SISTEMAS DISTRIBUÍDOS

FUNDAMENTOS

Sistemas Distribuídos

Coleção de processadores fracamente acoplados interconectados por uma rede de comunicação.

- Não compartilham memória: cada processador tem sua própria memória local;
- Não compartilham relógio (*clock*);
- Processadores se comunicam entre si por meio de redes de comunicação:
  - Do ponto de vista de um processador específico em um sistema distribuído:
    - os demais processadores e seus recursos são **remotos**
    - seus próprios recursos são **locais**
- Processadores podem variar em tamanho e função;
- Podem incluir: pequenos microprocessadores, estações de trabalho, minicomputadores e grandes sistemas de computação de uso geral;
- Processadores são designados por diferentes nomes, como:
  - *sites, nós, computadores, máquinas, host,* dependendo do contexto:
    - *Site* é usado para indicar a localização de uma máquina;
    - *Host* é usado para referenciar um Sistema específico no site;
    - *Servidor* é um host que tem um recurso que outro host, o cliente gostaria de usar;
- O propósito de um sistema distribuído é fornecer um ambiente **eficiente** e **conveniente** para esse compartilhamento de recursos;

Vantagens dos Sistemas Distribuídos

COMPARTILHAMENTO DE RECURSOS: usuários em um site podem utilizar recursos disponíveis em outro site;

Ex: Um usuário no site A pode estar usando uma impressora a laser disponível apenas no site B enquanto um usuário no site B pode acessar um arquivo que reside em A

Em Geral, o **compartilhamento de recursos** em um sistema distribuído fornece mecanismos para:

- Compartilhar arquivos em sites remotos;
- Processar informações em um banco de dados distribuído;
- Imprimir arquivos em locais remotos;
- Usar dispositivos de hardware especializados remotos;
-

VELOCIDADE DE COMPUTAÇÃO:

- Se determinada computação puder ser particionada em uma série de subcomputações que executam concorrentemente:
  - Um sistema distribuído possibilitará a distribuição da computação entre vários sites;
  - Permite executar cálculos de forma concorrente;
  - Permite obter maior **velocidade de computação;**
- Se determinado site estiver sobrecarregado de jobs, alguns deles podem ser movidos para outros sites menos carregados:
  - Esse movimento de Jobs é chamado de **compartilhamento de carga**;

CONFIABILIDADE:

- Se um site falhar em um sistema distribuído, os sites restantes poderão continuar operando:
  - se o sistema é composto por múltiplas instalações autônomas, a falha de uma delas não deverá afetar o resto;
  - se o sistema for composto por máquinas pequenas, cada uma responsável por alguma função crítica, então uma falha poderá interromper a operação do sistema inteiro;
- Em geral, com redundância suficiente (no hardware e nos dados), o sistema pode continuar a operação, mesmo se alguns sites tiverem falhado;

COMUNICAÇÃO

- Quando vários sites estão conectados uns aos outros por uma rede de comunicação, os usuários de diferentes sites têm a oportunidade de trocar informações;
- Em um baixo nível, mensagens são trocadas entre os sistemas
  - Com troca de mensagens, toda funcionalidade de alto nível encontrada em sistemas independentes pode ser expandida para incluir o sistema distribuído;
  - Tais funções incluem:
    - Transferência de arquivos, login, email, navegação pela internet e RPC;
    - Essas funções podem ser executadas através de grandes distâncias.

CONSEQUÊNCIAS:

- As vantagens dos sistemas distribuídos resultaram em uma tendência em toda indústria em direção do **downsizing**:
  - Muitas empresas estão substituindo seus mainframes por redes de estações de trabalho ou computadores pessoais:
    - Mais retorno do investimento (melhor funcionalidade pelo custo);
    - Mais flexibilidade na localização de recursos e expansão das instalações;
    - Melhores interfaces de usuário e manutenção mais fácil.

