Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica
Departamento de Engenharia da Computação e Automação Industrial

Este exemplar corresponde à redação final
da Tese defendida por Ivan Luiz Marques
Ricarte e aprovada pela Comissão Julgadora
em 08/junho/1987.

SISTEMAS DE GERÊNCIA DE DADOS PARA PROJETO AUXILIADO POR COMPUTADOR

por: **Ivan Luiz Marques Ricarte**
Orientador: **Prof. Dr. Léo Pini Magalhães**

/87

Tese de Mestrado apresentada à Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas

Junho de 1987

Este trabalho contou com o apoio financeiro da
Fundação de Amparo à Pesquisa do Estado de São Paulo -
FAPESP.

Dedico este trabalho
à minha família

AGRADECIMENTOS

Aos meus pais, cujo importante apoio auxiliou a realização deste trabalho.

À Nuria, presença indispensável.

Ao Professor Dr. Léo Pini Magalhães, não apenas pela orientação deste trabalho como também pelo apoio, confiança e amizade, que muito incentivaram a sua concretização.

Ao Professor Dr. Mario Jino, por sua valiosa revisão deste texto.

Ao Professor Dr. Yuzo Yano, pelo seu incentivo.

Ao Armando, Sérgio e Helga, colaboradores e amigos.

Aos colegas do Grupo PAC/BD, pelo companheirismo.

Aos estagiários que auxiliaram na implementação dos protótipos dos sistemas propostos.

À FAPESP, pelo apoio financeiro.

E a todos aqueles que colaboraram, direta ou indiretamente, para a realização deste trabalho.

RESUMO

Em informática, duas linhas de pesquisa vêm se destacando há mais de uma década: Sistemas de Banco de Dados e Projeto Auxiliado por Computador (PAC ou CAD, na sigla anglo-saxônica). Nos últimos anos, destacou-se a necessidade da integração destas duas áreas principalmente para as aplicações em engenharia, sendo que a tecnologia desenvolvida para aplicações convencionais de Sistemas de Banco de Dados não se mostrou totalmente adequada a aplicações PAC.

A proposta deste trabalho é no sentido de estudar sistemas para armazenamento e manipulação de dados com características adequadas a aplicações de Engenharia, sendo apresentados dois sistemas de software, SIGA e GERPAC. O SIGA inclui as funções básicas de tais sistemas de gerência de dados, levando em conta as necessidades das aplicações não-convencionais. O GERPAC constitui um Sistema de Gerência de Banco de Dados suportado a partir do primeiro sistema e com características adequadas a aplicações na área de Projeto Auxiliado por Computador que suporta um modelo de dados entidade-relacionamento estendido para PAC.

ABSTRACT

In Computer Science, two directions of research have been emphasized since the 70's, Data Base Systems and Computer Aided Design (CAD). In the last years, it has been a growing interest in the integration of both areas of research, mainly in Engineering applications. However, the conventional technology of Data Base Systems has not been adequate to CAD applications.

This work presents two software systems for data storage and manipulation with features adequate to Engineering Design: SIGA and GERPAC. SIGA has the basic functions needed by data systems for non-conventional applications. GERPAC is a Data Base Management System supported by SIGA, directed to CAD applications, with a data model derived from the Entity-Relationship Model.

Conteúdo

Lista de Figuras

Capítulo 1

INTRODUÇÃO

A utilização de sistemas computacionais em Engenharia, notadamente na área de auxílio ao desenvolvimento de projetos, motivou diversas pesquisas conjuntas em Ciência de Computação e Engenharia. Um exemplo clássico é o desenvolvimento alcançado na área de Computação Gráfica, não apenas na representação visual de objetos como também como ferramenta de interação entre usuários e sistemas computacionais.

Uma linha de pesquisa que não teve desenvolvimento conjunto nestas duas áreas, Ciência de Computação e Engenharia, foi a utilização de estruturas para o armazenamento e manipulação de dados. Na área administrativo-comercial, o interesse pelo desenvolvimento de tais estruturas é antigo, culminando com o desenvolvimento de Sistemas de Banco de Dados, sistemas de software que armazenam, manipulam e controlam grandes quantidades de dados. Na área de Engenharia, a necessidade da utilização de sistemas semelhantes para a gerência do armazenamento e manipulação de dados foi sentida somente quando tais sistemas já estavam consolidados para aplicações comerciais.

Em PAC — Projeto Auxiliado por Computador, a necessidade de se utilizar Sistemas de Gerência de Banco de Dados (SGBD) está identificada [/ENC80/,/ENC82/]. As tentativas de se adaptar sistemas utilizados na área administrativo-comercial diretamente para a área de Engenharia não se tem mostrado satisfatórias, havendo muitos autores que enfatizam a necessidade do desenvolvimento de Sistemas dedicados a cada aplicação [/DOL86/].

O presente trabalho vem propor uma alternativa no desenvolvimento de SGBD para aplicações em Engenharia, aproveitando parte da experiência adquirida na área administrativo-comercial (as aplicações ditas "convencionais") sem no entanto buscar adaptar SGBD disponíveis no mercado para o suporte das necessidades que são específicas da área de Engenharia. Tal trabalho se enquadra no contexto dos trabalhos do Departamento de Engenharia da Computação e Automação Industrial da Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas (DCA/FEE/UNICAMP), onde uma das áreas de pesquisa visa

o desenvolvimento de Sistemas PAC, sistemas onde Bancos de Dados assumem papel fundamental. Trabalhos anteriores levantaram as necessidades particulares de Bancos de Dados para tais sistemas, propondo ferramentas para a modelagem de dados [/DEL87/]. O atual trabalho vem mostrar como podem ser implementados SGBD com as características desejadas.

Esta dissertação é composta por sete capítulos. O primeiro deles constitui esta Introdução, onde foi colocado de maneira sucinta o problema que irá ser discutido nos próximos capítulos. O segundo capítulo introduz Sistemas de Bancos de Dados, apresentando diversos aspectos relativos a tais sistemas em aplicações convencionais. No Capítulo 3 aborda-se a questão da utilização de Sistemas de Banco de Dados em Engenharia, particularmente em PAC. Neste capítulo há uma breve descrição relativa aos aspectos da modelagem de dados em PAC, introduzindo o MER/PAC [/DEL87/], e sobre as particularidades de um SGBD para PAC.

A partir do quarto capítulo, passam a ser apresentadas as soluções propostas para os problemas levantados. Neste capítulo, apresenta-se o Sistema de Gerência de Armazenamento Associativo (SIGA), sistema de software proposto para ser utilizado como núcleo de outros sistemas que armazenam e manipulam dados, contendo as funções básicas para tais aplicações e apresentando características que o tornam adequado às necessidades detectadas. Já o Capítulo 5 contém a descrição do Sistema Gerenciador de Dados para PAC (GERPAC), sistema de software que utiliza SIGA como núcleo para um SGBD dedicado para aplicações PAC.

Os Capítulos 6 e 7 encerram esta dissertação, contendo as Conclusões e as Referências, respectivamente.

Capítulo 2

SISTEMAS DE BANCO DE DADOS

2.1 HISTÓRICO

Sistemas Computacionais foram inicialmente desenvolvidos como máquinas calculadoras com grande capacidade de processamento de operações aritméticas e velocidade muito superior àquelas obtidas por sistemas similares disponíveis. Aproveitando sua capacidade de processamento, já na década de 1950 iniciou-se a aplicação destes sistemas na área de processamento de dados comerciais. O desenvolvimento nesta área seguiu basicamente duas linhas, que foram [/FRY76/]:

estruturas de dados: era necessário projetar ferramentas que facilitassem a definição de estruturas manipuláveis pelos Sistemas Computacionais para o armazenamento de dados, e

geração de relatórios: era necessário definir pacotes de programas que, a partir das estruturas de dados definidas e dos dados associados armazenados, permitissem sua impressão de forma seletiva (obter apenas a informação desejada entre todos os dados armazenados) e inteligível (sem necessidade de se conhecer a estrutura interna para compreender as informações).

Os primeiros programas na área comercial apresentavam entre suas características uma forte dependência entre as estruturas de dados definidas e os programas de aplicação. Em geral, o programador também definia a estrutura que seria utilizada para armazenar os dados manipulados pela aplicação. Linguagens que ofereciam facilidades para definição de estruturas de dados, como COBOL, alcançavam grande sucesso entre os programadores da área de processamento de dados comerciais.

Em meados da década de 1960 havia uma crescente preocupação em relação às deficiências que tais sistemas de tratamento de informações apresentavam, principalmente devido ao

fato de que cada aplicação mantinha seus dados de forma privativa, acarretando problemas tais como redundância de dados (em aplicações distintas com parte dos dados iguais) e inconsistência entre arquivos de diferentes aplicações [/DAT84/]. Buscou-se então desenvolver sistemas que contornassem pelo menos parte destas deficiências, apresentando características semelhantes às dos posteriores SGBD (sigla adotada para Sistemas de Gerência de Banco de Dados); alguns destes sistemas propunham objetivos ainda hoje não plenamente alcançados, como linguagem natural de consulta e deduções de novas informações a partir de inferências sobre as informações existentes [/FRY76/]. Até o final dessa década, vários sistemas protótipos haviam sido desenvolvidos, muitos deles propostos como extensões a linguagens de programação visando a manipulação de bases de dados.

No início da década de 1970, diversos sistemas comerciais de tratamento de dados estavam disponíveis (IMS, IDMS, Total, ADABAS [/FRY76/,/TSI77/]). Alguns destes sistemas buscavam estar de acordo com as especificações do DBTG (Data Base Task Group), grupo subordinado ao Comitê de Linguagens de Programação da CODASYL (Conference on Data Systems Languages) que fez várias recomendações para o desenvolvimento de SGBD e baseadas principalmente na linguagem de programação COBOL, que deram origem ao Modelo Rede (veja Item 2.2). Embora esta proposta buscasse alcançar uma generalização na área de desenvolvimento de SGBD, não chegou a atingir o "status" de padrão, pois outras abordagens posteriores trouxeram soluções melhores para os mesmos problemas. Desta forma, não bastava a um SGBD seguir as especificações DBTG para garantir sua aceitação no mercado; era necessário convencer os usuários de suas qualidades [/GAL76/].

Grande parte dos trabalhos desenvolvidos na década de 70 diziam respeito à questão de modelos de dados, ou seja, à forma pela qual o usuário do SGBD visualiza o conjunto de suas informações. Modelos de Dados visam permitir que a aplicação dependa o mínimo possível de detalhes do armazenamento físico, alcançando a independência dos dados, principal objetivo de um SGBD [/DAT84/]. Os primeiros SGBD desenvolvidos ainda dependiam bastante dos detalhes de armazenamento, sendo necessário que o usuário especificasse os caminhos de acesso aos dados. Outras propostas, como o Modelo Relacional [/COD70/] e o Modelo Entidade-Relacionamento [/CHE76/], buscavam isolar cada vez mais a representação dos dados como estabelecida pelo usuário do SGBD da representação física destes dados, definida internamente. Este conceito, de níveis de abstração dos dados, será abordado no próximo item.

2.2 NÍVEIS DE ABSTRAÇÃO E MODELOS DE DADOS

Na utilização de Sistemas de Informação, são necessárias representações dos fatos do mundo real em termos formais. As fronteiras que separam o "mundo real" de uma "descrição formal" é tênue, pois qualquer descrição que dele se faça já deverá seguir alguma espécie de forma-

lização, utilizando conceitos menos abstratos. Quanto menos formal seja esta descrição, ou ainda, quanto mais inteligível ela seja para pessoas leigas, sem o conhecimento de ferramentas formais, diz-se que mais alto é o nível de abstração associado.

No caso de tratamento de informações por sistemas computacionais, o nível de abstração mais baixo é o que trata os dados na forma de "bits", unidades lógicas de informação que podem assumir apenas dois estados. As diversas representações que as informações podem assumir entre esta forma mais elementar e o mundo real estão contidas nos diversos níveis de abstração que podem ser definidos entre estes dois extremos. Não existe padronização sobre a quantidade e os nomes de níveis de abstração de informações e dados. A Figura 2.1 apresenta um quadro resumindo algumas propostas encontradas na literatura e a convenção que será adotada neste trabalho.

Quando se pretende obter uma representação do mundo real, os primeiros processos de abstração são executados naturalmente pela pessoa encarregada de descrevê-lo. Entre todas as informações existentes no mundo real, esta pessoa percebe apenas parte delas. Deste mundo perceptível, ela deve extrair quais as informações de interesse para o problema que está sendo tratado. Uma descrição informal (e.g., em linguagem natural falada ou escrita) destas informações constitui a representação a nível conceitual.

O próximo passo é a formalização desta representação informal. Para tanto são utilizadas ferramentas formais, como a Teoria dos Conjuntos [/VETS82/]. Neste nível pode-se representar conceitos tais como objetos, onde classes de objetos podem ser representadas por conjuntos e os objetos como elementos de conjuntos. Outros conjuntos podem representar as possíveis propriedades que podem ser associadas a objetos. Relacionamentos podem ser representados por associações entre elementos de diversos conjuntos, ligando objetos a propriedades e/ou a outros objetos. O conceito de o que é objeto e o que é propriedade é no entanto relativo, dependendo não só dos fatos que estão sendo descritos como do ponto de vista de quem faz a descrição. A descrição que se obtém por estes meios está associada ao nível descritivo.

A representação a nível organizacional corresponde a uma descrição lógica das informações contidas no nível descritivo, já utilizando termos computacionais como registros e arquivos lógicos. Neste ponto, quem define a representação já deve ser um usuário do sistema computacional, que utiliza para tanto alguma especificação mais formal, como linguagens de programação.

No nível físico, estas informações assumem uma representação bem próxima ao nível mais baixo de abstração, que corresponde ao mundo dos "bits". Neste nível os dados são tratados como registros físicos, aos quais estão associados endereços em memória (principal e/ou disco) que devem ser especificados para a recuperação dos dados.

Modelos de Dados são ferramentas que permitem formalizar uma representação da parte do mundo real sobre a qual se tem interesse. O nível de abstração associado a esta repre-

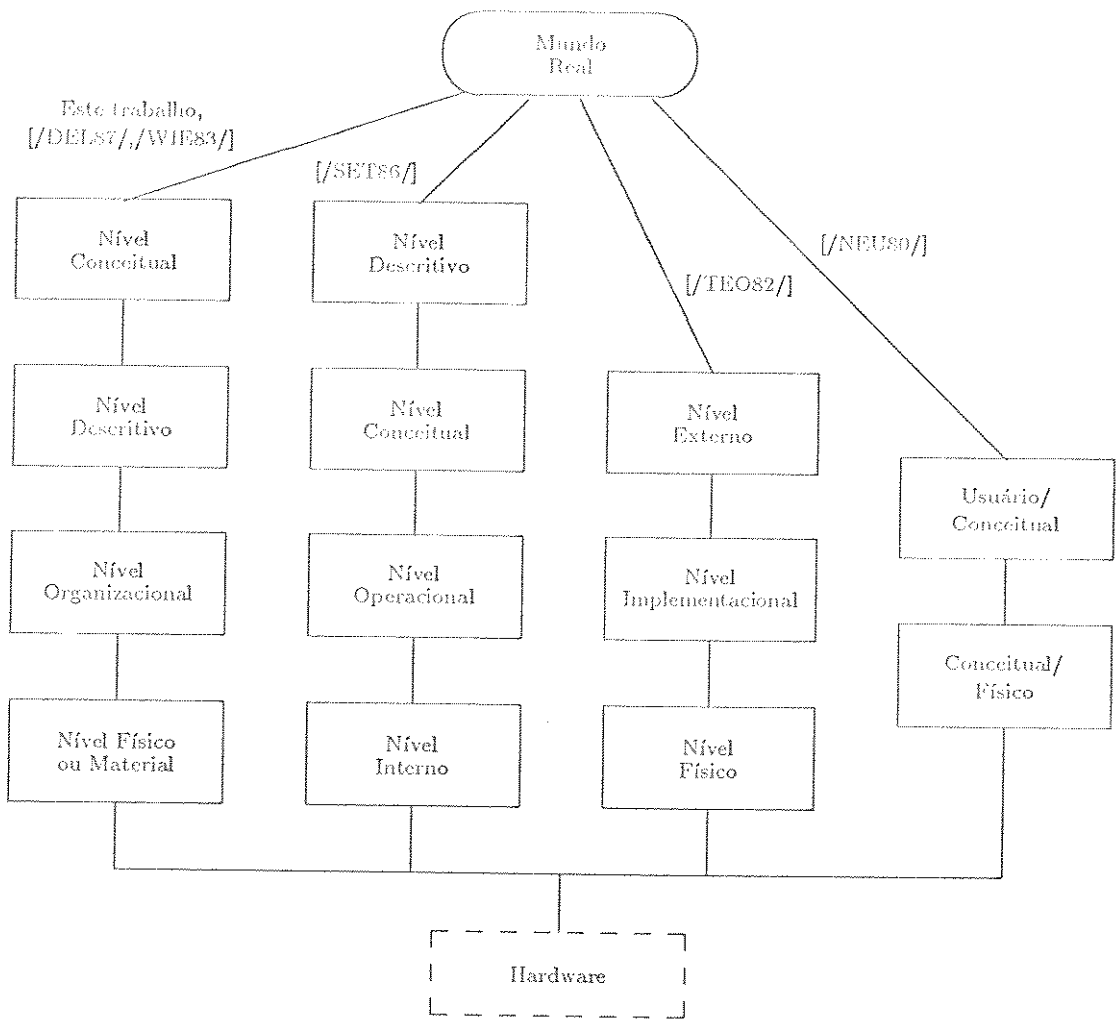

Figura 2.1: Níveis de Abstração de Dados.

sentação obtida pelo modelo de dados depende de cada proposta.

Os primeiros modelos de dados propostos surgiram como extensões a linguagens de programação. Desta forma, suas primitivas (elementos através dos quais o usuário representa as informações de seu interesse) estavam muito próximas das primitivas das linguagens utilizadas como base para o desenvolvimento, até mesmo com relação à necessidade de o usuário definir caminhos de acesso aos dados. Desta forma, a representação obtida por qualquer um destes modelos está contida no nível organizacional. Entre estes modelos, pode-se citar o Modelo Hierárquico e o Modelo Rede [/DAT84/,/TSI77/].

Outros modelos foram propostos na tentativa de isolar cada vez mais os detalhes computacionais da representação obtida pelo modelo de dados. Um exemplo é o Modelo Relacional, onde as informações são representadas como relações, ou subconjuntos de produtos cartesianos entre os conjuntos de valores envolvidos (domínios) [/COD70/,/DAT84/]. O conceito de relação pode ser enquadrado no nível descritivo; no entanto, SGBD que utilizam o Modelo Relacional não implementam relações mas sim tabelas, de forma que o ferramental oferecido nestes casos permite uma representação apenas a nível organizacional, com a diferença que o usuário não necessita ater-se a questões como as formas de acesso, estando assim este modelo a um nível mais alto que os dois outros citados anteriormente.

Diversas outras propostas de modelos de dados surgiram no sentido de obter representações a um nível mais abstrato, com maior conteúdo semântico. Entre as propostas que alcançaram maior aceitação sobressaíram-se aquelas que utilizam a abordagem entidade-relacionamento e, entre estas, destaca-se o Modelo Entidade-Relacionamento (MER) [/CHE76/], proposto inicialmente como uma forma de unificar a representação obtida com os modelos até então mais utilizados — Rede, Hierárquico e Relacional. Outra proposta de abstração da representação de dados está incluída nos conceitos de Agregação e Generalização [/SMI82/,/SMI80/]. Outros modelos de dados, como Binários, Infológicos e mesmo Redes Semânticas, utilizados na área de Inteligência Artificial, podem ser citados nesta categoria cuja representação dos dados se encontra mais próxima do nível descritivo [/TSI82/].

Em geral, propostas como as do MER são utilizadas como forma de permitir uma representação preliminar dos dados a um nível mais próximo do usuário, que deve posteriormente transcrever a descrição obtida em termos de modelos de dados que são suportados por computadores, principalmente para o Modelo Relacional. A tentativa de descrever particularidades do nível organizacional em modelos como o MER faz com que estas propostas se aproximem cada vez mais deste nível em detrimento de suas características de representação a nível descritivo, como no caso da introdução do conceito de dependência de identificação [/CHE77/], que está ligado a questão de chaves (atributos que permitem identificação única de um elemento) e assim ao nível organizacional.

A proposta do MER/PAC [/DEL87/], modelo de dados que estende o MER e sucintamente descrito no Item 3.3, procurou se ater a definições a nível descritivo apenas, não

havendo uma preocupação no sentido de mapear a representação obtida com este modelo para representações em modelos do nível organizacional. Este tipo de preocupação existia na proposta do MER (modelo do qual o MER/PAC é derivado) pois havia a necessidade de se obter a aceitação dos adeptos de sistemas já existentes para a abordagem proposta, ressaltando a aplicabilidade da abordagem Entidade-Relacionamento como uma ferramenta para o projeto lógico de bases de dados convencionais. Atualmente este modelo de dados é largamente adotado e o desenvolvimento de sistemas de tratamento de dados que o utilizem diretamente já é viável (sob o ponto de vista computacional) e aceitável (sob o ponto de vista de usuários potenciais), tendo sido esta a abordagem adotada neste trabalho.

2.3 ESQUEMAS DE UM SGBD

Em um SGBD, os dados armazenados podem ser percebidos sob diferentes pontos de vista. Um usuário que acesse estes dados através de um programa de aplicação (para obter seu saldo bancário, por exemplo) não está em geral interessado em conhecer detalhes das estruturas utilizadas para a represetação dos dados. Outros usuários podem estar interessados na descrição destes dados, ou em trabalhar com apenas uma parcela do conteúdo global dos arquivos ou ainda em otimizar o acesso às informações.

Para expressar estas diferentes visões, os dados armazenados em uma base de dados podem ser vistos de acordo com diferentes esquemas. Um dos estudos mais aceitos, do grupo ANSI/X3/SPARC, propõe três níveis de esquemas: Externo, Interno e Conceitual (Figura 2.2). O mapeamento entre os diversos esquemas é responsabilidade do Sistema de Gerência de Banco de Dados — o SGBD, que deve realizá-lo de forma transparente ao usuário.

O Esquema Externo foi proposto para permitir uma interface entre o usuário final e a descrição do conjunto de todos os dados armazenados. Através do Esquema Externo, os dados são representados tais como vistos pelo usuário final do SBD, que não necessita visualizar todos os dados armazenados na base de dados mas apenas aqueles que dizem respeito à sua aplicação. O Esquema Externo pode ser encarado como uma imagem do Esquema Conceitual e não apenas um subconjunto deste. O usuário não trabalha diretamente com a Base de Dados, mas sim com uma espécie de "Base de Dados virtual" composta pelo subconjunto dos dados que lhe interessa devidamente transformado para a sua visão (conceito de "subesquema", introduzido inicialmente na proposta CODASYL/DBTG [/MIN76/,/DE 81/]). Esta interação indireta pode ser realizada tanto através de programas de aplicação, onde o usuário pode ser totalmente isolado dos detalhes da descrição dos dados, como através de acesso interativo ao subconjunto dos dados que lhe interessa.

O Esquema Interno contém a descrição física dos dados, representada em termos associados ao nível físico de abstração (apresentado no item anterior). Este Esquema não é diretamente acessado por usuários do Sistema de Banco de Dados, sendo que os concei-

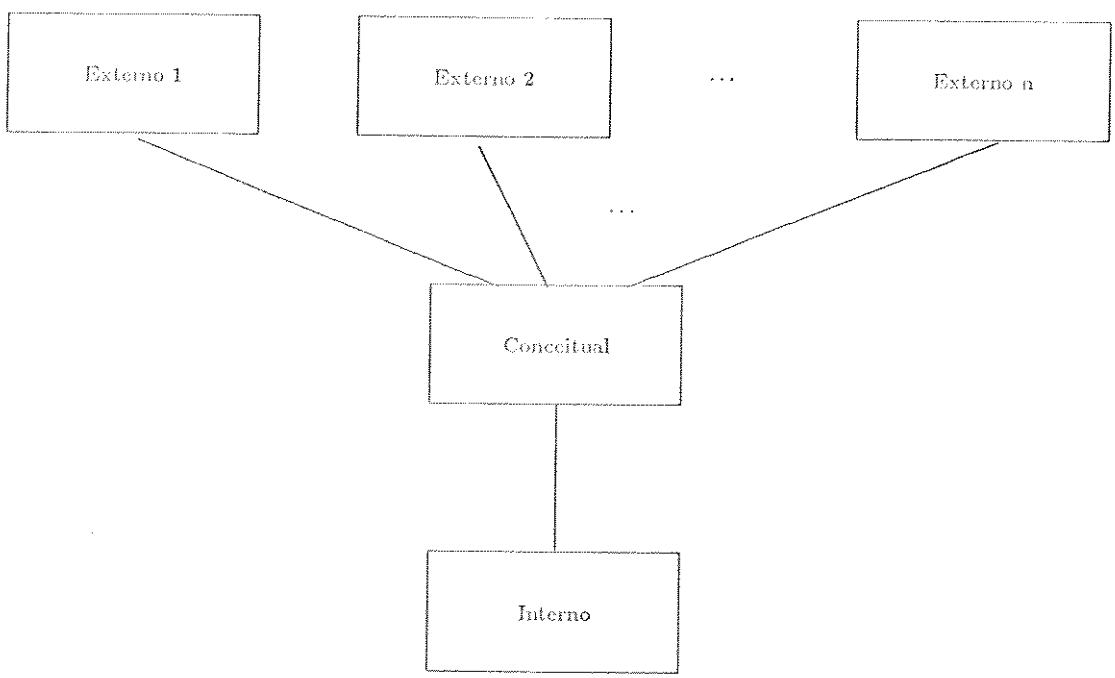

Figura 2.2: Esquemas de um SGBD.

tos envolvidos pela representação a este nível — estruturas de armazenamento, otimização de acesso — são de interesse do projetista do SGBD. As representações neste nível podem assumir variadas formas, podendo ser definidas de maneira a permitir uma otimização da tradução entre o Esquema Conceitual, que contém a descrição dos dados a um nível lógico mais elevado, e o Esquema Interno.

