MODELAGEM DE DADOS DE APLICAÇÕES NÃO CONVENCIONAIS: UM ESTUDO DE CASO

Cláudio Newton Ferreira Trotta *Guilherme Horta Travassos* *Jano Moreira de Souza*

UFRJ/COPPE Sistemas
Cidade Universitária - CT - sala H-319
Caixa Postal 68511 - CEP 21945 - Rio de Janeiro - RJ - Brasil
Telefones: (5521) 290-4540/ 590-2552/ 590-2652
Telex: (5521) 33817 UFCO BR - Fax: (5521)290-6626
E-mail: COS90001@UFRJ.BITNET

SUMÁRIO

Este trabalho descreve a modelagem conceitual de um conjunto de ferramentas gráficas para a especificação de sistemas, reunidas num ambiente integrado denominado FEGRES. Os requisitos de tal ambiente são apresentados, e uma extensão do modelo de Entidades e Relacionamentos que permite herança de restrições de integridade é usada para representar o esquema conceitual da ferramenta.

1. Introdução

Este artigo descreve a modelagem dos dados manipulados por algumas das ferramentas do ambiente FEGRES [TRAVASSOS 90a, 90b]. Apesar da notação e metodologia empregadas serem bem conhecidas e de largo uso para aplicações convencionais, o interesse aqui reside no fato das aplicações em questão (ferramentas para desenvolvimento de software) não serem o que se costuma chamar de aplicações convencionais.

O ambiente FEGRES compõem-se, atualmente, de três ferramentas distintas para apoio à fase de especificação de requisitos: o Editor de Diagrama de Fluxo de Dados (DFD), o Editor de Diagramas de Entidades-Relacionamentos (DER) e o Dicionário de Dados Automatizado (DD), devidamente estendido para suportar a documentação dos objetos manipulados pelo editor de DER. Estas ferramentas foram desenvolvidas a partir dos métodos definidos por [GANE 79] e [CHEN 76].

Dentre os vários requisitos de projeto que envolveram a construção do FEGRES, dois devem ser destacados por representarem as características mais marcantes do ambiente. O primeiro deles é o que diz respeito à integração do ambiente, obtida através da utilização de uma base de dados única, e o segundo é o que trata do

auxílio ao usuário no desenvolvimento da especificação: FEGRES deve auxiliar ao projetista na geração do maior número de informações contextuais.

Assim, por exemplo, ao se construir um determinado diagrama de fluxo de dados, a ferramenta deve trazer, para o usuário, toda a sintaxe e, se possível, toda a semântica envolvida no método. Desta forma, por exemplo, ao explodir-se um processo, uma nova tela é exibida mostrando o processo explodido em tamanho grande, acompanhado, automaticamente, dos fluxos de dados que nele entram e saem, e, ainda, mostrando os elementos na outra ponta dos fluxos. Ao desenhar os elementos dentro da explosão do processo, o projetista pode selecionar um destes fluxos e reconectá-los em um elemento interno ao processo explodido.

Outro exemplo é a chamada automática que é realizada a uma ficha do Dicionário de Dados quando se inclui um elemento no DFD. Novamente, alguns dados das fichas são automaticamente preenchidos com informções introduzidas no DFD, pois a base de dados é a mesma.

Outra característica importante é a inclusão (ou a remoção), no DFD, de outras representações gráficas para um elemento que já foi definido previamente. O sistema mantém sempre uma visão consistente para o usuário.

Mais informações sobre o ambiente FEGRES podem ser obtidas em [TRAVASSOS 90a, 90b].

O sucesso da implementação se deveu em grande parte à modelagem conceitual de dados que foi realizada na fase de concepção do sistema, permitindo a integração via base de dados. Este trabalho tem como objetivo relatar esta experiência de modelagem. Limitamo-nos, porém, a relatar a modelagem do Dicionário de Dados e do Diagrama de Fluxos de Dados, por restrições de espaço.

A seção 2 aborda rapidamente o problema da modelagem em aplicações não convencionais. Na seção 3, os requisitos do projeto da ferramenta são introduzidos, e o esquema conceitual da ferramenta é apresentado usando-se o modelo de entidades e relacionamentos, com extensões. Na seção 4 são apresentados alguns detalhes de implementação, e a seção 5 conclui o trabalho.

2. O Problema de Modelagem em Aplicações Não Convencionais

Para modelar uma ferramenta que implementa um método da análise estruturada, é necessário um cuidadoso estudo dos objetos que constituirão o esquema conceitual subjacente, ainda que os blocos construtores deste método sejam conceitualmente simples e pouco numerosos. Isto se deve, em parte, pelas regras que regem o método que o instrumento apoia e, em parte, pela qualidade da interface homem-máquina fornecida pela ferramanta. Por exemplo: muitos métodos são compostos de técnicas que permitem ao usuário trabalhar em diversos níveis de abstração, passando por estes níveis em um processo de refinamentos sucessivos. Isto sugere, naturalmente, uma estrutura de dados recursiva, onde cada nível de abstração corresponde a um nível de recursão. Por outro lado, manter representações visuais e diálogos coerentes com o objeto sendo construído (no

estilo: o-que-você-vê-é-o-que-você-tem) exige a construção de estruturas de dados adicionais, bem como a definição de regras de integridade que possibilitem manter estes dados da representação gráfica dos objetos compatíveis com os dados do objeto propriamente dito.