Tipos de Sistemas Operacionais Distribuídos:

- Duas categorias gerais de sistemas operacionais baseados em rede:

SISTEMAS OPERACIONAIS DE REDE

- Mais simples de implementar
- Mais difíceis para os usuários acessarem e utilizarem os recursos
- Fornece um ambiente no qual os usuários podem acessar recursos remotos:
  - Efetuando login na máquina remota apropriada, ou
  - Transferindo dados da máquina remota para suas próprias máquinas.
- Os usuários devem estar cientes da multiplicidade de máquinas
- Login remoto:
  - Uma importante função de um sistema operacional de rede é permitir que usuários efetuem login remotamente em outro computador;
- Transferência de arquivos remotos:
  - Outra importante função de um sistema operacional de rede é fornecer mecanismos para a transferência de arquivos remotos de uma máquina para outra;
  - Cada computador mantém seu próprio sistema de arquivos local;

SISTEMAS OPERACIONAIS DISTRIBUÍDOS

- Mais complexo de implementar
- Mais fáceis para os usuários acessarem e utilizarem os recursos
- Os usuários acessam recursos remotos da mesma forma que fazem com recursos locais (de forma transparente)
- A migração de dados e processos de um site para outro estão sob controle do sistema operacional distribuído
- Migração de dados:
  - Usuário no site A deseja acessar dados que residem no site B:
    - Transferir o arquivo inteiro para o site A (AFS):
      - A partir daí todo acesso será local;
      - Quando o usuário não precisar mais acessar o arquivo uma cópia do mesmo será enviada de volta ao site B.
    - Transferir apenas partes do arquivo para o site A (NFS e SMB)
      - Se outra parte for necessária, outra transferência é feita;
      - Quando o usuário não precisar mais acessar o arquivo toda parte que foi modificada será enviada de volta ao site B.
- Migração de Computação:
  - Em alguns casos, pode ser mais eficiente transferir a computação em vez dos dados;
  - Se o tempo para transferir os dados for maior que o tempo para executar o comando remoto, o comando remoto deverá ser usado;

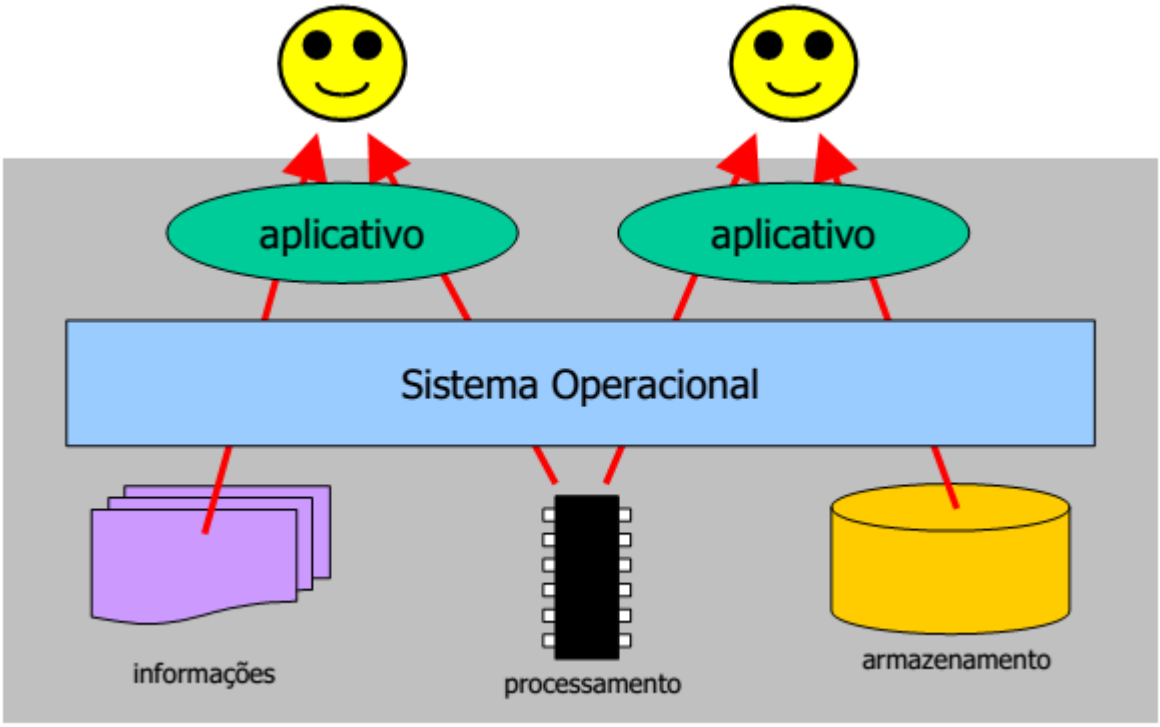
- Alternativas:
  - Chamada de procedimento remoto (RPC);
  - Envio de mensagens.
- Migração de processos:
  - É uma extensão lógica da migração de computação;
  - Pode ser vantajoso executar o processo inteiro, ou partes dele, em sites diferentes;
  - Vários motivos:
    - Balanceamento de carga;
    - Aumento na velocidade de computação;
    - Preferência de *hardware*;
    - Preferência de *software;*
    - Acesso a dados.
- A Web tem muitos aspectos de um ambiente de computação distribuído:
  - Migração de dados: um cliente Web pode acessar arquivos em um servidor Web;
  - Migração de computação: um cliente Web pode disparar uma operação de banco de dados em um servidor Web;
  - Migração de processos: applets java enviado de um servidor Web para um cliente Web, onde são executados.

