CONSTRUÇÃO DE FERRAMENTAS DE APOIO A PROJETOS UTILIZANDO UM GERENCIADOR DE BASES DE DADOS COM SUPORTE DE ESQUEMAS SUPLEMENTARES

Luiz Camolesi Júnior [1]
Caetano Traina Júnior [2]
USP Campus de São Carlos - ICMSC/SCE
Caixa Postal 668 - CEP: 13560.970 - São Carlos, SP
E.Mail: Caetano@uspfsc.ifqsc.usp.ansp.BR

RESUMO

Ferramentas de apoio a projetos e produção (CAD/CAM/CIM) requerem o uso de Gerenciadores de Bases de Dados que permitam a armazenagem flexível de informações. Os sistemas de Gerenciamento de Bases de Dados tradicionais, mesmo aqueles apoiados no modelo relacional, não são boas soluções pois não gerenciam adequadamente informações cuja estrutura seja muito complexa.

O desenvolvimento de sistemas mais flexíveis de gerenciamento de informações, como aqueles baseados em modelos semânticos e mais recentemente os modelos orientados a objetos, modificaram essa situação, porém às custas de maior rigidez na manipulação da estrutura das informações (Esquema de Dados), em particular nas aplicações onde se requer acesso multi-usuário.

Este artigo mostra como é possível manter a flexibilidade de estruturação e manipulação dos dados em modelos orientados a objetos, ao mesmo tempo em que permite-se que aspectos de interesse de uma aplicação (ferramenta) possam ser reestruturados (ou re-configurados) independentemente do restante da base, através de diferentes versoes do Esquema de Dados para cada aspecto que se deseja representar. Isto é conseguido através da definição de um Esquema Inicial que modela os aspectos gerais de um projeto, ao qual acrescem-se dinamicamente, a critério do usuário, Esquemas Suplementares que modelam os aspectos mais refinados ou particularizados do projeto.

ABSTRACT

Computer-aided tools to design and manufacturing (CAD/CAM/CIM) require the use of Databases Management systems that allow the flexible storage of informations. Tradicional DBMS, even those based in the relacional model, aren't good solutions, since they don't satisfactorily manage informations in which the structures are very complex.

The development of more flexible Informations Management Systems, like those based on semantic models and more recentely the object oriented models, change this situation, but settle a lower flexibility in the manipulation of the Scheme, moreover on aplications where is required a multi-user acess.

This paper shows a solution that makes possible to keep the flexibility in the data organization and manipulation, like those provided by the object oriented models, and providing also that an user have freedom in modifying some aspects and structures of the database scheme, independently (and isolated) of changes made by other users. This is accomplished through the definition of an Initial Scheme, modelling generic aspects of a design, to which, at the user's discretion and dinamically, are appended Supplemental Schemes modelling more refined (or private) aspects of the overall design.

KEYWORDS : Database, Scheme, Supplement Scheme, Computer-aided Design , Version Control.

---

1 - Bacharel e Mestre em computação pelo ICMSC - USP.

2 - Professor/Doutor e pesquisador do ICMSC - USP.

1. INTRODUÇÃO.

Um Sistema de Bases de Dados (SBD) gerencia a armazenagem e recuperação de dados de uma aplicação (ferramenta), os quais são definidos segundo uma estrutura que permite ao SBD reconhecer a maneira como os dados devem ser tratados. Essa estrutura, denominada **Esquema de Dados** da Base de Dados, determina como o acesso e armazenagem de todos os dados da Base são tratados. Uma Base de Dados é definida integralmente através de um único Esquema de Dados, embora o acesso possa ser feito através de visões, que correspondem a partes do Esquema. Em uma situação em que a própria Base de Dados é o produto da atividade de projetar [HARTZBAND_85], a evolução do Esquema tende a acompanhar dinamicamente a evolução da Base [GALLO_86]. Isso traz problemas no acesso multi-usuário, onde mais de um projetista podem alterar simultaneamente a Base de Dados, ou onde parte dela já está sendo utilizada para produção, enquanto outra parte ainda está associada a Projetos em andamento.

Os sistemas convencionais de Gerenciamento de Bases de Dados atendem em parte a esse problema, permitindo operações de reserva e travamento para acesso exclusivo aos dados de interesse particular. Porém não permitem realizar essas operações no Esquema, o qual, por ser único, deve estar sempre disponível a todos os usuários. Em consequência disso, se um Esquema for alterado, essa alteração é imediatamente visível para todas as aplicações e usuários que estejam acessando a Base de Dados.

