UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS DE RUSSAS

Arquitetura de Software

Estilos e Padrões Arquiteturais

(Continuação)

Profa. Dra. Anna Beatriz Marques

3

Principais Padrões Arquiteturais

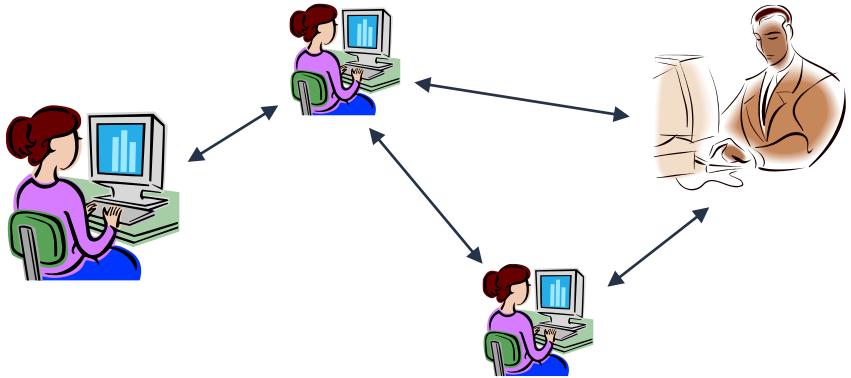
Ponto-a-Ponto (P2P)

Padrões arquiteturais: Ponto a Ponto (P2P)

- Contexto:
  - Componentes computacionais distribuídos precisam cooperar e colaborar para fornecer um serviço para uma comunidade de usuários distribuídos.
- Problema:
  - Como um conjunto de componentes computacionais semelhantes e distribuídos podem se conectar por um protocolo comum para se organizar e compartilhar seus serviços com alta disponibilidade e escalabilidade?

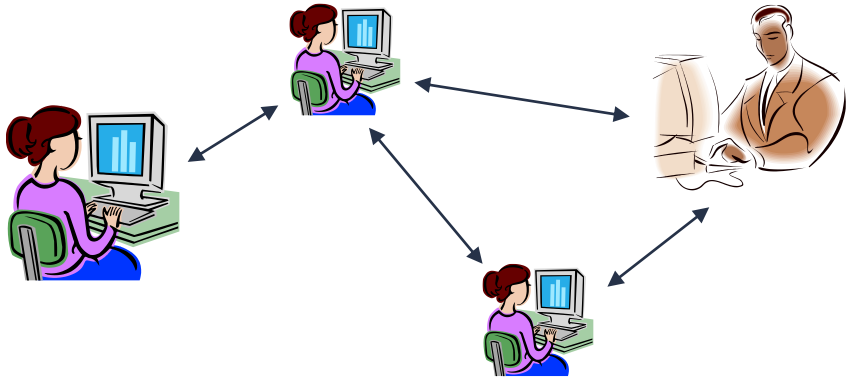
Padrões arquiteturais: Ponto a Ponto (P2P)

- Solução:
  - Componentes interagem diretamente como pares
  - Cada ponto (nó) mantém seus próprios dados e endereços conhecidos
  - Cada ponto (nó) é “cliente e servidor ao mesmo tempo”



Padrões arquiteturais: Ponto a Ponto (P2P)

- Solução:
  - Não há distinção entre nós
  - Cada nó fornece e consome serviços similares e utilizam o mesmo protocolo



Padrões arquiteturais: Ponto a Ponto (P2P)

- Vantagem:
  - Não há ponto de falha (nenhum nó ou grupos de nós são mais críticos que outros)
- Desvantagem:
  - Gerenciamento mais complexo: segurança, consistência dos dados, disponibilidade de serviços e dados.

