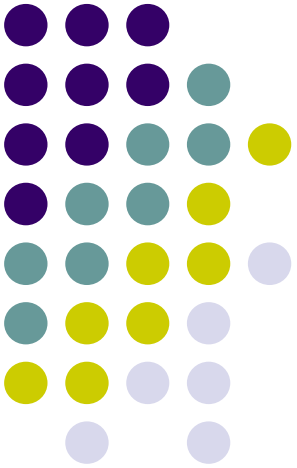
Sistemas Distribuídos

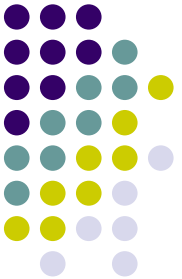
Universidade Federal do ABC

Turma:
Ciência da Computação

Prof. Dr. Francisco **Isidro** Massetto

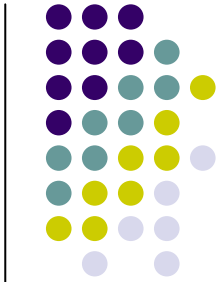


Bibliografia



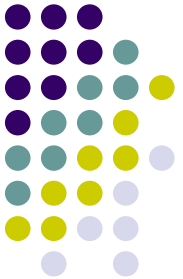
- Andrew S. Tanembaum, Sistemas Distribuídos, Princípios e Paradigmas 2a edição, 2008
- Andrew S. Tanenbaum, Distributed Operating Systems. Prentice-Hall, 1995
- Andrew S. Tanenbaum, Distributed Systems: Principles and Paradigms. Prentice Hall, 2002
- George Coulouris, Jean Dollimore, Tim Kindberg, Distributed Systems - Concepts and Design, 3rd Ed., Addison Wesley, September 2001

Critério de Avaliação



- A definir

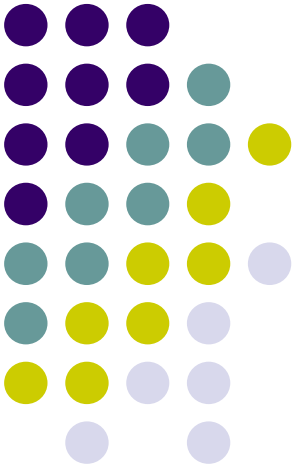
Plano de Ensino: Conteúdo Previsto



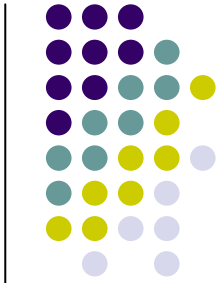
- Comunicação entre Processos
- Evolução, Conceito e Caracterização dos Sistemas Distribuídos;
- Modelo Cliente Servidor;
- Comunicação por passagem de mensagem – Sockets;
- RPC - Remote Procedure Call;
- Java RMI - Remote Method Invocation;
- Sincronização em SD
- Transações Distribuídas
- Sistemas de Arquivos Distribuídos / Case
- Comunicação de Grupos / P2P
- WebServices
- Tecnologia GRID, Cluster e Cloud;

