# Sumário

# Conteúdo 0 - Apresentação da Disciplina: Sistemas Distribuídos

**PLANO DE ENSINO**

**CURSO:** Ciência da Computação
**SÉRIE:** 7º semestre
**DISCIPLINA:** Sistemas Distribuídos
**CARGA HORÁRIA SEMANAL:** 04 horas-aula
**CARGA HORÁRIA SEMESTRAL:** 80 horas-aula

I – EMENTA

Definição e histórico de sistemas distribuídos. Arquiteturas. Processos. Comunicação. Nomeação. Sincronização. Tolerância à falha. Segurança. Cluster de Alto-desempenho Beowulf e Grids.

II – OBJETIVOS GERAIS

Apresentar ao aluno os principais aspectos que envolvem o projeto e a implementação de um sistema distribuído, com ênfase no Cluster Beowulf.

III – OBJETIVOS ESPECÍFICOS

Mostrar ao aluno: as principais características que um sistema operacional deve possuir, formas de organização, algoritmos envolvidos e formas de implementação.

IV – CONTEÚDO PROGRAMÁTICO

1. Introdução aos Sistemas Distribuídos;
2. Arquiteturas em Sistemas Distribuídos;
3. Processos em Sistemas Distribuídos;
4. Comunicação em Sistemas Distribuídos;
5. Tolerância à falha em SD;
6. Segurança em Sistemas Distribuídos;
7. Serviços WEB (Introdução aos Web Service).
8. Clusters de Computadores;
9. Entendendo o Cluster Beowulf;
10. Diferenças entre Cluster e GRID;
11. Introdução a Programação Paralela;
12. Ferramentas de Análise de Desempenho e Otimização em SD.

V – ESTRATÉGIA DE TRABALHO

Aulas expositivas. Listas de exercícios para serem resolvidos fora da sala de aula para fixação dos assuntos abordados nas aulas expositivas. Proposta de projetos extracurriculares.
Sugestão de Projetos: Construção de clusters utilizando (MPICH, PVM e OpenMosix) para testar a performance utilizar o POV RAY para renderizar imagens.

**VI – AVALIAÇÃO**

Provas bimestrais e trabalhos.

VII – BIBLIOGRAFIA

## Básica

TANENBAUM, Andrew S.; STEEN, Maarten Van - Sistemas Distribuídos – Ed. Prentice Hall Brasil - 2007.
TANENBAUM, Andrew S. - Sistemas Operacionais Modernos - Livros Técnicos e Científicos Ed. S.A - 1999.
COULOURIS, George; DOLLIMORE, Jean ; KINDBERG, Tim - Sistemas Distribuídos - Conceitos e Projeto - Ed. Bookman Companhia . – 2007.

## Complementar

MARQUES, Jose Alves; GUEDES, Paulo - Tecnologia de Sistemas Distribuídos – Ed. FCA - 1998.
RIBEIRO, Uira - Sistemas Distribuídos - Axcel Books - 2005.
SAMPAIO, Cleuton. SOA e Web services em Java. Rio de Janeiro: Brasport, 2006.
ALBINADER, Neto: Jorge, Abilio. Web services em Java: Rio de Janeiro: Brasport, 2006.

JOSUTTIS, Nicolai M..SOA na Prática, A Arte da Modelagem de Sistemas Distribuídos. Rio de Janeiro: Alta Books, 2008.

# Conteúdo 1 - Introdução aos Sistemas Distribuídos

Introdução

1.1 Definição de um Sistema Distribuído

“Um sistema distribuído é um conjunto de computadores independentes que se apresenta a seus usuários como um sistema único e coerente”

Essa definição tem vários aspectos importantes. O primeiro é que um sistema distribuído consiste em componentes (isto é, computadores) autônomos. Um segundo aspecto é que os usuários, sejam pessoas ou programas, acham que estão tratando com único sistema. Isso significa que, de um modo ou de outro, os componentes autônomos precisam colaborar. Como estabelecer essa colaboração é o cerne do desenvolvimento de sistemas distribuídos. Observe que nenhuma premissa é adotada em relação ao tipo de computadores. Em princípio, até mesmo dentro de um único sistema, eles poderiam variar desce computadores centrais (mainframes) de alto desempenho até pequenos nós em redes de sensores. Da mesma maneira, nenhuma premissa é adotada quanto ao modo como os computadores são interconectados.

Uma característica importante é que as diferenças entre os vários computadores e o modo como eles se comunicam estão, em grande parte, ocultas aos usuários e também deveria ser relativamente fácil expandir ou aumentar a escala de sistemas distribuídos. Essa característica é uma conseqüência direta de ter computadores independentes, porém, ao mesmo tempo, de ocultar como esses computadores realmente fazem parte do sistema como um todo.

1.2 Transparência da distribuição

Uma meta importante de um sistema distribuído é ocultar o fato de que seus processos e recursos estão fisicamente distribuídos por vários computadores. Um sistema distribuído que é capaz de se apresentar a usuários e aplicações como se fosse apenas um único sistema de computador é denominado transparente.

1.2.1 Tipos de transparência

- Acesso
  - Oculta diferenças na representação de dados e no modo de acesso a um recurso.
- Localização
  - Oculta o lugar em que um recurso está localizado.
- Migração
  - Oculta que um recurso pode ser movido para outra localização.
- Relocação
  - Oculta que um recurso pode ser movido para outra localização enquanto em uso.
- Replicação
  - Oculta que um recurso é replicado.
- Concorrência
  - Oculta que um recurso pode ser compartilhado por diversos usuários concorrentes.
- Falha
  - Oculta a falha e a recuperação de um recurso.

1.3 Ciladas

Sistemas Distribuídos são diferentes do software tradicional porque os componentes estão dispersos por uma rede. Não levar essa dispersão em conta durante o projeto é o que torna tantos sistemas desnecessariamente complexos e resulta em erros que precisam ser consertados mais tarde. Peter Deutsch, que, na época, trabalhava na Sun Mirosystems, formulou esses erros como as seguintes premissas falsas que todos adotam ao desenvolver uma aplicação distribuída pela primeira vez:

1. A rede é confiável.
2. A rede é segura.
3. A rede é homogênea.
4. A topologia não muda.
5. A latência é zero.
6. A largura de banda é infinita.
7. O custo de transporte é zero.
8. Há só um administrador.

Observe como essas premissas se referem a propriedades exclusivas de sistemas distribuídos: confiabilidade, segurança, heterogeneidade e topologia da rede; latência e largura de banda; custos de transporte e, por fim, domínios administrativos. No desenvolvimento de aplicações não distribuídas, é provável que a maioria dessas questões nem apareça.

## 1.4 Sistemas de informação distribuídos

Encontrado em organizações que se defrontarem com uma profusão de aplicações em rede para as quais a interoperabilidade se mostrou uma experiência dolorosa. Muitas das soluções de middleware existentes são resultado do trabalho com uma infra-estrutura na qual era mais fácil integrar aplicações a um sistema de informações de âmbito empresarial (Bernstein, 1996; Alonso et al., 2004).

À medida que as aplicações se tornavam mais sofisticadas e eram gradualmente separadas em componentes independentes, notavelmente distinguindo componentes de banco de dados de componentes de processamento, ficou claro que a integração também deveria ocorrer de modo que permitisse às aplicações se comunicar diretamente umas com as outras. Isso resultou, atualmente, em uma enorme indústria dedicada à integração de aplicações empresariais (Enterprise Application Integration – EAI).

## 1.5 Sistemas distribuídos pervasivos

Caracterizados por sua estabilidade: os nós são fixos e têm uma conexão mais ou menos permanente e de alta qualidade com uma rede.

Um exemplo: a profusão de técnicas para mascarar falhas e recuperação dará a impressão de que as coisas podem dar errado apenas raramente. Da mesma maneira, conseguimos ocultar aspectos relacionados com a real localização de um nó na rede, o que permite, efetivamente, que usuários e aplicações acreditem que os nós continuam onde estão.

Contudo, a questão ficou muito diferente com a introdução de dispositivos de computação móveis e embutidos. Atualmente encontramos sistemas distribuídos nos quais a instabilidade é o comportamento esperado. Nesses sistemas, que denominamos sistemas distribuídos pervasivos, os equipamentos costumam ser caracterizados por seu pequeno tamanho, pela alimentação por bateria, por sua mobilidade e por terem somente uma conexão sem fio, se bem que nem todas essas características se aplicam a todos os dispositivos. Além do mais, tais características não precisam ser necessariamente interpretadas como restritivas, como é ilustrado pelos smart phones. (Roussos et al., 2005).

Como seu nome sugere, um sistema distribuído pervasivo é parte de nosso entorno; por isso, é, em geral, inerentemente distribuído. Um aspecto importante é a ausência geral de controle administrativo humano. Na melhor das hipóteses, os dispositivos podem ser configurados por seus proprietários; porém, quanto ao mais, eles precisam descobrir automaticamente seu ambiente e se encaixar o melhor que puderem. Grimm et al. (2004) tornaram esse encaixar mais exato pela formulação dos três requisitos para aplicações pervasivas apresentados a seguir:

1. Adotar mudanças contextuais;
2. Incentivar composição ad hoc;
3. Reconhecer compartilhamento como padrão;

## 1.6 Sistemas domésticos

Um tipo cada vez mais popular de sistema pervasivo, mas que talvez seja o menos restrito, são sistemas montados ao redor de redes domésticas. Em geral, esses sistemas são compostos de um ou mais computadores pessoais. Porém, o mai importante é que integram eletrônicos de consumo típicos como aparelhos de TV, equipamentos de áudio e vídeo, dispositivos par jogos, smart phones, PDAs e outros equipamentos de uso pessoal em um único sistema. Além disso, podemos esperar que todos os tipos de dispositivos, como eletrodomésticos de cozinha, câmaras de vigilância, relógios, controladores de iluminação e assim por diante, serão conectados a um único sistema distribuído.

Da perspectiva de sistema, há vários desafios que precisam ser enfrentados antes que os sistemas pervasivos domésticos se tornem realidade. Um desafio importante é que tal sistema deve ser completamente autoconfigurável e autogerenciável. Não se pode esperar que usuários finais estejam dispostos ou sejam capazes de manter um sistema distribuído doméstico ligado e em funcionamento se seus componentes forem propensos a erros, como acontece com muitos dos dispositivos existentes hoje.

Muito já foi conseguido por meio dos padrões Universal Plug and Play (UPnP), pelos quais dispositivos obtêm automaticamente endereços IP, podem descobrir uns aos outros e assim por diante.

## 1.7 Sistemas eletrônicos para tratamento de saúde

Uma outra classe de sistemas pervasivos importante e que está começando a fazer sucesso é a relacionada ao tratamento eletrônico (pessoal) de saúde. Com o aumento do custo do tratamento médico, estão sendo desenvolvidos novos dispositivos para monitorar o bem-estar de indivíduos e entrar automaticamente em contato com médicos quando necessário. Em muitos desses sistemas, uma meta importante é evitar que as pessoas sejam hospitalizadas.

Sistemas para tratamentos de saúde costumam ser equipados com vários sensores organizados em uma rede de área corporal, de preferência sem fio. Uma questão importante é que, na pior das hipóteses, tal rede deve incomodar uma pessoa o mínimo possível. Com essa finalidade em vista, a rede deve ser capaz de funcionar quando a pessoa estiver em movimento, sem que esta precise estar presa por fios elétricos a dispositivos imóveis.

Da perspectiva do sistema distribuído, deparamos imediatamente com questões como:

1. Onde e como os dados monitorados deverão ser armazenados?
2. Como podemos evitar a perda de dados cruciais?
3. Qual é a infra-estrutura necessária par gerar e transmitir sinais de alerta?
4. Como os médicos podem dar retorno on-line?
5. Como pode ser alcançada a extrema robustez do sistema de monitoração?
6. Quais são as questões de segurança e como as políticas adequadas podem ser impostas?

Diferentemente dos sistemas domésticos, não podemos esperar que a arquitetura de sistemas pervasivos de tratamento de saúde tenda a passar para sistemas de um único servidor e que seus dispositivos de monitoração operem com funcionalidade mínima Ao contrário: por razões de eficiência, os dispositivos e redes de áreas corporais terão de suportar processamento de dados na rede, o que significa que os dados de monitoração terão de ser agregados antes de ser armazenados permanentemente ou enviados a um médico. Diferentemente do caso de sistemas de informação distribuídos, ainda não uma resposta clara para essas questões.

## EXERCÍCIOS

1

Qual a definição de um Sistema Distribuído?

- A)
Conjunto de computadores dependentes, interconectados por uma rede de comunicação, que se apresenta a seus usuários como um sistema de imagem único.
- B)
Conjunto de computadores paralelos, interconectados por uma rede de comunicação, que se apresenta a seus usuários como um sistema de imagem único.
- C)
Conjunto de computadores independentes, interconectados por uma rede de comunicação, que se apresenta a seus usuários como um sistema de imagem independente para cada um dos seus participantes.
- D)
Conjunto de computadores distribuídos, interconectados por um barramento único de rede, que se apresenta a seus usuários como um sistema de imagem único.
- E)
Conjunto de computadores independentes, interconectados por uma rede de comunicação, que se apresenta a seus usuários como um sistema de imagem único.

2

Quais os motivos para se utilizar um Sistema Distribuído?

- A)
Compartilhamento de recursos, diminuição de velocidade de computação, confiabilidade, comunicação
- B)
Compartilhamento de recursos, diminuição de velocidade de computação, confiabiliade, e ausência de comunicação
- C)
Compartilhamento de recursos, diminuição de velocidade de computação, confiabiliade, agrupamento de redes.
- D)
Compartilhamento de recursos, aumento de velocidade de computação, confiabilidade, comunicação.
- E)
Compartilhamento de recursos, aumento de velocidade de computação, ausência de confiabilidade, comunicação.

3

Um sistema distribuído que é capaz de se apresentar a usuários e aplicações como se fosse apenas um único sistema de computador é denominado...

- A)
Homogêneo.
- B)
Transparente.
- C)
Heterogêneo.
- D)
Paralelo.
- E)
Ineficiente.

4

Em sistemas computacionais com múltiplos processadores os sistemas distribuídos se caracterizam por:

- A)
Formados por computadores independentes, que utilizam apenas CPU, memória RAM e um dispositivo para conexão de rede, que se comunicam através da troca de mensagens.
- B)
Formados por um computador de grande porte, com vários processadores, que compartilham a mesma memória.
- C)
Formados por computadores completos sobre uma rede de longa distância como a internet, permitindo a utilização de computadores espalhados pelo mundo para o processamento de uma aplicação.
- D)
Formados por um computador de grande porte, com vários processadores, onde cada processador possui memória independente.

- E)
  Formados por computadores comuns que compartilham a memória RAM. Formados por computadores completos sobre uma rede de longa distância como a internet, permitindo a utilização de computadores espalhados pelo mundo para o processamento de uma aplicação.

5

Para suportar computadores e redes heterogêneos e, simultaneamente, oferecer uma visão de sistema único, os sistemas distribuídos costumam ser organizados por meio de uma camada de software que é situada logicamente entre uma camada de nível mais alto, composta de usuários e aplicações, e uma camada subjacente, que consiste em sistemas operacionais e facilidades básicas de comunicação. Tal camada é denominada...

- A)
  Distribuído.
- B)
  Paralelo.
- C)
  Operacional.
- D)
  Middleware.
- E)
  Cluster.

6

Podemos afirmar sobre sistemas paralelos e sistemas distribuídos.

- A)
  Sistemas paralelos – vários processadores, mesmo barramento, sistema fortemente acoplado.
- B)
  Sistemas paralelos – Compartilhamento de memória, relógio, etc.
- C)
  Sistemas distribuídos – Não compartilham a mesma memória, etc
- D)
  Sistemas distribuídos – Processadores se comunicam através de várias linhas de comunicação, sistema fracamente acoplado.
- E)
  Todas as alternativas estão corretas.

