Resenha – On the Criteria to be Used in Decomposing Systems into Modules
Leonardo Viana

O artigo On the Criteria to be Used in Decomposing Systems into Modules, escrito por David L. Parnas, me fez repensar bastante o que eu entendia por modularização. Sempre ouvi falar que modularizar era basicamente dividir o sistema em partes menores e mais simples, mas Parnas mostra que isso não é suficiente. Ele defende que o importante mesmo é pensar em quais decisões de projeto devem ser escondidas dentro dos módulos, porque são justamente essas decisões que mais podem mudar no futuro e causar dor de cabeça.

Um exemplo marcante que ele usa é o sistema de índice KWIC. Ele apresenta duas formas de dividir esse sistema: uma mais tradicional, que segue o fluxo do processo (entrada, shift circular, ordenação e saída), e outra baseada em esconder informações. A primeira parece mais simples, mas gera módulos muito dependentes entre si, o que dificulta qualquer mudança. Já a segunda foca em deixar cada módulo responsável por esconder detalhes internos, como a forma de armazenar as linhas ou de gerar os shifts. Isso torna o sistema mais flexível, porque se algo mudar, você ajusta só aquele módulo e o resto continua funcionando. Eu achei isso muito real, porque parece exatamente o tipo de erro que acontece quando alguém projeta pensando só no "agora" e não no "depois".

Outra parte que achei muito interessante foi quando ele fala dos benefícios de modularizar direito: flexibilidade, compreensão e independência no desenvolvimento. Um sistema bem modularizado é mais fácil de entender, porque você pode olhar um módulo de cada vez sem precisar abrir a cabeça de todos. Também facilita

mudanças e permite que diferentes pessoas trabalhem em partes diferentes sem ficarem presas umas às outras. Isso me fez pensar até na faculdade: quando você deixa tudo acumulado e dependente de um único esforço no fim, vira um caos. Mas se você consegue organizar as coisas em blocos independentes, consegue mudar ou ajustar sem tanto problema.

O ponto que mais me chamou atenção foi a crítica ao jeito comum de começar uma modularização a partir de um fluxograma. A gente sempre aprende a pensar no software como uma sequência de etapas, mas Parnas mostra que isso é um erro. Segundo ele, o certo é começar listando quais são as decisões de projeto mais complicadas ou mais prováveis de mudar, e aí criar módulos justamente para esconder essas decisões. No fim das contas, isso deixa o sistema muito mais preparado para evoluir.

A mensagem que ficou para mim é que modularizar não é só cortar o sistema em pedaços menores. É pensar em longo prazo, criando módulos que funcionem como caixas-pretas: eles mostram só o que é necessário e escondem o resto. Isso evita custos altos no futuro e deixa o sistema pronto para crescer sem tanto retrabalho. Assim como na vida, se a gente organiza bem desde o início, a chance de evitar problemas depois é muito maior.

Esse artigo me fez ver que modularização não é luxo de quem quer código "bonitinho", mas sim uma escolha estratégica. Se a gente não modulariza direito, pode até parecer mais rápido no começo, mas lá na frente o preço que se paga é muito maior. No fundo, o que Parnas mostra é que o segredo está em aceitar que mudanças sempre vão acontecer, e preparar o sistema para lidar com isso sem desmoronar.