Desta forma, alguns dos problemas que surgem para modelar conceitualmente uma ferramenta como FEGRES são típicos do que tem sido chamado de "aplicações não convencionais".

Historicamente, a modelagem conceitual de aplicações não convencionais se confunde com a criação de modelos, onde a ação ou o método de perceber os objetos do mundo real e aprisioná-los em um esquema formal ou semi-formal não é explicitado, isto é, até então tem havido muito investimento em modelos e não no processo de modelagem. Alguns modelos apresentados na literatura pretendem possuir um nível de abstração tão perto da realidade que a tarefa de modelagem se reduziria a descrever o mundo real utilizando os construtores do modelo. De qualquer forma, o processo de modelagem é ainda um exercício de raciocínio e entendimento da realidade que é atingido de uma forma mais ou menos caótica. Para modelar uma ferramenta que implementa um método da Engenharia de Software, um grau a menos de dificuldade se apresenta: não existe a figura do usuário (com todos os fatores humanos) entre a realidade e o sujeito que modela. Os requisitos são mais facilmente obtidos pois fazem parte da definição do instrumento que a ferramenta implementa.

Diversas linhas de pesquisa, com objetivos aparentemente diferentes, têm contribuído na direção de modelos que possam mais facilmente ser usados em aplicações não convencionais [TROTTA 90].
São elas:

. extensões realizadas em SGBDs que implementam o modelo relacional;
. modelos de dados semânticos;
. o paradigma da orientação a objetos;
. sistemas de gerência de banco de dados extensíveis.

Apesar do esforço da pesquisa, notamos a falta de um SGBD que pudesse se aproximar dos requisitos necessários para lidar com os objetos do FEGRES. Mesmo a modelagem destes objetos se ressentiu de um paradigma que pudesse descrever adequadamente os objetos, seu complicado interelacionamento e as simples operações do usuário que internamente envolviam múltiplas atualizações em diversos sub-objetos e o comprometimento com a integridade imposta pelos métodos que as ferramentas implementam.

Apesar das limitações do modelo relacional, resolveu-se desenvolver um SGBD, especialmente para o FEGRES, baseado neste modelo. O sistema em si não apresenta grandes novidades em relação aos anteriormente desenvolvidos pela equipa da COPPE/UFRJ [MATTOSO 87, SOUZA 88], exceto em ser compacto e rápido, dada a sua especifidade. À medida que novas ferramentas forem sendo incorporadas ao FEGRES, seus requisitos deverão ser atendidos por um novo sistema, com arquitetura não relacional, em desenvolvimento. Não obstante, o trabalho de modelagem aqui apresentado não invalida os requisitos mínimos que este novo sistema deve apresentar.

Da solução "ad hoc" de implementação seguiu a decisão de usar o MER com extensões para capturar o

aspecto estrutural dos dados. Restrições de integridade e operações sobre os objetos foram descritas informalmente, uma vez que a implementação destas se tornaram puro código na linguagem C, num nível de abstração muito abaixo do que desejávamos. Na próxima seção, onde apresentamos o esquema conceitual das ferramentas, utilizamos o MER. Algumas extensões são apresentadas, à medida que a modelagem do problema as motivam.

3. A Modelagem de FEGRES

Nesta seção abordamos os requisitos de projeto do ambiente FEGRES e descrevemos a modelagem do Dicionário de Dados e do Diagrama de Fluxo de Dados. O esquema conceitual correspondente à integração das duas ferramentas é obtido paulatinamente.

3.1 O Dicionário de Dados.

Quando analisamos os elementos que compõem o Diagrama de Fluxos de Dados (DFD) da Análise Estruturada, notamos que as principais entidades manipuladas pelo método são aquelas que existem como fichas do Dicionário de Dados: processos, entidades externas, fluxos, depósitos, estruturas e elementos de dados. A entidade fluxo desempenha um papel fundamental pois é ela que liga todas as outras entidades, mantendo coeso o diagrama. A consistência de um diagrama depende da coerência criada a partir dos fluxos de dados e dos elementos e estruturas de dados que eles carregam, garantindo que qualquer dado chegue no ponto onde deve ser processado e que nenhum dado fique estagnado no sistema.

Uma primeira versão do diagrama de entidades e relacionamentos (DER) é fornecida pela figura 1. Neste, notamos que a entidade FLUXOS representa o elemento de ligação entre as diversas outras entidades do diagrama. Os relacionamentos ESi (Entra-Sai) entre FLUXOS, PROCESSOS, ENTIDADES_EXTERNAS e DEPÓSITOS representam o fato que destas três últimas entidades podem entrar (e/ou sair) diversos FLUXOS. O relacionamento EXPLOSÃO modela o fato que um processo pode ser explodido em diversos outros de um nível inferior, de uma maneira hierárquica. Finalmente, os diversos (inter) relacionamentos (GUARDA, CONSTRõEM, COMPõEM, TRANSPORTA, CARREGA e ARMAZENA) existentes entre FLUXOS, DEPÓSITOS, ESTRUTURAS_DADOS e ELEMENTOS_DADOS representam o fato que fluxos e depósitos de dados são formados de elementos e/ou estruturas de dados, estas últimas sendo elas próprias formadas de estruturas e/ou elementos de dados mais simples.

