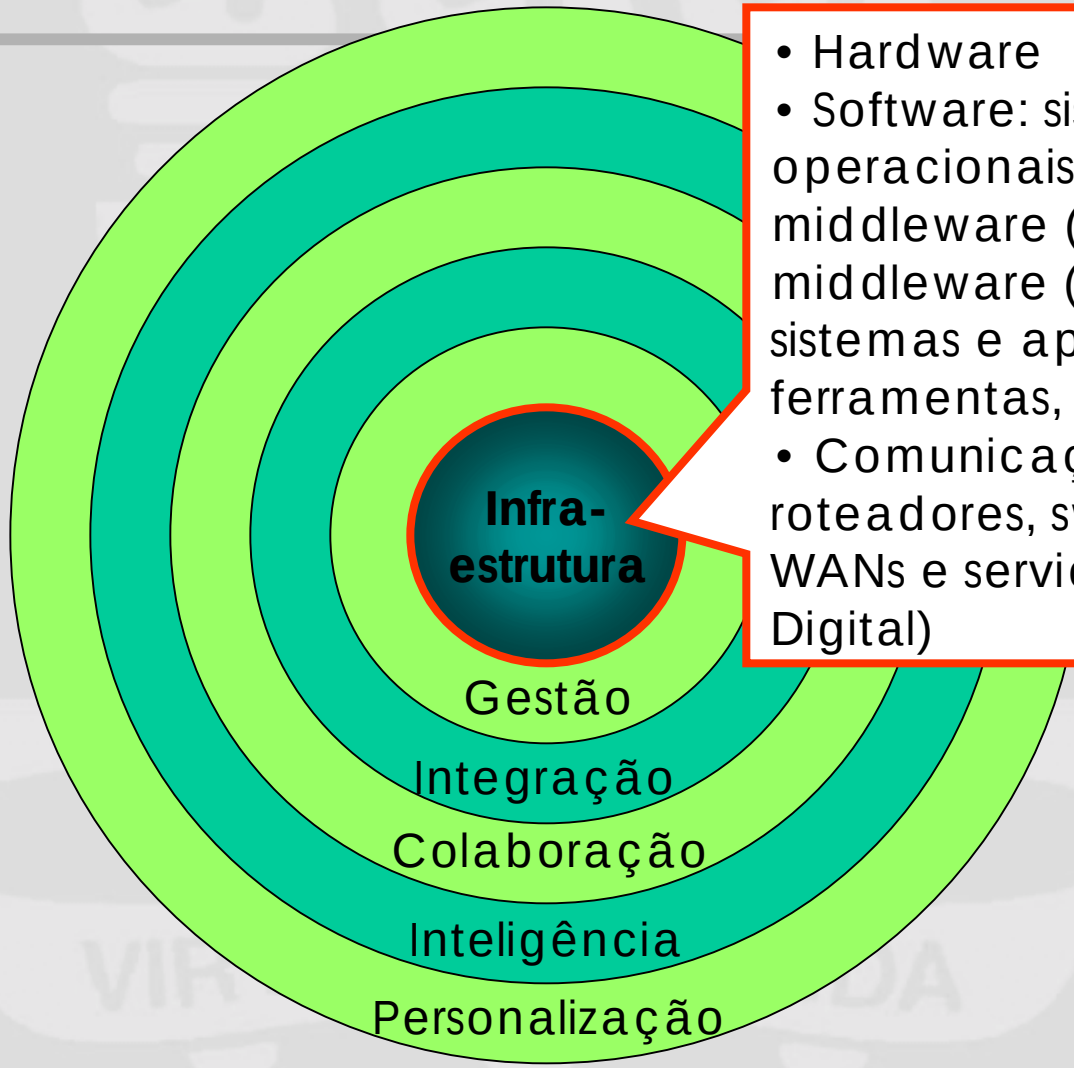
Projeto de Sistemas Distribuídos

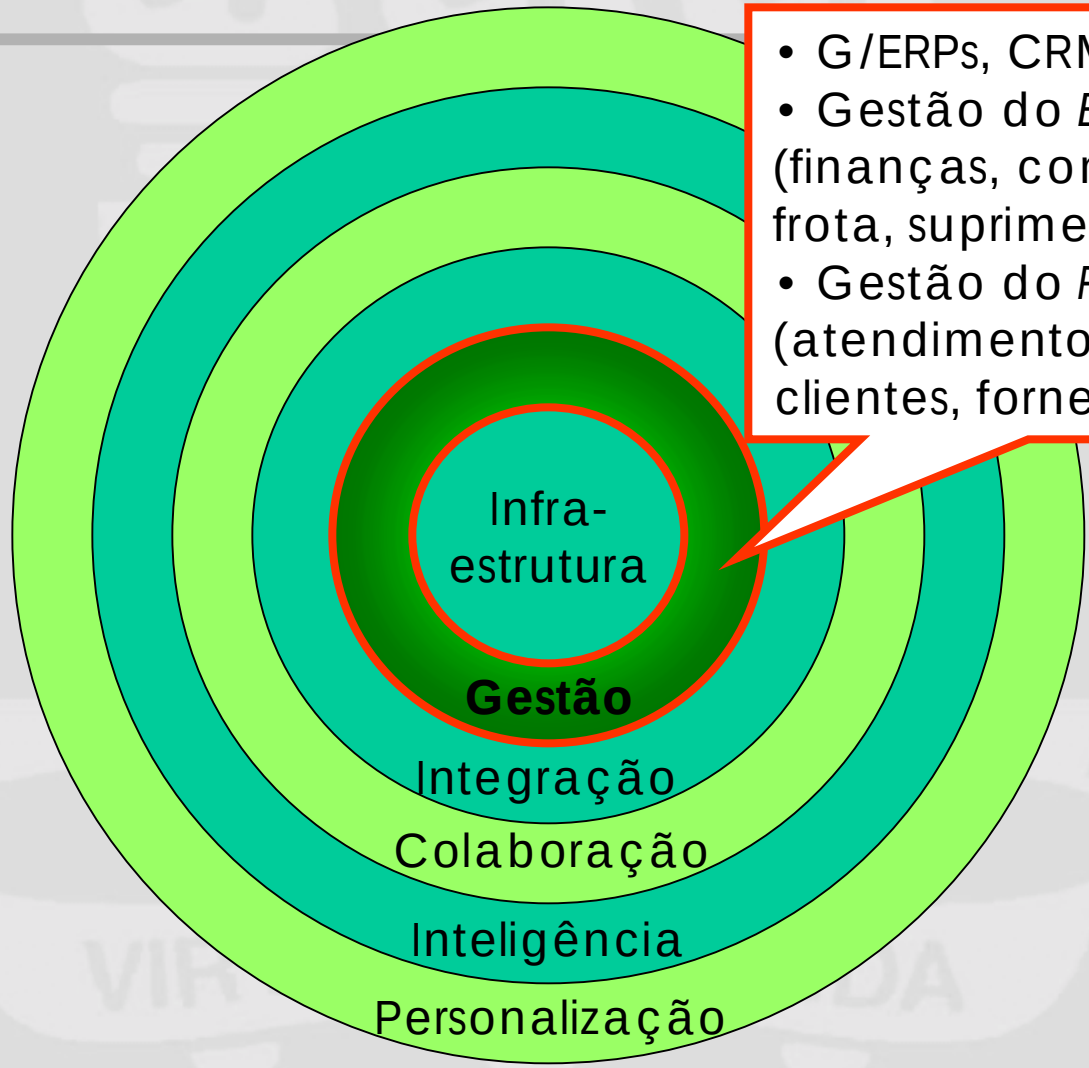
Considerações

Projeto de TI em Camadas



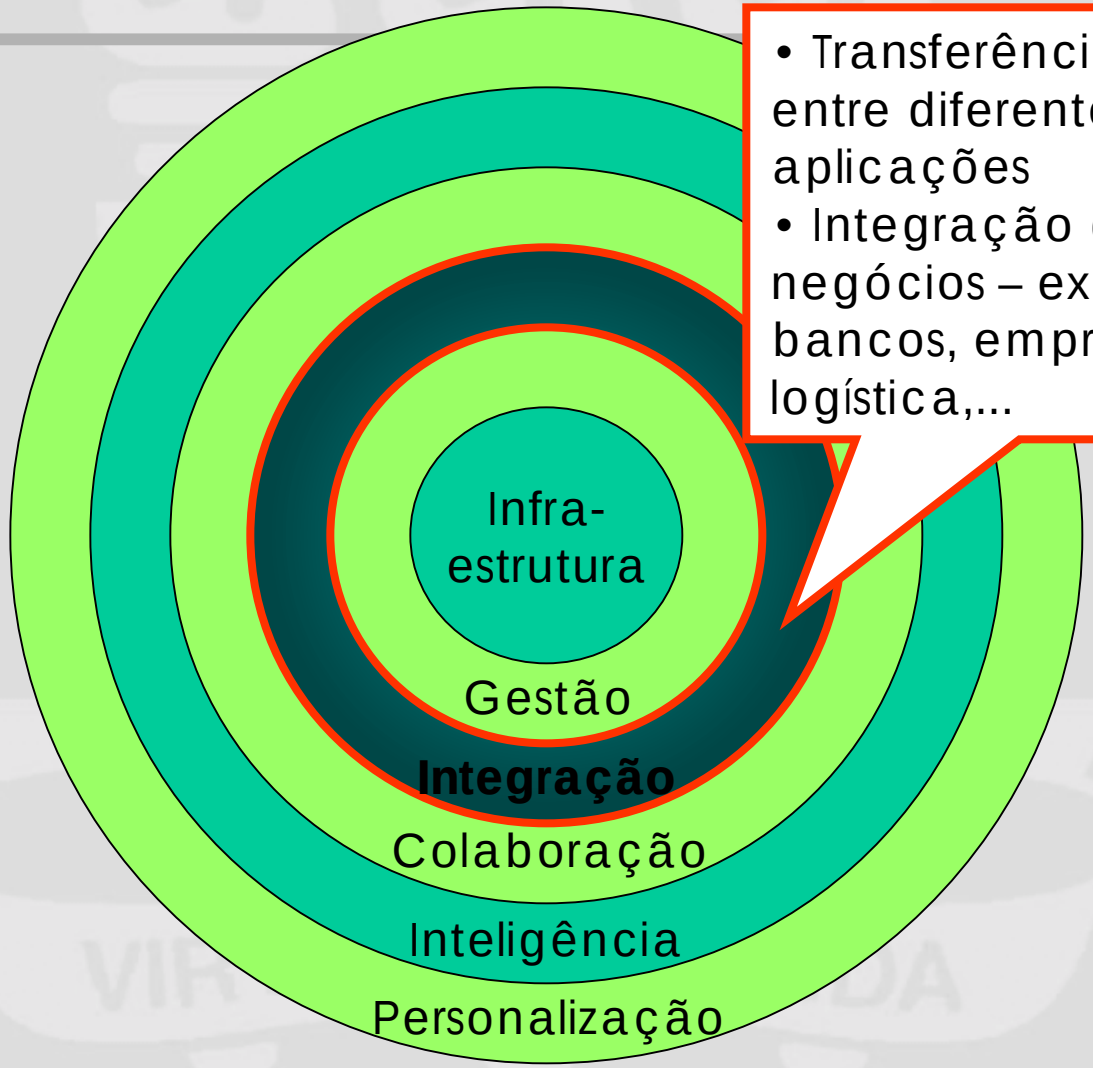
- Hardware
- Software: sistemas operacionais, SGBDs, middleware (serviços), middleware (integração de sistemas e aplicações), ferramentas, servidores,...
- Comunicação: hubs, roteadores, switches, LANs, WANs e serviços (ex. PE-Digital)

TI em Camadas



- G/ERPs, CRMs,...
- Gestão do *Back Office* (finanças, compras, RH, frota, suprimentos,...)
- Gestão do *Front Office* (atendimento a cidadãos, clientes, fornecedores...)

