Resenha – *On the Criteria To Be Used in Decomposing Systems into Modules*

Kaio Souza Oliveira Mayer

O artigo de Parnas é um marco na Engenharia de Software porque questiona os critérios tradicionais de modularização e propõe uma forma diferente de pensar a decomposição de sistemas. Até então, a prática comum era dividir um sistema em módulos seguindo o fluxo de processamento, quase como uma extensão natural de um fluxograma. Essa abordagem, segundo o autor, traz problemas de manutenção e dificulta a evolução do software.

Parnas defende que os módulos devem ser definidos com base no princípio do *information hiding*, ou seja, cada módulo deve esconder decisões de projeto que são propensas a mudar. Com isso, alterações futuras afetam apenas um módulo, sem gerar impacto cascata em todo o sistema. Essa ideia ainda hoje é um dos fundamentos do design de software.

Para explicar sua proposta, o autor apresenta o exemplo de um sistema de geração de índices KWIC (Key Word in Context) e compara duas formas de modularizá-lo:

- Modularização convencional: baseada em etapas do processamento (entrada, deslocamento circular, ordenação, saída).

- Modularização alternativa: baseada no ocultamento de decisões, como a forma de armazenamento das linhas ou a implementação da ordenação.

A análise mostra que a segunda abordagem traz vantagens claras: maior facilidade de modificação, desenvolvimento independente dos módulos e melhor compreensibilidade do sistema. Mesmo que a implementação possa parecer menos eficiente em termos de chamadas de funções, Parnas sugere técnicas que preservam a eficiência sem abrir mão dos benefícios de sua proposta.

O artigo ainda aborda a importância da hierarquia entre módulos, destacando que uma boa decomposição pode ser reutilizada em diferentes contextos (como compiladores e interpretadores). Essa ideia reforça a noção de que uma arquitetura bem planejada reduz retrabalho e aumenta a vida útil do software.

Em conclusão, Parnas nos alerta para um erro comum: iniciar a modularização a partir de um fluxograma. A verdadeira chave está em identificar as decisões críticas de projeto e encapsulá-las em módulos que protejam o sistema de mudanças futuras. Essa visão não apenas antecipou práticas modernas como *design patterns* e princípios de orientação a objetos, mas também consolidou a modularização como um pilar central da Engenharia de Software.

Minha percepção como estudante:
O texto é de 1972, mas é impressionante como continua atual. A ideia de *information hiding* parece óbvia hoje, mas pensar nisso naquela época foi revolucionário. Esse artigo me fez perceber que modularizar não é apenas dividir o código em partes menores, e sim pensar estrategicamente em como proteger o sistema das mudanças inevitáveis que virão.