O Esquema Conceitual tem como principal função prover meios para a descrição lógica dos dados armazenados na base de dados do Sistema. Constitui uma referência central tanto para os diversos usuários do Esquema Externo como para o Esquema Interno. A pessoa (ou grupo de pessoas) que manipula as informações do Esquema Conceitual é o Administrador do Banco de Dados (ABD).

O SGBD deve suportar um dos modelos de dados citados no item anterior como ferramenta para o ABD descrever o Esquema Conceitual. Desta forma, a descrição deste Esquema se enquadra em um dos níveis de abstração da Figura 2.1, em geral no Nível Organizacional (utilizando o Modelo Relacional, Rede ou Hierárquico). Descrições contidas no Nível Descritivo são possibilitadas através da utilização de modelos de dados com maior conteúdo semântico, como o Entidade-Relacionamento. Como os SGBD conhecidos não suportam diretamente descrições neste nível, o ABD em geral produz duas descrições do Esquema Conceitual, uma a Nível Descritivo (utilizando um diagrama Entidade-Relacionamento, por exemplo) e outra a Nível Organizacional (traduzindo o diagrama em termos do modelo de dados suportado pelo SGBD).

Para eliminar esta dicotomia de descrições do Esquema Conceitual, existem trabalhos no sentido de propor uma estruturação do SGBD em quatro esquemas [/DE 81/], subdividindo o Esquema Conceitual em dois, um considerando a descrição semântica ("enterprise schema") e outro considerando os aspectos relacionados ao SGBD suportado ("canonical schema"). Outra proposta é simplesmente não utilizar descrições a Nível Organizacional, implementando SGBD que suportem diretamente a representação semântica, sendo esta a abordagem do presente trabalho.

2.4 ASPECTOS DE CONTROLE

O papel do SGBD como um sistema que suporta um modelo de dados esteve em evidência no item anterior, onde o Sistema de Gerência foi apresentado como o responsável pelo mapeamento entre os diversos esquemas do Sistema de Banco de Dados. No entanto, existem aspectos totalmente distintos deste mas igualmente importntes sob responsabilidade do SGBD, que são os aspectos de controle.

O controle pelo qual o SGBD é responsável diz respeito principalmente à questão da consistência, ou seja, à garantia de que os dados armazenados correspondem a um estado correto em relação à realidade. Neste item serão abordados dois entre os principais aspectos

relativos ao controle de dados:

recuperação: abordando o problema de falhas em sistemas computacionais; e

integridade: apresentando questões relativas ao armazenamento de dados inválidos.

A exposição deste item está baseada principalmente no texto de Date [/DATE83/].

Recuperação é o processo de levar a um estado correto um banco de dados que foi levado a um estado incorreto ou suspeito após alguma falha de processamento. A única maneira de possibilitar a recuperação de um sistema é através do armazenamento redundante de informações referentes à base de dados original. Praticamente, o controle de recuperação de dados pode ser realizado utilizando dois arquivos:

- arquivo de "back-up", uma cópia completa da base de dados;
- arquivo "log", um registro de todas as alterações realizadas na base de dados desde a obtenção do último "back- up".

As técnicas básicas de recuperação são duas:

refazer as alterações, utilizando o arquivo "log" para atualizar o arquivo de "back-up", para os casos em que a base de dados foi danificada;

desfazer as alterações, utilizando apenas o arquivo "log", para os casos em que as alterações desejadas não foram completadas mas não houve prejuízos à base de dados.

Cada operação que um usuário realiza sobre a base de dados é uma transação. Uma transação pode ser composta por mais de uma alteração sobre a base de dados (uma operação de transferência, por exemplo); no entanto, as alterações em uma transação ou são todas realizadas ou nenhuma é realizada. Esta é a unidade com que o processo de recuperação trabalha.

Infelizmente, nem todas as falhas que podem ocorrer em um sistema são facilmente previsíveis. É possível classificar os tipos de falhas que ocorrem em sistemas computacionais em três categorias:

- falhas da transação, que por sua vez podem ser ou não explicitamente manipuláveis pela aplicação;
- falhas do sistema, que afetam as transações mas não prejudicam a base de dados;

- falhas do meio de armazenamento, que podem danificar total ou parcialmente a base de dados, afetando as transações correntes no momento da falha.

Uma maneira de agilizar o processo de recuperação é a tomada de "checkpoints". Uma tomada de "checkpoint" consiste do processo de efetivar (ou seja, transferir da memória principal para a memória secundária), em intervalos pré-especificados, as informações referentes ao arquivo "log" e as alterações na base de dados, armazenando informações referentes ao estado do sistema no momento desta operação em um arquivo de "checkpoint". Desta forma, no caso de uma falha do sistema, permite-se saber quais transações devem ser refeitas ou desfeitas quando do reinício da operação. Para este propósito, pode-se dividir transações em cinco categorias. Seja T_c o momento da tomada de "checkpoint" e T_f o momento (posterior a T_c) em que ocorreu a falha do sistema:

T_1. Transação iniciada e encerrada antes de T_c;

T_2. Transação iniciada antes de T_c e encerrada antes de T_f;

T_3. Transação iniciada antes de T_c mas não encerrada até T_f;

T_4. Transação iniciada após T_c e encerrada antes de T_f;

T_5. Transação iniciada após T_c mas não encerrada até T_f.

As transações do tipo T_1 não são afetadas por processos de recuperação, pois a tomada de "checkpoint" garantiu que as alterações realizadas foram efetivadas. As transações do tipo T_3 e T_5 (não encerradas até o momento da falha) devem ser desfeitas, utilizando o arquivo "log". As transações do tipo T_2 e T_4 (encerradas antes do momento da falha) devem ser refeitas, pois como não houve tomada de "checkpoint" entre o momento final da transação e o momento da falha, não há garantia que as alterações tenham realmente sido efetivadas.

Para garantir a consistência das informações da base de dados, a lógica de programas de recuperação - tanto para o *refazer* como para o *desfazer* - devem ser idempotentes, ou seja,

$$RECUP(RECUP(RECUP \ldots (RECUP(x)))) = RECUP(x)$$

onde RECUP representa ou a operação de refazer ou a operação de desfazer, e x é qualquer transação.

Integridade é a propriedade de um Sistema de Banco de Dados a qual assegura que os valores armazenados são válidos. Subsistema de Integridade é o componente do SGBD responsável pelo monitoramento das operações de atualização e detecção de violações à integridade, além da tomada de ações apropriadas. Este subsistema consiste de um conjunto de restrições (as regras de integridade) definindo quais erros devem ser checados, quando fazer os testes e o que fazer caso se verifique o erro. Estas regras, por sua vez, podem ser divididas em dois grandes grupos:

regras de domínio: referente ao valor isolado associado a um atributo; é a definição de um domínio;

regras de relação: referente a admissibilidade de um valor no contexto dos relacionamentos associados.

2.5 COMENTÁRIOS

Neste capítulo buscou-se apresentar uma visão geral sobre Sistemas de Banco de Dados e o sistema de software associado, o Sistema de Gerência de Banco de Dados (SGBD). Os pontos que foram abordados abrangem principalmente aqueles que constituirão a base para o desenvolvimento de capítulos posteriores, não havedo a pretensão de se tratar todas as questões referentes ao assunto ou mesmo detalhar os pontos expostos. Desta forma, diversos pontos de real interesse na área de Sistemas de Banco de Dados não foram abordados, entre alguns dos quais serão descritos a seguir.

- *Implementações de SGBD*

  Sistemas desenvolvidos e apresentados na literatura, tais como o Sistema R [/DAT84/, /BLA81/, /AST76/] e INGRES [/STO76/] incorporam diversos aspectos relativos a SGBD abordados neste capítulo, não apenas em relação à arquitetura como também em relação aos aspectos de controle. Em relação ao modelo de dados, estes dois sistemas utilizam a abordagem relacional, que é uma das principais tendências no desenvolvimento de SGBD. Outra tendência em destaque é a que busca a utilização de outros modelos de dados que permitam ao usuário trabalhar em níveis de abstração mais altos do que o atingido nas implementações baseadas no Modelo Relacional, seguindo principalmente a abordagem entidade-relacionamento. Neste grupo podem ser enquadrados GERM [/BEN81/], DBM-2 [/CHI81/] e o próprio trabalho apresentado nesta dissertação.

- *Outros aspectos de controle*

  Dois importantes aspectos relativos ao controle de dados sob responsabilidade do SGBD não foram abordados no Item 2.4. Um deles diz respeito à questão de segurança dos dados, problema que tem até mesmo reflexos sociais: como garantir que dados privativos não serão devassados por usuários não autorizados? Entre os vários processos utilizados, destacam-se a identificação de usuários no momento de acesso ao SBD, a utilização de um subsistema controlando regras de segurança, a classificação de dados e usuários em diversos níveis e ainda a criptografia [/DAT83/]. O outro aspecto diz respeito aos problemas que podem ocorrer quando mais de um usuário pode acessar a

base de dados simultaneamente; é a questão da concorrência. O problema que ocorre é conhecido como interferência entre transações, sendo que uma das técnicas mais utilizadas para este controle é conhecida como "locking", que define regiões da base de dados que só podem ser acessadas por um usuário de cada vez; outros usuários que tentem acessar uma área em utilização permanecem em estado de espera até que a área seja liberada. Esta técnica pode ocasionar um problema que é a chamada situação de bloqueio ("deadlock"), quando duas ou mais transações entram simultaneamente em estado de espera recíproco. Existem técnicas suplementares no sentido de evitar o problema de bloqueio, como o planejamento de transações (não permitir concorrência entre duas transações que possam gerar conflito) e a rejeição de reserva a uma área que possa levar a uma situação de bloqueio [/DAT83/].

- *Sistemas distribuídos*

  Uma das grandes áreas em estudo na área de Banco de Dados envolve sistemas cujos dados estão de alguma forma armazenados em localizações diversas, ou de forma mais precisa, cuja base de dados pode ser dividida em diversas partes e o acesso que um usuário tem a algumas das partes é muito mais lento que a outras [/DAT83/]. Entre as vantagens trazidas pela utilização de Sistemas de Gerência de Banco de Dados Distribuídos, estão a autonomia local, facilidade de crescimento, aumento da capacidade de armazenamento e flexibilidade. Entre as dificuldades encontradas, citam-se o processamento de questões, a propagação de atualizações e o controle de concorrência.

- *Máquinas de banco de dados*

  Quando da apresentação do nível físico de abstração, foi citado o fato de que a representação utilizada envolvia a questão de endereçamento de memória. No entanto, esta afirmação diz respeito principalmente a sistemas computacionais convencionais. Outra área de interesse em Bancos de Dados envolve o conceito de Máquinas de BD, que incorporam em hardware diversas facilidades que auxiliam o desenvolvimento de SGBD, como acesso associativo e funções básicas de auxílio ao SGBD em microcódigo [/DAT83/, /SMI79/]. Em sistemas desenvolvidos com tais ferramentas, a questão do acesso aos dados poderá ser sensivelmente modificada, de forma que se poderá obter um outro tipo de representação no nível físico.

- *SGBD e IA*

  Embora estudos em Inteligência Artificial (IA) não sejam recentes, atualmente esta área vem tomando grande impulso devido principalmente ao desenvolvimento de melhores ambientes para programação, incluindo linguagens de programação lógica, da qual PROLOG [/DAV85/,/DAH83/] é a mais conhecida. Sua aplicação na área de Banco de Dados abriu novas frentes de pesquisa, como o estudo de Sistemas de Banco

de Dados Dedutivos ou Inferenciais [/CAS86/,/GAL84/], a otimização de consultas em sistemas convencionais [/SIL86/], novas formas de especificação de tipos abstratos de dados [/DAV82/] e estudos em Sistemas de Gerência de Bancos de Conhecimentos [/BRO86/].

É importante ressaltar que o trabalho apresentado nesta dissertação está voltado para o desenvolvimento de sistemas que envolvem conceitos ligados ao Esquema Interno (aspectos apresentados no Capítulo 4) e ao Esquema Conceitual (aspectos abordados no Capítulo 5), buscando obter para este último descrições que se enquadrem no Nível Descritivo de abstração. As questões referentes particularmente ao Esquema Externo, como o controle de visões, não serão diretamente abordadas.

Capítulo 3

BANCOS DE DADOS E PAC

3.1 INTRODUÇÃO

PAC — Projeto Auxiliado por Computador — é um conceito em evolução. Inicialmente utilizado para designar alguns Sistemas de Auxílio à Engenharia, notadamente na área de Análise Estrutural (Engenharia Civil), passou posteriormente a ser utilizado na maior parte dos casos comoum sinônimo de Sistemas de Desenho Auxiliado por Computador (onde a sigla anglo-saxônica CAD, coincidente para os dois tipos de Sistemas — "Computer Aided Design" e "Computer Aided Drafting" — apenas aumentava a semelhança). Nos últimos anos, o termo ganhou uma acepção mais ampla, designando "uma disciplina que provê o conhecimento requerido em computadores (hardware e software), análise de sistemas e metodologia de engenharia para especificar, projetar, implementar, introduzir e utilizar sistemas baseados em computadores para propósitos de projeto" [/ENC83/].

O desenvolvimento de Sistemas PAC acompanhou de perto o desenvolvimento de facilidades gráficas em computadores, principalmente a partir do trabalho pioneiro de Sutherland na área de computação gráfica interativa ("Sketchpad: a Man-Machine Graphical Communication System", 1962) [/NEW81/].

Enquanto isto, o desenvolvimento de Sistemas de Gerência de Banco de Dados (SGBD) se caracterizava como um processo totalmente independente deste, estando voltado principalmente às necessidades comerciais. Somente em meados da década de 1970 é que se ressaltou a aplicabilidade e a necessidade de Banco de Dados em PAC [/VER84/]. Desde então, Bancos de Dados para a utilização em Sistemas PAC vêm sendo objeto de pesquisas, não só quanto ao aspecto da modelagem dos dados mas também quanto ao aspecto da implementação de SGBD adequados a PAC.

3.2 O PROCESSO PAC E BANCOS DE DADOS

Sistemas PAC são desenvolvidos para prover suporte computacional ao processo de projeto. Este processo engloba os procedimentos necessários para, a partir de conhecimentos relacionados a um problema (descrito através do objetivo que se busca alcançar), atingir sua concretização — o projeto, composto por informações que podem ser documentadas e/ou utilizadas para a produção (Figura 3.1) [/ENC83/].

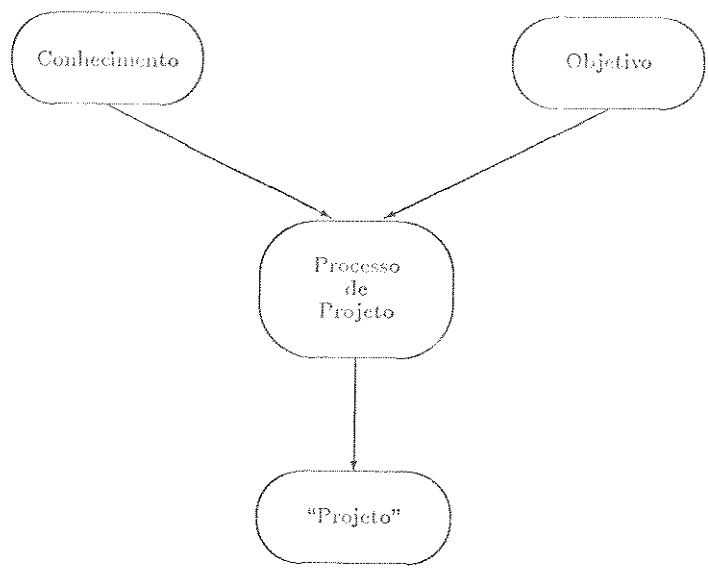

Figura 3.1: Modelo do Processo de Projeto.

Este modelo grosseiro não leva em conta características fundamentais deste processo, a saber:

- o processo não é auto-contido, estando sempre inserido em um contexto maior (processo de nível mais alto); e
- frequentemente o processo é iterativo. Decisões tomadas nas primeiras fases do processo (síntese) não prevêem em geral todo o impacto que podem causar com relação ao objetivo. Os resultados da fase de síntese devem ser analisados e avaliados e, caso não satisfaçam os requisitos especificados, devem ser corrigidos (voltando à síntese) e assim sucessivamente.

Um modelo um pouco mais refinado deste processo é apresentado na Figura 3.2 [/ENC83/]. Quando apoiado pelo computador, este processo é conhecido como o *Processo PAC*. Sistemas

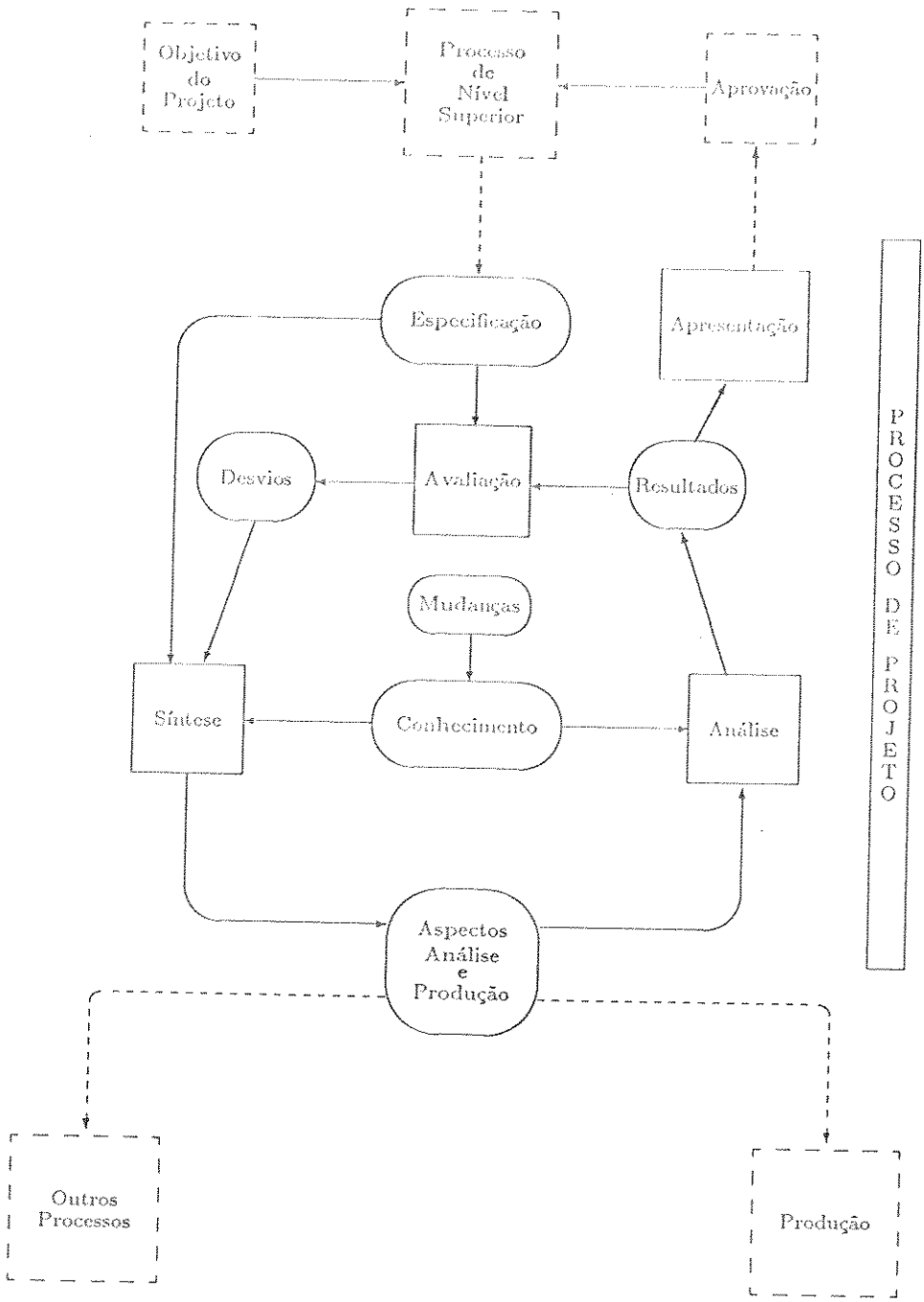

Figura 3.2: Modelo Refinado do Processo de Projeto.

computacionais que suportem o Processo PAC são conhecidos como Sistemas PAC. Por esta figura, torna-se evidente o papel fundamental que é exercido pelo gerenciador das informações dentro destes Sistemas, constituindo estas informações uma espécie de interface entre as fases e iterações do processo.

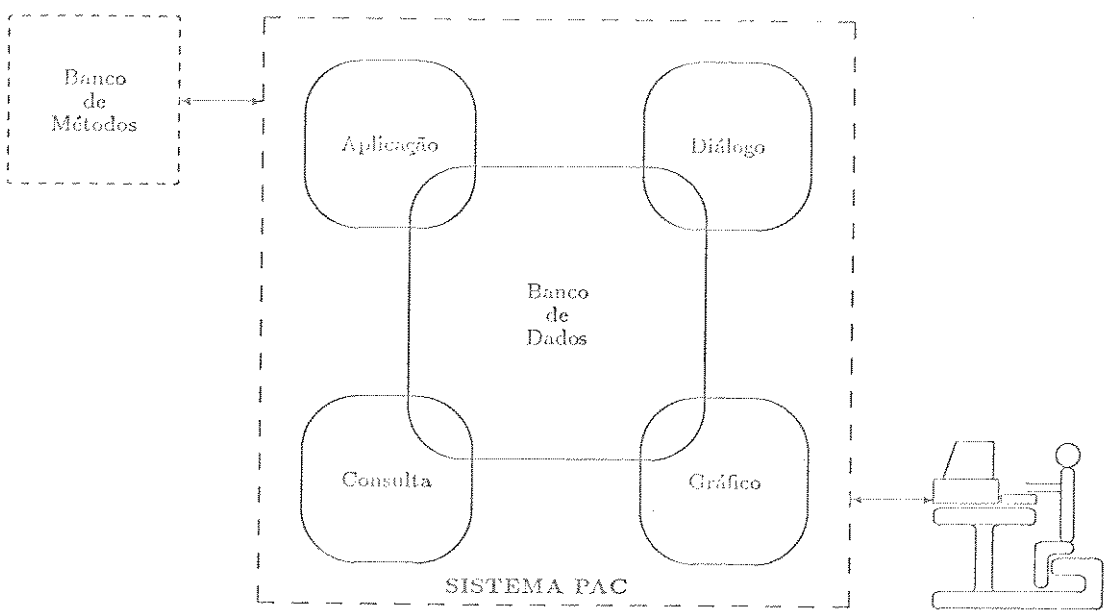

Figura 3.3: Módulos de um Sistema PAC.

A estrutura funcional de um Sistema PAC pode ser representada como na Figura 3.3, onde os diversos módulos do Sistema estão caracterizados [/ENC83/]. Nesta figura destaca-se o papel fundamental do Sistema de Banco de Dados na integração entre os demais módulos do Sistema PAC. Em Sistemas PAC não integrados, cada um destes módulos tem sua estrutura de dados particular, tornando difícil a compatibilização entre eles. A utilização de Sistemas de Banco de Dados como uma "central de dados" do Sistema PAC permite obter vantagens importantes, entre as quais podem ser citadas [/NEU80/]:

- Programas de Aplicação, ao se tornarem isolados de estruturas de dados particulares, são mais fáceis de serem desenvolvidos e mais portáteis, além de que diversos aspectos de desempenho são repassados para o SGBD;

- dados podem ser compartilhados entre os módulos sem a necessidade de reformatação, obtendo ainda a redução da redundância, ao contrário do que ocorre em sistemas tradicionais;

- controle mais completo sobre a consistência dos dados.

O desenvolvimento de estruturas de dados específicas para PAC foi a tônica geral na área até o início da década de 1970 [/VER84/]. Somente por volta de 1975 pode-se notar um interesse crescente na utilização de técnicas de Sistemas de Banco de Dados em aplicações PAC. As primeiras aplicações de SGBD para PAC estavam voltadas à utilização de sistemas baseados em modelos de dados clássicos, como o relacional [/WIL76/,/PAT82/]. No início dos anos 80, no entanto, a experiência adquirida já demonstrava que SGBD desenvolvidos para aplicações comerciais não eram adequados a aplicações PAC [/EAS81/]. Entre as diferenças entre os dois tipos de aplicações destacam-se [/VER84/]:

- SGBD comerciais manipulam poucos objetos com uma grande quantidade de ocorrências associadas a cada um deles; aplicações PAC contêm uma grande variedade de objetos com poucas ocorrências associadas;
- relacionamentos entre objetos são muito mais complexos em PAC do que em aplicações comerciais;
- os esquemas de dados são dinâmicos em PAC, ao contrário das aplicações comerciais onde os esquemas têm vida útil longa;
- o conceito de transação PAC representa uma operação que tem duração indeterminada, ao contrário do conceito de transação em sistemas comerciais (uma operação atômica);
- o usuário PAC não trabalha diretamente sobre o SGBD, mas em geral trabalha sobre outros módulos (como entrada gráfica interativa e simulação) que, por sua vez, traduzem as operações que envolvem dados em termos de operações do SGBD; em aplicações convencionais, esta tradução é praticamente direta, podendo haver casos em que o usuário manipula diretamente o SGBD; e
- as estruturas de tipos de dados necessárias em PAC em geral são complexas, como no caso de representações geométricas, sendo que os tipos de dados suportados em SGBD comerciais (como inteiros, caracteres, reais) não satisfazem às necessidades PAC.