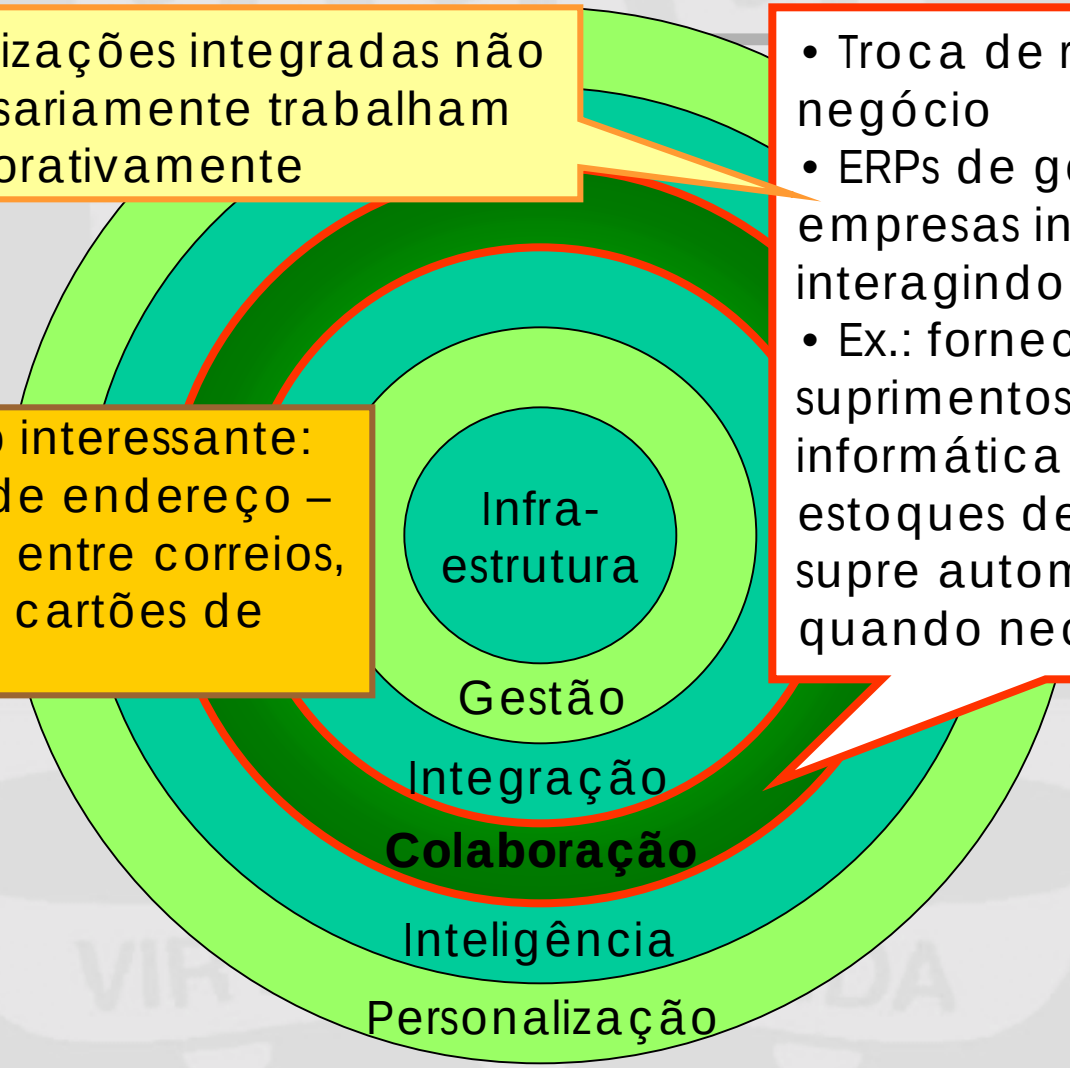
TI em Camadas



- Transferência de dados entre diferentes aplicações
- Integração de negócios – ex.: governo, bancos, empresas de logística,...