O diagrama mostrado na figura 1 tem o objetivo de dar uma visão inicial da modelagem do problema, sendo bastante incompleto em dois sentidos:

i) O diagrama está semanticamente pobre, não sendo incluída adequadamente a cardinalidade dos relacionamentos e diversas outras restrições de integridade;

ii) Não foram considerados diversos requisitos de projeto, alguns devido à definição do próprio método e

outros devido à interface homem-ferramenta.

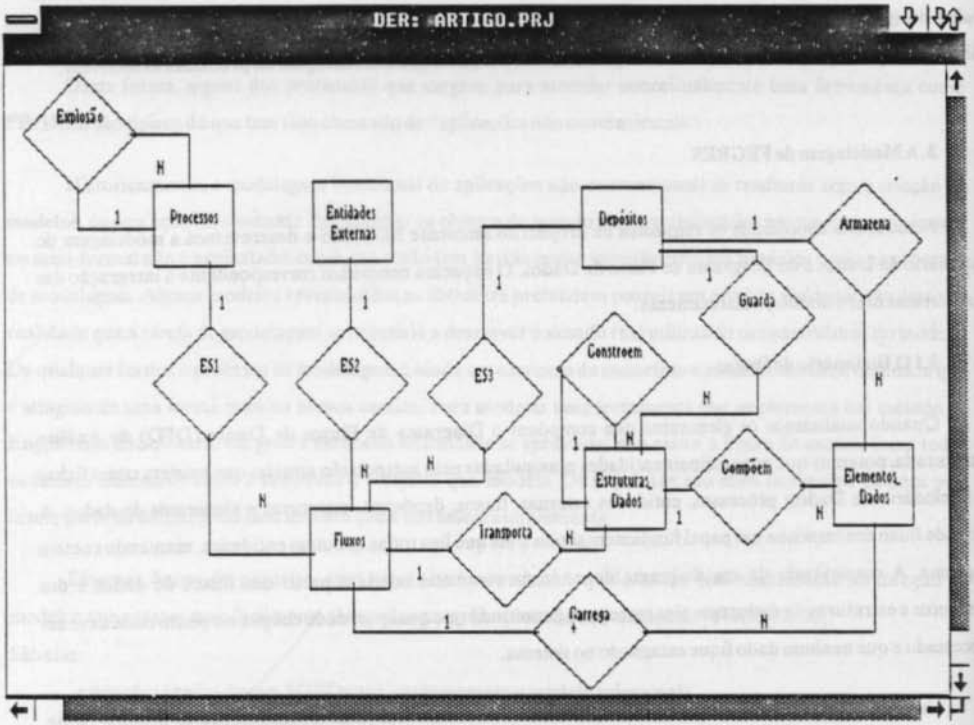

Figura 1

3.2 O Diagrama de Fluxos de Dados.

Diversos são os requisitos de projeto do editor de DFD de FEGRES. Alguns deles são relacionados com a definição do método, outros foram incorporados como imposições da interface. Estes requisitos são [TRAVASSOS 90a]:

Processos:

1.1 Todo processo possui um criador;

1.2 Não pode existir mais de um processo com o mesmo criador e a mesma identificação;

1.3 Todo processo possui pelo menos um fluxo entrando e um saindo;

1.4 Mover graficamente um processo, implica em mover os fluxos a ele associados;

1.5 Remover um processo implica em marcar os fluxos associados como inválidos, remover todas as suas explosões (filhos) e remover sua ficha no dicionário de dados. Um processo só pode ser removido no nível de explosão em que foi criado;

1.6 Um processo pode aparecer repetidamente no DFD, em 3 situações distintas: no nível onde é criado, explodido (visualizando-se seus filhos) e externo a uma explosão (como informação contextual);

1.7 Explodir um processo implica em apresentar todos os elementos internos e externos à explosão;

1.8 O somatório dos elementos de dados que chegam através de fluxos têm que ser igual ao somatório dos elementos de dados que saem.