Desde então, duas linhas principais de pesquisa vêm se desenvolvendo paralelamente na área de Bancos de Dados em PAC. A primeira delas diz respeito à adaptação de SGBD comerciais a aplicações PAC [/LOR82/,/NEU80/,/ULF82/], sendo utilizada principalmente a abordagem relacional. A outra linha de pesquisa é direcionada ao desenvolvimento de Sistemas de Banco de Dados específicos para as necessidades PAC, tanto a nível de modelagem de dados quanto a nível de gerenciadores [/FOI82/,/GRA82/]. É nesta última abordagem que se encaixa o presente trabalho.

3.3 MODELOS DE DADOS PARA PAC — O MER/PAC

A abordagem tradicional à questão da modelagem de dados para sistemas computacionais em geral define, independentemente do nível de abstração do modelo, que o mundo a ser modelado tem dois tipos de propriedades — estáticas e dinâmicas — que um modelo de dados deve ser capaz de representar [/TSI82/]. Para tanto, deve existir um conjunto de regras de geração (a Linguagem de Definição de Dados — LDD), que deve ser capaz de exprimir os dois tipos de informação que compõem o "mundo estático", Estruturas e restrições explícitas (Regras), e um conjunto de operações (contemplando as características dinâmicas) sobre este mundo (a Linguagem de Manipulação de Dados — LMD). Esta conceituação, adequada para o modelagem de sistemas tradicionais, não é das mais adequadas no caso de PAC, onde todas as informações, incluindo as estruturas e até mesmo as regras, têm caráter dinâmico.

As características da modelagem de dados em processos PAC foram objeto de estudos em diversos trabalhos em diferentes linhas [/GRA82/,/CHA82/,/LOR82/,/STO83/]. Além disto, outros trabalhos vêm sendo desenvolvidos cujas características procuradas são também adequadas a aplicações PAC, como na área de Tipos Abstratos de Dados [/BER83/,/GUT77/], informação temporal associada aos dados [/ANT80/,/KLO81/] e processos de refinamentos sucessivos aplicados à definição de esquemas conceituais [/BAT80/]. Um avanço importante na área de Banco de Dados foi a definição e utilização de modelos de dados com nível de abstração mais alto que os ditos modelos clássicos e as tentativas no sentido de implementar SGBD que suportem tais modelos. Entre estes modelos, destaca-se o Modelo Entidade-Relacionamento (MER) [/CHE76/], proposto como um modelo que permite uma abordagem unificada para o processo de projeto de bancos de dados, independente do modelo suportado pelo SGBD para a implementação a nível organizacional.

Uma proposta no sentido de unificação destes diversos estudos, abordando em particular a aplicações PAC, foi objeto de estudos em uma tese de mestrado desenvolvida no Departamento de Engenharia da Computação e Automação Industrial da Faculdade de Engenharia Elétrica de Campinas (DCA/FEE/UNICAMP). O principal resultado desta tese foi a proposta de um modelo de dados adequado a aplicações PAC, o MER/PAC [/DEL87/,/DEL86/].

O MER/PAC é uma extensão proposta ao MER que inclui primitivas que visam representar, a nível de modelos de dados, as características do processo PAC. Uma das principais características do MER/PAC diz respeito aos diferentes níveis do processo PAC, que podem ser encarados sob dois pontos de vista, os chamados micro-aspecto e o macro-aspecto. Como já foi dito, um processo de projeto não é auto-contido, sendo que um processo pode deflagrar outros que estarão subordinados ao processo inicial que, por sua vez, pode estar contido em um contexto maior. Esta hierarquia de processos de projeto (ou processos PAC) constitui o objeto de representação de dados Macro do MER/PAC, ou seja, do macro-aspecto. Os dados de cada um dos processos deflagrados são, por sua vez, também representados através do

MER/PAC, constituindo o seu micro-aspecto.

3.3.1 A Modelagem do Aspecto Micro

A representação do aspecto micro do processo PAC envolve os dados referentes aos objetos e suas associações. Parte destes dados pode ser representada através das primitivas originais do MER, tipos de entidades (objetos) e tipos de relacionamentos (associações) e informações associadas (atributos de entidades e relacionamentos). No entanto, conceitos fundamentais em PAC não podem ser representados no MER, como o conceito de objeto complexo [/LOR82/] e de relacionamentos exclusivos [/CHA82/]. Além disto, outro conceito importante, não só na área de PAC mas para Bancos de Dados em geral, é o de métodos de abstração para a representação de informações, destacando-se em particular os conceitos de Classificação, Generalização e Agregação [/SMI82/,/SMI80/].

Classificação representa o conceito de se agrupar instâncias em grupos com características comuns, por exemplo, Beatriz, José e Roberto na classe de Professores. No tratamento do conjunto, detalhes sobre as instâncias são suprimidos e as propriedades do grupo como um todo são enfatizadas. A coleção de instâncias constitui um novo tipo — a classe.

O conceito de generalização apresenta semelhanças com o conceito de classificação. A diferença é que agora não se agrupam mais instâncias, mas sim outras classes que passam a constituir as categorias do novo tipo definido. A operação é equivalente à união da Teoria Matemática de Conjuntos, sendo que são desconsideradas as diferenças entre as categorias e enfatizadas as propriedades em comum do novo tipo; por exemplo, Automóveis e Motocicletas pertencem à categoria de Veículos.

A agregação enfatiza relacionamentos entre diferentes classes, permitindo tratar associações na forma conjunto- relacionamento-conjunto como um todo, desconsiderando os detalhes relativos a cada componente (elemento da associação). A operação definida pela agregação é semelhante ao produto cartesiano da Teoria de Conjuntos.

A proposta de Smith e Smith [/SMI80/] no sentido de desenvolver Esquemas Conceituais é a construção de hierarquias de objetos através da aplicação sucessiva destes conceitos. Desta forma, haveriam hierarquias tridimensionais, sendo que a cada dimensão estaria associado um dos conceitos de abstração (hierarquia de classificação, hierarquia de generalização e hierarquia de agregação). É interessante observar como outras abordagens, principalmente a relacional e a entidade-relacionamento, utilizam ou tentam representar estes conceitos.

No caso da abordagem relacional, o conceito de definição de uma classe está associado à definição de uma relação, e tuplas são associadas a instâncias. Desta forma, pode-se dizer que a classificação é um conceito natural do modelo.

O conceito de agregação é definido através de domínios associados à relação, podendo ser encarado sob dois aspectos:

atributos: corresponde a uma agregação onde os relacionamentos agregados são associações que ligam elementos a suas propriedades. Os atributos constituem os componentes da agregação, sendo que parte destes atributos deve permitir uma identificação de forma unívoca das tuplas de modo que não haja instâncias repetidas, introduzindo desta forma o conceito de chaves; e

ligação entre relações: corresponde a associar tuplas de diferentes relações, o que é obtido neste modelo através da agregação aos atributos de uma relação de atributos que identificam instâncias da outra relação (através de sua chave). Desta forma introduz-se o conceito de chave imprópria ("foreign key"), ou seja, composta de atributos que não constituem propriedades da relação.

Um exemplo da aplicação deste conceito é a conexão entre Automóvel e Motor. Automóvel é um objeto que tem suas características, como fabricante, modelo e cor. Motor também tem seus atributos, como fabricante, modelo, combustível, volume e quantidade de cilindros. A definição destas propriedades constitui um exemplo de agregação onde cada componente é um atributo. Uma outra forma de agregação é a que permite representar que a cada instância de Automóvel está associada uma instância de Motor, sendo para tanto necessário incluir a chave de Motor como chave imprópria de Automóvel (por exemplo, um atributo "modelo do motor" em conjunto com "fabricante do motor" em acréscimo às propriedades de Automóvel). A interpretação semântica implícita é que Motor é parte de Automóvel.

O conceito de generalização não é representável na abordagem relacional. Um exemplo: expressar que as classes Automóvel e Motocicleta pertencem à categoria de Veículos. Para representar esta informação, deveria haver uma relação Veículos com duas tuplas, correspondentes às instâncias Automóvel e Motocicleta. No entanto, Automóvel e Motocicleta não são instâncias que possam ser identificadas por valores de seus atributos, mas sim outras relações que têm por sua vez suas próprias instâncias. Haveria a necessidade de criar um operador especial para permitir que um atributo assumisse não um valor em um domínio mas sim fizesse referência a outras relações de forma a representar informações do tipo Automóvel é um Veículo [/LOR82/].

Na abordagem entidade-relacionamento original [/CHE76/], os conceitos representáveis são os mesmos, com a diferença que parte das informações semânticas estão agora explícitas. A classificação é realizada diretamente através do conceito de tipos de entidades, onde as entidades são associadas às instâncias. O conceito de agregação é também representado através de atributos e na associação entre diferentes tipos de entidades através da definição de tipos de relacionamentos. A informação semântica contida pode ser explicitada através de papéis ("roles"). No entanto, a possibilidade de representar agregações não é ilimitada, pois não se permite definir um tipo de relacionamento que envolva outro tipo de relacionamento. Já o conceito de generalização também não é exprimível através do MER, basicamente pelos

mesmos motivos expressos na análise do caso relacional.

Várias propostas de extensão sobre o MER vêm tentando incluir ferramentas para permitir sanar estas deficiências na modelagem utilizando entidades e relacionamentos, principalmente no sentido de incluir a representação de generalização através de novos tipos de primitivas ou operadores [/SAK83/,/ATZ81/ e /BAT80/,/CAL81/,/KAT81/] e também facilidades para permitir a definição de tipos de relacionamentos envolvendo outros tipos de relacionamentos [/DOG81/,/SAN80/,/SCH80/].

A proposta MER/PAC no sentido de permitir a representação destes conceitos essenciais a PAC foi a primitiva Agrupamento, que permite tratar um grupo de primitivas como uma única. Através de agrupamentos, os conceitos não suportados pelo MER podem ser representados no MER/PAC.

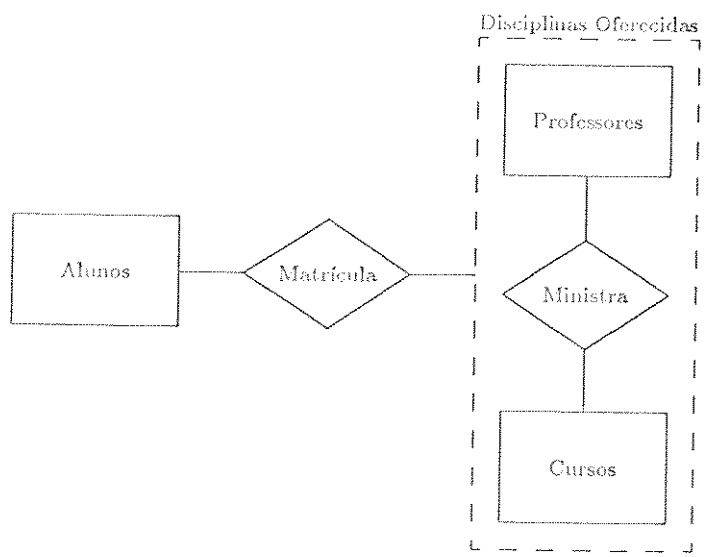

Figura 3.4: Exemplo de Agrupamento (Agregação).

Um exemplo de como o MER/PAC representa relacionamentos envolvendo outros relacionamentos está apresentado na Figura 3.4, através de um Diagrama MER/PAC (semelhante ao diagrama MER, com a introdução do retângulo tracejado para representar agrupamentos). Outro conceito possível de ser controlado através de agrupamentos é o de Objetos Complexos, como no exemplo da Figura 3.5.

Nos dois exemplos das Figuras 3.4 e 3.5, o agrupamento não necessita ter atributos próprios, pois as informações associadas ao objeto representado pelo agrupamento estão distribuídas em seus componentes (tipos de entidade, tipos de relacionamentos e/ou outros

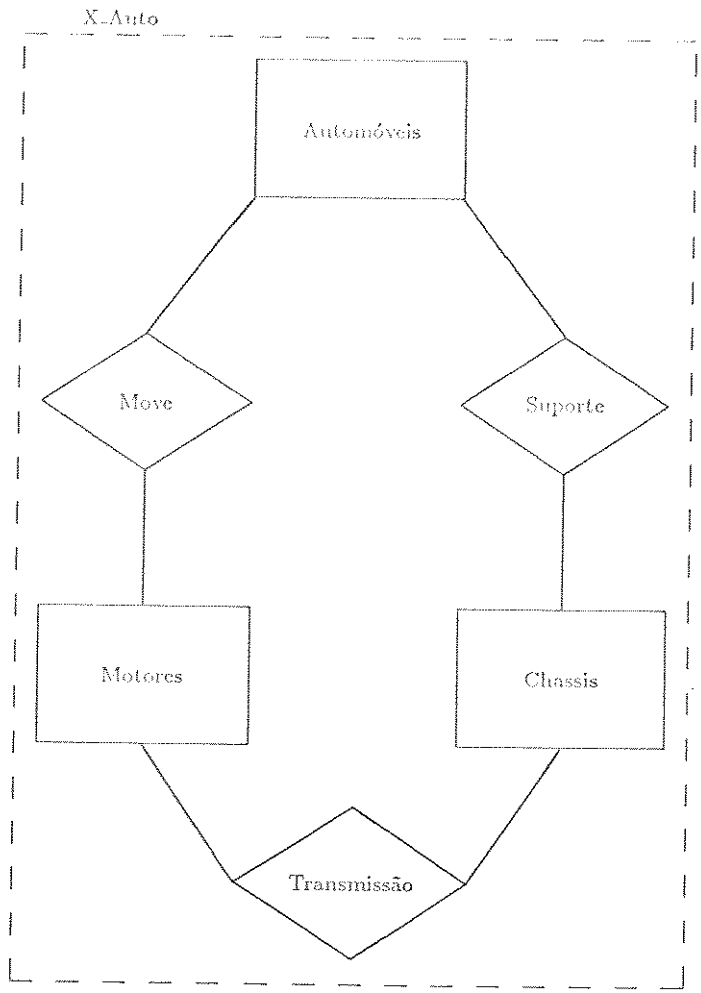

Figura 3.5: Exemplo de Agrupamento (Objeto Complexo).

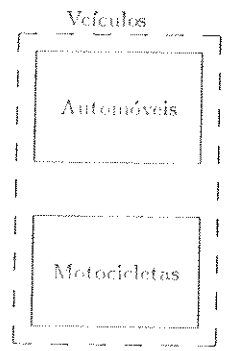

Figura 3.6: Exemplo de Agrupamento (Generalização).

agrupamentos). No caso de generalização (Figura 3.6), além dos atributos ligados a cada categoria podem existir atributos ligados ao agrupamento, representando as informações ligadas à classe geral. Como não há restrições de níveis de profundidade de agrupamentos que podem ser definidos, hierarquias de generalização podem ser igualmente representadas.

Outro tipo de informação que pode ser representada através de agrupamentos é o conceito de Relacionamentos Mutuamente Exclusivos (RME). Nestes casos, o que pode existir são dois ou mais tipos de relacionamentos ligando um tipo de entidade principal a outros tipos de entidades (suas variantes) com caráter excludente (Figura 3.7). Neste caso, o agrupamento também não necessita ter propriedades associadas.

A representação de tão diferentes conceitos através de uma mesma primitiva só é possível através da definição de um outro nível de informação a ela associada. Estas informações são as regras associadas às primitivas MER/PAC. O conceito de regras assemelha-se ao conceito de operadores sobre um conjunto de primitivas [/SAN80/], sendo no entanto mais flexível. Regras podem ser associadas a objetos (tipos de entidades e tipos de relacionamentos) e a agrupamentos (e também a esquemas, como será visto no Item 3.3.2). No caso de objetos, podem ser definidas regras sobre seus atributos especificando aspectos individuais, como faixas de valores válidos para cada atributo, ou interrelacionamentos entre valores de diferentes atributos. Particularmente para tipos de relacionamentos, regras podem ser utilizadas para representar condições que permitam definições e/ou remoções de ocorrências. No entanto, é em agrupamentos que as regras assumem um papel fundamental. Não há nenhuma regra implícita que defina o que constitui a ocorrência de um agrupamento em relação às primitivas que constituem seus elementos, sendo que é desta forma que se obtém a flexibilidade necessária para a representação dos diversos conceitos.

É interessante notar que o MER/PAC, mesmo mantendo o conceito de entidades e relacionamentos fracos (dependentes), não absorveu do MER a diferenciação entre dois tipos

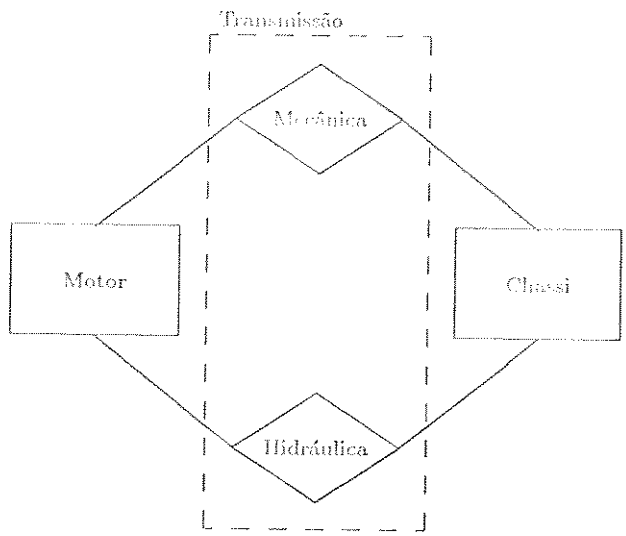

Figura 3.7: Exemplo de Agrupamento (RME).

de dependência (Existência e Identificação) [/CHE77/,/TSI82/]. Entendeu-se que, tratando a nível de entidades e relacionamentos, apenas o conceito de dependência de existência é significativo, enquanto que o conceito de dependência de identificação introduz um tipo de informação que não é relevante no nível descritivo. Este conceito está associado ao conceito de chave imprópria, estando mais ligado a implementações do modelo relacional; o próprio conceito de chave não é significativo no nível descritivo, onde se encaixa a abordagem entidade-relacionamento. Assim como entidades não necessitam de chaves para serem identificadas, também relacionamentos não utilizam o conceito de chaves de entidades associadas (chave imprópria) para sua identificação (a associação é definida entre entidades e não entre parte de seus atributos).

No Modelo Relacional, o conceito de chave imprópria é fundamental para permitir a representação de associações entre diferentes relações. No entanto, sua utilização pode gerar inconsistências, sendo possível definir associações envolvendo algum elemento que não existe (ou por nunca ter sido armazenado, ou por ter sido removido após a definição da associação ou por ter sido alterado o valor do atributo correspondente à chave imprópria). Para evitar este tipo de inconsistência, procedimentos auxiliares de verificação devem ser definidos.

Utilizando a abordagem entidade-relacionamento no Nível Descritivo (ou seja, sem que haja a preocupação de realizar mapeamentos para outros modelos de dados do Nível Organizacional), o conceito de chave imprópria é desnecessário, pois entidades são identificadas de forma independente dos valores de seus atributos. Desta forma, atualizações nos valores

de atributos de entidades não geram inconsistências nos relacionamentos onde estas entidades estejam porventura envolvidas. A definição de relacionamentos só pode ser especificada envolvendo entidades existentes e a remoção de uma entidade implica automaticamente na remoção de relacionamentos que a envolvam. Inconsistências em relacionamentos, como as que ocorriam no Modelo Relacional, não ocorrem na abordagem entidade-relacionamento.

3.3.2 A Modelagem do Aspecto Macro

O controle que deve ser exercido para gerenciar os diversos níveis de processos de projeto que são gerados a partir de um processo inicial é também objetivo do MER/PAC. Para tanto foi proposta a primitiva Esquema. Um Esquema permite referenciar todo um nível de primitivas MER/PAC (no micro-aspecto), representando um determinado processo. Desta forma, existe um Esquema Global que representa o processo inicial, onde são definidas as diretrizes do projeto através de primitivas MER/PAC (incluindo-se outros Esquemas) e de Regras associadas ao Esquema. Os Esquemas referenciados no Esquema Global representam cada um dos novos processos deflagrados que, por sua vez, devem obedecer às regras definidas no Esquema Global e às regras associadas ao processo local.

Não há limites para a quantidade de níveis de Esquemas que estão subordinados a um determinado Esquema Global. Desta forma, deve existir um sistema auxiliar para permitir o controle de consistência a nível global, ou seja, consistência entre diferentes esquemas associados. A execução deste controle caracteriza uma Transação Macro, um conceito particular a PAC, diferindo do conceito de transação em SBD comerciais principalmente com respeito a sua duração (uma transação macro pode perdurar durante todo o período de execução do projeto, quando se obterá um produto final consistente). Este sistema deve permitir não só o controle da evolução do projeto (características temporais) como deve incluir a opção de alternativas de projeto, que podem ser de dois tipos.

A primeira forma de alternativa de projeto diz respeito às diferentes concepções que o projetista estabelece para modelar o processo no nível em que está trabalhando. Sob o ponto de vista de Banco de Dados, estas diferentes concepções podem ser descritas por diferentes representações, ou seja, esquemas conceituais diferentes associados a um mesmo módulo de projeto. Este conceito é representado no MER/PAC através de Expansões associadas a um Esquema (Figura 3.8).

A outra alternativa de projeto diz respeito aos dados associados a uma determinada representação, onde pode ser interessante trabalhar simultaneamente com distintas configurações de dados operacionais como forma de, por exemplo, criar subsídios para uma tomada de decisão relativa ao processo (e.g., análises de custos, simulações). O conceito MER/PAC associado a este tipo de alternativa é a Versão. Podem existir diversas versões associadas a cada representação (Figura 3.8). O conceito de versão pode também ser usado para o controle

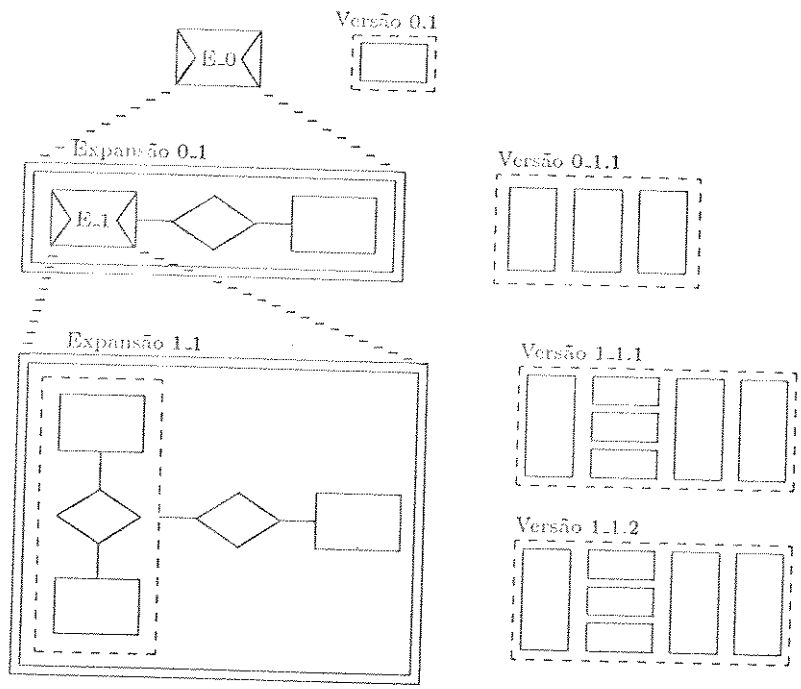

Figura 3.8: Expansões/Versões de um Esquema.

temporal do projeto.

A ocorrência de um Esquema é definida pela identificação de um par Expansão-Versão a ele associado.

O conceito de expansão/versão introduz outra forma de controle associada um Esquema, que é o controle da geração de novas expansões e/ou novas versões. Por exemplo, algumas classes de modificações poderiam invalidar dados já existentes na versão em uso. Se isto não é desejado, uma nova versão deve ser criada. O mesmo é válido com relação a expansões (estrutura), assim como para as regras (novas regras podem invalidar dados ou estruturas existentes, devendo neste caso gerar novas versões).

3.4 SGBD PARA PAC — O GERPAC

A necessidade de se utilizar SGBD em Sistemas PAC é atualmente reconhecida [/VER84/], embora não se tenha obtido o mesmo grau de desenvolvimento que foi atingido na área comercial. As tentativas de se utilizar SGBD comerciais na área de PAC visavam aproveitar as vantagens oferecidas por sistemas já desenvolvidos. No entanto, os modelos de dados suportados por estes sistemas (ditos convencionais) não conseguiriam representar os diversos tipos de informações existentes em aplicações PAC. Entre os SGBD desenvolvidos para aplicações PAC a partir de SGBD comerciais os que tem maior aceitação são aqueles que suportam a abordagem relacional [/ANSU80/]. No entanto, mesmo SGBD relacionais apresentam uma série de deficiências em relação à área de PAC [/LOR82/,/FOI82/]. Algumas das deficiências apresentadas por tais sistemas são:

- objetos PAC são em geral mais complexos do que os representáveis através de conjuntos de registros homogêneos;
- o suporte ao controle de transações não é adequado a PAC, onde existem dois tipos de transação: transações micro, cujo controle se assemelha ao caso convencional, e transações macro, não controláveis através dos mecanismos convencionais, com longos períodos de inconsistência;
- com a importância que assume o conceito de versão, a quantidade de dados que não correspondem ao estado mais atual mas que não devem ser descartados pode crescer muito; a tarefa de detectar e arquivar conjuntos de dados menos utilizados é importante;
- objetos contêm em geral elementos de dados não formatados (textos, descrições de imagens) que não são suportados por tais sistemas;
- não há distinção entre entidades e relacionamentos, que devem ser definidos através de atributos das entidades envolvidas.

