Sistemas Distribuídos

Aula I - O que são SDs?

PROF. DR. RAFAEL TEIXEIRA SOUSA

Objetivos da disciplina

1. O que são Sistemas Distribuídos (SD)
2. Principais arquiteturas de SDs
3. Comunicação
4. Serviço de nomes
5. Coordenação
6. Consistência e Replicação

O que é um SD?

“Um sistema distribuído é uma coleção de elementos computacionais autônomos que aparentam ao usuário um único sistema coerente”

Maarten van Steen

Nos levando as seguintes características

- Concorrência dos componentes
- Falta de um relógio global
- Falhas de componentes independentes

Exemplos

Motivação

Compartilhar Recursos

- Desde componentes de hardware quanto software

Escalabilidade

Confiabilidade

A pesquisa na Web

A tarefa é realizar a indexação do conteúdo da WWW

Grande variedade de conteúdo

Crescimento exponencial

- Número de pesquisas subiu para mais de 10 bilhões/mês – 2013
- 5,6 bilhões por dia - 2019

Google

Uma das maiores e mais complexas instalações de SD da história

Incluem:

- Infraestrutura física subjacente
- Um sistema de arquivo distribuído
- Um sistema de armazenamento distribuído
- Um serviço de bloqueio
- Um modelo de programação que suporta o gerenciamento de cálculos paralelos e distribuídos

Pioneiros no desenvolvimento de ferramentas distribuídas

Tendências

SD está passando por mudanças significativas devido

- Tecnologias de redes pervasivas (ubíqua)
  - Conjunto de computadores interligados heterogêneos
  - Diversidade de tecnologias de comunicação sem fio
  - Dispositivos podem ser interconectados a qualquer momento e em qualquer lugar
- Computação ubíqua combinado ao desejo de suportar mobilidade do usuário em SD
  - Integração de dispositivos móveis a SD
  - Introduz grande desafios aos SD

Tendências

- Crescente demanda por serviços multimídia
  - SD deve oferecer suporte ao armazenamento, transmissão e apresentação
  - Oferecer as mesmas funções para mídias contínuas
    - Incluem dimensão temporal
  - Webcasting
- Visão de SD como um serviço público
  - Maturidade da infraestrutura
  - Empresas investem nessa área para prover recursos em forma de aluguel e vender ao usuário final
  - Tanto para recursos físicos quanto lógicos

Desafios

Heterogeneidade

- Redes
  - Padronização através de protocolos
- Hardware do computador
  - Tipos de dados podem ser representados de forma diferentes em computadores diferentes
- Sistemas Operacionais
  - Chamadas internas são diferentes
- Linguagens de Programação
  - Utilizam representações diferentes para caracteres e estrutura de dados
- Implementações de diferentes fornecedores
  - Devem padronizar as especificações para os programas se comunicarem

Conceitos em SDs

Processos e comunicação:

- Middleware
- Migração de código e Máquinas virtuais

Coordenação

- Escalabilidade
- Acesso concorrente
- Tratamento de Falhas

Consistência

- Transparência

Minimizar estas diferenças

Middleware

- Camada de software entre aplicações e SOs
- Mascara a heterogeneidade
- Fornece um modelo computacional uniforme
  - Para serviços e aplicações distribuídas
- Os modelos incluem
  - Invocação remota de objetos
  - Notificação remota de eventos
  - Acesso remoto a banco de dados
  - Processamento de transação distribuída