Este trabalho mostra como é possível resolver esse problema no contexto dos SBD orientados a objetos [ATKINSON_89] [HULL_90] [JACKSON_91] [NIERSTRASZ_89], através da introdução do conceito de Esquemas Suplementares, que podem ser dinamicamente acoplados ao Esquema geral, dito Esquema Inicial, os quais têm efeito apenas dentro do contexto da aplicação específica que requisita seu acoplamento. Assim, Esquemas Suplementares podem ser criados e editados pela aplicação, e dinamicamente acoplados ao Esquema Inicial, para acesso restrito a essa aplicação. Isso permite que diferentes aspectos de uma Base de Dados possam ser explorados através de Esquemas parciais que são ativados apenas quando necessários. Estruturas conflitantes podem ser gerenciadas, permitindo por exemplo que Esquemas específicos possam ser ativados quando necessário, não obrigando que todas as estruturas sejam globais, pré-definidas e compatíveis entre si, tal como o fazem os métodos tradicionais de modelagem de dados.

Os conceitos de orientação a objetos e modelagem de dados [STONEBRAKER_90] são apresentados na seção 2, utilizando-se para isso um modelo de dados orientado a objeto

desenvolvido especificamente para aplicações de Projeto, denominado Modelo de Representação de Objetos (MRO) [TRAINA_88A] [TRAINA_88B], sobre o qual existe implementado um SBD Orientado a Objetos denominado de GErenciador de Objetos (GEO[TRAINA_91]). Como exemplo para se explicar os conceitos tratados neste artigo, utiliza-se a modelagem simplificada de um ambiente de competição automobilística.

Na seção 3 apresentam-se os conceitos de Projetos e sub-projetos, de contexto de dados, e como eles são suportados pelo MRO. Na Seção 4 estabelecem-se a existência e a forma de gerenciamento de Versões de Projeto. Na seção 5 descreve-se o conceito de gerenciamento de informações e acesso multi-usuário através de Esquemas Suplementares. Na seção 6 descreve-se estruturalmente a forma de acesso das ferramentas de Projeto na Base de Dados, e finalmente nas seções 7, 8 e 9 respectivamente estão as conclusões, agradecimentos e referências bibliográficas.

2. MODELAGEM DE DADOS.

Um Modelo de Dados é genericamente definido como uma ferramenta lógica de representação e descrição de um aspecto da realidade. Um modelo de dados possui um conjunto de construções sintáticas e uma linguagem de especificação, a qual permite estruturar os elementos do mundo real segundo as estruturas sintáticas do modelo.

A ação de modelar visa encontrar elementos que simbolizem as informações utilizadas em uma determinada aplicação. O processo de estruturação de um aspecto da realidade segundo um modelo de dados é denominado de **modelagem** desse aspecto, e a estrutura resultante, descrita segundo as construções sintáticas do modelo de dados utilizado, é denominada **Esquema de Dados**.

As construções sintáticas de um Modelo de Dados Orientado a Objetos concentram-se na representação da realidade através dos objetos que o compoem, modelando atributos, estados, comportamentos e formas de interação entre estes objetos.

As construções sintáticas básicas do MRO modelam um empreendimento através de **Objetos**, cada um dos quais possui um **Tipo**, um conjunto de **Atributos** e podendo participar de **Relacionamentos** com outros objetos. Um relacionamento é uma forma especial de objeto que pode ser tratado pelo próprio sistema. Os tipos de objetos podem ser especializados, criando hierarquias de **Subtipos**. Todos os atributos e relacionamentos possuem também um tipo, sendo que relacionamentos também podem possuir atributos. Todo atributo tem uma **Característica** definida como parte do modelo, que pode ser

entre outras **Propriedade, Sinônimo** ou **Regra**. As propriedades permitem associar valores ou textos aos objetos ou relacionamentos; sinônimos podem atuar como identificadores que permitem às aplicações localizar objetos; e regras permitem descrever ações que o objeto (ou relacionamento) toma quando alguma condição é satisfeita no contexto do objeto (ou do relacionamento).

Na figura 1 mostra-se o exemplo de modelagem de uma aplicação: um ambiente de competição automobilística modelada segundo o MRO, na sua representação gráfica. Nessa representação os tipos de objetos são representados por círculos (carro, pessoa, circuito e pneu), os tipos de relacionamentos são representados por arcos (trabalha com, usa, usado, testado, etc.) e tipos de atributos são representados por traços ligados aos tipos de objetos (equipe, nacionalidade, idade, país e fábrica). Como estão representados apenas os tipos e não os objetos, relacionamentos ou valores em sí, diz-se que esse é o Esquema de Dados da aplicação.