Entidades Externas (EE):

2.1 Toda EE possui uma identificação e só pode ser criada no nível de contexto;

2.2 Uma EE pode aparecer repetidamente em um DFD no mesmo nível, ou em outro, como informação contextual;

2.3 Mover graficamente uma EE implica em mover todos os fluxos a ela associados;

2.4 Remover uma EE implica em marcar os fluxos associados como inválidos. Caso a representação gráfica da EE removida seja a última, é necessário remover a ficha da EE do Dicionário de Dados;

2.5 Uma EE não pode ser representada dentro da área de explosão de um processo;

2.6 Uma EE deve estar ligada a pelo menos um fluxo de dados.

Depósitos de Dados:

3.1 Todo depósito possui um criador e uma identificação;

3.2 Um depósito pode aparecer repetidamente em um DFD no mesmo nível, ou em outro, como informação contextual;

3.3 As informações contidas em um depósito têm que ser aquelas que chegam através dos fluxos que entram;

3.4 Todo depósito possui pelo menos um fluxo entrando e um saindo;

3.5 Mover graficamente um depósito implica em mover todos os fluxos a ele associados;

3.6 Remover um depósito implica em marcar os fluxos associados como inválidos. Caso a representação gráfica do depósito removido seja a última, é necessário remover a ficha do depósito do Dicionário de Dados;

3.7 Um depósito não pode ser representado dentro da área de explosão de um processo;

3.8 Um depósito interno a um processo não pode ser representado fora da área de explosão deste processo.

Fluxos de Dados:

4.1 Todo fluxo tem um criador e um nome;

4.2 Não podem ocorrer fluxos distintos com o mesmo nome;

4.3 Um fluxo pode aparecer repetidas vezes no DFD (em uma mesma tela) com o mesmo nome mas com origem e destino diferentes. Pode aparecer com a mesma origem e mesmo destino (em telas diferentes) no caso de explosões de processos ainda não detalhadas;

4.4 Mover o elemento origem/destino de um fluxo implica em mover todos os vértices do fluxo, com

exceção do destino/origem;

4.5 Todo fluxo deve ter uma de suas pontas ligadas a um processo, a outra pode estar ligada a qualquer tipo de elemento;

4.6 Um fluxo tem 1 e somente 1 origem, e, pelo menos, um destino;

4.7 Um fluxo não pode entrar e sair do mesmo processo.

A partir destes requisitos, notamos uma forte interação entre os elementos que constituem o DFD e a representação gráfica destes. O DFD é essencialmente um instrumento gráfico, assim, na sua definição original, já existiam facilidades para sua elaboração, como, por exemplo, permitir a multiplicação da representação gráfica de depósitos de dados e entidades externas. É o caso dos requisitos 2.1, 2.2, 2.5, e 3.2. Outros requisitos são pertencentes puramente ao método (1.1, 1.2, 1.3, 1.8, 2.6, 3.1, 3.3, 3.4, 4.1, 4.2, 4.5, 4.6 e 4.7). Finalmente, alguns são específicos de FEGRES e do nível de facilidade que se pretendeu fornecer ao usuário (1.4, 1.5, 1.6, 1.7, 2.3, 2.4, 3.5, 3.6, 3.7, 3.8, 4.3, e 4.4). Estes últimos permitem o uso do ambiente no estilo relatado na seção 1.

Todos estes requisitos geram estruturas, operações e restrições adicionais. A figura 1, na verdade, representa a modelagem considerando-se apenas o Dicionário de Dados. Quando se acrescentam estes requisitos de manipulação gráfica e integram-se as visões, obtem-se um DER representado pela figura 2. Cada relacionamento ESi existente na figura 1 é dividido em dois, permitindo uma melhor modelagem do problema. Estes relacionamentos tornam-se de fato relacionamentos entre as representações gráficas dos elementos. Com a restrição de integridade que todo o elemento existente no Dicionário de Dados deve possuir pelo menos uma representação gráfica, i.e., deve aparecer pelo menos uma vez no DFD, podemos substituir os relacionamentos entre FLUXOS, PROCESSOS, ENTIDADES_EXTERNAS e DEPÓSITOS pelos relacionamentos entre suas representações gráficas. Atributos típicos daquelas entidades são os que aparecem no Dicionário de Dados, como nome e identificação. Atributos típicos das representações gráficas (entidades REP_PROC, REP_FLUX, REP_EE e REP_DEP) incluem as coordenadas da tela onde são exibidas.

3.3 Extensões ao DER e a Modelagem Final do DFD.

A notação utilizada para representar o Diagrama de Entidades e Relacionamentos apresentado na figura 2 apresenta algumas extensões, em relação a definição básica apresentada por [CHEN 76], a saber:

i) as cardinalidades dos relacionamentos são expressas em valores máximo e mínimo;

ii) representação da abstração de Categoria [EL-MASRI & NAVATHE 89] com extensões definidas neste trabalho;

iii) restrições de integridade definidas segundo o trabalho de [SETZER 86];

Para o entendimento completo do diagrama da figura 2, descrevemos a seguir a abstração de Categoria a partir de uma alternativa de modelagem deste problema.

