Sistemas Distribuídos

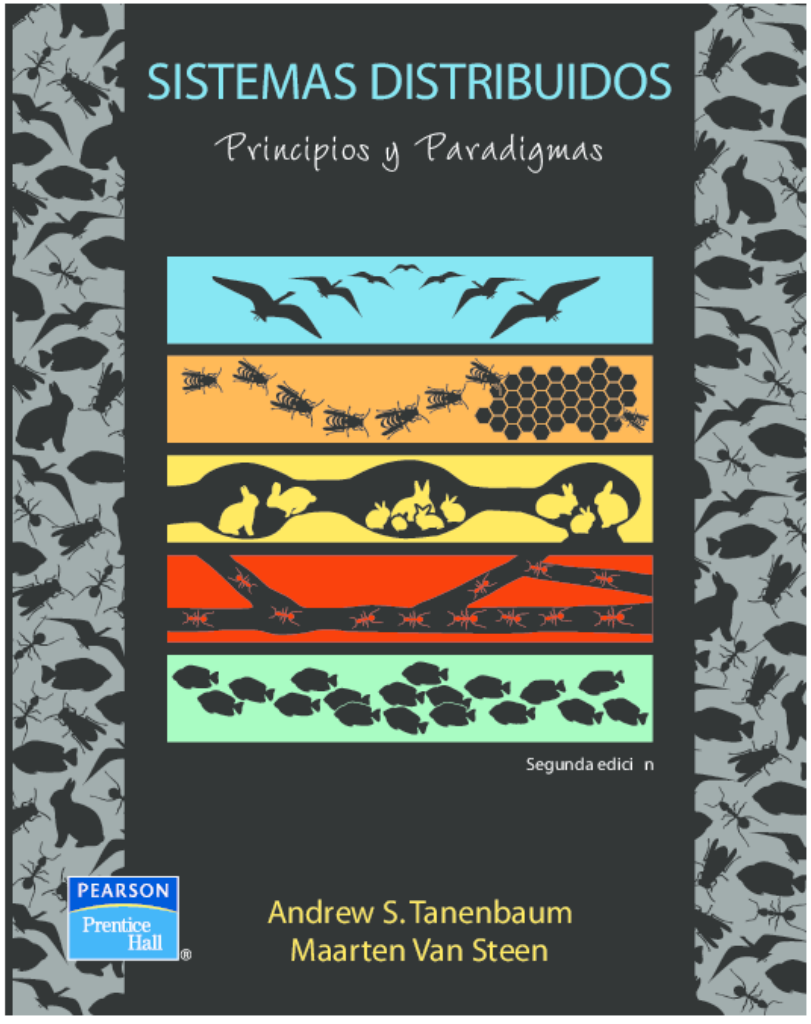
Introdução aos Sistemas Distribuídos

CHAPTER 1

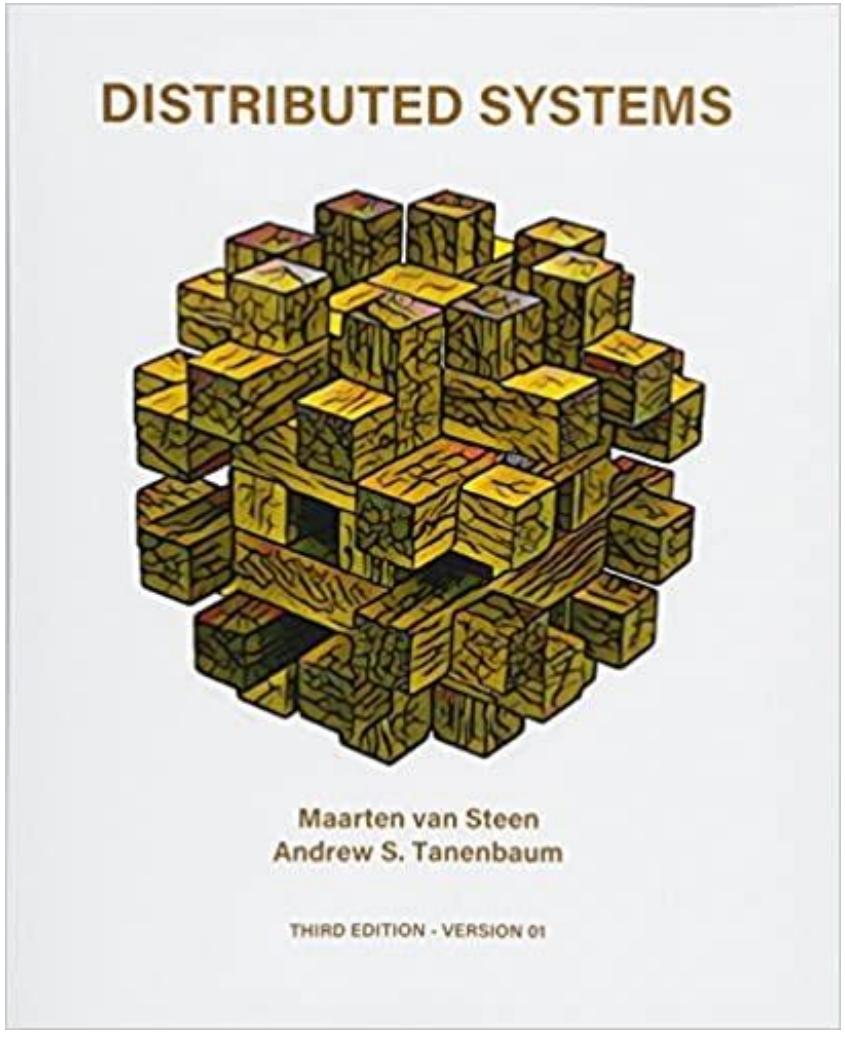
INTRODUCTION

Vladimir Rocha (Vladi)

CMCC - Universidade Federal do ABC


SISTEMAS DISTRIBUIDOS
Principios y Paradigmas
Segunda edici n
PEARSON
Prentice
Hall
Andrew S. Tanenbaum
Maarten Van Steen

2007


DISTRIBUTED SYSTEMS
Maarten van Steen
Andrew S. Tanenbaum
THIRD EDITION - VERSION 01

2017

Disclaimer

- Estes slides foram baseados nos do professor **Emilio Francesquini** 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`.

Agenda

- Definição de Sistemas Distribuídos
- Organização, Coerência, Middleware
- Objetivos
- Modelos de sistema
- Armadilhas

Agenda

- Definição de Sistemas Distribuídos
- Organização, Coerência, Middleware
- Objetivos
- Armadilhas

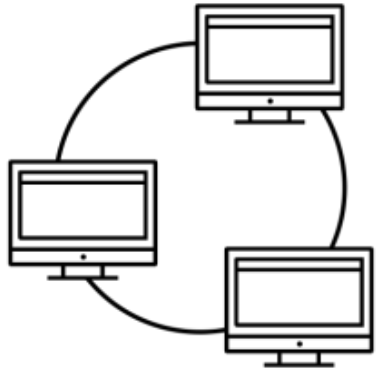
Sistemas Distribuídos: definição

Um sistema distribuído é um sistema de software que garante: *uma coleção de elementos de computação autônomos que são vistos pelos usuários como um sistema único e coerente*

Características importantes

- Elementos de computação autônomos, também denominados *nós* (ou *nodos*), sejam eles dispositivos de hardware ou processos de software
- Sistema único e coerente: usuários ou aplicações veem um único sistema ⇒ nós precisam colaborar entre si

Google



Coleção de nós autônomos

Comportamento independente

Cada nó é autônomo e, portanto, tem sua própria percepção de tempo: não há um relógio global. Leva a problemas fundamentais de sincronização e de coordenação.

Coleção de nós

- Como gerenciar *associações em grupos*
- Como saber se você realmente está se comunicando com um *(não-)membro autorizado do grupo*

