MCTA025-13 - SISTEMAS DISTRIBUÍDOS

TIPOS DE SISTEMAS DISTRIBUÍDOS

Emilio Francesquini

11 de junho de 2018

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

UFABC

DISCLAIMER

- Estes slides foram preparados para o curso de **Sistemas Distribuídos na UFABC**.
- Este material pode ser usado livremente desde que sejam mantidos, além deste aviso, os créditos aos autores e instituições.
- Estes slides foram adaptados daqueles originalmente preparados (e gentilmente cedidos) pelo professor **Daniel Cordeiro, da EACH-USP** que por sua vez foram baseados naqueles disponibilizados online pelos autores do livro "Distributed Systems", 3ª Edição em: `https://www.distributed-systems.net`.

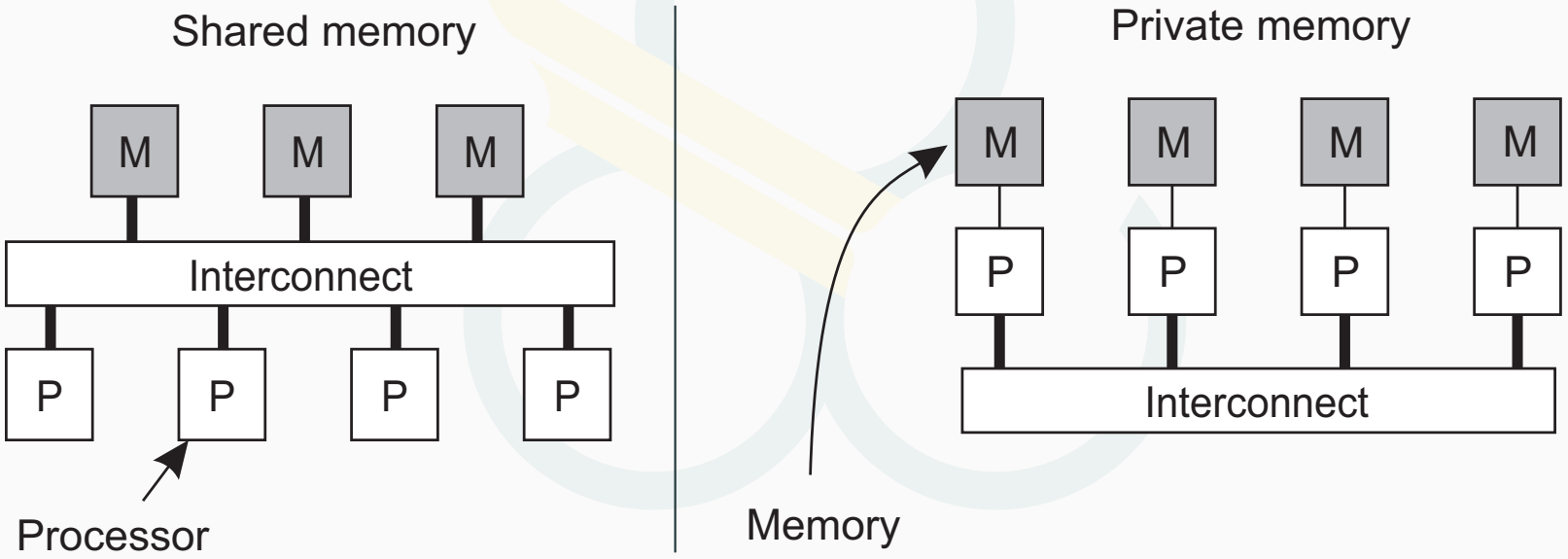
TRÊS TIPOS DE SISTEMAS DISTRIBUÍDOS

- Sistemas para computação distribuída de alto desempenho
- Sistemas de informação distribuídos
- Sistemas distribuídos para computação ubíqua

Observação

A computação distribuída de alto desempenho foi originada na computação paralela

Multiprocessadores e multicores versus multicomputadores



Observação

Multiprocessadores são relativamente fáceis de programar se comparados a multicomputadores, mas ainda assim os problemas aparecem quando o número de processadores (ou cores) aumentam. Solução: tentar implementar um modelo de memória compartilhada para multicomputadores.