Arquitetura orientada a serviços (SOA)

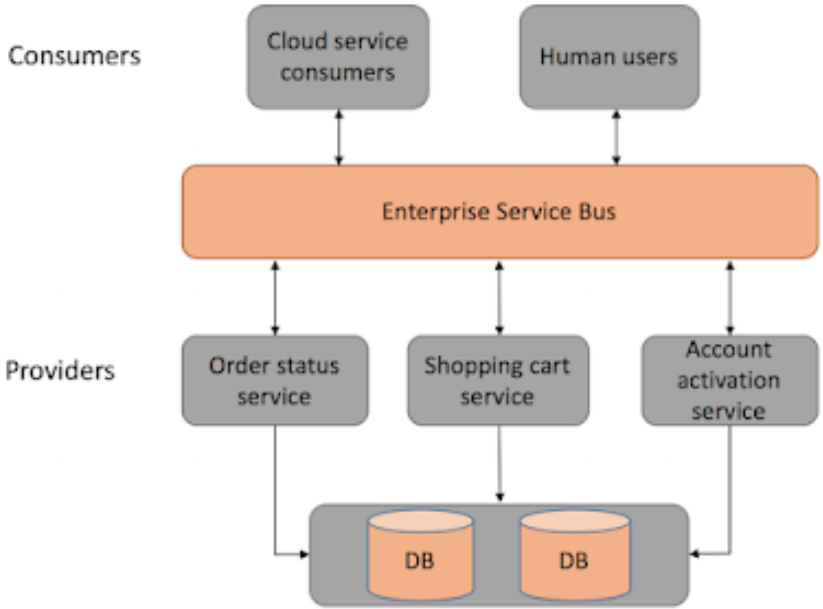
Padrões arquiteturais: Arquitetura Orientada a Serviços (Service-Oriented Architecture - SOA)

- Contexto:
  - Um conjunto de serviços são oferecidos por fornecedores de serviços e utilizados por consumidores de serviços.
  - Os consumidores precisam entender e utilizar os serviços fornecidos sem a necessidade de conhecer detalhes de sua implementação.
- Problema:
  - Como apoiar a interoperabilidade de componentes distribuídos rodando em diferentes plataformas, escritos em diferentes linguagens de programação e fornecidos por diferentes organizações, distribuídos pela Internet?

Padrões arquiteturais: Arquitetura Orientada a Serviços (Service-Oriented Architecture - SOA)

- Solução:
  - Uma coleção de componentes distribuídos que fornecem e/ou consomem serviços
  - Componentes (consumidores ou fornecedores) podem usar diferentes linguagens de implementação e plataformas.
    - Cada componente possui uma interface que descreve os serviços que solicitam e os serviços que fornecem.

Padrões arquiteturais: Arquitetura Orientada a Serviços (Service-Oriented Architecture - SOA)



- Fornecedor de serviço: fornecem um ou mais serviços por meio de uma interface. Podem ser consumidores também;
- Consumidor de serviço: solicita serviços diretamente ou por meio de um intermediário;
- ESB (Enterprise Service Bus): elemento intermediário que gerencia as mensagens entre consumidores e fornecedores.

Padrões arquiteturais: Arquitetura Orientada a Serviços (Service-Oriented Architecture - SOA)

- Vantagens:
  - Interoperabilidade
  - Reconfiguração dinâmica
- Desafios:
  - Complexidade de design e implementação (ligação dinâmica)
  - Sobrecarga de desempenho no middleware
  - Os serviços podem ter atualizações inesperadas