Agenda

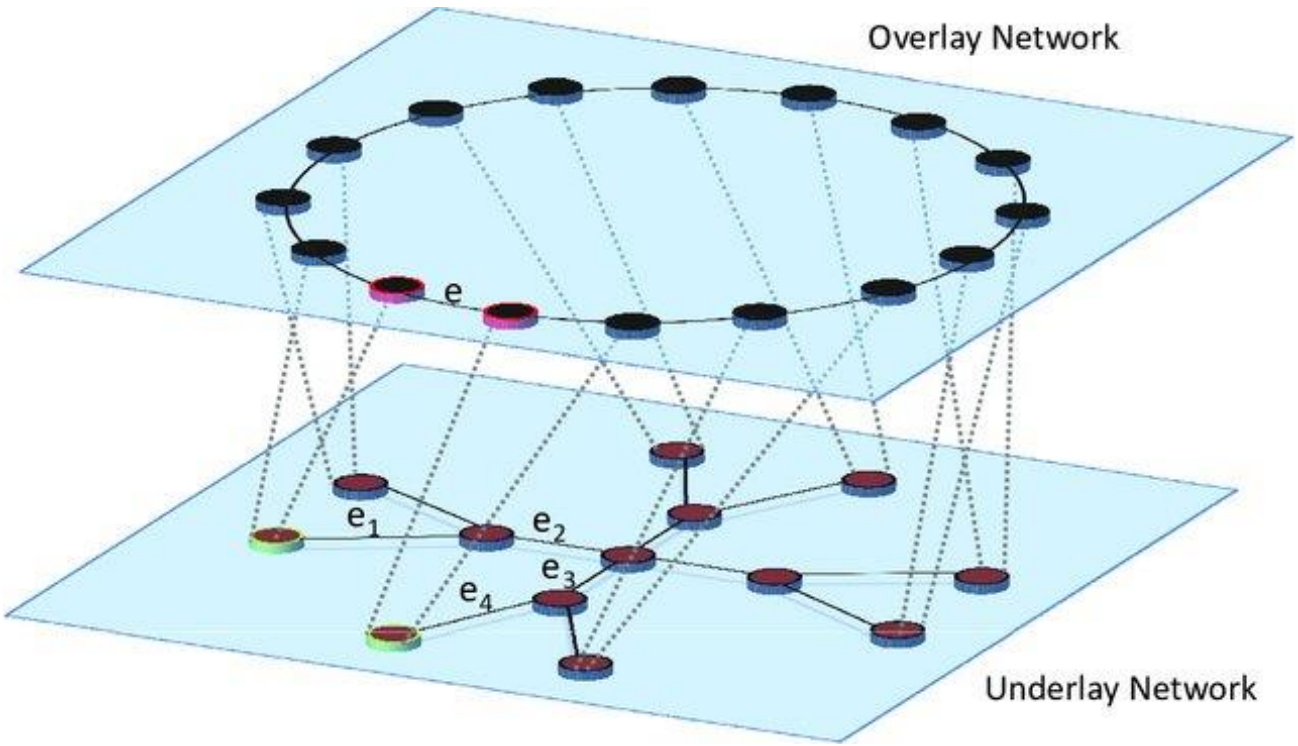
- Definição de Sistemas Distribuídos
- Organização, Coerência, Middleware
- Objetivos
- Armadilhas

Organização

Redes de overlay

Cada nó na coleção se comunica apenas com nós no sistema, seus vizinhos. O conjunto de vizinhos pode ser dinâmico, ou pode ser descoberto de forma implícita (ex: pode ser necessário procurá-lo)

Não necessariamente estão associados à estrutura física.



https://www.researchgate.net/publication/230774628_Harnessing_Complex_Structures_and_Collective_Dynamics_in_Large_Networked_Computing_Systems

Organização

Tipos de overlay: estruturada e não estruturada

Um exemplo bem conhecido de redes de overlay: *sistemas peer-to-peer*

A falha de um peer não compromete o funcionamento do sistema.

Organização

Tipos de overlay: estruturada e não estruturada

Um exemplo bem conhecido de redes de overlay: *sistemas peer-to-peer*

Estruturada cada nó tem um *conjunto bem definido de vizinhos* com os quais pode comunicar (árvore, anel)

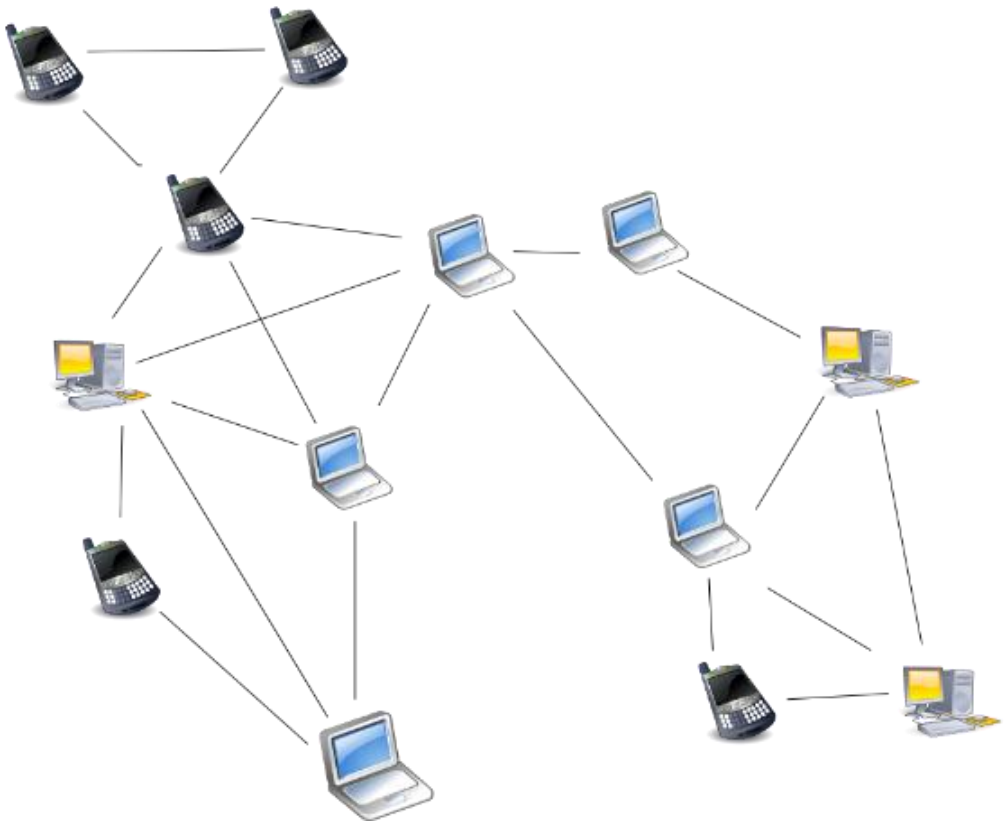
amazon cloudfront

Organização

Tipos de overlay: estruturada e não estruturada

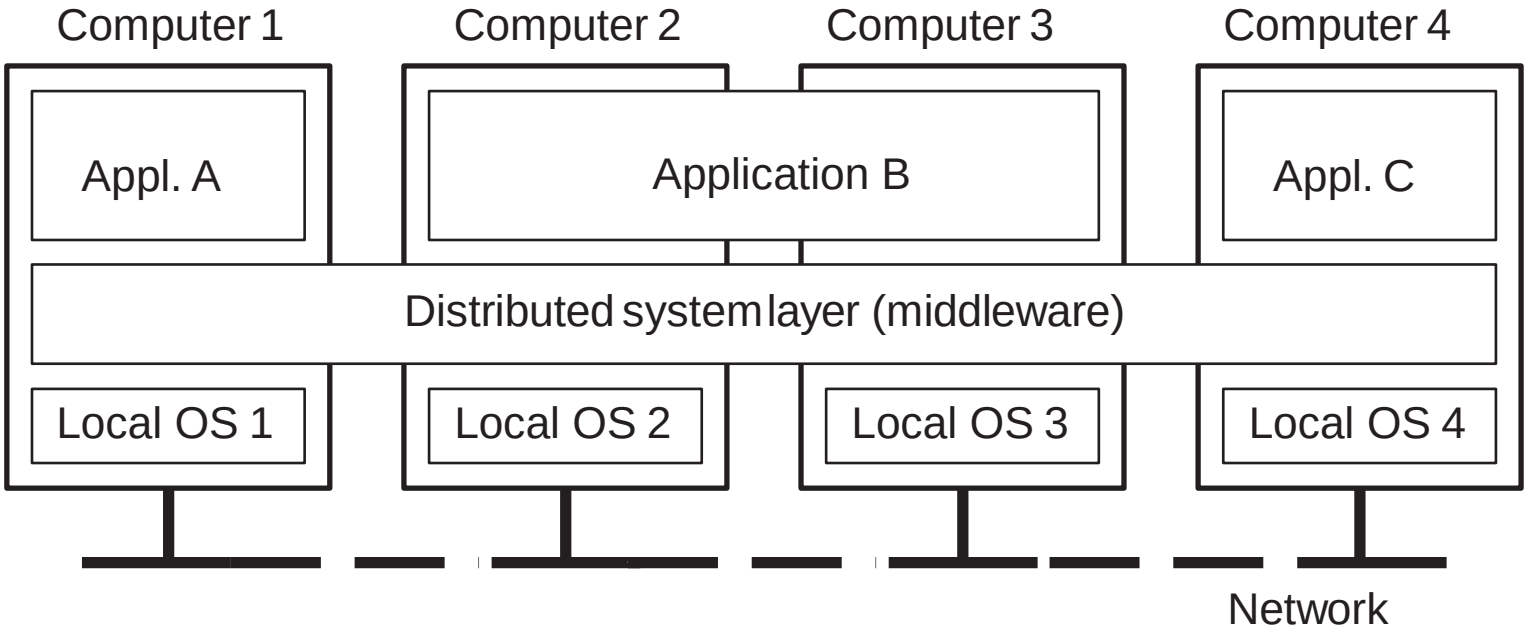
Um exemplo bem conhecido de redes de overlay: *sistemas peer-to-peer*

Não estruturada cada nó tem referências a *um conjunto aleatoriamente selecionado de outros nós* do sistema



https://en.wikipedia.org/wiki/Peer-to-peer

Middleware: o SO dos sistemas distribuídos



O que tem em um middleware?

