O Segredo por Trás das Paredes: Uma Análise sobre "On the Criteria To Be Used in Decomposing Systems into Modules"

Quando começamos a desenvolver software, uma das primeiras lições que aprendemos é a importância de modularizar. A promessa é tentadora: quebrar um problema gigante em pedaços menores, facilitando o trabalho em equipe, a manutenção e a evolução do sistema. Contudo, a realidade do dia a dia muitas vezes nos mostra que essa divisão, por si só, não é garantia de sucesso. É comum nos depararmos com módulos que, embora separados, estão tão entrelaçados que uma pequena mudança em um deles causa um efeito dominó desastroso. É exatamente para resolver essa frustração que o artigo "On the Criteria To Be Used in Decomposing Systems into Modules", de D.L. Parnas, se torna uma leitura tão fundamental.

Para nos mostrar o problema na prática, Parnas utiliza um exemplo simples: a criação de um sistema de índice KWIC. Ele nos apresenta duas formas de dividir o trabalho. A primeira é a que a maioria de nós faria por instinto, a abordagem convencional que segue o fluxo de processamento. Criamos um módulo para ler os dados, outro para processá-los, um terceiro para ordenar e, por fim, um para exibir o resultado. É uma lógica que faz todo sentido quando pensamos em um fluxograma, um passo a passo claro de execução.

Mas é na segunda abordagem que o artigo vira o jogo. Parnas propõe algo diferente: e se, em vez de dividir pelos passos do processo, dividíssemos o sistema com base nas decisões de design? Nessa nova visão, teríamos um módulo cuja única responsabilidade é o armazenamento das linhas. Outro, a lógica dos deslocamentos. Um terceiro, a ordenação. A grande sacada é que cada um desses módulos esconde os detalhes de como ele funciona. O resto do sistema não sabe, nem precisa saber, se os dados estão em memória ou em disco, ou qual algoritmo de ordenação está sendo usado. Ele apenas interage através de uma interface clara, pedindo o que precisa.

O impacto dessa mudança de perspectiva é gigantesco. Na abordagem do fluxograma, se decidirmos alterar a forma como os dados são armazenados, somos forçados a modificar quase todos os módulos, pois todos eles "sabem" demais sobre essa estrutura. É o pesadelo da manutenção. Já na segunda proposta, essa mesma mudança fica isolada, contida dentro das paredes do módulo de armazenamento. Enquanto sua interface pública não mudar, o resto do sistema nem percebe que algo aconteceu.

O critério por trás dessa mágica é o que Parnas batizou de "Information Hiding" (Ocultamento de Informação). A regra é simples: cada módulo deve ser projetado para esconder uma decisão de design importante, especialmente aquelas que podem mudar no futuro. Ao fazer isso, criamos barreiras intencionais entre as partes do

sistema. A comunicação passa a ser sobre "o que" um módulo faz, e não "como" ele faz.

Dessa forma, podemos concluir que a lição de Parnas transcende o tempo. Ele nos mostra que uma boa modularização não se trata apenas de organizar o código em pastas diferentes, mas sim de uma estratégia deliberada para gerenciar a complexidade e a mudança. Em um ambiente real, onde tudo muda o tempo todo, essa capacidade de isolar o impacto das decisões é o que separa um sistema resiliente de um frágil castelo de cartas. O verdadeiro desafio, portanto, não é simplesmente "dividir para conquistar", mas sim saber onde traçar as linhas, escondendo os segredos certos para que nosso software possa evoluir de forma saudável e sustentável.

Referências:

PARNAS, D. L. On the criteria to be used in decomposing systems into modules. In: *Communications of the ACM*, v. 15, n. 12, p. 1053-1058, dez. 1972.

Autor da resenha: Rafael de Faria Neves Alves Franco