Distributed Systems
Principles and Paradigms

Maarten van Steen

VU Amsterdam, Dept. Computer Science
(Tradução e Adaptação Ricardo Anido - IC/Unicamp)

Capítulo 01: Introdução

Versão: 27 de fevereiro de 2014

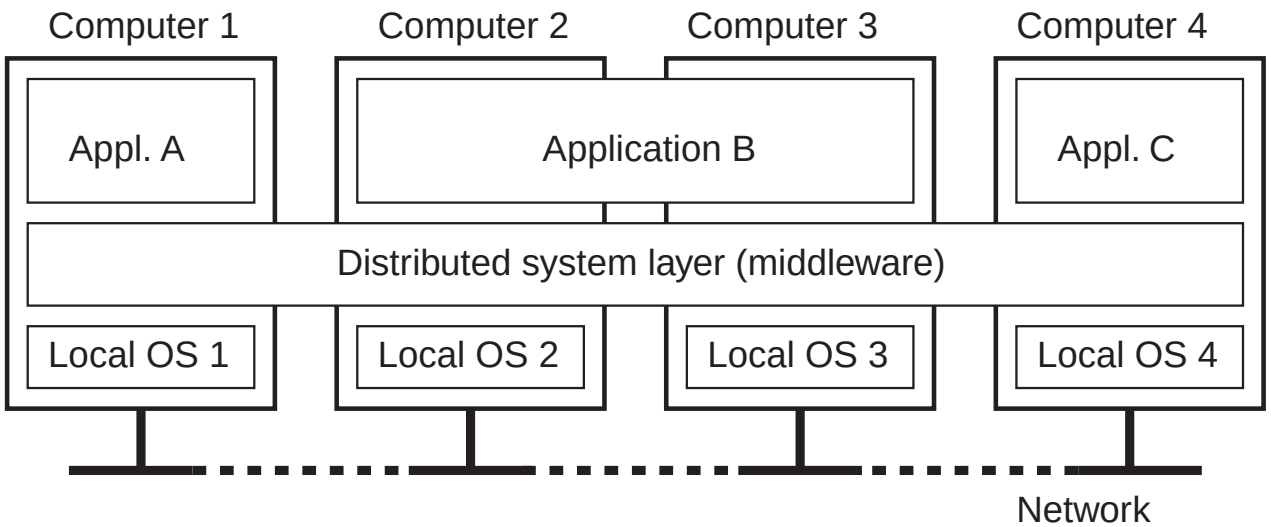
Conteúdo

Sistemas Distribuídos: definições

Um sistema distribuído é um pedaço de software que garante que:

> *uma coleção de computadores independentes aparece para seus usuários como um único sistema integrado*

Dois aspectos: (1) computadores independentes e
(2) sistema único $\Rightarrow$ middleware.



Metas de Sistemas Distribuídos

- Fazer com que os recursos estejam disponíveis
- Transparência na distribuição
- Abertura
- Escalabilidade

Transparência de Distribuição

Transparência	Distribuição
Accesso	Esconde diferenças na representação de dados e mecanismos de invocação
Localidade	Esconde onde os objetos residem
Migração	Esconde de um objeto a habilidade de o sistema alterar a localização daquele objeto
Relocação	Esconde de um cliente a habilidade de o sistema alterar a localização do objeto ao qual o cliente está conectado
Replicação	Esconde o fato de que um objeto ou seu estado pode estar replicado e que estas réplicas podem residir em locais distintos
Concorrência	Esconde a coordenação de atividades entre objetos para garantir consistência
Falhas	Esconde falhas e possíveis recuperações de objetos

Nota

Transparência de distribuição é uma meta boa, mas atingi-la não é fácil ou barato.

Transparência de Distribuição

Transparência	Distribuição
Accesso	Esconde diferenças na representação de dados e mecanismos de invocação
Localidade	Esconde onde os objetos residem
Migração	Esconde de um objeto a habilidade de o sistema alterar a localização daquele objeto
Relocação	Esconde de um cliente a habilidade de o sistema alterar a localização do objeto ao qual o cliente está conectado
Replicação	Esconde o fato de que um objeto ou seu estado pode estar replicado e que estas réplicas podem residir em locais distintos
Concorrência	Esconde a coordenação de atividades entre objetos para garantir consistência
Falhas	Esconde falhas e possíveis recuperações de objetos

Nota

Transparência de distribuição é uma meta boa, mas atingi-la não é fácil ou barato.