O GERPAC — Sistema Gerenciador de Dados para Aplicações PAC — é uma proposta de um SGBD que suporta os conceitos do modelo de dados MER/PAC, que foi proposto para englobar as necessidades de representação de informações PAC. Desta forma, tal Sistema permite a definição e manipulação de tipos de entidades, tipos de relacionamentos, agrupamentos e esquemas, entidades, relacionamentos e seus atributos. Permite ainda o suporte à definição de regras, sendo adequado tanto para utilização na gerência do aspecto micro como do aspecto macro do processo PAC.

3.4.1 Histórico

O primeiro trabalho desenvolvido pelo grupo de Banco de Dados do DCA/FEE/UNICAMP foi a implementação do Núcleo de Armazenamento Associativo CORAS (sigla para "Core System for Associative Storage"). A primeira implementação do CORAS foi desenvolvida na T. H. Darmstadt (RFA) em linguagem de tempo real PEARL, sendo sua especificação derivada de outros sistemas associativos (LEAP, SAM, DATAS) [/WU 83/]. Entre os trabalhos desenvolvidos com o CORAS em Darmstadt incluem-se extensões para manipulações de dados geométricos para aplicações PAC, mostrando a viabilidade da utilização deste sistema nesta área [/BAR82/]. A implementação do grupo da UNICAMP foi desenvolvida em FORTRAN, havendo versões para os Sistemas PDP 11/45, VAX 780 e compatíveis com IBM-PC. A utilização do sistema CORAS-UNICAMP como núcleo de um SGBD foi inicialmente proposta em /WU 84/ para o suporte a uma extensão do MER desenvolvida para aplicações em tempo real.

Da experiência adquirida destas propostas, as principais deficiências do CORAS foram identificadas, levando à definição de um novo núcleo para SGBD, o Sistema de Gerência de Armazenamento Associativo (SIGA). A proposta do SIGA absorve os principais conceitos do CORAS, introduzindo outros com a finalidade de otimizar sua utilização em SGBD não convencionais.

O GERPAC é a primeira proposta de SGBD que utiliza o SIGA como núcleo. O SIGA é independente de modelos de dados, de modo que as características particulares ao modelo, no caso o MER/PAC, são suportadas através de uma interface entre as rotinas do núcleo e as funções GERPAC.

3.4.2 Visão Geral

Os níveis do GERPAC podem ser visualizados através do diagrama da Figura 3.9. Os detalhes do armazenamento interno são tratados pelo SIGA que, por sua vez, incorpora parte do Sistema CORAS-UNICAMP. O SIGA oferece uma interface a nível lógico, onde são definidos objetos, relacionamentos binários (tríades) e atributos associados aos objetos. A interface

GERPAC suporta, a partir destes conceitos básicos, as primitivas MER/PAC. É interessante notar que, sendo o MER/PAC uma extensão em relação ao MER, o GERPAC pode também ser utilizado como um SGBD MER. Convém lembrar, no entanto, que a arquitetura do Sistema foi proposta visando principalmente aplicações PAC, levando-se em conta suas características particulares. Por exemplo, deseja-se que o Sistema possa acessar rapidamente um objeto, pois uma busca exaustiva poderia ser lenta devido à grande quantidade de objetos que se define em aplicações PAC. Para tanto, o SIGA suporta o acesso direto ao objeto simulando um acesso associativo. Levando-se em conta que a quantidade de ocorrências associada a cada objeto é em geral pequena, a estrutura de armazenamento destas informações é relativamente complexa a fim de atender a flexibilidade exigida nestas aplicações.

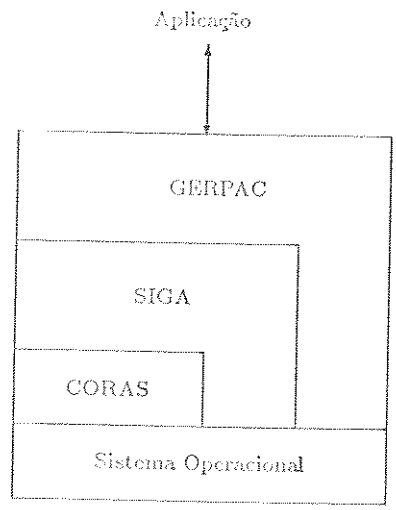

Figura 3.9: Arquitetura GERPAC.

O GERPAC suporta os conceitos estabelecidos pelo MER/PAC. Além da definição das primitivas já citadas, tem-se:

- Tipos de Relacionamentos suportam duas informações semânticas além de seus atributos: o conceito de papéis e o mapeamento entre as entidades envolvidas ($1:1$, $1:n$, $m:n$);
- atributos multivalorados são permitidos;
- Tipos Abstratos de Dados (TAD) são suportados, permitindo a utilização dos tipos de dados complexos necessários a aplicações PAC;

- conjuntos de valores associados a cada atributo são implementados de forma que não haja redundância quando do armazenamento de valores iguais em instâncias distintas do atributo.

O GERPAC suporta as características dinâmicas necessárias a PAC. Desta forma, Tipos de Entidade e de Relacionamento podem ser criados ou removidos sem qualquer impacto adicional sobre o SBD, da mesma forma que Agrupamentos e Esquemas (a não ser os impactos sobre o processo PAC em si). Atributos também podem ser acrescentados ou removidos sem impactos sobre os demais dados já existentes.

Levando-se em conta que os dados na maior parte das aplicações PAC não são diretamente acessados, a interface provida pelo GERPAC é procedural. No entanto, a implementação de uma linguagem gráfica não procedural está prevista como forma de comunicação com o usuário que deseje acessar diretamente o SGBD [/QUE87/].

3.5 COMENTÁRIOS

Neste capítulo foram abordados aspectos particulares à utilização de SBD em PAC. O primeiro ponto apresentado abordou a representação de informações no processo PAC, destacando a necessidade de modelos de dados adequados a este tipo de aplicação. A proposta apresentada neste sentido foi a utilização do MER/PAC [/DEL87/], modelo de dados que utiliza a abordagem entidade-relacionamento.

O outro ponto abordado envolve a questão da implementação de SGBD adequados a PAC. Neste sentido, o Item 3.4 contém uma primeira apresentação de um SGBD que incorpora as características MER/PAC — o GERPAC. Uma descrição mais detalhada sobre o GERPAC será apresentada posteriormente (Capítulo 5), após a apresentação do Sistema utilizado como seu núcleo, o SIGA, que será o assunto do próximo capítulo.

Capítulo 4

O SIGA

4.1 INTRODUÇÃO

O SIGA — Sistema de Gerência de Armazenamento Associativo — é um sistema de software utilizável como núcleo de SGBD, independente de modelos de dados, que evoluiu a partir da proposta de CORAS [/ENC83/,/WU 83/]. No CORAS, todos os elementos do mundo real de interesse são tratados como objetos, seus elementos primitivos, seja qual for a interpretação associada a eles (entidades, atributos ou valores associados aos atributos). Atributos podem ser representados através de uma classe especial de objetos denominada objetos-relação; a associação entre entidades e valores de atributos pode ser representada através de uma associação binária (tríade) que liga um valor a um objeto através de um atributo. Um outro tipo de objetos, os objetos-conjunto, permite definir classes de elementos que também podem ser associados através de objetos-relação. Embora esta representação seja muito adequada do ponto de vista do alto nível de abstração alcançado, sob o ponto de vista da implementação há diversas deficiências que devem ser levadas em conta, entre elas:

acesso: o acesso associativo simulado através de "hashing" não é flexível quanto quantidade de elementos que pode ser manipulada. Nesta abordagem, é difícil prever esta quantidade, pois cada elemento representado — entidades, atributos, valores, conjuntos — ocupa uma entrada nesta tabela "hash";

representação dos dados: todos os elementos do mundo CORAS são representados através de seu nome — uma sequência de caracteres com dimensão pré-definida. Não há forma de tratar diretamente outros tipos de dados ou casos em que se desejasse representar elementos com maior dimensão que a estabelecida pelo Sistema;

velocidade: todas as associações, sejam relacionamentos entre conjuntos ou ocorrências de atributos, são representadas através das tríades. Para agilizar o acesso a estas in-

formações, são armazenadas as três permutações de cada tríade, ou seja, as três possíveis representações para uma associação binária. Esta redundância acarreta uma sobrecarga de controle para a implementação, além da questão da ocupação de espaço;

interpretação: não há distinção entre entidades e valores, devendo qualquer interpretação ser suportada pela aplicação.

Para tornar este núcleo mais eficiente, embora reduzindo o nível de abstração suportado, o SIGA foi proposto. O Sistema será descrito nos próximos itens, englobando seus elementos primitivos, arquitetura interna e uma visão funcional. Finalmente, alguns aspectos particulares à implementação serão abordados.

4.2 ELEMENTOS PRIMITIVOS

Na representação de informações, estas podem em geral ser classificadas em uma das seguintes formas [/COS84/]:

- unidades de informação, com propriedades associadas;
- classes ou tipos, constituindo um conjunto de unidades de informação com características semelhantes; e
- relações, que permitem a conexão entre unidades de informação ou entre classes.

Os elementos primitivos do SIGA são objetos. Estes objetos têm por função representar classes, que internamente descreverão os elementos primitivos do modelo suportado pelo SGBD implementado sobre o SIGA. Desta forma, estes objetos podem representar relações, tipos de registros, tipos de conjuntos, tipos de relacionamentos, tipos de entidades, dependendo do modelo utilizado. Objetos podem ser de três tipos:

SIMPLES: representa uma classe, à qual podem estar associados atributos (ao contrário do que ocorria no CORAS);

CONJUNTO: define agrupamentos de outros objetos, que podem ser simples, conjuntos ou mesmo relações;

RELAÇÃO: permite estabelecer relacionamentos binários entre outros objetos que não sejam do tipo relação (simples/simples, conjunto/conjunto ou simples/conjunto).

Os objetos SIGA são identificados por seu nome. Este nome constitue a chave de acesso a todas as informações sobre o objeto. Para garantir a flexibilidade do Sistema, este nome não é necessariamente único para cada objeto, podendo ser definidos sinônimos que definem igualmente este acesso. No entanto, nomes não podem ser repetidos.

As associações entre objetos SIGA [illegible] na forma (OBJETO RELAÇÃO (OBJETO NÃO RELAÇÃO [illegible] OBJETO NÃO [illegible] 2)) [illegible] pode ser do tipo simples ou [illegible] relacionamentos binários ([illegible]). A representação de outras formas de associações [illegible] suportadas.

Cada objeto pode ser caracterizado por seus atributos. No SIGA estes atributos são armazenados em máscaras associadas ao objeto, que descrevem cada agregação de propriedades. Estas máscaras contém informações tais como nome e tipo de dado associado a cada atributo, definindo as características comuns entre os elementos que constituem as instâncias do objeto (suas ocorrências). Um mesmo atributo pode ser associado a diferentes objetos. Internamente o Sistema [illegible] estes elementos, facilitando o controle de acesso a informações por parte de diferentes usuários (concorrência e segurança).

A caracterização de um objeto pode não ser única. Em alguns casos pode haver mais de um conjunto de atributos associado a um objeto, representando diferentes tipos de informação além da descrição de propriedades, como o momento de criação do objeto, características que instâncias do objeto devem obedecer, descrições sobre os atributos e outras informações. A fim de atender estes casos o SIGA permite a declaração de mais de uma máscara de atributos associada a cada objeto, sendo cada máscara associada a um rótulo que a identifica. A forma de utilização das diferentes máscaras que podem ser associadas ao objeto depende da aplicação, de forma a tornar este Sistema flexível às diversas necessidades.

Ocorrências são instâncias do objeto definidas através de valores associados aos seus atributos. Estas instâncias — que podem ser visualizadas como registros lógicos na base de dados (embora não exista um "registro físico" associado) — constituem a unidade de manipulação dentro do objeto. O conceito de ocorrência pode ser utilizado para mapear tuplas, registros, entidades, relacionamentos, de acordo com o modelo suportado.

Uma das características do SIGA é a capacidade de controlar diversas bases de dados, onde cada uma delas constitue um Sistema de Arquivos. Esta é uma das necessidades básicas em aplicações PAC, onde é necessário haver o controle de informações a nível global e a nível local simultaneamente (controle de macro-aspecto e micro-aspecto). Outra vantagem de distribuir a base de dados em vários Sistemas de Arquivos (em relação à proposta de haver uma única base de dados para todas as aplicações, como ocorre em grande parte dos SGBD existentes) é principalmente a independência que se cria entre as aplicações. Existem duas formas de se definir um novo Sistema de Arquivos:

- iniciando uma nova base de dados;

- definindo uma nova base de dados a partir de outra existente, mantendo a base de dados original inalterada (criar uma nova versão da base de dados).

Os dados operacionais do Sistema estão armazenados na forma de ocorrências, definidas através de ligações entre valores de atributos, enquanto que as informações por ele mantidas sobre os objetos permitem a gerência a um nível mais alto (como uma espécie de Dicionário de Dados local a cada aplicação). O acesso a estes dados operacionais pode ser agilizado pelo SIGA através de índices mantidos automaticamente para atributos definidos pela aplicação.

O suporte ao controle dos Sistemas de Arquivos e Tipos de Dados suportados é provido no SIGA através de outra base de dados com as mesmas características das demais. Nesta base de dados, o Dicionário/Diretório integrado ao Sistema, Sistemas de Arquivos e Tipos de Dados são objetos com atributos pré-definidos.

Tipos de dados não são pré-definidos no SIGA como forma de manter a flexibilidade desejada. Todos os elementos internos são armazenados na forma de caracteres, estando a conversão a outros tipos subordinada ao SGBD. No entanto, o SIGA permite associar informações à definição de um tipo de dado, o que permite verificar por exemplo se o valor do elemento que está sendo armazenado é válido ou quais operações são permitidas. Este tipo de informação pode ser utilizado para especificar Tipos Abstratos de Dados, tais como datas ou elementos geométricos, definidos a partir dos tipos elementares (como inteiro, real e caráter). Especificações de consistência a nível de interrelacionamento dos valores dos atributos de um objeto ou entre objetos distintos devem ser controladas pelo SGBD, assim como o suporte a definições e tratamento de tipos de dados.

Cada Sistema de Arquivos também tem suas características armazenadas no Diretório, tais como versão a que corresponde, seu autor e data de criação. Além destas informações, particulares de cada Sistema, existe um outro tipo de informação que especifica o relacionamento entre as versões existentes, que permite controlar consistência entre versões.

4.3 TRATAMENTO DE OBJETOS

Cada objeto SIGA contém basicamente três tipos de informação:

nome, que permite sua identificação;

atributos, que caracterizam as propriedades da classe por ele representada;

ocorrências, que representam suas instâncias, cada uma delas com uma combinação diferente de valores associados a atributos.

A definição de um objeto passa por duas fases: a criação de seu nome e a caracterização da classe por ele representada. Dois objetos distintos podem ter um mesmo conjunto de propriedades sob o ponto de vista do usuário SIGA (por exemplo, a propriedade "endereço" pode estar associada tanto à classe dos "professores" como à classe dos "alunos"), mas internamente são associados conjuntos de valores distintos para cada atributo.

Os elementos dos conjuntos de valores de cada atributo (ou valores propriamente ditos) são definidos à medida que são definidas as ocorrências da classe. Cada ocorrência é especificada por um conjunto de associações atributo-valor. Internamente, se o valor especificado para um atributo já foi definido através de outra ocorrência, então apenas a associação deste valor com a nova ocorrência é estabelecida; caso contrário, o valor é inicialmente registrado no conjunto de valores correspondente para então ser definida a associação.

A associação de diferentes valores para um atributo em uma mesma instância é permitida, caracterizando um atributo multivalorado. Para permitir a representação de atributos que não devem ser multivalorados (por exemplo, "data de nascimento"), deve ser feita uma distinção a nível de definição dos atributos, que pode posteriormente ser alterada desde que se mantenha a consistência em relação aos dados já armazenados.

As informações sobre objetos podem ser atualizadas sem impacto para o Sistema. Um mesmo objeto pode ter diversos nomes associados através da definição de sinônimos. A remoção de nomes não traz impactos ao Sistema, desde que pelo menos um nome permaneça associado ao objeto. Atributos podem ser acrescidos às propriedades do objeto, nomes e funcionalidade (monovaloração ou multivaloração) de atributos podem ser alterados. A remoção de um atributo acarreta a remoção do conjunto de valores associados e, consequentemente, das associações com suas ocorrências.

4.4 TRATAMENTO DE OCORRÊNCIAS

Ocorrências representam as instâncias dos objetos SIGA. Dependendo da classificação do objeto (simples, conjunto ou relação), o tratamento de suas ocorrências pode diferir. No entanto, um conceito comum às instâncias de objetos é a identificação interna da ocorrência, uma chave interna de acesso às informações sobre as propriedades da instância. O usuário do Sistema não tem acesso a esta chave, que é definida e manipulada internamente, de forma que a única maneira de acessar instâncias é através de cláusulas especificando atributos e valores (na verdade, existe outra forma de acesso que será discutida no Item 4.5.6). Esta chave é independente dos valores de atributos da instância, de forma que alterações destes valores não acarretam inconsistências sobre as informações já existentes (como será visto adiante no caso de instâncias de objetos relação).

O tipo de ocorrência mais elementar é a que está associada a objetos simples. As ocorrências são totalmente definidas através dos valores associados aos atributos, sem de-

pender das associações do objeto com outros.

Para objetos conjuntos, há dois conceitos de instâncias associados: ocorrências como definidas para objetos simples e/ou objetos pertencentes ao conjunto. Um tratamento especial destas duas formas de ocorrências ou mesmo a interligação entre elas (por exemplo, definindo um atributo "nome do elemento" para o objeto conjunto) é de responsabilidade da aplicação; sob o ponto de vista de SIGA são duas informações distintas.

A remoção de uma destas ocorrências implica na remoção de todas as associações atributos-valores de uma determinada instância. Se algum dos valores associados fica, devido a esta operação, sem ocorrências associadas, então este valor é também removido da base de dados. Caso ainda existam associações do valor com outras instâncias, o valor permanece armazenado.

A atualização de um valor de atributo em uma ocorrência significa mudar a associação entre a instância e o elemento do conjunto de valores (o valor). Se não existir o novo valor que foi definido para a instância, este é inicialmente criado para então ser definida a associação.

O conceito de ocorrência de um objeto relação exige uma análise mais detalhada. Inicialmente, é interessante relembrar os conceitos de associação a um nível mais abstrato para depois particularizar para o caso de SIGA. Pode-se dizer que existem basicamente dois tipos de associações entre conjuntos, como mostrado na Figura 4.1: as que definem as propriedades de um elemento (por exemplo, p_1 e 'José') e as que definem relacionamentos entre elementos (por exemplo, p_1 e o_1). Note-se que especificar 'orientação' como uma classe distinta, e não apenas como arcos de associação, é a forma de permitir caracterizar cada elemento de orientação, permitindo associar propriedades a ela.

No SIGA, as associações também são definidas a nível de conjuntos. No primeiro caso, na caracterização de propriedades de elementos, esta associação é definida através dos atributos, sendo no entanto que cada objeto tem associado um conjunto de valores particular a cada atributo. Haveria internamente, neste exemplo, dois conjuntos de valores "nomes de professores" e "nomes de alunos", mesmo que o nome dos atributos fosse o mesmo ("nome") para os dois objetos.

O segundo tipo de associação é estabelecido através de objetos relações e tríades. Desta forma, para se permitir uma associação entre um elemento da classe dos 'professores' a um elemento da classe 'orientação', seria necessário criar uma tríade envolvendo os dois objetos — no exemplo, através de um objeto relação 'orientador'. Uma ocorrência de 'orientador' não tem, neste caso, propriedades; apenas define uma associação entre pares de elementos (por exemplo, $[p_1, o_1]$ e $[p2, o2]$). É interessante observar que, como estas associações são definidas em termos de identificadores internos, são independentes de valores de atributos dos elementos envolvidos, não sendo afetadas por atualizações nos objetos envolvidos (ao contrário do que ocorre em sistemas que suportam tabelas).

A remoção de ocorrências de objetos relações envolve outros cuidados. Algumas vezes é

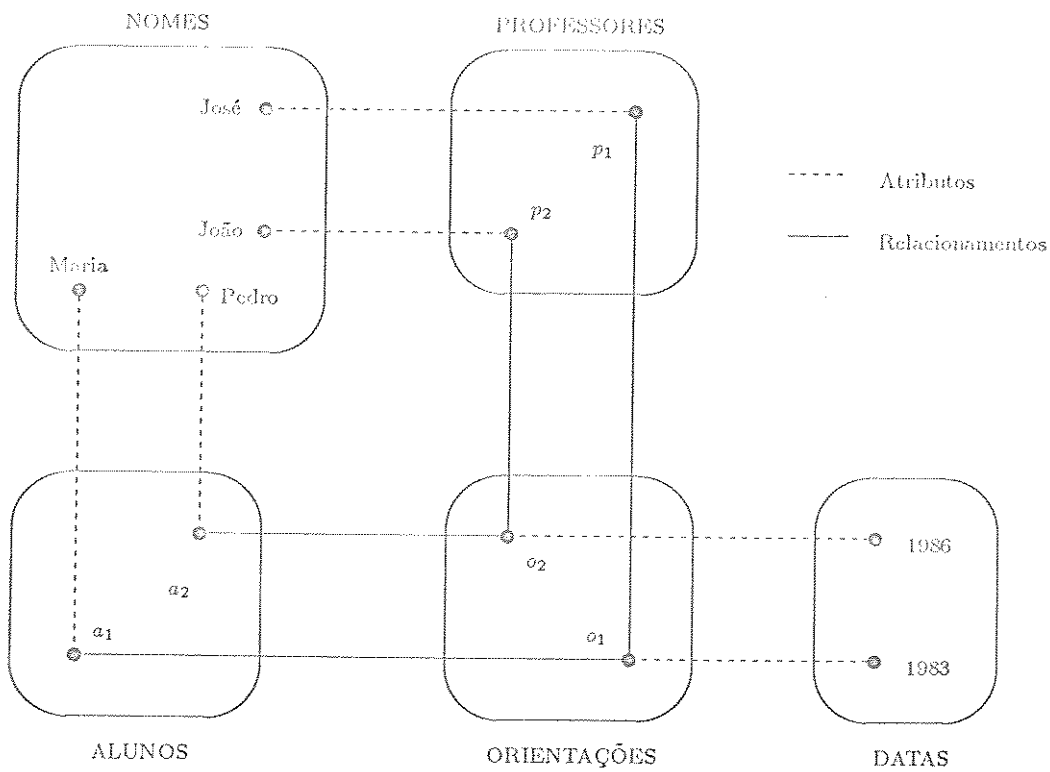

Figura 4.1: Associações entre conjuntos.

interessante que a remoção de um relacionamento provoque a remoção de ocorrências associadas (conceito de dependência), enquanto outras vezes isto não é desejado. Para enquadrar estes dois casos, é definida uma subclassificação do objeto relação, que especifica 'propagação' ou 'não-propagação'. Em uma tríade R(A,B), por exemplo, a remoção de uma ocorrência do objeto A implicará na remoção das ocorrências de B associadas através de R. No entanto, a remoção de uma ocorrência de B acarretará a remoção das ocorrências de A associadas via R se e somente se sua subclassificação especificar a 'propagação'. Esta subclassificação constitui uma propriedade do objeto como um todo e não apenas de seus elementos, como é o caso de atributos.

4.5 ARQUITETURA INTERNA

O SIGA armazena as informações distribuídas entre diversos domínios internos. São eles:

- Domínio de Nomes;
- Domínio de Objetos;
- Domínio de Tríades;
- Domínio de Valores;
- Domínio de Ocorrências;
- Domínio de Índices;
- Domínio de Acesso.

Destes domínios, os três primeiros são derivados diretamente da proposta do CORAS, diferindo apenas no tratamento que é dado às informações. A seguir, cada um destes domínios é descrito, sendo que os detalhes sobre suas estruturas internas serão apresentados no Item 4.7.

4.5.1 Domínio de Nomes

O Domínio de Nomes, a partir do qual os demais domínios são acessados, constitui a interface de acesso entre o usuário e as informações armazenadas sobre o objeto. Este acesso é estabelecido através do nome do objeto. Nomes não devem ser repetidos, mesmo para objetos de diferentes tipos, mas não necessitam ser únicos para um objeto. Nomes alternativos podem ser criados, constituindo assim os diversos sinônimos do objeto.

Além dos sinônimos de cada objeto, o Domínio de Nomes também mantém informação sobre a classe do objeto (se simples, conjunto ou relação) e seu identificador interno. Este identificador (ID) corresponde ao endereço onde se armazenam as informações no Domínio de Objetos referentes a cada objeto individual. No Domínio de Nomes traduz-se o nome do objeto para seu ID, de forma que os demais domínios referenciam o objeto apenas por este identificador e não por seu nome. Sinônimos são dois ou mais nomes distintos com o mesmo ID associado. A troca de nome de um objeto não tem, desta forma, qualquer impacto sobre os dados já armazenados para ele.