Exemplo usando técnicas de memória virtual

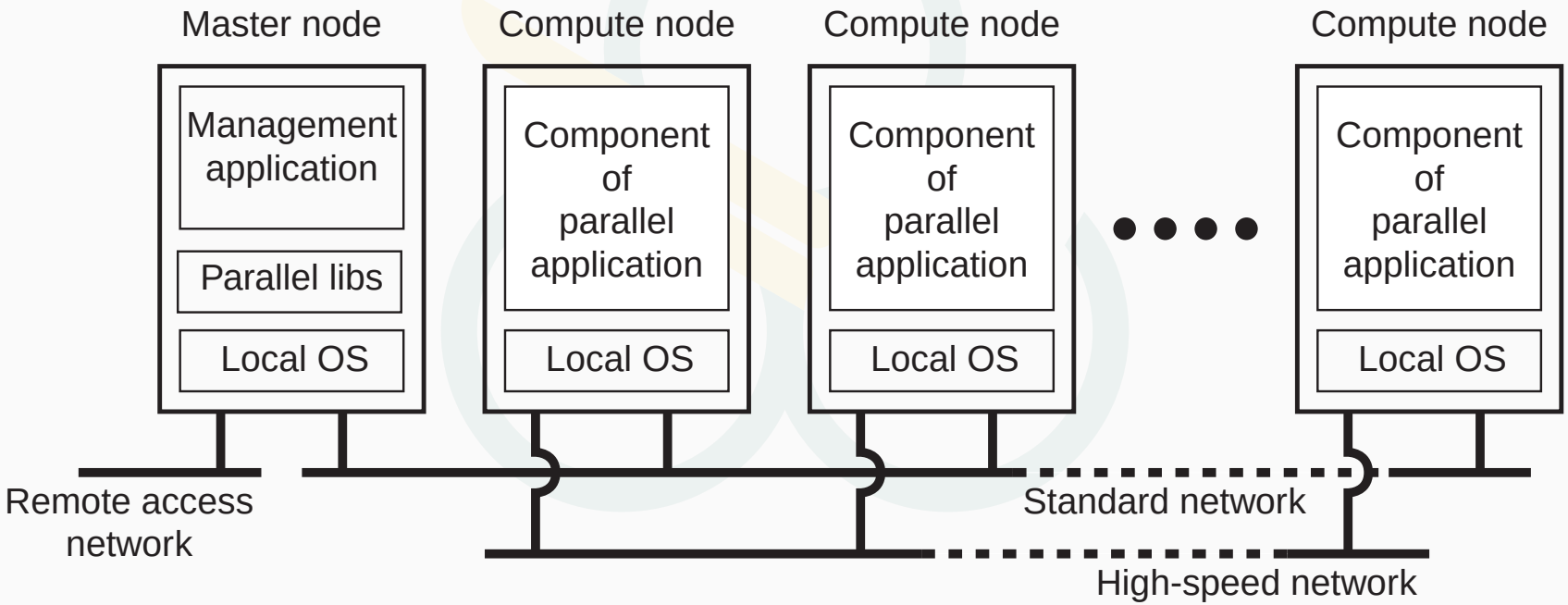
Mapear todas as páginas da memória principal (de todos os diferentes processadores) em um único espaço de endereçamento virtual. Se o processo no processador *A* referenciar uma página *P* localizada no processador *B*, o SO em *A* lança uma interrupção e recupera *P* de *B*, do mesmo modo que faria se *P* estivesse localizado no disco.

Problema

O desempenho de um sistema de memória compartilhada distribuída nunca poderia competir com o desempenho de multiprocessadores e, por isso, a ideia foi abandonada por enquanto.