Grau de Transparência

Observação

Almejar transparência completa pode ser excessivo:

- Usuários podem estar em continentes diferentes
- Esconder falhas de rede e de nós é (na teoria e na prática) impossível
  - Não é possível distinguir entre um computador lento e um falho
  - Não é possível ter certeza de que um servidor executou uma operação antes de falhar
- Transparência total tem custo no desempenho
  - Mantendo caches Web exatamente atualizados com a cópia principal
  - Forçando escrita de disco (flush) para tolerância a falhas

Grau de Transparência

Observação

Almejar transparência completa pode ser excessivo:

- Usuários podem estar em continentes diferentes
- Esconder falhas de rede e de nós é (na teoria e na prática) impossível
  - Não é possível distinguir entre um computador lento e um falho
  - Não é possível ter certeza de que um servidor executou uma operação antes de falhar
- Transparência total tem custo no desempenho
  - Mantendo caches Web exatamente atualizados com a cópia principal
  - Forçando escrita de disco (flush) para tolerância a falhas

Grau de Transparência

Observação

Almejar transparência completa pode ser excessivo:

- Usuários podem estar em continentes diferentes
- Esconder falhas de rede e de nós é (na teoria e na prática) impossível
  - Não é possível distinguir entre um computador lento e um falho
  - Não é possível ter certeza de que um servidor executou uma operação antes de falhar
- Transparência total tem custo no desempenho
  - Mantendo caches Web exatamente atualizados com a cópia principal
  - Forçando escrita de disco (flush) para tolerância a falhas

Grau de Transparência

Observação

Almejar transparência completa pode ser excessivo:

- Usuários podem estar em continentes diferentes
- Esconder falhas de rede e de nós é (na teoria e na prática) impossível
  - Não é possível distinguir entre um computador lento e um falho
  - Não é possível ter certeza de que um servidor executou uma operação antes de falhar
- Transparência total tem custo no desempenho
  - Mantendo caches Web exatamente atualizados com a cópia principal
  - Forçando escrita de disco (flush) para tolerância a falhas

Sistemas Distribuídos Abertos

Sistemas Distribuídos Abertos

Ser capaz de interagir com serviços de outros sistemas abertos, independente do ambiente subjacente:

- Sistemas devem ter interfaces bem definidas
- Sistemas devem permitir portabilidade de aplicações
- Sistemas devem interoperar facilmente

Atingindo abertura

Ao menos tornar o sistema distribuído independente da heterogeneidade do ambiente subjacente:

- Hardware
- Platformas
- Linguagens

Sistemas Distribuídos Abertos

Sistemas Distribuídos Abertos

Ser capaz de interagir com serviços de outros sistemas abertos, independente do ambiente subjacente:

- Sistemas devem ter interfaces bem definidas
- Sistemas devem permitir portabilidade de aplicações
- Sistemas devem interoperar facilmente

Atingindo abertura

Ao menos tornar o sistema distribuído independente da heterogeneidade do ambiente subjacente:

- Hardware
- Platformas
- Linguagens

Políticas versus Mecanismos

Implementação de abertura

Requer apoio a diferentes políticas:

- Que nível de consistência é necessário para dados de caches de clientes?
- Que operações são permitidas para códigos baixados da rede?
- Quais requisitos de QoS devem ser ajustados se houver variação de banda?
- Qual o nível de sigilo é necessário/suficiente para comunicação?

Implementação de abertura

Idealmente, um sistema distribuído provê apenas mecanismos:

- Permite alteração (dinâmica) de políticas de cache
- Permite diferentes níveis de confiança em código móvel
- Permite ajustar parâmetros QoS por canal de dados
- Oferece algorítmos de criptografia distintos

Políticas versus Mecanismos

Implementação de abertura

Requer apoio a diferentes políticas:

- Que nível de consistência é necessário para dados de caches de clientes?
- Que operações são permitidas para códigos baixados da rede?
- Quais requisitos de QoS devem ser ajustados se houver variação de banda?
- Qual o nível de sigilo é necessário/suficiente para comunicação?