Produtor-Consumidor

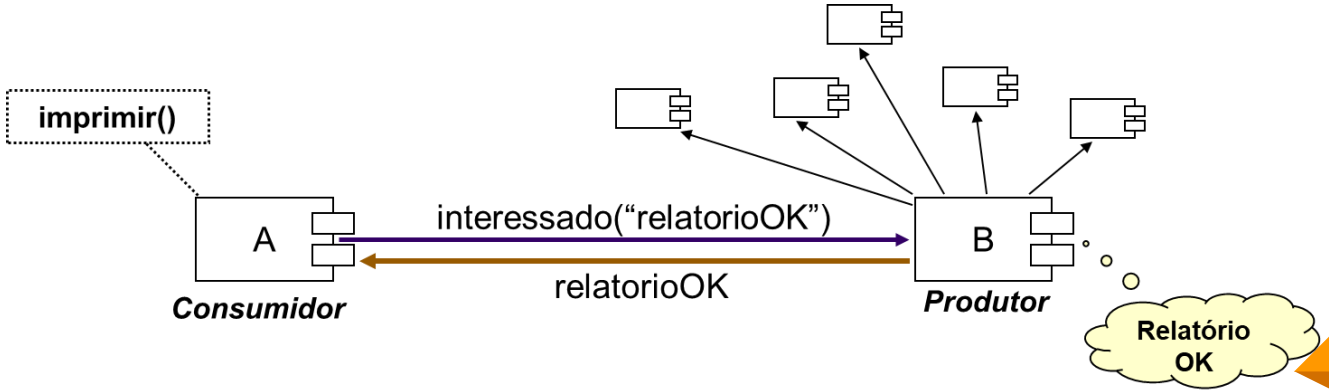
Publish-Subscribe

Padrões arquiteturais: Produtor-Consumidor

- Contexto:
  - Produtores e consumidores de dados são independentes e devem interagir.
  - A quantidade precisa de produtores e consumidores não é pré-determinada
  - A natureza dos dados compartilhados não é conhecida
- Problema:
  - Como criar mecanismos de interação para apoiar a transmissão de mensagens entre produtores e consumidores sem a necessidade de conhecerem a identidade ou existência uns dos outros?

Padrões arquiteturais: Produtor-Consumidor

- Solução
  - Produtores e consumidores de eventos
  - Consumidores se registram nos Produtores
  - Produtores notificam consumidores registrados



Padrões arquiteturais: Produtor-Consumidor

- Vantagens:
  - Produtores e consumidores são independentes
  - Escalabilidade no número de interessados
- Desvantagens:
  - Pode aumentar a latência e afetar negativamente a previsão do tempo de entrega de mensagens
  - Pouco controle na ordem de mensagens e na garantia da entrega de mensagens

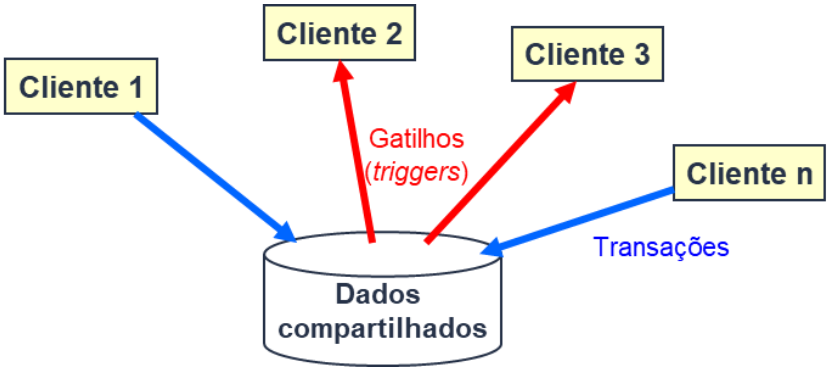
Dados compartilhados

Padrões arquiteturais: Dados compartilhados

- Contexto:
  - Diversos componentes precisam compartilhar e manipular um grande volume de dados.
  - Os dados não pertencem a nenhum dos componentes.
- Problema:
  - Como os sistemas podem armazenar e manipular a persistência dos dados que são acessados por múltiplos componentes independentes?

Padrões arquiteturais: Repositório

- Solução:
  - Troca de dados entre múltiplos clientes de dados e pelo menos um repositório de dados compartilhados.
  - A troca de dados pode ser iniciada pelos clientes de dados ou pelo repositório de dados.