Essencialmente um grupo de computadores de boa qualidade conectados via LAN

- Homogêneo: mesmo SO, hardware quase idêntico
- Um único nó gerenciador

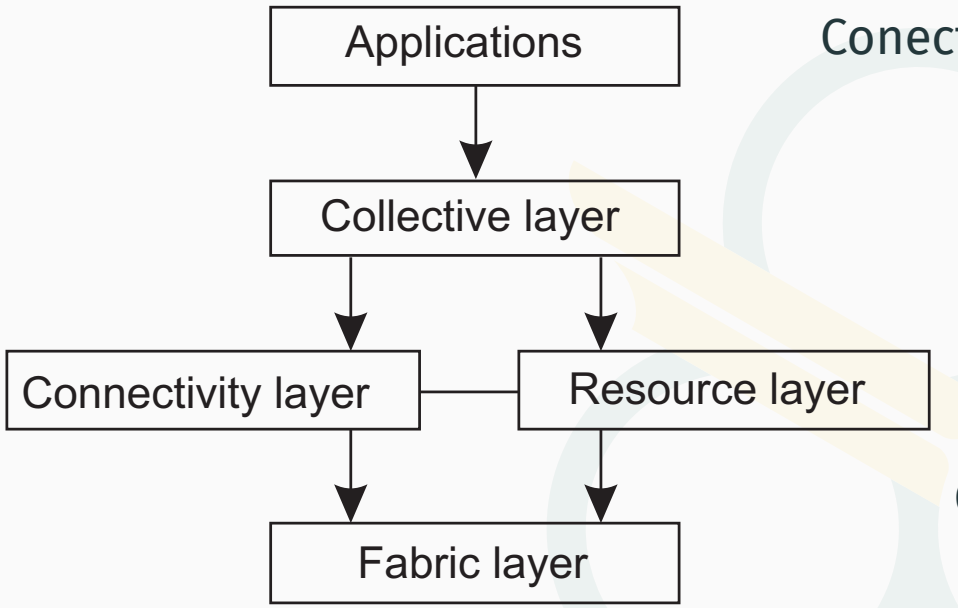


O próximo passo: vários nós vindos de todos os cantos:

- Heterogêneos
- Espalhados entre diversas organizações
- Normalmente formam uma rede de longa distância (*wide-area network*)

Nota:

Para permitir colaborações, grades normalmente usam *organizações virtuais*. Essencialmente, isso significa que os usuários (ou melhor, seus IDs) são organizados em grupos que possuem autorização para usar alguns recursos.



Camadas

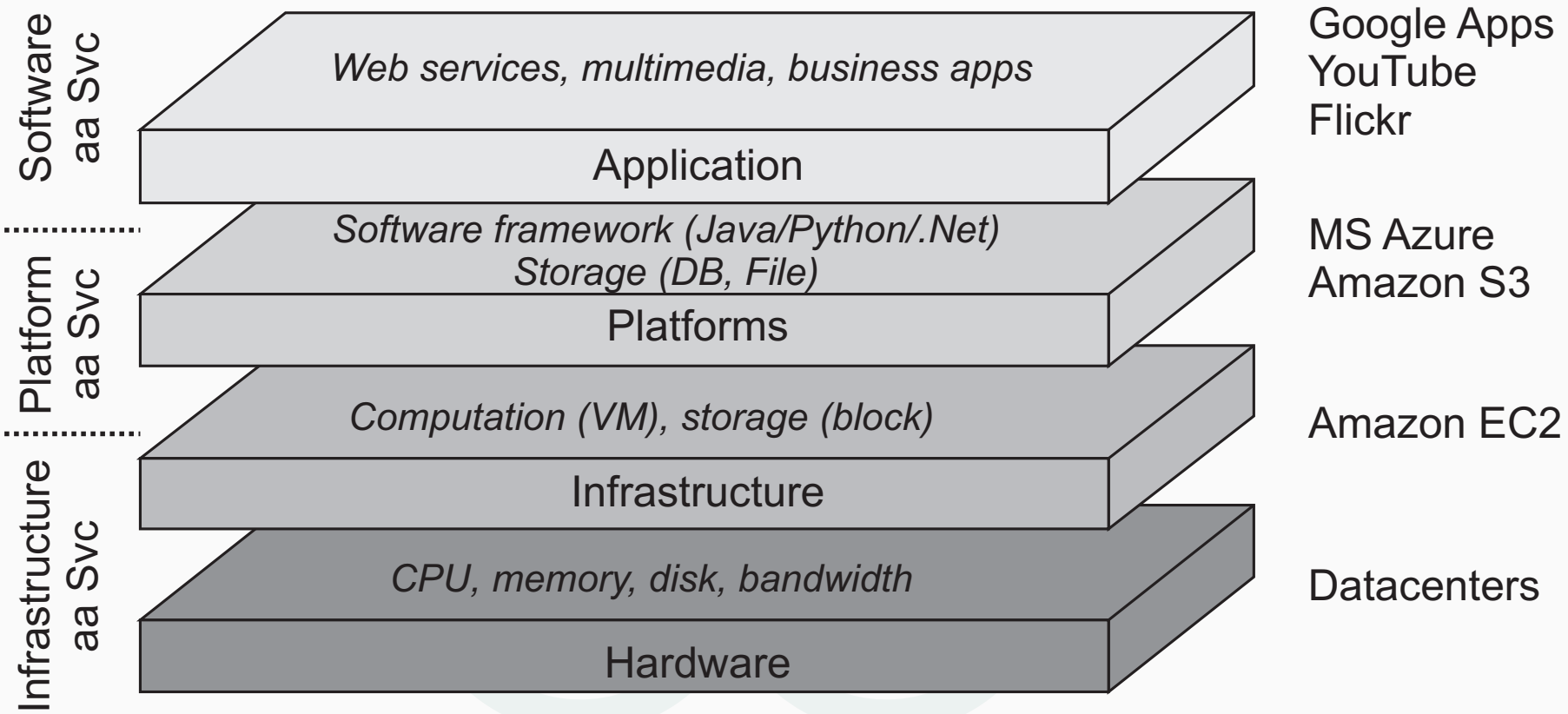
Infraestrutura provê interfaces para os recursos locais