Grosso modo, um conjunto de funções e componentes que não precisam ser reimplementados por cada aplicação separadamente.

Agenda

- Definição de Sistemas Distribuídos
- Organização, Coerência, Middleware
- Objetivos
- Armadilhas

Objetivos de sistemas distribuídos:

1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade

Objetivos de sistemas distribuídos:

1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade

Compartilhamento de recursos

Exemplos clássicos

- Compartilhamento de dados e arquivos na nuvem
- *Streaming* multimídia *peer-to-peer*
- Serviços de mensagens compartilhadas
- Serviços de hospedagem web compartilhados (à lá redes de distribuição de conteúdo)
- Serviços de busca e processamento

A ponto de pensarmos que:

"A rede é o computador"

John Gage, à época na Sun Microsystems

Objetivos de sistemas distribuídos:

1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade

Transparência de distribuição

Tipos (resumido):

Transparência	Descrição
Localização	Esconder onde o objeto está localizado
Replicação	Esconder que um objeto está sendo replicado
Falhas	Esconder falhas e a possível recuperação de um objeto

Graus de transparência

Expor a distribuição pode ser bom:

- Fazer uso de serviços baseados na localização (encontrando seus amigos mais próximos)
- Trabalhar com usuários em diferentes zonas horárias
- Quando é fácil para um usuário entender o que está acontecendo (e.g., falha no servidor pela conexão lenta).

Conclusão

- A transparência na distribuição é um bonito objetivo, mas alcançá-lo é uma questão bem diferente.

Objetivos de sistemas distribuídos:

1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade

Abertura de sistemas distribuídos

Sistemas distribuídos abertos

São capazes de interagir com outros sistemas abertos:

- devem respeitar interfaces bem definidas OK
- devem permitir a portabilidade de aplicações OK
- devem ser fáceis de estender OK
- devem ser facilmente interoperáveis NÃO

Objetivos de sistemas distribuídos:

1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade
    1. Tamanho, Geográfica, Administrativa
    2. Técnicas
    3. Problemas

Objetivos de sistemas distribuídos:

1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade
    1. Tamanho, Geográfica, Administrativa [Neuman 1994]
    2. Técnicas
    3. Problemas

Escalabilidade em Sistemas Distribuídos

Observação:

Muitos desenvolvedores de sistemas distribuídos modernos usam o adjetivo "escalável" sem deixar claro o porquê deles escalarem.

Escalabilidade em Sistemas Distribuídos

Observação:

Muitos desenvolvedores de sistemas distribuídos modernos usam o adjetivo "escalável" sem deixar claro o porquê deles escalarem.

Escalabilidade se refere a pelo menos três componentes:

- Número de usuários e/ou processos – escalabilidade de tamanho
- Distância máxima entre nós – escalabilidade geográfica
- Número de domínios administrativos – escalabilidade administrativa

Escalabilidade em Sistemas Distribuídos

Observação:

Muitos desenvolvedores de sistemas distribuídos modernos usam o adjetivo "escalável" sem deixar claro o porquê deles escalarem.

Escalabilidade se refere a pelo menos três componentes:

- Número de usuários e/ou processos – escalabilidade de tamanho
- Distância máxima entre nós – escalabilidade geográfica
- Número de domínios administrativos – escalabilidade administrativa

Observação:

A maior parte dos sistemas escalam apenas (e até certo ponto) em tamanho. Como?

Usando servidores poderosos – *scale-in*.

Hoje em dia, o desafio é conseguir escalabilidade geográfica e administrativa.

Problemas para obtenção de escalabilidade

Um sistema completamente descentralizado tem as seguintes características:

- Nenhuma máquina tem informação completa sobre o estado do sistema
- Máquinas tomam decisões baseadas apenas em informação local
- Falhas em uma máquina não devem arruinar a execução do algoritmo
- Não é possível assumir a existência de um relógio global

Problemas na escalabilidade de tamanho

Número de usuários e/ou processos – escalabilidade de tamanho

- Capacidade computacional, limitada pelas CPUs
- Capacidade de armazenamento e de transferência entre CPU e HD
- Capacidade de largura de banda entre o usuário e o serviço.

Resumo da notícia

- Amazon Prime Vídeo decepciona fãs ao não conseguir transmitir os jogos da NBA; fornecedor externo teria causado o problema
- Transmissões ao vivo são o calcanhar de Aquiles das plataformas de streaming, mas se tornaram essenciais para atrair assinantes com esportes

https://www.uol.com.br/splash/colunas/guilherme-ravache/2022/10/23/caos-da-nba-na-amazon-expoe-riscos-da-gigante-que-busca-dominar-esportes.htm

Como estimar a capacidade?

Problemas na escalabilidade de tamanho

Número de usuários e/ou processos – escalabilidade de tamanho

- *Back-of-the-Envelope estimation*

Serve para estimar a capacidade e desempenho do sistema para entender quais serão os requisitos (e.g., vazão e armazenamento) a serem atendidos.

Exemplo do Twitter:

400 milhões de usuários ativos em 2023.
50% dos usuários usam o Twitter diariamente.
Cada usuário envia na média 2 tweets por dia.
10% dos tweets têm arquivos de mídia de 1 MB.
Os dados devem ser armazenados por 5 anos.

Tweets por seg. TPS?

Total HD da mídia?

Problemas na escalabilidade de tamanho

Exemplo do Twitter:

400 milhões de usuários ativos em 2023.
50% dos usuários usam o Twitter diariamente.
Cada usuário envia na média 2 tweets por dia.
10% dos tweets têm arquivos de mídia de 1 MB.
Os dados devem ser armazenados por 5 anos.

Vazão por segundo:

- Usuários ativos por dia: 400 milhões * 0.5 = 200 milhões.
- TPS: 200 milhões * 2 tweets / (24 horas * 3600 seg) = ~ 4.600.
- Picos TPS: 2 * TPS = ~ 9.200.

Armazenamento (só da mídia):

- Por dia: 200 milhões * 2 tweets * 0.1 * 1 MB = 40 TB.
- Para 5 anos: 40 TB * 365 dias * 5 anos = 73 PB.

Problemas na escalabilidade geográfica

Distância máxima entre nós – escalabilidade geográfica

- Não é possível só ir da LAN para WAN: muitos dos sistemas distribuídos assumem interações cliente-servidor com latências pequenas.
- Links das WAN são pouco confiáveis: e.g.mover dados de streaming de vídeo da LAN para WAN falhará.
- O broadcast da camada de rede não pode ser aplicado.

Problemas na escalabilidade administrativa

Número de domínios administrativos – escalabilidade administrativa

Essência

Políticas conflitantes a respeito do uso (e pagamento), gerenciamento e segurança.

Exemplos

- Compartilhamento de recursos caros entre diferentes domínios.
- Processamento de informações em diferentes lugares de forma confidencial.
- Onde armazenar e para quem disponibilizar os dados mantendo políticas como LGPD.

Objetivos de sistemas distribuídos:

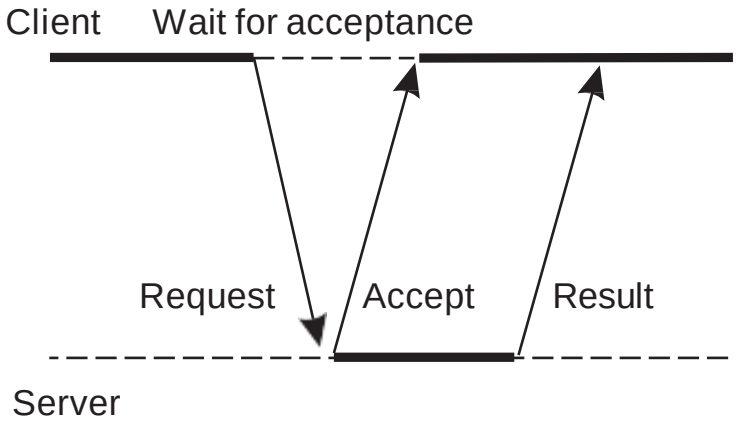
1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade
    1. Tamanho, Geográfica, Administrativa
    2. Técnicas
    3. Problemas