TI em Camadas

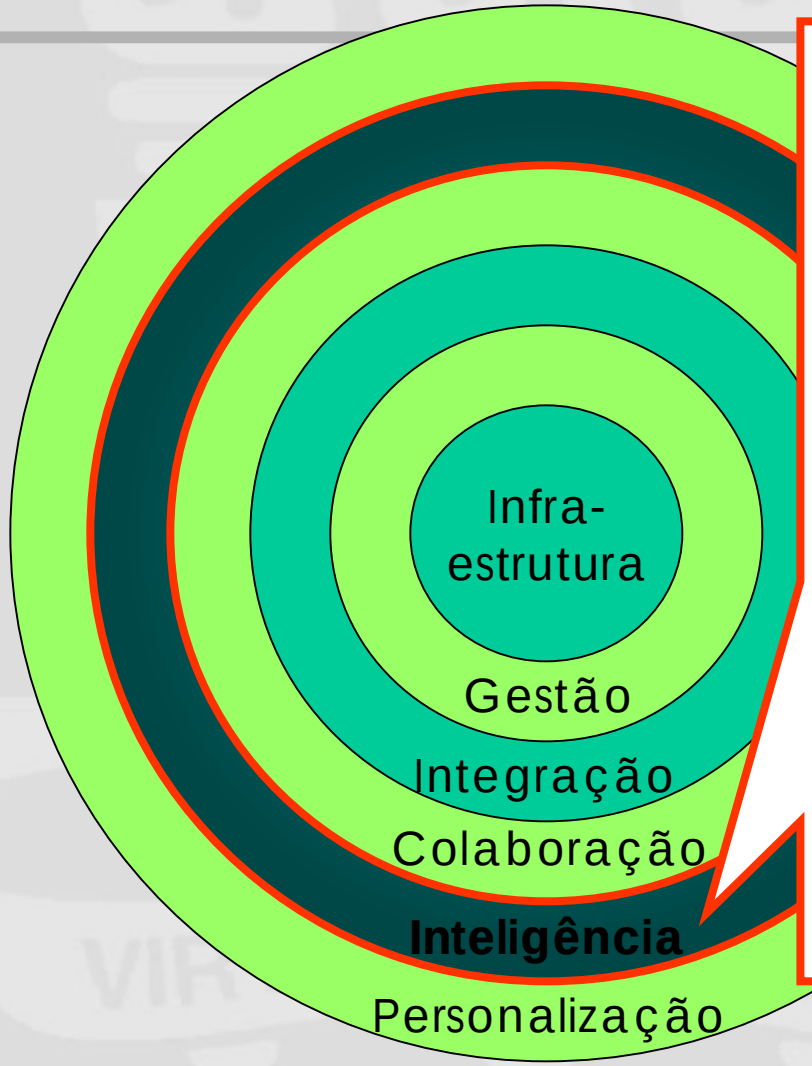
Organizações integradas não necessariamente trabalham colaborativamente



- Troca de regras de negócio
- ERPs de governo e empresas integrados interagindo diretamente
- Ex.: fornecedor de suprimentos de informática monitora estoques de governo e supre automaticamente quando necessário