SISTEMAS OPERACIONAIS

Classificação

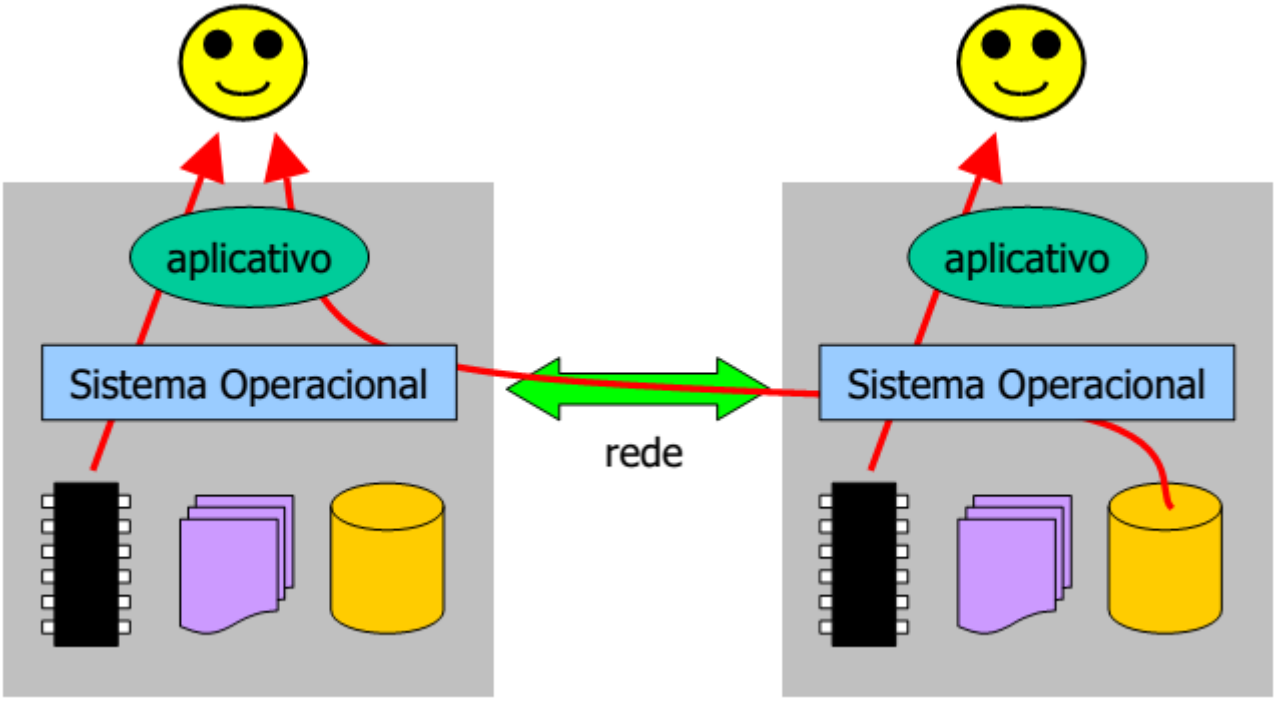
- Sistema Operacional **Centralizado**:
  - Sistema com um computador;
  - Um usuário acessa recursos locais.
- Sistema Operacional de **Rede**:
  - Vários sistemas distintos;
  - Recursos compartilhados entre usuários;
  - Usuários precisam saber onde estão os recursos.
- Sistema Operacional **Distribuído**:
  - Sistemas distintos, mas visão unificada;
  - Recursos estão acessíveis de forma transparente.

