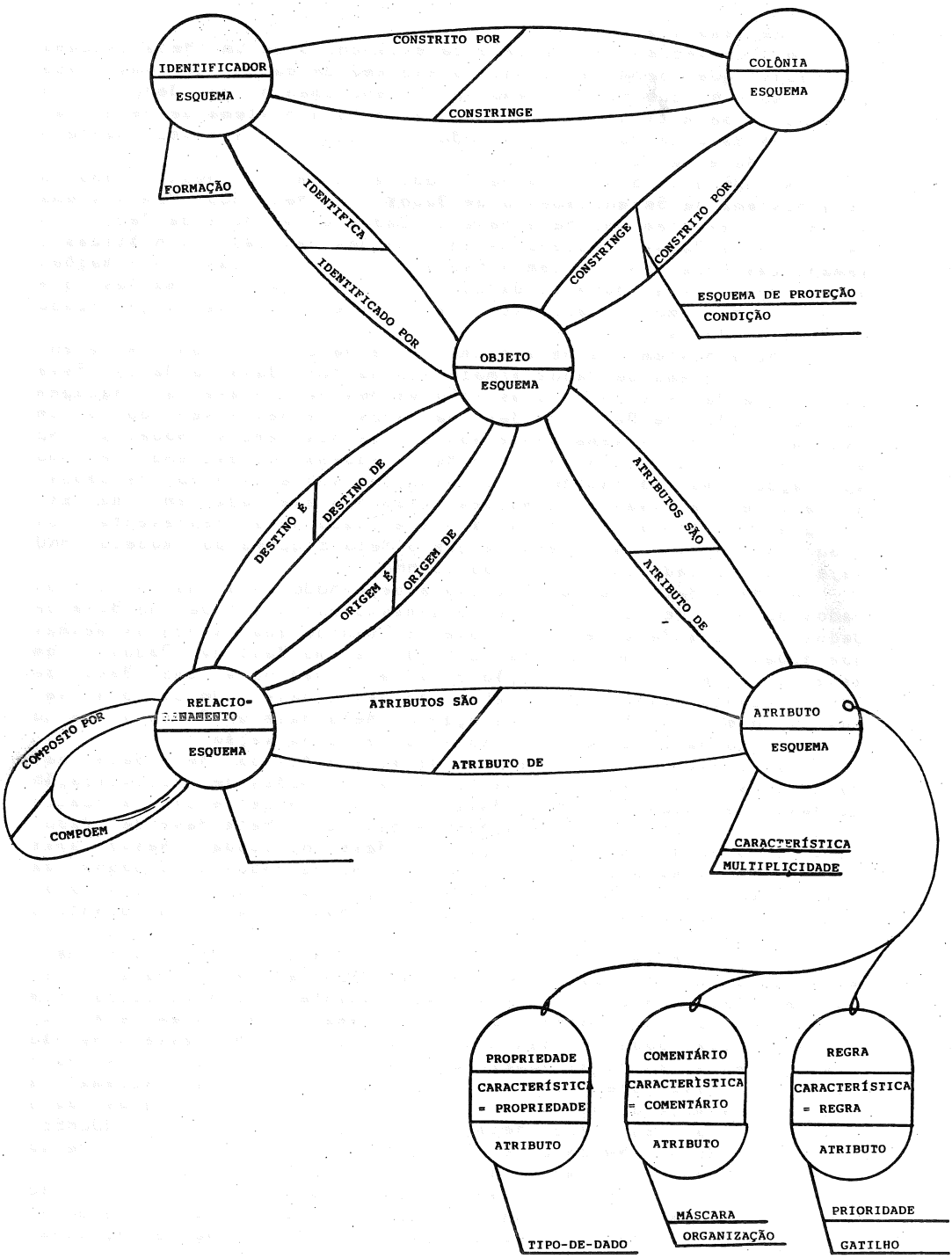

Figura 2 - Uma versão simplificada do esquema da Colônia βEsquema

cialmente adequados para essa colônia. No entanto, como o conteúdo da colônia "Esquema" corresponde ao esquema do restante da base, as operações semânticas da base de dados devem ser feitas usando-se os dados armazernados nessa colônia. Decorre daí que as operações que podem ser executadas na colônia "Esquema" são as mesmas que podem ser executadas em qualquer outra colônia, porém o sistema deve controlar essa colônia como um recurso próprio, uma vez que depende de seu conteúdo para reconhecer as operações solicitadas sobre as demais colônias. Portanto, o MRO não tem a Linguagem de Definição de Dados distinta da Linguagem de Manipulação de dados, sendo ambas as operações realizadas pela Linguagem de Manipulação de Dados. Quando esta se refere à Meta-Base de Dados, ela atua como uma Linguagem de Definição de Dados de um sistema convencional.