Introdução

Comunicação entre Processos

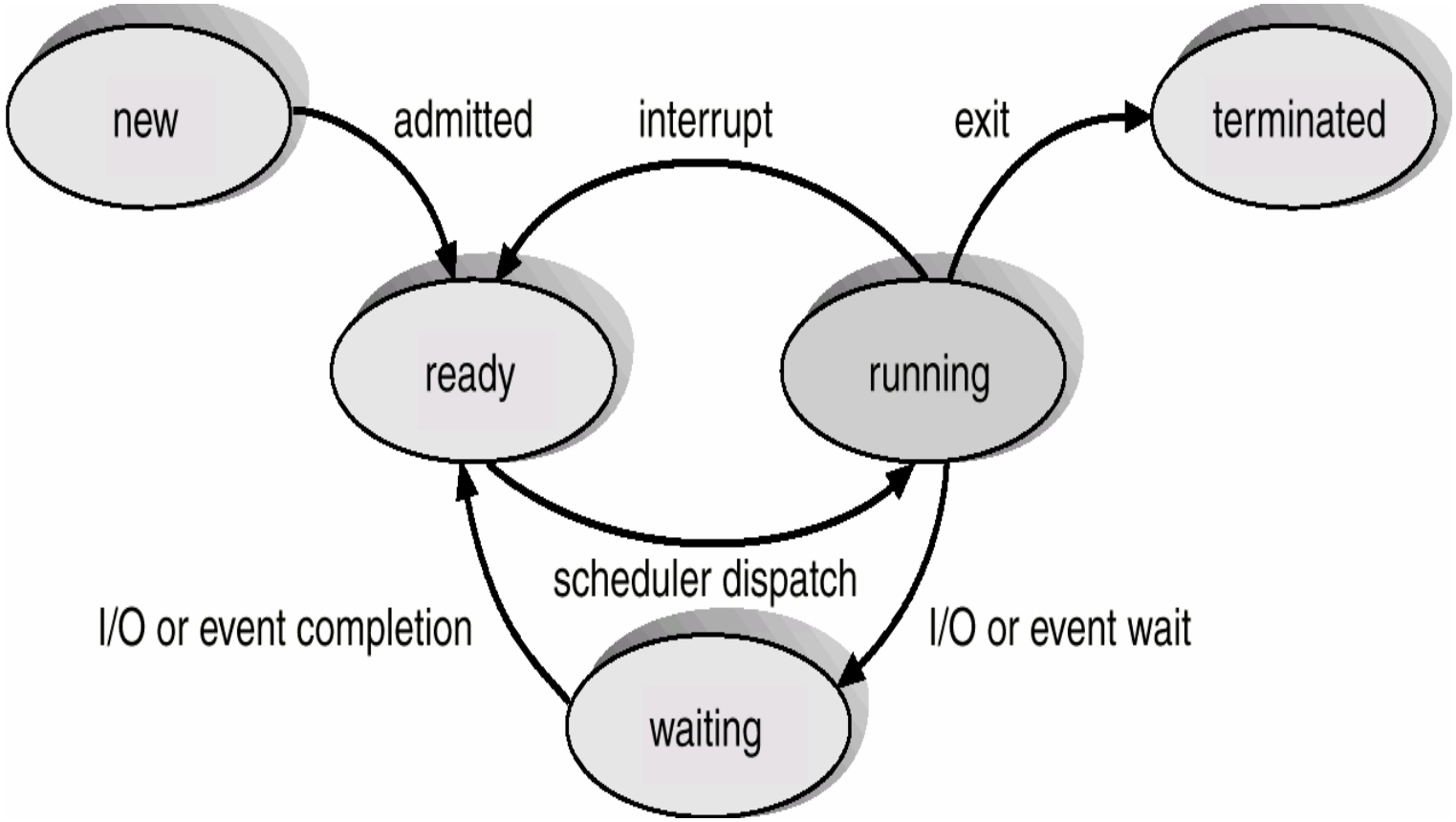


Introdução

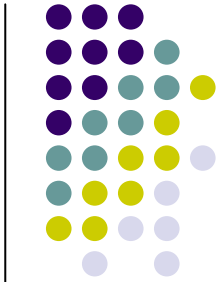


- Sistema Operacional
  - Gerenciador de Recursos
  - Máquina Abstrata/Estendida/Virtual
- Multitarefa
  - Capacidade de executar “ao mesmo tempo” vários processos
  - Alternância entre processos → troca de contexto
  - Escalonamento → Dispatching → Execução

Introdução

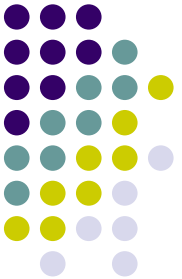

new
admitted
interrupt
exit
terminated
ready
running
scheduler dispatch
I/O or event completion
I/O or event wait
waiting

Comunicação entre Processos



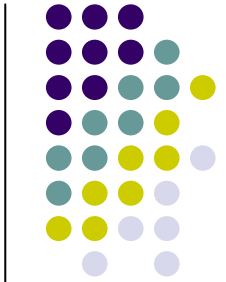
- Vários processos precisam trocar informações
  - Memória/Arquivos
- Áreas de memória específicas
  - Sistema Operacional (buffers, pipes, área de transferência)
  - Determinadas ém código
    - Primitivas de sistema Operacional para Determinar áreas comuns
  - Exemplos em Código C (Windows)

Comunicação entre Processos



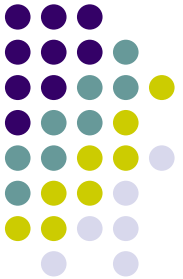
```
int main(int argc, char *argv[]) {
  int op,  *valor;
  void *sharedMemoryLoc;
  HANDLE sharedVar;
  sharedVar = CreateFileMapping(INVALID_HANDLE_VALUE, // use paging file
                        NULL, // no security attributes
                        PAGE_READWRITE, // read/write access
                        0, // size: high 32-bits
                        sizeof(int), // size: low 32-bits
                        "variavel");
  sharedMemoryLoc = MapViewOfFile(sharedVar,FILE_MAP_WRITE,0,0,0);

  valor = (int*)sharedMemoryLoc;
  do{
        scanf("%d",&op);
        if (op == 1)  {
            printf("Digite um valor\n");
            scanf("%d",valor);
        }
        else if (op == 2)  {
            printf("o valor atualizado vale: %d\n",*valor);
        }
  } while (op != 3);
  UnmapViewOfFile(sharedMemoryLoc);
  CloseHandle(sharedVar);
  return 0;
}
```

Comunicação entre Processos



- Linux / Windows
  - Pipe
    - bash$ **comando1** | **comando2**
    - Redireciona a saída do *comando1* para a entrada do *comando2*
    - *stdout* e *stdin*
      - stdout – saída padrão (geralmente vídeo)
      - stdin – entrada padrão (geralmente teclado)
      - “arquivos” específicos do sistema operacional
    - Área de memória do próprio Sistema Operacional para transferência de dados I/O entre processos

