7
Conclusões e Trabalhos Futuros

No cenário de execução de consultas tradicional, uma Máquina de Execução de Consultas (MEC) é o componente de um SGBD capaz de executar Planos de Execução de Consultas (PEC) fornecidos, em geral, pelo otimizador do SGBD. Além disso, as MEC tradicionais são eficientes na execução de consultas devido ao seu modelo de execução tradicional que emprega as técnicas de execução de paralelismo e distribuição.

Com o surgimento do modelo computacional baseado na *Internet* e do modelo de dados semi-estruturados, foram produzidos novos modelos de dados (tal como o modelo XML) e novos modelos de execução de consultas (tais como: modelo adaptativo, modelo contínuo e modelo baseado em *streams*), os quais não são suportados pelas MECs tradicionais. Isto resulta num esforço de construção considerável de novas MECs, a partir do seu início ou estendendo as MECs tradicionais, como o visto recentemente nos SGBDs comerciais para o suporte ao modelo de dados XML. Além disso, esse esforço de construção de MECs aumenta à medida que surjem novos modelos de execução, resultando na necessidade de se projetar MECs de fácil reuso e adaptação frente aos novos cenários.

Neste contexto, a solução proposta neste trabalho de tese foi o de desenvolver uma MEC extensível para o suporte a diferentes modelos de execução e de dados, de forma ortogonal. A abordagem escolhida foi utilizar a técnica de *framework* de *software*. Como resultado, produziu-se o *framework* QEEF (*Query Execution Engine Framework*). A escolha da técnica de *framework* deve-se principalmente ao fato dela ter sido bastante usada para construir sistemas com uma maior flexibilidade e que podem ser adaptados com maior rapidez e facilidade para atender aos requisitos de novas aplicações e, portanto, se mostra adequada ao nosso contexto.

O primeiro passo em direção à solução proposta foi o de reunir um conjunto de conceitos relativos à execução de consultas. Inicialmente, foram descritos os principais elementos de uma MEC, tais como: operadores (algébricos e de

controle), tuplas de dados, interface dos operadores e PECs. Em seguida, foram analisados alguns cenários de execução de consultas e capturados, certas técnicas de execução denominadas de características de execução de consultas. Por último, as características são implementadas no QEEF através de módulos de execução.

O QEEF fundamenta-se na combinação de módulos para permitir o suporte a diferentes modelos de execução. Neste caso, o usuário deve especificar um PEC contendo os módulos de execução associados ao plano algébrico da consulta. Para permitir esta especificação, foi proposto um Meta-Modelo de Execução de Consultas denominado de QUEM *(QUery Execution Meta-model).* Este meta-modelo define as regras de combinação dos módulos existentes no QEEF e de novos módulos que possam surgir. Os PECs especificados segundo o QUEM são denominados de meta-PECs e são pré-processados pelo QEEF resultando em PECs finais que, por sua vez são executados pela MEC instanciada do QEEF gerando o resultado da consulta.

O QEEF pode ser instanciado para diferentes modelos de dados de forma ortogonal ao modelo de execução. Isso possibilita tanto a instanciação de uma MEC baseada num modelo de dados específico quanto uma baseada em mais de um modelo de dados. Para se obter tal flexibilidade instancia-se, para cada modelo de dados, seus operadores algébricos e a estrutura de sua instância de dados.

O QEEF é um *framework* de infra-estrutura do tipo caixa-branca e pode ser instanciado segundo seus *hot-spots*. Além disso, sua arquitetura flexível permite que seja utilizado de forma *ad hoc* ou como um componente de *software*, a ser conectado aos demais componentes de um SGBD.

Para a validação do *framework* utilizamos três estudos de casos, os quais instanciam as MECs: RQEE, AQEE e XQEE. As consultas utilizadas nestes casos são executadas para diferentes modelos de execução (tradicional, adaptativo e baseado em streams) e diferentes modelos de dados (relacional e XML). Além disso, os meta-planos dessas consultas foram especificados através de documentos XML cuja DTD é baseada no meta-modelo QUEM (*QUery Execution Meta-model*)

Embora não se tenha encontrado na literatura uma MEC extensível com suporte a diferentes modelos de dados e de execução, e de forma ortogonal como o QEEF, existem alguns trabalhos relacionados com a extensibilidade de SGBDs, os quais são apresentados a seguir.

