2 - Pensamento Arquitetônico

Monday, 3 March, 2025 8:14 Manhã

- Pensamento arquitetônica != pensar na arquitetura
- Arquitetura != Design

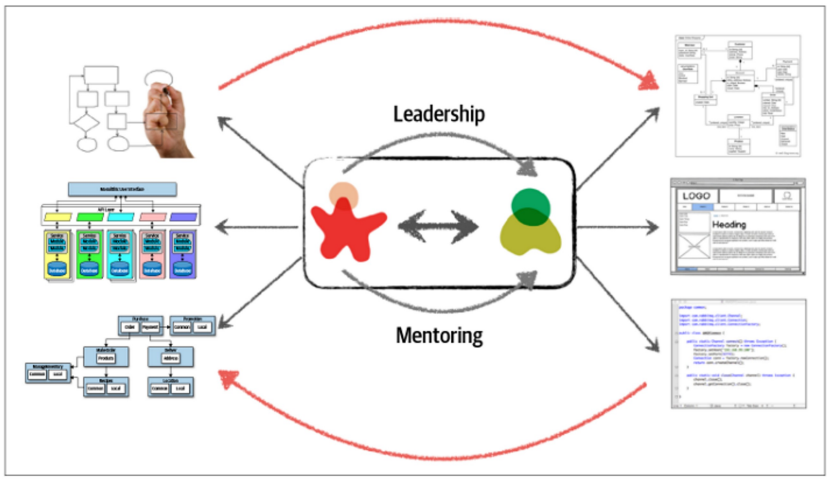


Figure 2-2. Making architecture work through collaboration

- Arquiteto deve estar em contato com devs
- Não é unidirecional - mentoria & liderança constante
- Iterativo

- **CONHECIMENTO**
  - Desenvolvedor:
    - Profundidade
  - Arquiteto
    - Amplitude
  - Difícil transição
  - Riscos de:
    - Especializações demais e rasas
    - Especializações obsoletas - dominar algo velho sem sabê-lo
      - "Frozen Caveman Antipattern"

- Tudo é um trade-off
- "Arquitetura é o que você não consegue pesquisar no Google"
- Tudo "Depende"
  - Não há certo ou errado
- Exemplo do Leilão:

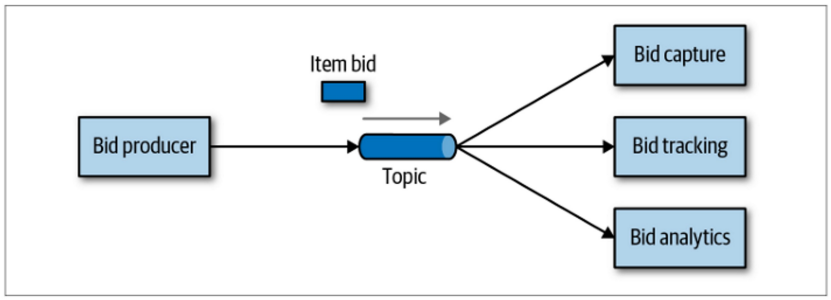


Figure 2-8. Use of a topic for communication between services

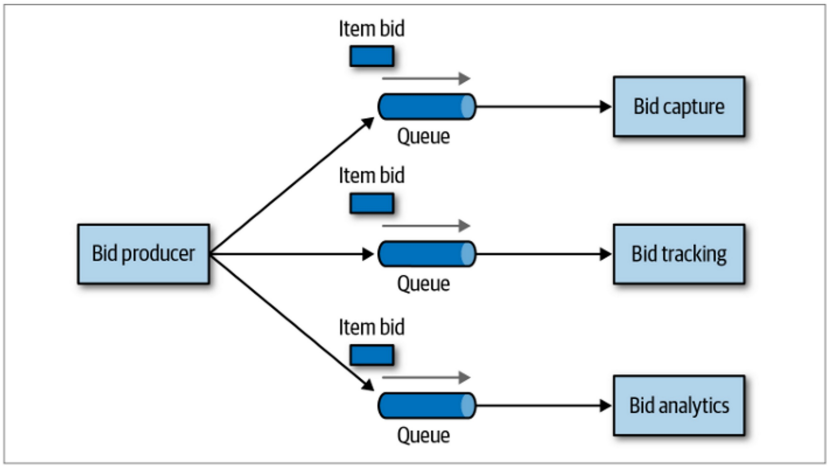
- 

Figure 2-9. Use of queues for communication between services

- Qual escolher?:
  - 1 tópico é mais fácil de estender mais uma funcionalidade
  - 3 filhas é mais seguro e separável, melhor load balancing
  - *Table 2-1. Trade-offs for topics*

Topic advantages	Topic disadvantages
Architectural extensibility	Data access and data security concerns
Service decoupling	No heterogeneous contracts
	Monitoring and programmatic scalability

- Equilibrar código com arquitetura nas suas funções como arquiteto
  - Arquiteto tem muitas funções, não pode se dedicar tanto ao desenvolvimento:
    - Delegar o caminho crítico a um dev
    - Codificar só coisas importante no futuro, não o gargalo atual
    - Fazer algo próximo do que devs fazem
    - POC - a melhor possível, já que código se tornará referência
    - Resolver déficit técnicos
    - Consertar erros
    - Criar ferramentas auxiliares - e.g. validadores
    - Code Review