Sistema operacional centralizado



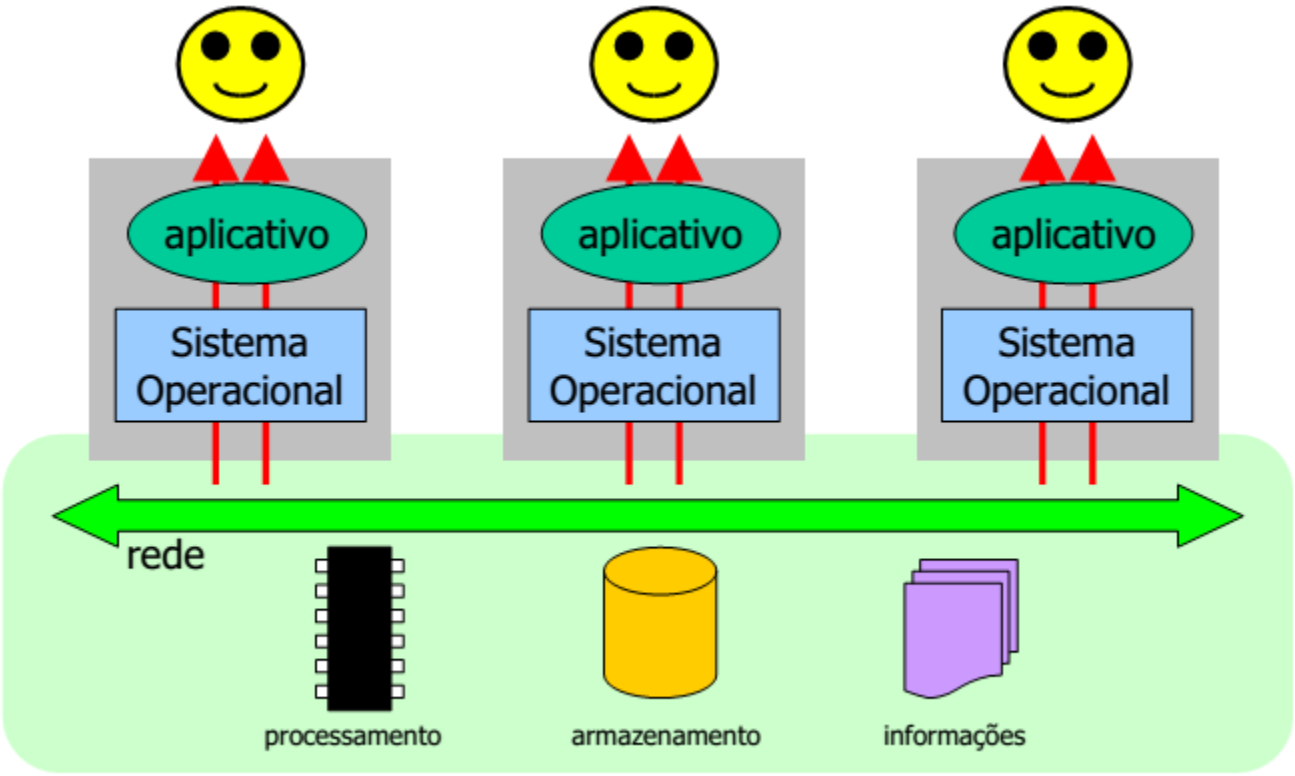
- Aplicado a sistemas convencionais:
  - Recursos centralizados;
  - Arquiteturas mono ou multi-processadas;
  - Sistemas multi-tarefas e multi-usuários.
- Principais características:
  - Compartilhamento de recursos através de interrupções;
  - Todos os recursos são acessíveis internamente;
  - Comunicação entre processos via memória compartilhada ou através de facilidades providas pelo núcleo do sistema.
- Objetivos:
  - Virtualizar os recursos do hardware;
  - Gerenciar uso dos recursos locais;
  - Sincronizar atividades.

Sistema Operacional de rede



- Coleção de computadores conectados através de uma rede:
  - Cada computador possui seu SO local;
  - Cada máquina possui alto grau de autonomia.
- Implementação relativamente simples:
  - SOs incorporam módulos para acessar recursos remotos;
  - Comunicação entre sistemas através de protocolos de transporte:
    - Sockets
    - RPC
- Transferência explícitas:
  - O usuário deve conhecer a localização dos recursos;
  - Os recursos pertencem a computadores específicos.
- Exemplos:
  - Compartilhamento de impressoras e arquivos;
  - Web, E-mail;
  - Serviços de autenticação.