7

O desenvolvimento de software é marcado pela mudança de paradigmas. Em um primeiro momento surgiu o modelo estruturado, a seguir o modelo orientado a objetos, o modelo orientado a aspectos também faz parte da lista, não vamos esquecer da computação em nuvens, agora a moda é o desenvolvimento orientado a serviços, mas se observamos ao nosso redor, a forma como as pessoas estão interagindo com seus celulares, tables, notebooks com sensibilidade ao toque da tela, será que nós desenvolvedores de software não estamos caminhando para uma nova forma de fazer software. A conectividade está impregnada ao nosso redor, pode estar no nosso carro, nos eletrônicos da nossa casa, quanto tempo será necessário para através do nosso celular possamos controlar tudo o que está ao nosso redor?. Recentemente um grupo de pesquisadores americanos apresentaram um experimento de transferência de pensamentos entre duas pessoas a certa distância. Será que o conceito de sistemas distribuídos tem alguma relação com as tendências relatadas pelo texto?
Qual a alternativa correta?

- A)
  O conceito de sistemas distribuídos trata de um ambiente onde computadores dependentes que apresentem á seus usuários um sistema único e coerente. Só se percebe a existência de um sistema distribuído quando a falha de um computador do qual você nunca ouviu falar faz com que o seu trabalho seja interrompido. Quando tratamos do tema conectividade para o usuário não importa a forma como se esta interagindo com os diversos dispositivos. No caso dos sistemas distribuídos móvel e pervasivos, a instabilidade é o comportamento esperado, bem como a heterogeneidade de rede e dispositivos, a segurança também é um problema grave.
- B)
  O conceito de sistemas distribuídos trata de um ambiente onde computadores independentes apresentem á seus usuários um sistema único e coerente. Só se percebe a existência de um sistema distribuído quando a falha de um computador do qual você nunca ouviu falar faz com que o seu trabalho seja interrompido. Quando tratamos do tema conectividade para o usuário não importa a forma como se esta interagindo com os diversos dispositivos. No caso dos sistemas distribuídos móvel e pervasivos, a instabilidade é o comportamento esperado, bem como a heterogeneidade de rede e dispositivos, a segurança também é um problema grave.

- C)
  O conceito de sistemas distribuídos trata de um ambiente onde computadores dependentes que apresentem á seus usuários um sistema multiplo e coerente. Só se percebe a existência de um sistema distribuído quando a falha de um computador do qual você nunca ouviu falar faz com que o seu trabalho seja interrompido. Quando tratamos do tema conectividade para o usuário importa a forma como se esta interagindo com os diversos dispositivos. No caso dos sistemas distribuídos móvel e pervasivos, a instabilidade é o comportamento esperado, bem como a heterogeneidade de rede e dispositivos, a segurança também é um problema grave.
- D)
  O conceito de sistemas distribuídos trata de um ambiente onde computadores independentes que apresentem á seus usuários um sistema único e coerente. Só se percebe a existência de um sistema distribuído quando a falha de um computador do qual você nunca ouviu falar faz com que o seu trabalho seja interrompido. Quando tratamos do tema conectividade para o usuário importa a forma como se esta interagindo com os diversos dispositivos. No caso dos sistemas distribuídos móvel e pervasivos, a estabilidade é o comportamento esperado, bem como a homogeneidade de rede e dispositivos, a segurança também é um problema grave.
- E)
  O conceito de sistemas distribuídos trata de um ambiente onde computadores independentes que apresentem á seus usuários um sistema transparente onde o número de participantes é bem definido. Só se percebe a existência de um sistema distribuído quando a falha de um computador do qual você nunca ouviu falar faz com que o seu trabalho seja interrompido. Quando tratamos do tema conectividade para o usuário não importa a forma como se esta interagindo com os diversos dispositivos. No caso dos sistemas distribuídos móvel e pervasivos, a instabilidade é o comportamento esperado, bem como a heterogeneidade de rede e dispositivos, a segurança também é um problema grave.

---

1E 2D 3B 4E 5D 6E 7B

# Conteúdo 2 - Arquiteturas em Sistemas Distribuídos

**Arquiteturas em Sistemas Distribuídos**

**Introdução**

Sistemas distribuídos podem ser organizados de diferentes modos. Podemos fazer uma distinção entre arquitetura de software e arquitetura de sistema. A última considera onde os componentes que constituem um sistema distribuído estão colocados nas várias máquinas. A primeira se preocupa mais com a organização lógica do software: como os componentes interagem, de que modos eles podem ser estruturados, como podem ficar independendentes e assim por diante.

Uma idéia fundamental quando falamos sobre arquiteturas é o estilo arquitetônico. Um estilo reflete o princípio básico que é sugerido na organização da interação entre os componentes de software que compreendem um sistema distribuído. Entre os estilos importantes estão disposição em camadas, orientação a objetos, orientação a eventos e orientação a espaço de dados.

Há variadas organizações de sistemas distribuídos. Uma classe importante é aquela em que as máquinas são divididas em clientes e servidores. Um cliente envia uma requisição a um servidor, que então produzirá um resultado que é retornado ao cliente. A arquitetura cliente-servidor reflete o modo tradicional de modularização de software pelo qual um módulo chama as funções disponíveis em um outro módulo. Colocando componentes diferentes, obtemos uma distribuição física natural de funções por um conjunto de máquinas.

Arquiteturas cliente-servidor costumam apresentar alto grau de centralização. Em arquiteturas descentralizadas, freqüentemente vemos um papel igual desempenhado pelos processos que constituem um sistema distribuído, também conhecidos como sistemas peer-to-peer. Em sistemas peer-to-peer, os processos são organizados em uma rede de sobreposição, que é uma rede lógica na qual todo processo tem uma lista local de outros pares com os quais ele pode se comunicar. A rede de sobreposição pode ser estruturada, caso em que são disponibilizados esquemas determinísticos para rotear mensagens entre processos. Em redes não estruturadas, a lista de pares é aleatória, o que implica que é preciso disponibilizar algoritmos de busca para localizar dados os outros processos.

**Estilos**

Os mais importantes para sistemas distribuídos:

1. Arquiteturas em camadas
2. Arquiteturas baseadas em objetos
3. Arquiteturas centradas em dados
4. Arquiteturas baseadas em eventos

As arquiteturas multidivididas sugerem várias possibilidades para a distribuição física de uma aplicação cliente-servidor por várias máquinas. A organização mais simples é ter só dois tipos de máquinas:

1. Uma máquina cliente que contém apenas os programas que implementam o nível de interface de usuário.
2. Uma máquina servidor que contém o resto, ou seja, os programas que implementam o nível de processamento de dados.

Nesta organização, tudo é manipulado pelo servidor, ao passo que, em essência, o cliente nada mais é do que um terminal burro, possivelmente com uma interface gráfica bonitinha. Uma abordagem para organizar clientes e servidores é distribuir os programas presentes nas camadas de aplicação.

**Sistemas Distribuídos Colaborativos**

Estruturas híbridas são disponibilizadas notavelmente em sistemas distribuídos colaborativos. A questão principal em muitos desses sistemas é conseguir dar a partida, para o que muitas vezes é disponibilizado um esquema cliente-servidor tradicional. Tão logo um nó se junte ao sistema, ele pode usar um esquema totalmente descentralizado para colaboração.

Para ficarmos no terreno concreto, em primeiro lugar vamos considerar o sistema de compartilhamento de arquivos BitTorrent. O BitTorrent é um sistema peer-to-peer de transferência (downloads) de arquivos. A idéia básica é que, quando um usuário final estiver procurando um arquivo, ele transfira porções do arquivo de outros usuários até que as porções transferidas possam ser montadas em conjunto, resultando no arquivo completo.

## EXERCÍCIOS

1

O que define a arquitetura de um sistema distribuído?
I - As relações entre estes componentes;
II - A divisão de responsabilidades (ex. cliente, servidor);
III - A localização dos componentes do sistema;
IV - Como estas componentes são mapeados para a infra-estrutura de rede

- A)
  Somente as afirmativas I e II são corretas.
- B)
  Somente as afirmativas I e IV são corretas.
- C)
  Somente as afirmativas III e IV são corretas.
- D)
  Somente as afirmativas I, II e III são corretas.
- E)
  Todas estão são corretas.

2

Com o passar dos anos, as aplicações empresariais evoluíram de aplicações monolíticas executando em computadores de grande porte, para aplicações do tipo cliente-servidor e, então, para aplicações em camadas. No caso de aplicação em três camadas, quais os nomes destas camadas?

- A)
  Visualização, lógica e negócio.
- B)
  Cliente, servidor de aplicação, servidor de banco de dados.
- C)
  Linguagens de programação, middleware e SGBD.
- D)
  Domínio, processos e servidor de banco de dados.
- E)
  Regras de negócio, visualização e Informação.

3

As arquiteturas multidivididas sugerem várias possibilidades para a distribuição física por várias máquinas; a organização mais simples é ter só dois tipos de máquinas. Como chamamos essas duas máquinas?

- A)
  Cliente e servidor.
- B)
  Cliente e serviço.
- C)
  Usuário e servidor.
- D)
  Usuário e serviço.
- E)
  Cliente e centralizador.

4

Quais são as principais caracteristicas das arquiteturas Peer-to-Peer?
I - Existem uma distinção clara entre cliente e servidor;
II - Um peer pode ser ora cliente, ora servidor;
III - Os hosts do SD atuam cooperativamente;
IV - Garante a robustez no caso de falhas devido a replicação dos dados através de múltiplos pares.

- A)
  Somente a I esta correta.
- B)
  Somente I, II e III estão corretas.
- C)
  Somente I, II e IV estão corretas.
- D)
  Somente II, III e IV estão corretas.

- E)
  Todas estão corretas.

5

Quais das opções abaixo representam tipo de arquiteturas em Sistemas Distribuídos?
I - Arquitetura em camadas;
II - Arquitetura baseada em objetos;
III - Arquitetura centrada em dados;
IV - Arquitetura baseada em publicação.

- A)
  Somente I e IV estão corretas.
- B)
  Somente I, II e IV estão corretas.
- C)
  Somente I, II e III estão corretas.
- D)
  Somente II, III e IV estão corretas.
- E)
  Todas estão corretas.

1E 2B 3A 4A 5C

# Conteúdo 3 - Processos em Sistemas Distribuídos

3. Processos

3.1 Introdução

Processos desempenham um papel fundamental em sistemas distribuídos porque formam uma base para comunicação entre máquinas diferentes. Uma questão importante é como os processos são organizados internamente e, em particular, se suportam ou não vários threads de controle. Threads em sistemas distribuídos são particularmente úteis para continuar usando a CPU quando é realizada uma operação bloqueadora de E/S. Desse modo, torna-se possível construir servidores de alta eficiência que executam vários threads em paralelo, entre os quais diversos podem estar bloqueados à espera da conclusão de E/S de disco ou de comunicação de rede.

Outra questão importante, em especial em sistemas distribuídos de longa distância, é movimentar processos entre máquinas diferentes. Migração de processo, ou, mais especificamente, migração de código, pode ajudar a conseguir escalabilidade, mas também pode ajudar a configurar dinamicamente clientes e servidores.

3.2 Threads

Embora processos formem um bloco de construção em sistemas distribuídos, a prática indica que a granularidade de processos proporcionada pelos sistemas operacionais sobre os quais os sistemas distribuídos são construídos não é suficiente. Em vez disso, observa-se que ter granularidade mais fina sob a forma de múltiplos threads de controle por processo facilita muito a construção de aplicações distribuídas e a obtenção de melhor desempenho. Em particular, um contexto de thread freqüentemente consiste em nada mais que o contexto da CPU, junto com algumas outras informações para gerenciamento de threads. Por exemplo, um sistema de threads pode monitorar o fato de um thread estar bloqueado em uma variável de mútua exclusão em dado instante, de modo a não selecioná-lo para execução.

As threads em sistemas distribuídos pode proporcionar um meio conveniente de permitir chamadas bloqueadoras de sistema sem bloquear o processo inteiro no qual o thread está executando. Essa propriedade torna os threads particularmente atraentes para utilização em sistemas distribuídos, uma vez que facilitam muito expressar comunicação na forma de manter múltiplas conexões lógicas ao mesmo tempo.

## EXERCÍCIOS

1

Uma questão importante, em especial em sistemas distribuídos de longa distância, é movimentar processos entre máquinas diferentes. Migração de processo, ou, mais especificamente migração de código, pode ajudar a conseguir escalabilidade, mas também pode ajudar a configurar dinamicamente clientes e servidores. Embora processos formem um bloco de construção em sistemas distribuídos, a prática indica que a granularidade de processos proporcionada pelos sistemas operacionais sobre os quais os sistemas distribuídos são construídos não é suficiente. Em vez disso, observa-se que ter granularidade mais fina sob a forma de múltiplos threads de controle por processo facilita muito a construção de aplicações distribuídas e a obtenção de melhor desempenho. Por que a utilização de threads em sistemas distribuídos pode ser interessante?

- A)
Porque podem proporcionar um meio conveniente de permitir chamadas bloqueadoras de sistema sem bloquear o processo inteiro no qual a thread está executando.
- B)
Porque evitam proporcionar um meio conveniente de permitir chamadas bloqueadoras de sistema bloqueando o processo inteiro no qual a thread está executando.
- C)
Porque podem garantir o assincronismo entre os sistemas.
- D)
Porque não conseguem manter múltiplas conexões lógicas ao mesmo tempo.
- E)
Porque uma thread fica restrita apenas ao cliente da aplicação distribuída.

2

COMANDO DA AERONÁUTICA
EXAME DE ADMISSÃO AO ESTÁGIO DE ADAPTAÇÃO DE FICIAIS TEMPORÁRIOS (EAOT 2009)
ESPECIALIDADE: ANÁLISE DE SISTEMAS (ANS)
Suponha que aplicações, usando base de dados, exigem acesso exclusivo a certa tabela e, para ganhar acesso exclusivo, pedem um travamento desta tabela. Se as uas aplicações clientes tentarem travar a mesma tabela ao mesmo tempo, então nem uma nem a outra aplicação receberá o acesso e a trava. Uma vez que não há meio geral para decidir para quem dar a trava, ambos os clientes esperarão eternamente pela mesma.” Em informática, assinale o nome dado para o resultado da suposição:

- A)
Threads.
- B)
Deadlock.
- C)
Grafos.
- D)
Rollback.
- E)
Colisão.

3

COMANDO DA AERONÁUTICA
EXAME DE ADMISSÃO AO ESTÁGIO DE ADAPTAÇÃO DE FICIAIS TEMPORÁRIOS (EAOT 2009)
ESPECIALIDADE: ANÁLISE DE SISTEMAS (ANS)
Sobre Thread, assinale a afirmativa correta:

- A)
Forma de um processo de dividir a si mesmo em duas ou mais tarefas que podem ser executadas simultaneamente.
- B)
Um impasse onde dois ou mais processos ficam impedidos de continuar suas execuções.
- C)
Qualquer processamento ou armazenamento em excesso.
- D)
Guarda todas as informações, mesmo quando não está recebendo energia.
- E)
Funciona como um semáforo onde ora esta verde para um processo, ora esta vermelho para o outro

4

POSCOMP 2011

O gerenciamento de processos em sistemas modernos é feito, quase sempre, com o uso de preempção de processos através de técnicas de compartilhamento de tempo; o que a introdução de processadores com vários núcleos altera nesse gerenciamento?

- A)
  Torna-se possível a paralelização efetiva de processos concorrentes.
- B)
  Torna-se possível eliminar a condição de corrida em processos concorrentes executados em paralelo.
- C)
  Torna-se possível o uso de threads para a execução de processos concorrentes.
- D)
  Torna-se possível separar os demais mecanismos de gerenciamento do sistema operacional do gerenciamento de processos.
- E)
  Torna-se possível o uso de sistemas operacionais multitarefas.