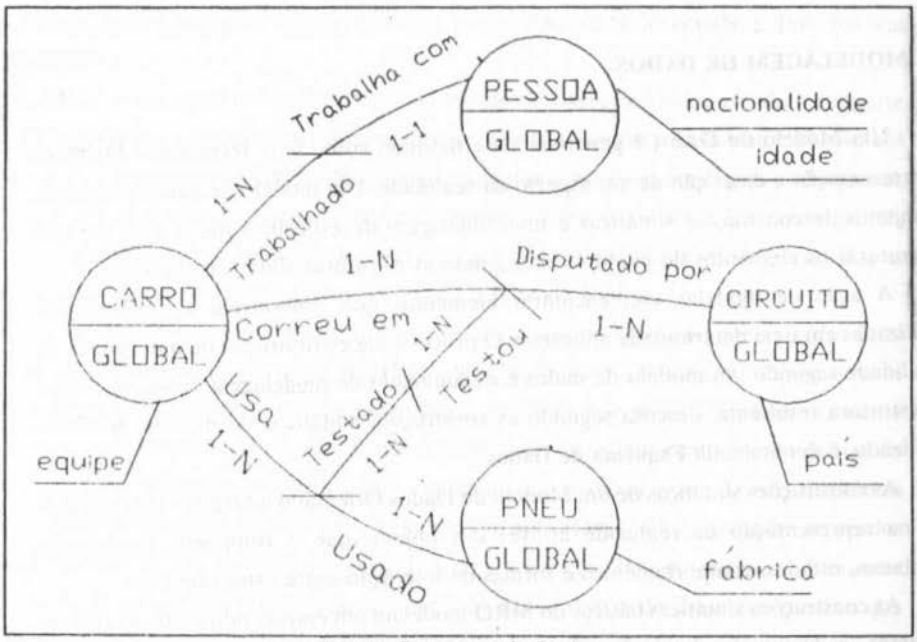

Fig. 1 - Esquema de Dados

Sobre essa estrutura (Esquema de Dados) pode-se definir valores que serão armazenados na Base de Dados, como mostra a figura 2, onde representam-se: uma PESSOA chamada Nelson; seu CARRO de número 20 e equipe a que pertence

(Benetton); sua idade (34) e nacionalidade (brasileira); um dos circuitos (Montecarlo) em que correu e o país onde se localiza (Mônaco); e o pneu (CD88) que usou nessa corrida bem como a fábrica que o produziu (Pirelli).

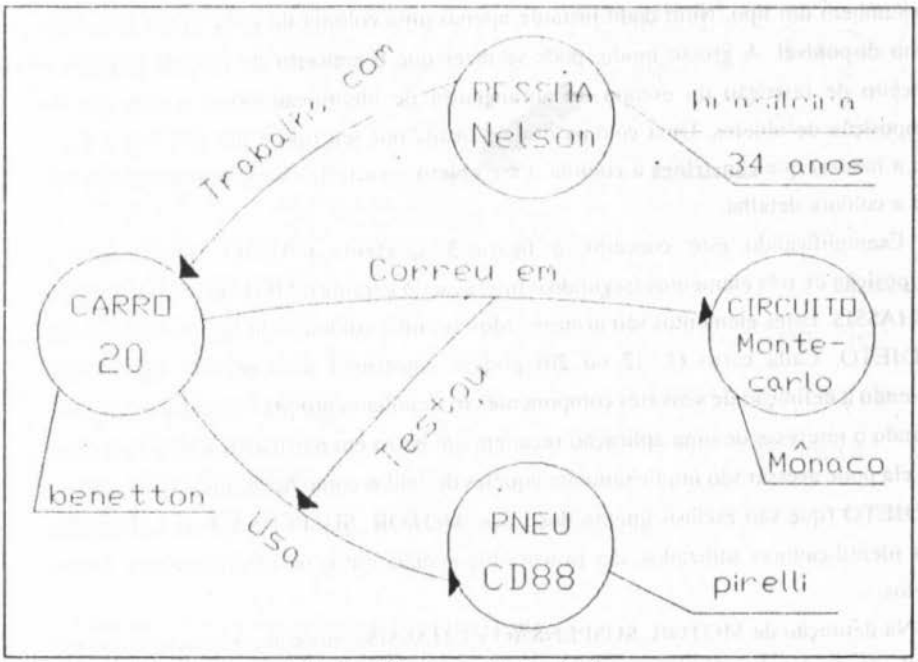

Fig. 2 - Diagrama de Instâncias

3. EXECUÇÃO DE PROJETOS E SUBPROJETOS.

Projetos de um modo geral caracterizam-se por possuírem componentes que podem ser tratados ou reprojetados separadamente como **subprojetos** ("Simultaneous Engineering" [BLACKBURN_90]), permitindo explorar a natureza geralmente evolutiva nos aspectos de performance, tecnologia e custo.