Middleware

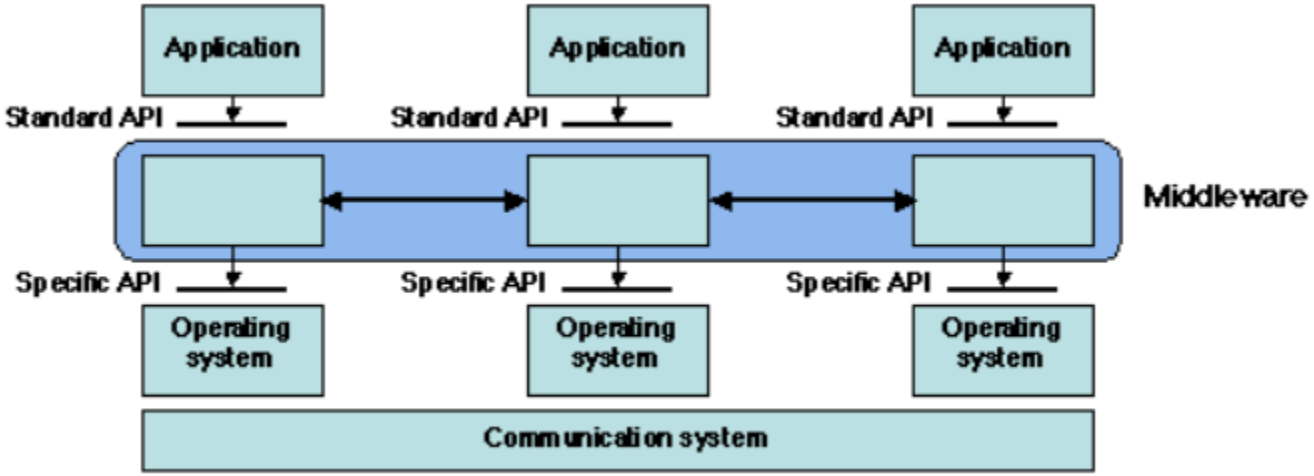
Alguns middleware suportam apenas uma LP

- RMI

A maioria implementados sobre os protocolos de Internet

Exemplo CORBA

- Fornece invocação remota de objetos
- Transparência para o usuário/desenvolvedor



Heterogeneidade e migração de código

Código de programa que pode ser transferido e ser executado no destino

- Exemplo: Applets JAVA

No entanto, código não estará adequado à execução

- Específicos a um conjunto de instruções e a um SO

Estratégia seria utilizar MV

- Tornaria o código executável em uma variedade de computadores hospedeiros
- Exemplo a JVM

Atualmente, a forma mais usada de código móvel na Web

- JAVA Script em páginas Web carregadas pelos navegadores dos clientes

Escalabilidade

Um sistema é dito escalável se permanece eficiente a medida que aumento o número de recursos e usuários

Desafios:

Controlar o custo dos recursos físicos

- Aumentar proporcionalmente os recursos com o número de usuários

Controlar a perda de desempenho

- À medida que o SD escala a perda de desempenho máxima não deve ser maior que O(log n)

Impedir que os recursos de software se esgotem

- Exemplo: a utilização de 32 BITS para a representação dos números Ips
- Uma nova versão está sendo implementada, utilizando 128 Bits

Evitar gargalos de desempenho

- Utilizar algoritmos descentralizados

Escalabilidade

3 medidas pra medi-la

Número de processos e ou usuários

- Escalabilidade de tamanho

Distância máxima entre os nós

- Escalabilidade geográfica

Número de domínios administrativos

- Escalabilidade administrativa

Limitações de Escalabilidade

Serviços centralizados

- Único servidor para todos os usuários

Dados centralizados

- Uma única lista de telefone online

Algoritmos centralizados

- Fazer roteamento com base em informações completas

Características dos algoritmos descentralizados

Nenhuma máquina tem informação completa do estado do SD

As máquina tomam decisões com as informações locais

A falha do algoritmo não “quebra” todo o sistema

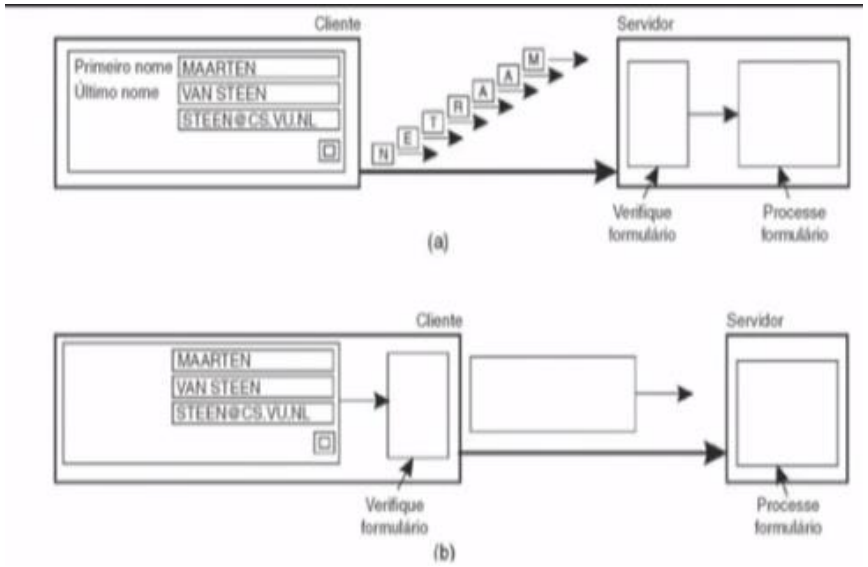
Não existe uma suposição de um relógio global

