Sistemas Distribuídos

Conceitos Introdutórios

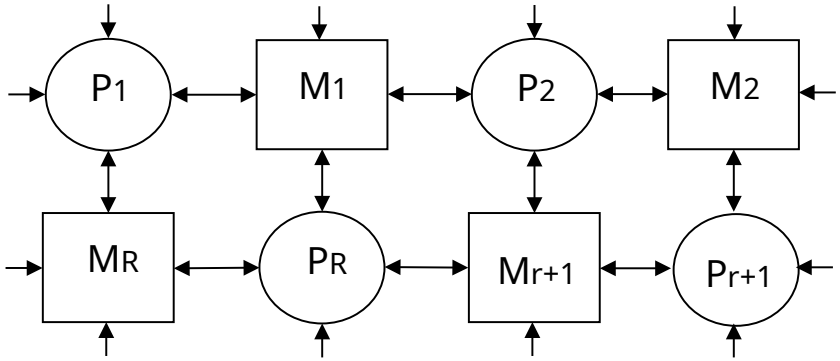
António Rui Borges

Resumo

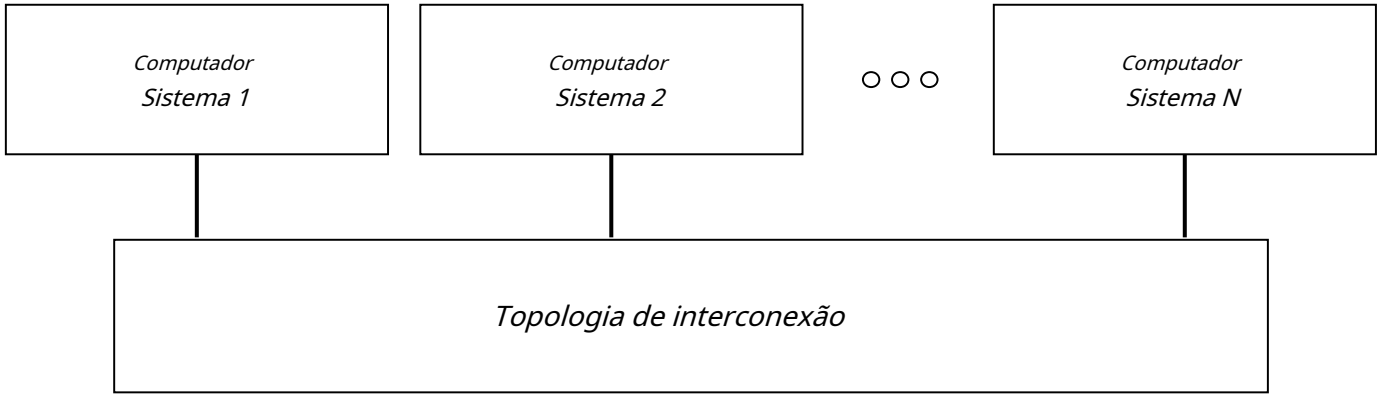
- *Sistemas de computadores paralelos*
- *Caracterização de sistemas distribuídos*
- *Questões a ter em conta no projeto de sistemas distribuídos*
  - *Heterogeneidade*
  - *Transparência*
  - *Abertura*
  - *Segurança dos fluxos de informação*
  - *Escalabilidade*
  - *Tratamento de falhas*
  - *Simultaneidade*
- *Computação ubíqua*(*Internet das Coisas*)
- *Computação em nuvem*
- *Leitura sugerida*

Sistemas de computadores paralelos

Multiprocessador de malha



Rede de computadores



Sistemas distribuídos - 1

A definição convencional de ***sistema distribuído*** afirma que se está lidando com *um* [*operativo*] *sistema* cujo *componentes, localizados nos diferentes nós de processamento de um sistema de computador paralelo, comunicam e coordenam suas ações através da passagem de mensagens* [Coulouris et al.].

A principal motivação subjacente à construção e utilização do *Sistemas distribuídos* é o compartilhamento de recursos. *Recurso* deve ser entendido aqui como uma entidade abstrata que incorpora algo *material*, como processadores, infraestruturas de rede, dispositivos de armazenamento e periféricos mais ou menos especializados, ou *imaterial*, como a informação captada no sentido mais amplo, ou as funcionalidades que sobre ela operam.