O MRO possui um conceito adicional, denominado **Colônia de Objetos**, segundo o qual todos os objetos que descrevem um aspecto particular de uma aplicação são agrupados, e tornam-se acessíveis (ou **disponíveis**) para aplicativos que acessam a Base de Dados apenas quando explicitamente solicitados e permitidos. Assim os aspectos que nao são de interesse para uma aplicação não são visíveis para ela, mesmo na

manipulação de objetos compostos [KIM_90], pois o conjunto de identificadores dos objetos manipulados restringem-se aos objetos disponíveis, devido ao ocultamento de identificadores não necessários.

Cada colônia é **habitada** apenas por objetos de determinados tipos, e possui por sua vez tambem um tipo. Num dado instante apenas uma colônia de cada tipo é assumida como disponível. A grosso modo, pode-se dizer que o conceito de colônia engloba o conceito de restrição do escopo de abrangência de identificadores e o conceito de composição de objetos. Uma colônia é identificada por seu tipo e por um objeto (que não a habita) que **constringe** a colônia. Esse objeto caracteriza o aspecto em particular que a colônia detalha.

Exemplificando este conceito, a figura 3 apresenta CARRO como sendo a **composição** de três elementos (segundo o interesse do gerente): MOTOR, SUSPENSÃO e CHASSIS. Estes elementos são armazenados em uma colônia cujo tipo é denominado PROJETO. Cada carro (1, 12 ou 20) poderá constringir uma colônia PROJETO, contendo a definição de seus três componentes (o detalhamento previsto para um carro). Quando o interesse de uma aplicação recai em um carro em particular, todos os objetos que ela pode acessar são implicitamente aqueles definidos como habitantes dessa colônia PROJETO (que são exclusivamente dos tipos: MOTOR, SUSPENSÃO ou CHASSIS), e os identificadores utilizados são procurados apenas entre os identificadores desses objetos.

Na definição de MOTOR, SUSPENSÃO e CHASSIS, seus componentes podem ser detalhados (ou projetados) separadamente de forma idêntica ao CARRO, simplesmente acrescentando-se, por exemplo, colônias MOTOR, cada uma definindo os componentes do MOTOR que pode ser utilizado no carro em questão.

A atividade de decompor elementos complexos em componentes cada vez menos complexos estabelece a estrutura de colônias na forma de uma árvore, que finaliza-se quando as informações necessarias se tornam suficientemente detalhadas. Os objetos que correspondem às informações menos detalhadas habitam a colônia que é o topo da hierarquia, a qual é denominada **Colônia Global**. Esta é a única colônia que não é constrita por nenhum objeto. Todos os objetos da colonia global estão sempre disponíveis a todos os usuários da Base de Dados, sendo a partir deles que cada usuário determina aspectos de seu interesse particular.