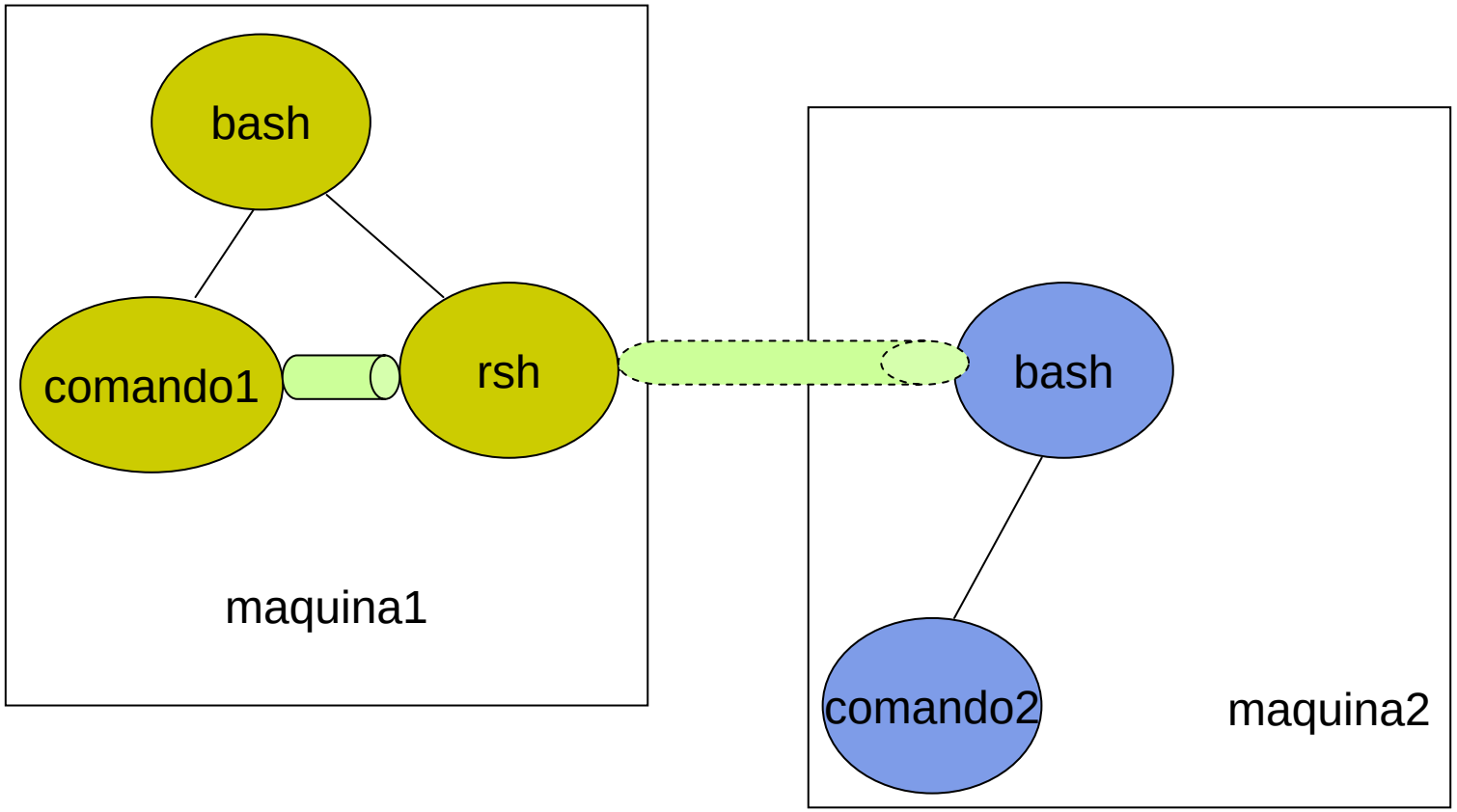
Comunicação entre Processos



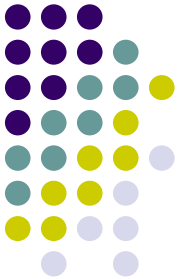
- Spooling
  - Área de impressão de documentos
    - Implementação: FILA de arquivos
    - Todos os processos enviam documentos para o controlador do *spool* que os organiza para, então, imprimi-los
- Área de Transferência do Windows
  - Buffer para armazenamento de objetos (texto, gráficos, tabelas)
  - Imprescindível para as operações CTRL+C e CTRL+V
  - Implementações semelhantes em outros Sistemas Operacionais (Linux, Solaris)

Pipes Remotos

- Linux
  - maq1$ **comando1** | *rsh* “**maquina2**” **comando2**

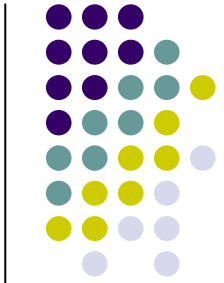


Problemas da Comunicação



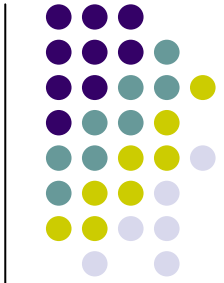
- Suponha a seguinte situação
  - Dois processos necessitam gravar dados em uma mesma área de memória (atualizar um arquivo ou variável, por exemplo)
  - Se ambos acessarem simultaneamente essa memória??
    - Inconsistência
    - Valor final da variável / Conteúdo do arquivo??
- **Condição de Corrida**
- **Seção Crítica**

Necessidade



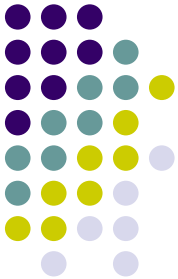
- Sincronização
  - Mecanismo que garanta que apenas um único processo irá acessar um determinado recurso por vez
  - Enquanto um processo utilizar o recurso, os demais esperam
  - Exemplo: Cruzamento
    - Processos: automóveis
    - Recuros: ruas
    - Recurso compartilhado: trecho de cruzamento das ruas

Soluções



- Semáforos
  - Mecanismos do SO que garantem acesso exclusivo (Exclusão mútua) a um recurso
  - Para utilizar o recurso, o processo deve solicitar ao semáforo
    - Caso o semáforo esteja liberado, ele é travado para que o processo use o recurso e os demais esperem
    - Caso o semáforo esteja travado, o processo deve ficar em modo de espera, até que ocorra um evento de liberação do semáforo

Exemplo



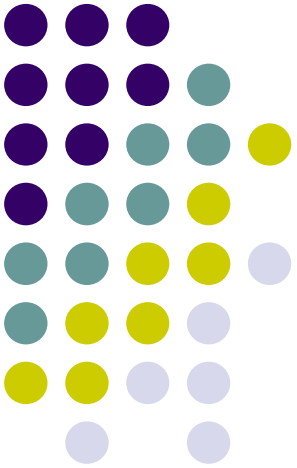
- Programa A

```
shared semaphore s
while(1) {
    lock(s);
    seção crítica();
    unlock(s);
    seção não crítica();
}
```

- Programa B

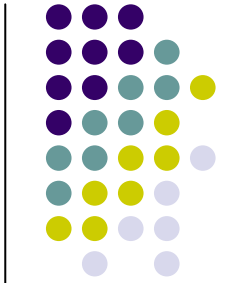
```
shared semaphore s
while(1) {
    lock(s);
    seção crítica();
    unlock(s);
    seção não crítica();
}
```

Conceitos Gerais



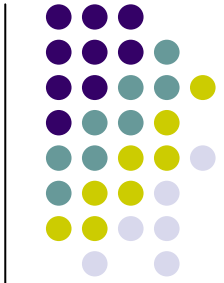
Evolução, Conceito
e Caracterização dos Sistemas
Distribuídos

Introdução: Sistemas Distribuídos



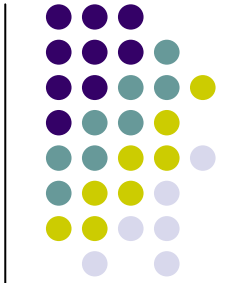
- Sistemas Distribuídos, há mais de uma década, deixaram de ser apenas uma promessa;
- Vários exemplos bem-sucedidos de implementações
  - Meio acadêmico
  - Comércio
  - Indústria
  - Residências;
- Novos desafios:
  - Diversidade dos ambientes computacionais envolvidos;
  - Necessidade de troca de mensagens através da rede;
  - Compreensão do que vem a ser uma aplicação distribuída.
- Ferramentas de apoio à construção dessas aplicações