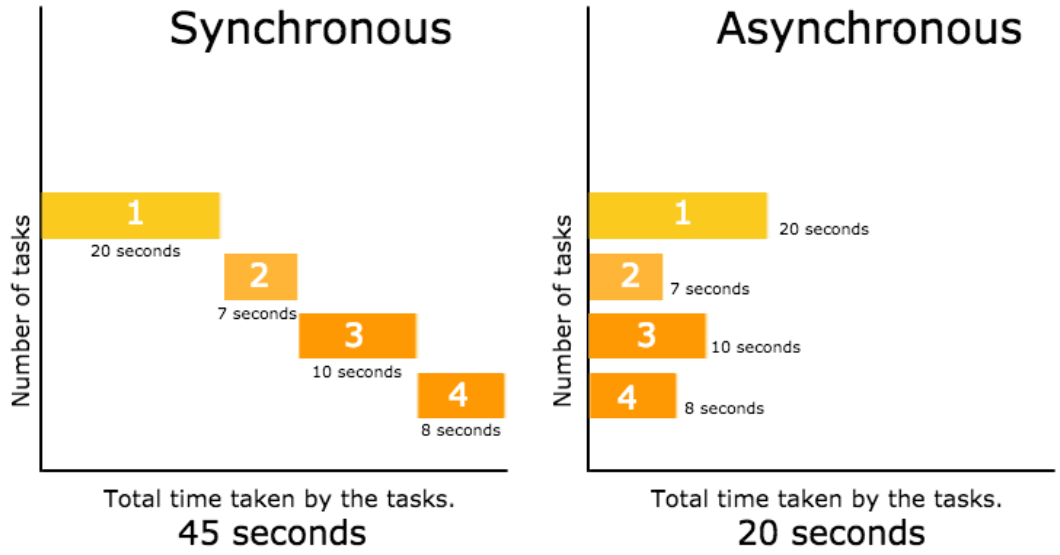
Técnicas de escalabilidade

Ideia geral: esconder latência de comunicação

1. Não fique esperando por respostas; faça outra coisa

- Use comunicação assíncrona → não confundir com modelo assíncrono
- Use diferentes *handlers* para tratamento de mensagens (multithread)
- Problema: nem toda aplicação se encaixa nesse modelo





Técnicas de escalabilidade

2. Particionamento de dados e computação em muitas máquinas

- Mova a computação para os clientes (ex: Javascript, Spark, etc.)
- Serviços de nomes decentralizados (DNS)
- Sistemas de informação decentralizados (WWW)
- Microsserviços
- Sharding (particionamento) de tabelas de bancos de dados ou de objetos

Escalabilidade Vertical/Horizontal (Scale-In/Scale-Out)

Técnicas de escalabilidade

3. Replicação/caching

Faça cópias dos dados e disponibilize-as em diferentes máquinas:

- Bancos de dados e sistemas de arquivos replicados
- Sites web "espelhados"
- Caches web (nos navegadores)
- Cache de arquivos (no servidor e nos clientes)
- Cache de dados (nos servidores – memcached)

Objetivos de sistemas distribuídos:

1. Disponibilização de recursos compartilhados
2. Transparência de distribuição
3. Abertura
4. Escalabilidade
    1. Tamanho, Geográfica, Administrativa
    2. Técnicas
    3. Problemas

Escalabilidade – O Problema

Observação

Aplicar técnicas para obtenção de escalabilidade é fácil, exceto por:

- Manter múltiplas cópias (em cache ou replicadas) leva a inconsistências: a modificação em uma cópia a torna diferente das demais
- Manter as cópias consistentes requer sincronização global em cada modificação (mas não há um relógio global)
- Sincronização global impossibilita soluções escaláveis

Escalabilidade – O Problema

Observação

Aplicar técnicas para obtenção de escalabilidade é fácil, exceto por:

- Manter múltiplas cópias (em cache ou replicadas) leva a inconsistências: a modificação em uma cópia a torna diferente das demais
- Manter as cópias consistentes requer sincronização global em cada modificação (mas não há um relógio global)
- Sincronização global impossibilita soluções escaláveis

Observação:

Se pudermos tolerar inconsistências, poderíamos reduzir a dependência de sincronização globais, mas tolerar inconsistências é algo que depende da aplicação.

Agenda

- Definição de Sistemas Distribuídos
- Organização, Coerência, Middleware
- Objetivos
- Modelos de sistema
- Armadilhas

Modelos de sistema [tanenbaum17]

- Síncrono
- Assíncrono
- Parcialmente Síncrono

Não confundir com comunicação síncrona e assíncrona entre cliente-servidor

Modelos de sistema [tanenbaum17]

Síncrono

- Os limites de tempo para transferir uma mensagem são conhecidos
- Os limites de tempo para processar uma ação são conhecidos

  LAN/Datacenters

Assíncrono

- Sem limites de tempo para transferir uma mensagem
- Sem limites de tempo para processar uma ação

  Internet

Parcialmente síncrono

- Inicialmente o comportamento do sistema é assíncrono
- Eventualmente (sim ou sim) o comportamento será síncrono

  Internet

Agenda

- Definição de Sistemas Distribuídos
- Organização, Coerência, Middleware
- Objetivos
- Modelos baseados no tempo
- Armadilhas

Armadilhas no desenvolvimento de sistemas distribuídos

Observação:

Muitos sistemas distribuídos se tornam desnecessariamente complexos por causa de "consertos" ao longo do tempo. Em geral, há muitas hipóteses falsas:

- A rede é confiável
- A rede é segura
- A rede é homogênea
- A topologia da rede não muda
- A latência é zero
- Largura de banda é infinita
- O custo de transporte é zero
- A rede possui um administrador

Conceitos adquiridos

- Definição de sistemas distribuídos.
- Não há relógio global.
- Overlay e Middleware.
- Escalabilidade.
- Cálculo *back-of-the-envelope*.
- Replicação e inconsistência.
- Comunicação:
    - Síncrona, Assíncrona
- Modelos de sistema:
    - Síncrono, Assíncrono, Parcialmente síncrono.

Computação Distribuída

Tipos de Sistemas Distribuídos

Vladimir Rocha (Vladi)

CMCC - Universidade Federal do ABC

Disclaimer

- Estes slides foram baseados nos do professor **Emilio Francesquini** 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.

Agenda

Tipos de Sistemas Distribuídos

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

Agenda

Tipos de Sistemas Distribuídos

- Sistemas para computação distribuída de alto desempenho
  - Cluster
  - Grade
  - Nuvem

Aglomerados de computação (cluster computing)

Essencialmente um grupo de computadores de boa qualidade conectados via LAN

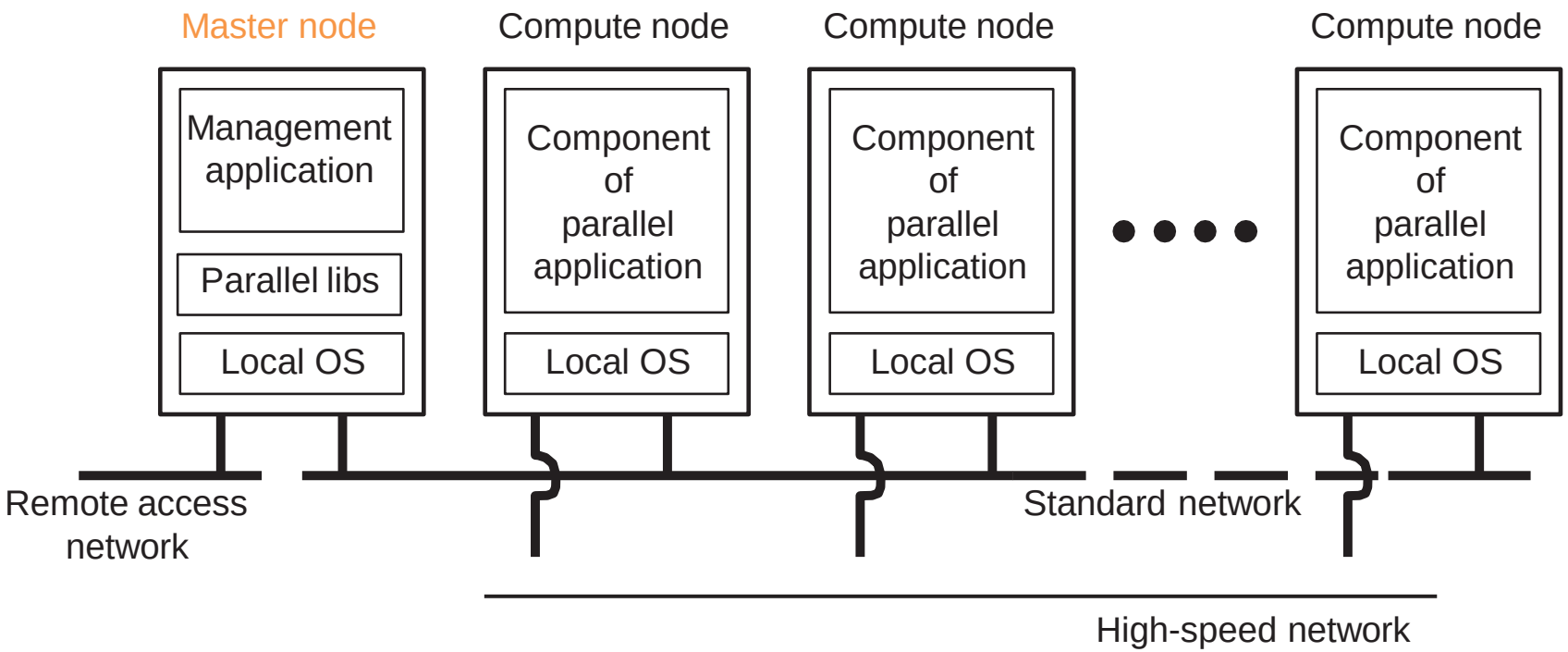
- Homogêneo: mesmo SO, hardware quase idêntico
- Um único nó gerenciador

https://en.wikipedia.org/wiki/Computer_cluster

Aglomerados de computação (cluster computing)

