SysKBZ$^+$, um Protótipo de SGBDOO Baseado no Modelo KBZ$^+$

Marco Toranzo Céspedes *Fernando da Fonseca de Souza*

Alexandre Marcos Lins Vasconcelos *Hednilson de Almeida Bezerra*

Fred Monteiro da Cruz Filho *Breno Gustavo Soares da Costa*

Universidade Federal de Pernambuco

Departamento de Informática

Caixa Postal 7851

CEP 50732-970,

Recife, PE, Brasil

E-mail: {matc,fdfd,amlv,hed,fmcf,bgsc}@di.ufpe.br

ABSTRACT

This work presents a description of the data model KBZ$^+$ as well as aspects of the construction of an OODBMS, called SysKBZ$^+$, which implements the data model KBZ$^+$ while combining the funcionality of OODBMS with RDBMS.

Such aspects are object identifiers and keys, data representation in both main and secondary memories and the system architecture. The system prototype was implemented in VisualWork/Smalltalk of the ParcePlace, under the Unix Operating System.

1 Introdução

Durante as três décadas passadas, a tecnologia de banco de dados passou por quatro gerações, e uma quinta geração está atualmente em desenvolvimento [Kim90a, BM93]: baseadas em arquivos, modelo hierárquico, modelo rede, modelo relacional e a abordagem orientada a objetos. A transição de uma geração para outra foi em resposta a requerimentos mais complexos dos usuários, como consequência do rápido incremento na performance e confiabilidade, acoplado com um dramático decréscimento em suas medidas e custos.

Os modelos de dados dos SGBDs da próxima geração deverão prover novas características não presentes nos sistemas prévios [Cat91, Cat94, Kim90a, BM93, Kim90b]. Atualmente, existem ao menos duas abordagens propostos para a transição da quarta tecnologia de banco de dados para a quinta: tecnologia de banco de dados relacional estendida e tecnologia de banco de dados orientada a objetos (não necessariamente divergentes). A quinta geração de tecnologia de banco de dados está sendo representada por uma mistura de produtos comerciais e protótipos de pesquisa. Alguns produtos industrais são: ObjetcStore [LLOW91] da ODI e Gemstone [BOS91] da Servio Corporation. Entre os protótipos industriais têm-se: Orion [KBC+87] da MCC e Iris [LHW90] da Hewlett-Parckard. Entre os protótipos de universidades podem ser citados: Postgres da universidade da California, ENCORE/Observer da universidade de Brown, e ultimamente, SysKBZ$^+$ (monousuário) [Tor95] da Universidade Federal de Pernambuco, o qual implementa o modelo de dados KBZ$^+$ [Tor95].

As principais contribuções deste trabalhos são as seguintes:

1. Apresentar o estado de avanço na construção de SysKBZ$^+$ desde [TFMS95];
2. Fornecer uma visão geral dos conceitos do modelo KBZ$^+$ e interface gráfica de SysKBZ$^+$;
3. Apresentar aspectos de implementação do protótipo SysKBZ$^+$;
4. Apresentar uma discusão com respeito à implementação de identificadores de objetos e chave primária em SysKBZ$^+$.

Este trabalho está organizado como segue. A seção 2 apresenta uma descrição geral do modelo KBZ$^+$. A seção 3 apresenta a arquitetura do sistema, implementação de um modelo, identificadores de objetos e chaves de objetos, além da representação de informação

nas memórias principal e secundária. Finalmente, a seção 4 apresenta as conclusões e direções de futuros trabalhos.

2 Descrição do Modelo KBZ$^+$

Um modelo de dados é um conjunto de conceitos e regras que permitem a modelagem de problemas de certos domínios. Os modelos de dados dos SGBDs da quinta geração deverão prover novas características além das existentes nos sistemas tradicionais. Um caso particular desses modelos é o modelo KBZ$^+$ que é uma extensão do modelo KBZ [OI88, Sou92, Sou93]. A notação empregada para expressar os conceitos do modelo KBZ$^+$ está influenciada pela notação da linguagem Z [Spi89]. Maiores detalhes sobre KBZ+ e suas diferenças e semelhanças com Z e KBZ, podem ser encontradas em [Tor95].