Neste trabalho, o termo Meta-Base de Dados corresponde ao esquema armazenado na Colônia "Esquema", ao passo que os dados dos usuários são armazenados nas demais colônias, e chamados simplesmente de Base de Dados. Ao conjunto da base de dados mais a colônia "Esquema" (a Meta-Base de Dados) denomina-se Base Unificada.

7. - O Controle de Versões do Esquema de Dados

Sendo armazenado em uma Colônia de Objetos, o esquema de uma aplicação está sujeito às mesmas regras de operações que governam as demais colônias. Isso vale também para os critérios de instanciação de colônias, o que significa que podem existir várias versões de esquemas. No entanto, a colônia "Esquema" é reconnecida pelo SGDE e tratada de maneira especial quando as operações se referem à Meta-Base de Dados. Portanto, para que um esquema de dados possa ser usado como tal, é necessário que a colônia "Esquema" esteja protegida contra escrita, ou seja, somente podem ser usadas variantes que sejam versões.

Assim, para que uma base de dados possa ser usada, ela deve ser inicializada, e sua Colônia "Global" e Colônia "Esquema" criadas. A Colônia "Global" passará a ter inicialmente um único objeto que constringe a Colônia "Esquema", e o esquema da aplicação deve ser definido. Tendo completado o esquema, ele deve ser "Congelado", e a partir daí poderá ser usado para que a base de dados propriamente dita seja usada. A Base de Dados estará então inicializada. A figura 3 mostra como estará uma base de dados recém-inicializada. Note-se que os objetos tipo βESQUEMA e βUSUÁRIO são controlados pelo sistema (O β do nome indica isso), e portanto não precisam ser explicitamente colôcados pelo usuário. Os objetos constritos pelo objeto βESQUEMA são os Tipos de Objetos, de Relacionamentos, etc. que definem a primeira versão do esquema da aplicação. Objetos do tipo βUSUÁRIO são usados para o controle de acesso à base de dados, e podem existir quantos forem necessários. Em uma base de dados existe sempre exatamente um objeto do tipo βESQUEMA, do qual podem existir muitas variantes.

Se alguma alteração for necessária, a colônia "Esquema" deve ser instanciada, e a nova alternativa para o esquema deve ser fornecida ao sistema. No entanto, para que possa ser usada para continuar o acesso à base de dados, esta terá que ser por sua vez congelada. Numa situação assim, existirão duas versões para o

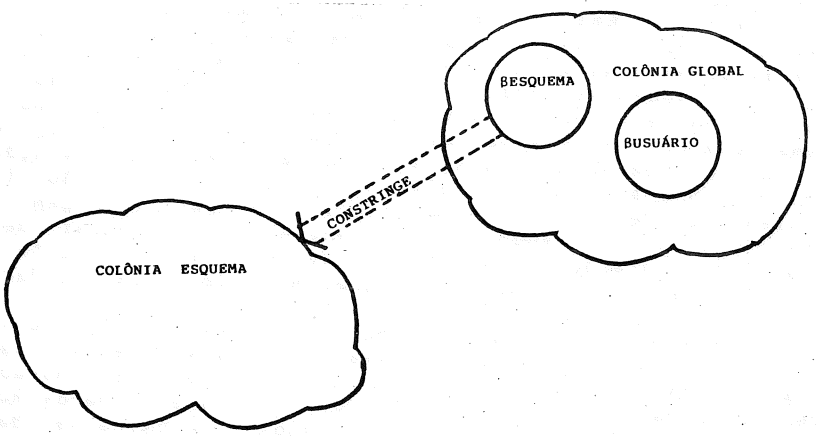

Figura 3 - DRI de uma base de dados recém-inicializada

esquema de uma base de dados. Será usada para um determinado projetista a versão que corresponde à sua variante corrente. Podem existir várias alternativas de um esquema, porém para que possam ser usadas como o Esquema da base de dados elas terão necessáriamente que ser congeladas, transformando-se em versões.