Essencialmente um grupo de computadores de boa qualidade conectados via LAN

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



Computação em Grade

O próximo passo: vários nós vindos de todos os cantos [Foster 2001]

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

Nota:

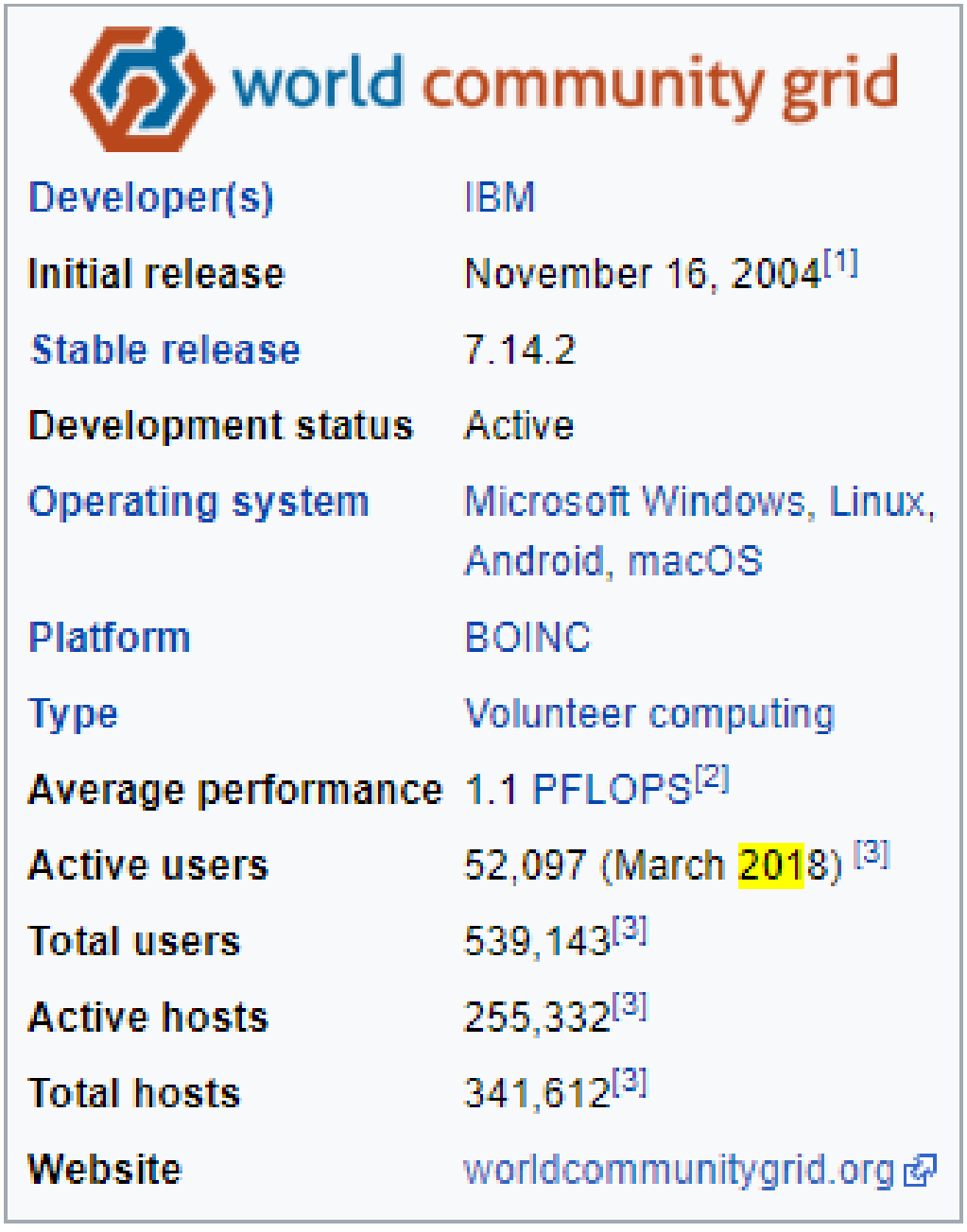
Para permitir colaborações, grades normalmente usam *organizações virtuais*. Essencialmente, isso significa que os usuários (ou melhor, seus IDs) são organizados em grupos que possuem autorização para usar alguns recursos.

Computação em Grade

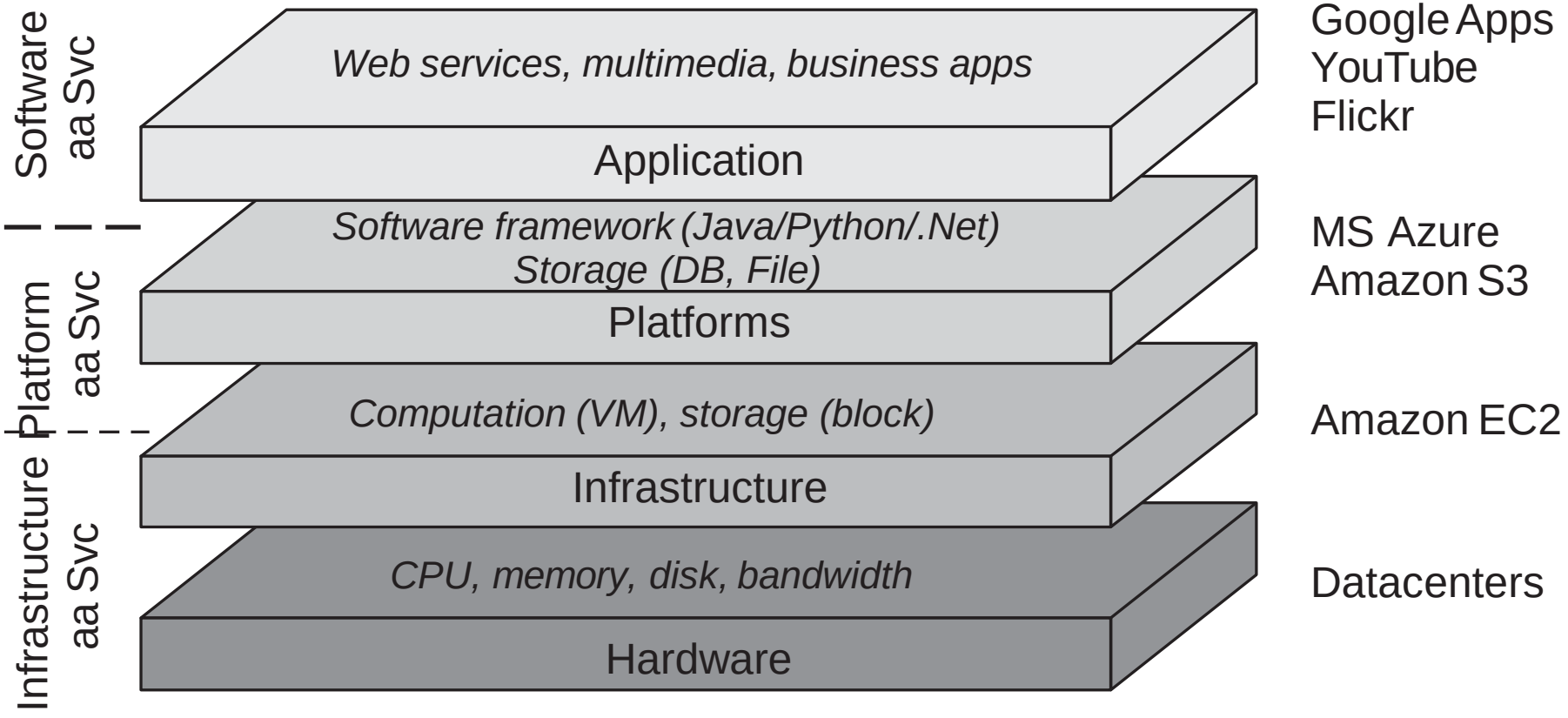
1999 ~ 2006 1 PFLOPS

2003 ~ 2009

2000 ~ 2020+ 1100 PFLOPS



Computação em Nuvem



IaaS: cobre as camadas de hardware (escondida do cliente) e infraestrutura.

PaaS: cobre a camada de plataforma.

SaaS: cobre a camada de aplicação.