4.5.2 Domínio de Tríades

O Domínio de Tríades mantém informações sobre os relacionamentos binários (as tríades) entre os objetos. Da mesma forma que no CORAS, as três permutações de uma tríade são armazenadas de forma que a associação possa ser consultada a partir de qualquer um de seus componentes. Cada objeto que faça parte de pelo menos uma tríade tem associada uma entrada no Domínio de Tríades. Esta entrada associada a cada objeto é acessada através do identificador interno deste domínio (IR).

No Domínio de Tríades, os objetos associados são identificados por seus identificadores internos do Domínio de Objetos (ID). Desta forma, a partir do Domínio de Tríades é possível acessar o Domínio de Objetos, que centraliza a comunicação com os demais domínios.

4.5.3 Domínio de Valores

O Domínio de Valores constitui o depósito dos conjuntos de valores definidos para os atributos dos objetos. Para cada atributo definido no Domínio de Objetos existe uma entrada no Domínio de Valores, acessada através de uma identificação interna (IC) que endereça uma "seção" do domínio que contém o conjunto de todos os valores associados ao atributo. Cada um dos elementos deste conjunto (ou seja, cada valor) tem associado um outro identificador interno deste domínio, o identificador do valor IV.

Valores estão associados a ocorrências. A informação sobre quais valores estão associados a quais ocorrências também está armazenada neste domínio. A cada ocorrência está associado um identificador interno do Domínio de Ocorrências (IO). A cada valor armazenado no Domínio de Valores está associada uma lista de IO, com pelo menos um elemento. Cada elemento desta lista corresponde a uma ocorrência que está associada a este valor.

É importante notar que o Domínio de Valores não suporta o conceito matemático de domínios, pois a dois atributos distintos são associadas listas de valores distintas, mesmo que os atributos tenham o mesmo tipo de dado associado. Entretanto, para cada atributo não há repetição de valores, mesmo que diversas ocorrências distintas atribuam um mesmo valor

para o atributo, evitando desta forma a redundância que existe em sistemas que manipulam registros ou tabelas.

A representação interna dos valores armazenados está na forma de caracteres. Não há limitações para a dimensão de cada valor, nem a obrigatoriedade de dimensões pré fixadas. Caso seja interessante fixar a dimensão que a representação de um atributo vai assumir em uma aplicação suportada pelo SIGA, esta tarefa deverá ser executada pela própria aplicação através do tratamento de regras, por exemplo.

4.5.4 Domínio de Ocorrências

Ocorrências são instâncias associadas a objetos. A cada ocorrência está associada uma entrada no Domínio de Ocorrências, endereçada por um identificador interno (IO). Uma ocorrência de um objeto é definida através da associação de valores aos atributos deste objeto.

No Domínio de Ocorrências, estes valores são representados pelos identificadores internos do Domínio de Valores IV. Assim, a informação completa sobre uma ocorrência é composta por sua identificação (IO) associada a uma lista de pares *[identificação do atributo, IV do valor associado]*, onde cada elemento da lista define uma associação atributo-valor.

É interessante notar que não há restrições quanto à unicidade da identificação do atributo nesta lista, de forma que atributos multivalorados não representam nenhum impacto sobre a estrutura de armazenamento do Sistema (dois ou mais elementos da lista com a mesma identificação de atributo). Por outro lado, atributos que não tenham valor associado em uma ocorrência (conceito de valor nulo) não ocupam espaço de armazenamento; simplesmente não há um elemento nesta lista associado a este atributo.

4.5.5 Domínio de Índices

A comunicação entre o Domínio de Objetos e o Domínio de Valores pode ser realizada diretamente, através do identificador IC, associando cada atributo ao seu conjunto de valores. Uma vez acessada a entrada ao conjunto de valores, a busca de um valor específico é realizada de maneira sequencial, o que pode representar uma sobrecarga de processamento se o atributo correspondente tem muitos valores associados e é frequentemente utilizado como item de busca.

O Domínio de Índices foi proposto para permitir uma otimização no acesso para estes casos. Se um atributo é definido como indexado, o identificador associado a ele no Domínio de Objetos indica não uma entrada para o conjunto de valores no Domínio de Valores, como no caso normal, mas sim uma entrada a uma estrutura de índice definida no Domínio de Índices. Esta entrada é endereçada por um identificador interno para este domínio (II). A partir desta entrada, o valor correspondente ao desejado é diretamente acessado no Domínio

de Valores através do identificador do valor IV.

A definição do tipo de estrutura índice utilizada neste domínio (e.g., árvore binária, árvore-B) corresponde a um detalhe de implementação não abordado nesta apresentação.

4.5.6 Domínio de Acesso

A informação sobre a definição de cada instância de um objeto está definida no Domínio de Ocorrências. A cada objeto está associado um conjunto de instâncias, que constituem suas ocorrências. Desta forma, o Domínio de Ocorrências contém a definição de todas as ocorrências de todos os objetos definidos.

O Domínio de Acesso define quais ocorrências, entre todas as definidas no Domínio de Ocorrências, estão associadas a um objeto específico. Para cada objeto definido no Domínio de Objetos está associada uma entrada para o Domínio de Acesso endereçada por seu identificador interno IA. A cada IA está associado um conjunto de endereços internos IO para o Domínio de Ocorrências correspondente ao conjunto de ocorrências do objeto.

A especificação deste domínio permite uma forma alternativa de acesso às ocorrências (não-associativo), pois a partir dele é possível acessá-las sem conhecimento prévio de seus valores. Entretanto, a seqüência definida pelo acesso através deste domínio não necessita seguir necessariamente nenhum tipo de ordenação (como momento de armazenamento ou índices), devendo-se buscar a otimização das formas de acesso suportadas internamente.

4.5.7 Domínio de Objetos

O Domínio de Objetos centraliza a comunicação com todos os demais domínios SIGA. Além desta função de comunicação, as informações sobre as estruturas de atributos de cada objeto e sobre a relação entre objetos do tipo conjunto e seus elementos também estão armazenadas neste Domínio.

A comunicação entre o Domínio de Objetos e o Domínio de Nomes (e, consequentemente, com os usuários do Sistema) é estabelecida através de uma lista de sinônimos associada ao identificador interno do objeto (ID). Assim, se for necessário por exemplo obter qual a classe (simples, conjunto, relação) correspondente a um objeto cujo ID é conhecido, esta informação é buscada no Domínio de Nomes a partir de um dos sinônimos desta lista.

A comunicação com o Domínio de Tríades é estabelecida através de uma lista de identificadores internos IR associada ao objeto. Esta lista contém cada entrada IR para o Domínio de Tríades definida para objetos que fazem parte de pelo menos uma tríade (como descrito no Item 4.5.2).

Existem duas listas associadas a cada identificador ID para representar informações sobre conjuntos. Uma delas mantém as informações sobre quais objetos conjuntos contêm o objeto

em questão. A outra descreve, para os objetos conjuntos, quais são seus elementos. Nos dois casos os objetos são representados através de seus identificadores ID.

Outra informação diz respeito às estruturas de atributos de cada objeto. Esta estrutura foi definida sobre a antiga lista de caracteres associada a cada objeto definida no CORAS [/WU 83/]. Por questões de flexibilidade, foi definido que um objeto pode ter mais de uma estrutura de atributos associada, pois pode haver mais de um tipo de informação associada a cada objeto. Por exemplo, um tipo de informação poderia corresponder às propriedades do objeto, outro às regras associadas ao objeto (formação, consistência) ou outro tipo de informação semântica. Desta forma, uma lista de caracteres contém as diversas estruturas de atributos definidas para um mesmo objeto, sendo que cada estrutura é identificada por um rótulo. O controle destes rótulos (qual o tipo de informação associada a cada estrutura de atributos e como esta informação é utilizada) é exercido pela aplicação suportada por SIGA.

Para cada estrutura está associado um identificador IA correspondente à comunicação com o Domínio de Acesso, que por sua vez permite a comunicação com o Domínio de Ocorrências (Item 4.5.6), permitindo assim agilizar buscas exaustivas (não associativas). Por sua vez, a comunicação com o Domínio de Valores e o Domínio de Índices é definida por indicadores associados a cada um dos atributos de cada estrutura; se o atributo é indexado, este indicador será um II (Item 4.5.5); caso contrário, o indicador será um IC (Item 4.5.3).

Uma visão geral dos Domínios SIGA é apresentada na Figura 4.2.

4.6 VISÃO FUNCIONAL

O SIGA provê funções para a definição e manipulação de suas primitivas. O usuário SIGA não necessita ter acesso à estrutura interna de armazenamento, à definição do protocolo de comunicação entre a memória principal e secundária ou à definição do formato de arquivos em disco. Para atingir estes objetivos, são definidos internamente três níveis de funções SIGA:

Nível Terciário: contém rotinas disponíveis ao usuário SIGA (a aplicação), englobando as rotinas de definição de bases de dados ("sistemas de arquivos") e rotinas de definição e manipulação dos elementos primitivos;

Nível Secundário: é o nível de visão lógica dos dados, onde são definidas e manipuladas as estruturas internas de dados referentes aos diversos domínios;

Nível Primário: contém as rotinas que manipulam o nível físico. Estas rotinas definem a forma de acesso aos dados em disco, controlando também o Sistema de Paginação (manutenção em memória principal apenas de dados relevantes, mantendo os demais em memória secundária).

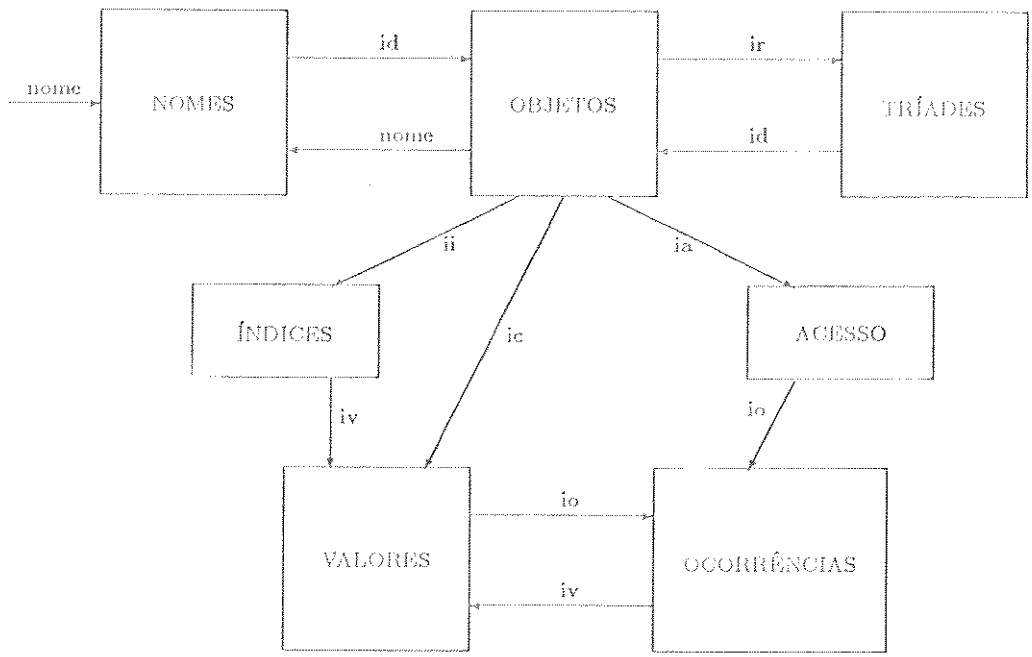

Figura 4.2: Domínios SIGA.

4.6.1 Nível Primário

As informações referentes à comunicação entre o dispositivo de armazenamento e a memória principal podem ser divididas em duas categorias:

Temporárias: que só têm relevância enquanto o Sistema está ativo;

Permanentes: que devem perdurar entre ativações distintas do Sistema.

As informações permanentes incluem a descrição da alocação do espaço em disco referente aos arquivos armazenados de cada Sistema de Arquivos. Como estas informações devem ser mantidas entre sessões de trabalho distintas, constituem um arquivo de dados de apoio ao Sistema de Arquivos de dados operacionais (dados referentes aos domínios internos de SIGA) que também deve ser armazenado em disco.

As informações temporárias dizem respeito ao sistema interno de paginação, ou seja, constituem uma descrição sobre quais registros físicos estão ocupando a memória principal e se estes registros foram ou não modificados enquanto permaneceram na memória.

Existem basicamente quatro grupos de rotinas do nível primário:

Rotinas de Acesso a Disco: definem as funções de criação e utilização (abrir/fechar) dos arquivos de dados em disco. Definem o acesso direto a registros tanto para escrita como para leitura. Constituem a interface entre o Sistema Operacional e SIGA;

Rotinas de Gerência das Informações: manipulam as informações temporárias e permanentes do nível primário. Através destas rotinas, tornam-se transparentes ao Sistema de Paginação a estrutura interna destas informações;

Rotinas do Sistema de Paginação: acessam as rotinas do Sistema de Gerência de Informações e as rotinas de Acesso a Disco (escrita e leitura, acesso direto), implementando os algoritmos de paginação para os arquivos de dados;

Rotinas de Interface com o Nível Secundário: as rotinas do nível primário tratam os arquivos de dados do sistema através de códigos internos. Como não é interessante que haja a utilização deste tipo de informação interna nos demais níveis, são definidas interfaces que as isolam. Estas rotinas englobam todas as funções relativas ao Nível Primário.

4.6.2 Nível Secundário

A cada um dos Domínios SIGA está associada uma estrutura de dados independente. Os arquivos de dados associados a estas estruturas são denominados Arquivos Internos. Desta forma, existe um Arquivo Interno de Nomes, um Arquivo Interno de Objetos e assim por diante. As funções responsáveis pela manipulação de cada um destes Arquivos Internos constituem as rotinas associadas ao Nível Secundário das funções SIGA.

Em seu nível mais alto, estas rotinas constituem a interface para as funções do Nível Terciário, que é o responsável pela interconexão das informações distribuídas pelos Arquivos Internos.

Internamente, as funções desempenhadas por estas rotinas dependem da estrutura de dados e das opções de implementação associadas a cada Arquivo Interno. Por exemplo, se as informações em um Arquivo Interno estão armazenadas na forma de listas, então deverão ser providas funções para a manipulação de listas e seus elementos.

As rotinas deste Nível utilizam rotinas do Nível Primário, de forma que alterações processadas no Nível Secundário (correspondente a uma visão lógica dos Arquivos Internos) têm reflexo sobre o nível físico, mantendo atualizadas informações tais como quantidade de páginas utilizadas pelo Arquivo Interno e o espaço disponível em cada página.

4.6.3 Nível Terciário

As rotinas deste nível constituem a interface externa do SIGA. As funções desempenhadas por estas rotinas são basicamente:

- tornar transparente ao usuário a utilização das funções de níveis inferiores;
- realizar a comunicação entre os diversos Domínios SIGA;
- uniformizar o tratamento de códigos de erro.

O primeiro objetivo visa isolar o usuário SIGA da arquitetura interna do Sistema, de forma que para utilizá-lo não é necessário conhecer, por exemplo, quais arquivos serão acessados durante uma operação.

A comunicação entre domínios é realizada através de chamadas às rotinas de interface do Nível Secundário, de forma que a troca de informações entre domínios ocorre apenas neste nível.

O grupo de rotinas associado a um domínio apresenta, em seu nível mais alto, códigos de erro que são independentes dos códigos definidos para os demais domínios. A tradução destes códigos internos em erros do Sistema também é tarefa das rotinas deste nível.

A especificação das rotinas deste Nível pode ser encontrada em /RICS86b/.

4.7 ASPECTOS DE IMPLEMENTAÇÃO

Serão apresentados dois aspectos referentes à implementação dos conceitos até aqui expostos:

- estruturas de dados associadas aos Arquivos Internos; e
- estratégia de alocação e paginação.

4.7.1 Estruturas de dados

De uma forma geral, as estruturas discutidas a seguir buscam atingir um objetivo comum, que é a flexibilidade. Para tanto, estas estruturas são definidas a partir de células, estruturas mais elementares associadas a cada Arquivo Interno. Sob o ponto de vista físico, células estão agrupadas em conjuntos de quantidades pré-fixadas que constituem as páginas dos arquivos. Páginas constituem a unidade de acesso a disco, ou seja, estão associadas aos registros físicos.

Arquivo Interno de Nomes

O Arquivo Interno de Nomes apresenta uma estrutura de dados simples. A opção de implementação para permitir o acesso associativo foi a de utilizar tabela "hash".

Colisões são tratadas através de definições de áreas de colisão, que podem ser de dois tipos: contidas na própria página que deveria conter a entrada ou definidas por páginas de colisão.

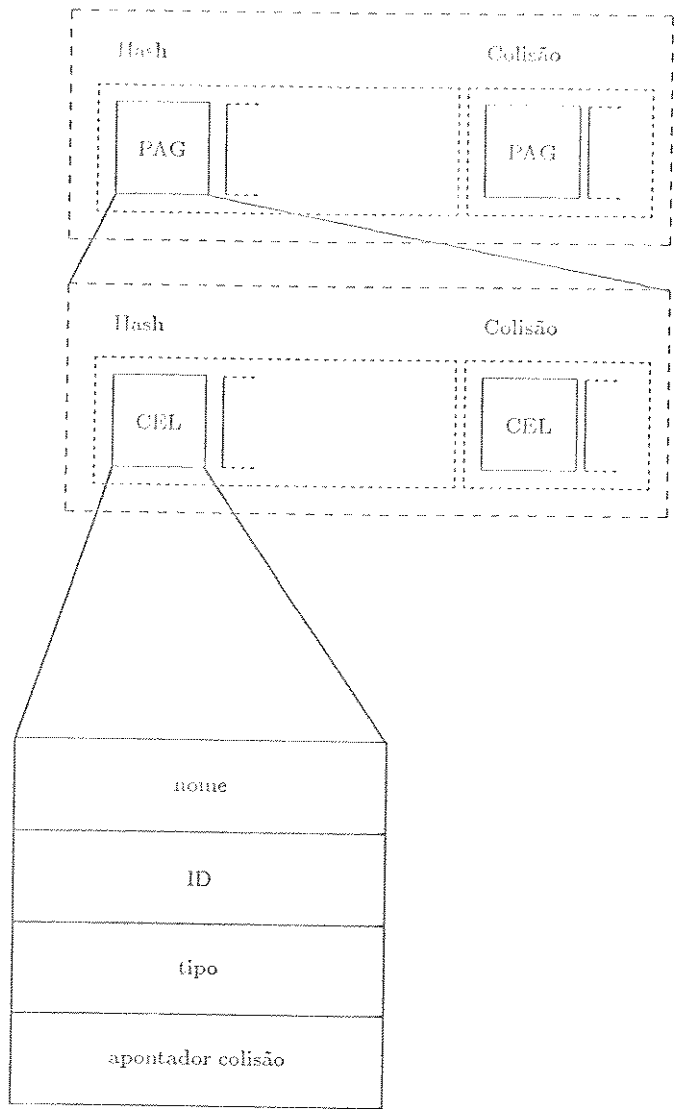

Figura 4.3: Estrutura do Arquivo Interno de Nomes

Em uma primeira tentativa, procura-se alocar as entradas que colidem na mesma página em que seriam alocadas. Desta forma, uma página é dividida em duas sub-áreas: de acesso direto e de colisão. A área de colisão é organizada na forma de lista de células. Eventualmente pode não haver células disponíveis nesta sub-área; nestes casos, a entrada é armazenada em páginas que têm a mesma estrutura que as páginas de acesso direto mas que diferem exatamente pela forma de acesso: nas páginas de acesso direto, a função "hashing" define página e célula que deve ser ocupada, enquanto nestas páginas apenas a célula é definida por "hashing" (a mesma posição que ocuparia na página de acesso direto correspondente), enquanto que a página é a que se encontra disponível no momento (Figura 4.3).

Arquivo Interno de Objetos

O Arquivo Interno de Objetos contém as informações sobre a estrutura dos objetos e apontadores às suas instâncias. Sua estrutura interna é baseada na estrutura do Arquivo de Entidades do Sistema CORAS-UNICAMP [/WU 83/], apresentando em acréscimo facilidades para a definição e manipulação de atributos.

Cada objeto está representado no Arquivo Interno de Objetos por um cabeçalho, que por sua vez permite o acesso às demais informações armazenadas no arquivo. A posição deste cabeçalho no Arquivo — definida não por seu endereço físico mas através de apontadores à página em que se encontra e à célula que ocupa dentro desta página — constitui a implementação do ID (identificador interno) definido no Item 4.5.1 e que é referenciado pelos outros arquivos internos do Sistema.

As informações sobre cada objeto são armazenadas em células que, ligadas, constituem as listas internas do Arquivo Interno de Objetos (Figura 4.4). Estas listas internas são cinco:

lista de nomes: contém todos os nomes definidos para o objeto (o nome inicial e os sinônimos definidos posteriormente);

lista de atributos: contém as máscaras de atributos definidas pela aplicação, sendo que a cada atributo está associado um apontador ao Arquivo de Valores ou de Índices. Mantém também a informação sobre a quantidade de ocorrências associada a cada máscara de atributos. Sua estrutura interna é a seguinte:

- Quantidade de máscaras de atributos definidas para o objeto;
- Entrada para cada máscara, composta por:
  - Rótulo da máscara, que permite identificar cada definição;
  - Dimensão da definição da máscara dentro da lista de atributos;
  - Apontador para o Arquivo Interno de Acesso;

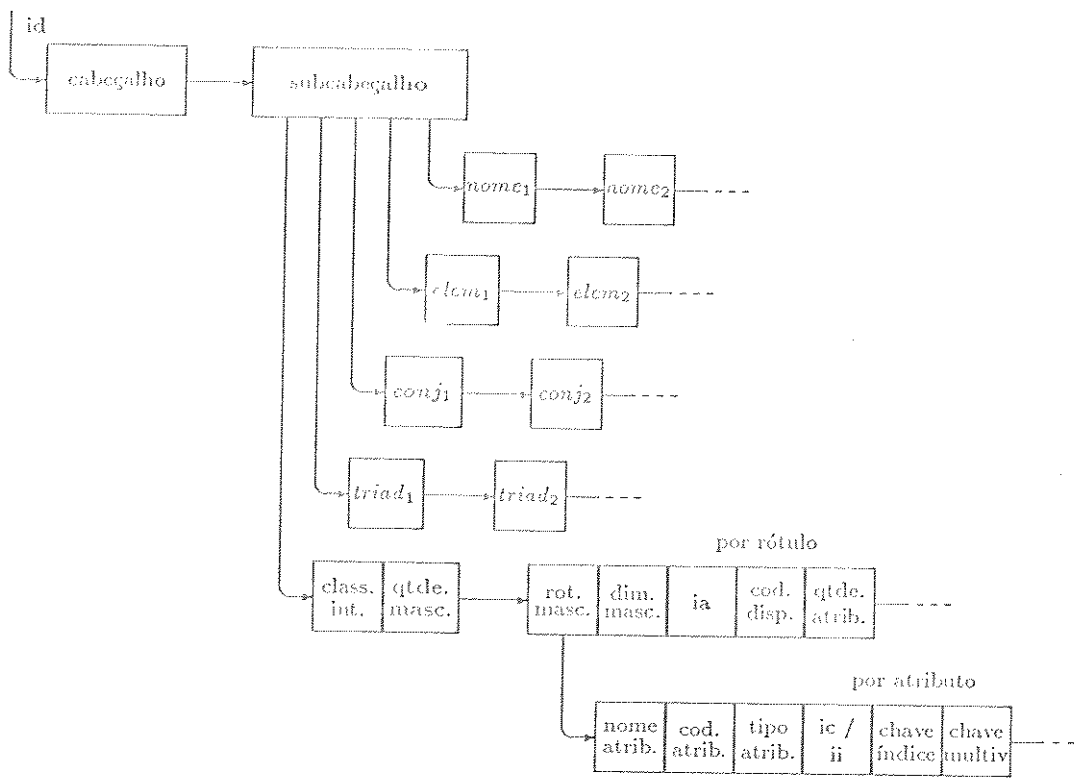

Figura 4.4: Listas Internas do Arquivo Interno de Objetos.

- Definição de cada máscara, composta por:
  - Quantidade de atributos;
  - Quantidade de ocorrências;
  - Descrição de cada atributo, composta por:
    - Nome do atributo;
    - Código interno associado ao atributo;
    - Tipo de dado associado ao atributo;
    - Apontador interno ao Arquivo de Valores (IC) ou Índices (II), conforme o atributo seja ou não indexado;
    - Indicador: atributo é indexado ou não;
    - Indicador: atributo é multivalorado ou não;

lista de elementos de conjuntos: armazena os ID dos objetos que pertencem ao objeto conjunto que contém esta lista interna;

lista de conjuntos associados ao objeto: armazena os ID dos objetos tipo conjunto aos quais o objeto que contém esta lista pertence;

lista de entrada ao arquivo de tríades: armazena apontadores semelhantes aos ID que são definidos para o Arquivo de Tríades — os IR, definindo a posição naquele arquivo onde estão armazenadas as informações sobre as tríades nas quais o objeto toma parte.

O cabeçalho contém informações sobre a quantidade de elementos em cada uma destas listas. A flexibilidade na definição do tamanho de uma lista é garantida através de uma estrutura intermediária entre o cabeçalho e as listas internas — os subcabeçalhos — que além dos apontadores ao início de cada lista na página permitem armazenar apontadores a uma possível continuação da lista em outra página (Figura 4.5).