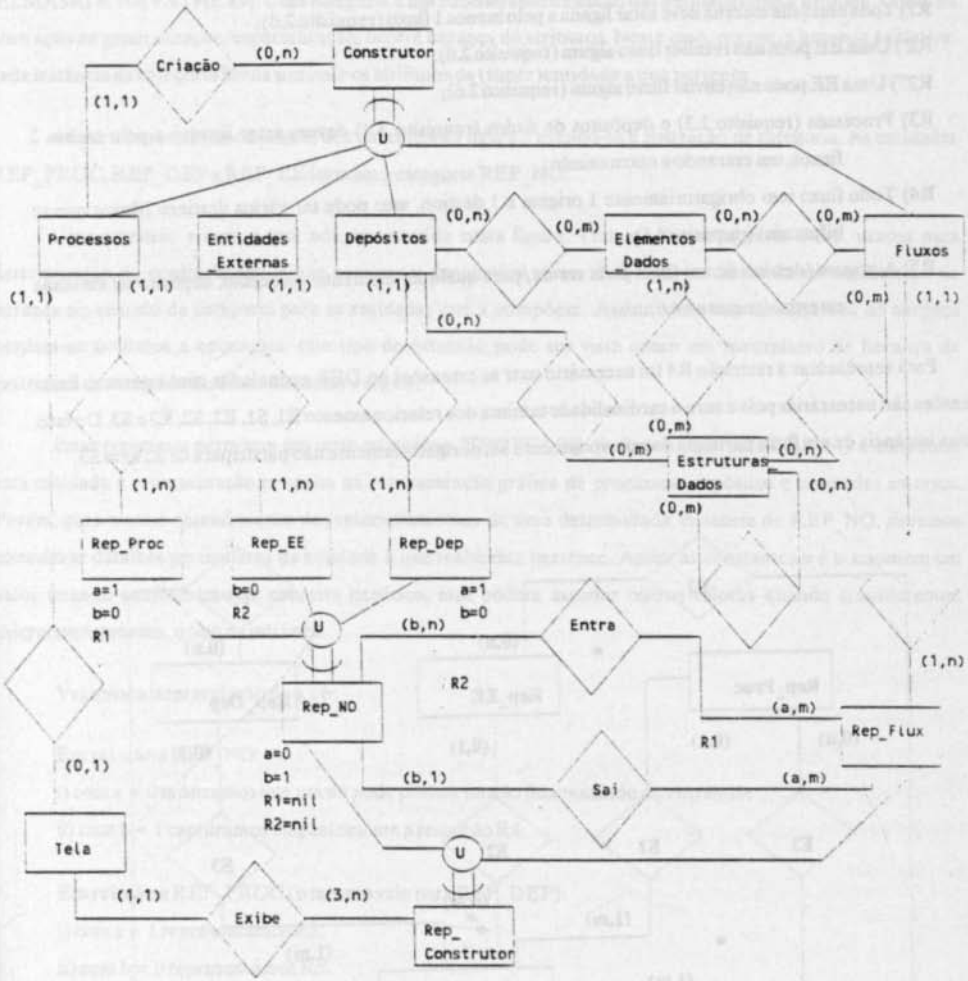

Figura 2

Suponhamos a figura 3. Neste diagrama, o DER é estendido usando-se uma notação proposta em [SETZER 86]. Um arco com um círculo ligando dois relacionamentos significa que uma instância da entidade (a entidade mais próxima do arco) deve participar obrigatoriamente de 1 dos dois relacionamentos, podendo participar dos dois (ou inclusivo). Se o círculo for cortado, então a instância deve participar de 1 - e no máximo 1 - dos relacionamentos (ou exclusivo).

Esta extensão e as cardinalidades mínimas e máximas apresentadas, permitem capturar as seguintes restrições:

R1) Todo fluxo deve ter pelo menos 1 ligação com um processo (requisito 4.5);

R2) Toda entidade externa deve estar ligada a pelo menos 1 fluxo (requisito 2.6);

R2') Uma EE pode não receber fluxo algum (requisito 2.6);

R2") Uma EE pode não enviar fluxo algum (requisito 2.6);

R3) Processos (requisito 1.3) e depósitos de dados (requisito 3.4) devem estar ligados a pelo menos 2 fluxos, um entrando e outro saindo;

R4) Todo fluxo tem obrigatoriamente 1 origem e 1 destino, mas pode ter vários destinos (fluxos que se bifurcam) (requisito 4.6);

R5) A origem/destino de um fluxo pode ser de/para qualquer elemento (processo, depósito ou entidade externa) (requisito 4.6).

Para representar a restrição R4 foi necessário usar as extensões no DER assinaladas com asterisco. Estas extensões são necessárias pois é zero a cardinalidade mínima dos relacionamento E1, S1, E2, S2, E3 e S3. De fato, se uma instância de um fluxo participa do relacionamento S1, obrigatoriamente não participará de S2 nem S3.