Sistemas Distribuídos: Evolução



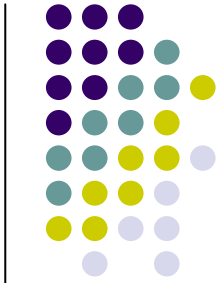
- De 1945 a 1985
  - Computadores de grande porte e alto custo
  - Trabalhavam de modo independente – não havia confiabilidade na comunicação
- De 1985 em diante
  - Microcomputadores com maior poder computacional
    - CPU 8, 16, 32 e 64 bits
    - HyperThreading e MultiCore (devido à limitação de 3.4 GHz na freqüência)
    - Nota: Lançado esta semana processador de 80 core (1.01 TFlops)
      - Pentium M 1.4GHz – 149 MFlops
  - Redes de alta velocidade
    - Local Area Network – LAN
    - Wide Area Network – WAN

Investimentos de Empresas



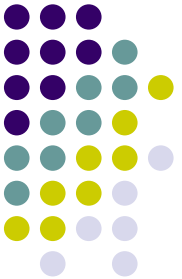
- Hardware
  - Intel
  - AMD
  - Sun
  - SGI
- Software
  - IBM
  - Microsoft
  - Iniciativas de Software Livre (RedHat)
  - Oracle

Sistemas Distribuídos: Conceito



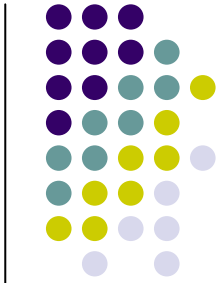
- Definição de Tanenbaum – aperfeiçoada por Mullender:
  - Um sistema distribuído é aquele que se apresenta aos seus usuários como um sistema centralizado, mas que, na verdade, funciona em diversas CPUs independentes;
  - Além disso, um sistema distribuído não deve ter pontos críticos de falha, ou seja, se um componente do mesmo quebrar, isto não deve fazer com que o sistema como um todo falhe;
  - Essa característica de estabilidade é uma de suas principais vantagens em relação a um sistema centralizado.

Sistemas Distribuídos: Conceito



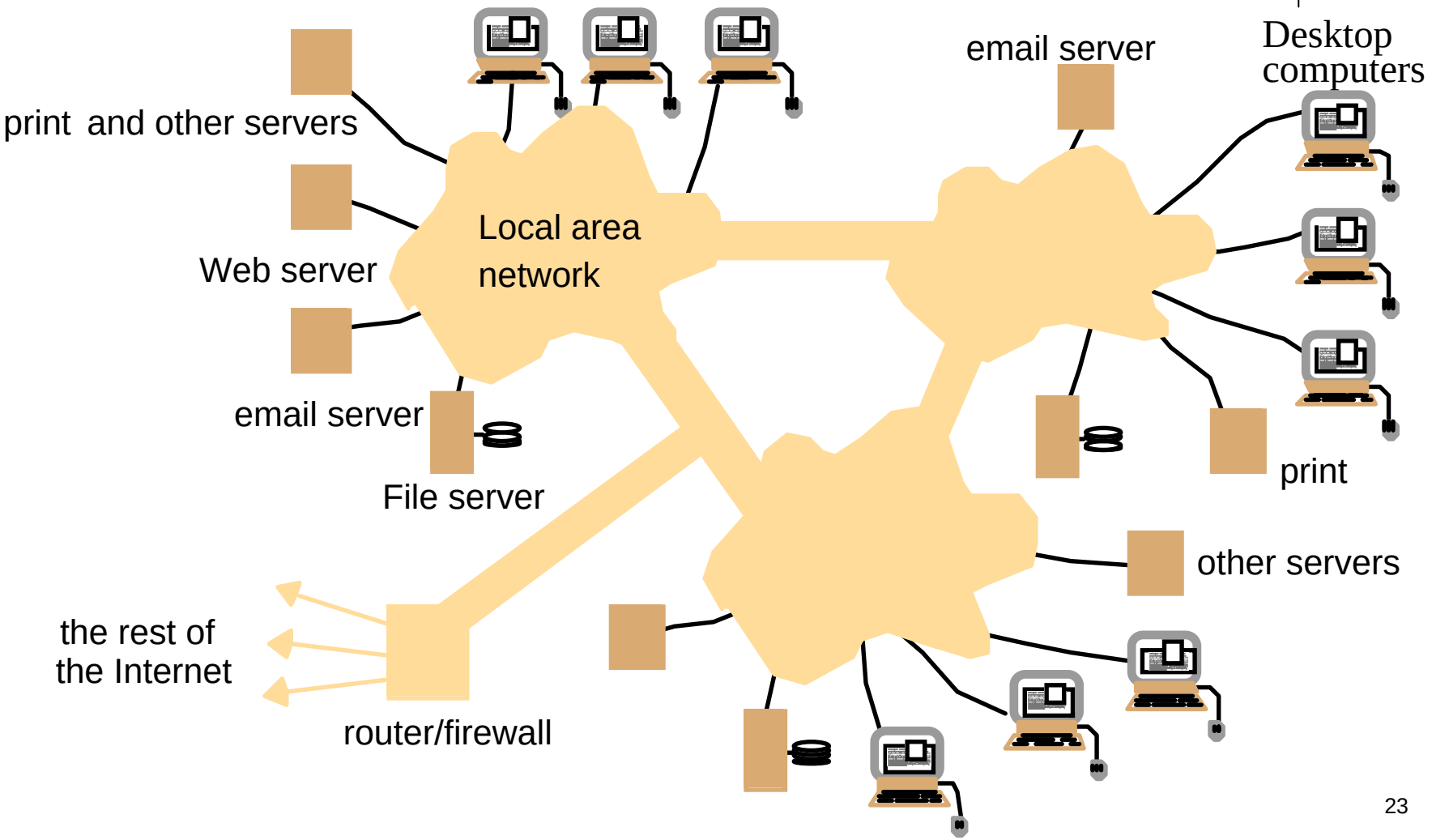
- Definição de Coulouris enfatiza:
  - Devem estar conectados através de uma rede
  - Não precisam estar localizados em uma única sala, ou mesmo próximos entre si
  - Não há limite para a área abrangida por um sistema desse tipo;
  - Computadores devem estar equipados com software de sistemas distribuídos
    - Usuários vêem o sistema como uma entidade única, integrada
    - Embora esteja funcionando em computadores diferentes, situados em locais diversos.