Técnicas de escalabilidade

Aumentar a disponibilidade dos recursos

Balanceamento de carga

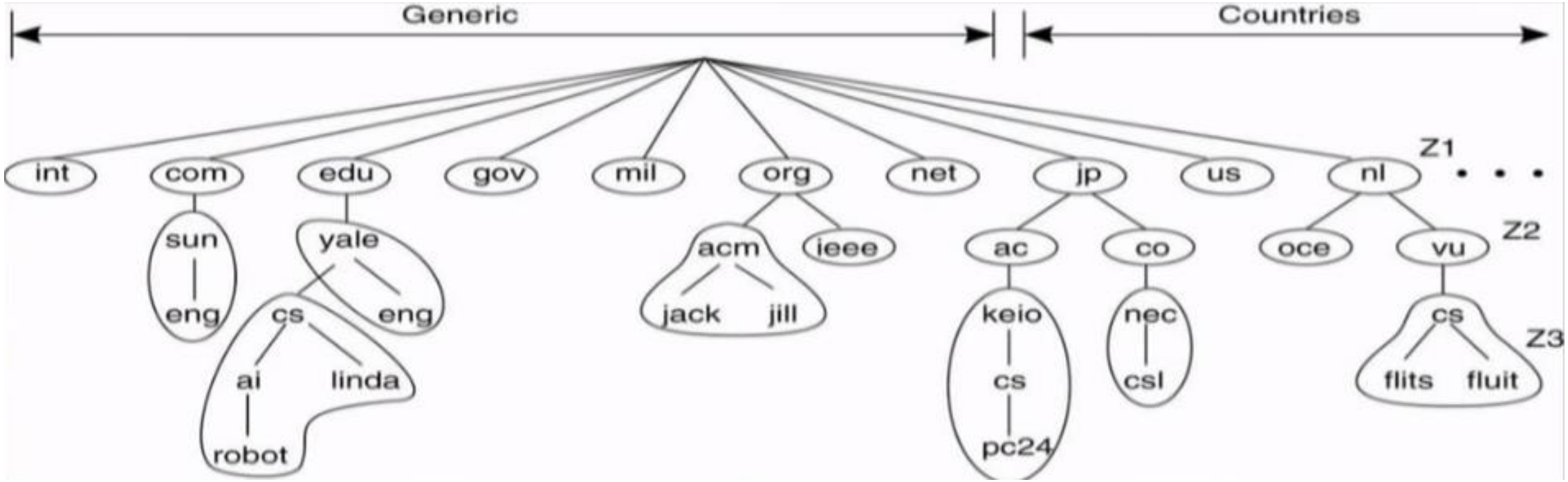
Ocultar a latência de comunicação caso haja grande dispersão geográfica



Técnicas de escalabilidade

Distribuir totalmente o SD.

Por exemplo o DNS



Tratamento de Falhas

Falhas em hardware e software produzem resultado incorretos

Em SD as falhas são parciais

- Portanto o tratamento de falhas é complexo

As técnicas são

- Detecção de falhas
  - Em algumas falhas há mecanismos que conseguem detectá-las
- Mascaramento de falhas
  - Algumas falhas podem ser ocultas ou menos sérias
- Tolerância a Falhas
  - Nem sempre mascaramos tudo
- Recuperação de Falhas
  - Recuperar os dados após as falhas ou retroceder após a falha
- Redundância
  - Para ser tolerante a falhas

Concorrência

Recursos e aplicativos podem ser compartilhados em SD

Portanto, vários usuários podem concorrer ao uso destes recursos

Sendo assim, qualquer objeto distribuído deve garantir o correto funcionamento em um ambiente concorrente

