Programação Orientada a Objetos versus Programação Procedural (Parte II - Final

Mário Leite

...

A **Parte I** abordou, de um modo geral, a Programação Orientada a Objetos (OOP) com suas características básicas relacionadas às classes e os objetos dessas classes. No final, deixou o foco nos dados: elementos fundamentais em qualquer tipo de sistema computacional. A literatura mostra muitos artigos sobre os bancos de dados orientado ao objeto (ODMS: **O**bject-oriented **D**ata**B**ase **M**anagement **S**ystem-Sistema Gerenciador de Banco de Dados Orientado a Objetos.

Aqui vou citar, *ipsis litteris*, o que foi escrito num *link* que trata, com bastante isenção, sobre isto: https://pt.stackoverflow.com/questions/113899/quais-s%C3%A3o-as-vantagens-de-utilizar-banco-de-dados-orientado-a-objetos (acesso em 06/03/2022 - 11:27).

Vantagens

- *Eles permitem uma modelagem que são ditas como mais próximas do mundo real, ainda que isto não seja possível de fato, em tese reduzindo a manutenção*
- *Embora seja possível em outros tipos de de DBs, a capacidade de criação de novos tipos de dados é melhor, o que obviamente permite criar estruturas de dados mais avançadas com melhores abstrações, mais flexíveis e teoricamente mais confiáveis, permitindo hierarquia*
- *Funciona melhor com linguagens orientadas a objeto, evitando a tal da impedance mismatch*
- *Navegação pelos dados é feita de forma mais natural e expressiva na maioria dos casos*
- *Alguns padrões de uso podem aumentar a performance (não usa JOIN)*
- *Melhor reuso*

Desvantagens

- *Falta padronização. Cada fornecedor usa uma forma diferente, determinada uma modelagem diferente. Há várias correntes que propões formas diferentes de uso deste modelo.*
- *Falta fundamentação matemática. Falta uma forma melhor de expressar consultas complexas da forma como as pessoas estão acostumadas fazer no modelo relacional*
- *Os produtos, apesar de longa existência, ainda não são maduros, as pessoas não o adotam porque não há adoção geral e por causa disto, não há investimentos suficientes para melhorias*
- *Para adotar o modelo fielmente há perda de performance em vários cenários. Para evitar isto, há uma quebra no modelo. Vários tipos de acesso precisam ser feitos de forma indireta através de objetos intermediários desnecessários naquela consulta*

- *A concorrência automática pode ser complicada ou trazer dificuldades. A manual exige mais do desenvolvedor*
- *A abstração pode esconder problemas reais no modelo adotado, pode ser difícil obter performance*
- *Faltam mecanismos já bem estabelecidos no modelo relacional para acesso aos dados de forma segura*
- *Faltam ferramentas, documentação, experiência, profissionais qualificados"*

Eu acrescentaria uma outra desvantagem: a dificuldade em utilizar uma linguagem de pesquisa eficiente como a SQL, a mais fácil de usar, mas que seria uma contradição num banco de dados estranho à sua própria gênese! E, ainda sobre o aspecto de alocação dos dados num banco de dados, acontece algo muito interessante, e que deve ser levado em conta na hora da argumentação a favor e contra esse tipo de banco de dados.

Embora os bancos OO possam, teoricamente, armazenar objetos (com *atributos* e *métodos* juntos) e permitir uma navegação mais "suave", o que se tem na maioria das vezes é o analista, mesmo optando por um banco OO, estabelecer os relacionamentos entre os locais onde estão os dados (entidades ==> tabelas) através do paradigma do Modelo Entidade-Relacionamento (MER), mesmo que a implementação seja numa linguagem orientada a objeto (C++, C#, Java, ect). Então, do meu ponto de vista, é uma contradição: sistema com banco relacional e implementação em linguagem orientada a objetos; o que seria um contrassenso para uma das vantagens deste paradigma! Alguns argumenta a favor dos bancos OO dizendo o seguinte: "*o mapeamento do MER pode ser feito adequadamente para o banco OO*". Mas, como isto pode ser trivial se MER diz respeito às relações baseadas no conceito matemático de Produto Cartesiano!? Assim, mesmo com a vantagem de poder armazenar propriedades e operações num só pacote, os bancos de dados orientados a objetos ainda são limitados à aplicações bem específicas, e não raramente muito caros para serem utilizados comercialmente além dos custos adicionais com pessoal especializado. De qualquer forma o que importa é a satisfação do USUÁRIO; isto deve ser SEMPRE considerado, pois a técnica de desenvolvimento de um sistema é a que menos importa, se o USUÁRIO não aprovar o produto final. O sistema pode ser muito eficiente, mas se não for eficaz, que que adianta!?