5

POSCOMP 2008

Analise as seguintes afirmativas.

I. Uma arquitetura multithreading executa simultaneamente o código de diversos fluxos de instruções (threads).

II. Em uma arquitetura VLIW, o controle da execução das várias instruções por ciclo de máquina é feito pelo compilador.

III. Uma arquitetura superescalar depende de uma boa taxa de acerto do mecanismo de predição de desvio para obter um bom desempenho.

IV. Os processadores vetoriais são um tipo de arquitetura SIMD.

V. Um processador dual-core é mais eficiente em termos de consumo de energia do que dois processadores single-core de mesma tecnologia.

A partir da análise, pode-se concluir que:

- A)
  Apenas a afirmativa IV está correta.
- B)
  Apenas as afirmativas III e IV estão corretas.
- C)
  Apenas as afirmativas I, IV e V estão corretas.
- D)
  Apenas as afirmativas I, III e V estão corretas.
- E)
  Todas as afirmativas estão corretas.

6

POSCOMP 2007

Considerando as diferenças existentes entre a execução de um algoritmo sequencial e a execução de um algoritmo distribuído, analise as seguintes afirmativas.

I. Somente na execução sequencial de um algoritmo existe a possibilidade de ocorrer um deadlock.

II. Um algoritmo sequencial apresenta mais de uma execução possível para uma dada entrada.

III. Um algoritmo distribuído tem sua complexidade medida pela quantidade de mensagens transmitidas durante sua execução.

IV. A execução de um algoritmo distribuído pode ser não determinística.

A análise permite concluir que

- A)
  Todas as afirmativas são falsas.
- B)
  Todas as afirmativas são verdadeiras.
- C)
  Apenas as afirmativas I e II são verdadeiras.
- D)
  Apenas as afirmativas I e IV são verdadeiras.
- E)
  Apenas a afirmativa IV é verdadeira.

---

1A 2B 3A 4A 5A 6E

# Conteúdo 4 - Comunicação em Sistemas Distribuídos

**Comunicação em Sistemas Distribuídos**

**Troca de mensagens**

As mensagens são objetos de dados cuja estrutura e aplicação são definidas pelas próprias aplicações que a usarão. Sendo a troca de mensagens feita através de primitivas explicitas de comunicação:
send(destino, mensagem) envio da mensagem para o destino
receive(origem, mensagem) recebimento da mensagem enviada pela origem
As primitivas acima podem ser classificadas do seguinte modo:
a) Forma de comunicação
a.1) Direta
send: há indicação do processo receptor.
send(process, msg)
receive: há indicação do emissor.
receive(process, msg)

a.2) Indireta
send: envio para uma porta ou mailbox sem o conhecimento de qual será o receptor.
send(mailbox, msg)
receive: obtenção da mensagem guardada no mailbox, possivelmente desconhecendo a identidade do processo emissor.
receive(mailbox, msg)

b) Forma de Sincronização
b.1) Síncrono ou Bloqueante
Send: espera até que a mensagem seja recebida pelo receptor.
Receive: aguarda até a mensagem estar disponível

c) Assíncrona ou Não Bloqueante
Send: envia a mensagem mas não espera até que a mensagem seja recebida pelo receptor.
Receive: se a meensagem estiver disponível, recebe a mensagem, caso contrário continua o processamento retornando uma indicação de que a mensagem não estava disponível.

Um sistema pode possuir diferentes combinações desses tipos de primitivas, sendo que um sistema de comunicação passa a ser conhecido como síncrono se ambas as primitivas (send, receive), forem do tipo bloqueante. Por outro lado, um sistema de comunicação é dito assíncrono se pelo menos uma das primitivas for assíncrona. A principal desvantagem dessa abordagem é o baixo nível de abstração que permite apenas a modelagem de troca indireta de informação entre processos.

**Modelo Cliente/Servidor**

É um paradigma de programação que representa as interações entre os processos e as estruturas do sistema. Neste tipo de paradigma existem dois de processos:
clientes: processos que requisitam serviços;
servidores: processos que recebem requisitos , realizam uma operação e retornam serviços

Em um modelo cliente/servidor, o processo cliente necessita de um serviço (ex. Ler dados de um arquivo), então ele envia uma mensagem para o servidor e espera pela mensagem de resposta. O processo servidor, após realizar a tarefa requisitada, envia o resultado na forma de uma mensagem de resposta ao processo cliente. Note que os servidores, em um sistema deste tipo, apenas responde as requisições dos clientes,e , tipicamente, não iniciam conversação com os clientes. A principal vantagem desta abordagem é a simplicidade, mas a implementação é menos eficiente do que a troca de mensagens pura.

Atenção:
- A visão lógica da comunicação cliente/servidor possibilita a troca de mensagens
- Possibilidade de vários clientes acessarem um recurso de forma consistente e transparente através do uso de um único servidor. Mas também é possível termos ambientes com múltiplos servidores.

No modelo cliente servidor é possível a existência de vários servidores, onde os mesmos podem ser replicados, isto é, quando há várias instâncias do mesmo servidor. A replicação muitas vez ocorre por questões de desempenho, distribuição natural dos recursos ou por tolerância a falhas. Também é possível termos servidores hierárquicos, isto é, servidores que usam o(s) serviço(s) de outros servidores. Nestes casos, as localizações e conversões entre servidores deverá ser transparentes aos clientes.

**Endereçamento**
Para que um cliente envie mensagens a um servidor, primeiro o cliente, deve necessariamente conhecer o endereço do servidor, desse modo, é preciso estabelecer um esquema de identificação:
1. um identificador único de processo quando na mesma máquina: com um único processo por máquina basta indicar o endereço da máquina pois o kernel consegue determinar qual é o processo servidor único. Esta não é abordagem viável, pois dificilmente teremos um único processo servidor em uma máquina.
2. Endereçamento indicando o processo e a máquina: quando se permite mais de um processo servidor por máquina, deve−se endereçar a mensagem a esse servidor especifico. Um esquema comum é o uso de um nome composto por duas partes. Ex: 12,4 ou , sendo que 12 é a máquina e quatro é o processo. Logo, o kernel da máquina cliente usa o número 12 para enviar ao máquina correspondente e com o número 4 e o Kernel remoto determina para qual servidor a mensagem é endereçada. Nessa abordagem a identificação é incluída no código do cliente. Com isso evita−se o custo de coordenação global e evita−se ambiguidades entre processos com identificador idênticos mas em máquinas distintas. Porém, perde−se em transparência pois é preciso que o cliente conheça a localização física do servidor. Isso pode causar problemas, como por exemplo: se um servidor de banco de dados normalmente executa na maquina 12, no momento em que a mesma for desligada para manutenção, pode−se disponibilizar o mesmo serviço em outra máquina. Ao usar este esquema com maquinas fixas, o serviço poderá estar disponível em outra máquina, mas não poderá ser usado.
3. Processos escolhem endereços que são detectados por broadcast: Cada processo deve receber um endereço único que não envolva o número da sua máquina. Este endereço pode ser atribuído de duas formas:
- Através de um escalonador de endereços de processo centralizado, que mantém um contador e ao receber uma requisição incrementa esse contador e envia o valor. A principal desvantagem é a utilização de um componente centralizado que compromete a tolerância a falhas e a escalabilidade;
- Cada processo escolhe um valor aleatoriamente a partir de um espaço de endereçamento grande (ex. 64 bits) e portanto com pouca probabilidade de colisão. O kernel emissor de uma requisição pode localizar para qual maquina enviar através do seguinte procedimento:
1) o emissor envia a mensagem para todas as maquinas (broadcast) com um pacote especial de localização contendo o endereço do processo destino;
2) em cada maquina da rede o kernel verifica se o processo está na máquina. Quem localizar envia mensagem indicando seu endereço na rede;
3) o kernel emissor guarda essas informações para requisições futuras.
Essa abordagem tem como vantagem ser transparente mas tem um custo da mensagem broadcast.
4. uso de servidor de nomes: neste esquema os servidores são indicados por um identificador de alto nível (nome ASCII) que não inclui nem identificação da maquina nem do processo. Quando um cliente executa uma requisição a um servidor pela primeira vez, uma mensagem especial é enviada para um servidor de mapeamento ou servidor de nomes solicitando o número da máquina onde está o servidor.

**Primitivas confiáveis e não confiáveis de comunicação**
Se as mensagens envolvidas na comunicação cliente/servidor for assumida como não confiáveis, recairá ao usuário a tarefa de controlar possíveis perdas de mensagens. Isso não é desejável uma vez que o objetivo é tornar o mais transparente possível para o usuário. Logo, é desejável que o kernel do S.O. Se preocupe em verificar se as mensagens forem corretamente recebidas garantindo dessa forma uma comunicação confiável. A figura abaixo ilustra dois protocolos para garantir a confiabilidade através do uso de mensagens especiais do tipo acknowledgment.

**Chamada Remota de Procedimento (RPC − Remote Procedure Call)**
Chamada remota de procedimento permite que programas invoquem procedimentos ou funções localizadas em outras máquinas como se ele estivesse localmente definidos. A nível do programa, as informações são passadas do chamador para o procedimento chamado através de parâmetros e resultados são retornados através do resultado do procedimento. Deste modo, nenhuma mensagem ou entrada/saída é visível ao programador. Apesar de simples há alguns problemas nos seguintes pontos:
- Chamador e chamado executam em espaços de endereçamento diferentes;
- Como fazer a passagem de parâmetros e resultados quando as maquinas envolvidas têm arquiteturas distintas;
- Possibilidades de falhas.

**Operação RPC Básica**
A chamada de procedimento remoto deve parecer o máximo com o máximo possível com a chamada local de procedimentos. Por isso, utiliza−se a estrutura de cliente/servidor stubs. A RPC segue os seguintes passos:
1. O procedimento chama o stub cliente de forma normal, isto é, como se fosse um procedimento local qualquer;
2. O stub cliente constrói mensagens e passa ao kernel;
3. O kernel remoto envia a mensagem ao stub servidor;
4. O kernel remoto entrega a mensagem ao stub servidor;
5. O stub servidor desempacota os parâmetros e chama o servidor;
6. O servidor realiza o trabalho solicitante e envia o resultado ao stub servidor;

7. O stub servidor empacota o resultado em uma mensagem e passa para o kernel;
8. O kernel remoto envia mensagem ao kernel cliente;
9. O kernel cliente entrega a mensagem ao stub cliente;
10.O stub desempacota o resultado e retorna ao stub cliente.
O efeito da execução de todos esses passos é a conversão da chamada local de um cliente a um stub cliente em uma chamada ao procedimento servidor sem que o cliente ou servidor percebam os passos intermediários. Via de regra os stubs são gerados automaticamente por um compilador.

a) Passagem de Parâmetros
É preciso tratar a passagem de parâmetros e a conversão de dados. Isso é feito pela operação de “empacotamento de parâmetros” (parameter marshalling). Essa operação trata problemas de representação distinta de dados (números/caracteres)para máquinas hetergêneas. tanto o cliente quanto o servidor sabem os tipos de parâmetros passados (identificador do procedimento + parâmetros) e devem conseguir obter a representação correta dos dados.
Opção 1: uso de um padrão de representação de dados, como por exemplo o XDR do RPC UNIX. Essa alternativa é muito interessante por permitir a comunicação entre maquinas heterogêneas.
A desvantagem é que entre máquinas homogêneas não há necessidade de incluir um custo adicional de conversão da representação da máquina para a representação padrão.
Opção 2: a mensagem é enviada no formato nativo com indicação de qual o formato utilizado. O receptor deve fazer a conversão quando for o caso. Essa opção evita custos de conversão em ambientes homogêneos, entretanto exige que clientes e servidores saibam converter qualquer formato para a sua representação nativa.

b) Passagem de Ponteiros
Um ponteiro representa um endereço de memória que não tem nenhum significado na máquina remota.
Opção 1: proibir a passagem de ponteiros, essa obviamente não é uma alternativa desejável pelo programador;
Opção 2: Copiar o valor apontado pela variável ponteiros. As alterações são feitas remotamente e no retorno as alterações são atualizadas no chamador (semântica cópia/restauração).
Uma otimização possível consite em, sabendo que um determinado parâmetro é apenas de entrada, evita o retorno do conteúdo do ponteiro. De forma análoga, se ele for apenas de saída, ele não precisa ser copiado no envio. Esse tipo de informação pode ser fornecido pelo programador ou extraído a partir da análise do código fonte(análise estática).
c) Semântica RPC na Presença de Falhas
Na execução de uma chamda remota de procedimentos, existem cinco classes diferentes de falhas possíveis.
- O cliente não consegue localizar o servidor;
- Mensagem de requisição perdida;
- Mensagem de resposta perdida;
- O servidor cai após receber uma requisição;
- O cliente falha após enviar uma requisição (falha do cliente).

**Questões de Implementação**
As questões de implementação analisadas nesta seção estão relacionadas as questões de desempenho.
a) Protocolo
Em principio qualquer mecanismo de troca de mensagem poderia dar suporte a implementação de RPC. Uma das questões a serem decididas na implementação é se o protocolo será orientado a conexão ou sem conexão. No primeiro caso, evita−se o problema de mensagens perdidas, pois isso é tratado em baixo nível, porém, o desempenho é menor. No segundo caso, o desempenho é melhor, sendo indicado para implementação em LANs que não são confiáveis do que WANs.
b)Confirmação (Acknowledgment)
As mensagens de confirmação são usadas para detecção de falhas. No momento de implementar, é preciso decidir se os pacotes serão confirmados individualmente ou não. Assim temos dois tipos de protocolos: stop−and−wait (no momento em que um pacote for danificado ou perdido, o cliente não recebendo o ACK providência o reenvio), e o blast (o servidor envia mensagem ACK apenas no final). No caso do protocolo Blast, há duas formas de tratar o fato de apenas alguns pacotes terem sido entregues e outros não. Uma hipótese é ignorar todos os pacotes e solicitar a retransmissão de todos os pacotes. Outra possibilidade é realizar uma repetição seletiva (selective repeat) não compensa devido ao custo de controle envolvido. Outro problema refere- se ao fato dos buffers de recepção serem limitados. O envio de várias mensagens consecutivas pode causar um overrum error. O overrum error caracteriza−se pela incapacidade do receptor de receber um pacote que chega corretamente causando dessa forma uma perda de mensagem. Na prática esse é um erro mais sério do que a perda por ruído ou outros problemas físicos na rede. Com o uso de stop−and−wait em princípio esse erro não ocorre pois uma mensagem é enviada de cada vez. No entanto, a mesma, poderia ocorrer caso diferentes emissores enviarem várias mensagens ao mesmo tempo.

**Comunicação de Grupo**

Há vária formas aplicações para as quais tem−se comunicação com múltiplos processadores.

Ex.: grupos de servidores de arquivos que fornecem serviço de arquivos tolerantes a falhas. Um processo pode enviar mensagens para um grupo de servidores sem saber quantos eles são ou onde localizam−se, sendo que tais parâmetros podem mudar na próxima chamada. Um grupo é uma coleção de processos que agem juntos. Quanto uma mensagem é enviada ao grupo, todos os membros recebem a mesma mensagem. Grupos são dinâmicos: podem ser criados e destruídos, processos podem entrar ou abandonar um grupo; e um mesmo grupo pode pertencer  a mais de um grupo.

## EXERCÍCIOS

1

Dispor de facilidades poderosas e flexíveis para comunicação entre processos é essencial para qualquer sistema distribuído. Em aplicações tradicionais de rede, a comunicação costuma ser baseada nas primitivas de troca de mensagens de baixo nível oferecidas pela camada de transporte. Uma questão importante em sistemas middleware é oferecer um nível mais alto de abstração que facilitará expressar comunicação entre processos mais do que o suporte oferecido pela interface com a camada de transporte. Uma das abstrações mais amplamente utilizadas é a chamada de procedimento remoto (RPC). A essência de um RPC é que um serviço é implementado por meio de um procedimento cujo corpo é executado por um:

- A)
  Cliente.
- B)
  Servidor.
- C)
  Cluster.
- D)
  Grid.
- E)
  Banco de Dados.