Algumas das principais características do modelo KBZ$^+$ são:

1. Captura aspectos estáticos e de comportamento dos objetos;
2. Introduz o conceito de relacionamento identificador [Sou92] no modelo, que é semelhante ao conceito de chave primária do modelo relacional original [Cod70];
3. Permite o usuário perceber e modelar um micromundo como constituído de objetos;
4. Sua notação é um subconjunto da linguagem Z, mas com diferente semântica;
5. Contrário a Z, o modelo KBZ$^+$ realiza um tratamento *implícito* do conceito de *estado*;
6. Introduz conceito do modelo relacional (restrição do domínio);
7. Introduz o esquema da linguagem Z como seu mecanismo de modularização;
8. A semântica do reuso de esquema em Z é modificada para tratamento da herança.

A seguir são apresentados os conceitos do modelo KBZ$^+$.

• **Entidade** - Uma entidade é um objeto atômico único com existência independente, empregada para representar um objeto concreto ou abstrato do mundo modelado, tal como evento, projeto, pessoa, instituição, etc. Definir uma entidade é uma forma de introduzir um tipo que pode ser empregado para definir outros tipos. Entidades são equivalentes a *Given Sets* na linguagem Z. A seguir é apresentada a forma de definir entidades em KBZ$^+$.

[EVENTO, PROJETO, PESSOA, INSTITUÇÃO, TEMÁTICA , TRABALHO]

• **Atributos** - Uma entidade é significativamente descrita por seus atributos [Hug88]. Em KBZ+, um atributo é um objeto que possui um nome, um tipo e, opcionalmente, uma propriedade funcional sobre suas instâncias que permite capturar restrições de implementação. Por exemplo, o esquema *salário*, mostrado adiante, define o atributo de tipo real com valores entre 100 e 4000. Desta forma o modelo garante a integridade do domínio de um atributo.

• **Relacionamentos** - Um relacionamento é um objeto complexo composto de objetos (entidades, atributos). Sua especificação captura toda a informacão de uma associação (grau, cardinalidade, os objetos associados, etc). O seguinte exemplo define o relacionamento *TEM_SALARIO* que age sobre os objetos PESSOA e SALARIO. A palavra "PREDICADO" representa um conjunto de métodos do objeto (comportamento).

SALARIO
salário: String
salario ∈ [100..4000]

TEM_SALARIO
salário : PESSOA → SALARIO
salário_inv : SALARIO → PESSOA
PREDICADO

• **Entidade Agregada** - Uma entidade agregada é um objeto complexo virtual composto de objetos relacionamentos. É virtual porque esse objeto não manipula suas próprias instâncias. O maior benefício deste conceito, além de possuir comportamento, é manipular um conjunto de relacionamentos como uma unidade lógica. Entidades agregadas podem se sobrepor umas às outras. A parte declarativa do esquema seguinte declara os *objetos relacionamentos* que compõem a entidade agregada *ENT_PESSOA*; a parte predicado apresenta a definição do método *Create* cuja função é instanciar os objetos relacionamentos.

ENT_PESSOA
RCPF RNOME RLOCAL RPERÍODO
Create(cod_cpf?: COD_CPF; nome?: NOME; período?: PERÍODO; local?: LOCAL; message!: String) (cod_eve? ∈ ran fcod ⇒ message!='Instance already exist') ∧ (cod_eve? ∉ ran fcod ⇒ (∃ evento: new(EVENTO) | evento ∉ dom fcod • ((fcod' = fcod ∪ {evento ↦ cod_eve?}) ∧ (fnome' = fnome ∪ {evento ↦ nome?}) ∧ (fperíodo' = fperíodo ∪ {evento ↦ período?}) ∧ (flocal' = flocal ∪ {evento ↦ local?}) ∧ message!='Ok'))) end

• **Subtipo** - Um subtipo pode ser definido através de relacionamentos e/ou uma entidade agregada acompanhada de uma restrição que define a condição de subtipo. A estrutura de um subtipo é análoga à de uma entidade agregada.