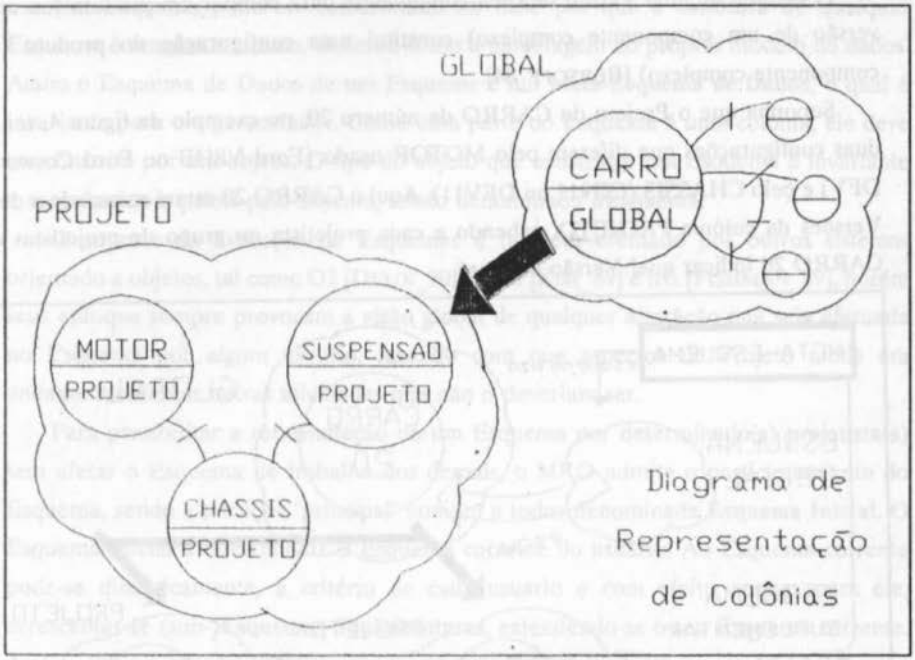

Fig. 3 - Hierarquia de Colônias

4. VERSÕES E ALTERNATIVAS DE PROJETO.

O MRO contempla a evolução do Projeto de um determinado componente ou de todo o sistema mantendo **Variantes** na Base de Dados. As variantes de componentes que encontram-se em fase de elaboração são mantidas como **Alternativas** de Projeto. As alternativas têm seu acesso restrito ao projetista responsável até serem consideradas finalizadas, quando então passam a ser **Versões**. As Versões não podem ser modificadas (a não ser que seja feita uma cópia para desenvolver-se uma nova alternativa) e podem ser acessadas e utilizadas por quaisquer aplicativos e usuários.

O MRO efetua o controle de Versões e alternativas considerando cada uma como uma colônia (ou uma sub-hierarquia de colônias), e gerenciando individualmente o acesso permitido a cada colônia. Assim as colônias que são colocadas como Versão têm seus objetos acessíveis apenas para consulta, enquanto as colônias alternativas podem ter suas informações alteradas.

A escolha de uma Versão de cada componente para compor um produto (ou uma versão de um componente complexo) constitui uma **configuração** do produto (ou componente complexo) [BERSOFF_84].

Suponha que o Projeto do CARRO de número 20, no exemplo da figura 4, tenha duas configurações que diferem pelo MOTOR usado (Ford V8HB ou Ford Cosworth DFV) e pelo CHASSIS (CG911 ou DFV11). Aqui o CARRO 20 estará associado a duas Versões da colônia PROJETO, cabendo a cada projetista ou grupo de projetistas do CARRO 20 indicar qual Versão utilizará.

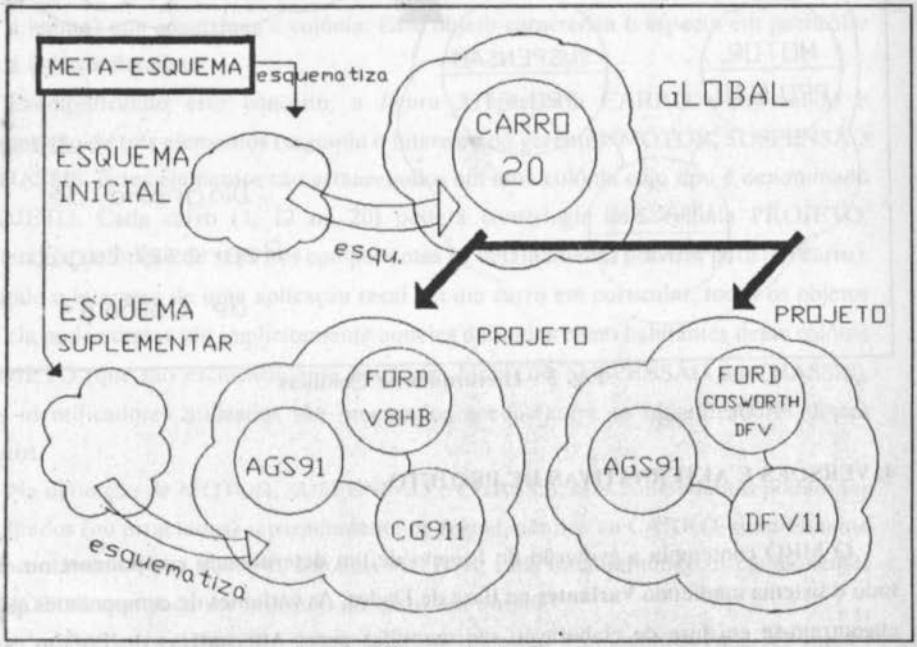

Fig. 4 - Hierarquia de Esquemas

5. ESQUEMA DE DADOS E ESQUEMAS SUPLEMENTARES.

Uma característica importante do MRO é permitir que o Esquema de Dados possa ser armazenado na Base de Dados de forma idêntica a qualquer outra informação. O GEO, ou qualquer gerenciador apoiado no MRO, armazena o Esquema em colônias específicas controlada pelo gerenciador, denominada **Colônias Esquema**. É assim possível contemplar a evolução de Esquemas como parte da evolução do Projeto como um todo.