Conectividade protocolos de comunicação/transação e autenticação

Recurso gerencia um único recurso, por exemplo criando processos ou lendo dados

Coletiva realiza acesso à múltiplos recursos: descoberta, escalonamento, replicação

Aplicação contém a aplicação real da grade em uma única organização


Software aa Svc
Platform aa Svc
Infrastructure aa Svc
Web services, multimedia, business apps
Application
Software framework (Java/Python/.Net)
Storage (DB, File)
Platforms
Computation (VM), storage (block)
Infrastructure
CPU, memory, disk, bandwidth
Hardware
Google Apps
YouTube
Flickr
MS Azure
Amazon S3
Amazon EC2
Datacenters

Computação em nuvem

Faz uma distinção entre quatro camadas:

Hardware processadores, roteadores, energia, sistemas de refrigeração

Infraestrutura Utiliza técnicas de virtualização para alocação e gerenciamento de armazenamento e servidores virtuais

Plataforma Provê abstrações de alto nível para os serviços da plataforma. Ex: Amazon S3 para armazenamento de arquivos em *buckets*

Aplicação as aplicações propriamente ditas, tais como as suítes de aplicativos para escritórios.

Observação

Uma razão importante para o sucesso de computação em nuvem é que ela permite que organizações terceirizem sua infraestrutura de TI: hardware e software. A pergunta é: terceirizar é mesmo mais barato?

Abordagem

- Considere *aplicações corporativas*, modeladas como uma coleção de componentes (C_i), cada qual precisando de N_i servidores
- Podemos ver a aplicação como um grafo dirigido, com um vértice representando um componente e um arco $\langle i, \vec{j} \rangle$ representando o fluxo de dados de C_i para C_j.
- Cada arco tem dois pesos associados:
  - $T_{i,j}$, o número de transações por unidade de tempo que causam o fluxo de dados de C_i para C_j
  - $S_{i,j}$, a quantidade de dados total associada a $T_{i,j}$

Plano de migração

Encontre para cada componente C_i, quantos dos n_i dentre seus N_i servidores deveriam migrar, tal que a economia no orçamento menos os cursos de comunicação via Internet sejam maximais.

Requisitos para o plano de migração

1. As restrições impostas pelas políticas devem ser respeitadas
2. Latências adicionais não violarão nenhuma das restrições
3. Todas as transações continuarão a operar corretamente; requisições ou dados não serão perdidos durante a transação

Economia no orçamento

- B_C: economias com a migração de um componente computacionalmente intensivo
- M_C número total de componentes computacionalmente intensivos
- B_S: economias com a migração de um componente intensivo em armazenamento
- M_S número total de componentes intensivos em armazenamento

A economia total, obviamente, é: $B_C \times M_C + B_S \times M_S$.

Tráfego de/para a nuvem

$$Tr_{\text{local, inet}} = \sum_{C_i} (T_{\text{usuário},i} S_{\text{usuário},i} + T_{i,\text{usuário}} S_{i,\text{usuário}})$$

- $T_{\text{usuário},i}$: transações por unidade de tempo que causam fluxo de dados do usuário para C_i
- $S_{\text{usuário},i}$ quantidade de dados associados com $T_{\text{usuário},i}$

TAXA DE TRANSAÇÕES APÓS MIGRAÇÃO

Mais notações:

- $C_{i,\text{local}}$: conjunto de servidores de C_i que continuam executando localmente
- $C_{i,\text{cloud}}$: conjunto de servidores de C_i migrados para o cloud
- Assuma que a distribuição de tráfego é a mesma para o servidor local ou no cloud.

Note que $|C_{i,\text{cloud}} = n_i|$. Seja $f_i = n_i/N_i$ e s_i um servidor de C_i.

$$T^*_{i,j} = \begin{cases} (1-f_i)\cdot(1-f_j)\cdot T_{i,j} & \text{quando } s_i \in C_{i,local} \text{ e } s_j \in C_{j,local} \\ (1-f_i)\cdot f_j \cdot T_{i,j} & \text{quando } s_i \in C_{i,local} \text{ e } s_j \in C_{j,cloud} \\ f_i \cdot (1-f_j)\cdot T_{i,j} & \text{quando } s_i \in C_{i,cloud} \text{ e } s_j \in C_{j,local} \\ f_i \cdot f_j \cdot T_{i,j} & \text{quando } s_i \in C_{i,cloud} \text{ e } s_j \in C_{j,cloud} \end{cases}$$