Implementação de abertura

Idealmente, um sistema distribuído provê apenas mecanismos:

- Permite alteração (dinâmica) de políticas de cache
- Permite diferentes níveis de confiança em código móvel
- Permite ajustar parâmetros QoS por canal de dados
- Oferece algorítmos de criptografia distintos

Escala em Sistemas Distribuídos

Observação

Muitos desenvolvedores de sistemas distribuídos modernos usam o adjetivo “scalable” sem especificar claramente o **por que** seus sistemas escalam.

Escalabilidade

Ao menos três componentes:

- Número da usuários e/ou processos (escalabilidade de tamanho)
- Distância máxima entre nós (escalabilidade geográfica)
- Número de domínios possíveis (escalabilidade administrativa)

Observação

A maioria dos sistemas leva em conta, e até um certo ponto, escalabilidade de tamanho. Hoje em dia, os desafios estão em escalabilidade geográfica e administrativa.

Escala em Sistemas Distribuídos

Observação

Muitos desenvolvedores de sistemas distribuídos modernos usam o adjetivo “scalable” sem especificar claramente o **por que** seus sistemas escalam.

Escalabilidade

Ao menos três componentes:

- Número da usuários e/ou processos (escalabilidade de tamanho)
- Distância máxima entre nós (escalabilidade geográfica)
- Número de domínios possíveis (escalabilidade administrativa)

Observação

A maioria dos sistemas leva em conta, e até um certo ponto, escalabilidade de tamanho. Hoje em dia, os desafios estão em escalabilidade geográfica e administrativa.

Escala em Sistemas Distribuídos

Observação

Muitos desenvolvedores de sistemas distribuídos modernos usam o adjetivo “scalable” sem especificar claramente o **por que** seus sistemas escalam.

Escalabilidade

Ao menos três componentes:

- Número da usuários e/ou processos (escalabilidade de tamanho)
- Distância máxima entre nós (escalabilidade geográfica)
- Número de domínios possíveis (escalabilidade administrativa)

Observação

A maioria dos sistemas leva em conta, e até um certo ponto, escalabilidade de tamanho. Hoje em dia, os desafios estão em escalabilidade geográfica e administrativa.

Técnicas para Escalabilidade

Esconder latências de comunicação

Evitar esperar por respostas; fazer outra coisa:

- Usar comunicação assíncrona
- Ter um tratador separado para respostas que cheguem
- Problema: nem toda aplicação se encaixa nesse modelo

Técnicas para Escalabilidade

Distribuição

Particionar dados e computações entre múltiplas máquinas:

- Mover computações para os clientes (Java applets)
- Serviços de distribuição de nomes descentralizados (DNS)
- Serviços de informação descentralizados (WWW)

Técnicas para Escalabilidade

Replicação/caching

Fazer cópias de dados em máquinas distintas:

- Servidores de arquivos e bancos de dados replicados
- Sítios Web espelhados
- Caches Web (em navegadores e proxies)
- Cache de arquivos (no servidor e no cliente)

Escalabilidade – O Problema

Observação

Aplicar técnicas de escalabilidade é relativamente fácil, exceto por:

- A existência de múltiplas cópias (por exemplo, caches replicados) pode levar a inconsistências: alteração de uma cópia a faz distinta das demais.
- Manter cópias sempre consistentes de maneira geral requer sincronização global para cada modificação.
- Sincronização global dificulta escalabilidade.

Observação

Se podemos tolerar inconsistências, podemos reduzir a necessidade de sincroniação global, mas tolerar inconsistências é dependente da aplicação.

Escalabilidade – O Problema

Observação

Aplicar técnicas de escalabilidade é relativamente fácil, exceto por:

- A existência de múltiplas cópias (por exemplo, caches replicados) pode levar a inconsistências: alteração de uma cópia a faz distinta das demais.
- Manter cópias sempre consistentes de maneira geral requer sincronização global para cada modificação.
- Sincronização global dificulta escalabilidade.

Observação

Se podemos tolerar inconsistências, podemos reduzir a necessidade de sincroniação global, mas tolerar inconsistências é dependente da aplicação.

Escalabilidade – O Problema

Observação

Aplicar técnicas de escalabilidade é relativamente fácil, exceto por:

- A existência de múltiplas cópias (por exemplo, caches replicados) pode levar a inconsistências: alteração de uma cópia a faz distinta das demais.
- Manter cópias sempre consistentes de maneira geral requer sincronização global para cada modificação.
- Sincronização global dificulta escalabilidade.