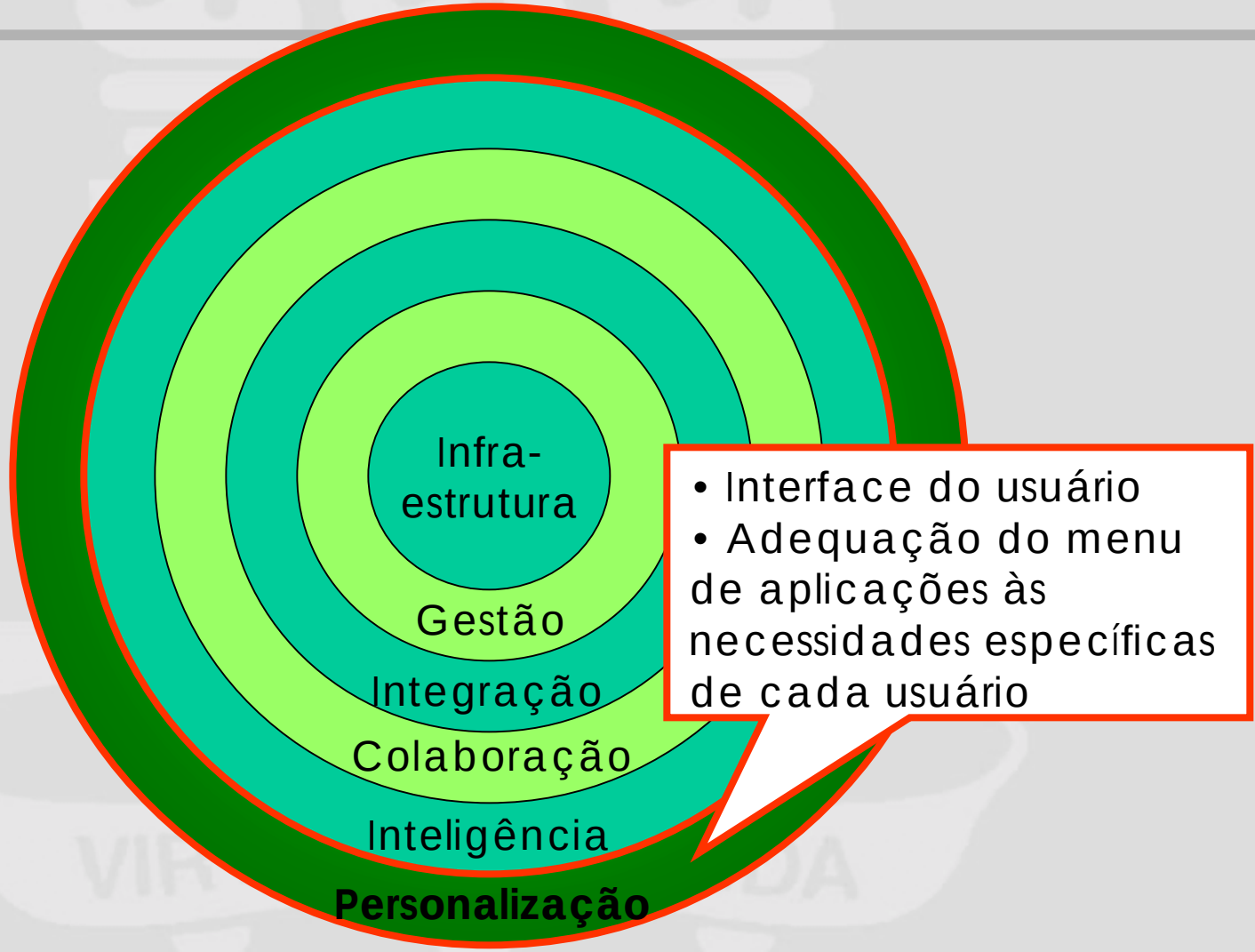
Ex. de serviço interessante: atualização de endereço – colaboração entre correios, detran, adm. cartões de crédito, ...

TI em Camadas



- ***Business Intelligence***: reúne informações críticas e apresenta aos executivos, de maneira rápida, íntegra e simples, os indicadores para tomadas de decisões mais precisas
- Transformação de relatórios analíticos do G/ERP em gráficos para melhor entendimento e tomada de decisão
- Relacionamento de dados, avaliação de tendências,...

TI em Camadas



- Problemas
- Objetivos
- Requisitos de usuário

Como são estruturados?

Problemas-chave

- Nomeação
- Alocação de carga
- Manutenção de consistência
- Comunicação
- Estrutura de software

Problema-chave

- Nomeação
  - *Nome*: interpretado por usuários ou programas
  - *Identificador* : interpretado apenas por programas
  - Nome resolvido (*identificador de comunicação*): traduzido para uma forma que pode ser usada para invocar um recurso ou objeto
  - Nomes dão transparência de localização

Problema-chave

- Alocação de carga
  - Ambiente: cargas mutantes
  - Objetivo: bom desempenho

Problema-chave

- Manutenção de consistência
  - Razões para inconsistência
    - Distribuição (separação) de recursos
    - Concorrência
  - Problemas
    - Dados (atualização, replicação, cache)
    - Falha
    - Relógio
    - Interface de usuário (atrasos na interação)

