

SISTEMAS ESPECIALISTAS PARA ESPECIFICAÇÃO MODULAR
DE BANCOS DE DADOS

A. L. Furtado *
M. A. Casanova **
L. Tucheran **

* Pontificia Universidade Catolica do RJ
** IBM do Brasil

Adotando-se o modelo relacional de dados, podemos considerar que uma aplicação de bancos de dados envolve:

- tabelas (relações)
- restrições de integridade
- operações

As restrições de integridade estabelecem que estados do banco de dados são válidos (restrições estáticas) e que transições entre estados são válidas (restrições de transição). Uma estratégia conveniente para manter o banco de dados íntegro, especialmente quanto às restrições estáticas, consiste em impor pre-condições às operações consideradas críticas - tais operações teriam pois seus efeitos condicionados a testes apropriados, incluídos no corpo dos respectivos programas.

Adotada essa estratégia, podemos então acrescentar aos três componentes acima o seguinte relacionamento:

- operações garantem restrições

A especificação de uma aplicação é geralmente grande e complexa, o que nos obriga a construí-la de forma modular. Iniciamos o projeto com módulos PRIMITIVOS, envolvendo componentes das quatro classes mencionadas para descrever segmentos de uma aplicação que façam sentido isoladamente. Os dois tipos de módulos não-primitivos destinam-se a construir novos módulos a partir de outros módulos (primitivos ou não) além de permitir introduzir e garantir restrições de integridade globais a eles (módulos do tipo SUBJUGAMENTO) ou definir os limites de autorização, para consultar e atualizar partes do banco de dados (módulos do tipo EXTENSÃO). O construtor de subjugamento envolve a composição de módulos, enquanto o de extensão envolve composição ou decomposição.

Entre operações de um módulo e restrições de outro módulo que subjuga o primeiro pode valer mais um tipo de relacionamento:

- operações podem-violar restrições

Dentro da orientação de tipos abstratos de dados, este problema é enfrentado impondo que tais operações só sejam chamadas a partir

de outras operações, definidas no modulo subjugante (no qual as novas restrições foram introduzidas).

Nos modulos definidos por extensão todas as tabelas são na verdade visões (tabelas virtuais) e as operações neles introduzidas se traduzem em chamadas a operações dos modulos estendidos onde as tabelas reais residem. Quaisquer restrições que neles figurem devem ser consequencias logicas das restrições dos modulos estendidos, de modo que de fato não são novas restrições e portanto nada adicionam ao problema da integridade.

Em termos da arquitetura ANSI/X3/SPARC, o esquema conceitual é constituído por uma hierarquia de modulos primitivos e construídos por subjugamento. As raizes das arvores que constituem a hierarquia são os modulos ativos. Os esquemas externos correspondem aos modulos criados por extensão, os quais formam uma ordem parcial construída a partir dos modulos ativos do esquema conceitual.

Para dar apoio a especificações seguindo nossa metodologia para projeto modular, estamos desenvolvendo um prototipo de sistema especialista. O prototipo é escrito em micro-PROLOG utilizando a extensão para sistemas especialistas APES [1].

A metodologia está formalizada por um conjunto de REQUISITOS, que definem o que seja um projeto consistente. Os requisitos refletem ainda algumas exigencias para que os esquemas e operações estejam ativos. Dos requisitos se depreende uma ordem natural para criar ou modificar os componentes da especificação: 1. esquemas, 2. restrições, 3. podem-violar, 4. operações, 5. garantem. Por outro lado, a criação ou modificação dos modulos segue a ordem de definição (se o modulo A subjuga ou estende B, B deve ser definido antes de A).

O prototipo se destina a auxiliar nas fases de:

I. PROJETO

Para esta fase temos:

- a) um programa que escalona, de acordo com a ordem natural, a criação de modulos e seus componentes;
- b) a facilidade de consulta-ao-usuario, do APES;
- c) um analisador sintatico, que reconhece expressões em calculo de predicados de primeira ordem e na linguagem formal de programação proposta em [2];
- d) os proprios requisitos, expressos em forma declarativa.

Interagindo com o usuario (projetista de banco de dados), o prototipo de projeto resulta na criação de um dicionario de dados. Atraves da aplicacao dos requisitos, pretende-se que o

projeto seja correto por construção. A sintaxe livre de contexto (formulas bem formadas para as restrições, comandos escritos corretamente para as operações, etc.) é verificada pelo proprio analisador sintatico. A sintaxe sensivel ao contexto (ex: operações introduzidas em um modulo só podem atualizar diretamente tabelas definidas no modulo) faz parte dos requisitos e é verificada apos a aplicação do analisador, o qual produz uma arvore sintatica.