Note-se que dessa maneira, alterações no esquema de uma aplicação não causam impactos na estrutura da base de dados, uma vez que as informações da base estarão sempre associadas à versão do esquema que a criou, às quais sempre estarão disponíveis. Quando um usuário acessar as informações no entanto, ele terá acesso apenas às informações que ele tem esquematizadas em sua versão corrente da colônia "Esquema".

Feita essa caracterização, pode-se analisar agora como cada tipo de alteração do esquema pode ser tratado pelo MRO. Alterações que incluam novos tipos de objetos ou de associações (relacionamentos ou atributos) entre tipos de objetos já existentes não causam problemas, pois apenas irá ocorrer que na parte da base criada anteriormente à alteração do esquema não existem objetos e associações dos tipos novos.

Alterações que retirem tipos de objetos ou de associações do esquema poderão causar a existência de dados na Base de Dados que não estariam definidos no novo esquema. Nesse caso, três alternativas podem ser escolhidas pelo projetista de um SGBD apoiado no MRO: pode-se ignorar dados que não estão definidos na versão corrente do esquema; pode-se indicar que o dado somente pode ser acessado quando o esquema corrente é o mesmo usado para a definição daquele dado; ou finalmente, pode haver a possibilidade de que o dado possa ser acessado com a indicação que as informações da Meta-Base de dados devem sempre vir acompanhadas da indicação da versão do esquema a que elas se referem.

8. - Conclusões

Uma consideração deve ser feita aqui quanto aos nomes dados aos conceitos do MRO. A partir da popularização da linguagem de programação SmallTalk-80 [REN 82] desenvolvida pela Xerox Parc no inicio dos anos 80, o conceito de objetos desenvolveu-se paralelamente também como um paradigma de programação. Apesar de serem conceitos semelhantes, como um paradigma de programação o conceito de objeto apresenta características diferentes, e as vezes conflitantes, em relação às características que lhe são atribuídas como um modelo de dados para o desenvolvimento de bases de dados orientadas a objetos.

O Paradigma de Programação orientada à objetos designa os vários objetos de um mesmo tipo como uma "Classe de Objetos", e cada ocorrência de um objeto como uma "Instância de Objeto". As associações entre objetos são feitas através das propriedades de cada classe de objetos, e raramente as associações são explicitadas. No MRO, a associação entre objetos é sempre feita explicitamente, e dessa maneira, uma uniformidade dos termos usados para designar objetos e relacionamentos deve ser seguida. Por isso, o termo "Objeto" é usado para designar uma instância de um objeto, da mesma maneira que o termo "Relacionamento" é usado para designar um relacionamento único entre dois dados objetos. Analogamente, o termo "Tipo de Objeto" é usado para designar as características importantes que devem ser reconhecidas em objetos, as quais permitem identificar determinado objeto como sendo daquele determinado tipo; e o termo "Tipo de Relacionamento" é usado para designar as características importantes que devem ser reconhecidas em relacionamentos, aquelas que permitam identificar cada relacionamento como sendo daquele tipo.

Um conceito importante no MRo corresponde às Colônias de Objetos. Esse conceito permite não só o acompanhamento do foco de interesse de um projetista na Base de Dados, mas também a instanciação e controle de versões e alternativas dentro da Base de Dados.

Também importante é a fundamentação do modelo na Teoria dos Grafos porque, permitindo a homogeneidade da representação estrutural dos dados, propicia a construção do SGD como um Meta-Sistema. A fundamentação do MRO na Teoria dos Grafos e o conceito de Colônias de Objetos se aliam para criar o suporte onde a manutenção de um esquema de dados pode ser feita dinamicamente, em conjunto com a operação normal do SGD sem impacto sobre os dados já armazenados.

Por fim é importante ressaltar que o objetivo principal que norteou o desenvolvimento do MRO é o seu emprego em Ambientes de Desenvolvimento de Software. Duas implementações de SGDEs para esse fim apoiadas nos conceitos apresentados estão em fase adiantada, uma no ICMSC-USP em São Carlos, feita para o Sistema SACDS [TRA_82], e outra no CTI em Campinas para o Projeto SIPS [TSU_85].