Observação:

Uma quantidade enorme de sistemas distribuídos em uso hoje em dia são formas de sistemas de informação tradicionais, *integrando* sistemas legados. Exemplo: sistemas de processamento de transações.

```
BEGIN_TRANSACTION(server, transaction)
READ(transaction, file-1, data)
WRITE(transaction, file-2, data)
newData := MODIFIED(data)
IF WRONG(newData) THEN
    ABORT_TRANSACTION(transaction)
ELSE
    WRITE(transaction, file-2, newData)
    END_TRANSACTION(transaction)
END IF
```

Observação:

Uma quantidade enorme de sistemas distribuídos em uso hoje em dia são formas de sistemas de informação tradicionais, *integrando* sistemas legados. Exemplo: sistemas de processamento de transações.

```
BEGIN_TRANSACTION(server, transaction)
READ(transaction, file-1, data)
WRITE(transaction, file-2, data)
newData := MODIFIED(data)
IF WRONG(newData) THEN
    ABORT_TRANSACTION(transaction)
ELSE
    WRITE(transaction, file-2, newData)
    END_TRANSACTION(transaction)
END IF
```

Nota:

Transações formam uma operação atômica.

Uma transação é um conjunto de operações sobre o estado de um objeto (banco de dados, composição de objetos, etc.) que satisfazem as seguintes propriedades (ACID):

Atomicidade ou todas as operações são bem sucedidas, ou todas falham. Quando uma transação falha, o estado do objeto permanecerá inalterado.

Uma transação é um conjunto de operações sobre o estado de um objeto (banco de dados, composição de objetos, etc.) que satisfazem as seguintes propriedades (ACID):

Atomicidade ou todas as operações são bem sucedidas, ou todas falham. Quando uma transação falha, o estado do objeto permanecerá inalterado.

Consistência uma transação estabelece um estado de transição válido. Isto não exclui a existência de estados intermediários inválidos durante sua execução.

Uma transação é um conjunto de operações sobre o estado de um objeto (banco de dados, composição de objetos, etc.) que satisfazem as seguintes propriedades (ACID):

Atomicidade ou todas as operações são bem sucedidas, ou todas falham. Quando uma transação falha, o estado do objeto permanecerá inalterado.

Consistência uma transação estabelece um estado de transição válido. Isto não exclui a existência de estados intermediários inválidos durante sua execução.

Isolamento transações concorrentes não interferem entre si. Para uma transação T é como se as outras transações ocorressem ou *antes* de T, ou *depois* de T.

Uma transação é um conjunto de operações sobre o estado de um objeto (banco de dados, composição de objetos, etc.) que satisfazem as seguintes propriedades (ACID):

Atomicidade ou todas as operações são bem sucedidas, ou todas falham. Quando uma transação falha, o estado do objeto permanecerá inalterado.

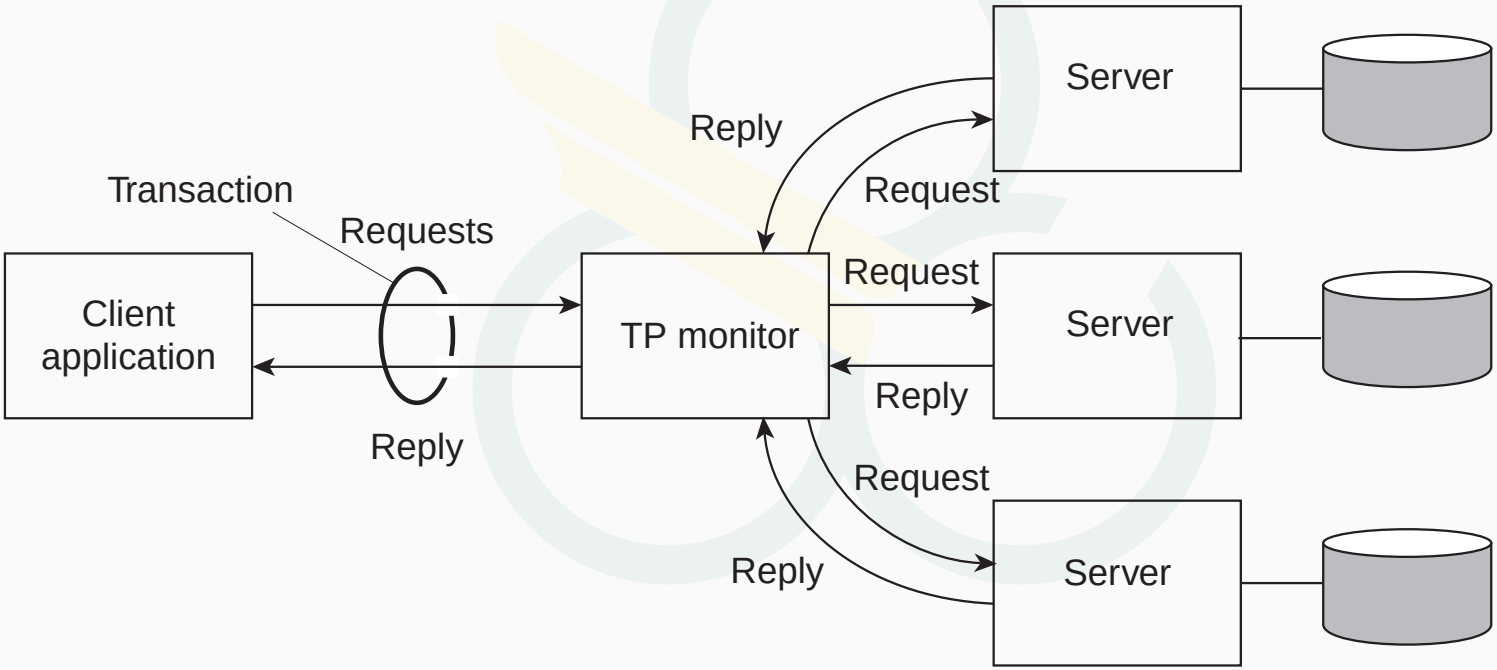
Consistência uma transação estabelece um estado de transição válido. Isto não exclui a existência de estados intermediários inválidos durante sua execução.