Sistema operacional distribuído



- Objetivos:
  - Construção de um ambiente computacional virtual;
  - Localização dos recursos é abstraída;
  - Localização do processamento é abstraída;
  - Mecanismos transparentes de distribuição, replicação e tolerância a faltas.
- O usuário vê o sistema como um **ambiente virtual**, e não como um conjunto de computadores conectados por uma rede.
- O SO distribuído deve:
  - Controlar a alocação de recursos para tornar seu uso eficiente;
  - Prover um ambiente de computação virtual de alto nível;
  - Esconder a distribuição dos recursos e do processamento.

Quadro Comparativo

Tipo	Serviços	Objetivos
Centralizado	Gerenciamento de processos, memória, dispositivos, arquivos	Gerenciar recursos Máquina estendida Virtualização
de Rede	Acesso Remoto Troca de Informações	Compartilhar recursos Interoperabilidade
Distribuído	Visão global dos recursos (processadores, memória, arquivos, usuários, tempo) Uso do poder computacional	Unificar os computadores em uma visão global Diversas transparências

Tabela Comparativa

	Centralizado (mono ou multi-processado)	de Rede	Distribuído
Se parece com um único processador virtual ?	Sim	Não	Sim
Todas as máquinas executam o mesmo sistema operacional ?	Sim	Não	Sim
Quantas cópias do sistema operacional existem ?	1	N	N
Como a comunicação ocorre ?	Memória compartilhada	Arquivos compartilhados Protocolos de transporte	Trocas de mensagens
Há uma única fila de execução ?	Sim	Não	Não

Definição

Coleção de **sistemas independentes** que se apresentam ao usuário como um **único computador** (Tanenbaum);

- Essa definição implica em:
  - Estações de hardware autônomas;
  - Camada de software que unifica e torna homogênea a visão do sistema.

Vantagens

- Economia:
  - Aproveitar recursos ociosos; é mais barato ter vários processadores interconectados do que um supercomputador.
- Distribuição inerente:
  - Algumas aplicações são distribuídas por natureza;
- Tolerância a falhas:
  - Em caso de falhas de uma máquina, o sistema pode sobreviver, mesmo com desempenho degradado.
- Crescimento incremental:
  - Poder computacional pode ser aumentado através da inclusão de novos equipamentos.
- Flexibilidade:
  - Maior flexibilidade na alocação dos recursos, permitindo que usuários compartilhem dados, processamento e dispositivos.

Desvantagens

- Aplicações mais complexas:
  - Pouco software de alto nível disponível para sistemas distribuídos.
- Segurança:
  - Necessidade de construir mecanismos para controle de acesso às informações.
- Dependência da rede:
  - Falhas;
  - Capacidade de tráfego insuficiente.

Transparência

- Objetivo:
  - Fornecer aos usuários uma imagem única e abstrata do Sistema computacional.
- Níveis de transparência:
  - **Nível de usuário:** o usuário tem a impressão de estar usando um sistema centralizado.
  - **Nível de programador:** o programador tem a ilusão de programar um sistema centralizado.
    - Sintaxe e semântica das chamadas deve ser semelhante.

TIPOS DE TRANSPARÊNCIAS

- **Localização**: os usuários não precisam conhecer a localização dos recursos.
- **Migração**: os recursos podem se mover no Sistema sem alterar seus nomes.
- **Replicação**: os usuários não sabem quantas cópias de um recurso existem.
- **Concorrência**: múltiplos usuários podem compartilhar um recurso sem o perceber (e sem conflitos).
- **Paralelismo**: atividades podem ocorrer em paralelo sem que o usuário tenha de explicitá-las.

TRANSPARÊNCIA DE LOCALIZAÇÃO

- Os usuários não devem estar conscientes da localização física dos recursos.
  - Por exemplo: o nome do recurso não deve conter o nome da máquina na qual o recurso reside.
- Os sistemas transparentes quanto à localização devem possuir um **serviço de nomes**, que mapeia o nome abstrato ao endereço do recurso.