3 Uma Descrição de SysKBZ$^+$

Nesta seção apresentamos vários tópicos relacionados com a construção de SysKBZ$^+$, tais como, a arquitetura, identificação de objetos, representação de objetos nas memórias e implementação dos conceitos de KBZ$^+$.

3.1 A Arquitetura de SysKBZ$^+$

A arquitetura de SysKBZ$^+$ está composta basicamente de três nivéis: O gerenciador de acesso, o gerenciador de objetos e o gerenciador de armazenamento. A figura 1 apresenta a arquitetura do sistema mostrando os principais subsistemas que compõem cada um dos nivéis de SysKBZ$^+$. O gerenciador de acesso controla o uso do sistema contra pessoas não autorizadas; cada usuário do SysKBZ$^+$ é identificado por um *login* e *password*. O gerenciador de objetos é responsável pelo controle sobre o esquema do banco de dados e instanciação dos objetos do esquema e requerer serviços de recuperação/gravação de página de objetos. O gerenciador de armazenamento é responsável pela recuperação/gravação de página de objetos. SysKBZ$^+$ também possui uma interface gráfica para usuários visando facilitar o acesso desses usuários para definir e manipular aplicações.

A construção de SysKBZ$^+$ implementa vários *modelos* de janelas com seus respectivos gerenciadores baseando-se na arquitetura MVC de Smallaltalk [ABB92]. Os principais benefícios disto são: modularidade e fraco acoplamento entre as classes para representar e gerenciar instâncias. A figura 2 b) apresenta esses modelos; a figura 2 a) apresenta as classes que empregam os modelos, por exemplo, as classes AppEntModel, AppAttModel e AppRelModel empregam as classes EntityEntryBox, AttributeEntryBox e RelationshipEntryBox, respectivamente.

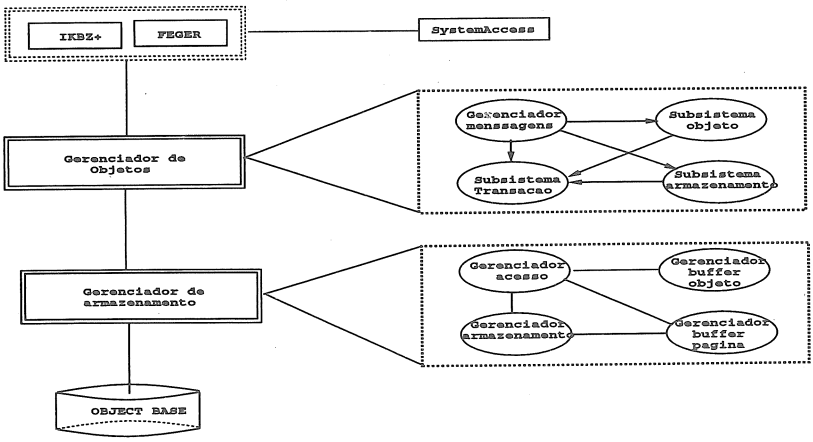


Figura 1: A Arquitetura de SysKBZ$^+$ e seus Principais Subsistemas

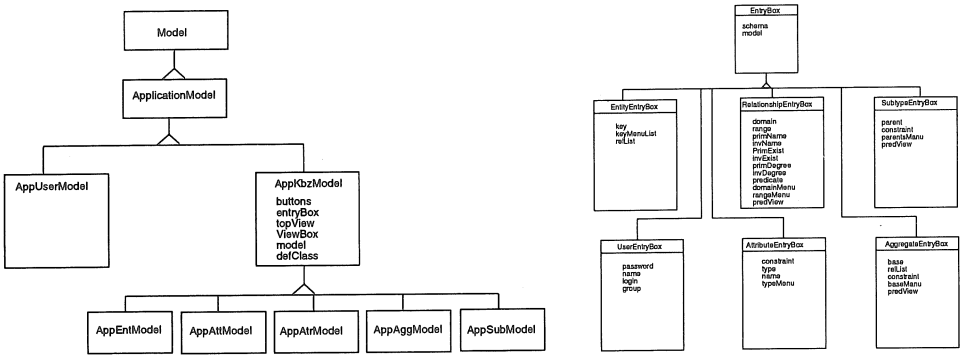
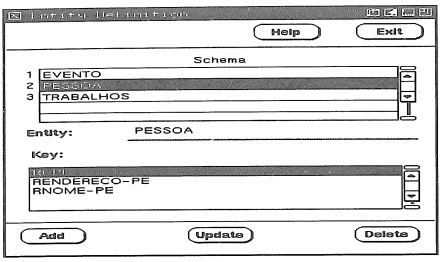