Esta partilha traduz-se de duas formas distintas

- *paralelização de aplicativos* – aproveitando os múltiplos processadores e outros *hardware* componentes do sistema computacional paralelo, tenta-se garantir uma execução mais rápida e eficiente do programa
- *serviço disponível* – ao gerenciar um conjunto de recursos de alguma forma relacionados, tenta-se organizá-los de tal forma que uma interface de comunicação uniforme baseada em uma API bem definida possa ser fornecida.

Sistemas distribuídos - 2

Uma definição alternativa de ***sistema distribuído***, embora essencialmente equivalente, é dizer que se trata de *uma coleção de sistemas de computação independentes percebidos pelos usuários como um sistema coerente* [Tanenbaum et al.].

Várias consequências decorrem disso

- *paralelismo* – a existência de sistemas computacionais independentes dá origem a threads de execução simultâneos e autônomos
- *estado interno global* – a necessidade de fornecer uma imagem de trabalho coerente requer alguma forma de coordenação de atividades entre os diferentes nós de processamento
- *comunicação através da passagem de mensagens* – o facto de não serem feitas suposições sobre a forma como os sistemas informáticos estão interligados implica um mecanismo de comunicação minimalista baseado na passagem de mensagens
- *escalabilidade* – a faculdade de expandir o sistema pela integração de mais nós de processamento é um resultado imediato da organização prescrita
- *tratamento de falhas* – qualquer um dos componentes do sistema distribuído pode falhar a qualquer momento, deixando os demais componentes em operação.

Tipos de sistemas distribuídos - 1

O caminho *Sistemas distribuídos* estão organizados depende estritamente do tipo de problemas que pretendem resolver.

Quatro tipos de sistemas distribuídos estão entre os mais comuns na prática

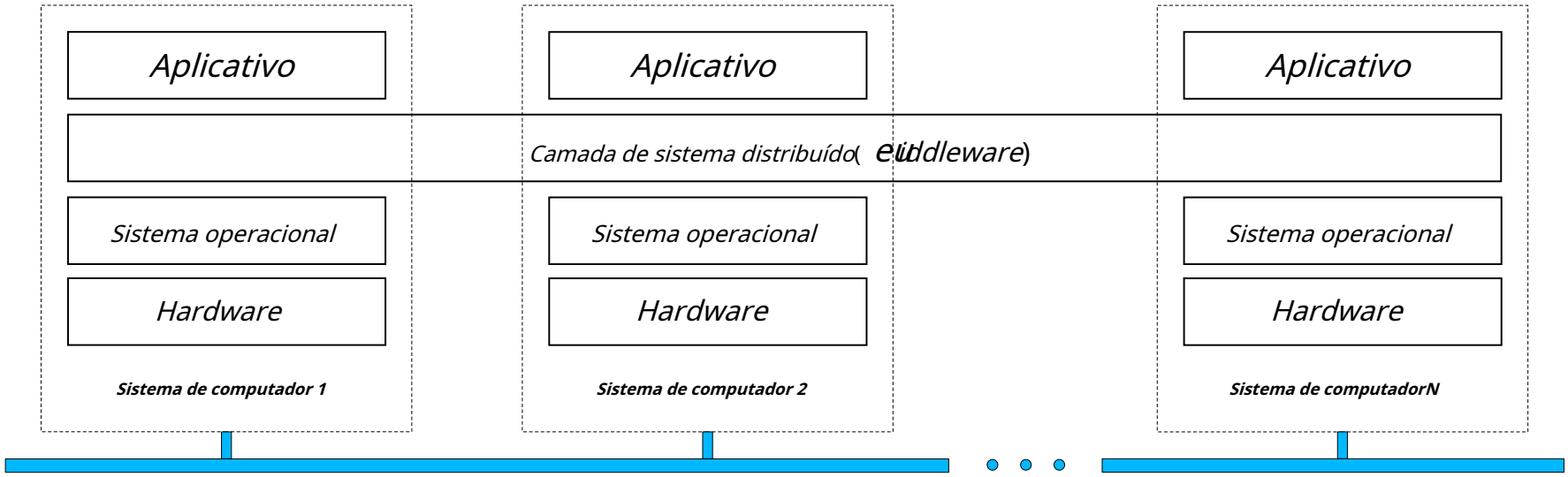
- *sistemas de cluster* – consistem tipicamente num conjunto de sistemas informáticos interligados por uma rede de alta velocidade; o objetivo é a paralelização de aplicações; normalmente, os sistemas de computador são idênticos, todos executam o mesmo sistema operacional e compartilham a mesma rede; existe um único nó mestre que trata da alocação de nós de processamento para um programa específico, mantém uma fila de trabalhos enviados e interfaces com os usuários do sistema
- *sistemas de rede* – não são feitas aqui suposições relativas às plataformas de hardware que formam os nós de processamento, nem aos sistemas operacionais que eles executam, nem às redes de interconexão; a questão fundamental é reunir os recursos de diferentes organizações para que uma comunidade de pessoas possa cooperar; forma-se assim uma organização virtual, onde apenas seus membros têm direitos de acesso aos recursos disponibilizados