TRANSPARÊNCIA DE MIGRAÇÃO

- Os recursos podem trocar de lugar no sistema.
- Um sistema transparente quanto à migração é também transparente quanto à localização, mas também deve observar outras características de projeto.
- O que pode migrar?
  - Dados;
  - Computação;
  - Processos.
- Dependência residual:
  - Quando um componente do sistema migra, podem haver solicitações em andamento no sistema para ele, que não tomaram ainda conhecimento de sua nova localização. Neste caso, os nós podem guardar um histórico do movimento dos recursos, para que o processo que possua sua localização antiga (nome antigo) possa encontrá-lo.

Migração de Dados

- Transferência de arquivos:
  - Quando um usuário necessita acessar um arquivo x, o arquivo x completo é transferido para a sua máquina local. Se houver alterações, o arquivo deve ser transferido de volta ao site origem.
- Transferência de partes do arquivo: somente as partes do arquivo que serão acessadas são realmente transferidas.

Migração de Computação

- Quando se necessita de um grande volume de dados que se encontra em outro nodo, é mais eficiente transferir a computação do que transferir os dados.
- Migração de computação pode ser feita via RPC ou pelo envio de mensagens (geralmente no modelo cliente-servidor).

Migração de Processos

- A migração de um processo, depois de iniciada a sua execução, pode ser justifica pelas seguintes razões:
  - Balanceamento de carga;
  - Queda de uma máquina;
  - Preferências de hardware;
  - Preferências de software;
  - Proximidade dos recursos.
- Poucos sistemas implementam esse recurso: MOSIX.

TRANSPARÊNCIA DE REPLICAÇÃO

- Por razões de desempenho, o Sistema pode manter cópias de recursos em vários nodos, sem que o usuário ou programador estejam conscientes deste fato.
- Deve ser garantido pelo Sistema que as múltiplas cópias do recurso serão sempre vistas como uma única cópia (coerência entre as cópias).

TRANSPARÊNCIA DE CONCORRÊNCIA

- Os usuários não devem notar que existem outros usuários no sistema. Se dois usuários acessam simultaneamente um mesmo recurso, o Sistema deve garantir a coerência.
- Em sistemas distribuídos, devem ser garantidas as mesmas condições de concorrência de um sistema centralizado.

TRANSPARÊNCIA DE PARALELISMO

- O próprio sistema operacional deve decidir que recursos (e.g. processadores) alocar a uma aplicação distribuída de maneira que critérios de otimização sejam atendidos (balanceamento de carga, tempo de resposta, etc);
- O usuário não deve interferir nessa escolha;
- O número de recursos alocados a uma aplicação pode variar de uma execução para outra.

Flexibilidade

- A inserção de novos módulos no sistema deve ser uma tarefa simples;
- Duas abordagens para a estruturação de um sistema distribuído:
  - Kernel monolítico (e.g. Unix distribuído);
  - Micro-kernel (Mach, Chorus, Amoeba, etc).
- Um microkernel fornece somente serviços básicas:
  - Mecanismo de comunicação entre processos – IPC;
  - Gerência básica de memória;
  - Gerência de processos de baixo nível (trocar de contexto);
  - Estrada e saída de baixo nível.
- Os demais serviços (gerência de arquivos, escalonamento, etc) são providos por serviços em nível de usuário.

Confiabilidade

- Em teoria:
  - Se uma máquina falhar, outra pode assumir suas tarefas;
  - Confiabilidade do grupo aumenta.
- Na prática:
  - Alguns componentes ou serviços são vitais para o sistema;
  - Caso parem, todo o sistema pode cair.
- Aspectos da confiabilidade:
  - Disponibilidade;
  - Segurança;
  - Tolerância a faltas.

Disponibilidade

- Fração de tempo em que o Sistema está disponível para uso;
- Alcançada através de:
  - Redundância de componentes críticos;
  - Se um componente falhar, pode ser substituído.
- Técnicas geralmente utilizadas:
  - Redundância de hardware
    - Processadores, discos.
  - Redundância de software
    - Dois programas distintos efetuando a mesma função.

Segurança

- Autenticidade
  - Os usuários comprovam suas identidades;
  - Senhas, chaves, etc.

- Autorização
  - Estabelecimento de controles de acesso aos recursos;
  - Listas de controle de acesso.
- Privacidade
  - As informações somente podem ser lidas por quem tiver direito;
  - Mecanismos de criptografia.
- Integridade
  - Os dados não podem ser destruídos ou corrompidos por terceiros.
- Não-repudiação
  - Todas as ações podem ser imputadas a seus autores;
  - Mecanismos de auditoria.