Computação em Nuvem

Computação em nuvem

Faz uma distinção entre quatro camadas:

Hardware processadores, roteadores, energia, sistemas de refrigeração

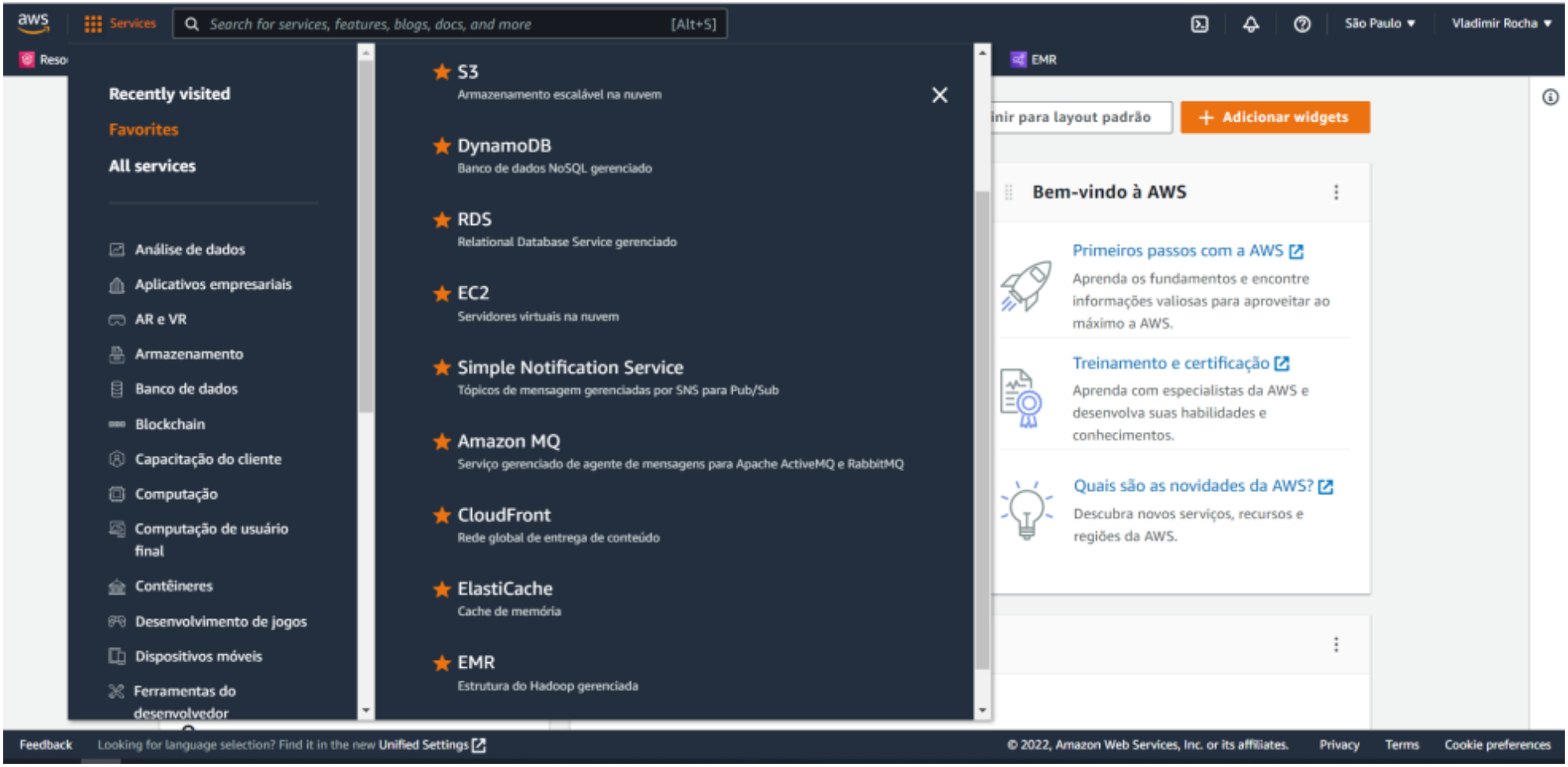
Infraestrutura Utiliza técnicas de virtualização para alocação e gerenciamento de armazenamento e servidores virtuais

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

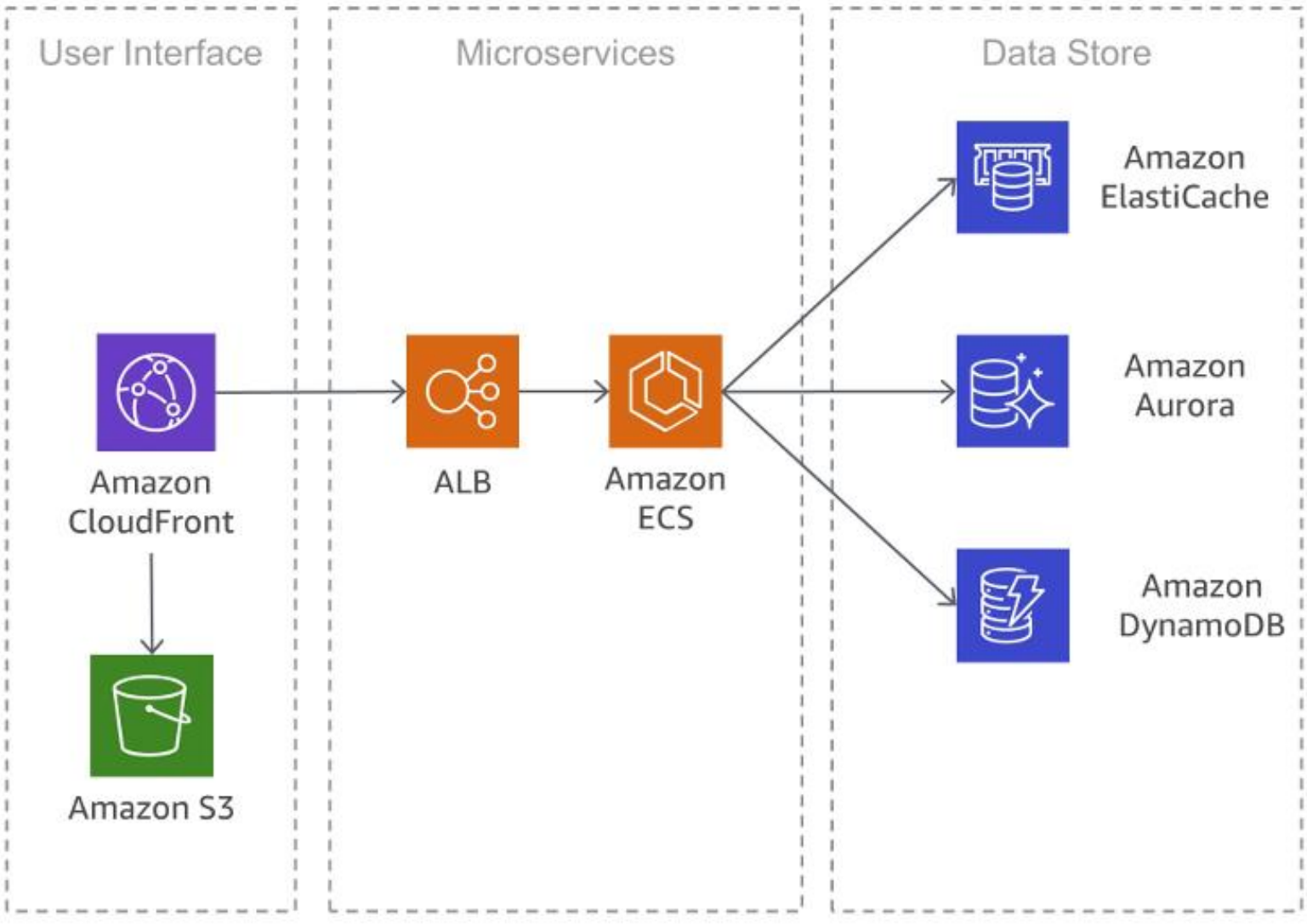
Aplicação as aplicações propriamente ditas, tais como as suítes de aplicativos para escritórios.

Vídeo de como acessar a nuvem da Amazon e criar uma instância (opcional)
https://www.youtube.com/watch?v=_8erPrlPm9Y

Computação em Nuvem



Computação em Nuvem



https://docs.aws.amazon.com/whitepapers/latest/microservices-on-aws/simple-microservices-architecture-on-aws.html

Agenda

Tipos de Sistemas Distribuídos

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

Sistemas de Informação Distribuídos

Observação:

Uma quantidade enorme de sistemas em uso hoje em dia são formas de sistemas de informação tradicionais.

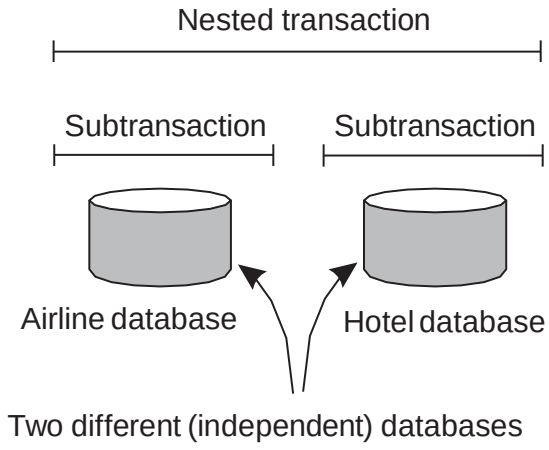
Geralmente são aplicações que se conectam à rede para intercambiar informações.

As palavras-chave que caracterizam os sistemas de informação distribuído são transação e integração.

Sistemas de Informação Distribuídos

Sistemas de processamento de transações.

```
01 BEGIN_TRANSACTION(transaction tx)
02 READ(tx, air_db, data-air)
03 READ(tx, hotel_db, data-hotel)
04 updated := WRITE(tx, air_db, new_data_air)
05 IF NOT updated THEN
06        ABORT_TRANSACTION(tx)
07 ELSE
08        WRITE(tx, hotel_db, new_data_hotel)
09 END_TRANSACTION(tx)
10 END IF
```



Nota:

Transações formam uma operação atômica.