Sistemas Distribuídos: Definições

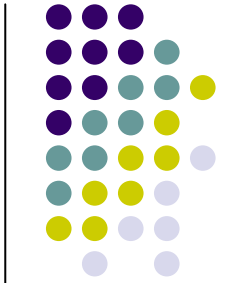


- Definição em 2 aspectos
  - Hardware: Máquinas independentes e autônomas
  - Software: O sistema apresenta-se ao usuário como uma única máquina

Esquema

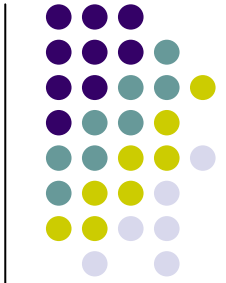

print and other servers
Web server
email server
File server
Local area network
email server
Desktop computers
print
other servers
the rest of the Internet
router/firewall

Sistemas Distribuídos: Características



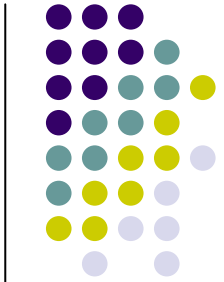
- Comunicação por Trocas de Mensagens:
  - Utilização de meios de comunicação introduz características diferentes no modelo de comunicação;
  - A comunicação fica sujeita a um conjunto de fatores que podem afetar sua confiabilidade → perdas/interferências;
  - Protocolos
    - Garantir confiabilidade e ordem das mensagens
  - Interligação de várias redes
    - Atraso
    - Falha na Transmissão
    - Tempo máximo de espera por uma mensagem

Sistemas Distribuídos: Características



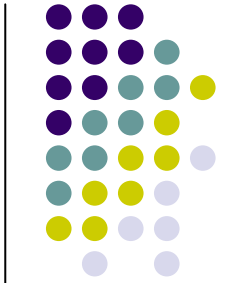
- Modelo de Falhas:
  - Maior probabilidade de falhas → maior quantidade de equipamentos
  - Falhas individuais não podem afetar o sistema como um todo
    - Algoritmos de detecção e recuperação de falhas
    - Replicações, Redundâncias
  - Fatores que levam a falhas
    - Elementos de interligação
      - Interferências, cabeamento mal estruturado, intempéries naturais
      - Falta de alimentação elétrica nos equipamentos
    - Nodos (Nós) do sistema
      - Falhas de software (erro de programação)
      - Falhas físicas (*crash* em equipamento)

Sistemas Distribuídos: Características



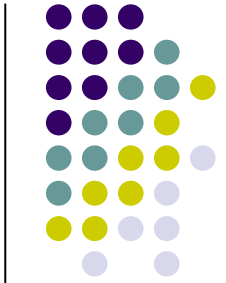
- Sincronismo:
  - Sistema centralizado
    - Sistema de sincronismo concentrado em um único núcleo com regras
  - Sistema Distribuído
    - Informação está necessariamente dividida por diversas máquinas e discos;
  - Problemas com cada Nodo do sistema
    - Não compartilham relógio global → não possuem mesmos "horários"
    - Disputa por recursos → algoritmos mais complexos e que podem ser afetados pela comunicação entre os processadores

Sistemas Distribuídos: Características



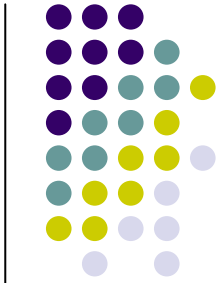
- Segurança:
  - Vulnerabilidade de Redes
    - Observação das Mensagens → *sniffing*
    - Ataques → DoS, DDoS, Buffer Overflow
  - Validação da identidade dos usuários
    - Usuário válido
    - Credenciais para utilizar recursos
  - Interligação de serviços públicos e privados
    - Internet
    - Políticas de segurança diversas (níveis de acesso)
    - Sistemas Heterogêneos

Sistemas Distribuídos: Características



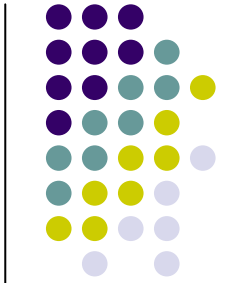
- Heterogeneidade:
  - Sistema amplo
  - Variedade de Arquiteturas
    - CISC, RISC, Vetoriais
    - 32, 64 bits
    - Intel, Mac, Sparc, Mainframes, etc
  - Variedade de Sistemas Operacionais
    - Linux, Windows, MacOS, Solaris, AIX, HP-UX, BSD
  - Representação interna de valores.
    - O primeiro octeto é o de peso mais significativo (big endian);
    - O primeiro octeto é o de peso menos significativo (little endian).

Sistemas Distribuídos: Características



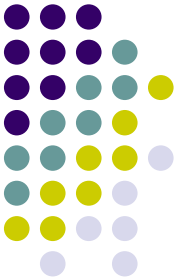
- Desempenho:
  - Desempenho de um SD deve ser compatível com um Sistema Centralizado
  - Divisão do processamento entre os diversos nós
  - Custo da comunicação
  - Com SD é possível atingir desempenhos jamais imagináveis com Sistemas Centralizados

Sistemas Distribuídos: Características



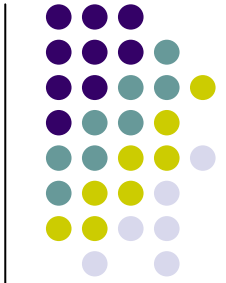
- Custo:
  - Pode-se obter um SD com a mesma quantidade de processadores de um SC com um custo muito menor
  - Utilização de múltiplos processadores de baixo custo interligados em rede
  - Capacidade de se obter um desempenho muito maior com o mesmo investimento do que em um Sistema Centralizado

Sistemas Distribuídos: Características



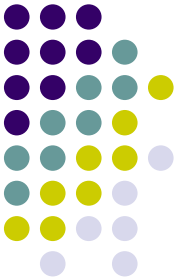
- Distribuição Geográfica:
  - Componentes fisicamente distantes uns dos outros
  - Aplicações Inerentemente Distribuídas
    - Venda de passagens aéreas
    - Sistema integrado de gestão empresarial
    - Internet...