Figura 2: a) Manipuladores das Janelas b) Modelo de Janelas usando MVC

As figuras 3, 4 e 5 apresentam algumas das diferentes janelas controladas pelas classes da figura 2 b). Por exemplo, os objetos EntityEntryBox, AttributeEntryBox, RelationshipEntryBox, AggregateEntryBox e SubtypeEntryBox controlam as janelas *Entity Definition, Attribute Definition, Relationship Definition, Aggregate Definition* e *Subtype Definition*, respectivamente. A figura 5 b) forma parte do gerenciador de objetos.



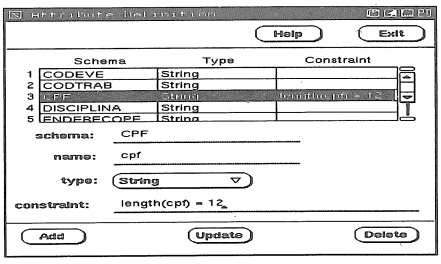
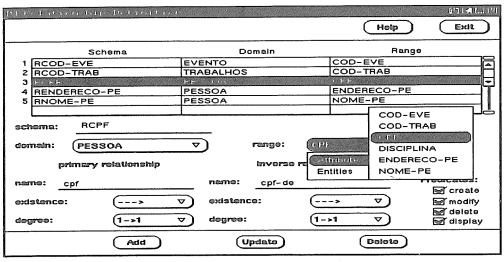


Figura 3: a) Definição de Entidade b) Definição de Atributos



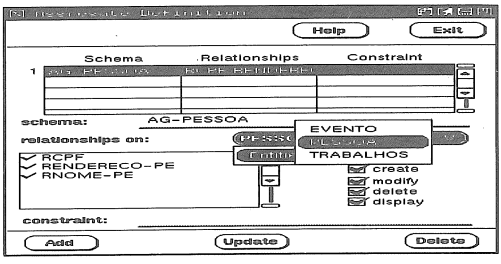
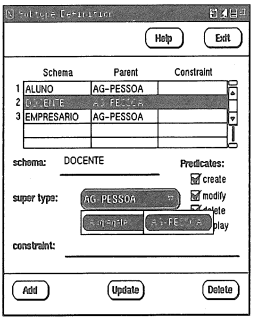


Figura 4: a) Definição de Relacionamentos b) Definição de Entidade Agregada



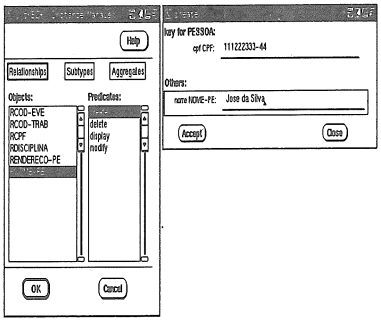


Figura 5: a) Definição de Subtipos b) Instanciação de objetos

3.2 Implementação dos Conceitos do Modelo KBZ$^+$

A figura 6 apresenta a estrutura hierárquica de classes que implementam os conceitos do modelo KBZ$^+$ no sistema. Um esquema de banco de dado é um conjunto de instâncias

dessas classes. Cada classe possui *métodos de instâncias básicos* para modificar/acessar seus valores de estados. As definições das classes das figura 2 b) e 6 são semelhantes, mas de funcionalidades diferentes.