REFERÊNCIAS BIBLIOGRÁFICAS

[ABI_87] Abiteboul, S.; Hull, R. - "IFO: A Formal Semantic Database Model", ACM TODS, Vol. 12, Nro. 4, December 1987, pp. 525-565.

[BER_84] Bersoff, E. - "Elements of Software Configuration Management", IEEE Trans. Software Engineering, Vol. SE-10, Nro. 1, January 84, pp. 79-87.

[BIC_86] Bic, L.; Gilbert, J. P. - "Learning from AI: New Trends in Database Technology", IEEE Computer, Vol. 19, Nro. 1, January 86, pp. 44-54.

[EAR_85] Earl, A. N.; Whittingtom, R. P. - "Capturing the Semantics of an IPSE Database" Data Processing, Vol.27, Nro. 9, November 1985, pp. 33-43.

[FAR_85] Farmer, D. B.; King, R.; Myers, D. A. - "The Semantics Database Constructor" IEEE Trans. Software Engineering, Vol. SE-11 Nro. 7, July 85, pp. 583-591.

[HAM_81] Hammmer, M.; McLeod, D. - "Database Description with SDM: A Semantic Database Model", ACM Trans. Database Systems, Vol 6, Nro. 3, September 81, pp. 351-386.

[HAR_85] Hartzband, D. J.; Maryanski, F. J. - "Enhancing Knowledge Representation in Engineering Databases" IEEE Computer, Vol. 18, Nro. 9, Septenber 85, pp. 39-48.

[KAT_84] Katz, R. H.; Lehman, T. J. - "Database Support for Versions and Alternatives of Large Design Files", IEEE Trans. Computers, Vol. SE-10, Nro. 2, march 84, pp. 191-200.

[KAT_86] Katz, R. H.; Chang, E.; Bhateja, R. - "Version Modeling Concepts for Computer-Aided Design Databases", SIGMOD 86 Conference, in ACM SIGPLAN Vol. 21, Nro. 12, pp. 379-386.

[KEL_87] Kelter, U. - "Concurrency Control for Design Objects With Versions in Cad Databases", Information Systems, Vol. 12, Nro. 2 pp. 137-143.

[MAT_81] Matsuka, H.; Uno, S.; Sibuya, M. - "Specific Requirements in Engineering Data Base", Lecture Notes in Computing Science, Vol. 133, New York: Springer-Verlag, 1982, pp. 345-356.

[NAV_86] Navathe, S. et al. "Integrating User Views in Database Design", IEEE Computer, Vol. 19, Nro. 1, January 86, pp. 50-62.

[NG*_81] Ng, A. P. - "Further analysis of the Entity-Relationship approach to Database Design", IEEE Trans. Software Engineering Vol. SE-7 Nro. 1, January 81, pp. 85-99.

[PAR_85] Parent, C.; Spaccapietra, S. - "An Algebra for a General Entity-Relationship Model", IEEE Trans. Software Engineering, Vol. SE-11 Nro. 7, July 85, pp. 634-643.

[REN_82] Rentsh, T. - "Object-Oriented Programming", ACM SIGPLAN Notices, Vol. 17, Nro. 9, September 1982, pp. 76-87.

[TRA_82] Traina Jr., C. - "Sistema de Apoio por Computador à Documentação de Sistemas", Dissertação apresentada ao ICMSC-USP para obtenção do Título de Mestre, Dezembro 82.

[TRA_86] Traina Jr., C. - "Máquina e Modelo de Dados Dedicados para Aplicações de Engenharia", Tese apresentada ao IFQSC-USP para obtenção do Título de Doutor, Dezembro 86.

[TRA_88] Traina Jr., C.; Slaets, J. F. W. - "Um Modelo de Representação de Objetos", in Anais do 3º Simpósio Brasileiro de Banco de Dados, Recife, Março de 1988, pp. 227-242.

[TSU_85] Tsukumo, A. N. et alli - "Um Sistema Expansível para Produção de Software", in Anais do 2º Congresso Nacional de Automação Industrial, São Paulo, Novembro 85, pp. 379-386.