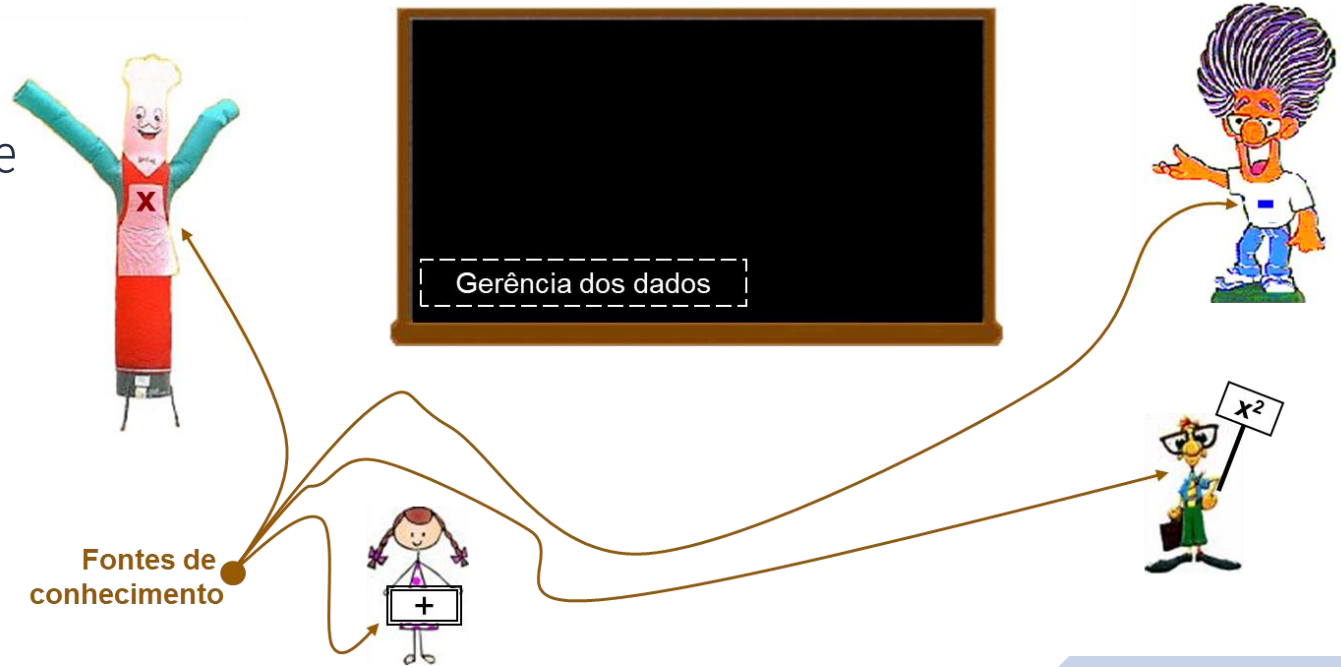


Padrões arquiteturais: Repositório

- Vantagens:
  - Modificabilidade: separa os produtos de dados dos consumidores de dados
  - Desempenho: consolidação dos dados em um único ou poucos locais
  - Consistência dos dados, segurança, privacidade, disponibilidade.
- Desvantagens:
  - Único ponto de falha
  - Gargalo de desempenho