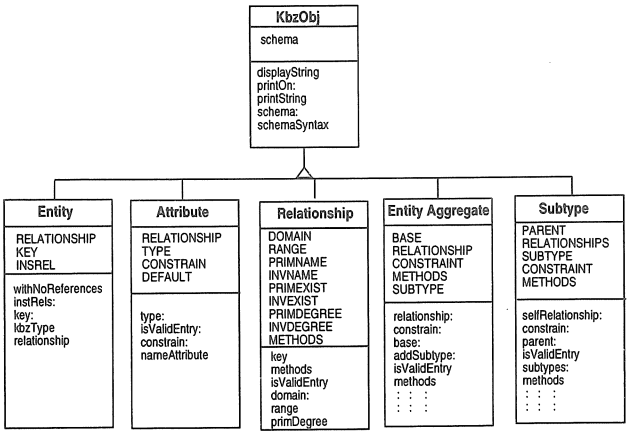


Figura 6: Implementação do Modelo KBZ$^+$

3.3 Representação de um Esquema em SysKBZ$^+$

A figura 7 a) apresenta três classes que representam um modelo de um esquema conceitual e o controle de acesso ao sistema. A classe *KbzModel* é empregada para armazenar em cada uma de suas variavéis de instância (coleções homogênias) ponteiros para instâncias das classes da figura 6. O usuário ingressa a definição de um conceito KBZ$^+$ através de uma classe da figura 2 b), a qual após verificada, o sistema envia uma menssagem para criar uma instância de um conceitos KBZ$^+$ (figura 6) e passa para esta última a definição do usuário em termos de dados (nome do objeto) e ponteiros para outro objetos. Cada instância das classes da figura 6 é referenciada por um ponteiro que é armazenado numa das variavéis de instância de *KbzModel*; a variável *verify* é um flag (true/false) que indica se a definição do esquema conceitual foi verificada.

A classe *ObjModel* armazena a informação para armazenar/recuperar um esquema conceitual para/do disco. Com respeito aos atributos relevantes dessa classe, pode-se afirmar

que pemitem ao sistema:

1. Guardar o último identificador de objetos empregado para a definição de um objeto;
2. Armazenar os nomes do diretório e esquema de banco de dados;
3. Conhecer se a definição do modelo conceitual foi modificada.

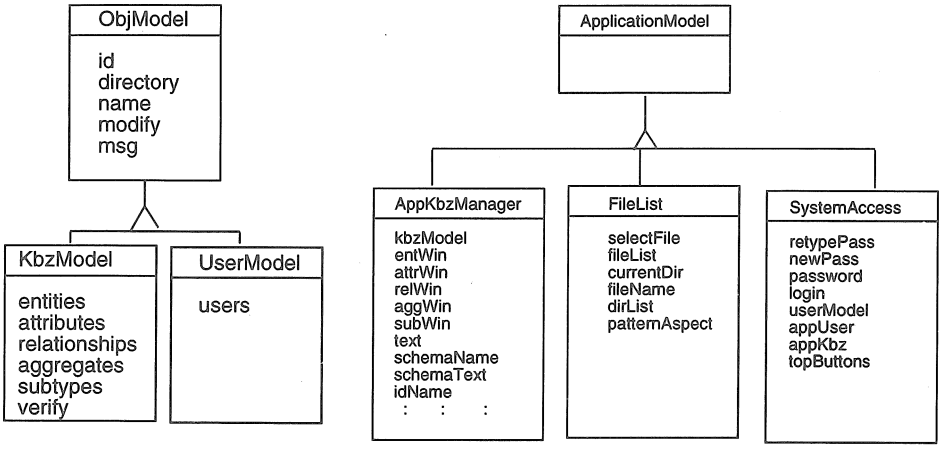


Figura 7: a) Modelo do Esquema Conceitual b) Gerenciamento de 7 a)

A classe *UserModel* armazena toda a informação relacionada com os usuários do sistema (login, password, group, etc) necesária para o controle de acesso ao sistema (ver figura 8 a)). Cada classe da figura 7 b) é instanciada somente uma vez na execução do sistema. Uma instância da classe *AppKbzManager* conterá todas as definições e instâncias dos objetos relacionamentos. É importante destacar que as entidades agregadas e subtipos não manipulam suas próprias instâncias, esses objetos gerenciam as instâncias dos relacionamentos (através de seus identificadores) sobre os quais foram construídos. A instância de classe *SystemAccess* é empregada para controlar o acesso ao sistema.