Observação

Se podemos tolerar inconsistências, podemos reduzir a necessidade de sincroniação global, mas tolerar inconsistências é dependente da aplicação.

Escalabilidade – O Problema

Observação

Aplicar técnicas de escalabilidade é relativamente fácil, exceto por:

- A existência de múltiplas cópias (por exemplo, caches replicados) pode levar a inconsistências: alteração de uma cópia a faz distinta das demais.
- Manter cópias sempre consistentes de maneira geral requer sincronização global para cada modificação.
- Sincronização global dificulta escalabilidade.

Observação

Se podemos tolerar inconsistências, podemos reduzir a necessidade de sincroniação global, mas tolerar inconsistências é dependente da aplicação.

Escalabilidade – O Problema

Observação

Aplicar técnicas de escalabilidade é relativamente fácil, exceto por:

- A existência de múltiplas cópias (por exemplo, caches replicados) pode levar a inconsistências: alteração de uma cópia a faz distinta das demais.
- Manter cópias sempre consistentes de maneira geral requer sincronização global para cada modificação.
- Sincronização global dificulta escalabilidade.

Observação

Se podemos tolerar inconsistências, podemos reduzir a necessidade de sincroniação global, mas tolerar inconsistências é dependente da aplicação.

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Desenvolvendo Sistemas Distribuídos: Dificuldades

Observação

Muitos sistemas distribuídos são desnecessariamente complexos devido a erros de projeto que necessitaram ajustes posteriores. Há muitas falsas premissas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia não muda
- A latência é próxima de zero
- A banda é próxima de infinito
- Transporte tem custo zero
- Há um único administrador

Tipos de Sistemas Distribuídos

- Sistemas de Computação Distribuídos
- Sistemas de Informação Distribuídos
- Sistemas Distribuídos Pervasivos

Tipos de Sistemas Distribuídos

Observação

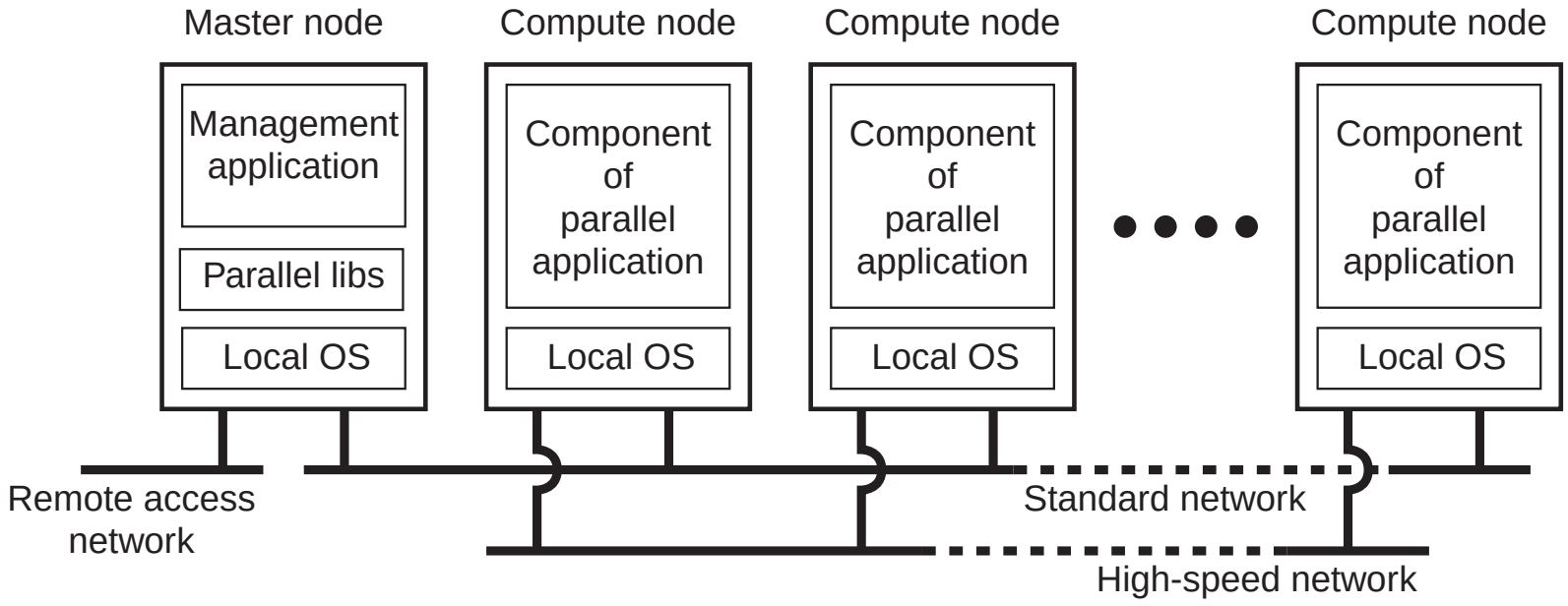
Muitos sistemas distribuídos são configurados para Computação de Alto Desempenho