Entretanto, os requisitos semanticos (ex: as pre-condições das operações devem ser suficientes para que as restrições de integridade sejam mantidas) não são verificados diretamente, limitando-se o prototipo a alertar o usuario e a confiar em suas respostas. Para verificar/testar a suficiencia das pre-condições e a tradução correta de operações sobre visões, um prototipo desenvolvido separadamente pode ser utilizado pelo usuario, ou previamente ou interrompendo temporariamente a sessão com o prototipo de projeto - trata-se do GERADOR DE PLANOS, descrito em [3], escrito em micro-PROLOG.

Para modificar uma especificação, o prototipo preve a fase de

II. REPROJETO

Para esta fase, temos as facilidades b), c) e d) da fase anterior, juntamente com:

- a') um programa proprio de escalonamento;
- e') regras de propagação, expressas na mesma forma declarativa dos requisitos

sendo que o programa de escalonamento segue a mesma ordem natural. O usuario indica a REMOÇÃO ou MODIFICAÇÃO de alguma tabela, restrição ou operação. A INSERÇÃO de varios componentes de uma mesma classe e do mesmo modulo é tambem possivel. O prototipo, alem de processar a alteração, leva o usuario a examinar todos os demais componentes que possam ser afetados (propagação), processo esse que continua sobre os componentes afetados que por sua vez sofram alteração. Tanto a nova formulação dos componentes como, no caso de o usuario decidir manter um componente afetado, sua formulação antiga são verificadas pela aplicação dos requisitos do mesmo modo que na fase de projeto.

Novamente o problema dos requisitos semanticos recai na dependencia da consulta ao usuario, o qual poderá tambem aqui apoiar-se no prototipo de geração de planos.

As regras de propagação classificam-se em manuais (atingindo tabelas, restrições e operações) e automaticas (para relacionamentos podem-violar e garantem). Durante a propagação são feitas consultas ao usuario, que tambem recebe mensagens. Em

virtude da ordem natural, quando cada componente afetado é alcançado todas as possíveis linhas de propagação que conduzem a ele já foram exploradas. Assim, se uma tabela e uma restrição foram modificadas e ambas afetam uma operação, quando esta for considerada as consequências das duas modificações já estarão disponíveis.

A fase de reprojeto trabalha sobre o dicionário de dados, produz como resultado intermediário um registro de alterações e, a cada alteração validada quanto aos requisitos, atualiza imediatamente o dicionário de dados. As alterações são efetuadas sobre uma cópia do dicionário residente na memória, de modo que se no final da sessão o usuário não estiver satisfeito com o resultado (em vista, por exemplo, de alterações que tenha sido obrigado a efetuar no processo de propagação) basta que não determine sua gravação sobre o arquivo anterior residente em memória secundária.

As duas fases de nosso protótipo, bem como o protótipo de geração de planos (que no momento estamos procurando integrar em um mesmo projeto de pesquisa), funcionam em máquinas PC-compatíveis. A metodologia em que se baseia tem sido objeto de publicações [4,5]. O projeto de pesquisa representa esforço conjunto da PUC/RJ e da IBM do Brasil.

Tratando-se de implementação em micro-computador, temos a todo instante que resolver problemas de espaço, inclusive com a colocação de partes do sistema em disco. Os planos futuros incluem o aprofundamento do estudo dos requisitos semânticos e o desenvolvimento da implementação do protótipo, tanto para incluir novos aspectos quanto para aperfeiçoar sua eficiência.

REFERENCIAS

- [1] P. Hammond e M. Sergot - "APES : augmented prolog for expert systems" - manual de referencia -Logic Based Systems Ltd. (1984).
- [2] M. A. Casanova e P. A. Bernstein - "A formal system for reasoning about programs accessing a relational data base" - ACM Transactions on Programming Languages and Systems, 2, 3 (1980) 386-414.
- [3] A. L. Furtado e C. M. O. Moura - "Expert helpers to data-based information systems" - Anais do International Workshop on Expert Database Systems (1984) 298-313.
- [4] L. Tucheran, M. A. Casanova e A. L. Furtado - "A pragmatic approach to modular database design" - Anais da 9a. International Conference on Very Large Data Bases (1983) 219-231.
- [5] L. Tucheran, A. L. Furtado e M. A. Casanova - "An expert system for modular database design" - Anais da 11a. International Conference on Very Large Data Bases (em publicação em 1985).