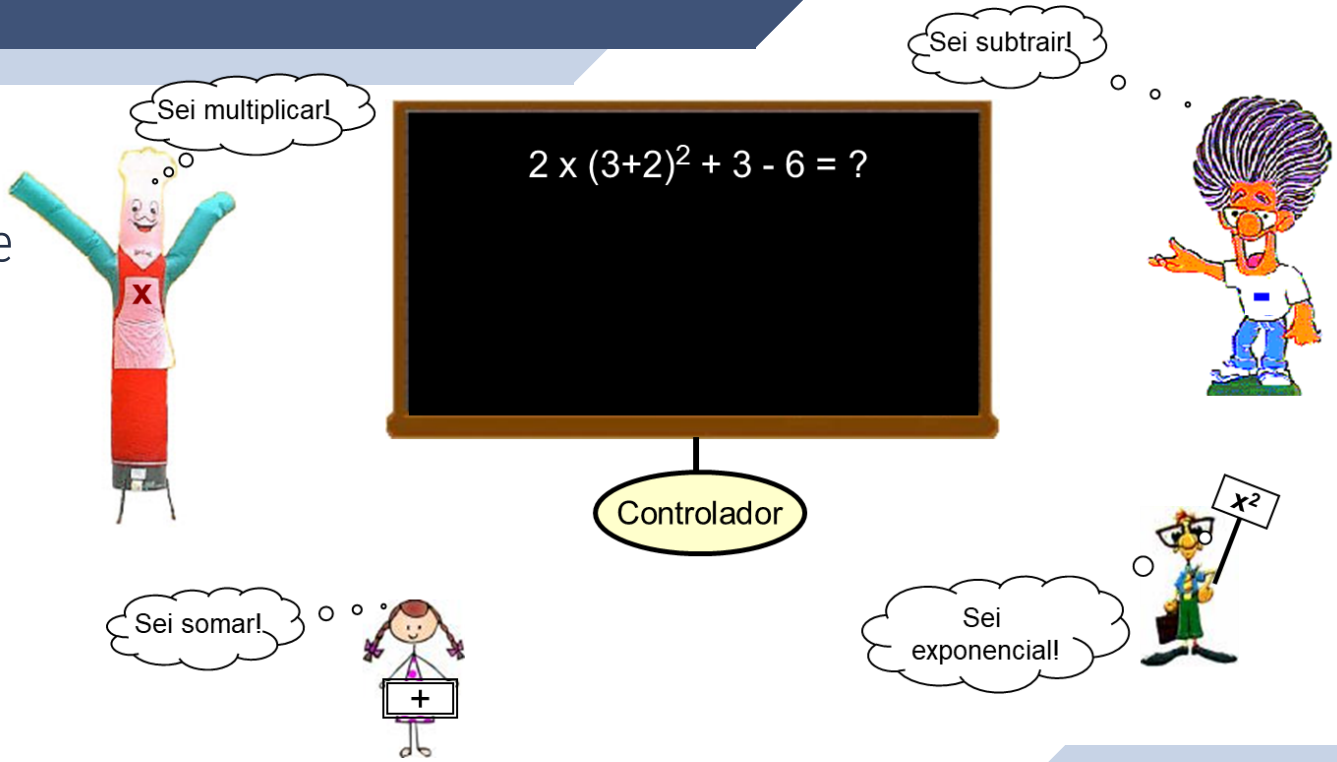
Padrões arquiteturais: Blackboard (Quadro negro)

- Solução:
  - A manipulação de dados é gerenciada pelo repositório
  - Toda operação sobre os dados é iniciada pelo repositório



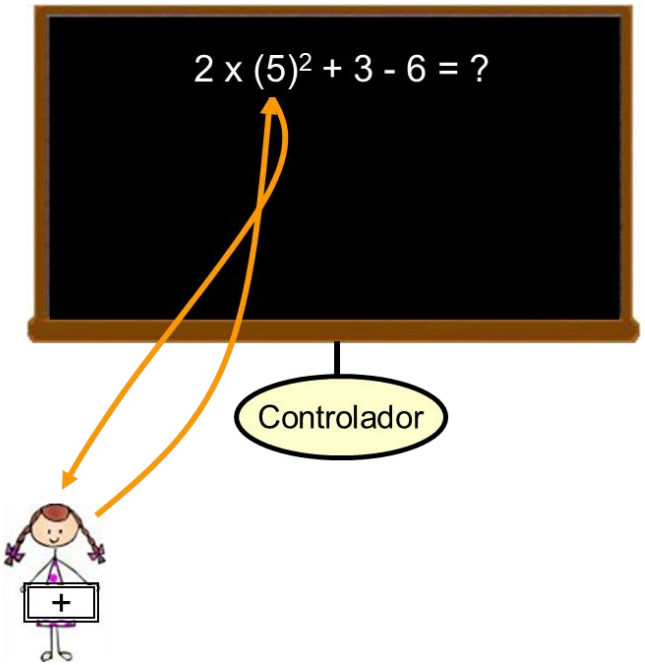
Padrões arquiteturais: Blackboard (Quadro negro)

- Solução:
  - A manipulação de dados é gerenciada pelo repositório
  - Toda operação sobre os dados é iniciada pelo repositório