Consistência de atualização

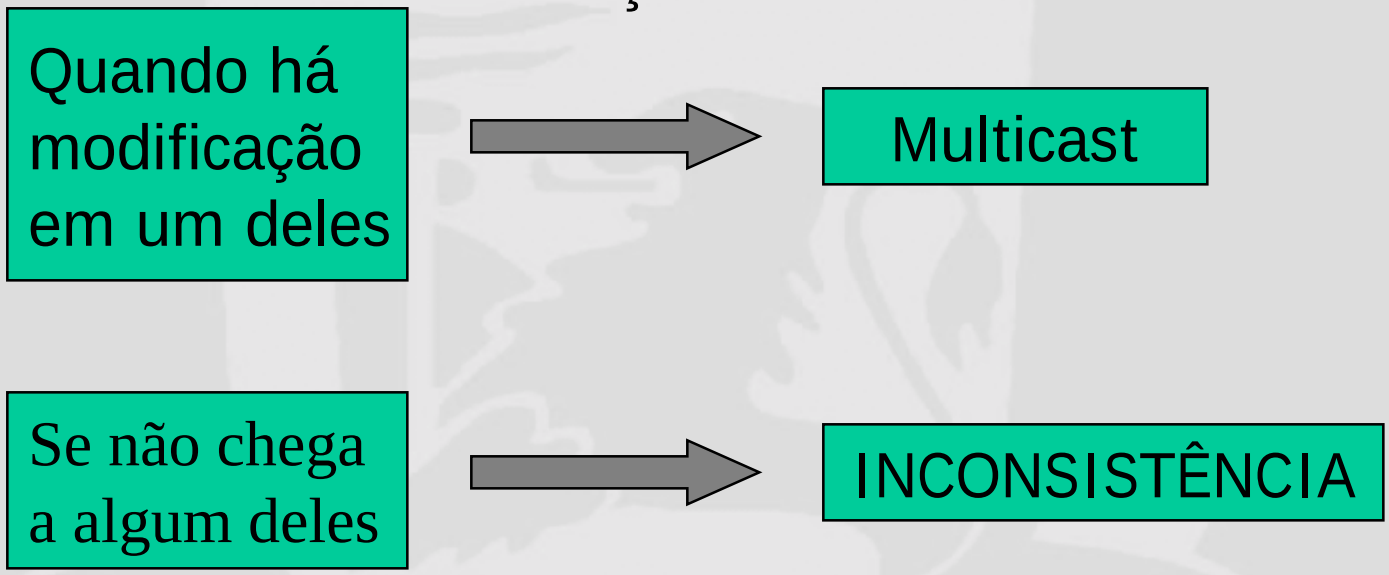
- Perde-se quando a escrita concorrente em dados compartilhados não se realiza como uma única transação atômica
- Solução: transações (ACID)

Exemplo: reserva de trecho de viagem

```
BEGIN_TRANSACTION;
        Reserva (Recife, Salvador);
        Reserva (Salvador, Rio);
        Reserva (Rio, Recife);          ERRO!!!
END_TRANSACTION;
                              ABORT_TRANSACTION
```

Consistência de réplica

- Quando um conjunto de dados deve se manter replicado em várias estações



Ex: jogo multi-usuário em rede

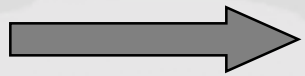
Consistência de *cache*

- Para agilizar o acesso a dados compartilhados

  Memória *cache*

- Quando um cliente modifica a sua *cache*

As cópias dos outros clientes ficam desatualizadas



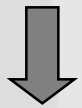
INCONSISTÊNCIA

Consistência de falha

- Uma falha em um sistema centralizado

Todos os programas falham → um único comportamento

- Uma falha em um sistema distribuído



Apenas os componentes que executam no sistema com problema falham → os componentes cooperantes falham posteriormente → comportamentos diversos

Consistência de relógio

- Muitos algoritmos dependem de tempo (*timestamps*)
- Há que sincronizar relógios para a realização consistente de operações (eventos) distribuídas

