

# Sistemas Distribuídos - ESP625

Prof<sup>a</sup> Ana Carolina Sokolonski

Bacharelado em Sistemas de Informação  
Instituto Federal de Ciência e Tecnologia da Bahia  
Campus de Feira de Santana

*carolsoko@ifba.edu.br*

July 18, 2024

# Conceitos Fundamentais de Sistemas Distribuídos

- 1 O que é um Sistema Distribuído?
- 2 Middleware
- 3 Concorrência em Sistemas Distribuídos
- 4 Inexistência de Relógio Global
- 5 Tolerância a Falhas
- 6 Desempenho
- 7 Confiabilidade e Maior Disponibilidade do Sistema
- 8 Transparência em Sistemas Distribuídos
- 9 Desafios
- 10 Referências

# O que é um Sistema Distribuído?

# O que é um Sistema Distribuído?

## Definição:

Os Sistemas Distribuídos surgiram a partir de uma premissa forte da computação presente em diversas áreas da computação:

# O que é um Sistema Distribuído?

## Definição:

Os Sistemas Distribuídos surgiram a partir de uma premissa forte da computação presente em diversas áreas da computação:

**DIVIDIR PARA CONQUISTAR**

# O que é um Sistema Distribuído?

## Definição:

Os Sistemas Distribuídos surgiram a partir de uma premissa forte da computação presente em diversas áreas da computação:

## DIVIDIR PARA CONQUISTAR

Basicamente, existiam diversas demandas computacionais muito grandes, que precisavam de alto poder computacional, e como solução, os cientistas computacionais idealizaram um sistema que compartilhava o recurso obsoleto de todos os computadores que estivessem disponíveis numa rede de computadores. Solucionando sua escassez de recursos. **Paralelismo e Tolerância a falhas** são duas propriedades fundamentais dos SDs.

# O que é um Sistema Distribuído?

## Definição:

Um Sistema Distribuído (SD) é um sistema computacional composto de vários computadores se comunicando através de uma rede de computadores, onde computadores abrigam conjuntos de processos que se comunicam através de mensagens que obedecem um conjunto de protocolos distribuídos com o intuito de prover a execução coerente das atividades distribuídas.

# O que é um Sistema Distribuído?

## Definição:

Um Sistema Distribuído (SD) é um sistema computacional composto de vários computadores se comunicando através de uma rede de computadores, onde computadores abrigam conjuntos de processos que se comunicam através de mensagens que obedecem um conjunto de protocolos distribuídos com o intuito de prover a execução coerente das atividades distribuídas.

Para o usuário do Sistema Distribuído, essa condição de "distribuição" deve ser transparente, ou seja, o Sistema Distribuído deve se apresentar ao usuário como um sistema único e coerente. Não deixando que o usuário perceba todo o seu aparato tecnológico. O fato do sistema ser distribuído, ou não, deve ser transparente ao usuário.



# O que é um Sistema Distribuído?

## Definição:

Um SD consiste em vários computadores autônomos que se comunicam, coordenam suas ações e colaboram entre si para executar processamentos diversos, levando à concorrência de componentes, falta de um relógio global e falha de componentes independentes. [Coulouris et al. 2013]

# O que é um Sistema Distribuído?

## Definição:

Um SD consiste em vários computadores autônomos que se comunicam, coordenam suas ações e colaboram entre si para executar processamentos diversos, levando à concorrência de componentes, falta de um relógio global e falha de componentes independentes. [Coulouris et al. 2013]

Essa condição de colaboração transparente ao usuário e compartilhamento de recursos possibilita que o sistema usufrua de um alto poder de processamento, muitas vezes espalhado pelo globo.

# O que é um Sistema Distribuído?

## Definição:

Um SD consiste em vários computadores autônomos que se comunicam, coordenam suas ações e colaboram entre si para executar processamentos diversos, levando à concorrência de componentes, falta de um relógio global e falha de componentes independentes. [Coulouris et al. 2013]

Essa condição de colaboração transparente ao usuário e compartilhamento de recursos possibilita que o sistema usufrua de um alto poder de processamento, muitas vezes espalhado pelo globo.

Usurfluindo de paralelismo, tolerância a falhas, replicação e transparência de localização.