Se cair em qualquer lugar: ABORT_TRANSACTION

Sistemas de Informação Distribuídos: Transações

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.

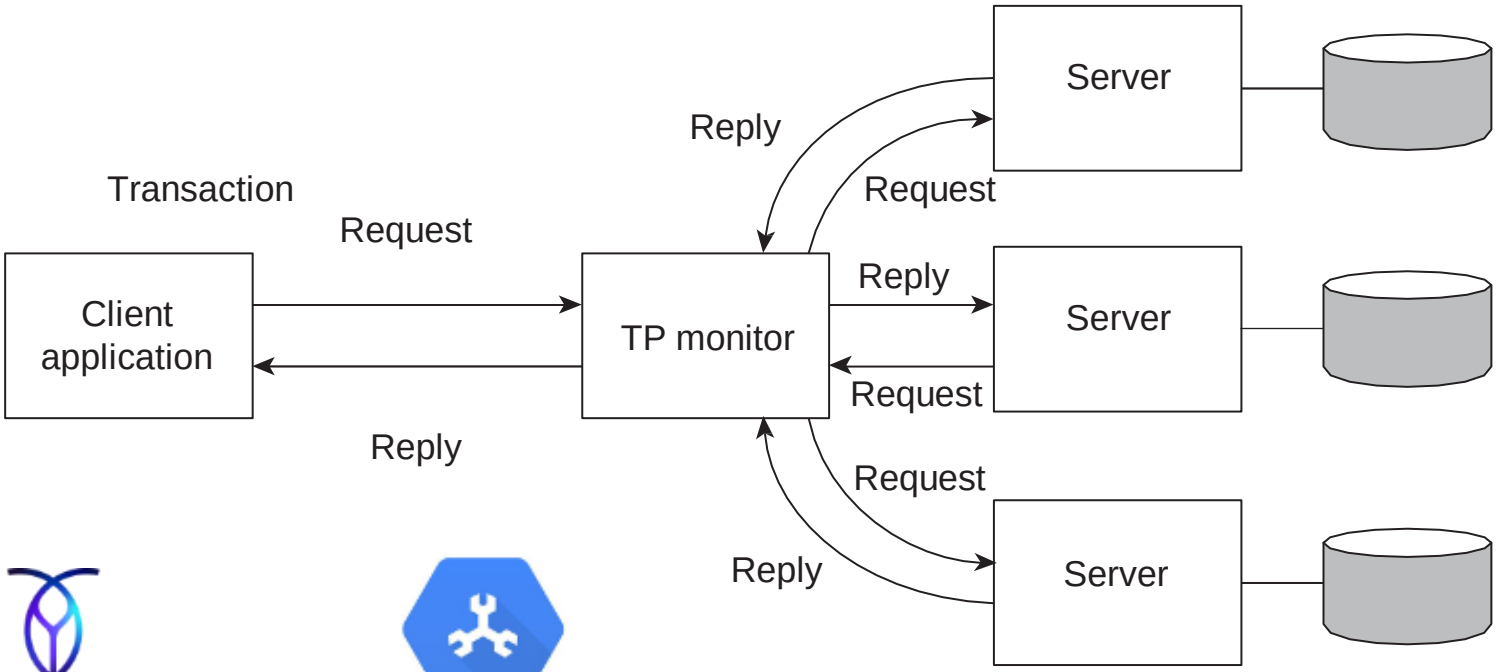
Durabilidade Após o término de uma transação, seus efeitos são permanentes: mudanças de estado sobrevivem a falhas.

Monitor de Processamento de Transações

Capítulo 8

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.



CockroachDB

[2015 -]

Cloud Spanner

[2012 -]

Diferentes Datacenters

Integração de Aplicações Corporativas

Situação

As organizações posuem diversas aplicações muitas delas sem interoperabilidade.

Solução básica analisada (monitor)

Combinar as requisições das aplicações em somente um servidor, quem realizará o envio e coleta das respostas, apresentando ao cliente um resultado coerente.

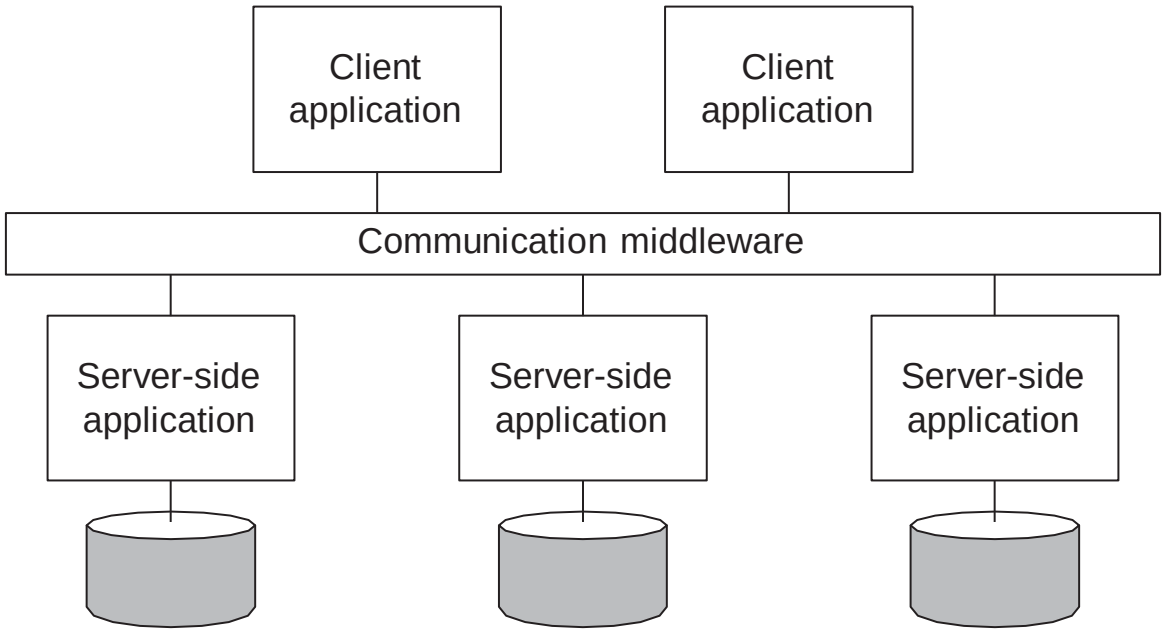
Próximo passo

Integrar direamente a comunicação entre aplicações, levando ao EAI (Enterprise Application Integration).

Integração de Aplicações Corporativas

Problema

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

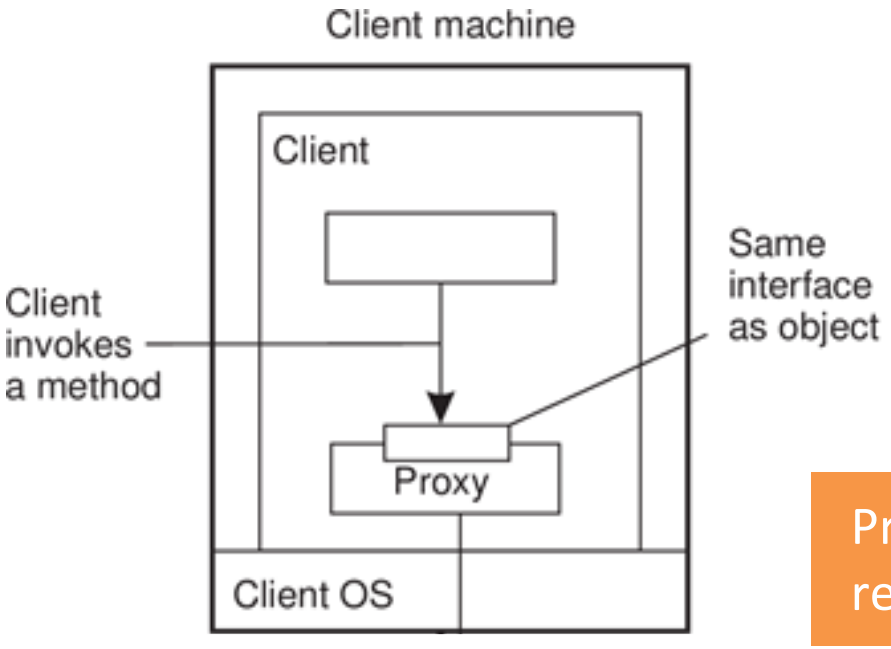


- Chamada de Procedimento Remoto (RPC) [Nelson 1984]

  Java RMI [Sun 2002]

  Google RPC (GRPC) [Google 2015]
- Middleware Orientado a Mensagens (MOM)

Como integrar as aplicações: RPC



Proxy: componente que encaminha requisições ao servidor responsável.

O cliente faz chamadas ao método como estivesse sendo executado na máquina local.

Mas o método pode estar sendo executado em máquinas remotas.