Isolamento transações concorrentes não interferem entre si. Para uma transação T é como se as outras transações ocorressem ou *antes* de T, ou *depois* de T.

Durabilidade Após o término de uma transação, seus efeitos são permanentes: mudanças de estado sobrevivem a falhas.

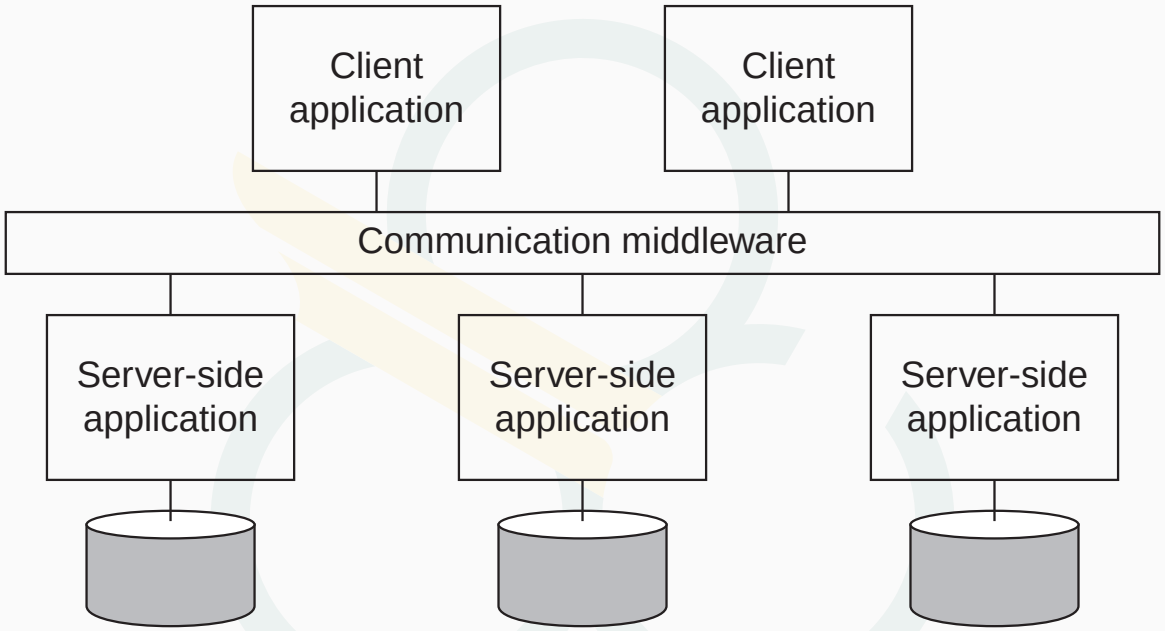
Observação:

Em muitos casos, o conjunto de dados envolvidos em uma transação está distribuído em vários servidores. Um TP Monitor é responsável por coordenar a execução de uma transação.



Problema

Um TP Monitor não basta, também são necessários mecanismos para a comunicação direta entre aplicações.