Sistemas Distribuídos: Características



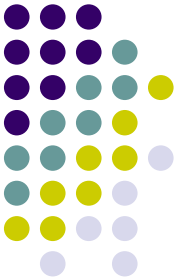
- Compartilhamento de Recursos:
  - Periféricos de alto custo
    - Impressoras laser coloridas, discos RAID com interface SCSI
  - Dados em um ambiente centralizado
    - Bases de dados de transações financeiras
  - Problema
    - Controle de acesso e concorrência
    - Mecanismos mais complexos

Sistemas Distribuídos: Características



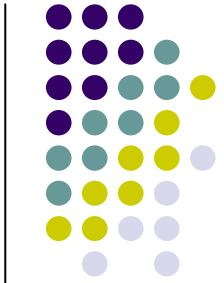
- Capacidade de Expansão (Scalability):
  - Sistema Centralizado
    - Limite físico para o número máximo de processadores
    - Limite para discos, memória
  - Sistema Distribuído
    - Necessidade de mais desempenho → acoplar mais máquinas
    - Em função da demanda, aumenta-se o número de nós do sistema
    - Problema → interligação → congestionamento

Sistemas Distribuídos: Características



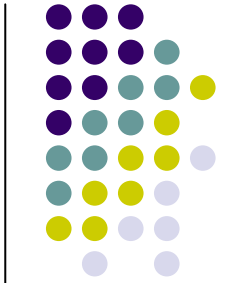
- Disponibilidade:
  - Tempo em que o sistema é “utilizável”
  - Desejável = 100%
  - Máquinas independentes podem continuar mantendo o sistema em operação no caso de falhas em outras máquinas
  - Sistema deve ser projetado para tal
  - Exemplos
    - Sistema bancário
    - Web Servers
  - Garantia de Disponibilidade
    - Redundância (software, hardware)
    - Algoritmos de recuperação

Sistemas Distribuídos: Características



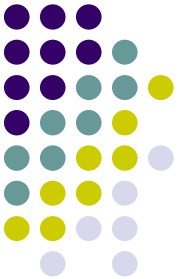
- Concorrência:
  - Mais complexo do que em um Sistema Centralizado
  - Mecanismos de controle de concorrência devem ser revistos (semáforos, mutexes)
  - Maior número de máquinas → maior concorrência
  - Rede influencia o acesso aos recursos

Sistemas Distribuídos: Características



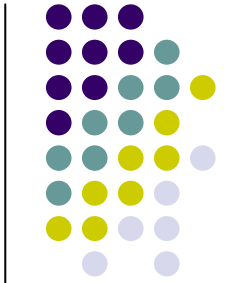
- Transparência
  - Localização: o usuário não precisa saber onde estão os recursos
  - Replicação: não é necessário saber quantas cópias do recurso existem
  - Migração: recursos podem mudar de lugar sem a alteração de nomes
  - Concorrência: recursos podem ser disputados sem conhecimento do usuário
  - Paralelismo: várias atividades podem ocorrer simultaneamente sem o conhecimento dos usuário

Vantagens



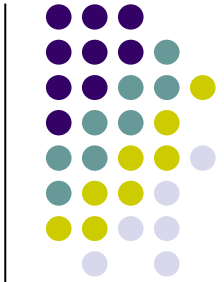
- Compartilhamento de dados: base de dados comum;
- Compartilhamento de dispositivos: acesso compartilhado a periféricos;
- Comunicação: torna-se mais simples e mais rápida a comunicação entre pessoas. Além disso, é possível: transferência de arquivos entre nós, login remoto, etc;
- Flexibilidade: dividir a carga de trabalho entre os nós da rede;
- Confiabilidade: se um nó falha os demais poderão continuar operando;
- Velocidade de computação: maior poder computacional obtido através de concorrência. Há a possibilidade de distribuir uma computação particionada a vários nós para executarem concorrentemente;
- Performance a baixo custo: preço baixo dos PCs;
- Escalabilidade: aumentar o número de nós

Desvantagens (Desafios)



- Software
  - Falta de experiência
  - Mudança de Paradigma
  - Conhecimento sobre a distribuição
  - Quanto deve ser feito pelo sistema e quanto pelo usuário?
- Rede
  - Perda de mensagens
  - Sobrecarga na comunicação
  - Dimensionamento da rede
- Segurança
  - Autenticação, credenciais, bloqueios

Conceitos de Hardware

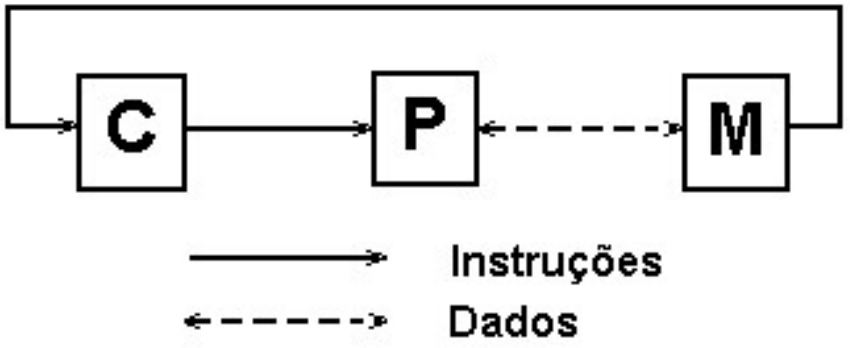


- Taxonomia de Flynn

	(SD) Single Data	(MD) Multiple Data
(SI) Single Instruction	SISD Máquinas de Von Newmann convencionais	SIMD Máquinas Vetoriais (CM-2, MasPar)
(MI) Multiple Instruction	MISD Sem representante (até o momento)	MIMD Multiprocessadores e Multicomputadores (nCUBE, Intel Paragon, Cray T3D)