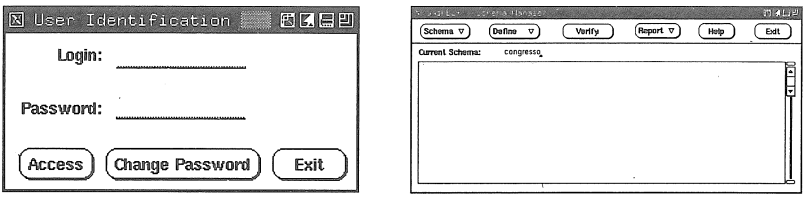


Figura 8: a) Controle de Acesso b) Janela Principal de SysKBZ+

A seguir, a figura 9 apresentada uma visão mais detalhada da relação entre *AppKbzManager* e *KbzModel* no gerenciamento e representação de um esquema conceitual. Nessa figura, as elipses representam instâncias de classes, e as setas representam ponteiros para instâncias.

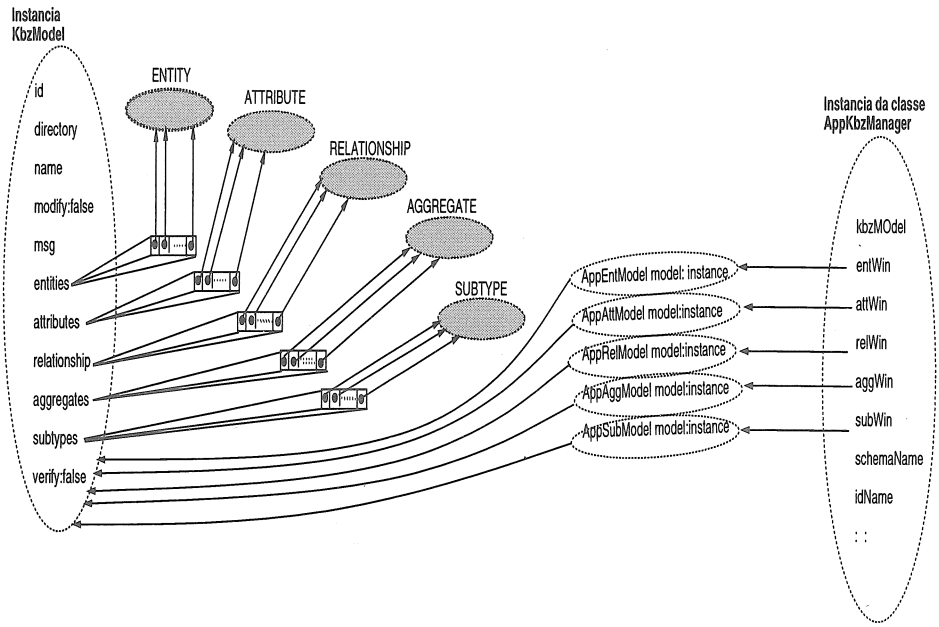


Figura 9: Representação de um Modelo Conceitual na Memória Primária

Quando o sistema é executado, então é gerada uma instância da classe *AppKbzManager* cujas variáveis de instâncias (kbzModel, entWin, attrWin, etc) contêm instâncias de outras classes, por exemplo, as variáveis *kbzModel* e *entWin* contêm uma instância das classes *KbzModel* e *AppEntModel*, respectivamente (ver figuras 2 a) e 7 a)).