- Chamada de Procedimento Remoto (RPC)
- Middleware Orientado a Mensagens (MOM)

Tendência em sistemas distribuídos; nós são pequenos, móveis e normalmente embutidos em um sistema muito maior.

Alguns requisitos:

- Mudança contextual: o sistema é parte de um ambiente onde mudanças devem ser rapidamente levadas em consideração
- Composição ad hoc: cada nó pode ser usado em diferentes maneiras, por diferentes usuários. Deve ser facilmente configurável.
- Compartilhar é o padrão: nós vão e vêm, fornecendo serviços e informação compartilháveis. Pede simplicidade.

Tendência em sistemas distribuídos; nós são pequenos, móveis e normalmente embutidos em um sistema muito maior.

Alguns requisitos:

- Mudança contextual: o sistema é parte de um ambiente onde mudanças devem ser rapidamente levadas em consideração
- Composição ad hoc: cada nó pode ser usado em diferentes maneiras, por diferentes usuários. Deve ser facilmente configurável.
- Compartilhar é o padrão: nós vão e vêm, fornecendo serviços e informação compartilháveis. Pede simplicidade.

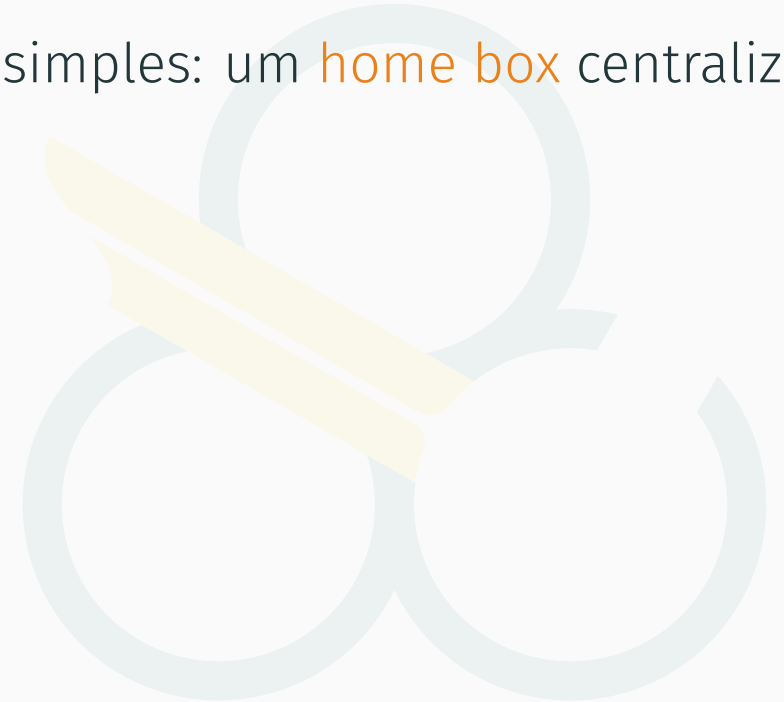
Nota:

Ubiquidade e transparência de distribuição formam um bom par?

Sistemas domésticos

Devem ser completamente auto-organizáveis:

- Não deve haver um administrador do sistema
- Solução mais simples: um home box centralizado?



Sistemas domésticos

Devem ser completamente auto-organizáveis:

- Não deve haver um administrador do sistema
- Solução mais simples: um home box centralizado?

Monitorando uma pessoa

Dispositivos ficam fisicamente próximos a uma pessoa:

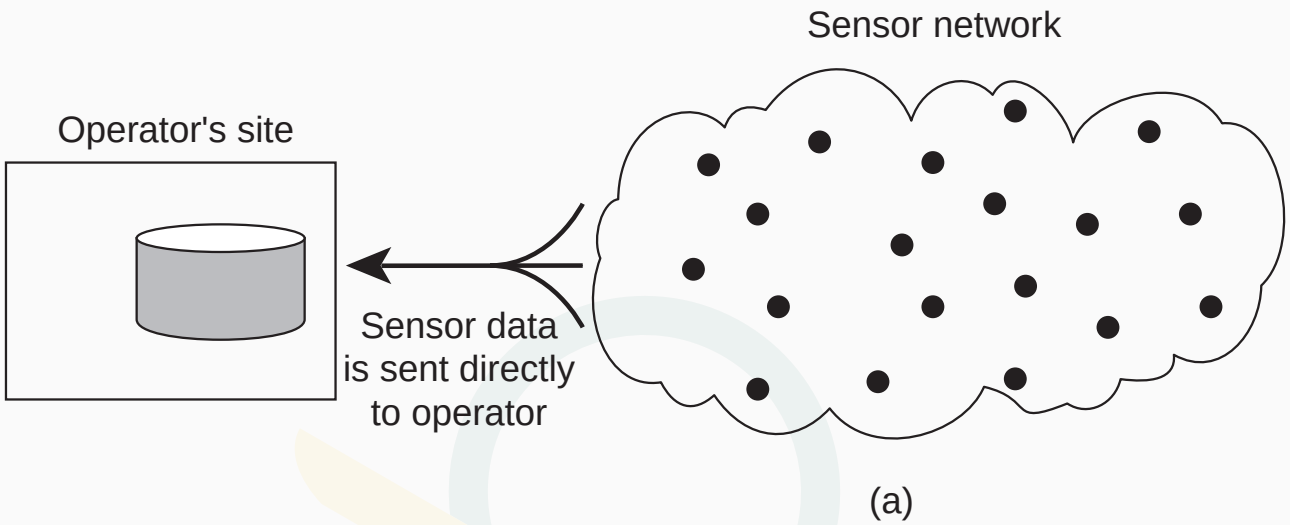
- Onde e como são armazenados os dados monitorados?
- Podemos prevenir perda de dados importantes?
- Há necessidade de gerar e propagar alertas?
- Como fazer para garantir segurança?
- Como o ambiente pode prover *feedback* online?