Agenda

Tipos de Sistemas Distribuídos

- Sistemas para computação distribuída de alto desempenho
- Sistemas de informação distribuídos
- Sistemas distribuídos para computação pervasiva
    - Ubíqua
    - Móvel
    - Sensores

Sistemas Pervasivos

Tendência em sistemas distribuídos; nós são pequenos, móveis e normalmente embutidos em um sistema muito maior.
Associado com a Internet das coisas (IoT – Internet of Things).

Três tipos (com sobreposição):

- Sistemas ubíquos: continuamente presentes
- Sistemas móveis: inerentemente móvel
- Sistemas de sensores: sente e atúa no ambiente.

Sistemas Ubíquos

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.
- Autonomia: nós operam de forma autônoma sem intervenção humana.
- Inteligência: pode emerger um comportamento inteligente dada a dinâmica e as interações dos nós.

Comportamento emergente: flocking
https://www.youtube.com/watch?v=9zLu5quQfYw

Nest Learning Thermostat + Nest Temperature Sensor

Sistemas Móveis

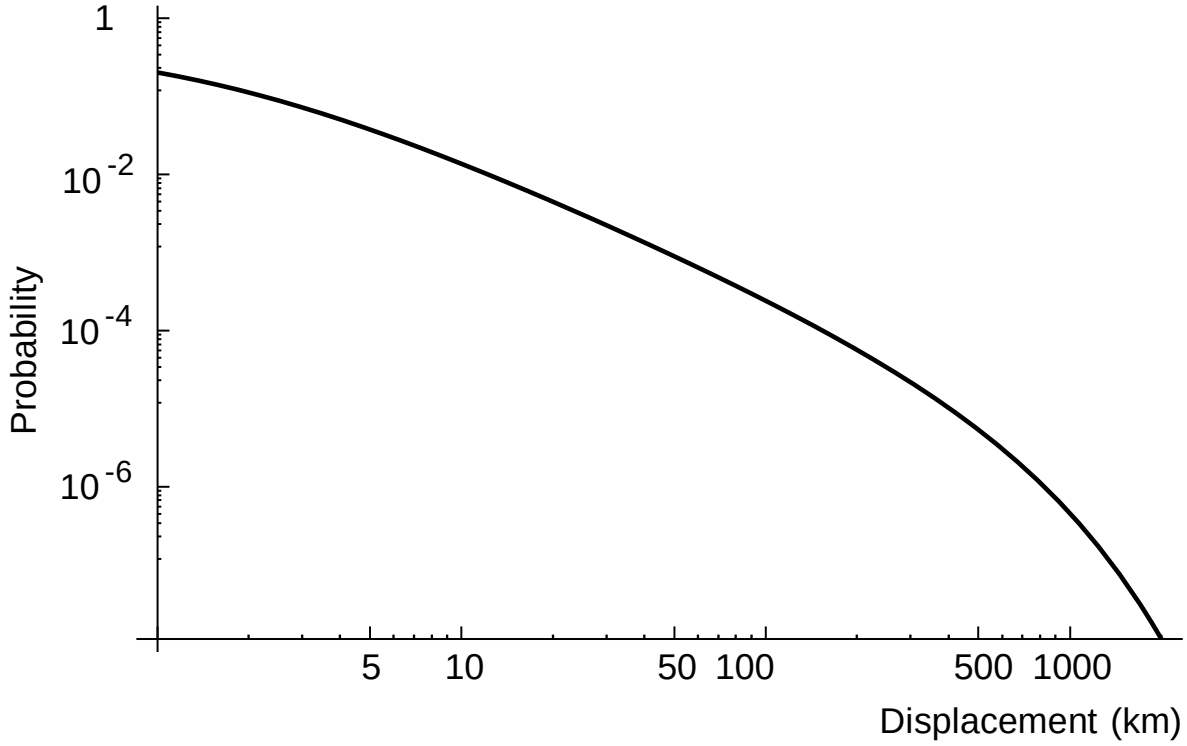
Miríade de dispositivos móveis: smartphones, tablets, óculos AR/VR, etc.

Características:

- Comunicação sem fio
- Mobilidade implica em que a localização do dispositivo mudará no tempo. Palavra-chave: descoberta
- A comunicação pode ser difícil, pois não há uma rota estável. Leva às redes tolerantes a disrupções

Padrões de mobilidade

Quão móveis somos? Rastreando 100 mil celulares durante seis meses [Gonzalez 2008].



Além disso: as pessoas tendem a voltar ao mesmo lugar depois de 24, 48 ou 72 horas ⇒ não somos tão móveis.

Redes de sensores

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

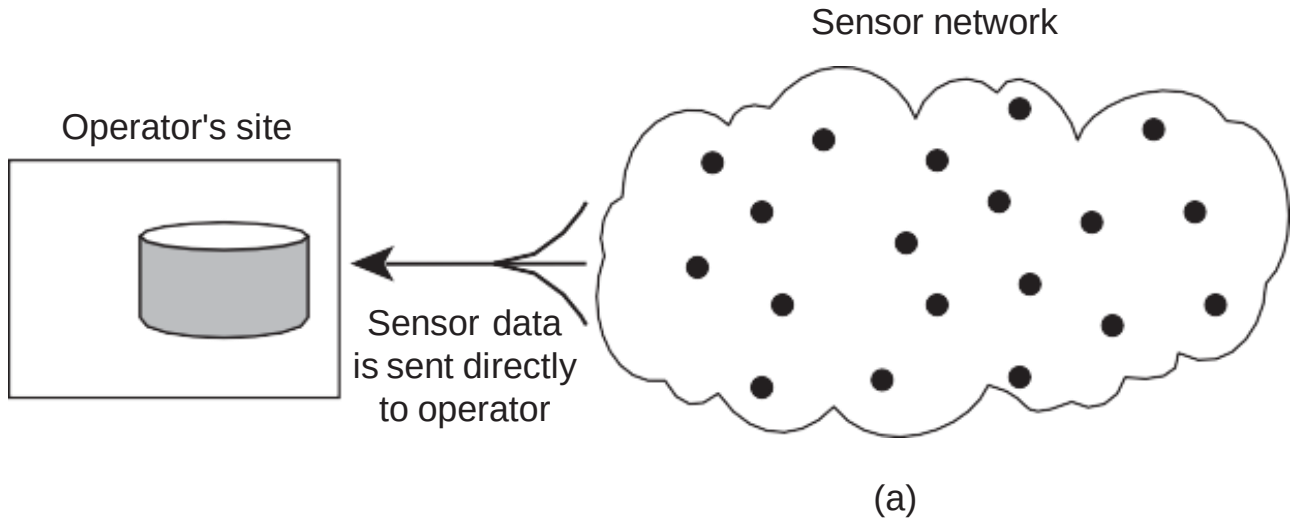
Crowd-sourced weather apps claim accuracy, but watch the sky anyway

Barometer-equipped smartphones are slowly becoming a vast sensor network for weather data

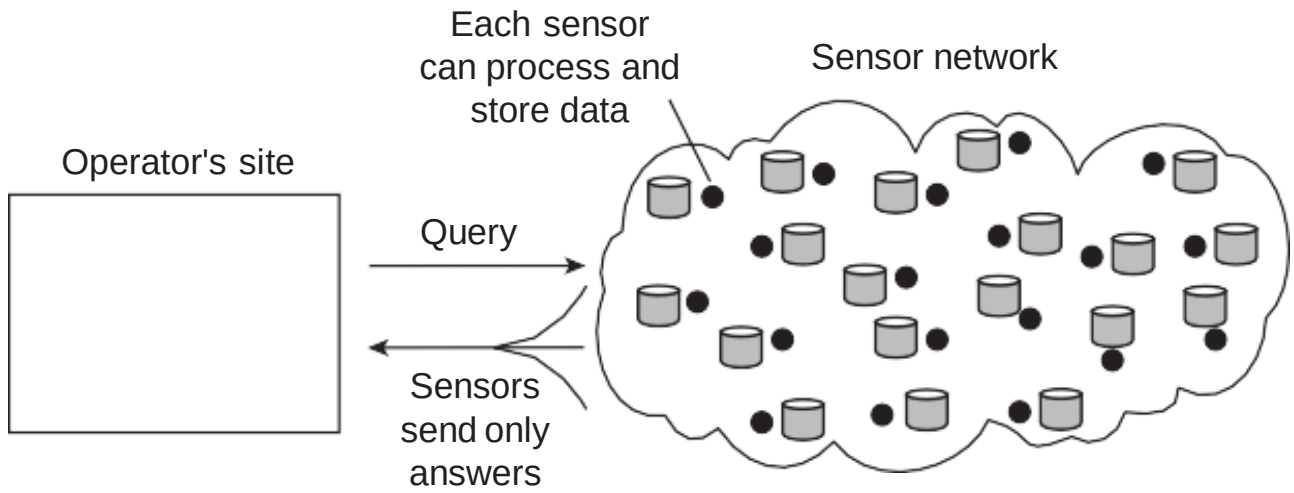
By Tim Hornyak

Tokyo Correspondent, IDG News Service | AUGUST 28, 2015 09:00 AM PT

Redes de sensores como um sistema distribuído



(a)



(b)

Conceitos adquiridos

- Cluster, grade e nuvem.
- Transações/ACID.
- Proxy.
- Sistemas móveis e redes de sensores.