10 - Arquitetura em Camadas

2025-12-20 19:20 pm

- Um dos estilos mais comuns
    - Simples e barato
    - Segue a lei de Conway, copia a comunicação da empresa
    - Às vezes acaba ocorrendo por acidente (anti-pattern)

Topologia

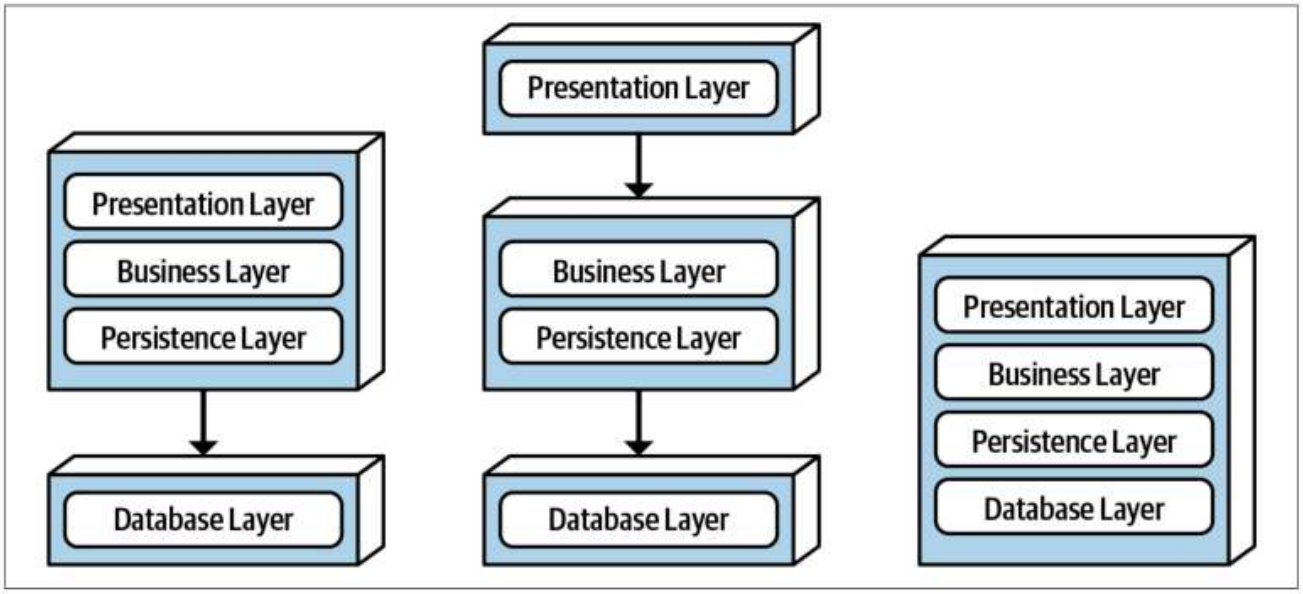

Figure 10-2. Physical topology (deployment) variants

- Organizado em camadas lógicas horizontais, cada camada tem uma responsabilidade.
- Geralmente quatro camadas padrão:
    - **Apresentação**
        - e.g. lidar com interação do usuário
    - **Negócio**
    - **Persistência**
        - Pode estar fundida à de negócio
    - **Database**
- Tem uma série de variações, combinando ou não as diferentes camadas.
- Produtos *on-prem* costumam combinar todas.
- Cada camada não se preocupa com as demais
    - e.g. camada de negócio não se preocupa com o formato de visualização
- Arquitetura particionada por técnica (e não pode domínio)
    - agrupamento de elementos por papel técnico
    - Ou seja, domínio se espalha por toda a arquitetura
        - e.g. todas as camadas têm que ter informações a respeito de o que é um Customer.
        - Incompatível com domain-driven design
- Difícil de ser alterada com agilidade.

Camadas de Isolamento

- Camada pode ser aberta ou fechada.
    - **Fechada**:
        - Se comunica só com as camadas adjacentes na hierarquia
    - **Aberta:**
        - Se comunica com qualquer camada sem passar pelas que não precisam
        - e.g. Camada de apresentação talvez precise acessar DB diretamente.
- Interdependência entre camadas causa acoplamento.
- Camadas fechadas facilitam isolamento e diminuem número de mudanças necessárias.

Adicionando Camadas

- Às vezes faz sentido ter camadas abertas.
    - E.g. quando um componente é compartilhado pela camada de apresentação e de negócio.
    - Seria ruim se a camada de apresentação demandasse um componente (e.g. dateutil) que pertence à camada de negócio)

Outras Considerações

- Costuma ser um bom começo de aplicação.
    - Depois é alterada para algo mais robusto.
- Manter uma hierarquia de objetos rasa e reuso de objetos ajuda migração futura.
- Respeitar a arquitetura em camadas só por respeitar mas elas não realizam nenhuma função, só a formalidade da hierarquia, é um *anti-pattern - architetural sinkhole*.
    - Um pouquinho disso acaba sendo difícil de evitar, mas muito é um problema.
    - Solução de abrir todas as camadas para evitar este *anti-pattern* gera outras problemas

Por que Usar este Estilo de Arquitetura

- Bom para pequenas aplicações simples ou websites, ou como ponto de partida.
- Bom quando há incerteza sobre a melhor arquitetura.
- Deve ser evitado para qualquer aplicação um pouco maior.

Architecture characteristic	Star rating
Partitioning type	Technical
Number of quanta	1
Deployability	★
Elasticity	★
Evolutionary	★
Fault tolerance	★
Modularity	★
Overall cost	★★★★★
Performance	★★
Reliability	★★★
Scalability	★
Simplicity	★★★★★
Testability	★★

- Principal benefício é custo e simplicidade.
- Toda mudança requer muito cuidado no código e é difícil de testar.