Um Esquema pode ser armazenado na Base por que a estrutura de qualquer Esquema é sempre a mesma, definida como a modelagem do próprio modelo de dados. Assim o Esquema de Dados de um Esquema é um **Meta-Esquema de Dados**, o qual é invariante para um gerenciador. Como uma parte do Esquema é uma colônia, ele deve ser constrito por um objeto. O tipo do objeto que constringe um Esquema é invariante e reconhecido a priori pelo sistema, sendo denominado **βEsquema**.

O suporte da evolução de Esquemas é também efetuado por outros sistemas orientado a objetos, tal como O2 [DEUX_90] Orion [KIM_89] e Iris [FISHMAN_89], porém seus enfoque sempre provocam a visão global de qualquer alteração que seja efetuada no Esquema por algum usuário, fazendo com que aspectos de Projeto ainda em andamento afetem outras atividades que não o deveriam ser.

Para possibilitar a reformulação de um Esquema por determinado(s) projetista(s) sem afetar o Esquema de trabalho dos demais, o MRO admite o particionamento do Esquema, sendo a partição "principal" comum a todos denominada **Esquema Inicial**. O Esquema Inicial é inicialmente o Esquema corrente do usuário. Ao Esquema corrente pode-se dinamicamente, a critério de cada usuário e com efeito apenas para ele, acrescentar-se (sub-)**Esquemas Suplementares**, extendendo-se o seu Esquema corrente. Assim, cada versão de qualquer aspecto de um Projeto pode ter seu próprio Esquema Suplementar associado.

No exemplo da figura 4, cada motor (ou cada componente em geral que constrinja uma colônia) pode ter seu próprio Esquema Suplementar, permitindo que a própria especificação daquele subprojeto seja independente do restante, ou das demais Versões do Projeto.

Resumindo, o MRO estabelece a existência de uma colônia contendo o **Esquema Inicial** para esquematizar os aspectos gerais da Base de Dados, e outras colônias contendo Esquemas Suplementares para esquematizar aspectos do empreendimento que podem assumir estruturas diferenciadas, em multiplas Versões ou instanciações.

Qualquer Esquema (Inicial ou Suplementar) pode abranger uma única colônia, ou toda uma sub-hierarquia de colônias. O Esquema Inicial em particular sempre esquematiza a Colônia Global e outras colônias hierarquicamente inferiores, até que uma colônia subordinada tenha seu próprio Esquema Suplementar. Note-se que um mesmo Esquema Suplementar pode esquematizar uma ou mais colônias, sempre com um tipo e uma estrutura subordinada semelhantes, assim como cada colônia de um mesmo tipo pode ter seu próprio Esquema Suplementar, permitindo estruturas e tipos totalmente diversos.

No exemplo da figura 4, as colônias de tipo PROJETO estão esquematizadas através de um Esquema Suplementar. Isso significa que para uma colônia desse tipo ser acessada, é necessário que o Esquema Suplementar correspondente seja acrescentado ao Esquema de Dados corrente: o Esquema Inicial não esquematiza as colônias PROJETO e nem qualquer tipo de colônia hierarquicamente subordinada a estas. O Esquema Inicial esquematiza a Colônia Global (definindo por exemplo os tipos CARRO, PESSOA, etc.) e qualquer colônia constrita por esses objetos que não sejam esquematizadas por seu próprio Esquema Suplementar.

6. FERRAMENTAS DE PROJETO: ELABORAÇÃO, CONSTRUÇÃO E GERENCIAMENTO.

Na área de Projeto apoiado por computador existem basicamente três tipos de ferramentas, no que diz respeito à forma como elas acessam um Sistema de Gerenciamento de Bases de Dados.

O primeiro tipo de ferramenta consiste daquelas que atuam como editores, criando e consultando informações na Base, as quais são em grande parte obtidas através de um processo interativo com o usuário (editores de desenhos, de especificacoes, planilhas, ferramentas de suporte ao gerenciamento de projeto, etc.).