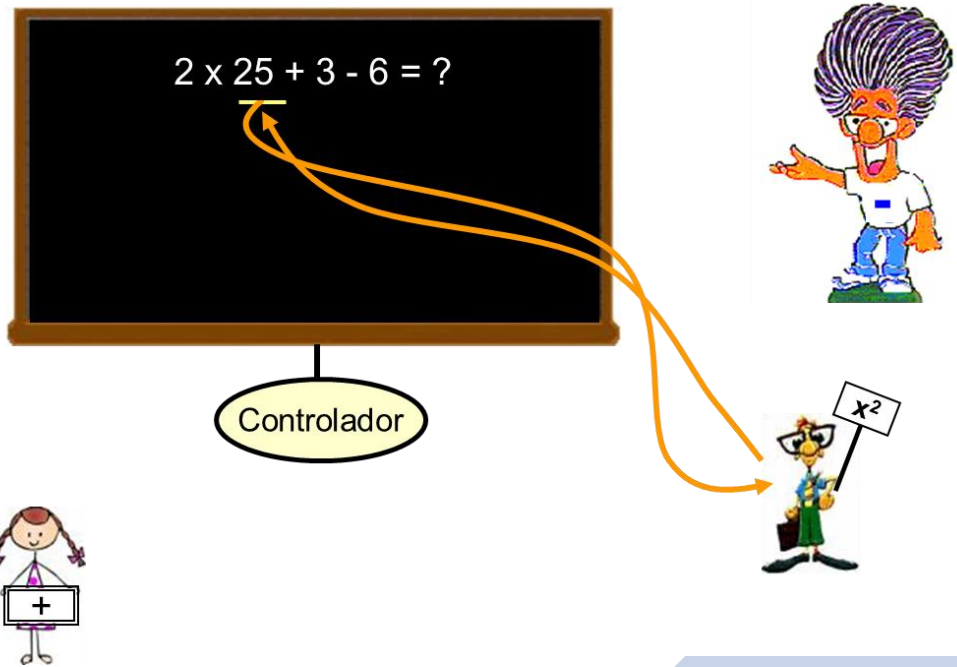
Padrões arquiteturais: Blackboard (Quadro negro)

- Solução:
  - A manipulação de dados é gerenciada pelo repositório
  - Toda operação sobre os dados é iniciada pelo repositório



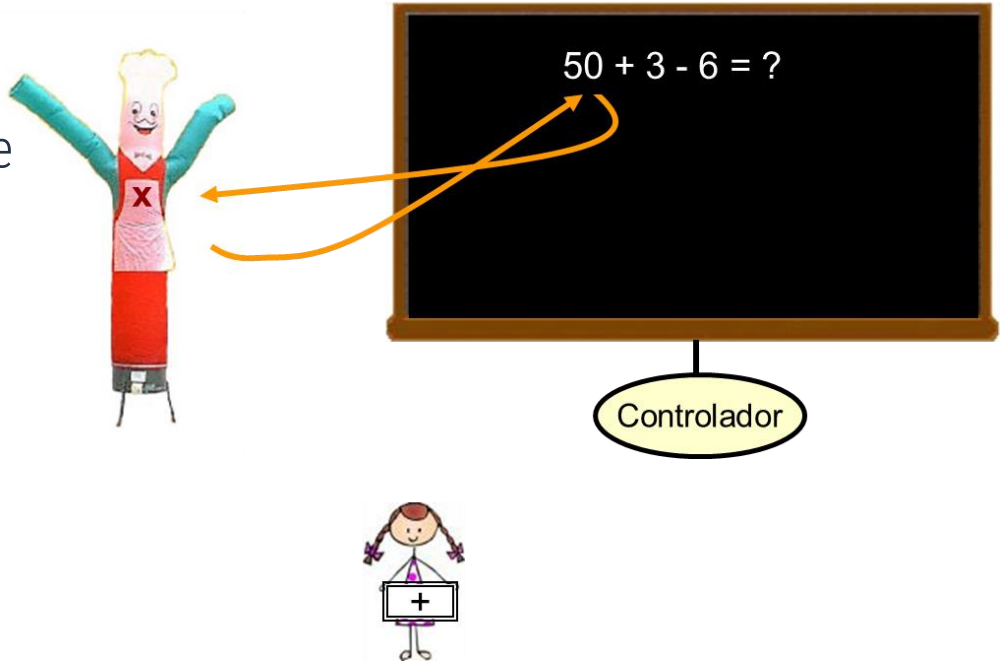
Padrões arquiteturais: Blackboard (Quadro negro)