# O que é um Sistema Distribuído?

## TROCANDO EM MIÚDOS

Coleção de sistemas computacionais (*software* ou *hardware*), independentes, mas com alguma forma de comunicação, que colaboram na execução de alguma tarefa.

# Tarefas executadas por Sistemas Distribuídos

## Tarefas executadas por Sistemas Distribuídos:

Vejamos alguns exemplos de tarefas executadas por Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

# Tarefas executadas por Sistemas Distribuídos

## Tarefas executadas por Sistemas Distribuídos:

Vejamos alguns exemplos de tarefas executadas por Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 Entregue este email para fulano@knowhere.uni.

# Tarefas executadas por Sistemas Distribuídos

## Tarefas executadas por Sistemas Distribuídos:

Vejam os alguns exemplos de tarefas executadas por Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 Entregue este email para fulano@knowhere.uni.
- 2 Envie o item I para o endereço E, após cobrança de X reais da conta C.

# Tarefas executadas por Sistemas Distribuídos

## Tarefas executadas por Sistemas Distribuídos:

Vejamos alguns exemplos de tarefas executadas por Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 Entregue este email para fulano@knowhere.uni.
- 2 Envie o item I para o endereço E, após cobrança de X reais da conta C.
- 3 Autorize a transferência de D reais da conta C para a conta C'.



# Tarefas executadas por Sistemas Distribuídos

## Tarefas executadas por Sistemas Distribuídos:

Vejam os alguns exemplos de tarefas executadas por Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 Entregue este email para fulano@knowhere.uni.
- 2 Envie o item I para o endereço E, após cobrança de X reais da conta C.
- 3 Autorize a transferência de D reais da conta C para a conta C'.
- 4 Movimente o braço mecânico que está segurando um bisturi, 3cm à direita, então abaixe-o 3mm, e movimente-o 4cm à esquerda.

# Tarefas executadas por Sistemas Distribuídos

## Tarefas executadas por Sistemas Distribuídos:

Vejam os alguns exemplos de tarefas executadas por Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 Entregue este email para fulano@knowhere.uni.
- 2 Envie o item I para o endereço E, após cobrança de X reais da conta C.
- 3 Autorize a transferência de D reais da conta C para a conta C'.
- 4 Movimente o braço mecânico que está segurando um bisturi, 3cm à direita, então abaixe-o 3mm, e movimente-o 4cm à esquerda.
- 5 Inclua o comentário "LOL!!!" na lista de comentários do item XYZ, com marca de tempo T.

# Tarefas executadas por Sistemas Distribuídos

## Exemplos de Sistemas Distribuídos:

Vejamos alguns exemplos de Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

# Tarefas executadas por Sistemas Distribuídos

## Exemplos de Sistemas Distribuídos:

Vejamos alguns exemplos de Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 A internet é o maior exemplo de sistema distribuído.

# Tarefas executadas por Sistemas Distribuídos

## Exemplos de Sistemas Distribuídos:

Vejamos alguns exemplos de Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 A internet é o maior exemplo de sistema distribuído.
- 2 Qualquer aplicação intranet.

# Tarefas executadas por Sistemas Distribuídos

## Exemplos de Sistemas Distribuídos:

Vejamos alguns exemplos de Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 A internet é o maior exemplo de sistema distribuído.
- 2 Qualquer aplicação intranet.
- 3 Qualquer aplicação mobile.

# Tarefas executadas por Sistemas Distribuídos

## Exemplos de Sistemas Distribuídos:

Vejamos alguns exemplos de Sistemas Distribuídos, que você usa em seu Dia-a-Dia:

- 1 A internet é o maior exemplo de sistema distribuído.
- 2 Qualquer aplicação intranet.
- 3 Qualquer aplicação mobile.
- 4 Aplicações e serviços baseados na Computação em Nuvem.

# Middleware



# O que é um Middleware?

## Middleware:

Como forma de contornar a heterogeneidade dos computadores e redes de computadores, o Sistema Distribuído, por vezes, se organiza como um **Middleware**, que nada mais é do que uma camada de *software* situada logicamente entre duas camadas: a camada de nível mais alto, composta de usuários e aplicações e a camada de nível mais baixo, composta dos Sistemas Operacionais e Sistemas de Comunicações.