2

POSCOMP 2006

A comunicação entre processos em um sistema distribuído pode ser realizada por um mecanismo conhecido como RPC - chamada de procedimento remoto. Sobre este mecanismo, assinale a opção correta abaixo:

- A)
  Processos comunicantes compartilham o mesmo espaço de endereçamento.
- B)
  Os stubs cliente e servidor são responsáveis pela conversão de formato dos parâmetros de entrada e saída, caso haja necessidade.
- C)
  A geração dos stubs é comumente realizada por compilação a partir de uma especificação de interface realizada em uma linguagem de execução de interface (IEL).
- D)
  O mecanismo faz uso de uma porta fixa, de número 8080, para comunicar diferentes processos e serviços entre computadores de um sistema distribuído.
- E)
  A falha de um cliente RPC gera uma chamada dita orfã no servidor que neste caso repassa sempre os resultados do procedimento remoto para um proxy de retorno especificado na chamada

3

POSCOMP 2006

Um sistema distribuído pode manter diferentes cópias de um mesmo item de dado a fim de melhorar o desempenho de leitura e aumentar a disponibilidade de acesso. A modificação deste item de dado é realizada de acordo com protocolos de consistência de cópias. Assinale a alternativa correta sobre esses protocolos.

- A)
  O protocolo baseado em cópia primária permite sempre a atualização da cópia mais próxima e difunde o novo valor via unicast para todos os nós que mantém uma outra cópia.
- B)
  A atualização de todas as cópias, no protocolo baseado em cópia primária, é realizada através de um processo síncrono, onde o cliente é liberado para continuar o fluxo de execução imediatamente após ter solicitado a atualização da cópia primária.
- C)
  Nos protocolos baseados em quorum, os conflitos leitura-escrita e escrita-escrita são evitados por autorizações de bloqueio (lock) emitidas por um coordenador central ou sequenciador.
- D)
  Protocolos baseados em coerência de cache são mecanismos de consistência de cópias que repassam a responsabilidade de manter essa consistência para os servidores que detém cópias.
- E)
  No protocolo de replicação ativa, todas as réplicas são atualizadas através de uma única operação de escrita realizada por um mecanismo de multicast totalmente ordenado.

4

POSCOMP 2003

Algoritmos distribuídos podem usar passagem de "token" por um anel lógico para implementar exclusão mútua ou ordenação global de mensagens. Nesses algoritmos, apenas o processo que possui o "token" tem a permissão de usar um recurso compartilhado ou numerar mensagens, por exemplo. Considerando o conceito acima podemos afirmar que:

- A)
  A abordagem deve tratar no mínimo dois tipos de defeitos: perda do "token" e colapso de processos.
- B)
  Para usar essa a abordagem os computadores precisam estar conectados em uma rede com topologia em anel.
- C)
  Nessa abordagem é impossível evitar a geração espontânea de vários "tokens" mesmo em sistemas livre de falhas.
- D)
  A abordagem é adequada apenas para sistemas onde possa ser controlado o tempo que cada computador permanece com o "token".
- E)
  A abordagem é pouco robusta pois a perda do "token" por um processo provoca o bloqueio do algoritmo distribuído que a usa.

5

Para que um cliente envie mensagens a um servidor, primeiro o cliente, deve necessariamente conhecer o endereço do servidor, desse modo, é preciso estabelecer um esquema de identificação. Quais são esses mecanismos?

I - Identificador único de processo quando na mesma máquina;
II - Endereçamento indicando o processo e a máquina;
III - Processos escolhem endereços que são detectados por broadcast
IV - Uso de DNS.

- A)
  Somente a I esta correta.
- B)
  Somente I e II estão corretas.
- C)
  Somente I, II e III estão corretas.
- D)
  Todas estão corretas.
- E)
  Nenhuma esta correta.

---

1B 2B 3E 4A 5A

# Conteúdo 5 - Tolerância a Falhas em Sistemas Distribuídos

## Tolerância a Falhas em Sistemas Distribuídos

Sistemas distribuídos são construídos por vários nodos processadores independentes. Esses sistemas se diferenciam de computadores paralelos pelo acoplamento fraco entre os nodos, ou seja, os elementos de um sistema distribuído não tem acesso a uma memória comum. Toda a interação deve ser feita por troca de mensagens através de canais de comunicação. Nodos de um sistema distribuído também não tem acesso a um relógio global, portanto não possuem uma base de tempo comum para ordenação de eventos. Além disso, sistemas distribuídos são geralmente construídos com elementos não homogêneos e assíncronos. Sistemas distribuídos não são sinônimo de redes de computadores. Uma rede de computadores pode fornecer a infraestrutura computacional para um sistema distribuído, mas nem toda aplicação de rede é necessariamente distribuída. Por exemplo, uma rede onde cada servidor roda uma aplicação e os demais nodos são clientes dessa aplicação não apresenta os problemas de consistência de dados e sincronização típicos de problemas distribuídos. Por outro lado, um cluster de alto desempenho com centenas e até milhares de nodos pode ser considerado um sistema distribuído, sem necessariamente ser suportado por um rede. Sistemas distribuídos apresentam um redundância natural, extremamente proveitosa para o emprego de técnicas de tolerância a falhas. Defeito em um nodo processador ou na rede de comunicação não precisa provocar necessariamente a queda de todo o sistema, e o sistema pode ser reconfigurado usando apenas os nodos disponíveis. A área de tolerância a falhas em sistemas distribuídos é vasta e excitante. Garantir dependabilidade envolve solucionar problemas de consenso, ordenação e atomicidade na troca de mensagens entre grupos de processos, sincronizar relógios quando necessário, implementar réplicas consistentes de objetos, garantir resiliência de dados e processos num ambiente sujeito a quedas de estações tanto clientes como servidoras, particionamento de redes, perda e atrasos de mensagens e, eventualmente, comportamento arbitrário dos componentes do sistema. Leitores interessados no assunto podem encontrar maiores detalhes nos livros do Jalote (Jal94), Birman (Bir96) e Mullende (Mul93).

### Motivação para tolerância a falhas em sistemas distribuídos

Um sistema distribuído deve prover operação continuada, com apenas pequena queda de desempenho, mesmo na presença de qualquer tipo de falha. Apesar de se conhecer um bom conjunto de técnicas de tolerância a falhas, sua aplicação a sistemas distribuídos, principalmente os de tempo-real, ainda é muito recente e seus resultados são muitas vezes insatisfatórios. Sistemas distribuídos tempo-real diferem dos sistemas convencionais por apresentarem restrições de tempo e tratarem, em muitos casos, com situações críticas. Isto implica que qualquer tipo de falha, inclusive falha temporal, pode causar conseqüências irreparáveis. Tolerância a falhas nesses sistemas é uma exigência óbvia e consolidada. É suportada regularmente pelo hardware do sistema. Com a complexidade aumentada por um número maior de componentes de software e sofisticadas interações, novas soluções para sistemas distribuídos passíveis de implementação em vários níveis de software e hardware precisam ser desenvolvidas e validadas quanto a correção e temporização.

### Falhas em sistemas distribuídos

Algumas características básicas devem ser consideradas para alcançar dependabilidade em sistemas distribuídos. Dependabilidade pode ser necessária, em maior ou menor grau, dependendo da aplicação. Mas todos os sistemas distribuídos devem suprir um mínimo de confiabilidade para permitir operação continuada do sistema mesmo com queda de um ou mais de seus nodos e canais. Modelos clássicos de falhas para sistemas distribuídos são os de Cristian - falhas de crash, omissão, temporização, resposta e arbitrária Schneider, que estende o modelo de Cristian. - fail-stop, crash, omissão de envio, omissão de recepção, temporização, resposta e arbitrária. Os dois modelos refletem falhas que afetam as trocas de mensagem entre os nodos.

## Processadores Fail-stop e armazenamento estável

O sistema deve preservar seu estado global mesmo na presença de falhas. Para tanto é necessário que seus nodos possuam armazenamento estável. O armazenamento estável preserva as informações armazenadas mesmo na ocorrência de falhas. Deve ser lembrado, entretanto, que falhas podem ocorrer inclusive no armazenamento secundário, seja disco magnético ou ótico. Assim o armazenamento secundário não pode ser considerado armazenamento estável. Infelizmente armazenamento estável ideal é apenas uma abstração teórica, mas algumas soluções práticas como RAID ou espelhamento de memória permitem chegar próxima a essa abstração. O melhor comportamento sob falha para os nodos de um sistema distribuído é simplesmente parar toda e qualquer operação na presença de uma falha interna irrecuperável, assim os demais nodos podem detectar seu estado pela ausência de mensagens. Armazenamento estável e componentes fail-stop são dois conceitos importantes aqui.

Assim como armazenamento estável, processadores fail-stop são apenas uma abstração. Processadores fail-stop, apesar de serem assumidos por grande parte das soluções práticas

de tolerância a falhas, são existem. Uma aproximação são processadores k+1 fail-stop que toleram k falhas.

**Tratamento de Falhas em Sistemas Distribuídos**

Técnicas possíveis para o tratamento de falhas em Sistemas Distribuídos:

- Detecção de falhas: Um exemplo sobre esse processo é utilizar checksums para detectar alterações de pacotes de rede; sabemos sobre a dificuldade em falhas como crash de servidores, porém, o desafio é gerenciar as ocorrências que podem ser presumidas.
- Mascaramento de falhas: As falhas detectadas podem ser escondidas ou tornadas menos severas; Mensagens podem ser retransmitidas; dados podem ser armazenados em multíplos discos; se um falha, o outro pode continuar em uso; no pior caso, essas técnicas podem não funcionar.
- Tolerar falhas: A maioria dos serviços na Internet apresentam falhas; não seria prático tratar todas elas e escondê-las dos usuários; o navegador não pode ficar indefinidamente a espera de um retorno do site solicitado; nesse caso, um erro é apresentado ao usuário.
- Recuperação de falhas: envolve o projeto do software, como por exemplo, efetuar um roll-back numa falha de servidor.
- Redundância: São exemplos:

* Duas rotas para acessar um servidor importante; =No DNS, as tabelas são replicadas em pelo menos 2 servidores diferentes;
* Um BD pode ser replicado em vários servidores;
* Servidores projetados para detectar falhas em seus pares;
* Clientes são redirecionados a um backup quando um servidor falha.

## EXERCÍCIOS

1

Tolerância a falha é uma questão importante no projeto de sistemas distribuídos; é definida como a característica:

- A)
Pela qual um sistema pode mascarar a ocorrência e a recuperação de falhas somente quando os servidores estiverem disponíveis.
- B)
Pela qual um sistema chama imediatamente o suporte a fim de garantir a sua integridade física.
- C)
Pela qual um sistema monocamada garante o seu funcionamento durante 6 dias da semana.
- D)
Pela qual um sistema pode mascarar a ocorrência e a recuperação de falhas. Um sistema é tolerante a falha se puder continuar a funcionar na presença de falhas.
- E)
Pela qual um sistema utilizando terminais burros com processamento local garante o seu funcionamento 24 horas por dia durante os sete dias da semana, ou seja, o sistema jamais ficará fora do ar.

2

Em sistemas distribuídos a segurança desempenha um papel de extrema importância.
Sobre as afirmativas é possível afirmar:
I - Deve oferecer facilidades para estabelecer canais seguros entre processos. Um canal seguro fornece os meios para autenticar mutuamente as partes comunicantes e proteger mensagens contra modificação durante sua transmissão. De modo geral, um canal seguro também oferece confidencialidade, de maneira que ninguém, exceto as partes comunicantes, podem ler as mensagens que passam pelo canal.
II - A autorização trata de proteger recursos de modo tal que somente processos que tenham os direitos de acesso adequados possam realmente acessar e usar esses recursos. O controle de acesso sempre ocorre depois que um processo foi autenticado. Relacionada com o controle de acesso está a prevenção contra o ataque de recusa de serviço, que é um problema difícil para sistemas que são acessíveis pela Internet.
III - O gerenciamento de chaves inclui a distribuição de chaves criptográficas na qual os certificados emitidos por terceiros de confiança desempenham papel importante. Um aspecto relevante em relação ao gerenciamento de autorização são os certificados de atributo e delegação.
Qual das alternativas é verdadeira?

- A)
Somente os itens I e II são verdadeiros.
- B)
Somente os itens II e III são verdadeiros.
- C)
Somente os itens I e III são verdadeiros.
- D)
Os itens I, II e III são falsos.
- E)
Os itens I, II e III são verdadeiros.

3

Tolerância à falha é uma questão importante no projeto de sistemas distribuídos. Definida como a característica pela qual um sistema pode mascarar a ocorrência e a recuperação de falhas. Em outras palavras, um sistema é tolerante a falha se puder continuar a funcionar na presença de falhas. Existem vários tipos de falhas. Uma falha por queda ocorre quando um processo simplesmente para. Uma falha por omissão ocorre quando um processo não responde a requisições que chegam. Quando um processo responde muito cedo ou muito tarde a uma requisição, diz-se que ele exibe uma falha de temporização. Responder a uma requisição que chega, mas do modo errado, é um exemplo de uma falha de resposta. As falhas mais difíceis de tratar são aquelas em que um processo exibe qualquer tipo de falha, denominadas falhas arbitrárias ou bizantinas. Qual a técnica utilizada para contornar uma falha em um ambiente distribuído?

- A)
Redundância.
- B)
Confiabilidade.
- C)

Tolerância.

- D)
  Disponibilidade.
- E)
  Comunicação.

4

POSCOMP 2004
Os tipos mais comuns de defeitos em sistemas distribuídos provocados por falhas físicas de componentes ou interferência eletro-magnética são:

- A)
  Perda de arquivos, colapso de servidores, captura de senhas.
- B)
  Captura de senhas, sobrecarga de servidores, mensagens duplicadas.
- C)
  Perda de mensagens, mensagens com vírus e mensagens órfãs.
- D)
  Colapso de servidores, queda do enlace e perda de mensagens.
- E)
  Páginas web com endereço errado, spam e mensagens duplicadas.

5

Sabemos que computadores e sitemas falham! Não é uma questão de SE eles vão falhar, mas de QUANDO! Em um falha, um computador pode devolver um resultado errado, mas, em geral, eles param antes de dar a resposta; num SD, falhas são parciais: outros componentes continuam funcionando; quais são as técnicas que podem ser usadas no tratamento desse tipo de ocorrência?

- A)
  Resolução de falhas, mascaramento de falhas, tolerar falhas, recuperação de falhas e redundância.
- B)
  Detecção de falhas, mascaramento de falhas, recuperação de falhas e redundância.
- C)
  Detecção de falhas, mascaramento de falhas, tolerar falhas e recuperação de falhas.
- D)
  Detecção de falhas, mascaramento de falhas, tolerar falhas, recuperação de falhas e redundância.
- E)
  Detecção de falhas, mascaramento de falhas, tolerar falhas, recuperação de falhas e concordância.

1D 2E 3A 4D 5D

# Conteúdo 6 - Clusters de Computadores

**Clusters de Computadores**

**Introdução**

Quando o assunto é computação de alto desempenho, não é difícil pensarmos em servidores sofisticados e caros respondendo por este trabalho. No entanto, é possível obter resultados tão bons quanto ou superiores a partir de alguma solução de **cluster** - uma tecnologia capaz de fazer computadores mais simples trabalharem em conjunto, como se formassem uma máquina só. Neste texto, você saberá o que é computação em cluster, verá quais são as principais características do conceito e conhecerá algumas de suas aplicações, assim como soluções do tipo.

