Resenha: *Hexagonal Architecture (Ports and Adapters)*

Diogo Brunoro

Outubro 2025

O artigo *"Hexagonal Architecture (Ports and Adapters)"*, escrito por Alistair Cockburn em 2005, apresenta uma proposta arquitetural inovadora para o desenvolvimento de sistemas de software mais flexíveis, testáveis e independentes de tecnologias específicas. O autor parte de um problema recorrente no design de aplicações: o acoplamento excessivo entre a lógica de negócio e elementos externos, como interfaces gráficas e bancos de dados.

Cockburn critica o modelo tradicional em camadas, no qual o código tende a "vazar" entre as fronteiras — por exemplo, quando regras de negócio acabam sendo misturadas à camada de apresentação. Essa mistura dificulta testes automatizados e a manutenção do sistema, além de limitar sua evolução. Como solução, ele propõe a Arquitetura Hexagonal, também conhecida como Ports and Adapters, que busca separar claramente o núcleo da aplicação de suas dependências externas.

A ideia central é que a aplicação deve ser construída para funcionar sem depender diretamente de um banco de dados ou interface gráfica. Em vez disso, o núcleo do sistema — onde está a lógica de negócio — se comunica com o mundo externo por meio de portas (ports), que definem as interfaces de comunicação, e adaptadores (adapters), que traduzem as interações dessas portas para tecnologias específicas, como um banco de dados SQL, uma API HTTP ou uma interface de usuário.

Um ponto interessante é a analogia que o autor faz entre as portas da aplicação e as portas de dispositivos eletrônicos: qualquer componente que respeite o protocolo de uma porta pode ser "plugado" nela. Essa metáfora ajuda a visualizar o propósito da arquitetura — tornar a aplicação independente da tecnologia que está do outro lado da comunicação.

Cockburn também reforça que essa independência traz benefícios práticos, como a possibilidade de rodar testes automatizados completos sem precisar de uma interface ou banco real, o que agiliza o desenvolvimento e garante maior confiabilidade. Além disso, o artigo mostra exemplos de código que ilustram como o conceito pode ser aplicado gradualmente, adicionando adaptadores de teste, de interface e de banco de dados.

Outro aspecto relevante é a distinção entre portas primárias (que dirigem a aplicação, como um usuário ou teste automatizado) e portas secundárias (que são conduzidas pela aplicação, como um repositório de dados). Essa separação facilita a compreensão do fluxo de comunicação e reforça a ideia de isolamento entre o "dentro" (a lógica de negócio) e o "fora" (as tecnologias e serviços).

O artigo ainda destaca as conexões da arquitetura hexagonal com outros padrões e princípios, como o Adapter Pattern, o MVC (Model-View-Controller) e o Dependency Inversion Principle, mostrando como ela se alinha a boas práticas de design orientado a objetos.

Em síntese, Cockburn propõe uma forma de organização de sistemas que prioriza testabilidade, manutenibilidade e independência tecnológica. A metáfora do hexágono é usada apenas como recurso visual — o formato em si é irrelevante, servindo apenas para representar múltiplas conexões possíveis.

Como estudante, considero que o artigo é uma leitura essencial para compreender as origens das arquiteturas modernas, como Clean Architecture e Onion Architecture, que evoluíram a partir dessas ideias. Apesar de ter sido escrito há duas décadas, o texto permanece atual por defender princípios fundamentais de design de software: isolamento de responsabilidades, baixo acoplamento e alta coesão.