Consistência de interface de usuário

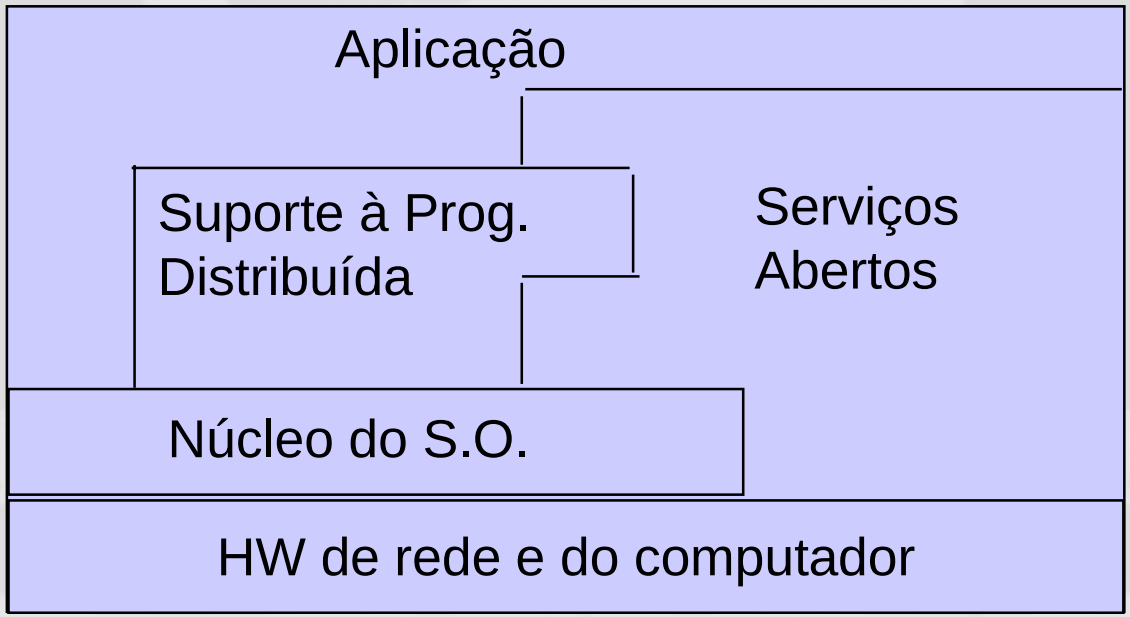
- Exemplo: em uma aplicação distribuída interativa, o usuário aperta um botão e o resultado na tela não aparece de imediato, como se esperaria
  - Pode ter havido um retardo (de comunicação) maior do que o usuário suportaria para ter a impressão de dispor de um sistema dedicado

Problema-chave

- Comunicação
  - Desempenho
    - Atraso
  - Confiabilidade
    - Dado não chega
    - Dado deturpado

Problema-chave

- Estrutura de software



Objetivos

- Desempenho
- Confiabilidade
  - Disponibilidade
  - Segurança
  - Tolerância a falhas
- Escalabilidade
- Consistência
- Transparência

Trade-offs: o projeto de qualquer sistema de computação (SD em particular) envolve compensações

- negociação envolvendo os objetivos
  - Ex: desempenho da comunicação *versus* confiabilidade / segurança (o custo da comunicação)

Requisitos de usuários

- Funcionalidade
- Reconfigurabilidade
- Qualidade de serviço

- *Trade-offs* não devem ser uma preocupação dos usuários
  - usuários devem receber garantias com respeito aos objetivos do projeto e aos requisitos impostos por eles próprios (usuários)
    - Ex: garantias de *consistência* de um serviço de arquivos com respeito a atualizações concorrentes

Requisitos (cont.)

- Funcionalidade: o que o sistema deve fazer pelos usuários - em geral espera-se que um SD traga melhoramentos sobre serviços fornecidos por outros sistemas

- Lembrando do apelo de SD:
  - economia, em função do compartilhamento de recursos
  - melhor desempenho e disponibilidade

Requisitos (cont.)

- Reconfigurabilidade: acomodação de mudanças sem causar interrupções no provimento de serviços existentes

- Qualidade de serviço: envolvendo tópicos como
  - desempenho
  - confiabilidade e disponibilidade
  - segurança