**O que é cluster?**

Cluster (ou clustering) é, em poucas palavras, o nome dado a um sistema que relaciona dois ou mais computadores para que estes trabalhem de maneira conjunta no intuito de processar uma tarefa. Estas máquinas dividem entre si as atividades de processamento e executam este trabalho de maneira simultânea.

Cada computador que faz parte do cluster recebe o nome de nó (ou node). Teoricamente, não há limite máximo de nós, mas independentemente da quantidade de máquinas que o compõe, o cluster deve ser "transparente", ou seja, ser visto pelo usuário ou por outro sistema que necessita deste processamento como um único computador.

Os nós do cluster devem ser interconectados, preferencialmente, por uma tecnologia de rede conhecida, para fins de manutenção e controle de custos, como a Ethernet. É extremamente importante que o padrão adotado permita a inclusão ou a retirada de nós com o cluster em funcionamento, do contrário, o trabalho de remoção e substituição de um computador que apresenta problemas, por exemplo, faria a aplicação como um todo parar.

A computação em cluster se mostra muitas vezes como uma solução viável porque os nós podem até mesmo ser compostos por computadores simples, como PCs de desempenho mediano. Juntos, eles configuram um sistema de processamento com capacidade suficiente para dar conta de determinadas aplicações que, se fossem atendidas por supercomputadores ou servidores sofisticados, exigiriam investimentos muito maiores.

Não é necessário haver um conjunto de hardware exatamente igual em cada nó. Por outro lado, é importante que todas as máquinas utilizem o mesmo sistema operacional, de forma a garantir que o software que controla o cluster consiga gerenciar todos os computadores que o integram.

**Tipos de clusters**

Há uma enormidade de aplicações que só podem ser atendidas satisfatoriamente com computação de alto desempenho: sistemas meteorológicos, ferramentas de mapeamento genético, simuladores geotérmicos, programas de renderização de imagens tridimencionais, entre tantos outros. Com o advento da computação em nuvens, este cenário se torna ainda mais amplo: pode-se ter uma infraestrutura tecnológica respondendo a vários clientes simultaneamente de maneira remota, por exemplo.

Em todos estes casos e em qualquer outro tipo de aplicação crítica - que não pode parar de funcionar ou não pode perder dados (os sistemas bancários, por exemplo) -, o cluster pode se mostrar como uma solução viável, desde que o tipo mais adequado seja escolhido.

Há vários tipos de cluster, mas os principais são: cluster de alto desempenho, cluster de ata disponibilidade e cluster de balanceamento de carga.

**Cluster de Alto Desempenho (High Performance Computing Cluster)**

**Clusters de alto desempenho** são direcionados a aplicações bastante exigentes no que diz respeito ao processamento. Sistemas utilizados em pesquisas científicas, por exemplo, podem se beneficiar deste tipo de cluster por necessitarem analisar uma grande variedade de dados rapidamente e realizar cálculos bastante complexos.

O foco deste tipo é o de permitir que o processamento direcionado à aplicação forneça resultados satisfatórios em tempo hábil, mesmo que haja centenas de milhares de gigaflops envolvidos com a tarefa (1 gigaflop corresponde a 1 bilhão de instruções de ponto flutuante executadas por segundo).

**Cluster de Alta Disponibilidade (High Availability Computing Cluster)**

Nos **clusters de alta disponibilidade**, o foco está em sempre manter a aplicação em pleno funcionamento: não é aceitável que o sistema pare de funcionar, mas se isso acontecer, a paralização deve ser a menor possível, como é o caso de soluções de missão crítica que exigem disponibilidade de, pelo menos, 99,999% do tempo a cada ano, por exemplo.

Para atender a esta exigência, os clusters de alta disponibilidade podem contar com diversos recursos: ferramentas de monitoramento que identificam nós defeituosos ou falhas na conexão, replicação (redundância) de sistemas e computadores para substituição imediata de máquinas com problemas, uso de geradores para garantir o funcionamento em caso de queda de energia, entre outros.

Em determinadas circunstâncias, é tolerável que o sistema apresente algum grau de perda de desempenho, especialmente quando esta situação é consequência de algum esforço para manter a aplicação em atividade.

**Cluster para Balanceamento de Carga (Load Balancing)**

Em **clusters de balanceamento de carga**, as tarefas de processamento são distribuídas o mais uniformemente possível entre os nós. O foco aqui é fazer com que cada computador receba e atenda a uma requisição e não, necessariamente, que divida uma tarefa com outras máquinas.

Imagine, por exemplo, que um grande site na internet receba por volta de mil visitas por segundo e que um cluster formado por 20 nós tenha sido desenvolvido para atender a esta demanda. Como se trata de uma solução de balanceamento de carga, estas requisições são distribuídas igualmente entre as 20 máquinas, de forma que cada uma receba e realize, em média, 50 atendimentos a cada segundo.

Não basta ao cluster de balanceamento de carga ter um mecanismo meramente capaz de distribuir as requisições - é necessário que este procedimento seja executado de forma a garantir um "equilíbrio" na aplicação. Para tanto, o mecanismo pode monitorar os nós constantemente para verificar, por exemplo, qual máquina está lidando com a menor quantidade de tarefas e direcionar uma nova requisição para esta.

O balanceamento de carga pode ser utilizado em vários tipos de aplicações, mas o seu uso é bastante comum na internet, já que soluções do tipo têm maior tolerância ao aumento instantâneo do número de requisições, justamente por causa do equilíbrio oriundo da distribuição de tarefas.

**Combinação de tipos de clusters**

É válido frisar que uma solução de cluster não precisa se "prender" a apenas um tipo. Conforme a necessidade, podemos combinar características de tipos diferentes no intuito de atender plenamente à aplicação.

Por exemplo, uma loja na internet pode utilizar um cluster de alta disponibilidade para garantir que suas vendas possam ser realizadas 24 horas por dia e, ao mesmo tempo, aplicar balanceamento de carga para suportar um expressivo aumento eventual no número de pedidos causados por uma promoção.

**Funcionamento básico dos clusters**

Para que um cluster seja constituído, é necessário fazer uso de alguns elementos básicos. O primeiro deles você já conhece: os equipamentos a serem utilizados como nós.

Para isso, pode-se usar máquinas construídas especificamente para funcionar como nós. Neste caso, os computadores teriam apenas dispositivos de hardware imprescindíveis ao cluster.

Mas, também é possível utilizar computadores "convencionais", como desktops para fins domésticos ou para uso em escritório. Assim, uma universidade ou uma empresa, por exemplo, pode utilizar máquinas que foram substituídas por modelos mais recentes para criar um cluster e, eventualmente, economizar com a aquisição de servidores.
Os nós podem ainda ser não dedicados ou dedicados. No primeiro caso, cada computador que faz parte do cluster não trabalha exclusivamente nele. No segundo, o nó é utilizado somente para este fim, fazendo com que dispositivos como teclados e monitores sejam dispensáveis - se, por algum motivo, for necessário acessar uma máquina em particular, pode-se fazê-lo via terminal, a partir do nó principal, por exemplo.
Outro elemento importante é o sistema operacional. Como já informado, os nós não precisam ser exatamente iguais no que diz respeito ao hardware, mas é essencial que todas os computadores utilizem o mesmo sistema operacional.
Esta homogeneidade é importante para diminuir a complexidade de configuração e manutenção do sistema, e garantir que os procedimentos rotineiros ao cluster, como monitorização, distribuição de tarefas e controle de recursos sejam executados de maneira uniforme. Para reforçar estes aspectos, pode-se até mesmo adotar sistemas operacionais preparados especialmente para clustering.
Do ponto de vista do software, o cluster conta ainda com o elemento que faz o papel de middleware: trata-se de um sistema que permite o controle do cluster em si e, portanto, está intimamente ligado ao sistema operacional. É o middleware que lida, por exemplo, com as bibliotecas que fazem toda a comunicação do cluster - uma delas é o padrão MPI (Message Passing Interface).
Além de trabalhar com o gerenciamento do cluster, o middleware oferece uma interface para que um administrador possa configurar o cluster, ferramentas para manutenção e otimização, recursos de monitoramento e assim por diante.
Por padrão, o middleware é instalado em uma máquina chamada de nó controlador (ou nó mestre). O nome deixa claro: trata-se do já mencionado nó principal, que efetivamente controla o cluster a partir da distribuição de tarefas, do monitoramento e de procedimentos relacionados.
A comunicação entre os nós - que é onde está a delimitação do que constitui o cluster em si - é feita a partir de uma tecnologia de rede local. Os padrões Ethernet (Gigabit Ethernet, Fast Ethernet, etc) são bastante utilizados justamente por serem mais comuns e, portanto, melhor suportados e menos custosos. Mas há outras opções viáveis, entre elas, o Myrinet e o InfiniBand, ambos com características bastante apropriadas para clustering.

**Cluster Beowulf**
O **Beowulf** não é, necessariamente, um middleware, como muitas pensam. Na verdade, este nome faz referência a um padrão de clustering disponibilizado pela NASA (National Aeronautics and Space ) em 1994 e amplamente adotado desde então.
Originalmente, Beowulf é o nome de um poema extenso e bastante antigo, cujo manuscrito foi encontrado no século XI. A obra descreve os atos de um herói de mesmo nome que se destaca por sua força descomunal e que, portanto, enfrenta um perigoso monstro para salvar um reino. A história serviu de inspiração para que os pesquisadores Thomas Sterling e Donald Becker, da NASA, batizassem o projeto de cluster no qual trabalhavam de Beowulf.
Um cluster Beowulf se define, basicamente, pela ênfase nas seguintes características:
- entre os nós, deve haver pelo menos um que atue como mestre para exercer o controle dos demais. As máquinas mestres são chamadas de front-end; as demais, de back-end. Há a possibilidade de existir mais de um nó no front-end para que cada um realize tarefas específicas, como monitoramento, por exemplo;
- a comunicação entre os nós pode ser feita por redes do tipo Ethernet, mais comuns e mais baratas, como você já sabe;
- não é necessário o uso de hardware exigente, nem específico. A ideia é a de se aproveitar componentes que possam ser encontrados facilmente. Até mesmo PCs considerados obsoletos podem ser utilizá-los;

- o sistema operacional deve ser de código aberto, razão pela qual o Linux e outras variações do Unix são bastante utilizados em cluster Beowulf. O MOSIX, a ser abordado no próximo tópico, é uma opção bastante usada para este fim;
- os nós devem se dedicar exclusivamente ao cluster;
- deve-se fazer uso de uma biblioteca de comunicação apropriada, como a PVM (Parallel Virtual Machine) ou a MPI (Message Passing Interface). Ambas são direcionadas à troca de mensagens entre os nós, mas o MPI pode ser considerado mais avançado que o PVM, uma vez que consegue trabalhar com comunicação para todos os computadores ou para apenas um determinado grupo.

Perceba que, com estas características, pode-se construir um cluster "poderoso" e, ao mesmo tempo, poupar gastos com equipamentos, licenças de software e manutenção. O cluster montado por Thomas Sterling e Donald Becker para a NASA, por exemplo, era composto por 16 PCs com processador Intel 486 DX4 e sistema operacional Linux conectados por uma rede Ethernet de 10 Mb/s. Como se vê, esta foi uma solução consideravelmente mais barata e, possivelmente, menos complexa que um supercomputador. É possível saber mais sobre o Beowulf em www.beowulf.org.

**Algumas soluções de clusters**

Há uma quantidade razoável de soluções para clusters, mas algumas se sobressaem, especialmente aquelas que se relacionam com Linux e outros sistemas baseados em Unix. Vejamos rapidamente opções do tipo que se destacam bastante.

**MOSIX**

O **MOSIX** (Multicomputer Operating System for Unix) é uma das opções mais tradicionais quando o assunto é clustering. Trata-se, resumidamente, de um conjunto de softwares que permite a implementação de clusters em sistemas baseados no Unix, tendo forte ênfase em balanceamento de carga e alto desempenho.

Entre as suas principais características estão: possibilidade de trabalhar com nós dedicados e não dedicados; suporte não apenas a CPUs, mas também a GPUs (a partir da versão 2); migração dinâmica de processos (um nó mais potente pode assumir determinada tarefa para evitar sobrecarga em outro); e possibilidade de remoção e inclusão de nós sem interromper o cluster.

Como o MOSIX também trabalha com nós não dedicados, é possível inclusive fazer com que as máquinas de um escritório passem a trabalhar no cluster após o horário do expediente, por exemplo. Mas, mesmo se houver usuários utilizando os nós, é possível manter o cluster em funcionamento, já que as atividades do MOSIX são totalmente "transparentes", ou seja, seu trabalho no computador não é perceptível.

O MOSIX não é uma solução gratuita. Houve uma versão sua de código aberto chamada OpenMosix que, infelizmente, foi descontinuada em março de 2008.

**OpenSSI**

O **OpenSSI** é uma solução aberta para clusters focada em ambientes Linux. O nome tem como base o conceito de SSI (Single System Image), ou seja, um sistema que considera vários nós, mas se parece, no ponto de vista do usuário, apenas como um único computador.

A "visão" de apenas uma única máquina também é válida para os softwares: este não precisam ser alterados para "enxergar" cada nó e assim rodar no cluster, facilitando a implementação da solução.

O OpenSSI pode lidar tanto com alto desempenho quanto com alta disponibilidade, além de possuir recursos para balanceamento de carga.

**Kerrighed**

O **Kerrighed** é outra opção SSI aberta para clusters que tem como base o Linux. Esta solução se destaca principalmente por fazer uso do conceito de Distributed Shared Memory (DSM) - algo como "Memória Compartilhada Distribuída" -, onde a memória de cada nó se "soma" a dos demais, como se formassem um volume único à disposição de todo o cluster.

Esta abordagem oferece vários benefícios: parte da memória pode ser disponibilizada para todos os clientes da aplicação; sistemas desenvolvidos para rodar de maneira centralizada podem ser executados de maneira distribuída dentro da solução; o cluster pode se comportar como se fosse uma máquina com múltiplos processadores; entre outros.

**Vantagens e desvantagens dos clusters**

Neste ponto do texto, você certamente já compreendeu as vantagens de um cluster. Eis as principais:

- pode-se obter resultados tão bons quanto ou até superiores que um servidor sofisticado a partir de máquinas mais simples e mais baratas (ótima relação custo-benefício);
- não é necessário depender de um único fornecedor ou prestador de serviço para reposição de componentes;
- a configuração de um cluster não costuma ser trivial, mas fazer um supercomputador funcionar poder ser muito mais trabalhoso e exigir pessoal especializado;
- é possível aumentar a capacidade de um cluster com a adição de nós ou remover máquinas para reparos sem interromper a aplicação;
- há opções de softwares para cluster disponíveis livremente, o que facilita o uso de uma solução do tipo em universidades, por exemplo;
- relativa facilidade de customização para o perfeito atendimento da aplicação;
- um cluster pode ser implementado tanto para uma aplicação sofisticada quanto para um sistema doméstico criado para fins de estudos, por exemplo.

Mas, apesar destes benefícios, os clusters não são a solução perfeita para todo e qualquer problema computacional. Há aplicações onde o uso de outra tecnologia se mostra mais adequado. Entre as razões para isso estão:

- a facilidade de expansão do cluster pode ser uma "faca de dois gumes": a quantidade de máquinas pode aumentar tanto que a manutenção se torna mais trabalhosa, o espaço físico pode ficar impróprio, etc;
- a tecnologia de comunicação utilizada pode não oferecer a velocidade de transferência de dados ou o tempo de resposta necessário, dependendo da aplicação;
- um cluster tem como base uma rede local, logo, não se pode acrescentar máquinas que estejam muito distantes geograficamente.

Por estes aspectos, fica evidente que as necessidades e os requisitos de uma aplicação devem ser bem avaliados para que se possa decidir entre a implementação de um cluster ou outra tecnologia. Se o clustering for a opção escolhida, deve-se seguir com a avaliação, desta vez para se decidir sobre as soluções e recursos disponíveis.

