ANÁLISE DO ARTIGO
“ON THE CRITERIA TO BE USED IN DECOMPOSING SYSTEMS INTO MODULES”

VINÍCIUS GOMES RODRIGUES
ENGENHARIA DE SOFTWARE - PUC MINAS
BELO HORIZONTE - 2025

O artigo "On the Criteria To Be Used in Decomposing Systems into Modules" (1972), de David L. Parnas, é um marco na engenharia de software e no desenvolvimento de sistemas modulares. Nele, Parnas desafia a prática comum de decompor sistemas com base em fluxos de processo (flowcharts) e propõe uma abordagem alternativa fundamentada no princípio de ocultação de informação (information hiding). O autor argumenta que a modularização deve ser orientada pela necessidade de esconder decisões de design propensas a mudanças, rather than por etapas sequenciais de processamento.

Parnas inicia o artigo destacando a falta de critérios claros para a decomposição modular, citando que a literatura da época enfatizava benefícios gerais (como facilidade de gestão e manutenção), mas não oferecia diretrizes práticas. Para ilustrar sua tese, ele compara duas modularizações distintas para um mesmo sistema: um produtor de índice KWIC (Key Word In Context).

1. Modularização convencional

Baseada em etapas de processamento (input, circular shift, alphabetizing, output e master control). Os módulos compartilham estruturas de dados complexas e acopladas, tornando o sistema rígido a mudanças

2. Modularização proposta por Parnas

Baseada em ocultação de informação. Cada módulo esconde uma decisão de design específica (ex.: armazenamento de linhas, geração de shifts circulares). As interfaces são abstraídas por funções (ex.: CHAR(r, w, c)), minimizando dependências internas.

Parnas demonstra que a segunda abordagem oferece maior flexibilidade, independente de desenvolvimento e compreensibilidade. Por exemplo, alterações na representação de dados ou algoritmos afetam apenas um módulo, enquanto na primeira abordagem impactam múltiplos componentes.

O autor também discute a importância de uma estrutura hierárquica (módulos de baixo nível independentes) e aborda preocupações com eficiência, sugerindo que técnicas de implementação inteligentes (ex.: inline code) podem mitigar o overhead de chamadas de funções.

Quase 50 anos após sua publicação, o artigo ainda permanece relevante nos temas:

1. Arquitetura de Software: Princípios de ocultação de informação são aplicados em APIs, microsserviços e containers.
2. Gestão de Mudanças: Sistemas que escondem decisões voláteis (ex.: formatos de dados) são mais adaptáveis a requisitos em evolução.
3. Segurança: Ocultação de detalhes de implementação reduz superfícies de ataque.