Arquivo Interno de Tríades

O Arquivo Interno de Tríades mantém informações sobre as tríades envolvendo os objetos definidos no Arquivo de Objetos. Uma tríade é uma tripla ordenada de objetos (ou seja, a ordem da associação é relevante) dos quais um deles — e somente um — deve ser do tipo relação.

Por exemplo, seja o caso em que há três objetos, O_1, O_2 e O_3, onde um deles é do tipo relação. Pode ser definida uma associação $O_1 - O_2 - O_3$, que é equivalente às associações $O_2 - O_3 - O_1$ e $O_3 - O_1 - O_2$, mas é diferente de outra associação $O_1 - O_3 - O_2$ (pois a ordem dos elementos nesta última associação foi invertida). Cada uma destas associações

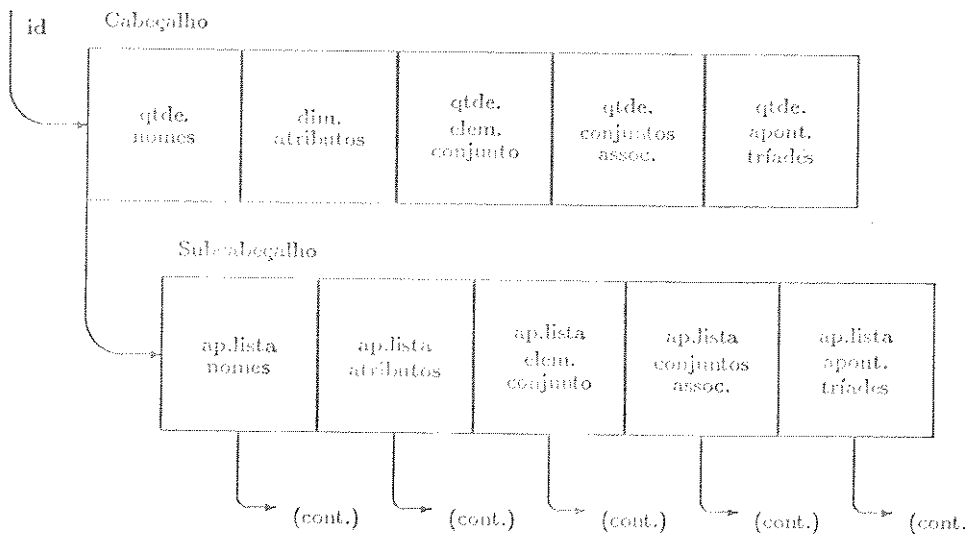

Figura 4.5: Estrutura cabeçalho - subcabeçalho.

equivalentes constitui uma permutação da tríade. Para agilizar o acesso às informações deste Arquivo, as três permutações de uma tríade são armazenadas.

As informações são armazenadas internamente através de três listas que estão em diferentes níveis (Figura 4.6). O primeiro nível indica o objeto que define a entrada àquelas permutações que o contêm; é o chamado Anel Principal ou Anel A. Por exemplo, ao se armazenar duas permutações iniciadas pelo mesmo objeto $O_1 - O_2 - O_3$ e $O_1 - O_4 - O_5$, este Anel conteria o objeto O_1 (representado por seu ID) e um indicador para a lista do nível inferior. Neste segundo nível está o Sub-anel B, que contém o segundo elemento das permutações iniciadas pelo mesmo objeto — no exemplo, uma lista contendo os objetos O_2 e O_4. Finalmente no último nível, definido como Sub-anel C, estão armazenados os últimos elementos de cada permutação — no exemplo, os objetos O_3 (indicado a partir do Sub-anel B pelo objeto O_2) e O_5 (indicado pelo objeto O_4).

O fato de se armazenar as três permutações de uma tríade permite agilizar a recuperação destas informações visto que a cada objeto que está associado através de tríades corresponde uma entrada independente neste Arquivo, entrada esta que está armazenada em uma das listas do Arquivo de Objetos (lista de IR). Este procedimento permite atingir um tempo de busca quase simétrico (o tempo de acesso a informações desconhecidas a partir de qualquer informação conhecida é praticamente o mesmo).

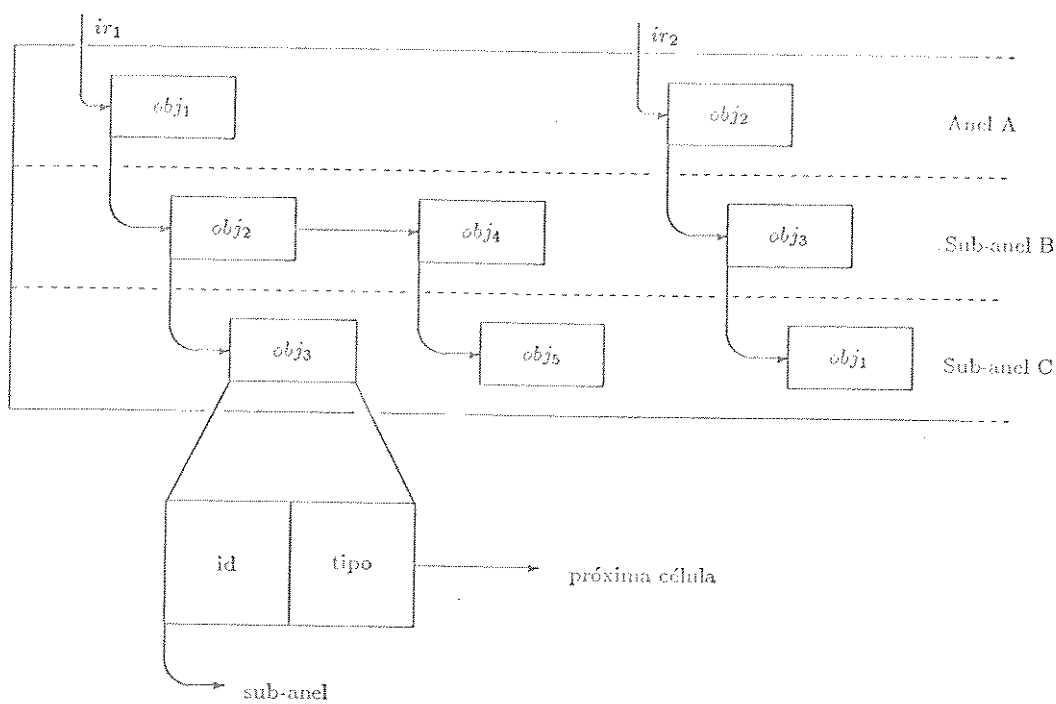

Figura 4.6: Estruturas do Arquivo Interno de Tríades.

Arquivo Interno de Valores

As estruturas de dados dos Arquivos Internos de Valores e de Ocorrências estão intimamente interligadas: associadas a cada valor de um atributo estão referências a ocorrências, enquanto que cada ocorrência é composta por referências a valores.

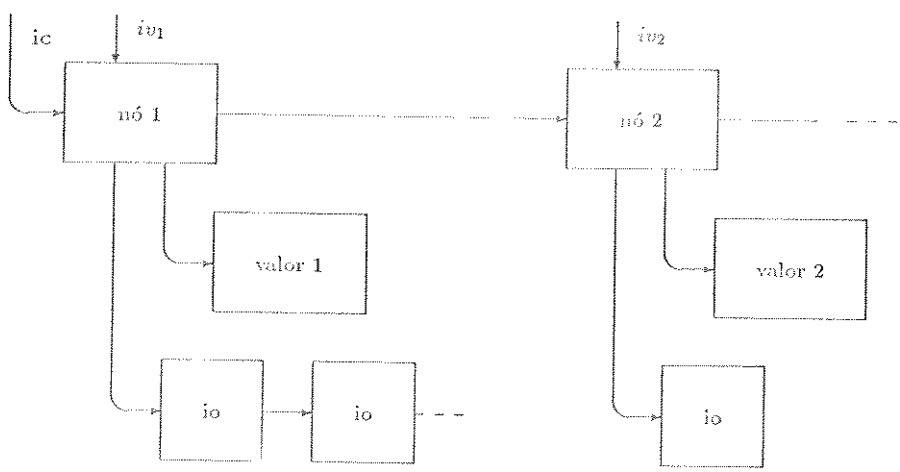

Figura 4.7: Estruturas do Arquivo Interno de Valores.

A cada atributo definido no Arquivo Interno de Objetos está associada uma entrada no Arquivo Interno de Valores. Esta entrada (que constitui a implementação do IC definido no Item 4.5.3) encabeça uma lista de "nós", sendo que a cada nó (cujo endereço interno constitui a implementação do indicador IV, também definido no Item 4.5.3) estão subordinadas duas informações: um valor (representado através de listas de caracteres) e uma lista de apontadores ao Arquivo Interno de Ocorrências, onde estão definidas as ocorrências que contêm aquele valor. Um exemplo desta estrutura é apresentado na Figura 4.7.

Arquivo Interno de Ocorrências

Uma ocorrência é definida no Arquivo Interno de Ocorrências através de uma lista que contém indicadores ao Arquivo Interno de Valores (IV). Associada a cada indicador está a informação sobre a qual atributo (representado por seu código interno) se refere o valor indicado (Figura 4.8).

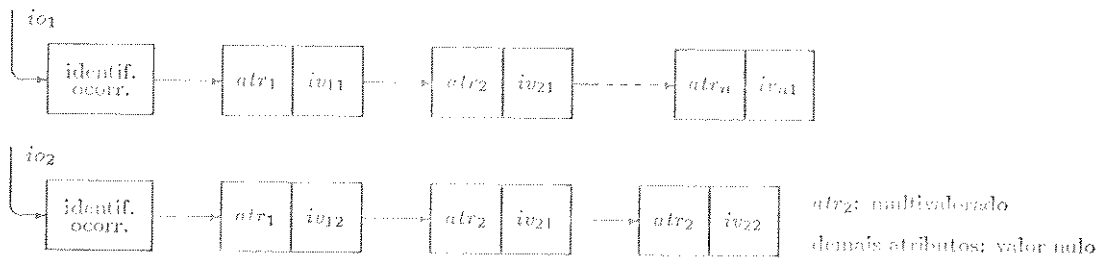

Figura 4.8: Estruturas do Arquivo Interno de Ocorrências.

Como não há restrições sobre a quantidade de vezes que um atributo aparece em uma ocorrência, não há limitações com relação a atributos multivalorados (mais de um valor para um atributo em uma instância). Também não há a obrigatoriedade de que todos os atributos de um objeto estejam definidos em uma ocorrência, de forma que o tratamento de valores nulos é natural.

Arquivo Interno de Acesso

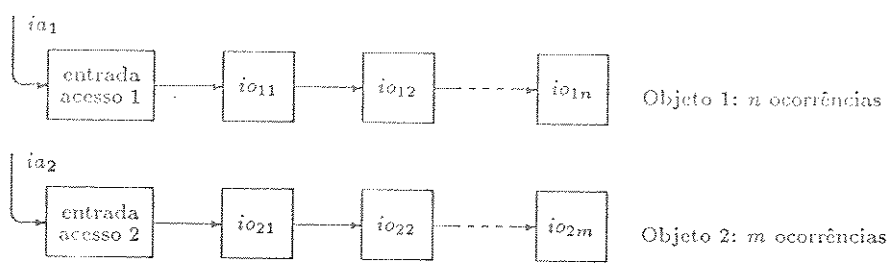

Figura 4.9: Estruturas do Arquivo Interno de Acesso.

A estrutura associada ao Arquivo Interno de Acesso (Figura 4.9) corresponde a uma lista simples encabeçada pelo indicador interno IA (definido no Item 4.5.6). Os elementos da lista contêm os apontadores para o Arquivo Interno de Ocorrências correspondentes aos indicadores internos IO definidos no Item 4.5.4.

Esta estrutura, além de permitir o acesso exaustivo às ocorrências de um objeto, facilita o controle de alocação de páginas para o Arquivo Interno de Ocorrências.

Arquivo Interno de Índices

O Arquivo Interno de Índices tem por função agilizar o acesso ao Arquivo Interno de Valores, não sendo portanto um elemento essencial no conjunto dos Domínios SIGA. Sua estrutura interna não foi definida nesta proposta.

4.7.2 Estratégias de Alocação e Paginação

A alocação da memória principal é pré-estabelecida em SIGA, sendo que existem áreas reservadas a cada um dos arquivos internos. A quantidade de páginas que cada área pode conter é definida na configuração do Sistema, assim como a dimensão de cada página. A designação da área em memória principal que pode conter uma página de um arquivo interno é um "frame"; o conjunto de "frames" associado a cada arquivo interno constitui um "pool".

A cada arquivo interno está associado um arquivo armazenado em disco. A alocação do espaço em disco era limitada no Sistema CORAS-UNICAMP, sendo definida no momento da iniciação do sistema de arquivos, podendo ser expandida durante sua operação. Havia nesse Sistema um limite superior de quantidade de páginas que poderiam ser manipuladas. No SIGA, tal limitação de gerência não existe, sendo que também não é necessário especificar a dimensão inicial dos arquivos internos. Esta alocação é dinâmica, ocorrendo à medida que o Sistema o exige.

Com relação ao Sistema de ocupação das páginas, as estratégias são diferenciadas de acordo com os diversos arquivos internos. Por exemplo, no Arquivo de Nomes a ocupação é definida através de "hashing", sendo que a alocação das páginas de colisão é definida sequencialmente. Já para o Arquivo de Objetos, a estratégia é buscar a primeira página que tenha espaço suficiente para a execução da operação especificada. Para o Arquivo de Tríades a estratégia é semelhante, com a diferença de que se busca reservar uma página para cada objeto que componha uma tríade, enquanto for possível. Para os demais Arquivos, buscou-se uma estratégia que permitisse uma modularização física dos dados, através da reserva de páginas diferentes para informações associadas a objetos distintos. Assim, uma página do Arquivo de Ocorrências contém informações sobre as ocorrências de um único objeto, e no Arquivo de Valores cada página está associada basicamente a um atributo distinto.

O Sistema de Paginação, que permite a troca de registros entre o disco e a memória principal, é controlado por rotinas do Nível Primário do SIGA. A estratégia adotada busca inicialmente ocupar "frames" livres quando se requisita a leitura de uma página para a memória principal. Caso não haja "frame" disponível, desocupar-se-á aquele que contenha a página que tenha sofrido menos ativações (operações de armazenamento, remoção ou leitura) durante o período de ocupação no "frame". Se a página em questão não sofreu alterações, é simplesmente descartada; caso contrário, o arquivo armazenado é atualizado.

4.8 DICIONÁRIO/DIRETÓRIO SIGA

Uma das características do SIGA é permitir a manipulação simultânea de mais de um Sistema de Arquivos (base de dados), forma adotada para permitir a integração entre dados de distintas aplicações. Um dos Sistemas de Arquivos, que permanecerá ativo durante toda uma sessão de trabalho SIGA (período entre sua ativação e desativação), corresponde ao Dicionário/Diretório SIGA (DDS).

As funções do DDS são basicamente duas:

- controle de Sistemas de Arquivos existentes e suas versões, e
- registro de todos os tipos de dados disponíveis aos usuários SIGA.

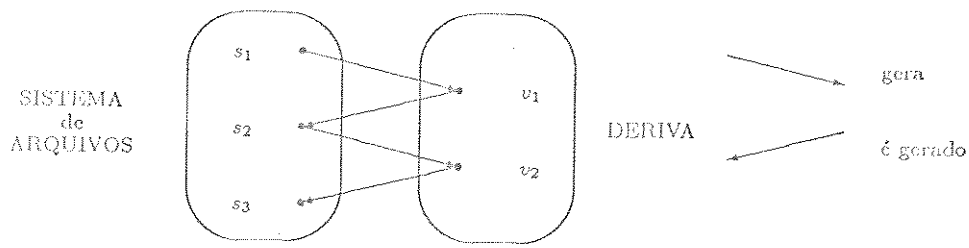

Figura 4.10: Representação de Sistemas de Arquivos e Versões.

Estas tarefas são suportadas através de funções SIGA e as informações são representadas em termos de seus elementos primitivos.

O DDS reúne sob seu controle um conjunto de Sistemas de Arquivos. A cada um destes Sistemas podem estar associadas diversas versões, cada uma constituindo uma base de dados distinta. A geração de uma versão a partir de outra é uma operação que deve estar registrada, pois algumas vezes uma modificação em alguma versão pode exigir, segundo o controle da aplicação, medidas em relação às versões subordinadas, sendo importante manter esta informação (Figura 4.10).

Em termos de primitivas SIGA, cada Sistema de Arquivos é representado por um objeto simples, agrupados sob um objeto conjunto que congrega todos os Sistemas de Arquivos definidos para o DDS. Dois objetos relação são utilizados para controlar as associações existentes entre duas versões e a operação que gerou a versão subordinada; o primeiro objeto relação especifica qual base de dados desempenha o papel de 'geradora' na operação, e o segundo

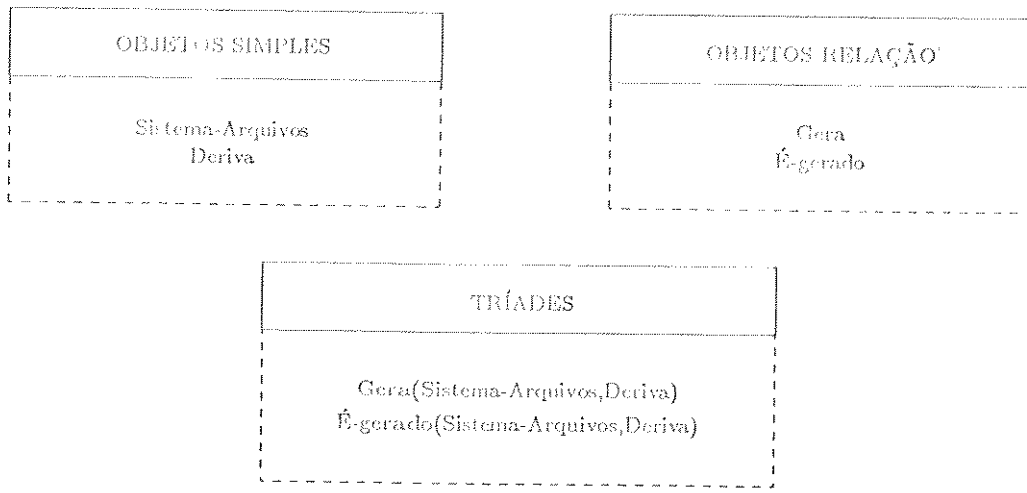

Figura 4.11: Representação do Diretório em termos SIGA.

especifica o papel de base de dados 'gerada' (Figura 4.11). Note-se que como uma tríade não pode ser especificada sobre um único objeto (auto-associação), foi necessário acrescentar um outro objeto simples que indica apenas as ocorrências de operações que levam à geração de novas versões.

Os atributos associados a estes objetos definem as propriedades de cada base de dados, como 'data de criação', 'data de última atualização', 'código de acesso', 'espaço ocupado', 'consistência' etc.

Tipos de Dados constituem uma classe, representada por um objeto simples. Os atributos dos objetos especificam a forma de conversão do tipo básico SIGA (representação em caracteres) para o tipo desejado, o que pode ser realizado através de um subsistema integrado a SIGA. Outros atributos podem especificar, por exemplo, operações permitidas sobre cada tipo de dado.

É necessário, como no caso de versões de Sistemas de Arquivos, especificar as associações entre tipos de dados como forma de documentar dependências entre suas definições. Por exemplo, um tipo de dado 'Ponto 2D' poderia ser especificado a partir do tipo de dado 'Real', que por sua vez pode ter sido especificado a partir do tipo básico 'Caráter'. Se houver modificações na especificação do tipo de dado 'Real', pode ser necessário modificar também os tipos de dados subordinados, entre os quais estaria 'Ponto 2D'.

4.9 COMENTÁRIOS

O Sistema de Gerência de Armazenamento Associativo -- SIGA -- constitui uma alternativa a outros sistemas associativos que manipulam objetos e conjuntos, como é o caso de CORAS. Apesar do alto nível de abstração apresentado por tais sistemas, o desempenho em geral deixa a desejar. Particularmente em relação ao CORAS, pode-se citar:

acesso: o SIGA distingue o acesso a objetos do acesso a valores. No CORAS, um valor deveria ser definido como um objeto para permitir sua representação e conseqüentemente seria acessado da mesma forma que os objetos, através de "hashing". No SIGA, valores independem de "hashing". Em termos de implementação, a quantidade de valores que pode ser acessada não tem limitação lógica, como ocorre com algoritmos de "hashing";

representação interna de valores: a vantagem com relação à implementação do CORAS se encontra no fato de que não há limitação lógica para a dimensão da lista que contém a representação de um valor (no CORAS, sendo cada valor representado como um objeto havia a limitação correspondente à dimensão de um nome, que constitui a entrada para a Tabela Hash), sendo que em comum se manteve a representação na forma de caracteres. A restrição permanece para nomes de objetos;

velocidade: o fato de não se utilizar tríades para representar associações do tipo atributo (como era necessário em CORAS para representar a associação entre um objeto-instância e um objeto-valor através de um objeto-atributo) alivia o volume das informações relacionadas a este domínio. As associações armazenadas no Domínio de Tríades representam apenas associações do tipo relacionamento entre classes;

interpretação: ao pré-especificar algumas abstrações (o conceito de atributos representa uma agregação dos conjuntos correspondentes aos conjuntos de valores possíveis) permite-se agilizar o tratamento computacional. No entanto, é exigido que haja um conhecimento sobre os dados armazenados, o que não era exigido no CORAS, onde valores, conjuntos e atributos poderiam ser acessados da mesma forma (eram todos acessados através do nome dos objetos correspondentes). No SIGA, o acesso a um valor é definido através do acesso a um atributo que, por sua vez, é acessado através do conhecimento do objeto a que pertence.

Outra característica do SIGA é a existência do Dicionário e Diretório, uma base de dados manipulada paralelamente à base de dados da aplicação. Este princípio de manipulação simultânea pode ser expandido de duas para diversas bases de dados, onde várias aplicações podem ser interconectadas através do DDS, que controlaria também a alocação de áreas em memória para as bases de dados.

Os tipos de aplicações que podem utilizar o SIGA são diversos. A flexibilidade de estruturas de armazenamento e a possibilidade de representação de mais de um tipo de informação por objeto permite representar uma quantidade de informações semânticas muito maior que a obtida em sistemas de armazenamento de dados convencionais. Por outro lado, é um sistema computacionalmente viável, principalmente levando em conta o desenvolvimento de processadores dedicados (manipulação de listas, funções de acesso a dados, máquinas de inferências) e sistemas multiprocessadores.

Dentre as aplicações possíveis para o SIGA, uma será analisada em maior detalhe. Trata-se de sua utilização como núcleo de Sistemas de Gerência de Banco de Dados, em particular para o Sistema Gerenciador de Dados para PAC, o GERPAC. A implementação de um SGBD para PAC utilizando internamente um sistema voltado para a manipulação de objetos constitui uma alternativa em relação a outras propostas, que em geral partem de um SGBD baseado em modelos de dados convencionais, utilizados em aplicações comerciais, com camadas externas que visam adequá-los a estas aplicações. A especificação do SIGA o torna muito adequado a representações de dados através da abordagem entidade-relacionamento, permitindo que SGBD que sejam suportados por ele possam suportar diretamente modelos como o MER e o MER/PAC, que é o caso de GERPAC, que será apresentado no próximo capítulo.

Capítulo 5

O GERPAC

5.1 INTRODUÇÃO

No Capítulo 3, as vantagens da utilização de Sistemas de Banco de Dados em aplicações PAC foram ressaltadas, assim como a dificuldade da adaptação de Sistemas já existentes a estas aplicações. As principais dificuldades dizem respeito à diferença de características de dados em PAC, não representáveis de forma adequada pelas estruturas de dados providas por sistemas convencionais, e ao aspecto de controle, com conceitos de transação, integridade e consistência essencialmente distintos do caso convencional.

A representação de dados em aplicações PAC foi estudada por Delgado, resultando no Modelo Entidade-Relacionamento estendido para PAC (o MER/PAC, apresentado no Item 3.3) [/DEL87/]. O aspecto de estruturas de dados foi apresentado no Capítulo 4; o SIGA é um sistema de software básico que permite manipular estruturas de dados flexíveis, adequadas a aplicações não convencionais na área de gerência de dados incluindo, como se pretende mostrar neste capítulo, aplicações PAC.

O objetivo do presente capítulo é apresentar o Sistema Gerenciador de Dados para PAC — GERPAC, Sistema de Gerência de Banco de Dados que suporta os elementos primitivos do MER/PAC. A implementação deste Sistema é suportada pelo SIGA, que é responsável pela estrutura interna do armazenamento dos dados. Na análise do modo pelo qual o SIGA suporta o GERPAC, basicamente dois aspectos devem ser observados:

- como os elementos primitivos do GERPAC (que são os elementos primitivos do MER/PAC) podem ser representados em termos de elementos primitivos do SIGA, e
- como as funções GERPAC são definidas em termos de funções SIGA.

Os elementos primitivos do GERPAC são apresentados no Item 5.2. O aspecto de tradução de primitivas GERPAC em termos de primitivas SIGA é abordado no Item 5.3, seguido pela apresentação das funções GERPAC (Item 5.4). O outro aspecto relevante no suporte que o SIGA provê ao GERPAC, o funcional, é abordado no Item 5.5. Finalmente, o suporte a regras é analisado no Item 5.6.

5.2 ELEMENTOS PRIMITIVOS

Os elementos primitivos do GERPAC são aqueles referentes ao MER/PAC, ou seja, Tipos de Entidades, Tipos de Relacionamentos, Agrupamentos e Esquemas.