- Mecanismos
- Evitar resultados inconsistentes

Transparência

Ocultação dos componentes distribuídos

O usuário/desenvolvedor tem a impressão de um único sistema

A ANSA (Reference Model for Open Distribuided Processing) define 8 formas de transparência

- Transparência de acesso
  - Ocultar diferenças na representação dos dados e no modo de acesso de um recurso
- Transparência de localização
  - Permite acessar os recursos sem o conhecimento de sua localização física
- Transparência de concorrência
  - Oculta que um recurso pode ser compartilhado por diversos usuários concorrentes

Transparência

- Transparência de replicação
  - Permitem várias instâncias dos recursos para aumentar a confiabilidade e desempenho, sem o usuário saber
- Transparência de falhas
  - Permite a ocultação de falhas, permitindo a conclusão dos programas sem que o usuário saiba
- Transparência de mobilidade
  - Permite a movimentação de recursos e de clientes, sem afetar a execução dos programas
- Transparência de desempenho
  - Permite que o sistema seja reconfigurado para melhor desempenho à medida que as cargas variam
- Transparência de escalabilidade
  - Permite que o sistema e os aplicativos se expandam em escala, sem alterar a estrutura do sistema ou os algoritmos de aplicação

Qualidade de Serviço

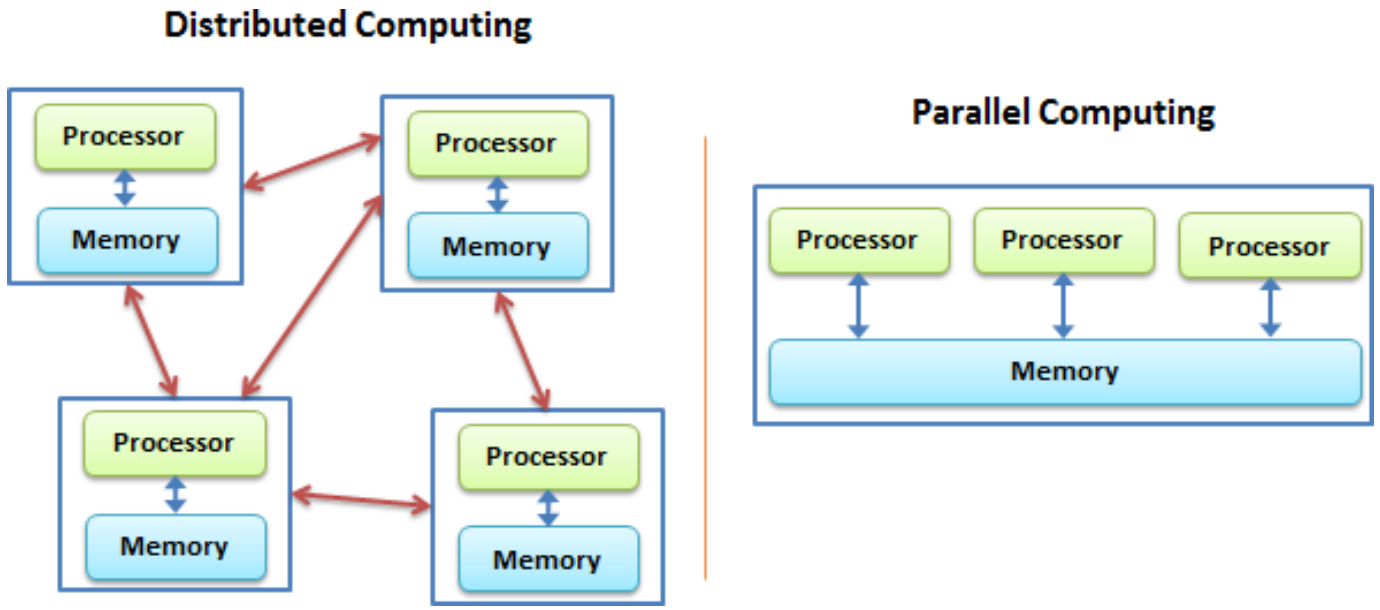
Uma vez fornecido um serviço podemos questionar a sua qualidade em termos

- Confiabilidade
- Segurança
- Desempenho

E em SD podemos questionar também sobre

- Adaptabilidade
- Disponibilidade

Sistemas Distribuídos VS Computação Paralela



Sistemas forte e fracamente acoplados

Compartilhamento de memória