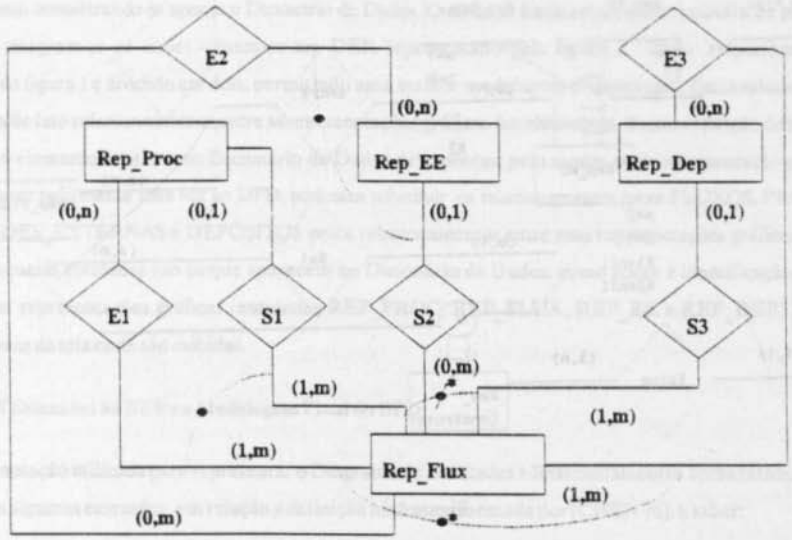

Figura 3

Esta restrição tão simples do modelo foi representada de uma forma complicada porque REP_PROC, REP_DEP e REP_EE são entidades aparentemente independentes, mas no fundo representam os nós de uma rede cujos arcos são fluxos.

Para modelarmos mais naturalmente esta situação, lançamos mão de uma abstração denominada categoria

[ELMASRI & NAVATHE 89]. Uma categoria é um subconjunto da união das entidades que a formam. Como na abstração de generalização/especialização, ocorre herança de atributos. Neste caso, porém, a herança é seletiva: cada instância da categoria herda somente os atributos da (super)entidade a que pertence.

A figura 2 mostra a modelagem dos conceitos da figura 3 usando-se a abstração de categoria. As entidades REP_PROC, REP_DEP e REP_EE formam a categoria REP_NO.

Uma extensão a mais é, por nós, introduzida nesta figura. Trata-se das constantes a e b, usadas para determinação da cardinalidade. Estas constantes funcionam como atributos, participando no mecanismo de herança no sentido da categoria para as entidades que a compõem. Assim como nos mecanismos de herança herdam-se atributos e operações, este tipo de extensão pode ser visto como um mecanismo de **herança de restrições de integridade**, e será discutida a seguir.

Estas constantes permitem um certo relativismo [TROTTA 90]: podemos olhar para REP_NO e considerar esta entidade a representação genérica da representação gráfica de processos, depósitos e entidades externas. Porém, para o total entendimento dos relacionamentos de uma determinada instância de REP_NO, devemos considerar detalhes particulares da entidade a que realmente pertence. Assim as constantes a e b assumem um valor quando consideramos o conceito genérico, mas podem assumir outros valores quando consideramos, microcospicamente, o tipo da instância.

Vejamos a interpretação de a e b:

Em relação a REP_NO:

i) com a = 0 mostramos que um nó pode possuir ou não fluxos saindo ou entrando;

ii) com b = 1 capturamos elegantemente a restrição R4.

Em relação a REP_PROC (o mesmo vale para REP_DEP):

i) com a = 1 representamos R3;

ii) com b = 0 representamos R5.

Em relação a REP_EE:

i) com a = 0 (herdado) representamos R2' e R2";

ii) com b = 0 representamos R5.

A restrição R1 não é resolvida puramente com o uso de constantes. Podemos, porém, definir que R1 é nula para a categoria REP_NO, mas, na entidade REP_PROC, assume o seguinte valor (usando-se a notação de [SETZER 86] onde oui significa ou inclusivo):

$$R1 = \nexists f \in REP\_FLUX [\ \exists p \in REP\_PROC [(f,p) \in ENTRA\ ]\ oui\ \exists p' \in REP\_PROC [(f,p') \in SAI]\ ] \\ p <> p'$$

Adicionalmente, com o predicado " p < > p' " incluímos em R1 a restrição:

R6) Uma instância de um fluxo não pode ter origem e destino em um mesmo processo (requisito 4.7).

De forma similar, definimos R2 como nula para a categoria REP_NO, mas com o seguinte valor para REP_EE:

R2 = ∀ e ∈ REP_EE [∃ f ∈ REP_FLUX [(e,f) ∈ ENTRA] ou
∃ f ∈ REP_FLUX [(e,f) ∈ SAI]]

Na figura 2, a abstração de categoria é usada mais duas vezes. A entidade CONSTRUTOR representa a união das entidades que representam os construtores básicos de um DFD, a saber: PROCESSOS, ENTIDADES_EXTERNAS, DEPÓSITOS e FLUXOS, enquanto a entidade REP_CONSTRUTOR tem o mesmo papel quando se considera suas representações gráficas.

O relacionamento EXPLOSÃO da figura 1 foi substituído pelo relacionamento CRIAÇÃO na figura 2. A razão disto é melhor representar o fato que um CONSTRUTOR é sempre criado como filho de um processo, isto é, quando se explode um processo em outros, um nível abaixo, também podem ser criados, além de processos, novos fluxos e depósitos. Esta modelagem, porém, não é completa em relação a entidades externas, pois não representa o requisito 2.1.