Computação em Cluster

Essentialmente um grupo de sistemas high-end conectados por uma LAN:

- Homogêneo: mesmo OS, hardware quase-idêntico
- Nó gerenciador único

Sistemas de Computação Distribuídos



Sistemas de Computação Distribuídos

Computação em Grid

Muitos nós por toda parte:

- Heterogêneo
- Disperso entre organizações
- Espalhado por WAN

Nota

Para permitir colaborações, grids geralmente usam organizações virtuais. Em essência, um grupo de usuários (seus IDs) que permitem gerenciar autorizações e alocação de recursos.

Sistemas de Computação Distribuídos

Computação em Nuvem

Infraestrutura como serviço

- Heterogêneo
- Espalhado por WAN

Sistemas de Informação Distribuídos

Observação

A vasta maioria de sistemas distribuídos em uso hoje em dia são formas de sistemas de informação tradicionais, que agora integram 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
```

Sistemas de Informação Distribuídos

Observação

A vasta maioria de sistemas distribuídos em uso hoje em dia são formas de sistemas de informação tradicionais, que agora integram 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
```

Note

Transações são operações atômicas.

Sistemas Distribuídos Pervasivos

Observação

Sistemas distribuídos em que nós são pequenos, móveis, e geralmente embutidos em um sistema maior.

Alguns requisitos

- Troca contextual: O sistema é parte de um ambiente em que mudanças devem ser imediatamente percebidas.
- Composição Ad hoc: Cada nó pode ser usado de diferentes formas por diferentes usuários. Requer facilidade de configuração.
- Compartilhamento é o padrão: Nós entram e saem, provendo serviços compartilhados e informação.

Note

Pervasão (??) e transparência de distribuição: bons parceiros?

Sistemas Distribuídos Pervasivos

Observação

Sistemas distribuídos em que nós são pequenos, móveis, e geralmente embutidos em um sistema maior.

Alguns requisitos

- Troca contextual: O sistema é parte de um ambiente em que mudanças devem ser imediatamente percebidas.
- Composição Ad hoc: Cada nó pode ser usado de diferentes formas por diferentes usuários. Requer facilidade de configuração.
- Compartilhamento é o padrão: Nós entram e saem, provendo serviços compartilhados e informação.

Note

Pervasão (??) e transparência de distribuição: bons parceiros?

Sistemas Pervasivos: Exemplos

Sistemas para casas

Devem ser completamente auto-organizáveis:

- Sem administrador
- Provê espaço pessoal para cada usuário
- Solução simples: home box centralizada?

Sistemas de saúde pessoal

Dispositivos fisicamente próximos às pessoas:

- Onde e como devem ser armazenados os dados monitorados?
- Como prevenir perda de dados cruciais?
- Como propagar alertas?
- Como garantir segurança?
- Como médicos podem prover feedback online?

Sistemas Pervasivos: Exemplos

Sistemas para casas

Devem ser completamente auto-organizáveis:

- Sem administrador
- Provê espaço pessoal para cada usuário
- Solução simples: home box centralizada?

Sistemas de saúde pessoal

Dispositivos fisicamente próximos às pessoas:

- Onde e como devem ser armazenados os dados monitorados?
- Como prevenir perda de dados cruciais?
- Como propagar alertas?
- Como garantir segurança?
- Como médicos podem prover feedback online?