O segundo tipo corresponde às ferramentas que processam as informações já armazenadas, gerando outros dados, em geral mantidos em estruturas distintas dos dados originais (ferramentas de calculo, geradores de planilhas de serviços, compiladores de programas numéricos, etc.).

O ferramentas do terceiro tipo correlacionam dados já armazenados, gerando saídas para outros sistemas sem mudar os dados armazenados (geradores de relatórios, conversores de formato, etc.).

Com a utilização de gerenciadores de dados poderosos, a construção de ferramentas do primeiro e do terceiro tipo ficam bastante simplificadas, porém a menos que o controle de proteção de acesso multi-usuário seja efetivamente suportado pela Base. As ferramentas do primeiro tipo ainda precisam assumir internamente o controle sobre a difusão dos dados que trata: os gerenciadores tradicionais evitam que as informações sejam acessadas simultaneamente, mas não podem impedir o acesso às informações incompletas quando o editor que auxilia sua definição não estiver ativo. O mesmo ocorre

em relação às ferramentas do segundo tipo, quando lhes submetem para processamento dados que ainda não são definitivos.

Com o suporte oferecido pelo MRO (e pelo GEO), as ferramentas dos segundo e terceiro tipos, que modificam dados armazenados na Base, não precisam estabelecer critérios internos de controle da difusão dos dados, uma vez que o gerenciador pode assumir esse controle. O mesmo ocorre com as ferramentas do terceiro tipo, que não precisam controlar, por mecanismos próprios, o fato dos dados que acessa estarem atualizados ou não, uma vez que o gerenciador de dados somente lhes permitirá acesso a dados corretos.

7. CONCLUSÕES.

Os Sistemas de Gerenciamento de Bases de Dados foram inicialmente desenvolvidos para atender principalmente aplicações de processamento de dados comerciais e para isso existem atualmente Gerenciadores poderosos. No entanto eles não atendem bem as necessidades de aplicação mais sofisticadas, tais como ferramentas de apoio a Projeto e automação de fabricação. Para isso novos gerenciadores, apoiados em modelos de dados mais ricos, têm sido desenvolvidos. A abordagem mais promissora e que mais tem recebido atenção ultimamente é a "Orientação a Objetos".

No entanto essa abordagem está desenvolvida o suficiente para permitir somente a construção de protótipos, cuja aplicação prática em produtos comercializáveis não é ainda satisfatória. Um dos problemas que ocorrem é a hierarquia de classes definida pelos tipos de objetos ser única e globalmente visível por todos os usuários. A modificação de um tipo por um usuário não pode ser ocultada dos demais, ocasionando problemas em situações onde parte dos dados estão sendo usados em produção, mas outros ainda estão em fase de desenvolvimento.

Este trabalho aborda esse problema e através de uma conceituação mais rígida de Esquemas no contexto de uma Base de Dados orientada a objetos, identifica uma solução prática, adequada para o desenvolvimento de ambientes de apoio a Projeto e produção.

O conceito Esquema Inicial e Esquemas Suplementares permite a construção de gerenciadores que suportam o isolamento de dados provido pelo conceito de Visões de Usuários da Arquitetura ANSI/X3/SPARC, aliada à flexibilidade das modificações no Esquema do sistema relacional e com a riqueza de conceitos e estruturas sintáticas dos modelos Orientados a Objetos.

A solução apresentada é descrita no contexto do Modelo de Representação de Objetos - MRO, desenvolvido no SCE-ICMSC do campus da USP de São Carlos. É nesse modelo que está apoiado o GErenciador de Objetos - GEO em desenvolvimento nesse instituto. Esse Gerenciador vem sendo utilizado como uma bancada de testes dos conceitos de modelagem de dados orientada a objetos, bem como de estruturas de implementação de gerenciadores de objetos. Os conceitos apresentados foram validados utilizando esse gerenciador, através da implementação de um módulo gerenciador de Esquemas que suporta os conceitos apresentados e de uma linguagem que permite a definição de Esquemas Suplementares, bem como a ativação seletiva dos mesmos, a critério do usuário (ou dos programas de aplicação). Este artigo no entanto procura apresentar os conceitos fundamentais desenvolvidos, de maneira que possam ser utilizados em outros modelos de dados, independente do MRO. Maiores detalhes dos conceitos apresentados neste artigo, além da linguagem de manipulação e a descrição da implementação do sistema descrito podem ser encontrados em [CAMOLESI_92].