- Solução:
  - A manipulação de dados é gerenciada pelo repositório
  - Toda operação sobre os dados é iniciada pelo repositório



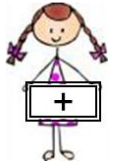
Padrões arquiteturais: Blackboard (Quadro negro)

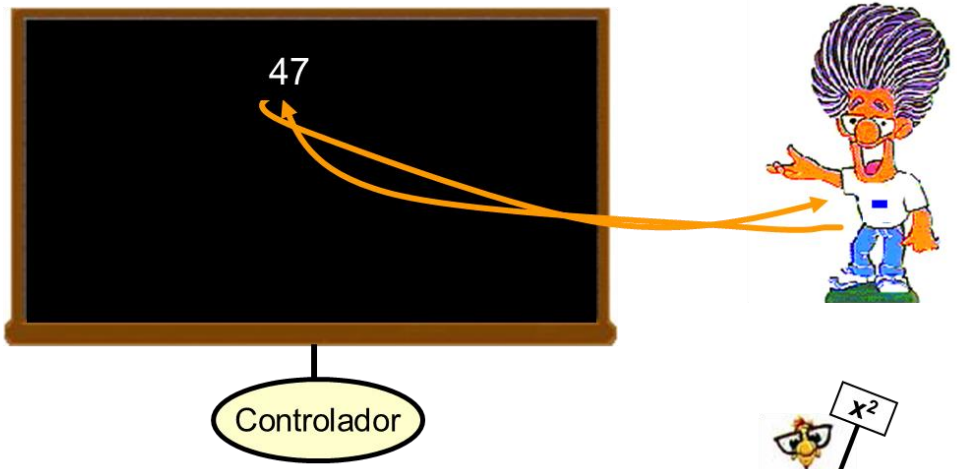
- Solução:
  - A manipulação de dados é gerenciada pelo repositório
  - Toda operação sobre os dados é iniciada pelo repositório



Padrões arquiteturais: Blackboard (Quadro negro)

- Solução:
  - A manipulação de dados é gerenciada pelo repositório
  - Toda operação sobre os dados é iniciada pelo repositório





Padrões arquiteturais: Blackboard (Quadro negro)

- Vantagens:
  - Sistemas complexos
    - Resolução Distribuída de Problemas – RDP
    - Arquitetura usada no paradigma de agentes
  - Aplicações independentes
    - Escalabilidade
- Desvantagens:
  - Ponto de falha - Quadro negro

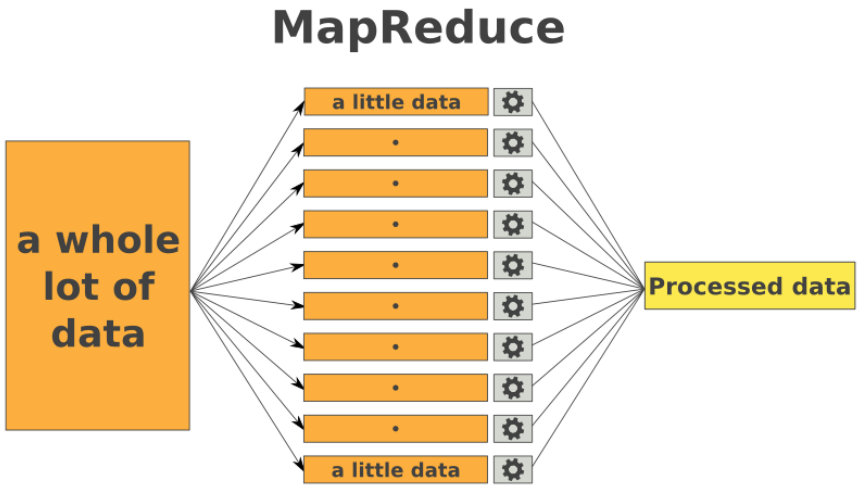
MapReduce

Padrões arquiteturais: MapReduce

- Contexto:
  - Alguns negócios tem a necessidade de analisar rapidamente um grande volume de dados que geram ou acessam.
  - Ex: logs de interação, repositório de dados, weblinks de máquinas de busca
- Problema:
  - Como organizar eficientemente e de forma paralela e distribuída um grande volume de dados e fornecer uma maneira simples para um desenvolvedor analisa-los como desejar.