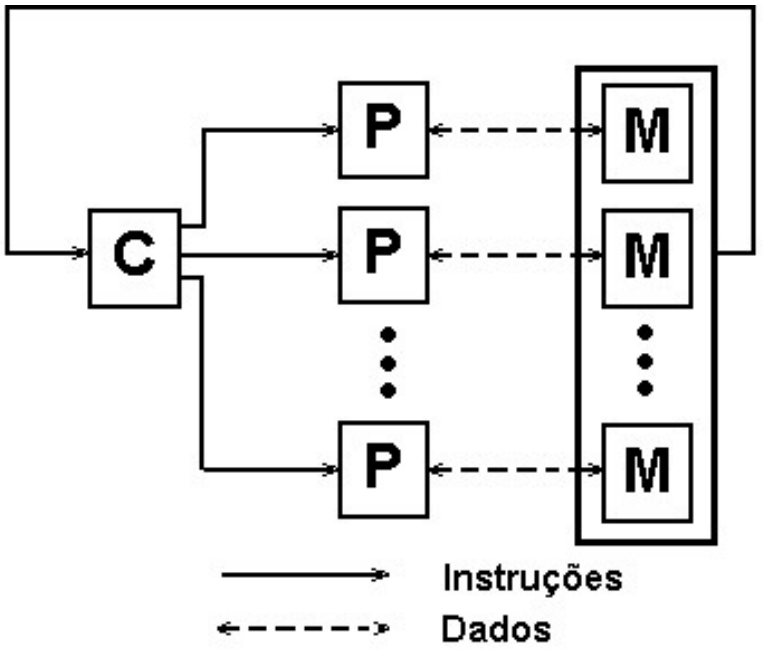
SISD

- Computadores com um único processador



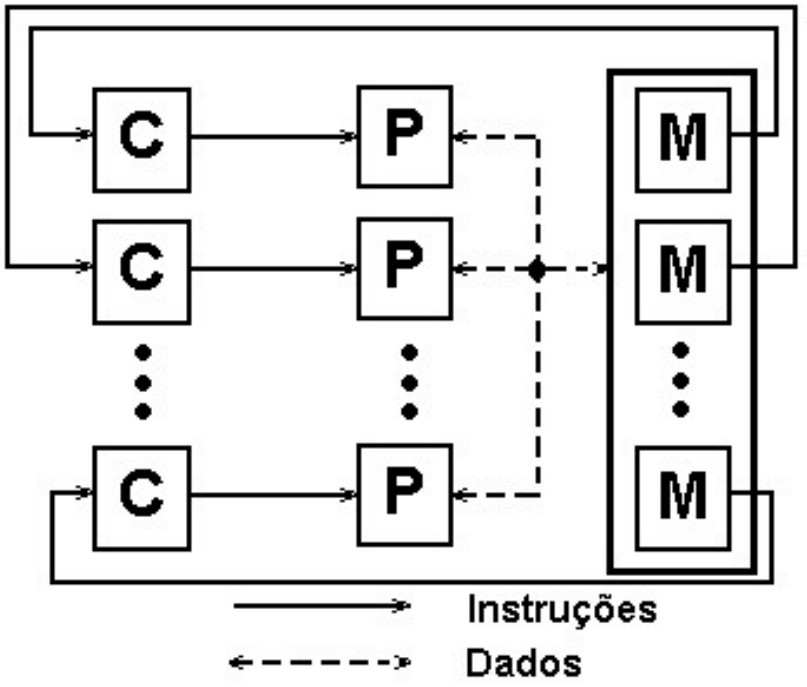
SIMD

- Máquinas vetoriais e matriciais que executam a mesma instrução em paralelo sobre vários conjuntos de dados



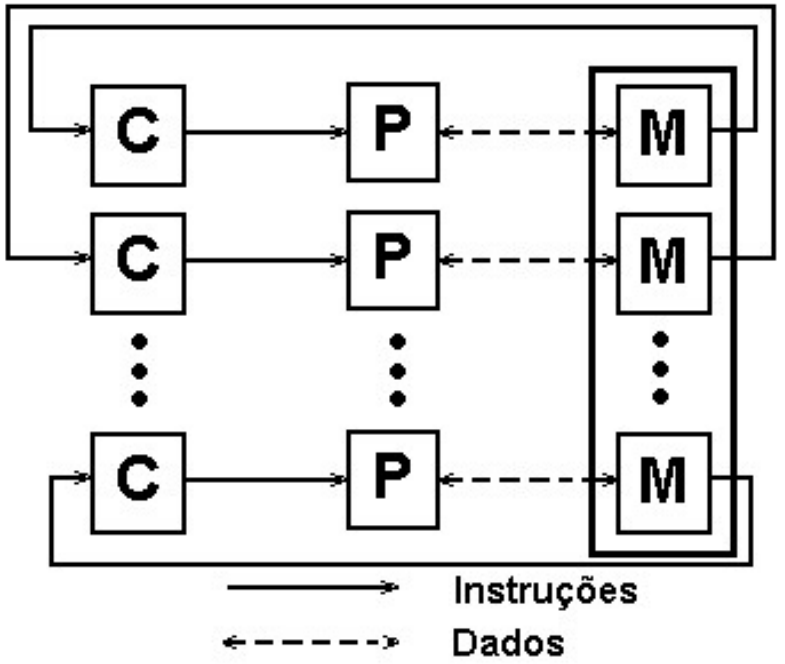
MISD

- Modelo Conceitual
- Não existem implementações como produtos
- Pesquisas desenvolvidas (Flynn)

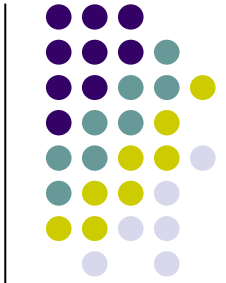


MIMD

- Memória Compartilhada: máquinas SMP
- Memória Distribuída: clusters

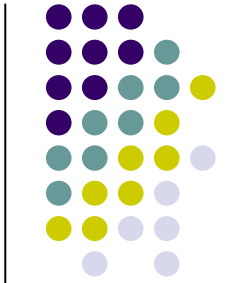


Conceitos de Software



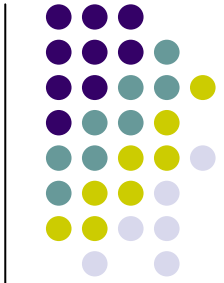
- Acoplamento
  - Grau de dependência entre os diversos componentes do sistema
  - Também define como o usuário vê o sistema
- Fortemente Acoplados
  - Os diversos processadores cooperam na execução de uma tarefa (processamento paralelo)
- Fracamente Acoplados
  - Máquinas independentes, com sua própria memória, HD, processador que se comunicam quando necessário

Sistema Operacional de Rede



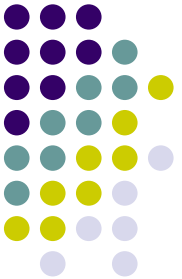
- Software Fracamente Acoplado
- Hardware Fracamente Acoplado
- Rede de Estações de Trabalho conectadas por uma LAN
  - Cada estação de trabalho pode ter ou não disco rígido
  - Cada estação de trabalho tem o seu próprio sistema operacional
  - Todos os comandos são normalmente executados localmente
  - Eventualmente é possível fazer uma conexão remota com outra estação de trabalho
- Forma mais conveniente de comunicação e compartilhamento de informação: Sistema de Arquivo Global Compartilhado (NFS)
  - Implementado em uma ou mais máquinas chamadas Servidores de Arquivos
  - Servidores de Arquivos aceitam requisições de programa de usuários (chamados clientes)