3.4 Representação de Dados nas Memórias Primária e Secundária

A figura 10 apresenta uma outra abstração relacionada com a manipulação dos dados do banco de dados. Esta figura apresenta uma visão do conteúdo da variável *entities*, que é uma coleção de ponteiros para instâncias da classe *ENTITY*. Cada uma dessas instâncias possui uma organização que é representada pelo retângulo maior nessa figura.

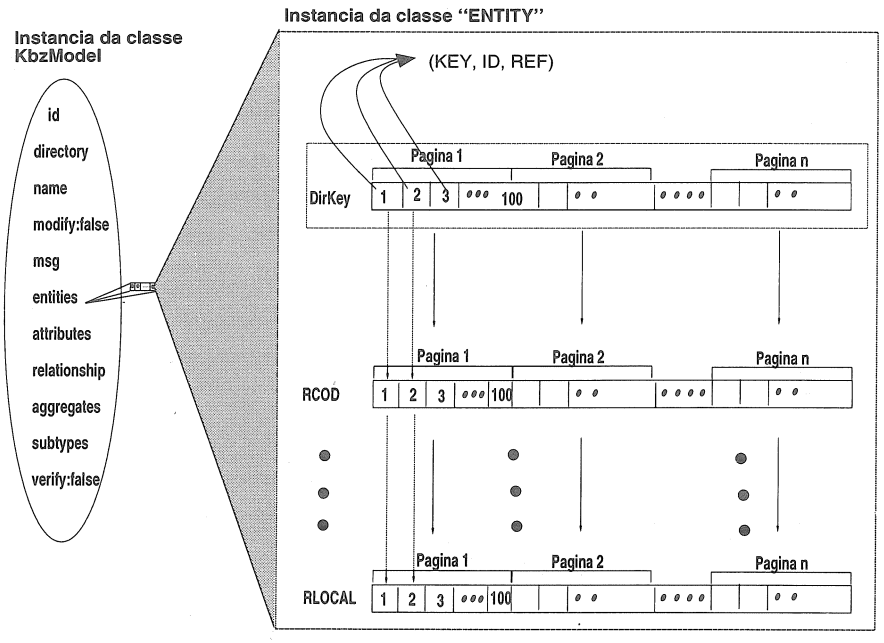


Figura 10: Representação de Dados nas Memórias Primária e Secundária

Cada instância da classe *entity* possui um diretório de chaves (DirKey) e um conjunto de coleções que representam cada um dos relacionamentos definidos no esquema conceitual. O objetivo do diretório, além de permitir identificar as objetos pela valor da chave primária, é estabelecer uma correspondência bidirecional entre a chave primária e um identificador do sistema. Esse diretório é um objeto cujos principais serviços são :

- Incluir/apagar/modificar/deletar uma chave do diretório;
- Retornar um identificador do sistema (surrogate) associado com uma chave primária;
- Retornar o contador de referência para uma instância;

Referências

[ABB92] Brian. Alexander, Adony. Benaire, and Rick. Berman. *ObjectWork Smalltalk.* ParcPlace Systems, 1992.

[BM93] E. Bertino and L. Martino. *Object-Oriented Database Systems: Concepts And Architectures.* Addison-Wesley, 1993.

[BOS91] Paul Butterworth, Allen Otis, and Jacob Stein Stein. The gemstone database management system. *ACM*, 34(10):50–63, October 1991.

[Cat91] R.G.G. Catell. Next-generation database systems. *ACM*, 34(10):31–33, October 1991.

[Cat94] R.G.G Cattell. *Object data management: object-oriented and extended relational.* Addison-Wesley, 1994.

[Cod70] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 6(13):377–387, June 1970.

[Hug88] J. G. Hughes. *Database Technology: A Software Engineering Approach.* Prentice-Hall, 1988.

[KBC+87] W. Kim, N. Ballou, H. T. Chou, J. Garza, and D. Woelk. Features of the orion object-oriented database systems. In Won. Kim and F. Lochovsky, editors, *Object-oriented Concepts, Databases, and Applications*, pages 371–394. Addison-Wesley, February 1987.

[Kim90a] Won Kim. Introduction to object-oriented database systems. *MIT press*, 1990.