Uma nova entidade - TELA - também está incluída na figura 2. Para cada processo que foi explodido existe uma correspondência 1:1 com uma instância de TELA. Seus atributos típicos representam características gráficas como cor de fundo e posicionamento do diagrama em relação à janela, entre outros. Uma TELA por sua vez, EXIBE ao menos 3 representações gráficas de construtores. Isto é devido ao fato de um processo estar ligado a pelo menos dois fluxos, e estes por sua vez, estão ligados a pelo menos um nó (restrições R3 e R4).

3.4 Operações e Restrições de Integridade

Algumas restrições de integridade foram expressas no DER da figura 2. O modelo, porém, não tem poder de representar diversas outras. Algumas destas são as expressas pelos requisitos 1.6, 2.5, 3.2, 3.7 e 3.8. Estas restrições envolvem, em sua maioria, o posicionamento de representações gráficas na tela, podendo ser traduzidas como valores de coordenadas gráficas (atributos) válidos em um determinado instante ou contexto. Outras restrições não representadas dizem respeito ao método (1.8 e 3.3).

As operações sobre os elementos do DFD podem ser definidas a partir dos requisitos 1.4, 1.5, 1.7, 2.3, 2.4, 3.5, 3.6 e 4.4. Estas operações envolvem a manipulação de diversas instâncias do modelo.

As operações definidas para cada elemento e as restrições de integridade encontram-se completamente definidas no trabalho de [TRAVASSOS 90a].

Pela incapacidade do SGBD de FEGRES suportar a definição de operações e restrições de integridade na

base de dados, o acesso aos elementos do DFD no FEGRES é feito por um protocolo onde a base é acessada somente através de funções pré-definidas, as quais implementam as operações chamando primitivas do banco e incorporam, em si próprias, os testes de integridade.

4. Implementação da Base de Dados do Ambiente FEGRES

A implementação do banco de dados foi uma tarefa à parte no ambiente FEGRES. A utilização do MS/WINDOWS trouxe a necessidade de se adotar características específicas no desenvolvimento do banco de dados, de maneira a explorar todos os recursos do ambiente.

O gerente de interface MS/WINDOWS cria um ambiente multitarefa sob sistemas operacionais DOS [MICROSOFT 88], além de provocar uma série de modificações no enfoque tradicional de programação. O ambiente possui um objeto principal, denominado janela ("Window"), o qual tem o relevante papel de ser principal objeto de manipulação no WINDOWS. Janelas se comportam como entidades isoladas, possuindo características próprias e executando tarefas distintas entre si. O controle destas janelas é feito através de um dispositivo de troca de mensagens, que pode ser considerado a base de toda a máquina WINDOWS.

Aplicações devem ser desenvolvidas especificamente para rodar sob o ambiente WINDOWS. Existem basicamente dois tipos de aplicações: uma aplicação propriamente dita e uma biblioteca dinâmica. A diferença principal entre uma biblioteca e uma aplicação é que a primeira não executa por si, necessitando que uma aplicação faça o controle para ela. Bibliotecas dinâmicas executam em conjunto com as aplicações e suas funções são ligadas em tempo de execução e não em tempo de compilação.

O banco de dados do FEGRES foi construído como uma biblioteca dinâmica. Isto facilitou a inclusão de uma série de características importantes, das quais valem a pena serem destacadas:

1. O conceito de transação: as ferramentas do ambiente utilizam a base de dados através de transações bem definidas que são realizadas quando do acesso à base de dados. As operações são pré-definidas, na forma de funções, e as ferramentas se utilizam somente destas funções para acessar a base;

2. O conceito de esquema: devido a ter sido criado um banco de dados genérico, a definição da base de dados é feita através de um esquema que contém a definição das várias relações, juntamente com seus atributos componentes, proporcionando a facilidade do usuário acessar esta base de dados, fora do ambiente FEGRES (vide figura 4);

3. Concorrência: Cada ferramenta funciona como uma aplicação independente, acessando concorrentemente a base. Para evitar conflitos de acesso, foi usado o próprio esquema de filas de mensagens do MS/WINDOWS para bloquear a base durante a execução de uma transação;

4. Linguagem de Consulta: toda a consulta à base é feita através de uma linguagem de consulta própria, baseada no modelo relacional. As funções mencionadas acima utilizam esta linguagem para acessar a base. Esta

linguagem também está disponível fora do ambiente FEGRES (figura 4).