**Finalizando**

A origem da denominação "cluster" não é clara, mas sabe-se que as primeiras soluções de processamento paralelo remontam à década de 1960, havendo, a partir daí, alguns princípios que hoje formam a base da ideia de clustering.

O fato é que o passar do tempo não torna o conceito ultrapassado. Há um motivo especial para isso: os clusters se relacionam intimamente à otimização de recursos, uma necessidade constante em praticamente qualquer cenário computacional. E este aspecto pode se tornar ainda mais atraente quando a ideia de cluster é associada a conceitos mais recentes, como cloud computing e virtualização.

Escrito por Emerson Alecrim - Publicado em 22_03_2013 - Atualizado em 22_03_2013

## EXERCÍCIOS

1

Qual a definição de um cluster de Alta Performance?

- A)
Tipo de sistema para processamento paralelo ou distribuído que consiste de uma coleção de computadores interconectados, trabalhando isoladamente como um recurso de computação simples e integrado. Constituído de vários nós. O computador mestre é o responsável pela execução de todas as tarefas.
- B)
Os equipamentos são usados em conjunto para manter um serviço ou equipamento sempre ativo, replicando serviços e servidores, evitando-se máquinas paradas, ociosas, esperando apenas o outro equipamento ou serviço paralisar, passando os demais a responder por elas normalmente.
- C)
Tipo de sistema para processamento paralelo ou distribuído que consiste de uma coleção de computadores interconectados, trabalhando juntos como um recurso de computação simples e integrado. Constituído de vários nós escravos, controlados por um computador mestre, ao qual se tem acesso direto. O computador mestre é o responsável pela distribuição das tarefas. Os escravos limitam-se a processar os dados que lhes são enviados.
- D)
Os equipamentos são usados em conjunto para manter um serviço ou equipamento sempre desativado, replicando serviços e servidores, evitando-se máquinas paradas, ociosas, esperando apenas o outro equipamento ou serviço paralisar, passando os demais a responder por elas normalmente.
- E)
Os equipamentos são usados independentemente para manter um serviço ou equipamento sempre ativo, não replicando serviços e servidores, evitando-se máquinas paradas, ociosas, esperando apenas o outro equipamento ou serviço paralisar, passando os demais a responder por elas normalmente.

2

No processo de execução de um cluster qual a função da biblioteca PVM?

- A)
Gerenciar o sistema de arquivos do cluster.
- B)
Gerenciar a memória do Cluster.
- C)
Gerenciar a troca de mensagens entre os nós e o mestre.
- D)
Gerenciar a rede do cluster.
- E)
Gerenciar os processadores do mestre.

3

No início da computação a grande preocupação estava voltada para o desenvolvimento do hardware, que era muito caro e sem um fim que justifica-se tal investimento, muitas fases se passaram com resultados assombrosos do ponto de vista da evolução tecnológica. De uma máquina que custava milhões de dólares e executava uma instrução por segundo, chegamos a máquinas que custam mil dólares e podem executar um bilhão de instruções por segundo, um ganho preço/desempenho de 10 elevado a 13. O segundo desenvolvimento marcante foram as redes de computadores. O resultado dessas evoluções tecnológicas foi à possibilidade de se montar sistemas de computação compostos por grandes quantidades de computadores conectados por uma rede de alta velocidade. Como os sistemas distribuídos se encaixam neste contexto.

- A)
Os sistemas distribuídos são dependentes de um computador de grande porte onde o processamento é concentrado. As estações dos clientes não executam nenhum processamento.
- B)
São compostos por um conjunto de computadores independentes que se apresentam a seus usuários como um sistema único e coerente.
- C)
São compostos por um conjunto de computadores dependentes entre si que se apresentam a seus usuários como um sistema em várias partes.
- D)

São compostos por um conjunto de computadores independentes que se apresentam a seus usuários como um sistema único e possuem como grande fator de qualidade a falta de uma rede de comunicação entre as máquinas.

- E)
São de grande importância pois permitem que os dados trafeguem grandes distâncias sem a necessidade de novo carregamento.

4

A estrutura de agrupamento de computadores apresenta vantagens competitivas em relação aos ambientes multiprocessados de memória compartilhada (computadores com diversos processadores em uma placa-mãe), permitindo que o acréscimo de computadores torne o sistema mais rápido, possuindo componentes de fácil disponibilidade e manutenção. A afirmação se refere a:

- A)
Cluster de alta-disponibilidade.

- B)
Cluster de alto-desempenho.

- C)
Grid.

- D)
Super-computador paralelo.
- E)
Aplicação em camadas.

5

Qual a definição de um cluster de Alto Desempenho, que utiliza a biblioteca de passagem de mensagem MPI?

- A)
Tipo de sistema para processamento paralelo ou distribuído que consiste de uma coleção de computadores interconectados, trabalhando isoladamente como um recurso de computação simples e integrado. Constituído de vários nós. O computador mestre é o responsável pela execução de todas as tarefas.
- B)
Os equipamentos são usados em conjunto para manter um serviço ou equipamento sempre ativo, replicando serviços e servidores, evitando-se máquinas paradas, ociosas, esperando apenas o outro equipamento ou serviço paralisar, passando os demais a responder por elas normalmente.
- C)
Tipo de sistema para processamento paralelo ou distribuído que consiste de uma coleção de computadores interconectados, trabalhando juntos como um recurso de computação simples e integrado. Constituído de vários nós escravos, controlados por um computador mestre, ao qual se tem acesso direto. Qualquer um dos nós pode assumir o papel de mestre.
- D)
Tipo de sistema para processamento paralelo ou distribuído que consiste de uma coleção de computadores interconectados, trabalhando juntos como um recurso de computação simples e integrado. Constituído de vários nós escravos, controlados por um computador mestre, ao qual se tem acesso direto. O computador mestre é o responsável pela distribuição das tarefas. Os escravos limitam-se a processar os dados que lhes são enviados.
- E)
Tipo de sistema para processamento paralelo ou distribuído que consiste de uma coleção de computadores heterogêneos interconectados, trabalhando juntos como um recurso de computação simples e integrado. Constituído de vários nós escravos, controlados por um computador mestre, ao qual se tem acesso direto. Qualquer um dos nós pode assumir o papel de mestre.

6

Petrobras 2012
Um cluster tolerante à falha assegura que todos os recursos de um sistema estão sempre disponíveis. A função de trocar aplicações e recursos de armazenamento de dados de um sistema que falhou para um sistema alternativo do cluster é denominada

- A)
Autorestore.
- B)
Autorecover.
- C)
Failoff.

- D)
  Failback.
- E)
  Failover.

7

Petrobrás 2012
A organização de clusters constitui uma alternativa para os multiprocessadores simétricos. Dois principais objetivos ou requisitos de projeto desse tipo de organização são a escalabilidade.

- A)
  Parcial e a média disponibilidade.
- B)
  Parcial e a alta disponibilidade.
- C)
  Incremental e a média disponibilidade.
- D)
  Incremental e a alta disponibilidade.
- E)
  Absoluta e a média disponibilidade.

8

Petrobras 2012
A denominação SMP refere-se a um sistema de computador independente, no qual existem dois ou mais processadores similares com capacidade de computação comparável, que:

- A)
  Compartilham a mesma memória principal e as facilidades de E/S (Entrada/Saída).
- B)
  Compartilham a mesma memória principal, mas não compartilham as facilidades de E/S (Entrada/Saída).
- C)
  Compartilham as facilidades de E/S (Entrada/Saída), mas não compartilham a mesma memória principal.
- D)
  Utilizam um esquema de conexão que impede um tempo de acesso uniforme à memória.
- E)
  Desempenham, necessariamente, funções distintas.

1C 2C 3B 4B 5D 6E 7D 8A

# Conteúdo 7 - Cluster Beowulf

**Cluster Beowulf**

Origem: Wikipédia, a enciclopédia livre.

Beowulf é o nome de um projeto para aglomerados de computadores (ou Clusters) para computação paralela, usando computadores pessoais, não especializados e portanto mais baratos. O projeto foi criado por Donald Becker da NASA, e são utilizados em todo mundo, por exemplo no processamento de dados com finalidade científica e na renderizacão de filmes de animação 3D.

**Introdução**

Para entender o que é um Cluster Beowulf, é necessário primeiro entender o conceito de um Cluster. Cluster é um termo bastante utilizado para definir a utilização de dois ou mais computadores independentes, interligados via rede, que trabalham em conjunto trocando informações entre si em torno de uma única tarefa. Indo mais a fundo, ele pode ser definido como um conjunto de nós processadores que, interligados, comportam-se como um sistema distribuído. Existem dois tipos de clusters:

- Alta Disponibilidade (HA- High Availability) - tem a finalidade de manter um determinado serviço de forma segura o maior tempo possível;
- Alta Performance (HPC- High Performance Computing) - é uma configuração designada a prover grande poder computacional, maior que somente um único computador pudesse oferecer em capacidade de processamento.

É possível a utilização de clusters HA e HPC juntos, onde se resulta num cluster com características combinadas entre estes dois tipos de clusters.

**Cluster Beowulf: O que é?**

Cluster Beowulf são clusters de desempenho escaláveis, baseados numa infraestrutura de hardware comum, rede privada e software 'open source' (Linux). O 'hardware comum' pode ser qualquer tipo de computador, significando que não é necessário usar equipamentos próprios para cluster, bastando utilizar equipamentos comuns a redes tradicionais e PCs. Para Clusters Beowulf, existe um servidor responsável por controlar todo o cluster, principalmente quanto à distribuição de tarefas e processamento. Como utiliza computadores comuns, o desempenho pode melhorar de acordo com as máquinas (nós) que o desenvolvedor (Linux) acrescentar, uma vez que o processamento é feito paralelamente. Isso porque a tarefa a ser processada é separada em partes independentes, distribuídas nos vários nós que estão na estrutura do cluster, nos quais as informações são processadas pela máquina que é designada como servidor do sistema. Um cluster Beowulf é um cluster voltado para desempenho, normalmente utilizado para processamento científico, ou seja, processamento em larga escala.

**Como implementar?**

Como não depende de equipamento específico, o cluster Beowulf é basicamente composto de uma máquina chamada de servidor e outras máquinas chamadas escravas, sendo uma ou mais. A configuração do cluster consiste essencialmente de dois passos: Configuração no nó mestre (servidor) e configuração dos nós escravos. A implementação acontece através de modificações no Linux, ou através do uso de ferramentas e bibliotecas de programação específicas para esse fim. Em todos os casos, o objetivo é permitir a distribuição das tarefas entre os PCs que fazem parte do cluster.

**Vantagens e Desvantagens**

**Vantagens:**

Quanto às vantagens de um Cluster Beowulf podemos citar:

- Sistemas escaláveis, sendo possível pôr em rede e coordenar um grande número de nós, não existindo um limite definido para o tamanho do cluster.
- Os equipamentos utilizados são facilmente comercializados, não necessitando de um equipamento específico para a criação do cluster.
- No caso de um nó defeituoso, a substituição é tão simples quanto mudar um PC. Desta forma, é possível gerenciar as falhas de maneira eficiente, baseando-se na fácil substituição de equipamentos.
- Existe uma mobilidade em relação ao fornecedor de hardware, uma vez que os equipamentos são comuns. Desta forma, quem for implementar o cluster Beowulf não fica amarrado a comprar peças de um único fornecedor.
- Com sistema operacional ‘open-source’, o software inclui o código fonte, permitindo o debug dos erros e correções.

**Desvantagens**

Clusters Beowulf parecem muito bons, no entanto, também existem algumas desvantagens:

- A grande maioria do hardware de rede não foi criada para funcionar em processamento paralelo.
- Mesmo com a popularidade crescendo, existem ainda poucos softwares que suportem e tratem este tipo de cluster como um sistema único.

**Opções de Software**

A comunidade sobre Beowulf vem crescendo bastante e estes clusters já estão totalmente integrados com softwares comercializados pelos seguintes fornecedores:

- HP
- Northrop Grumman
- Accelerated Servers
- Penguin Computing
- Aspen Systems
- PSSC Labs
- Custom Fit, Inc.
- Atipa
- Turbotek
- Microway

## EXERCÍCIOS

1

As afirmativas a seguir se referem a características marcantes dos Supercomputadores e dos clusters Beowulf:
I - Altíssimo custo;
II - Utiliza hardware popular;
III - Softwares proprietários;
IV - Total dependência de fornecedores;
V - Independência de fornecedores;
VI - Facilidade de Manutenção e Ampliação;
VII - Dificuldade de atualização;
VIII - Sistema Operacional Aberto (Linux).
Defina quais são as alternativas relacionadas a cada tipo de estrutura.

- A)
  Supercomputadores (I , II, IV, VI) e Cluster (III, V, VII,VIII).
- B)
  Supercomputadores (II, V, VI, VIII) e Cluster ( I, III, IV, VII).
- C)
  Supercomputadores (II, III, IV,VIII) e Cluster (I,V,VI,VII).
- D)
  Supercomputadores (I, IV, VI, VII) e Cluster (II, III, V, VIII).
- E)
  Supercomputadores ( I, III, IV ,VII) e Cluster (II, V, VI, VIII).

2

O sistema de arquivos de um Cluster Beowulf deve ser capaz de compartilhar os seus arquivos de forma transparente entre o mestre(servidor) e os nós (escravos). Qual o sistema de arquivos utilizado?

- A)
  Sistema de arquivos local em cada máquina.
- B)
  Sistema de arquivos semelhante ao NTFS.
- C)
  Sistema de arquivos em rede utilizado para acessar de forma remota as máquinas windows.
- D)
  Sistema de arquivos em rede utilizado para compartilhar o arquivo que deverá ser executado pelo cluster, entre todos os nós participantes NFS.
- E)
  Sistema de arquivos que concorre como EXT3, EXT4 e NTFS.

3

Um Cluster é formado por um conjunto de computadores interligados através de uma rede, as máquinas membros deste cluster são denominadas nó ou node; na maioria das vezes o cluster é formado por computadores convencionais e se apresenta de forma transparente ao usuário, como sendo um único computador de grande porte; De acordo com Zem (2005), quais são os tipos de cluster mais utilizados atualmente?

- A)
  Cluster de Alto Processamento, Cluster de Alta Disponibilidade (High Availability - HA) e Cluster de Balanceamento de Carga (Horizontal Scaling - HS).
- B)
  Cluster de Alto Desempenho, Cluster de Alta Disponibilidade (High Availability - HA) e Cluster de Balanceamento de Carga (Horizontal Scaling - HS).
- C)
  Cluster de Alto Desempenho, Cluster de Alta Confiabilidade e Cluster de Balanceamento de Carga (Horizontal Scaling - HS).
- D)
  Cluster de Alto Custo, Cluster de Alta Disponibilidade (High Availability - HA) e Cluster de Balanceamento de Carga (Horizontal Scaling - HS).

- E)
Cluster de Alto Desempenho, Cluster de Alta Disponibilidade (High Availability - HA) e Cluster de Balanceamento de Conexão.

4

Qual a biblioteca de passagem utilizada pelo Cluster Beowulf?

- A)
Mpich.
- B)
SSH.
- C)
SCP.
- D)
TCP/IP.
- E)
Protocolo UDP.

5

Qual a função da biblioteca de passagem Mpich?

- A)
A biblioteca de troca de mensagens deve conter um conjunto de funções que serão utilizadas para distribuir o processamento entre as diversas máquinas do cluster.
- B)
É a biblioteca que atua nas estações que fazem parte do cluster.
- C)
É a biblioteca que efetua a comunicação TCP entre as estações do cluster.
- D)
É a biblioteca que efetua a comunicação TCP entre os servidores do cluster.
- E)
Essa biblioteca não é utiliza no Cluster BeoWulf.

