João Vitor Santana Lopes

A maneira como um sistema de software é dividido em partes, ou "módulos", é fundamental para seu sucesso. Uma boa modularização pode melhorar a flexibilidade e a compreensão do sistema, além de permitir que o desenvolvimento seja mais rápido e gerenciável. A eficácia desse processo, contudo, depende inteiramente dos critérios usados para criar os módulos.

Uma abordagem comum para a modularização é baseá-la nas principais etapas do processamento. Por exemplo, em um sistema de indexação de texto, isso resultaria em módulos separados para a entrada de dados, para o processamento, para a ordenação alfabética e para a saída final. Cada passo na sequência de trabalho se torna um módulo distinto. Embora pareça lógico, esse método pode criar sistemas rígidos e difíceis de modificar no futuro.

Existe, no entanto, uma abordagem alternativa baseada no critério de "ocultamento de informação" (information hiding). Nessa forma de modularização, a divisão não segue as etapas do processamento. Em vez disso, cada módulo é projetado para encapsular e "esconder" uma decisão de design complexa ou que tenha grande probabilidade de mudar no futuro. Por exemplo, a forma como os dados são armazenados na memória é uma decisão que fica inteiramente contida dentro de um módulo de armazenamento. As outras partes do sistema não sabem como o armazenamento funciona internamente; elas apenas se comunicam com este módulo através de uma interface estável e bem definida, como um conjunto de funções.

A principal vantagem dessa abordagem é um ganho imenso em flexibilidade e facilidade de manutenção. Se uma decisão de design importante precisar ser alterada, como o formato de armazenamento dos dados, a mudança fica isolada dentro do único módulo responsável por ela. No método convencional, essa mesma mudança poderia exigir alterações em quase todos os módulos do sistema, tornando o processo caro e arriscado. Com o ocultamento de informação, o sistema se torna mais adaptável, pois as mudanças são localizadas. Isso também permite que as equipes de desenvolvimento trabalhem com mais independência e que o sistema seja mais fácil de entender em partes isoladas.