Tolerância a Faltas

- O que fazer em caso de falha de um servidor?
- Sistemas distribuídos podem ser projetados para mascarar falhas;
- Abordagens:
  - Replicação de servidores;
  - Execução sem estado (*Stateless execution*).

Faltas, erros e falhas

- Faltas
  - Situações incorretas no estado interno de um Sistema;
  - Ex: um bit de memória inválido, um cabo de rede rompido.
- Erro
  - Decorrência da falta;
  - Estado interno incorreto do software;
  - Ex: queda de uma conexão TCP, variável com valor errado.
- Falha
  - Decorrência do erro;
  - Serviço oferecido ao usuário não cumpre sua especificação;
  - Ex: banco de dados fora do ar, aplicação mostrando dados incorretos.
- Portando: FALTAS → ERROS → FALHAS

Desempenho

- Métricas para medir desempenho:
  - Tempo de resposta;
  - *Throughput* (número de tarefas / tempo);
  - *Utilização do Sistema;*
  - *Uso da capacidade da rede.*
- *Em um Sistema distribuído:*
  - *+ processadores, + memória, + capacidade de armazenamento;*
  - *Pode-se distribuir os processos entre os processadores;*
  - *+ velocidade final de computação?*
  - *+ custo de comunicação!*

Custo de Comunicação

- Componentes do custo de comunicação:
  - Tempo de processamento do protocolo;
  - Tempo de latência do hardware e software de rede;
  - Tempo de transmissão da mensagem.
- Para obter um bom desempenho:

- Reduzir a **comunicação** entre os processadores;
- Buscar manter um bom nível de **paralelismo;**
- Encontrar um ponto de equilíbrio entre ambos!

Granularidade das Tarefas

- Granularidade: tamanho do elemento básico que será distribuído;
- Fina
  - Pequenos conjuntos de instruções executados em paralelo;
  - Muita comunicação → desempenho ruim;
- Média
  - Funções executadas em paralelo (RPC);
- Grossa
  - Processos executados em paralelo;
  - Grande quantidade de código para cada processo;
  - Pouca comunicação → ótimo desempenho.

Escalabilidade

- Noção intuitiva:
  - Um sistema distribuído que opera bem com 10 máquinas também deve funcionar bem com 10.000 máquinas;
  - O desempenho do sistema não deve ser degradado na medida que o número de nós cresce.
- Inimigos da escalabilidade:
  - Componentes centralizados (por exemplo, um único servidor de e-mail para todos os usuários);
  - Tabelas centralizadas (por exemplo, uma única relação on-line de telefones);
  - Algoritmos centralizados (por exemplo, o roteamento de mensagens baseado em informações completas de caminho).

Níveis de Escalabilidade

- Escalabilidade de Arquitetura
  - Escalabilidade de uma arquitetura mede a parte de paralelismo inerente à aplicação que pode ser realizada sobre a arquitetura.
  - O tempo de execução do algoritmo é limitado por suas próprias características e não por características da arquitetura.
- Escalabilidade do Sistema Operacional
  - Um sistema operacional escalável também não deve limitar o desempenho de uma aplicação.
  - Adicionar processadores não vai diminuir o tempo de resposta das chamadas ao sistema, porque nós estamos introduzindo mais recursos a gerenciar.
- Linguagem de programação
  - Que permitam o uso de recursos não centralizados de forma simples;
  - Exemplo: tabelas e hashes distribuídos.
- Aplicação
  - Algoritmos baseados em informação descentralizadas.

Melhorando a Escalabilidade

- Algoritmos descentralizados com as seguintes características:
  - Nenhuma máquina possui informações completas sobre o estado do sistema;
  - Máquinas tomam decisões baseadas apenas nas informações disponíveis localmente;
  - Falha de uma das máquinas não impede o funcionamento do algoritmo;
  - Não existe um relógio global implícito.
- Sistemas escaláveis:
  - Servidores distribuídos: vários servidores cooperam para a execução de um serviço;
  - Estruturas de dados distribuídas, divididas em partes e armazenadas em vários locais do sistema;

- Algoritmos distribuídos: cada servidor executa uma parte do algoritmo.