Paradigmas: Para que servem!? - Parte I

Mário Leite

...

Em minhas publicações tenho enfatizado que desenvolver um programa é observar a sequência de três etapas: *Pensar-Algoritmizar-Codificar,* rigorosamente nesta ordem. As duas primeiras etapas: *Pensar-Algoritmizar* é que eu chamo de Programação; é a etapa que, efetivamente, cria a solução do problema. A terceira parte (Codificar), embora seja a que produz o código-fonte do programa, é puramente técnica pois, só traduz para o computador, na sintaxe de alguma linguagem, o que a lógica da Programação decidiu previamente. É aí que surgem algumas críticas à maneira como eu coloco a atividade do programador (ou da equipe de programação): "*Programar é criar a solução de um problema apresentado pelo usuário*"; e não interessa se é um sistema complexo (Tratamento de imagens, Conversão voz-texto, Sistema ERP, Equações diferenciais, Solução do Gato de Schrödinger, Sistema de monitoração de um foguete, Criptografia de Chave Assimétrica, etc, etc). A maneira de como o sistema é criado não interessa ao principal interessado: o USUÁRIO; aquele que avalia o sistema, pois é o *stakeholder* mais importante de um projeto. É ele quem dá a palavra final sobre o sistema, decidindo se é bom ou ruim. Entretanto, no âmbito acadêmico é muito comum esquecerem deste "pequeno" detalhe, e colocar o desenvolvimento de um sistema numa camisa de força para justificar certas tecnologias que, embora IMPORTANTES, muitas vezes bagunçam a cabeça dos programadores iniciantes com termos não muito bem explicados, mas são ditatorialmente impostos para justificar algum "paradigma". E é exatamente este termo: "paradigma", que muitos críticos usam para qualificar um programa e enquadrá-lo dentro da sua visão de julgador: se a solução é moderna ou ultrapassada!

Mas, o que significa PARADIGMA?!

A Enciclopédia Eletrônica, no *link* (https://pt.wikipedia.org/wiki/Paradigma - acesso em 23/11/2021 - 11:26), o define assim:

"***Paradigma*** *(do latim tardio paradigma, do grego παράδειγμα, derivado de παραδείκνυμι «mostrar, apresentar, confrontar») é um conceito das ciências e da epistemologia (a teoria do conhecimento) que define um exemplo típico ou modelo de algo. É a representação de um padrão a ser seguido. É um pressuposto filosófico, matriz, ou seja, uma teoria, um conhecimento que origina o estudo de um campo científico; uma realização científica com métodos e valores que são concebidos como modelo; uma referência inicial como base de modelo para estudos e pesquisas*."

Eu prefiro algo mais simples e direto: "*É a maneira padrão de fazer alguma coisa*". Então, a famosa ação de "**quebrar o paradigma**" nada mais é do que fazer uma coisa diferente do que vinha sendo feita até então. Pronto, só isto! Não tem nada de misterioso ou complexo! É como se de repente, num jantar elegante, cheio de pompa e *salamaleques*, enquanto os ilustres convidados tentam pegar, elegantemente, o *escargot* com aquele alicate, você resolve pegar com a mão, por achar mais prático! Então, você quebrou o paradigma de "*como degustar um escargot no casulo*"; mas, embora não o tenha degustado pelo paradigma vigente, você comeu o *escargot*; ponto final! É claro que este meu exemplo é bem simples e direto para os iniciantes (não quer dizer que você vá comer um frango numa mesa, pegando-o inteiro com as mãos, como faziam os *vikings*) mas, mostra que "paradigma" nada mais é do que um padrão estabelecido para executar uma ação; e esse padrão é susceptível de análises mais críticas, e pode ser alterado.

No ambiente de desenvolvimento de sistemas as linguagens de programação podem ser classificadas de várias maneiras: *quanto à geração*, *quanto ao nível*, *quanto ao propósito, quanto ao tipo de tradução*, etc, etc); e também podem ser classificadas ***quanto ao paradigma***.

Assim, dentro dos sete principais tipos de paradigmas, as linguagens de programação podem ser classificadas em:

- Funcionais
- Imperativas
- Declarativas
- Estruturais
- Orientadas a Objetos
- Orientadas a Eventos
- Lógicas

Continua na Parte II