Tipos de Entidades representam classes que contêm entidades, que são as unidades de informação do Modelo. Ao tipo de entidade podem ser associados atributos, que definem as propriedades da classe. Um tipo de entidade pode ser regular, quando a existência de entidades componentes da classe é definida independentemente das entidades de outras classes, ou fraco, quando há dependência em relação a entidades de outras classes.

Tipos de Relacionamentos representam classes que contêm relacionamentos, que associam elementos dos conjuntos ligados através do tipo de relacionamento. Atributos podem também estar associados ao tipo de relacionamento, definindo suas propriedades. Tipos de relacionamentos podem ser regulares, quando a remoção de um relacionamento não afeta os elementos envolvidos na associação, ou fracos, quando a remoção de um relacionamento pode acarretar a remoção de parte dos elementos envolvidos na associação.

Agrupamentos contêm conjuntos de elementos primitivos do GERPAC, permitindo diferentes níveis de abstração dentro do micro-aspecto de projeto (Item 3.3.1). Assim como tipos de entidades, agrupamentos podem ser regulares ou fracos. Agrupamentos permitem representar tanto abstrações do tipo agregação quanto abstrações do tipo generalização.

Esquemas são os elementos primitivos que permitem a representação do macro-aspecto de projeto (Item 3.3.2). Cada representação de um micro-aspecto de projeto (expansão), em conjunto com os dados operacionais associados (versão), representa uma ocorrência do esquema ao qual a representação está associada. Esquemas também podem ser classificados quanto a sua dependência em regulares ou fracos.

Além dos elementos primitivos, uma ferramenta importante na representação das informações no GERPAC é a especificação de Regras, que podem ser associadas a qualquer dos elementos primitivos acima.

Regras assumem papel fundamental na descrição de características semânticas adicionais àquelas representáveis diretamente pelo modelo, como ocorre no caso de agrupamentos (através de regras é possível especificar a função do agrupamento: agregação, generalização, objeto complexo, relacionamento exclusivo). Podem também ser utilizadas para suportar diversos

aspectos de controle, como será destacado no Item 5.6.

5.3 MAPEAMENTO GERPAC/SIGA

Embora os elementos primitivos de SIGA sejam adequados a representações que utilizem a abordagem entidade-relacionamento, não existe uma associação direta e/ou pré determinada entre estes elementos e os primitivos de GERPAC. A flexibilidade do núcleo é um dos motivos pelo qual esta associação não é direta, pois dependendo do tipo de modelo adotado para o SGBD este mapeamento será diverso.

Chen propõe uma estrutura preliminar para classificar modelos que utilizam a abordagem entidade-relacionamento [/CHE81b/]. São propostas duas grandes categorias, GERM (Generalized Entity-Relationship Models), que permite relacionamentos n-ários, e BERM (Binary Entity-Relationship Models), que permite apenas relacionamentos binários. Existe uma subclassificação que leva em conta a possibilidade de representação de atributos, envolvendo mais três classes: atributos para entidades e relacionamentos, atributos apenas para entidades e sem atributos. Assim, são definidas no total seis classes:

(1) GERM com atributos para entidades e relacionamentos;

(2) GERM com atributos para entidades apenas;

(3) GERM sem atributos;

(4) BERM com atributos para entidades e relacionamentos;

(5) BERM com atributos para entidades apenas;

(6) BERM sem atributos.

A implementação de um Sistema com um modelo da classe 6 poderia ser suportado a partir do CORAS, que não inclui facilidades para a manipulação de atributos. Já um da classe 5 poderia utilizar o SIGA quase que diretamente, com objetos simples representando tipos de entidade e objetos relação representando tipos de relacionamentos. As associações binárias seriam representadas diretamente por tríades.

O modelo suportado pelo GERPAC, MER/PAC, está enquadrado na classe 1. Portanto, o mapeamento aqui descrito foi proposto como forma de permitir o máximo de recursos que a abordagem entidade-relacionamento proporciona como ferramenta para a modelagem de informações. Além do mais, como esta é a classe mais geral dentre as seis, um mapeamento para um modelo de outra classe poderá ser derivado a partir das características deste.

5.3.1 Tipos de Entidade e de Relacionamento

Existem no MER/PAC, assim como no MER, dois elementos primitivos para representar o conceito de classe primária (definida a partir de unidades de informação e não a partir de outras classes). São o tipo de entidade e o tipo de relacionamento. Estes dois elementos são representados no SIGA como objetos simples, que permitem a representação de classes e definição de atributos associados a elas. Desta forma, atinge-se um dos requisitos, que é a representação de atributos para entidades e para relacionamentos. As instâncias destas classes, entidades e relacionamentos, são representadas por ocorrências do objeto simples correspondente.

Resta um outro requisito, que é a possibilidade de se representar relacionamentos n-ários. Objetos relação e tríades permitem apenas relacionamentos binários, de forma que não são diretamente adequados. O mecanismo adotado para contornar esta limitação foi representar o papel da associação como um relacionamento binário, o que é verdadeiro pois um papel sempre associa um tipo de entidade (ou primitiva equivalente, no MER/PAC) a um tipo de relacionamento, caracterizando um relacionamento do tipo binário.

Cada objeto relação está associado a um papel. A representação de relacionamentos n-ários é possível considerando-se que as associações de um relacionamento são compostas pela união de todas as tríades que ligam outras primitivas ao tipo de relacionamento através dos papéis. O objeto relação que representa o papel não tem atributos definidos pelo usuário, mas apenas os pré-definidos pelo Sistema, que permitem representar o mapeamento e a dependência dos objetos associados. Esta forma de mapeamento, além de conter uma informação semântica adicional (o papel), ainda permite a representação de auto-relacionamentos, o que não seria possível se tipos de relacionamentos estivessem diretamente associados a objetos relação.

5.3.2 Agrupamentos

Agrupamentos são mapeados como objetos conjunto do SIGA. Desta forma, permite-se que um agrupamento possa conter tipos de entidade, tipos de relacionamento, agrupamentos (um objeto conjunto pode conter outros objetos conjunto) e mesmo esquemas. A fim de suportar a flexibilidade necessária, a definição de como são constituídas as ocorrências de cada agrupamento em particular não é pré-definida pelo Sistema, devendo ser suportada através de definição de regras associadas ao agrupamento.

A definição de ocorrências de um agrupamento exige esta flexibilidade. Por exemplo, a ocorrência de um agrupamento que representa agregação é distinta da ocorrência de um agrupamento que representa uma generalização. No primeiro caso, uma ocorrência seria definida a partir de ocorrências dos elementos constituintes, podendo não haver atributos associados ao agrupamento. No caso de generalização, uma ocorrência seria definida associando atributos

aos elementos do agrupamento.

5.3.3 Esquemas

Esquemas têm representação diferenciada de acordo com o nível em que o usuário está trabalhando. Se um usuário está definindo a representação de um esquema, então este esquema será encarado como um Sistema de Arquivos SIGA (uma base de dados distinta para cada esquema). Ao definir uma nova expansão, uma nova base de dados será criada, e o mesmo ocorre para a criação de novas versões de cada expansão. Em outras palavras, cada instância de um esquema corresponde a uma base de dados SIGA. Cada uma destas bases de dados definidas durante um projeto é identificada de modo completo através do nome do esquema, identificação da expansão e identificação da versão associada à expansão.

Além desta representação, para cada base de dados associada a uma instância de um esquema existe um objeto conjunto associado, definido internamente, que permite representar as informações locais associadas a cada representação (e.g., elementos constituintes e regras particulares a cada representação).

Se o usuário estiver referenciando um esquema interno à representação de um outro esquema distinto, esta referência a um esquema é mapeada como um objeto simples do SIGA, pois é uma referência à classe que contém todas as bases de dados associadas àquele esquema. As ocorrências deste objeto simples estão univocamente associadas às bases de dados correspondentes às instâncias do esquema, sendo que 'identificação da expansão' e 'identificação da versão' são dois atributos pré-definidos pelo Sistema, assim como 'consistência', contendo um código que representa o estado de consistência de cada base de dados.

5.3.4 Dependências

A dependência dos elementos primitivos (se regulares ou fracos) é definida através dos mesmos objetos relação que representam os papéis. Como existe uma subclassificação associada a cada objeto relação que define ou não a propagação de remoções, é possível representar associações regulares como tríades utilizando objetos relação com a chave 'não propagação', enquanto que associações fracas são representadas por objetos relação com a chave 'propagação'.

Para fins de manipulação, a ordem dos elementos em uma tríade estabelecida no mapeamento é importante, sendo especificada sempre na forma Papel (Elemento associado, Tipo de Relacionamento). Deste modo, a propagação de remoções em elementos dependentes pode ser automaticamente tratada pelo SIGA, evitando inconsistências. Elementos do segundo objeto em uma tríade são automaticamente removidos quando um elemento do primeiro objeto é removido; no mapeamento proposto, isto implica que a remoção de, por exemplo, uma entidade, acarreta a remoção de todos os relacionamentos que estiverem relacionados com ela.

A chave de propagação permite que a remoção de elementos do segundo objeto implique a remoção de elementos do primeiro objeto da tríade, o que permite mapear a dependência (remoção de relacionamentos provocando a remoção de instâncias dos objetos dependentes).

Outro ponto a destacar é que a definição da dependência de qualquer elemento, inclusive para o tipo de relacionamento, é realizada apenas no momento de definição de associações, quando deve ser especificada a regularidade de cada associação. Esta forma de manipulação é mais adequada que definir a priori se um tipo de entidade, por exemplo, será ou não dependente; no momento em que se define uma associação fraca envolvendo o tipo de entidade, seus elementos passarão a ser tratados como entidades dependentes, e no momento em que esta associação fraca for removida estes elementos serão tratados como entidades regulares, sem necessidade de explicitar esta modificação. O mesmo ocorre para as instâncias de agrupamentos e esquemas.

A fim de diferenciar a função que cada objeto SIGA pode assumir no mapeamento (por exemplo, objetos simples podem representar tipos de entidade, tipos de relacionamento ou referências a esquemas), um código interno (ou classificação interna) é armazenado. No SIGA, para armazenar este código é utilizado o campo reservado para "uso interno" na lista de atributos do objeto (Figura 4.4).

Um quadro resumindo este mapeamento é apresentado na Figura 5.1.

5.4 FUNÇÕES GERPAC

O GERPAC provê funções que permitem a representação e a manipulação tanto do micro-aspecto quanto do macro-aspecto do processo de projeto. As ferramentas que são utilizadas para a representação do micro-aspecto são as mesmas utilizadas para a representação do macro-aspecto, sendo que a conexão entre diferentes níveis de processo é estabelecida através de instâncias de esquemas, representadas por expansões (representações) e versões (dados operacionais) associados.

As funções providas pelo Sistema GERPAC podem ser englobadas em três categorias principais:

representação: permitem a definição e manipulação de um esquema conceitual utilizando as primitivas MER/PAC. Trabalham a nível de cada expansão de um esquema MER/PAC;

instâncias: permitem a definição e manipulação de conjuntos de dados operacionais associados a cada expansão de um esquema MER/PAC. Trabalham a nível de unidades de informação;

apoio: permitem a manipulação a nível de diretório/dicionário de dados, além de prover

GERPAC \ SIGA	Objeto Simples	Objeto Conjunto	Objeto Relação	Tríade	Base de Dados
Tipo de Entidade	○				
Tipo de Relacionamento	○				
Agrupamento		○			
Esquema	○ (referência)	○ (elementos)			○ (instância)
Papel			○		
Associação				◎	

Figura 5.1: Mapeamento SIGA-GERPAC.

funções que permitem o controle do Sistema e a agilização do acesso aos dados operacionais.

A seguir serão apresentadas estas funções, de acordo com os grupos que as caracterizam.

5.4.1 Funções: Representação

As funções deste grupo permitem a definição e manipulação da representação de um esquema através da criação das primitivas MER/PAC (tipos de entidade, de relacionamento, agrupamentos e esquemas) e de suas associações.

Definição:

- CRIAR TIPO DE ENTIDADE: permite criar uma referência a um objeto através da definição de um nome a ele associado. Esta função não define os atributos do tipo de entidade;
- CRIAR TIPO DE RELACIONAMENTO: semelhante à função acima. Criar um tipo de relacionamento não implica definir associações, mas apenas definir a classe;
- CRIAR AGRUPAMENTO: a criação de um agrupamento não implica definir quais são os seus elementos, mas criar uma referência (seu nome) para a futura definição de seus elementos (através da função 'Inserir no Agrupamento');
- REFERENCIAR ESQUEMA: permite citar outros esquemas subordinados ao que está sendo definido. Além de criar a referência ao esquema, as identificações (expansões/versões) já existentes são registradas como instâncias do esquema (através dos atributos inerentes a cada esquema, 'expansão', 'versão' e 'consistência'). Se o esquema referenciado não foi ainda especificado (não há base de dados associada), o esquema atual é sinalizado como "inconsistente";
- ESTABELECER ASSOCIAÇÃO COM TIPO DE RELACIONAMENTO: permite a definição de associações n-árias entre tipos de entidades, agrupamentos ou esquemas com tipos de relacionamento, especificando também o papel correspondente à associação e sua regularidade;
- INSERIR NO AGRUPAMENTO: permite a união de quaisquer primitivas já definidas em agrupamentos previamente criados;
- DEFINIR ATRIBUTO: acrescenta um atributo à primitiva especificada. A definição de um atributo implica a definição de seu nome e tipo, que pode ser um dos tipos suportados através do Dicionário.

Atualização

- ALTERAR NOME DA PRIMITIVA: o nome constitue a identificação de uma primitiva. Esta função permite alterar o nome de qualquer primitiva referente ao micro-aspecto de um esquema. Referências a outros esquemas não podem ser alteradas deste modo, devendo qualquer alteração ser realizada explicitamente através de remoções e criações, como forma de manter a consistência das informações sobre expansões e versões;
- REMOVER PRIMITIVA: permite eliminar da definição do esquema qualquer primitiva que tenha sido definida, levando em conta o fato de conter informações e/ou associações com outras primitivas, ou seja, se houver informações associadas estas serão removidas e as remoções podem se propagar de acordo com as dependências das associações existentes;
- REMOVER ASSOCIAÇÃO: permite desassociar primitivas que estavam associadas por um tipo de relacionamento;
- RETIRAR DO AGRUPAMENTO: permite a desagregação parcial (retira um elemento) ou total (retira todos os elementos) de um agrupamento;
- REMOVER UM ATRIBUTO: permite remover da definição de atributos de uma primitiva o atributo especificado;
- ALTERAR NOME DO ATRIBUTO: permite alterar a forma de referência a um determinado atributo de uma primitiva;
- ALTERAR TIPO DO ATRIBUTO: permite alterar o tipo de dado associado a um atributo.

Leitura

- OBTER ELEMENTOS DO ESQUEMA: permite a obtenção de todos os elementos e associações constituintes de um esquema;
- VERIFICAR EXISTÊNCIA DA PRIMITIVA: permite analisar a existência de uma determinada primitiva referenciada através de seu nome;
- VERIFICAR CATEGORIA DA PRIMITIVA: para uma determinada primitiva, verifica sua categoria (se é tipo de entidade, de relacionamento, agrupamento ou esquema);
- VERIFICAR DEPENDÊNCIA DA PRIMITIVA: para a primitiva especificada, analisa sua dependência em relação a outras primitivas (regular, fraca);

- OBTER OS ELEMENTOS DE UMA ASSOCIAÇÃO: permite questionar uma associação, seja a partir do papel, do tipo de relacionamento ou das primitivas associadas, permitindo definir se há e quais são os elementos dependentes;
- OBTER OS ELEMENTOS DE UM AGRUPAMENTO: permite a obtenção de quais primitivas estão reunidas através de um mesmo agrupamento;
- OBTER ATRIBUTOS: descreve quais os atributos definidos para uma primitiva.

5.4.2 Funções: Instâncias

Estas funções permitem a gerência dos dados correspondentes às unidades de informação associadas às primitivas MER/PAC, ou seja, dos dados referentes a entidades, relacionamentos e ocorrências de agrupamentos e esquemas, através da definição de valores de seus atributos. Não existem funções para a criação de ocorrências de esquemas, pois estas são definidas a partir das informações existentes no Dicionário do Sistema sobre suas expansões e versões.

As operações referentes à definição (criação e remoção) e atualização provocam verificações das regras que porventura estejam associadas à primitiva, seja esta um tipo de entidade, tipo de relacionamento, agrupamento ou esquema.

Definição

- DEFINIR ENTIDADE: permite especificar valores de atributos para uma ocorrência no tipo de entidade;
- DEFINIR RELACIONAMENTO: a especificação de um relacionamento implica a definição de quais ocorrências dos elementos envolvidos estão associadas. Também permite especificar valores de atributos para o relacionamento;
- DEFINIR OCORRÊNCIA DE AGRUPAMENTO: similar à função 'Definir Entidade';
- REMOVER ENTIDADE: elimina as referências a uma ou mais ocorrências do tipo de entidade, efetivando as propagações que forem necessárias de acordo com a dependência de suas associações;
- REMOVER RELACIONAMENTO: semelhante em efeito à função 'Remover Entidade', com diferença na forma possível de se especificar quais ocorrências serão removidas: através dos valores de seus atributos próprios (como no caso de entidades) ou através da especificação das ocorrências de elementos associados pelo tipo de relacionamento;
- REMOVER OCORRÊNCIA DE AGRUPAMENTO: similar à função 'Remover Entidade'.

Atualização

- ATUALIZAR VALOR DO ATRIBUTO: permite alterar valores já associados aos atributos de ocorrências de elementos primitivos do GERPAC, incluindo referências a esquemas.

Leitura

- OBTER VALORES DOS ATRIBUTOS: permite a recuperação dos valores de atributos de ocorrências associadas a elementos primitivos do GERPAC.

5.4.3 Funções: Apoio

São funções que permitem a gerência global das Bases de Dados e do Sistema, definição de esquemas e apoio às funções de manipulação de ocorrências. Parte destas funções refere-se ao Dicionário do Sistema, que não é necessariamente único para todas as aplicações: quando o Sistema é iniciado, permite-se especificar a base de dados correspondente ao Dicionário que irá ser utilizado.

- DEFINIR REPRESENTAÇÃO DO ESQUEMA: corresponde a iniciar a definição de uma representação (expansão) de um esquema;
- DEFINIR NOVA VERSÃO DO ESQUEMA: permite iniciar a definição de um esquema a partir de um esquema já existente, incluindo suas ocorrências;
- REMOVER ESQUEMA: função a nível de gerência do Sistema, permite remover uma ou mais bases de dados referentes a um esquema, atualizando as informações no Dicionário do Sistema;
- OBTER ESQUEMAS DEFINIDOS: obtém a lista de esquemas definidos relativos ao Diretório que está sendo utilizado;
- ESTABELECER ÍNDICE PARA ATRIBUTO: permite agilizar o acesso através do valor do atributo especificado;
- REMOVER ÍNDICE PARA ATRIBUTO: libera o índice criado para o atributo especificado;
- REGISTRAR TIPO DE DADO: permite registrar no Dicionário do Sistema um tipo de dado, especificado através de um nome de identificação e suas características (e.g., domínio, operações/operadores). Através desta função é possível especificar outros tipos de dados que não 'caráter';

- OBTER TIPOS DE DADOS SUPORTADOS: permite a verificação de quais tipos de dados são suportados pelo Sistema;
- OBTER CARACTERÍSTICAS DO TIPO DE DADO: descreve os atributos do tipo de dado especificado.

5.5 INTERFACE GERPAC/SIGA

Neste item serão descritas as funções GERPAC, apresentadas no Item 5.4, em termos de funções suportadas pelo SIGA. Serão destacadas as funções que não são diretamente traduzidas, sendo que muitas das funções apresentadas (como as de manipulação de atributos) têm correspondência direta com funções SIGA.

5.5.1 Funções: Representação

Definição

- CRIAR TIPO DE ENTIDADE: cria o objeto simples correspondente, armazenando-o como elemento do objeto conjunto definido internamente que representa o esquema. Um código interno é definido para registrar que a classe representa um tipo de entidade;
- CRIAR TIPO DE RELACIONAMENTO: cria o objeto simples correspondente, armazenando-o como elemento do objeto conjunto definido internamente que representa o esquema. Sua classificação interna armazenada é a correspondente a "tipo de relacionamento";
- CRIAR AGRUPAMENTO: cria o objeto conjunto correspondente ao agrupamento, armazenando uma classificação interna associada a "agrupamento";
- REFERENCIAR ESQUEMA: cria o objeto simples correspondente, com classificação interna de "esquema". Analisa se já existe representação definida para este esquema no Diretório, permitindo sinalizar possíveis inconsistências. Se já existem expansões e versões registradas no Diretório, estas são registradas como ocorrências do esquema;
- ESTABELECER ASSOCIAÇÃO: ao se estabelecer uma associação, é necessário definir qual o papel por ela representado, quais os elementos envolvidos e qual a sua regularidade. O papel é traduzido pela criação do objeto relação correspondente no SIGA, e sua associação com os elementos envolvidos é traduzido pela criação de uma tríade na forma Papel (Primitiva diferente de Tipo de Relacionamento, Primitiva Tipo de Relacionamento);

- INSERIR NO AGRUPAMENTO: armazena o objeto especificado como elemento do objeto conjunto que representa o agrupamento;
- DEFINIR ATRIBUTO: acrescenta um atributo à lista de atributos do objeto especificado.

Atualização

- ALTERAR NOME DA PRIMITIVA: define o novo nome como sinônimo do objeto, removendo posteriormente o nome antigo;
- REMOVER PRIMITIVA: remove o objeto correspondente. A remoção de um esquema através desta função permite remover apenas a referência e não a base de dados correspondente;
- REMOVER ASSOCIAÇÃO: desassocia a tríade que define a associação, removendo posteriormente o objeto relação correspondente;
- RETIRAR DO AGRUPAMENTO: remove o objeto especificado do objeto conjunto que representa o agrupamento. Se especificado, pode remover todos os elementos do agrupamento através da função SIGA correspondente a "Remover todos os elementos do conjunto";
- REMOVER ATRIBUTO: remove da lista de atributos do objeto o atributo especificado;
- ALTERAR NOME DO ATRIBUTO: altera o nome do atributo na lista de atributos do objeto;
- ALTERAR TIPO DO ATRIBUTO: altera o tipo armazenado para o atributo especificado na lista de atributos do objeto.

Leitura

- OBTER ELEMENTOS DO ESQUEMA: obtém elementos do objeto conjunto definido internamente como representação do esquema. Para tanto, o Sistema de Arquivos correspondente ao Esquema deve estar ativo;
- VERIFICAR EXISTÊNCIA DA PRIMITIVA: verifica a existência do objeto;
- VERIFICAR A CATEGORIA DA PRIMITIVA: analisa o objeto e sua classificação interna, retornando a primitiva que é mapeada para o objeto;

- VERIFICAR DEPENDÊNCIA: através da análise das tríades ligadas ao objeto e obtenção dos objetos relação a ele associado, e após analisar para estes objetos relação as chaves de propagação correspondente, permite retornar sua classificação quanto à dependência;
- OBTER ELEMENTO DA ASSOCIAÇÃO: a partir de qualquer elemento conhecido da associação (papel, tipo de relacionamento, outra primitiva) é possível questionar outros elementos associados diretamente através do questionamento de tríades suportado pelo SIGA;
- OBTER ELEMENTOS DO AGRUPAMENTO: obtém os elementos do objeto conjunto que representa o agrupamento;
- OBTER ATRIBUTOS: obtém nomes e tipos dos atributos definidos pelo usuário para os elementos primitivos a partir da lista de atributos do objeto.

5.5.2 Funções: Instâncias

Definição

- DEFINIR ENTIDADE/RELACIONAMENTO: armazena uma ocorrência correspondente ao objeto simples especificado;
- REMOVER ENTIDADE/RELACIONAMENTO: remove uma ocorrência do objeto simples especificado;
- ATUALIZAR VALOR DO ATRIBUTO: utiliza a função SIGA correspondente;
- OBTER VALORES DOS ATRIBUTOS: utiliza a função SIGA correspondente.

5.5.3 Funções: Apoio

- DEFINIR REPRESENTAÇÃO DO ESQUEMA: inicia um Sistema de Arquivos cujos parâmetros de iniciação são definidos pela interface e cujo nome é o definido para o Esquema. Além de definir o Sistema de Arquivos, cria o seu primeiro elemento — o objeto conjunto interno que permite representar os atributos locais associados ao esquema; sua classificação interna é registrada para o valor correspondente a "esquema". O Diretório do Sistema é automaticamente atualizado pelo SIGA.

As demais funções deste grupo utilizam funções SIGA correspondentes, sendo que as funções referentes a índices têm efeito interno a cada base de dados, enquanto que as demais são gerais (são funções a nível de Diretório/Dicionário SIGA).

5.6 SUPORTE A REGRAS

Foi destacada a importância da utilização de regras para a representação das informações em PAC, inclusive na definição de ocorrências em agrupamentos e manutenção da consistência entre esquemas. No entanto, a especificação do GERPAC apresentada não inclui, pelo menos não como uma unidade integrada ao SGBD, o sistema de avaliação de regras requerido por MER/PAC. O que o GERPAC provê, através da possibilidade de se definir múltiplos conjuntos de atributos para um mesmo objeto no SIGA, são meios para a definição e o armazenamento destas regras.

