Programação Estruturada e os primórdios da informática

Salve padawan, graças aos feedbacks e comentários sobre a crise do Software, resolvi expandir o tema e falar sobre um dos primeiros paradigmas de programação, justamente criado para minimizar e auxiliar os desenvolvedores de software.

Se perdeu o artigo da Crise do Software, tem resumao.

Nos finais dos anos 60 do século passado, a programação era uma atividade iniciante: sem muitas regras, sem muitas ferramentas, exasperando líderes e usuário; por isso programar era uma tarefa de poucos, sem padrão, com muito código macarrônico, estruturas cheias de GOTO, gerando o problema conhecido como Spaghetti Code.

O software entregue não cumpria com o esperado, gastava-se mais tempo que o estimado, o custo final era maior e a manutenção evolutiva e as equipes de mantenimento sofriam para corrigir ou manipular o código, pois não tinha documentação e regras claras.

Qual foi a solução dada a primeira crise do software?

Inúmeros técnicos e acadêmicos, uniram esforços para melhorar a qualidade do software, criando inúmeras ferramenta utilizadas até nossos dias, cito algumas tais como o DFD, o Fluxograma, técnicas de programação e criação de variáveis (nomenclaturas) e ajuste nas linguagens, com a supressão do famigerado comando GOTO.

Em 1970 Michael Jackson um consultor informático publicou um livro, onde explica e defende a Programação Estruturada e juntamente com outros grandes da computação, fornece as bases que dirigiu o desenvolvimento de software por várias décadas.

O que é Programação Estruturada?

Em linhas gerais é um paradigma de programação, top-down, onde o código é totalmente legível, seguindo uma linha logica, fácil de ser entendida e lida em linguagem natural, usada principalmente nas linguagens de segunda geração, devido a sua simplicidade ainda hoje este paradigma é usado em diversas linguagens de programação.

Usada principalmente em projetos mainframe, sendo o método preferido dos programados old-school, sendo o COBOL, PL/I e, Natural os principais expoentes deste paradigma, onde o GOTO, apesar de presente, foi banido da codificação.

A facilidade de uso deve-se ao uso de três regras: Sequencia, Decisão e Iteração, através de estrutura simples e de fácil entendimento.

O problema do GOTO

No passado os programadores acostumados a linguagens de primeira geração, baseados em hardware, necessitavam implementar diversas estruturas além das regras de negócio, gerando programas enormes, era a época do cartão perfurado, que por motivo de performance e facilidade de uso abusavam do GOTO.

Resolvia rapidamente o problema, mas gerava um código sujo e de difícil entendimento e debug, piorando progressivamente quanto maior fosse o programa, os DEVs acostumados a essas facilidades continuaram com o vício do GOTO, criando códigos monstruosos e de difícil manutenção.

A Evolução do Paradigma Estruturado: Programação Modular

O código gerado continuava enorme e pouco reutilizável, por isso, com a experiência adquirida a PE, recebeu mais algumas melhorias e novas regras foram adicionados, visando programas mais limpos, com menos código e melhor reutilização, surgiu a programação modular.

Neste novo estilo convencionou-se dividir o programa em modulo, pequenos blocos ou funções para resolver problemas específicos, mas sem saírem ou desviarem-se do fluxo procedural principal.

A programação estrutura usa e abusa das sub-rotinas e funções, resolvendo o problema de código extenso e facilitando a manutenção, pois desta maneira o programador altera apenas um bloco de código, fácil de testar e isolar em caso de problemas.

O que são sub-rotinas na programação estruturada?

E um trecho de código isolado, que processa dados sem devolver resultado ou valor, muito útil em validação de campos e formulários, cálculos repetitivos, sendo chamado dentro da lógica principal, concluído seu processamento, retorna para o próximo comando, sequencialmente.

O que são Funções na programação estruturada?

A exemplo das funções matemáticas são trechos de códigos, que recebem parâmetros e devolvem um valor ao final do processamento, devolvendo o ponteiro do programa para o próximo comando, seguindo linearmente a lógica do programa.

Entendendo o paradigma.

Fala-se tanto sobre paradigma, mas afinal o que é um paradigma? Podemos resumidamente definir como um tipo de estruturação ao qual a linguagem deve respeitar e obedecer. Diferente de sintaxe de comandos, que é inflexível, obrigando o programador a obedecer, o paradigma é mais flexível na tratativa dada ao problema.

Um avanço e tanto

Para nós, que usamos linguagens de quarta e quinta geração, menosprezamos um pouco o paradigma estruturado e sua programação procedural e modularizada, por desconhecer o poder deste paradigma.

Diariamente milhares de Bancos pelo mundo afora, processam em seus cpds bilhões de registros em programas escritos em linguagem de segunda geração, altamente performáticos e otimizados para o obter o máximo dentro do mínimo.

Um dos problemas que persistem em nossos dias, é a cada nova evolução de hardware, aumenta o poder das cpus, porem a complexidade das linguagens de programação, aliadas a escassez de profissionais elevam o custo do desenvolvimento de software.

Jovens fogem do Mainframe

Devido há um erro no marketing, aliado a sisudez do ambiente de trabalho, ligado ao setor financeiro, grande usuário dos sistemas mainframe e programação estruturada, a maior parte dos novos programadores, olham com desdém e torcem o nariz para este trabalho.

Acreditando não ter o glamour de um aplicativo mobile ou web, com suas cores, botões e interfaces artísticas. Deveria ser revisto esta posição e mostrar as novas gerações , que o mainframe é cool.

Conclusão

O surgimento dos paradigmas de programação deve-se a evolução do hardware e software, que permitem a criação de estruturas de códigos, mais complexas e cada mais semelhante a estrutura natural do ser humano.

Pense que nos primórdios o código era em linguagem de máquina, que poucas pessoas fora do Centro de Processamento de Dados, entendiam e sequer conseguiam ler, devido ao uso de cartões perfurados.

Conforme as linguagens de programações evoluíram, alguns vícios da época da linguagem de máquina persistiram, o uso descontrolado do GOTO e os programas gigantescos com inúmeros trechos de código repetido e logicas macarrônicas eclodiram com a crise do software.

Inúmeros procedimentos e trabalhos de pesquisa criaram diversos paradigmas de programação, cada qual, tentando solucionar a crise de software seguinte, caso surja interesse da maioria, podemos escrever novos artigos falando mais detalhadamente cada um.