1E 2D 3B 4A 5A

# Conteúdo 8 - Cluster x Grid

**Introdução**

Os altos custos que envolvem a aquisição de supercomputadores dedicados e a constante melhoria nos projetos de computadores pessoais e na estrutura das redes locais levou a ideia de se utilizar computadores pessoais independentes interconectados em rede como plataforma para executar aplicações paralelas e distribuídas. Todo o dia uma grande quantidade de usuários está insaciável por utilizar uma rede mundial de computadores, constituída por diversos dispositivos heterogêneos tanto ao nível de arquitetura quanto ao nível de sistemas, de maneira a utilizar e compartilhar esses recursos computacionais de uma forma transparente e eficiente. Governos, empresas e organizações de pesquisa estão trabalhando em associação para criar redes de supercomputação, e que disponibilizarão estes recursos computacionais agregados de um grupo para qualquer computador conectado. As empresas e universidades que investiram nos últimos anos em clusters de alta performance de computação, estão dando o próximo passo para a futura computação em grades computacionais (grid computing), e os pesquisadores estão trabalhando para criar uma interface padronizada de Web para que diversas pessoas em várias áreas distintas possam utilizar a supercomputação em grade como um serviço público tradicional, semelhantes aos sistemas elétricos e/ou água/esgoto, sem se preocupar em como os recursos estarão disponibilizados para o mesmo. John MacCarty do MIT em 1961 já previa o que hoje temos como grid. “Se os computadores do tipo que eu imagino, se tornarem os computadores do futuro, então a computação poderá algum dia ser organizada como um serviço público, assim como a telefonia o é... Esse serviço poderá se tornar a base da nova e importante indústria”

A Internet é formada por sistemas totalmente heterogêneos, fracamente acoplados e geograficamente distribuída. Pense na quantidade de máquinas que estão ociosas nesse momento. Os sistemas em cluster são caracterizados em uma interligação física localizada, construídos para resolver os problemas de uma forma mais rápida possível por meio de aplicações paralelas e distribuídas. E isso já foi explicado em artigos posteriores. Já começam a ser consideradas que algumas aplicações já não podem ser mais resolvidas dentro de um ambiente de cluster e podem necessitar o uso de recursos em localidades remotas. O uso dos ciclos fracamente acoplado de computadores dispersos em uma Wan pode representar um ambiente muito interessante para processamento de alto desempenho. Com isso um usuário acessaria via estação de trabalho uma aplicação dentre um conjunto de aplicações disponíveis e submeter suas tarefas para serem executados nesse ambiente. Seria de responsabilidade de o próprio sistema cuidar de questões como sincronização de tarefas, disponibilidade de recursos (processadores, armazenamento, compartilhamento de dados), transferências de informações, protocolos de comunicação, troca de mensagens e segurança, fossem resolvidos de forma transparente ao usuário. O sistema seria visto pelo usuário como um supercomputador virtual, capaz de executar inclusive tarefas concorrentemente. Esses recursos poderiam ser utilizados para prover muito mais do que simples serviços de comunicação, e-mail eletrônico ou transferência de arquivos; eles teriam o potencial de fornecer um ambiente computacional simples no quais ciclos de processamento, comunicação e até mesmos dados seriam compartilhados através de estações de trabalhos espalhados em prédios, estados e até mesmo entre continentes. Uma experiência de integração de processamento distribuído é o projeto SETI@home (https://setiathome.ssl.berkeley.edu/), uma continuação do projeto da NASA de busca de inteligência extraterrestre. Usando um software que pode ser baixado pela Internet, um microcomputador pode analisar sinais do rádio telescópio de Arecibo. Atualmente, existem 4 milhões de assinantes em 224 países, criando um computador virtual com uma performance de 20 Tflops. Um outro exemplo são as famosas redes peer-to-peer, como Emule (Edonkey), Kazaa, Gnutella, em que se compartilham arquivos por exemplo, mas sem nenhum controle de acesso e não interoperam entre si. Com a evolução dessas aplicações elas acabaram por inter-operar e haverá uma convergência de interesses entre computação ponto a ponto, Internet e computação em Grade. A Computação em Grade é próxima geração da Internet – são protocolos adicionais construídos sobre a tecnologia da Internet. Qualquer recurso que esteja na Grade, também está na Rede.

**Grid Computing**

O Grid Computing é um novo conceito que explora as potencialidades das redes de computadores, com o objetivo específico de disponibilizar camadas virtuais que permitem a um usuário ter acesso a aplicações altamente exigentes, bem como aderir a comunidades virtuais de grande escala, com uma grande diversidade de recursos de computação e de repositórios de informações. A diferença existente entre a computação distribuída e computação em grade de dá pelo fato de que a computação distribuída é um conceito que vem dos anos 80 e 90, e consiste na possibilidade de resolver um determinado problema computacional através da utilização de diferentes recursos distribuídos geograficamente. A computação distribuída passa a ser uma “Computação em Grade” no momento em que existe uma infra-estrutura física e uma infra-estrutura lógica (software) que permita coordenar os trabalhos que vão ser processados e garantir a sua qualidade de serviço. O Grid é um caso particular da computação distribuída, uma vez que os Grids são orientados essencialmente para aplicações que precisam de uma grande capacidade de cálculos, ou enormes quantidades de dados transmitidos de um lado para o outro, ou as duas. O nome Grid foi idealizado baseado nas malhas de interligação dos sistemas de energia elétrica (power-grids), em que um usuário utiliza a eletricidade sem ao menos saber em que local ele foi gerada, sendo totalmente transparente aos seus usuários. O surgimento das Grids Computacionais nasceu da comunidade de Processamento de Alto Desempenho (PAD). O conceito foi apresentado pelos pesquisadores Ian Foster e Carl Kesselman, sendo composto por uma infraestrutura de hardware e software que permite-nos acesso a grandes

capacidades computacionais geograficamente distribuídas, de forma confiável, consistente, econômica e persistente. Na verdade o conceito é antigo, mas com uma nova dinâmica, em que se pode utilizar a capacidade de computação (ex. Storage/CPU) sem ter que se preocupar de onde vem, como é mantida, fazendo uma metáfora às redes elétricas. Chamamos de Organização Virtual (VO) quando temos participantes que desejam compartilhar recursos para poder concluir uma tarefa. Além disso, o compartilhamento esta além de apenas troca de documentos, isto pode envolver acesso direto a software remoto, computadores, dados, sensores e outros recursos. Ian Forster traduz os conceitos de Grid de duas formas clássicas: “Compartilhamento de recursos coordenados e resolução de problemas em organizações virtuais multi-institucionais dinâmicas” e “Grids Computing são sistemas de suporte à execução de aplicações paralelas que acoplam recursos heterogêneos distribuídos, oferecendo acesso consistente e barato aos recursos, independente de sua posição física. A tecnologia de Grids Computing possibilita agregar recursos computacionais variados e dispersos em um único ´supercomputador virtual´, acelerando a execução de várias aplicações paralelas. Grids se tornaram possíveis nos últimos anos, devido a grande melhoria em desempenho e redução de custo, tanto de redes de computadores quanto de microprocessadores”. A computação em grade está gerando especulações não somente na área científica, mas também na empresarial (IBM, HP/Compaq, Oracle, Sun e Fujitsu), pois irá permitir a redução de custos e tempo, aumento de produtividade, compartilhamento de recursos e informações, dentre outras possibilidades. Com sua expansão, pode-se chegar, no final, em algo como a formação de um Grid Global, uma rede distribuída global de colaborações entre seus participantes, capaz de prover recursos talvez inatingíveis por um cluster isolado. E isto já está sendo feito em diversos países do mundo, inclusive no Brasil. Grids são construídos como um grupamento de serviços básicos independentes. Um aspecto essencial dos serviços de Grid é que esses estão disponíveis uniformemente através dos ambientes distribuídos na Grid. Os serviços são agrupados em um sistema integrado, também chamado de middleware. Exemplos de ferramentas atuais de Grid incluem Globus, Legion, OpenGrid, AppLeS.

O Grid permite também o uso de técnicas de programação paralela por passagem de mensagens. O ambiente MPI (“Message Passing Interface”) está disponível no Grid através da versão MPICH-G2 (versão portátil do MPI para o Globus). O padrão MPI define uma biblioteca de rotinas que implementam uma comunicação ponto a ponto, em que a operação “send” é usada para iniciar uma transferência de dados entre dois programas concorrentes e a operação “receive” é usada para obter dados do sistema no espaço de memória da aplicação; existem ainda operações coletivas envolvendo múltiplos processos. Mas que devido a alta latência provocada na comunicação entre processos, as aplicações devem ser construídas com uma granularidade bem projetada de tal forma que se comuniquem o mínimo possível. Então, as aplicações mais adequadas ao Grid são as que possuem tarefas independentes (bag of tasks), pois as tarefas não dependem da execução de outras e podem ser executadas em qualquer ordem. O Grid Computing é um desafio bem maior do que formas mais simples de computação paralela e distribuída. Hoje, a maioria dos projetos de Grid permanece localizada nos centros de supercomputação e laboratórios universitários. Os centros de pesquisa são ligados a conexões em rede cerca de 20 vezes mais rápidas do que as conexões de banda largas normais, são equipadas com sistemas de armazenamento capazes de lidar com vastos arquivos de dados e com computadores de alta performance. O Grid Computing é um conceito sobre o qual existe ainda uma grande expectativa e que poderá evoluir em diferentes direções, mas que é já hoje entendido como a próxima geração da Web para a comunidade científica.

**Internet x Grid x Cluster**

A internet nasceu no início da década de 70, e somente a partir de 1995, com o aparecimento dos serviços www, foi que se tornou tão popular, saindo dos meios militares e acadêmicos. Seu objetivo era a interligação de diferentes ambientes computacionais e geograficamente dispersos. Os web sites desenvolvidos pela indústria sempre foram interoperáveis em relação usuário-site, por meio de aplicações criadas neste contexto, em que o usuário dispõe de um menu de serviços fechados. O que ocorre em um ambiente de Grid é o inverso, onde o usuário tem de submeter suas aplicações para serem resolvidas dentro do ambiente por ele montado. Um ambiente de cluster constitui em um sistema formado por hardware e software conectados em um local apenas, servindo a usuários que estão trabalho somente em um projeto, usado exclusivamente para resolver os problemas computacionais de uma determinada organização. Por outro lado, um Grid presta serviços de uma forma geograficamente distribuída. Em um cluster, os recursos são gerenciados por uma entidade central, e os computadores agem como se fosse um único dispositivo. Nas configurações em Grid, cada “organização virtual” faz o gerenciamento de seus recursos não tendo a visão de uma imagem única do sistema. Ou seja, o usuário tem consciência dos diversos serviços disponíveis e que deverá requisitá-los para sua utilização. Portanto, os Grids são mais heterogêneos, complexos e distribuídos. Para o Grid Computing, as aplicações leves são as ideais pois estas requisitam relativamente menos das redes. Mesmo tendo máquinas com uma alta conexão, estas redes com baixo fluxo de dados constitui uma espécie de gargalo ao requisito fundamental para aplicações pesadas. A multiplicidade das velocidades das diversas redes implica em alguns pontos de gargalo, e que compromete a performance do nosso supercomputador virtual.

Para sacramentarmos Grid x Cluster, utilizaremos a referência do Prof Buyya. “Se acontece o compartilhamento de recursos gerenciado por um único sistema global sincronizado e centralizado, então é um cluster. Em um cluster, todos os nós trabalham cooperativamente em um objetivo comum e o objetivo é a alocação de recursos executada por um gerente centralizado e global. Em Grid, cada nó, possuí seu próprio gerente recursos e política de alocação".

**Benefícios de um Grid**

1. Organizações podem agregar recursos - a computação em Grid permite que as organizações possam agregar recursos com toda a infraestrutura dos ITs, não importando localização global. Isso elimina situações onde um site esteja sendo executado com sua capacidade máxima, enquanto outros tenham ciclos disponíveis.
2. Poderosa plataforma de suporte a Organizações Virtuais - organizações podem melhorar dramaticamente sua qualidade e velocidade de produtos e serviços disponibilizados, enquanto os custos de IT são reduzidos por habilitar a colaboração transparente dos recursos compartilhados
3. Acesso distribuído a diversos tipos de recursos - permite que empresas acessem e compartilhem bases de dados de forma remota. Isto é essencialmente benéfico para as ciências da saúde ou comunidades de pesquisa, onde volumes grandiosos de dados são gerados e analisados durante todo dia.
4. Colaboração entre centro de pesquisas - possibilita a larga dispersão das organizações para que facilmente possam colaborar em projetos pela criação da habilidade do compartilhamento de tudo, desde aplicações a dados, até projetos de engenharia, etc.
5. Melhor utilização de largura de banda - pode-se criar a mais robusta e resistente infraestrutura de informações.
6. Aproveitamento de recursos ociosos – pode-se aproveitar os ciclos de processamento idle disponíveis dos PCs desktops que se encontram em várias localidades pelo planeta. Por exemplo, os computadores que se encontram tipicamente ociosos durante a noite de uma empresa em Tókio pode ser utilizado durante o dia para operações na América do Sul.

Desafios operacionais e de pesquisa a serem vencidos

- Localização dos recursos;
- Reserva de recursos;
- Capacidade para adaptar- se a mudanças no ambiente;
- Criação e escalonamento das tarefas;
- Autonomia de cada grupo participante para definir suas próprias políticas de segurança;
- Recursos requisitados podem estar em diferentes localidades;
- Qualidade de serviço exigida por cada aplicação.

Para se utilizar um sistema em Grid são necessários diversos protocolos, padrões e ferramentas de software. Para desenvolver padrões comuns de processamento a Universidade do Sul da Califórnia, o Laboratório Nacional de Argonne e a Universidade de Chicago através do Projeto Globus (https://www.globus.org/) estão desenvolvendo um conjunto de especificações técnicas e ferramentas de software para Grid Computing, comandados pelo pesquisador Ian Forster. A padronização da tecnologia de Grid já vem ocorrendo a bastante tempo, sendo que, devido ao uso de diversas empresas, os esforços passaram a ter uma prioridade muito maior nesses últimos três anos.

## EXERCÍCIOS

1

Qual(is) a(s) opção(ções) que descreve(m) a computação em grade? Assinale a alternativa correta:
I - A computação distribuída passa a ser uma “Computação em Grade” no momento em que existe uma infra-estrutura física e uma infra-estrutura lógica (software) que permita coordenar os trabalhos que vão ser processados e garantir a sua qualidade de serviço.
II - O Grid é um caso particular da computação distribuída, uma vez que os Grids são orientados essencialmente para aplicações que precisam de uma grande capacidade de cálculos, ou enormes quantidades de dados transmitidos de um lado para o outro, ou as duas.
III - O nome Grid foi idealizado baseado nas malhas de interligação dos sistemas de energia elétrica (power-grids), em que um usuário utiliza a eletricidade sem ao menos saber em que local ele foi gerada, sendo totalmente transparente aos seus usuários.
IV - O surgimento das Grids Computacionais nasceu da comunidade de Processamento de Alto Desempenho (PAD).

- A)
  Somente I esta correta.
- B)
  Somente I e II estão corretas.
- C)
  Somente I, II e III estão corretas.
- D)
  Todas estão corretas.
- E)
  Nenhuma esta correta.

2

A afirmação "Permite que as organizações possam agregar recursos com toda a infraestrutura dos ITs, não importando localização global; isso elimina situações onde um site esteja sendo executado com sua capacidade máxima, enquanto outros tenham ciclos disponíveis." é um beneficio de qual estrutura?

- A)
  Grid.
- B)
  Cluster.
- C)
  Cluster de Alto Desempenho.
- D)
  Cluster de Alta Disponibilidade.
- E)
  Cluster de Balanceamento de Carga.

3

Quais são os desafios operacionais que o Grid precisa equacionar?
I - Localização dos recursos;
II - Reserva de recursos;
III - Capacidade para adaptar-se a mudanças no ambiente;
IV - Capacidade computacional de cada nó do sistema.