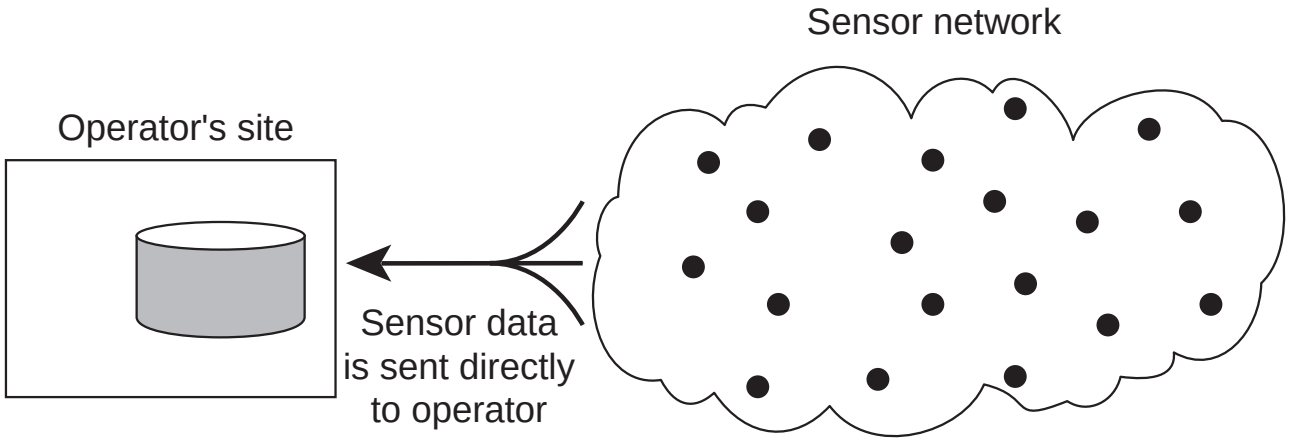
Redes de Sensores

Características

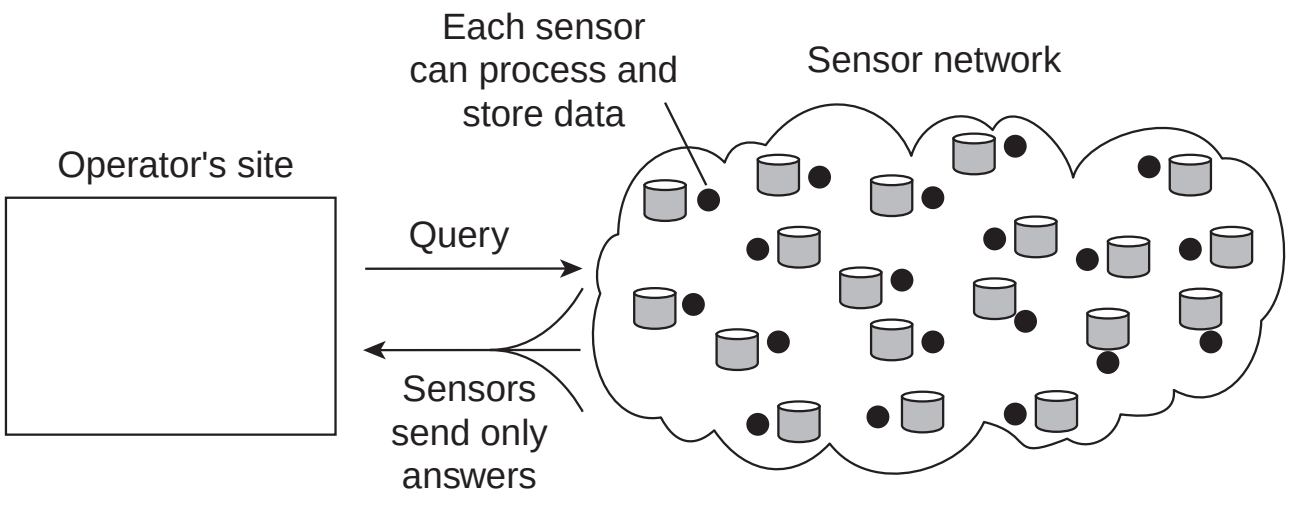
Os nós aos quais sensores estão ligados são:

- Muitos (10s-1000s)
- Simples (pouca memória/poder de processamento/capacidade de comunicação)
- Normalmente operam por bateria (ou mesmo sem baterias)

Redes de sensores como Sistemas Distribuídos



(a)



(b)