A nível do SIGA, regras são tratadas como atributos normais, podendo ser definidas, atualizadas, consultadas ou removidas. Para tanto, é utilizada a facilidade que o SIGA provê de especificar mais de uma máscara de atributos para um mesmo objeto. O GERPAC é responsável, neste sistema, por registrar quais os rótulos (identificação de cada um dos conjuntos de atributos de um objeto) que estão associados a regras e quais estão associados a outros tipos de atributos.

A estrutura das regras, ou seja, a forma como estas são expressas, não é pré fixada pelo GERPAC; ela depende do sistema de tratamento de regras que será integrado ao módulo central como um subsistema do SGBD. Assim, se tal sistema for definido como um sistema de inferências utilizando programação lógica, o tipo de dado "regra" que ser registrado no Diretório/Dicionário SIGA deverá ter características tais que reflitam a estrutura de claúsulas, por exemplo. Se tal subsistema utilizar regras de produção, o tipo de dado "regra" apresentará características diferentes. De qualquer modo, o que se pretendeu foi que o GERPAC suportasse regras de forma modular, de modo que não se perca a flexibilidade necessária a tal sistema.

É importante ressaltar a importância destas regras GERPAC também no suporte aos aspectos de controle em SGBD, principalmente quanto à Integridade, Consistência e Segurança. Para tanto, regras podem ser definidas com relação a um atributo de um objeto (especificando faixa de valores permitidos, por exemplo), envolvendo interrelacionamentos entre atributos de um mesmo objeto ou de objetos distintos (neste último caso, através da definição de um agrupamento contendo os objetos envolvidos e estando a regra associada ao agrupamento, por exemplo) e para agrupamentos, permitindo especificar seu aspecto semântico.

Com relação ao aspecto de Segurança, é possível aproveitar o fato de que o banco de dados de um projeto controlado pelo GERPAC é segmentado, isto é, composto por diversas bases de dados independentes [/CHA80/]. Desta forma, é possível estabelecer autorizações de acesso a algumas bases de dados e bloquear o acesso a outras simplesmente estabelecendo regras no esquema de mais alto nível na hierarquia de esquemas (o "super-esquema"). Neste nível, usuários podem ser cadastrados em um tipo de entidade e as bases de dados (os esquemas definidos no Diretório do projeto) em outro tipo de entidade. O acesso pode ser especificado

como um tipo de relacionamento entre estes dois tipos de entidade, onde uma regra pode estabelecer a verificação da permissão. É interessante notar que esta forma de definição pode ser estendida para prover uma forma de Controle de Concorrência.

Com relação ao conceito de transação PAC, o tratamento das regras é de importância vital, pois é principalmente através delas que se pode especificar a consistência entre esquemas (a consistência a nível de um único esquema, ou seja, no micro-aspecto, recai no problema de tratamento de transações convencionais). Dependendo da quantidade de níveis na hierarquia de esquemas, pode se tornar inviável a verificação imediata das regras em todos os níveis a cada ação que altere as caracterísiticas de um esquema. Uma alternativa neste caso é a especificação de momentos posteriores ao momento da ação para a verificação das regras. Outra possibilidade é aproveitar as características de bancos de dados segmentados (caso do GERPAC) e explorar as possibilidades que estes apresentam com relação ao multiprocessamento, permitindo a verificação de regras em diversos níveis de forma simultânea e no momento da tomada da ação.

5.7 COMENTÁRIOS

Um ponto a destacar na apresentação deste capítulo foi a possibilidade de se desenvolver um SGBD que suporte um modelo de dados de nível mais próximo ao conceitual (como definido na Figura 2.1, referência /DEL87/), como é o caso do MER/PAC e de outros modelos que utilizam a abordagem entidade-relacionamento. É interessante notar que a utilização do SIGA como núcleo do Sistema permitiu evitar os problemas que normalmente são encontrados nas tentativas de se mapear modelos tradicionais, como o relacional e o hierárquico, para modelos tipo entidade-relacionamento, como questões de definição de chaves e dependência de identificação.

Convém notar que o Sistema proposto não engloba todos os aspectos relativos a um SGBD, não tendo sido abordados Sistemas de Integridade, Segurança, Recuperação e Concorrência (relativos a SGBD convencionais) e o Sistema de Regras. O módulo apresentado pretende constituir o módulo central de um SGBD que suporte, no futuro, estes outros módulos como seus subsistemas, motivo pelo qual não foram detalhados nesta apresentação.

A principal característica do GERPAC que define sua adequação a aplicações PAC é a possibilidade de se gerenciar o processo de projeto (o macro-aspecto) através das mesmas ferramentas com que se manipulam os elementos do projeto (seu micro-aspecto). A seguir é descrita uma das possíveis formas de utilizar o MER/PAC para modelar este processo de projeto.

O início de um projeto é a definição de seu esquema global, onde são definidas as diretrizes do projeto. Neste nível podem ser definidos os diversos esquemas que representam os diferentes módulos constituintes do projeto, relacionados de forma a orientar o processo

de projeto. A definição de um esquema é flexível, pois à primitiva Esquema podem estar associadas distintas expansões representando as diferentes opções possíveis de projeto. São as representações de um esquema. Expansões de um esquema podem conter outros esquemas, sendo que nos níveis mais elevados de definição estarão representadas as estratégias de projeto (tratamento do esquema como um todo). Uma expansão de um esquema representa o nível local para este esquema em particular, mas pode representar o nível global para outros esquemas que o constituem.

Pode existir mais de um desenvolvimento associado a uma representação, ou seja, dados operacionais diferentes para uma mesma definição de um esquema. São as distintas versões que podem estar associadas a cada representação. Versões podem ser iniciadas ou geradas a partir de versões já existentes, mantendo a versão original inalterada. Este tipo de operação pode ser interessante para desenvolver linhas de projeto alternativas durante o processo.

O controle das operações ao nível de representações e versões de esquemas (controle global) é suportado através de regras associadas aos esquemas. A sequência de operações que são executadas a partir de alterações neste nível (a alteração em si, a verificação de regras e as operações que devem ser consequentemente executadas) constitui a transação macro. O momento de se tratar a transação macro depende do sistema computacional que suporta a implementação do GERPAC. Em máquinas convencionais, esta macro-transação poderia ser postergada para um momento posterior a fim de não prejudicar a velocidade de interação entre o usuário e o Sistema, correndo o risco de se trabalhar com informações inconsistentes. Em um sistema computacional com arquitetura de processamento paralelo, a macro-transação poderia ser tratada paralelamente a outras operações, de forma que não houvesse prejuízo para a interação e agilizando o tratamento da consistência global da aplicação.

Através da utilização da estrutura de representações e versões de esquemas é possível também gerenciar a sequência de projeto, permitindo o controle de suas características temporais.

Capítulo 6

CONCLUSÕES

O objetivo inicial do presente trabalho foi o de analisar a viabilidade de se utilizar um sistema para armazenamento de dados, o CORAS, como núcleo para a implementação de SGBD, com interesse principalmente voltado para aplicações de Engenharia. Em particular, na Faculdade de Engenharia Elétrica da UNICAMP havia dois trabalhos na área de especificação de Modelos de Dados para este tipo de aplicações, uma para Controle de Processos [/WU 84/] e outra para Projeto Auxiliado por Computador (PAC) [/DEL87/], sendo que ambas especificavam extensões ao Modelo Entidade-Relacionamento [/CHE76/].

Foi estudada a utilização do CORAS como núcleo para SGBD que suportasse este tipo de modelos de dados [/RIC85/, /RIC86/, /DEL86b/], e a conclusão atingida foi que este sistema apresentava características extremamente importantes para suportar tais sistemas, porém para este tipo de abordagem apresentava algumas deficiências tais como a não distinção entre objetos e seus atributos.

Assim o primeiro resultado deste trabalho foi a proposta de um sistema para representação de dados, o SIGA, com especificação derivada a partir do CORAS, que inclui facilidades adicionais para o suporte aos SGBD para Engenharia e que, devido a suas características, pode ser utilizado também em outros tipos de aplicação que envolvam o mesmo tipo de abordagem (entidades-relacionamentos-atributos). Este Sistema foi descrito no Capítulo 4, sendo que a implementação de um protótipo está em andamento. Descrições mais detalhadas com relação a aspectos de implementação do SIGA podem ser encontradas em relatórios técnicos [/RIC86b/].

Uma vez especificado o Sistema núcleo, o interesse foi verificar a viabilidade de sua utilização para os Sistemas desejados. O modelo adotado como caso de teste foi o MER/PAC, a extensão do MER para aplicações PAC [/DEL87/] apresentada no Capítulo 3, sendo especificado um SGBD que suportasse todas as características definidas. Este Sistema, o GERPAC, o segundo resultado deste trabalho, foi apresentado no Capítulo 5 e constitui o módulo central

de um SGBD para aplicações em Engenharia, incluindo facilidades para o armazenamento e manipulação das variadas formas de informações que surgem nestas aplicações.

Uma outra contribuição deste trabalho foi a proposta de uma alternativa para o desenvolvimento de SGBD para PAC, não partindo de sistemas disponíveis no mercado (de características convencionais) nem tampouco abordando uma aplicação em particular. Ao suportar um modelo como o MER/PAC, pretende-se que este SGBD proposto possa suportar um amplo espectro de aplicações, não só em projetos de Engenharia como também em outras aplicações onde as estruturas convencionais não se mostraram adequadas (por exemplo, Engenharia de Software e Recuperação de Textos). Foi importante mostrar a viabilidade de se implementar tal Sistema que, embora ainda não tenha um protótipo totalmente implementado, tem seu módulo central especificado. Este módulo central do SGBD cobre as necessidades referentes à representação dos dados, restando realizar estudos com relação aos aspectos de controle (descritos no Capítulo 2). Uma alternativa neste caso é a utilização de subsistemas baseados nas regras GERPAC, conforme descrito no Suporte a Regras (Item 5.6).

A implementação de um protótipo GERPAC deverá seguir imediatamente a finalização do primeiro protótipo SIGA, sendo que deverá incluir as funções básicas relativas ao micro-aspecto do processo de projeto, conforme descrito no Capítulo 3. Como ainda não há a especificação do Sistema de Regras, o tratamento de Agrupamentos será apenas parcialmente suportado neste protótipo inicial, assim como as referências a esquemas; entretanto, a manipulação de tipos de entidade, de relacionamentos n-ários, papéis e instâncias deverá ser totalmente suportada. Após a implementação deste protótipo, há duas linhas de desenvolvimento que poderão evoluir paralelamente.

A primeira envolve o Sistema de Regras, cuja especificação permitirá não só extrair de Agrupamentos toda a sua potencialidade na representação de objetos mais complexos como também poderá englobar os Subsistemas de Integridade, Consistência e Segurança (estes aspectos poderiam ser também especificados como regras do Sistema). É interessante ressaltar que as regras GERPAC são definidas como qualquer outro atributo de um objeto, de forma que a transferência das regras da base de dados da aplicação para o Sistema de Regras se resume a uma operação normal de consulta à base de dados. Deste modo, a definição de regras é plenamente suportada pelo módulo central GERPAC.

A outra linha de trabalho refere-se à manipulação automática de mais de uma base de dados, como forma a permitir o suporte total a conexão entre esquemas representando diferentes níveis do processo de projeto (a representação do macro-aspecto). Para tanto, há de se explorar a característica do SIGA que permite a manipulação simultânea de diversas bases de dados controladas através do seu Dicionário e Diretório.

Com relação à linguagem de consulta para SGBD, é importante destacar o trabalho já realizado no sentido de se propor uma linguagem gráfica para a definição e consulta de bases

de dados GERPAC. Deste trabalho, LIGG [/QUE87/], já existe um protótipo desenvolvido em um Sistema VAX utilizando recursos baseados no núcleo gráfico GKS [/HAR86/]. O desenvolvimento completo desta linguagem, utilizando outros tipos de recursos no sentido de agilizar sua utilização, constitui uma outra linha de trabalho futuro.

A implementação dos protótipos do SIGA e do GERPAC vem sendo realizada em linguagem C. A máquina em utilização é um sistema microcomputador compatível com IBM-PC-XT, com Winchester de 40 Mbytes e 640 Kbytes de memória principal, o que é satisfatório para os desenvolvimentos iniciais.

O desenvolvimento obtido por grupos de trabalho na área de arquiteturas não convencionais de processadores, utilizando por exemplo multiprocessamento [/TOZ87/], e outros trabalhos na linha de processadores dedicados a bancos de dados e a inferências, permite abrir uma outra linha de pesquisa no sentido de se desenvolver máquinas GERPAC e/ou máquinas SIGA, permitindo satisfazer os requisitos de eficiência não atingidos por máquinas convencionais, como a manipulação simultânea de diversos esquemas, do Sistema de Regras e do Dicionário do Sistema.

Capítulo 7

BIBLIOGRAFIA

/ANT80/ de Antonellis, V. *et alii*: "Extending the Entity-Relationship Approach to Take into Account Historical Aspects of Systems"; *in* /CHE80/.

/AST76/ Astrahan, M. M. *et alii*: "System R — Relational Approach to Database Management"; ACM Transactions on Database Systems, 1(2), 1976.

/ATZ81/ Atzeni, P. *et alii*: "INCOD: A System for Conceptual Design of Data and Transactions in the Entity-Relationship Model"; *in* /CHE81/.

/BAR82/ Baron, N. *et alii*: "An Approach to the Integration of Geometrical Capabilities into a Database for CAD Applications"; *in* /ENC82/.

/BAT80/ Batini, C. *et alii*: "Top-Down Design in the Entity-Relationship Approach"; *in* /CHE80/.

/BEN81/ Benneworth, R. L. *et alii*: "The Implementation of GERM, an Entity-Relationship Data Base Management System"; Proc. 7th. International Conference on Very Large Data Bases, 1981.

/BER76/ Berg, J. L. (Ed.): "Data Base Directions — The Next Steps"; Proc. Workshop of National Bureau of Standards and the ACM, Setembro 1976.

/BER83/ Berztiss, A. T.; Thatte,S.: "Specification and Implementation of Abstract Data Types"; Advances in Computers 22, Academic Press, 1983.

/BLA81/ Blasgen, M. W. *et alii*: "System R — An Architectural Overview"; IBM Systems Journal, 20(1), 1981.

/BRO86/ Brodie, M. L.: "Integrating Artificial Intelligence and Database Technologies"; V Escola de Computação, Belo Horizonte, 1986.

/CAL81/ Caldiera, G.; Quitadamo, P.: "Conceptual Representation of Data and Logical IMS Design"; *in* /CHE81/.

/CAS86/ Casanova, M. A.: "Uma Introdução a Bancos de Dados Dedutivos"; *in* /MEI 6/.

/CHA80/ Chang, S. K.; Cheng, W. H.: "A Methodology for Structured Database Decomposition"; IEEE Transactions on Software Engineering, 6(2), 1980.

/CHA82/ Challis, M. F.: "Typing in Data Base Models"; *in* /ENC82/.

/CHE76/ Chen, P. P.-S.: "The Entity-Relationship Model — Toward a Unified View of Data"; ACM Transactions on Database Systems, 1(1), 1976.

/CHE77/ Chen, P. P.-S.: "The Entity-Relationship Approach to Logical Data Base Design"; The Q.E.D. Monograph Series No. 6, 1977.

/CHE80/ Chen, P. P.-S. (Ed.): "Entity-Relationship Approach to Systems Analysis and Design"; Proc. 1st. International Conference on Entity-Relationship Approach, North-Holland, 1980.

/CHE81/ Chen, P. P.-S. (Ed.): "Entity-Relationship Approach to Information Modelling and Analysis"; Proc. 2nd. International Conference on Entity-Relationship Approach, ER Institute, 1981.

/CHE81b/ Chen, P. P.-S.: "A Preliminary Framework for Entity-Relationship Models"; *in* /CHE81/.

/CHI81/ Chiang, T. C.; Rose, G.R.: "Design and Implementation of an E-R Data Base Management System"; *in* /CHE81/.

/CLA82/ Clark, K. L.; Tärnlund, S. (Eds.): "Logic Programming"; Academic Press, 1982.

/COD70/ Codd, E. F.: "A Relational Model of Data for Large Shared Data Banks"; Communications of ACM, 13(6), 1970.

/COS84/ Costa Martins, M. A.: "Concepção duma Base de Dados"; Editora Rés (Portugal), 1984.

/DAH83/ Dahl, V.: "Logic Programming as a Representation of Knowledge"; IEEE Computer, Outubro 1983.

/DAT83/ Date, C. J.: "An Introduction to Database Systems — Volume II"; Addison-Wesley, 1983.

/DAT84/ Date, C. J.: "Introdução a Sistemas de Bancos de Dados"; Editora Campus, 1984.

/DAV82/ Davis, R. E.: "Runnable Specification as a Design Tool"; *in* /CLA82/.

/DAV83/ Davis, C. G. *et alii*. (Eds.): "Entity-Relationship Approach to Software Engineering"; Proc. 3rd. International Conference on Entity-Relationship Approach, North-Holland, 1983.

/DAV85/ Davis, R. E.: "Logic Programming and PROLOG: A Tutorial"; IEEE Software, Setembro 1985.

/DE 81/ De, P. *et alii*: "Four-Schema Approach: an Extended Model for Database Architecture"; Information Systems, 6(2), 1981.

/DEL86/ Delgado, A. L. N.; Magalhães, L. P.: "Assimilando a Semântica de P.A.C. através de um Modelo de Dados"; *in* /FER86/.

/DEL86b/ Delgado, A. L. N. *et alii*: "Sistemas de Gerenciamento de Banco de Dados para P.A.C."; *in* /MEL86/.

/DEL87/ Delgado, A. L. N.: "Bancos de Dados no Contexto de PAC"; Dissertação de Mestrado, FEE/UNICAMP, Janeiro 1987.

/DOL86/ Dolenc, A.: "Um Gerenciador de Banco de Dados para Aplicações em Projeto de Circuitos Integrados"; *in* /MEL86/.

/EAS81/ Eastman, C. M.: "Database Facilities for Engineering Design"; Proceedings of the IEEE, 69(10), 1981.

/ENC80/ Encarnação, J.: "Computer Aided Design Modelling, Systems Engineering, CAD- Systems"; Lecture Notes in Computer Science 89, Springer-Verlag, 1980.

/ENC82/ Encarnação, J; Krause, F. (Eds.): "File Structures and Data Bases for CAD"; North-Holland, 1982.

/ENC83/ Encarnação, J. Schlechtendahl, E. G.: "Computer Aided Design — Fundamentals and System Architectures"; Springer-Verlag, 1983.

/FER86/ Fernandes, C. J. G. (Ed.): "Anais do VI Congresso da Sociedade Brasileira de Computação"; SBC, Recife, 1986.

/FOI82/ Foisseau, J.; Valette, F. R.: "A Computer Aided Design Data Model: FLO-REAL"; *in* /ENC82/.

/FRE80/ Freeman, P; Wasserman, A. I. (Eds.): "Tutorial on Software Design Techniques"; IEEE Computer Society, 1980.

/FRY76/ Fry, J. P.; Sibley, E. H.: "Evolution of Data-Base Management Systems"; ACM Computing Surveys, 8(1), 1976.

/GAL76/ Gall, R. M.: "What Experience has Taught Us"; *in* /BER76/.

/GAL84/ Gallaire, H. *et alii*: "Logic and Databases: A Deductive Approach"; Computing Surveys, 16(2), 1984.

/GRA82/ Grabowski, H.; Eigner, M.: "A Data Model for a Design Data Base"; *in* /ENC82/.

/GUT77/ Guttag, J.: "Abstract Data Types and the Development of Data Structures"; Communications of the ACM, 20(6), 1977.

/HAR86/ Hartelt, H. I. M.: "GAV — Um Gerenciador de Áreas de Visualização Baseado na Norma GKS"; Dissertação de Mestrado, FEE/UNICAMP, Dezembro de 1986.

/KAT81/ Katz, R. H.; Goodman, N.: "View Processing in MULTIBASE, a Heterogeneous Database System"; *in* /CHE81/.

/KLO81/ Klopprogge, M. R.: "TERM: An Approach to Include the Time Dimension in the Entity-Relationship Model"; *in* /CHE81/.

/LOR82/ Lorie, R. A.: "Issues in Database for Design Applications"; *in* /ENC82/.

/MEL86/ Melo, R. N. (Coord.): "Anais do I Simpósio Brasileiro de Banco de Dados"; SBC, Rio de Janeiro, 1986.

/MIN76/ Minsky, N.: "On Interaction with Data Bases"; *in* /RUS76/.

/NEU80/ Neumann, T.: "CAD Databases Requirements and Architectures"; *in* /ENC80/.

/NEW81/ Newman, W. M.; Sproull, R. F.: "Principles of Interactive Computer Graphics"; McGraw-Hill, 1981.

/PAS86/ Passos, E. L. (Coord.): "Anais do III Simpósio Brasileiro de Inteligência Artificial"; SBC, Rio de Janeiro, 1986.

/PAT82/ Patnaik, L. M.; Ramesh, N.: "Implementation of Interactive Relational Graphics Database"; Computer & Graphics, 6(3), 1982.

/QUE87/ Quezada Gonzales, S.: "LIGG – Uma Lingnagem Gráfica baseada em Grafos para o Modelo Entidade-Relacionamento – PAC"; Dissertação de Mestrado, FEE/UNICAMP, Janeiro de 1987.

/RIC85/ Ricarte, I. L. M. *et alii*: "Mapeamento do Modelo Entidade Relacionamento Binário sobre CORAS"; Relatório Interno, DCA/FEE/UNICAMP, Março de 1985.

/RIC86/ Ricarte, I. L. M.: "Definição e Implementação de um Modelo de Dados para o Núcleo CORAS-UNICAMP"; Relatório de Atividades 2, FAPESP Proc. 84/2591-1, Março de 1986.

/RIC86b/ Ricarte, I. L. M.: "Definição e Implementação de um Modelo de Dados para o Núcleo CORAS-UNICAMP"; Relatório de Atividades 3, FAPESP Proc. 84/2591-1, Agosto de 1986.

/RUS76/ Rustin, R. (Ed.): "Workshop on Data Description, Access and Control"; ACM SIGMOD, 1976.

/SAK83/ Sakai, H.: "Entity-Relationship Approach to Logical Database Design"; *in* /DAV83/.

/SAN80/ dos Santos, C. S. *et alii*: "A Data Type Approach to Entity-Relationship Model"; *in* /CHE80/.

/SCH80/ Scheuerman, P. *et alii*: "Abstraction Capabilities and Invariant Properties Modelling within the Entity-Relationship Approach"; *in* /CHE80/.

/SET85/ Setzer, V. W.: "Projeto Lógico e Projeto Físico de Bancos de Dados"; V Escola de Computação, Belo Horizonte, 1986.

/SIL86/ da Silveira, P. M.: "O Uso de uma Máquina de Inferências em Bancos de Dados"; *in* /PAS86/.

/SMI79/ Smith, D. C. P.; Smith, J. M.: "Relational Data Base Machines"; IEEE Computer, 12(3), 1979.

/SMI80/ Smith, D. C. P.; Smith, J. M.: "Conceptual Database Design"; *in* /FRE80/.

/SMI82/ Smith, J. M.; Smith, D. C. P.: "Principles of Database Conceptual Design"; *in* /YAO82/.

/STO76/ Stonebraker, M. *et alii*: "The Design and Implementation of INGRES"; ACM Transactions on Database Systems, 1(3), 1976.

/STO83/ Stonebraker, M. *et alii*: "Application of Abstract Data Types and Abstract Indices to CAD Data Bases"; Proceedings on Engineering Design Applications of ACM-IEEE Database Week, Maio 1983.

/TEO82/ Teorey, T. J.; Fry, J. P.: "Design of Database Structures"; Prentice-Hall Inc., 1982.

/TOZ87/ Tozzi, C. L.: "Arquiteturas para Multiprocessamento utilizadas no Laboratório de Sistemas Mini-Microcomputadores do DCA/FEE/UNICAMP (Título Provisório)"; II Encontro de Pesquisadores do Programa de Cooperação Brasil Argentina, SEI-SADIO, Tandil, Fevereiro de 1987.

/TSI77/ Tsichritzis, D. C.; Lochovsky, F. H.: "Data Base Management Systems"; Academic Press, 1977.

/TSI82/ Tsichritzis, D. C.; Lochovsky, F. H.: "Data Models"; Prentice-Hall Inc., 1982.

/ULF82/ Ulfsby, S. *et alii*: "TORNADO: A DBMS for CAD/CAM Systems"; *in* /ENC82/.

/VER84/ Vernadat, F. B.: "A Commented and Indexed Bibliography on Data Structuring and Data Management in CAD/CAM: 1970 to Mid-1983"; National Research Council of Canada No. 23373, Março 1984.

/VET82/ Vetter, M.; Maddison, R. N.: "Database Design Methodology"; Prentice-Hall International Inc., 1982.

/WIE83/ Wiederhold, G.: "Database Design"; McGraw-Hill International Book Company, 1983.

/WIL76/ Willians, R.; Giddings, G. M.: "A Picture Building System"; IEEE Transactions on Software Engineering, 2(1), 1976.

/WU 83/ Wu S.-T.: "Implementação de um Núcleo de Banco de Dados baseado na Filosofia de CORAS"; Documentação Preliminar, DCA/FEE/UNICAMP, 1983.

/WU 84/ Wu S.-T.: "MERB — Um Modelo de Dados para Aplicação em Controle de Processos"; Dissertação de Mestrado, FEE/UNICAMP, Setembro 1984.

/YAO82/ Yao, S. B. *et alii* (Eds.): "Data Base Design Techniques I — Requirements and Logical Structures"; Lecture Notes in Computer Science 132, Springer-Verlag, 1982.