# O que é um Middleware?

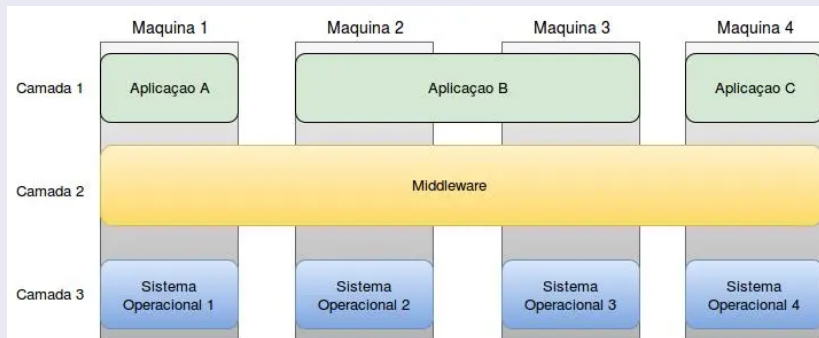
## Middleware:

Como forma de contornar a heterogeneidade dos computadores e redes de computadores, o Sistema Distribuído, por vezes, se organiza como um **Middleware**, que nada mais é do que uma camada de *software* situada logicamente entre duas camadas: a camada de nível mais alto, composta de usuários e aplicações e a camada de nível mais baixo, composta dos Sistemas Operacionais e Sistemas de Comunicações.

O *Middleware* é responsável por prover a comunicação correta entre os dispositivos heterogêneos. Normalmente, essa "conversa" se dá através de protocolos, como por exemplo, chamadas remotas (RMI).

# O que é um Middleware?

## Middleware:



# Concorrência em Sistemas Distribuídos

## Concorrência em Sistemas Distribuídos:

Quando pensamos em Sistemas Distribuídos, é natural que o façamos em termos do **paralelismo** inato que surge do uso de múltiplos processos executando ao mesmo tempo em (normalmente) diferentes computadores (hosts).

## Concorrência em Sistemas Distribuídos:

Quando pensamos em Sistemas Distribuídos, é natural que o façamos em termos do **paralelismo** inato que surge do uso de múltiplos processos executando ao mesmo tempo em (normalmente) diferentes computadores (hosts).

Contudo, é importante pensar também em termos de paralelismo dentro de cada um dos processos que compõem o sistema pois, no mínimo, componentes podem necessitar manter várias “conversas” em paralelo uns com os outros. Por isso, é “impossível” pensar em sistemas distribuídos sem pensar em **concorrência** na forma de múltiplos *threads* nos processos.

## Concorrência em Sistemas Distribuídos:

Em uma rede de computadores, a execução concorrente de programas é a norma. Posso fazer meu trabalho em meu computador, enquanto você faz o seu em seu computador, compartilhando recursos como *sites* ou arquivos do **Google Drive**, quando necessário.

## Concorrência em Sistemas Distribuídos:

Em uma rede de computadores, a execução concorrente de programas é a norma. Posso fazer meu trabalho em meu computador, enquanto você faz o seu em seu computador, compartilhando recursos como *sites* ou arquivos do **Google Drive**, quando necessário.

A capacidade do sistema de manipular recursos compartilhados pode ser ampliada pela adição de mais recursos (por exemplo, computadores) na rede. Essa capacidade extra pode ser distribuída em muitos pontos de maneira útil. A coordenação de programas em execução concorrente e que compartilham recursos também é um assunto importante e recorrente.



# Inexistência de Relógio Global

## Relógios:

Quando os programas precisam cooperar, eles coordenam suas ações trocando mensagens. A coordenação freqüentemente depende de uma noção compartilhada do tempo em que as ações dos programas ocorrem.

## Relógios:

Quando os programas precisam cooperar, eles coordenam suas ações trocando mensagens. A coordenação freqüentemente depende de uma noção compartilhada do tempo em que as ações dos programas ocorrem.

Entretanto, verifica-se que existem limites para a precisão com a qual os computadores podem sincronizar seus relógios em uma rede - não existe uma noção global única do tempo correto. Cada computador tem o seu próprio relógio interno e eles diferem entre si.

## Tempo:

Em SD, tempo é um problema importante! Por exemplo, precisamos que os computadores do mundo todo indiquem o **tempo exato e consistente**, de uma transferência bancária que ocorreu de um banco da China para um banco da Suíça, ou ainda, uma compra num comércio eletrônico chinês realizada por uma pessoa no Brasil.