Sistema TimeSharing Multiprocessadores



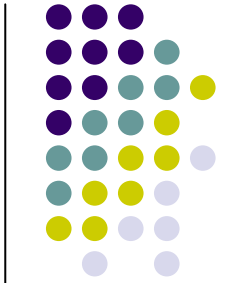
- Software Fortemente Acoplado
- Hardware Fortemente Acoplado
- Máquinas voltadas para propósitos específicos, como bases de dados dedicadas e processamento de imagens;
- Todo o projeto pode ser centralizado => visão de um único processador.
- Existência de uma única fila de espera (processos que não estão bloqueados e prontos para execução);
- Memória compartilhada por todos os processadores;
- Único sistema de arquivos;

Sistema Distribuído



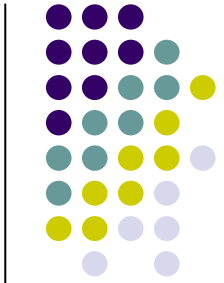
- Software Fortemente Acoplado
- Hardware Fracamente Acoplado
- Cada usuário tem a mesma imagem do sistema.
- Impressão de um único processador Virtual
- Um mecanismo de comunicação interprocesso único e global - qualquer processo pode se comunicar com qualquer outro;
- Gerenciamento de processos precisa ser o mesmo no sistema todo (criação, destruição, começo, interrupção de processos);
- Único conjunto de chamadas de sistema;
- Sistema de arquivo também precisa ter as mesmas caracteristicas;
- Cópias identicas do kernel executam em todas as CPUs do sistema (escalonamento, swapping, paginação, etc).

Diferenças



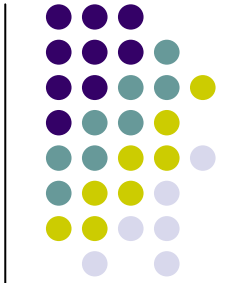
	SO de Rede	Sistema Distribuído	Sistema Timesharing Multiprocessadores
Parece um único processador virtual?	Não	Sim	Sim
Tem o mesmo SO?	Não	Sim	Sim
Número de cópias do SO	N	N	1
Como é sua comunicação?	Arquivos Compartilhados	Mensagens	Memória Compartilhada
Protocolos Requeridos?	Sim	Sim	Não
Uma única fila de execução?	Não	Não	Sim
Arquivos tem a mesma semântica?	Usualmente não	Sim (para ter uma imagem única do sistema)	Sim

Aspectos de Projeto



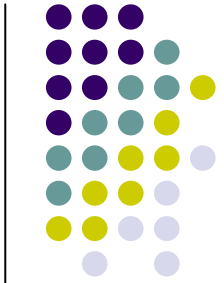
- Transparência
  - Esconder o fato do usuário estar tratando com um sistema Distribuído
    - Ex.: Make no Unix → compilações em paralelo
  - Transparência para programadores (nível mais baixo)
    - Interface das chamadas do sistema oculta a existência de múltiplos processadores

Aspectos de Projeto



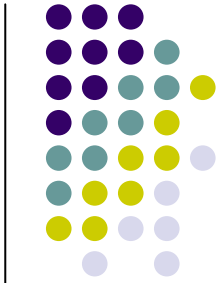
- Flexibilidade
  - Kernel Monolítico
    - Concentra todas as funcionalidades, embora não sejam totalmente utilizadas
  - MicroKernel
    - Kernel com funcionalidades mínimas, para que máquinas com propósitos específicos tenham serviços configuradas

Aspectos de Projeto



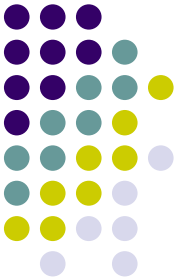
- Confiabilidade
  - Teoria: se uma máquina falha, outra pode realizar seu trabalho
  - Prática: cooperação de alguns servidores específicos
- Confiabilidade: probabilidade do sistema estar operando normalmente em um intervalo de tempo

Aspectos de Projeto



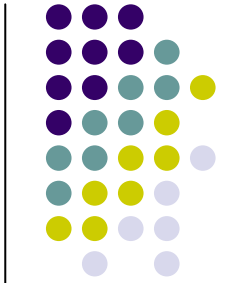
- Disponibilidade
  - Fração do tempo em que o sistema é “usável”
  - Pode ser aumentada por meio de redundância
    - Partes de software ou hardware podem ser replicadas
    - Maior número de réplicas → maior probabilidade de inconsistências
  - Segurança: integridade das informações
  - Tolerância a Falhas: capacidade do sistema se recuperar e esconder a falha do usuário

Aspectos de Projeto



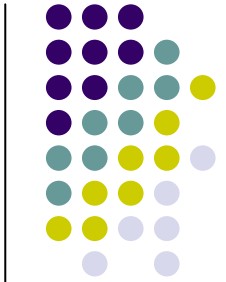
- Desempenho
  - Não pode ser pior do que um sistema Centralizado
  - Medidas
    - Tempo de resposta
    - Throughput – número de jobs por intervalo de tempo
    - Quantidade de recursos consumidos na rede
    - Resultados de Benchmarks
  - Comunicação
    - Paralelismo de Granularidade Fina
      - Pouca computação
      - Muita interação
    - Paralelismo de Granularidade Grossa
      - Muita computação
      - Pouca interação

Aspectos de Projeto



- CGM
  - Coarse-Grained Multiprocessor
  - Ideal é sempre manter o processador ocupado e a rede pouco utilizada
  - Dificuldade
    - Encontrar o equilíbrio entre comunicação e processamento
    - Dificuldade em encontrar aplicações que façam uso desta abordagem

Aspectos de Projeto



- Escalabilidade
  - Com 10 processadores
    - Sistema OK
    - Desempenho proporcional à quantidade de máquinas
  - Com 50 processadores
    - Sistema Ok
    - Desempenho começa a ficar comprometido
  - Com 100 processadores
    - Sistema falha
  - Desafio: Conciliar desempenho com escalabilidade