Tipos de sistemas distribuídos - 2

- *sistemas de informação de transações*–os principais nós de processamento executam aplicações cujo objetivo principal é gerenciar informações majoritariamente organizadas em bancos de dados, chamados*servidores*; essas informações são disponibilizadas para programas executados remotamente, o *clientes*, através de uma sucessão de operações do tipo solicitação/resposta; A integração no nível mais baixo permite que os clientes reúnam um grupo de solicitações, possivelmente para servidores diferentes, em uma única solicitação maior e execute-a como um *transação*, ou seja, todas ou nenhuma das subsolicitações são executadas
- *sistemas de integração empresarial*–à medida que as aplicações se tornam dissociadas das bases de dados associadas, torna-se cada vez mais evidente que são necessários meios para a integração de aplicações independentes das bases de dados; em particular, os componentes da aplicação devem ser capazes de comunicar directamente entre si e não apenas através do comportamento de pedido/resposta suportado pelos sistemas de informação de transacções.

Sistemas distribuídos - 3

Assumindo sistemas informáticos genéricos, como nós de processamento, e redes de comunicação diversas, como infra-estruturas de interligação, mantendo simultaneamente uma visão integrada da sua funcionalidade, *Sistemas distribuídos* pode ser entendida como uma camada de software, localizada entre as aplicações e os sistemas operacionais locais, com o objetivo de criar um modelo de máquina virtual programável que esconda toda a heterogeneidade subjacente – a chamada *intermediário*.



Projeto de sistemas/aplicativos distribuídos

O projeto de *Sistemas distribuídos*, bem como as aplicações de desenvolvimento neles executadas, exige que sejam consideradas diversas questões cuja importância é crucial para o bom desempenho.

São destacados os seguintes

- o possível] *heterogeneidade* dos componentes que formam o sistema de computador paralelo
- isso é *transparência*
- o grau de *abertura*
- o *segurança* dos fluxos de informações trocadas e das informações armazenadas
- o potencial que apresenta para *escalabilidade*
- o *tratamento de falhas*
- seu potencial para *simultaneidade*.

Heterogeneidade - 1

O *heterogeneidade* de um sistema de computador paralelo torna-se aparente em múltiplas formas

- a interconexão de diferentes tipos de redes de computadores
- o uso de diferentes plataformas de hardware como nós de processamento
- diferentes sistemas operacionais executados simultaneamente em nós diferentes
- diferentes linguagens de programação usadas na codificação de diferentes componentes de software do mesmo aplicativo
- a interligação de funcionalidades disponibilizadas por diferentes fornecedores.

Face a uma realidade tão diversa, é obrigatório encontrar soluções que proporcionem uma abordagem coerente.

Heterogeneidade - 2

As soluções deverão ser *localizado*, introduzindo um maior ou menor grau de uniformidade em um nível específico da cadeia de comunicação

- especificação e implementação de uma arquitetura de rede
- conversão das especificidades de representação de dados apresentadas por cada processador em um formato comum
- harmonização do *interface de programação de comunicação* oferecido pelos diferentes sistemas operacionais
- ajuste das estruturas de dados implementadas por diferentes linguagens de programação em um formato comum
- desenvolvimento de padrões aceitos e cumpridos por diferentes fabricantes.

Transparência - 1

O grau de*transparência*de um sistema distribuído é uma característica que expressa o maior ou menor sucesso em mascarar a complexidade subjacente. Assim, tornar a operação de um sistema distribuído*transparente*significa descrever sua funcionalidade de forma integrada, conceitualmente simples, em vez de resultar da interação de um conjunto de componentes independentes.

O*transparência*O objetivo é, portanto, ocultar os recursos que não são diretamente relevantes para a tarefa que está sendo realizada, tornando-os anônimos tanto para o usuário quanto para o programador da aplicação.

O grau de*transparência*com os quais os diferentes recursos podem ser acessados e operados, retrata imediatamente o grau de abstração e operatividade do *intermediário*camada.

