Paradigmas: Para que servem!? - Parte III

Mário Leite

...

Continuando com os tipos de paradigmas das linguagens de programação, nesta parte serão apresentados os últimos três tipos.

5) Paradigma Orientado a Objetos

Este é, de todos, talvez o mais difundido e mais "badalado" entre os paradigmas de programação. Para os iniciantes em programação é um pouco trabalhoso o aprendizado, principalmente para aqueles que estão acostumados a programar em linguagens tradicionais. Neste paradigma, ao invés de se construir os sistemas com um conjunto de procedimentos (tal como nas linguagens imperativas), na orientação a objetos utiliza-se uma lógica mais próxima do mundo real; são criados objetos, oriundos de estruturas (*classes*) tentando uma compreensão melhor do projeto. Essas classes são moldes para produzir objetos com *atributos* e *funcionalidades*, os quais podem ser acessados através de mensagens externas. Isto, na verdade, torna este paradigma uma extensão melhorada da "programação modular". Exemplos de linguagens que seguem este paradigma: *C++*, *C#*, *Python*, *Visual Basic*, *Smalltalk*, *Ruby*, *Object Pascal*, *Java*, *Oberon*, *Eiffel*, *Simula*.

6) Paradigma Orientado a Eventos

É representado por todas as linguagens que fazem uso de interface gráfica, criando uma interatividade direta com o usuário, o qual age sobre elementos da interface, disparando ações sobre os objetos pré-definidos. O programa responde a essas ações, através de eventos predefinidos na classe do elemento acionado. Por exemplo, um clique sobre um botão de pressão numa interface, ou mesmo o pressionamento de alguma tecla numa caixa de texto durante a digitação. Normalmente, as linguagens com orientação a eventos também são orientadas a objetos, já que esses controles da interface são instâncias das classes definidas pela linguagem. Alguns exemplos de ambientes que fornecem linguagens desse tipo: *C#*, *Delphi*, *Visual Basic*, *Java*, que definem um novo modelo de criação de programas, às quais se convencionou chamar de "linguagens tipo RAD" (**R**apid **A**pplication **D**evelopment), pois agilizam o desenvolvimento de sistemas.

7) Paradigma Lógico

Também conhecido como "restritivo", é utilizado em aplicações de Inteligência Artificial. Este paradigma chega ao resultado esperado a partir de avaliações lógico-matemáticas. Para se obter a informação desejada utiliza-se **inferências** (ilações), que são operações "intelectuais" mediante a qual afirma-se a *verdade de uma proposição em decorrência de sua ligação com outras proposições já reconhecidas como verdadeiras*. De acordo com Vera Wannmacher Pereira (CAMPOS, Jorge; VANIN, Aline, p 10, 2009), temos a seguinte afirmação:

> "*O termo inferência é encontrado tanto na Psicolinguística como na Pragmática. Na Psicolinguística, a inferência consiste numa estratégia de leitura, assim como a predição leitora, exigindo processamentos cognitivos que manipulam pistas textuais deixadas pelo leitor, com o objetivo de chegar à compreensão do texto. Na Pragmática, constitui-se num percurso cognitivo que ocorre entre uma afirmação inicial e uma afirmação final (conclusão), sendo a base para cálculos de relevância*."

Na prática, este paradigma consiste em produzir conclusões a partir de premissas conhecidas, ou efetivamente verdadeiras. Um exemplo clássico de aplicação deste paradigma é com relação às seguintes proposições atribuídas ao pensamento filosófico:

“Todos os homens são mortais”
“Sócrates é um homem”

Conclusão: “Sócrates é mortal”

As linguagens de programação baseadas neste padrão trabalham com sentenças lógicas para deduzir verdades e optar por caminhos com base em análises do cruzamento de várias proposições, aplicando inferências cabíveis ao contexto. Exemplos de linguagens deste padrão: *Popler*, *Conniver*, *QLISP*, *Planner*, *Prolog*, *Mercury Oz*, *Frill*.

Continua na Parte IV