A maioria das abordagens estendem SGBDs com novos tipos de dados, os quais provêem a estrutura e comportamento requeridos para a independência de modelo de dados. Nesta direção, os principais trabalhos são:

- *Exodus* (Graefe&DeWitt, 1987), um gerador de otimizadores de consulta que adapta funcionalidades específicas dos SGBDs, tendo como seu sucessor, o sistema *Shore* (Carey et al., 1994);
- *Volcano* (Graefe&McKenna, 1993), considerado mais extensível do que o *Exodus*, também é um gerador de otimizadores e criou o operador de controle *exchange* que desacopla a comunicação entre os operadores algébricos;
- *Predator* (Seshadri&Paskin, 1997) foi construído em camadas, onde a camada de mais baixo nível contém o subsistema de armazenamento e a de mais alto nível contém tipos abstratos de dados que encapsulam o processamento de consultas, permitindo a criação de diferentes processadores de consulta para novas álgebras.

Mais recentemente, surgiu o pacote XXL (Bercken et al., 2001) que fornece uma biblioteca de operadores algébricos (por exemplo: operadores *sort-hash-merge-spatial joins*) que podem ser conectados uns aos outros e executados numa aplicação Java, segundo o modelo de execução tradicional. Neste caso, o desenvolvedor da aplicação é quem deve gerenciar a montagem e a execução dos PECs. Por último, o trabalho apresentado em (McCann, 2003) propõe uma arquitetura de um sistema adaptativo, baseado em componentes, auto-reconfiguráveis com a ajuda de monitores que geram estatísticas sobre a sua performance. Esta arquitetura de componentes se propõe a ser extensível o suficiente para eliminar a fronteira entre um sistema operacional e um SGBD.

As contribuições desta tese são as seguintes:

- A análise dos modelos de execução dos cenários apresentados resultando numa tabela que associa as características de execução aos respectivos cenários;
- A criação do meta-modelo QUEM que permite a criação e a combinação de novos módulos de execução, resultando na extensibilidade para o suporte a novos modelos de execução;

- A implementação do QEEF e sua instanciação de forma ortogonal para diferentes modelos de execução e de dados produzindo as MECs RQEE, AQEE e XQEE. Cada instanciação reutiliza uma infra-estrutura básica, sem que o construtor tenha que construí-la do estágio inicial;
- A integração do QEEF aos projetos de pesquisa em andamento descritos na Seção 1.6;
- As publicações (Ayres et al, 2002a, b, c, 2003).

Os trabalhos em andamento são listados a seguir:

- Implementação da MEC RXQEE (*Relationl-XML Query Execution Engine*). Esta MEC está sendo implementada como parte da dissertação de mestrado em informática (DI/PUC-Rio) da aluna Amanda Lopes. Esta MEC executará consultas baseadas num ambiente de integração de dados na *web* e contendo junções sobre fontes de dados heterogêneas XML e Relacional. Aplicações como essa apresentam desafios, na medida em que se combina diferentes modelos de dados numa mesma consulta, havendo a necessidade de conversão entre os modelos. Neste caso, são implementados operadores de conversão entre os dois modelos.
- Implementação da MEC denominada ROSA-QEE (*ROSA Query Execution Engine*). O *framework* QEEF está sendo instanciado para o modelo de dados ROSA (Porto et al., 2003), no contexto da dissertação de mestrado do aluno Fábio José Coutinho da Silva no Instituto Militar de Engenharia (IME). A álgebra do modelo ROSA está baseada no processamento de coleções de Objetos de Aprendizado e de seus relacionamentos semânticos. As operações da álgebra incluem aquelas derivadas da álgebra relacional (seleção, projeção, junção), bem como operações específicas do modelo, tal como *browse*.

Com relação aos trabalhos futuros, pretende-se utilizar o QEEF para a descoberta de novos cenários de execução de consultas a partir de experimentos que combinem diferentes (ou novos) modelos de execução e de dados. Para isso, pretende-se utilizar o ambiente de integração CODIMS (ver Seção 1.6).