Transparência - 2

Modos de transparência

- *transparência de acesso*–quando as mesmas operações são realizadas para acessar recursos locais e remotos
- *transparência de posição*–quando o acesso a recursos é realizado sem o conhecimento de sua localização física ou de rede precisa
- *transparência da rede*–quando a transparência de acesso e posição existe concomitantemente
- *transparência da mobilidade*–quando o local de acesso do cliente aos recursos pode mudar dentro do sistema sem afetar a operação que está sendo executada
- *transparência da migração*–quando os recursos podem ser movidos sem afetar seu acesso
- *transparência de realocação*–quando os recursos podem ser movidos sem afetar o seu acesso, mesmo quando um acesso está ocorrendo (versão forte do *transparência da migração*)

Transparência - 3

- *transparência de replicação*–quando é possível instanciar múltiplas cópias do mesmo recurso sem que isso se torne óbvio
- *transparência de desempenho*–quando uma reconfiguração dinâmica do sistema pode ocorrer para lidar com variações de carga
- *escalando transparência*–quando o sistema e as aplicações podem expandir-se em escala sem exigir qualquer alteração na estrutura do sistema e nos algoritmos da aplicação
- *transparência de simultaneidade*–quando o acesso a recursos compartilhados é realizado em paralelo por múltiplas entidades sem conhecimento umas das outras (os dados devem sempre permanecer consistentes)
- *transparência de falha*–quando falhas ocorridas nos componentes de hardware e/ou software do sistema podem ser mascaradas e, portanto, as tarefas em execução podem ser encerradas.

Abertura

O grau de *abertura* de um sistema computacional é uma característica que determina sua maior ou menor capacidade de ser ampliado e reimplementado de diferentes maneiras.

Para *Sistemas distribuídos*, *abertura* está fundamentalmente relacionada com a capacidade de incorporar novos serviços e de os disponibilizar para uma ampla variedade de aplicações.

É, portanto, necessário

- estabelecer um mecanismo de comunicação uniforme sobre acesso a recursos compartilhados
- publicar as principais APIs
- garantir uma estrita conformidade, ao nível da concepção e implementação, de cada novo componente com a norma relevante.

Segurança da informação - 1

A disponibilidade generalizada de serviços e a apetência demonstrada pelos sistemas informáticos paralelos para a execução cooperativa de aplicações levantam a questão de quão seguros são os fluxos de informação trocados e de quão seguros são os dados armazenados nos recursos associados.

O *segurança da informação* pode ser previsto em três níveis

- *confidencialidade* – proteção contra a sua divulgação a entidades não autorizadas
- *integridade* – proteção contra sua modificação ou sua corrupção
- *disponibilidade* – proteção contra interferências que perturbem os canais de acesso.

Segurança da informação - 2

Algumas medidas podem ser tomadas para reduzir os riscos

- *introdução de firewalls*–criando uma barreira em torno de uma rede local que limita o tráfego de entrada e saída a canais bem definidos
- *criptografia de mensagens*–embaralhando o conteúdo dos fluxos de informação
- *uso de assinaturas eletrônicas*–para certificação do autor da mensagem.

Existem, no entanto, situações de difícil solução

- *ataques de negação de serviço* -quando os prestadores de serviços são inundados por um grande número de solicitações falsas, o que leva ao encerramento do serviço, impedindo o acesso dos verdadeiros usuários
- *segurança do código móvel*–arquivos executáveis enviados em anexo às mensagens, download de*plug-ins*e*miniaplicativos*, por exemplo, pode levar a um comportamento indesejável do sistema, completamente fora do controle do usuário local.

Escalabilidade

Sistemas distribuídos deve permitir *aumentos de escala*. Ou seja, face a um aumento significativo do número e/ou da dimensão dos recursos disponíveis e/ou do número de utilizadores que os solicitam, deverá ainda manter um desempenho adequado.

Alguns dos desafios que devemos enfrentar são

- *expansão dos recursos físicos* – aumento de escala significa, neste sentido, que a quantidade de recursos necessários para atender *n* os usuários não devem ser maiores que $Ó(n)$
- *perdas de desempenho* – aumento de escala significa, neste sentido, que a perda de desempenho resultante do processamento de uma tarefa deve ser de cerca de $Ó(registro_2 n)$ no

  a maioria, onde *n* é algum parâmetro que mede o tamanho da tarefa
- *evitando gargalos futuros* – o aumento de escala significa, neste sentido, que deve existir um planeamento eficaz sobre a dimensão dos recursos e os procedimentos de acesso para que, tendo em conta o previsível crescimento futuro, evite o seu esgotamento e as consequentes perdas de desempenho.