Padrões arquiteturais: MapReduce

- Solução
  - Uma infraestrutura especializada realiza a alocação de software para componentes de hardware em um ambiente de computação massiva e paralela e organiza os dados
  - Dois componentes tem a função de mapa e redutor, respectivamente.
  - O mapa realiza a extração e transformação de uma porção de dados e o redutor realiza o carregamento dos resultados.



Padrões arquiteturais: MapReduce

- Vantagens:
  - Permite o processamento de um grande volume de dados
  - Baixa latência e alta disponibilidade
- Desvantagens:
  - Se a divisão dos dados não for igualitária, as vantagens do paralelismo podem ser perdidas
  - Operações que requerem múltiplos redutores possuem gerenciamento complexo.

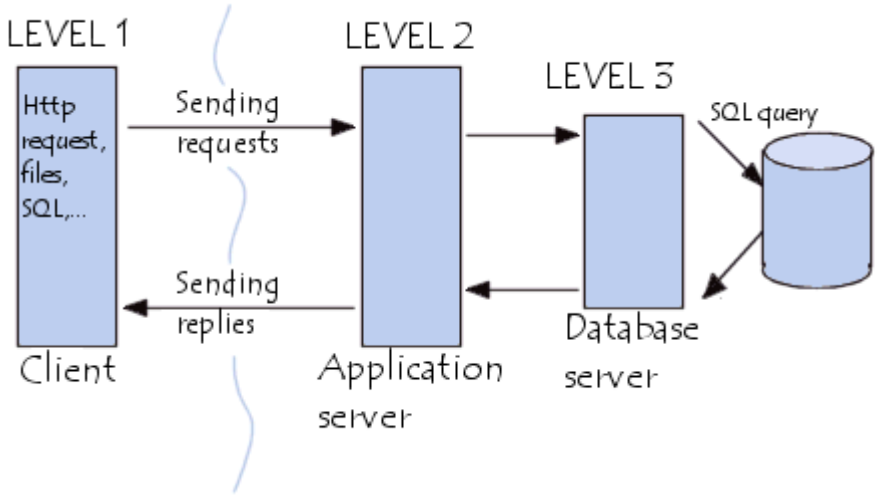
Arquitetura Multinível

Padrões arquiteturais: Arquitetura Multinível

- Contexto:
  - Em um sistema implantando de forma distribuída, é necessário distribuir a infraestrutura do sistema em subgrupos distintos.
  - A distribuição pode ocorrer por questões operacionais ou de negócio
- Problema:
  - Como dividir o sistema em estruturas de execução independentes (hardware + software) conectadas por meios de comunicação?

Padrões arquiteturais: Arquitetura Multinível

- Solução:
  - As estruturas de execução do sistema são organizadas como subgrupos lógicos de componentes
  - Cada grupo é denominado **nível**
  - Critério de agrupamento: tipo de componente, ambiente de execução, propósito do componente.
  - Nível != Camada
  - Deriva do padrão cliente-servidor



Padrões arquiteturais: Arquitetura Multinível

- Vantagens:
  - Facilitam a garantia de segurança
  - Otimizam o desempenho e disponibilidade
- Desvantagens:
  - Alto custo e complexidade

Para pensar....

Os padrões arquiteturais podem ser usados de forma combinada? Se sim, descreva uma combinação que você consegue visualizar ☺

Respondam nos comentários do vídeo

Referências

- Bass, L., Clements, P., & Kazman, R. (2012). *Software architecture in practice*. 3a edição. Addison-Wesley Professional.
- Pressman, R. & Maxim, B. (2016) Engenharia de Software: Uma abordagem professional. 8a edção.

Obrigada