[Kim90b] Won Kim. Object-oriented databases: Definition and research - directions. *IEEE Transactions on Knowledge and data Engineering*, 2(3):327–341, 1990.

[LHW90] P. Lyngbaek, W. Hasan, and K. Wilkinson. The iris architecture and implementation. *IEEE Transactions on Knowledge and Data Engineering*, 2(1):63–75, March 1990.

[LLOW91] Charles Lamb, Gordon Landis, Orenstein, and Dan Weinreb. The objectstore database system. *ACM*, 34(10):50–63, October 1991.

[OI88] E. Oxborrow and H. M. Ismail. Kbz-an object-oriented approach to the specification and management of knowledge bases. In *6th BNCOD*, pages 21–46, Cambridge University Press, 1988. University of Kent at Canterbury.

[Sa94] Filho Marum Simão. Uma ferramenta gráfica de apoio ao projeto de banco de dados. Master's thesis, Universidade Federal de Pernambuco, March 1994.

[Sou92] Fernando da F. de Souza. *A Platform for Implementing Object-Oriented Database Based on KBZ.* PhD thesis University of Kent at Canterbury, 1992.

[Sou93] Fernando da F. de Souza. Object-oriented databases and the kbz approach. In *VIII Simpósio Brasileiro de Banc de Dados*, May 1993.

[Spi89] J. M. Spivey. *The Z Notation : a reference manual.* Prentice Hall, UK, 1989.

[TFMS95] Marco Toranzo, Hednilson Fonseca, Fernando snd Bezerra, Fred Monteiro, and Breno Soarez. Syskbz: Um sgb baseado no modelo kbz. *IX Brazilian Symposium on Sofware Engineering*, pages 451–454, October 1995.

[Tor95] Marco. A. Toranzo. Syskbz+, um protótipo de sgbdoo baseado no modelo kbz+ e desenvolvido num ambiente (programação orientada a objetosa{, year = 1995. Master's thesis, Universidade Federal de Pernambuco, Novemb 1995.

- Verificar a existência de uma instância;
- Associar um valor de chave primária com um identificador interno do sistema de forma transparente para o usuário;
- Todos os objetos são identificados por valores do usuário ou por identificadores do sistema, mas não ambos.

O diretório de chaves e as coleções que armazenam as instâncias dos diferentes relacionamentos estão logicamente divididos em páginas. Cada página contém 25 unidades lógicas. O diretório possui uma coleção de triplas (KEY,ID,REF). **KEY** é uma chave de usuário; **ID** é um identificador do sistema (surrogate) associado com *KEY*; **REF** é um contador de referências empregado para manter a integridade referencial do banco de dados. Só os objetos com REF=0 podem ser eliminados.

No diretório é armazenada a informacão para gerenciar os objetos, enquanto que nas coleções (relacionamentos) ficam os dados que descrevem as diferentes instâncias de uma classe. A organização e acesso ao banco de dados é sequencial, mas está sendo mudada para B-tree. A figura 7 apresenta a organização de um banco de dados gerenciado por SysKBZ$^+$.

4 Conclusões e Futuros Trabalhos

Neste trabalho foi apresentada uma visão geral do protótipo de SysKBZ$^+$, além de uma descrição simplificada do modelo KBZ$^+$, cuja sintaxe está fundamentada sobre a linguagem Z, expressando uma modelagem em termos matemáticos (tipos, funções, etc). Também foi abordado aspectos de implementação relacionados com representação do modelo KBZ$^+$ e representacão nas memórias primária e secundária do esquema conceitual e dos dados. O protótipo do sistema foi implementado utilizando-se Visualwork/Smalltalk R. 4.1 [ABB92] sob o sistema operacional Unix [1]. Futuros trabalhos deverão estar orientados a otimizar a organização e acesso aos dados, criar uma interface para FEGER [Sa94] que é um editor gráfico que suporta o modelo KBZ, revisar toda a performance do sistema, e terminar o compilador para a sintaxe do modelo KBZ$^+$.

[1] Unix é marca registrada de AT&T