Características

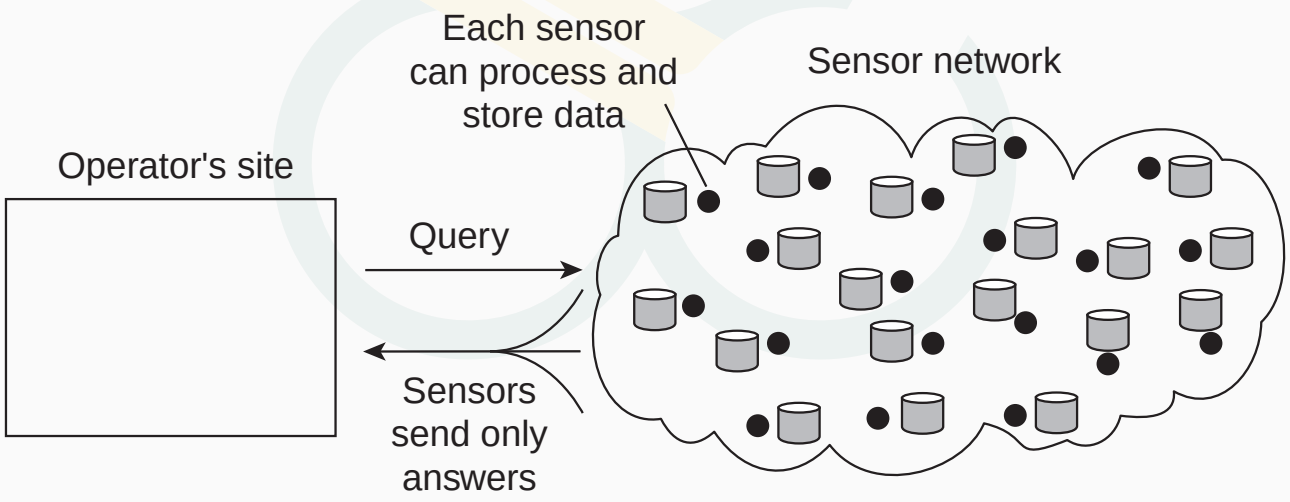
Os nós aos quais os sensores estão presos são:

- Muitos (10s–1000s)
- Simples (pouca capacidade de memória/computação/comunicação)
- Normalmente necessitam de uma bateria

REDES DE SENSORES COMO UM SISTEMA DISTRIBUÍDO



(a)



(b)

Gerenciamento de multidões

- Situação: um grande evento sem rotas fixas (exposições, festivais, etc.)
- Objetivo: guiar as pessoas de acordo com suas posições sociais:
  - direcionar pessoas com interesses similares para os mesmos locais
  - direcionar membros de um grupo para uma mesma saída no caso de uma emergência
- Objetivo: manter grupos unidos (p.ex.:, famílias)

Estimulando a mistura

- Situação: conferência com pessoas de diferentes grupos
- Objetivo: estimular pessoas de diferentes grupos a interagirem.
- Abordagem: acompanhar as interações entre os grupos:
  - Quando um aluno de BC&T fala com um aluno de BC&H: pontos de bônus para os dois alunos e para os seus respectivos grupos.
  - Pontos para o grupo são distribuídos entre os seus membros
  - Conquistas são mostradas em crachás eletrônicos (*feedback* e *intervenções sociais*)

CENÁRIOS DE APLICAÇÃO: JOGOS SOCIAIS