Tratamento de falhas - 1

Sistemas distribuídos, sendo construídos a partir de vários componentes de hardware e software, são suscetíveis a falhas parciais de tipos extremamente variados. Lidar com isso é, portanto, muito difícil.

Alguns tipos de falhas são facilmente detectáveis, mas muitos outros são mascarados pela complexidade do sistema. O grande desafio é como gerir o sistema num ambiente onde algumas das falhas não podem ser detectadas, mas cuja existência se pode apenas suspeitar.

Estratégias básicas para tratamento de falhas

- *mascarando as falhas*–algumas das falhas detectadas podem ser ocultadas ou, pelo menos, seu efeito pode ser menos severo
  - mensagens perdidas, se detectadas, podem ser transmitidas novamente em muitos casos
  - a replicação de recursos pode permitir a proteção das informações armazenadas quando há corrupção de dados em algumas das cópias

Tratamento de falhas - 2

Estratégias básicas para tratamento de falhas (continuação)

- *se recuperando de falhas*–para que isso seja possível é necessário projetar os componentes de software de forma que os estados internos dos processos envolvidos sejam armazenados periodicamente; quando ocorre uma falha, o processamento é reiniciado a partir do último estado salvo

na prática, isso também pode exigir a migração de código para outros recursos de hardware e o uso de cópias atualizadas dos dados

- *tolerando falhas*–algumas falhas têm de ser toleradas; qualquer tentativa de resolvê-los é inútil ou impraticável; nestes casos, em vez de deixar o usuário perdido, enquanto são realizadas sucessivas tentativas de acesso, é melhor avisá-lo do fato.

Simultaneidade

Sendo a partilha de recursos a principal motivação que leva à concepção e implementação de *Sistemas distribuídos*, é fundamental garantir que a entidade ou entidades que gerem o acesso ao recurso partilhado imponham exclusão mútua para que a operação seja realizada de forma consistente.

Quando o controle de acesso é *centralizado*, ou seja, dependente de uma única entidade, existe uma solução simples baseada em dispositivos padrão para imposição de exclusão mútua e sincronização, desenvolvida para ambientes de multiprogramação em monoprocessadores.

Quando, no entanto, o controle de acesso ao recurso é *distribuído*, dependente de várias entidades independentes, a solução é muito mais complexa porque é necessário primeiro abordar o problema de como sincronizar entidades distintas não sujeitas a um relógio global.

Computação ubíqua (*Internet das Coisas*)

Princípios

- *para tornar a comunicação natural* – fornecer dispositivos de interface inteligentes entre sistemas de computador e seres humanos, ou outros sistemas de computador, a fim de tornar espontânea a troca de informações
- *para tornar a computação sensível ao contexto* – para selecionar as ações a serem tomadas que são específicas para a interpretação do cenário atual.

Áreas de aplicação importantes

- *casas inteligentes* (*domótica* ou *automação residencial*) – instalação de eletrodomésticos, controle e monitoramento remoto, estabelecimento de procedimentos de vigilância, etc.
- *veículos autônomos* – comunicação veículo a veículo, aplicação de regras de trânsito dependentes do cenário, etc.

Princípios

- *recursos sob demanda*– disponibilizar aos usuários potenciais diferentes tipos de serviços
  - aplicação específica (*Programas como um serviço*)
  - meios de hardware específicos para computação (*plataforma como serviço*)
  - recursos específicos, como armazenamento em massa ou ambientes de execução personalizados (*Infraestrutura como um serviço*)
- *acesso indireto*–os recursos são acessados remotamente por meio de protocolos padrão executados a partir de sistemas de computadores locais, tablets ou smartphones
- *escalabilidade*–o sistema deve ter a capacidade de se adaptar às necessidades dos utilizadores, criando a ilusão de que os recursos são ilimitados.

Leitura sugerida

- *Sistemas Distribuídos: C uma vez* *n, 4ªEdi* *ção, Coulouris*, Dollimore, Kindberg, Addison-Wesley
  - Capítulo 1:*Caracterização de sistemas distribuídos*
- *Sistemas Distribuídos: Princípios e Paradigmas, 2ªEdição,*Tanenbaum, van Steen, Pearson Education Inc.
  - Capítulo 1:*Introdução* *nd Design*