8. AGRADECIMENTOS.

Os autores agradecem ao CNPq e à FAPESP pelo apoio financeiro que permitiu a realização deste trabalho, e aos integrantes do grupo de pesquisa em Bases de Dados e Engenharia de "Software" do Instituto de Ciências Matemáticas de São Carlos (ICMSC) pelas contribuições a esta pesquisa.

9. REFERÊNCIAS BIBLIOGRÁFICAS.

[ATKINSON_89] M. Atkinson et alli - **"The Object Oriented Database System Manifesto"**, Technical Report Altair 30-89, GIP ALTAIR in 2-INRIA-LRI, 1989.

[BERSOFF_84] E. Bersoff - **"Elements of Software Configuration Management"**, IEEE Trans. on Soft. Eng., Vol. SE-10, Nro. 1, 1984, pp. 79-87.

[BLACKBURN_90] J. Blackburn - **"Time-Based Competition"**, Strategic Manufacturing, Dynamic New Directions for the 1990s, Moody, P. E. Ed., Dow-Jones Irwin Company, 1990, pp. 189-206.

[CAMOLESI_92] L. Camolesi Jr. - **"Suporte a Acesso Multi-usuário em Bases de Dados Orientadas a Objetos através de Esquemas Suplementares"**, Dissertação de Mestrado, ICMSC-USP, São Carlos, 1992.

[DEUX_90] O. Deux et alli - **"The Story of O2"**, IEEE Trans. on KDE, Vol. 2, Nro. 1, 1990, pp. 91,108.

[FISHMAN_89] D. H. Fishman et alli - **"Overview of the Iris DBMS"**, Object Oriented Concepts, Databases, and Applications, W. Kin & F. Lochovsky Ed., Addison-Wesley Pub. Co., 1989, pp. 219-250.

[GALLO_86] F. Gallo, R. Minot, I. Thomas - **"The Object mangement System of PCTE as a Software Engineering Database Management System"**, proc ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments, Palo Alto, California, 1986.

[HARTZBAND_85] D. J. Hartzband, F. J. Maryanski - **"Enhancing Knowledge Representation in Engineering Databases"**, IEEE Computer, Vol. 18, Nro. 9, 1985, pp. 39-48.

[HULL_89] R. Hull, J. Su - **"On Accessing Object-Oriented Databases: Expressive Power, Complexity, and Restrictions"**, ACM Inter. Conf. on Management of Data, ACM-SIGMOD, Vol.18, Nro. 2, 1989, pp. 147-158

[JACKSON_91] M. S. Jackson - **"Tutorial on Object-Oriented Databases"**, Information and Software Technology, Vol. 33, Nro. 1, 1991, pp. 4-12.

[KIM_89] W. Kim et alli - **"Features of the Orion Object-oriented Database"**, Object Oriented Concepts, Databases, and Applications, W. Kin & F. Lochovsk Ed. Addison-Wesley Pub. Co., 1989, pp. 251-282.

[KIM_90] W. Kim - **"Object-Oriented Databases: Definition, and Research Directions"**, IEEE Trans. on Knowledge and Data Engineering, Vol. 2, Nro. 3, 1990, pp. 327-341.

[NIERSTRASZ_89] O. Nierstrasz - **"A Survey of Object-Oriented Concepts"**, Object Oriented Concepts, Databases, and Applications. W. Kim & F Lochovsky Ed., Addison-Wesley Pub. Co., 1989, pp. 3-21.

[STONEBRAKER_90] M. Stonebraker et alli - **"Third-generation Database System Manifesto"**, Proc. IFIP TC2 Conf. Object Oriented Database - UK, 1990.

[TRAINA_88a] C. Traina Jr., J. F. W. Slaets - **"Gerenciamento de Esquemas de Dados Dinamicamente Modificáveis em Modelo de Base de Dados Orientado a Objetos"**, Anais da XIV Conf. Latinoamericana de Informática, Buenos Aires, Argentina, 1988.

[TRAINA_88b] C. Traina Jr., J. F. W. Slaets - **"Um Modelo de Representação de Objetos"**, Anais do 3º SBBD, Recife, 1988, pp. 227-242.

[TRAINA_91] C. Traina Jr - **"GEO: Um Sistema de Gerenciamento de Bases de Dados Orientado a Objetos - Estado Atual de Desenvolvimento e Implementação"**, in anais VI SBBD, 1991, pp. 95-107.