- A)
  Somente a I é verdadeira.
- B)
  Somente a I e a II são verdadeiras.
- C)
  Somente a I, II e III são verdadeiras.
- D)
  Todas são verdadeiras.
- E)
  Nenhuma é verdadeira.

4

Segunda Ian Foster, quais são as aracteristicas básica de um Grdi:
I - Recursos coordenados que não se sujeitam a um controle centralizado: sistemas em Grid podem englobar recursos entre os mais variados tipos, desde o desktop de um usuário até um supercomputador; pode haver um controle local em uma empresa, mas não existe um controle central para todo o grid.
II - Utilizar padrões abertos, interfaces e protocolos de propósito geral: a utilização de protocolos e padrões abertos é essencial para que os sistemas em Grid possam realizar funções fundamentais como autenticação,

autorização, descobrimento de recursos e acesso a eles, sem perder a capacidade de escalar e interagir com diferentes plataformas de hardware e software.
III - Prover o mínimo em qualidade de serviços: como segurança, tempo de resposta e disponibilidade.
IV - Prover recursos coordenados centralizados: como processos e threads.

- A)
  Somente a I e II são verdadeiras.
- B)
  Somente II e III são verdadeiras.
- C)
  Somente III e IV são verdadeiras.
- D)
  Todas são falsas.
- E)
  Todas são verdadeiras, com exceção da IV.

5

Quais são as maneiras de explorar os recursos computacionais em um Grid?
I - Executar uma aplicação em qualquer máquina disponível do Grid, independentemente de onde esteja localizada;
II - Quebrar o aplicativo em partes menores para que estas possam ser executadas paralelamente através do Grid;
III - Executar uma tarefa que precisa rodar várias vezes em uma única máquina do Grid.
IV - Executar uma tarefa que precisa rodar várias vezes em diferentes máquinas do Grid.

- A)
  Somente I e II estão corretas.
- B)
  Somente II e III estão corretas.
- C)
  Somente I, III e IV estão corretas.
- D)
  Somente I, II e IV estão corretas.
- E)
  Todas estão corretas.

1D 2A 3C 4E 5D

# Módulo Complementar I

## EXERCÍCIOS

1

POSCOMP 2003

Um sistema centralizado é um concentrador de recursos; um sistema distribuído apresenta seus recursos dispersos. Entretanto nem todo o conjunto de recursos computacionais dispersos pode ser considerado um sistema distribuído. Considerando um conjunto de computadores, assinale a alternativa que melhor corresponde às características necessárias para considerá-lo um sistema distribuído:

- A)
  Existência de memória compartilhada e relógios locais sincronizados.
- B)
  Suporte de rede e funções primitivas de comunicação.
- C)
  Suporte de rede e um relógio global.
- D)
  Existência de sistema operacional idêntico e hardware padronizado em todos os computadores.
- E)
  Existência de memória secundária compartilhada e protocolos de sincronização de estado.

2

COMPANHIA DO METROPOLITANO DE SÃO PAULO - METRÔ

Analista Trainee Ciências da Computação – 2008

Em relação a sistemas distribuídos, considere:

I. Visando a otimização do maior número possível de máquinas e tipos de computadores, o protocolo ou canal de comunicação pode ser dotado de um interpretador para compatibilizar a informação que possa não ser entendida por certas máquinas.

II. O Middleware é composto por um conjunto de processos ou objetos num grupo de computadores, que interagem entre si de forma a implementar comunicação e oferecer suporte para partilha de recursos a aplicações distribuídas.

III. Um sistema distribuído pode ser definido pela capacidade que um único processador tem para distribuir várias tarefas simultaneamente.

IV. Um sistema distribuído pode ser definido como uma coleção de processadores fracamente acoplados, que não compartilham memória nem relógio, interconectados por uma rede de comunicação.

É correto o que consta APENAS em:

- A)
  Somente I e III estão corretas.
- B)
  Somente I e IV estão corretas.
- C)
  Somente II e IV estão corretas.
- D)
  Somente II e III estão corretas.
- E)
  Somente I e II estão corretas.

3

Universidade Federal Fronteira Sul

Considere o seguinte conjunto de informações a respeito de conceitos de segurança em sistemas distribuídos:

1 - A assinatura digital garante as propriedades de autenticidade, integridade e confidencialidade;

2 - Uma assinatura digital é gerada usando a chave pública para cifrar o resumo digital (hash) do documento a ser assinado;

3 - O protocolo SSL usa criptografia de chave assimétrica para estabelecer uma sessão de comunicação com a criptografia de chave simétrica;

4 - A propriedade de confidencialidade garante que os usuários do sistema só podems ler informações para as quais esteham autorizados.

Assinale a alternativa correta:

- A)
  Somente a alternativa 1 é correta.
- B)
  Somente as alternativas 1 e 2 são corretas.
- C)
  Somente as alternativas 3 e 4 são corretas.
- D)
  Somente as alternativas 1, 3 e 4 são corretas.

- E)
Somente as alternativas 2, 3 e 4 são corretas.

4

Ciências Exatas e da Terra - Ciência da Computação/Redes de Computadores e Sistemas Dist - IF/PR
Assinale a alternativa que não apresenta vínculo ou ligação com a arquitetura Middelware.

- A)
Servidor Web.
- B)
SGBD (Sistema de Gerenciador de Banco de Dados).
- C)
Servidor de Banco de Dados.
- D)
Plataforma .Net.
- E)
Internet.

5

Ciências Exatas e da Terra - Ciência da Computação/Redes de Computadores e Sistemas Dist - IF/PR
Assinale a alternativa que apresenta um elemento que não contém uma característica de um sistema distribuído.

- A)
Escalabilidade.
- B)
Concorrência.
- C)
Portabilidade.
- D)
Imprevisibilidade.
- E)
Complexidade.

6

Ciências Exatas e da Terra - Ciência da Computação/Redes de Computadores e Sistemas Dist - IF/PR
A RPC (Chamada de Procedimento Remota) são portas de comunicação entre os processos. No ambiente de rede, o nome mais simples para isso é, simplesmente, portas. Assinale a alternativa que apresenta o range de portas conhecidas, normatizado atualmente.

- A)
0 até 1.000.
- B)
1 até 2.048.
- C)
0 até 64.
- D)
0 até 1.024.
- E)
1 até 65.535.

7

Ciências Exatas e da Terra - Ciência da Computação/Redes de Computadores e Sistemas Dist - IF/PR
O paralelismo é usado, hoje em dia, em locais onde é necessário usar computadores de alto desempenho. Assinale a alternativa que não apresenta uma evolução natural do paralelismo.

- A)
Virtualização.
- B)
Trunking.
- C)
Processadores Dual Core.
- D)
Cluster.
- E)
Cloud.

8

UEPB 2012

Dadas as opções seguintes, assinale aquela improcedente quanto aos desafios de segurança na Internet.

- A)
  A arquitetura de uma aplicação baseada na Web inclui, geralmente, um cliente Web, um servidor e sistemas seguros de informação ligados a bancos de dados.
- B)
  Tanto o comércio como os negócios eletrônicos exigem que as empresas sejam, ao mesmo tempo, mais abertas e mais fechadas.
- C)
  O firewall impede que comunicações não autorizadas entrem ou saiam da rede, permitindo que organização imponha uma política de segurança ao fluxo de tráfego entre sua rede e a Internet.
- D)
  Computadores permanentemente conectados a internet via cabo DSL estão mais sujeitos à invasão por estranhos.
- E)
  Para lidar efetivamente com a segurança na Internet, às vezes são necessárias políticas e procedimentos corporativos, responsabilidades de usuários e treinamento em conscientização de segurança mais abrangentes.

1B 2C 3C 4B 5C 6E 7B 8A

# Módulo Complementar II

# EXERCÍCIOS

1

**POSCOMP 2011**
**Questão 29**
**Relacione a coluna da esquerda com a coluna da direita.**
**(I) Multicore** **(A) Múltiplos pipelines que operam em paralelo.**
**(II) Superpipeline** **(B) Execução de instruções fora de ordem em um pipeline.**
**(III) Superescalar** **(C) Pipelines com grande número de estágios.**
**(IV) Pipeline dinâmico** **(D) Múltiplos processadores compartilhando um espaço de endereços.**
**(V) Multiprocessadores** **(E) Múltiplos processadores em um único encapsulamento.**
**Assinale a alternativa que contém a associação correta.**

- A)
  I-B, II-A, III-C, IV-E, V-D.
- B)
  I-C, II-A, III-B, IV-D, V-E.
- C)
  I-D, II-E, III-B, IV-A, V-C.
- D)
  I-E, II-C, III-A, IV-B, V-D.
- E)
  I-E, II-C, III-A, IV-D, V-B.

1D

# Módulo Complementar III

## EXERCÍCIOS

1

Enade 2008 - questão 19

Uma alternativa para o aumento de desempenho de sistemas computacionais é o uso de processadores com múltiplos núcleos, chamados multicores. Nesses sistemas, cada núcleo, normalmente, tem as funcionalidades completas de um processador, já sendo comuns, atualmente, configurações com 2, 4 ou mais núcleos. Com relação ao uso de processadores multicores, e sabendo que threads são estruturas de execução associadas a um processo, que compartilham suas áreas de código e dados, mas mantêm contextos independentes, analise as seguintes asserções.

Ao dividirem suas atividades em múltiplas threads que podem ser executadas paralelamente, aplicações podem se beneficiar mais efetivamente dos diversos núcleos dos processadores multicores porque o sistema operacional nos processadores multicores pode alocar os núcleos existentes para executar simultaneamente diversas seqüências de código, sobrepondo suas execuções e, normalmente, reduzindo o tempo de resposta das aplicações às quais estão associadas.

Acerca dessas asserções, assinale a opção correta.

- A)
  As duas asserções são proposições verdadeiras, e a segunda é uma justificativa correta da primeira.
- B)
  As duas asserções são proposições verdadeiras, mas a segunda não é uma justificativa correta da primeira.
- C)
  A primeira asserção é uma proposição verdadeira, e a segunda, uma proposição falsa.
- D)
  A primeira asserção é uma proposição falsa, e a segunda, uma proposição verdadeira.
- E)
  Tanto a primeira quanto a segunda asserções são proposições falsas.

1A

# Módulo Complementar IV

## EXERCÍCIOS

1

POSCOMP 2007
Questão 70.
Em relação aos sistemas distribuídos, analise as seguintes afirmativas.
I. Um sistema assíncrono apresenta medida de tempo global.
II. A passagem de mensagens é o instrumento empregado para efetuar a comunicação entre os processos de um sistema assíncrono.
III. È possível simular um computador paralelo de memória compartilhada usando-se um sistema distribuído.
IV. Quando um determinado elemento de um sistema distribuído efetua a difusão de uma mensagem por meio de um multicast, todos os elementos do sistema distribuído recebem a mensagem.
A análise permite concluir que

- A)
  Somente a afirmativa IV está correta.
- B)
  Somente as afirmativas I e II estão corretas.
- C)
  Somente as afirmativas I e III estão corretas.
- D)
  Somente as afirmativas II e III estão corretas.
- E)
  Somente as afirmativas I e IV estão corretas.

1D

# Módulo Complementar V

## EXERCÍCIOS

1

POSCOMP 2007
Questão 69.
Analise as seguintes afirmativas concernentes a questões de projeto de sistemas distribuídos.
I - Um sistema distribuído tolerante a falhas deve continuar operando na presença de problemas, podendo ocorrer uma degradação tanto no seu desempenho, como nas suas funcionalidades.
II - No que diz respeito à escalabilidade, o projeto de um sistema distribuído deve prever que a demanda nos serviços em qualquer dos equipamentos seja limitada por uma constante dependente do número de nodos envolvidos.
III - Em um sistema distribuído transparente quanto à concorrência, a informação de quantos usuários estão empregando determinado serviço deve ser omitida.
A análise permite concluir que:

- A)
  Somente a afirmativa I está incorreta.
- B)
  Somente a afirmativa II está incorreta.
- C)
  Somente a afirmativa III está incorreta.
- D)
  Somente as afirmativas I e III estão incorretas.
- E)
  Todas as afirmativas estão incorretas.

1B

# Módulo Complementar VI

## EXERCÍCIOS

1

Enade 2005
Questão 42.
Duas possibilidades para a construção de sistemas com múltiplos processadores são: processadores idênticos com um único espaço de endereçamento interligados por um barramento único (SMP); e máquinas monoprocessadas conectadas por uma rede (cluster). Com relação a esses sistemas, assinale a opção correta.

- A)
  A comunicação entre processadores de um cluster é, potencialmente, muito mais rápida que a comunicação entre processadores de um sistema SMP, pois redes atuais possuem taxa de transmissão da ordem de gigabits/s, enquanto as melhores memórias operam somente com freqüências da ordem de centenas de megahertz.
- B)
  Comunicação entre processos pode ser implementada de forma muito mais eficiente em um cluster que em um sistema SMP, pois, nesse último, todos os processos precisam compartilhar os mesmos dispositivos de entrada e saída.
- C)
  Em um sistema SMP, é mais simples substituir um processador defeituoso, pois, em um cluster, toda a rede de comunicação deve ser desabilitada para que a troca seja efetuada sem prejudicar a troca de mensagens entre os processos.
- D)
  Alocação de memória para processos é muito mais simples em um cluster, pois cada processador executa um único processo na sua memória exclusiva e, dessa forma, não existe o problema de distribuição de processos no espaço de endereçamento único da máquina SMP.
- E)
  Em um cluster, o custo da escalabilidade é muito menor, pois, para a interconexão entre as máquinas, podem ser utilizados equipamentos comuns usados em uma rede local de computadores, ao passo que um sistema SMP exige conexões extras no barramento e gabinetes especiais.

1E

# Módulo Complementar VII

## EXERCÍCIOS

1

POSCOMP 2004
Questão 65.
Considere dois sistemas computacionais formados por múltiplos computadores que manipulam dados comuns. No primeiro, existe um computador central, chamado servidor, que é o único responsável pelos serviços de leitura e escrita desses dados. No segundo, existe um grupo de computadores responsáveis pelos serviços de leitura e escrita, e cada um desses servidores deve manter uma réplica idêntica dos dados. Entre as vantagens e desvantagens de um único servidor em relação a vários servidores, podemos afirmar que o servidor único apresenta:
I - Vantagem de maior velocidade de execução das operações de escrita e leitura e desvantagem de maior custo de armazenamento
II - Vantagem de não necessitar de coordenação distribuída para ordenação das operações de escrita e desvantagem de menor velocidade nas leituras devido a serialização de operações
III - Vantagem de manter mais facilmente a consistência dos dados mas desvantagem de perda de disponibilidade em caso de falha do servidor.
Dessas afirmativas são verdadeiras:

- A)
  Somente I e III.
- B)
  Apenas II.
- C)
  Apenas I.
- D)
  Somente I e II.
- E)
  Somente I, II e III.

1A

# Módulo Complementar VIII

## EXERCÍCIOS

1

POSCOMP 2004
Questão 66.
Um cluster é definido como um sistema distribuído formado por máquinas homogêneas, executando o mesmo sistema operacional, interligadas por uma rede de alta velocidade. Como vantagens deste cluster em relação a uma máquina de grande porte poderíamos citar:
I - Melhor escalabilidade de unidades de processamento;
II - Possibilidade de executar programas com múltiplas threads;
III - Possibilidade de continuar o processamento isolando máquinas defeituosas;
IV - Possibilidade de executar programas Java usando RMI.
Dessas afirmativas são verdadeiras:

- A)
  Somente I e III são verdadeiras.
- B)
  Somente II e IV são verdadeiras.
- C)
  Somente I e IV são verdadeiras.
- D)
  Somente a III é verdadeira.
- E)
  Todas estão corretas.

1A

# Módulo 09: Período de suspensão de aulas - COVID-19