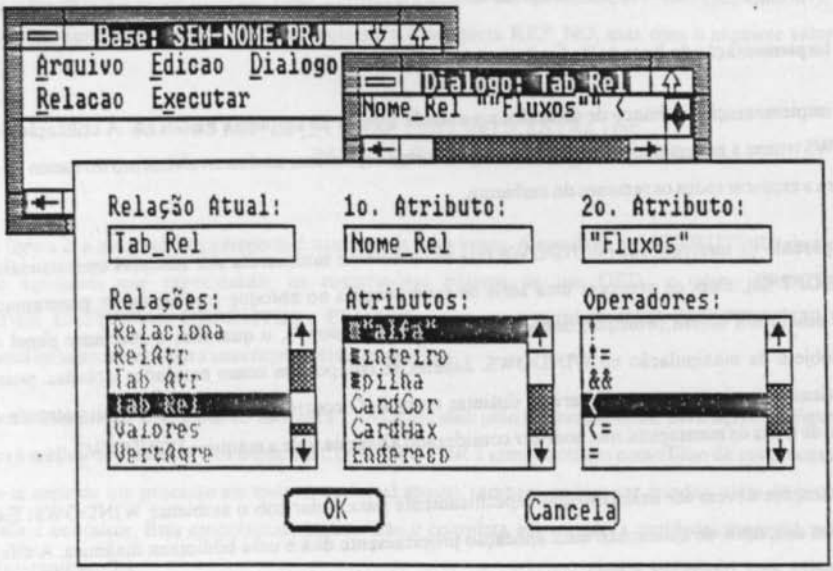

Figura 4

A implementação do DER da figura 2 no modelo relacional pode ser encontrada em [TRAVASSOS 90a, TROTTA 90].

5. Conclusões

Da necessidade do trabalho de modelagem conceitual de FEGRES, ficou claro a falta de um modelo conceitual que pudesse representar, adequadamente, todos os requisitos de projeto, mais ainda, a falta de um SGBD adequado, dificultando, assim, tanto a concepção quanto a implementação do sistema. Diversas restrições de integridade não puderam ser capturadas, como mostrado na seção 3.4. Outras restrições surgiram devido ao complexo interelacionamento entre as entidades e relacionamentos, principalmente quando se fecham "círculos" no DER.

O diagrama da figura 2 representa um estado consistente do DFD e do DD. Algumas restrições de integridade podem (e devem) ser relaxadas durante a construção do DFD (por exemplo, para atender aos requisitos 1.5, 2.4 e 3.6 ou para permitir a existência de fichas no DD sem a correspondente representação no DFD). Tal procedimento é devido a falta do conceito de transações longas, que permitiriam um estado inconsistente durante uma seção de projeto.

Do exercício de modelagem surgiram sugestões de aperfeiçoamentos tanto na ferramenta quanto no próprio método. Por exemplo, a estrutura de dados subjacente permite implementar uma facilidade que permita ao usuário, a partir das fichas do Dicionário de Dados, pular diretamente para a(s) representação(ões) gráfica(s) no DFD, à moda hipertexto. Quanto ao método, é necessário que as fichas permitam alguns campos multivalorados onde originariamente eram monovalorados. É o caso dos campos origem e destino da ficha de fluxos de dados que são sempre multivalorados quando um fluxo está ligado a um processo. Por exemplo, se um fluxo tem destino no processo 1, no nível 1 do DFD, é possível que outros destinos válidos sejam 1.5, 1.5.3, 1.5.3.2, etc, nos outros níveis.

Pesquisas adicionais no Ambiente FEGRES estão sendo desenvolvidas no sentido de se obter uma representação generalizada de métodos estruturados que permita a inclusão de novas ferramentas no ambiente, de forma a suportar outras fases (e formas) do ciclo de vida. Acredita-se que a modelagem conceitual das ferramentas já implementadas, descrita neste trabalho, tenha fornecido subsídios para a continuação desta tarefa.

Referências:

CHEN 1976 - Chen,P.P., "The Entity-Relationship Model - Toward a Unified View of Data" ACM Transactions on Data Base Systems, vol 1 number 1, march 1976.

ELMASRI 1989 - Elmasri,R., Navathe,S.B. "Fundamentals of Database Systems" The Benjamin/Cummings Publishing Company, Inc, 1989.

GANE 1979 - Gane, C. e Sarson, T., "Structured Systems Analysis", Prentice-Hall, 1979.

MATTOSO 1987 - Mattoso, M.L.Q., "A Incorporação de Ferramentas de Apoio aos Usuários do SGBD COPPEREL", dissertação de Mestrado, Progr. de Sistemas - COPPE/UFRJ, maio 1987.

MICROSOFT 1988 - Corp., Microsoft "Microsoft MS-DOS User's Guide: Operating System Version 3.2"

SETZER 1986 - Setzer, V.W. "Projeto Lógico e Projeto Físico de Banco de Dados". V Escola de Computação - Belo Horizonte - 1986.

SOUZA 1988 - Souza, J.M. e Mattoso, M.L.Q., "COPPEREL-PC: A Versão do SGBD COPPEREL para Microcomputadores Tipo PC", Anais do 3o. Simpósio Brasileiro de Banco de Dados, março 1988.

TRAVASSOS 1990a - Travassos, G. H., "FEGRES: Ferramentas Gráficas para Engenharia de Software", Dissertação de Mestrado Programa de Engenharia de Sistemas - COPPE/UFRJ, 1990.

TRAVASSOS 1990b - Travassos, G. H., "Automatização de Métodos Estruturados para o Desenvolvimento de Sistemas: O Ambiente FEGRES", Aceito para publicação nos Anais da SUCESU - 1990

TROTTA 1990 - Trotta, C.N.F., Travassos,G.H., Souza,J.M. "Modelagem de Dados no Projeto de um Ambiente de Desenvolvimento de Software", Aceito para publicação nos Anais do Congresso da SUCESU 1990.