## Tempo:

Em SD, tempo é um problema importante! Por exemplo, precisamos que os computadores do mundo todo indiquem o **tempo exato e consistente**, de uma transferência bancária que ocorreu de um banco da China para um banco da Suíça, ou ainda, uma compra num comércio eletrônico chinês realizada por uma pessoa no Brasil.

O tempo também é uma construção teórica importante para entender como o Sistema Distribuído funciona. Por isso, ao longo do tempo, os cientistas da computação estudaram e desenvolveram algoritmos para realizar sincronizações aproximadas de relógios.

# Tolerância a Falhas

## Definição:

A tolerância a falhas é a propriedade que garante a correta e eficiente operação de um sistema, apesar da ocorrência de falhas em qualquer um dos seus componentes. Um Sistema Distribuído tolerante a falhas garante o funcionamento do sistema e a entrega de mensagens mesmo que parte do sistema falhe.

## Definição:

A tolerância a falhas é a propriedade que garante a correta e eficiente operação de um sistema, apesar da ocorrência de falhas em qualquer um dos seus componentes. Um Sistema Distribuído tolerante a falhas garante o funcionamento do sistema e a entrega de mensagens mesmo que parte do sistema falhe.

Tolerância a Falhas é um dos principais desafios dos Sistemas Distribuídos modernos, pois tudo precisa estar funcionando. A vida *online* não admite mais que sistemas “caiam” ou “quebrem”. Não podemos perder dinheiro guardado num banco virtual ou perder arquivos salvos na nuvem. Tudo precisa funcionar perfeitamente. Então, tolerância a falhas é **imprescindível** em Sistemas Distribuídos.



## Falhas:

todos os sistemas computacionais podem falhar e é responsabilidade dos projetistas pensar e projetar o sistema para ser o mais tolerante a falhas possível.

## Falhas:

todos os sistemas computacionais podem falhar e é responsabilidade dos projetistas pensar e projetar o sistema para ser o mais tolerante a falhas possível.

Nos Sistemas Distribuídos, as falhas são diferentes. Falhas na rede resultam no isolamento dos computadores conectados, sem significar que eles parem de funcionar. Na verdade, seus programas, por vezes, nem conseguem detectar se a rede falhou ou se tornou muito lenta.

## Falhas:

todos os sistemas computacionais podem falhar e é responsabilidade dos projetistas pensar e projetar o sistema para ser o mais tolerante a falhas possível.

Nos Sistemas Distribuídos, as falhas são diferentes. Falhas na rede resultam no isolamento dos computadores conectados, sem significar que eles parem de funcionar. Na verdade, seus programas, por vezes, nem conseguem detectar se a rede falhou ou se tornou muito lenta.

A falha de um computador ou o término inesperado de um programa também não é imediatamente percebida pelos outros componentes. Cada componente do sistema pode falhar independentemente, deixando os outros ainda em funcionamento.

# Desempenho

## Desempenho:

A diminuição do tempo de execução da aplicação pode ser concluída através do uso do paralelismo existente em um sistema distribuído. Alguns programas terão seus tempos de execução diminuídos se partes desses programas forem executados em processadores diferentes ao mesmo tempo.

## Desempenho:

A diminuição do tempo de execução da aplicação pode ser concluída através do uso do paralelismo existente em um sistema distribuído. Alguns programas terão seus tempos de execução diminuídos se partes desses programas forem executados em processadores diferentes ao mesmo tempo.

Uma vez que o sistema não está implantado em uma única máquina mas em várias máquinas diferentes, pode-se dividir o processamento nessas várias máquinas e com isso aproveitar o paralelismo para concluir o processamento em um tempo menor do que seria feito em uma única máquina.

## Desempenho:

Exemplo de aplicações são a compilação paralela de módulos de um dado programa em máquinas diferentes, e a implementação de algoritmo de busca heurística. Qualquer processamento em paralelo traz benefícios.

## Desempenho:

Exemplo de aplicações são a compilação paralela de módulos de um dado programa em máquinas diferentes, e a implementação de algoritmo de busca heurística. Qualquer processamento em paralelo traz benefícios.

Não significa que não possa haver processamento paralelo em uma única máquina, mas com a distribuição, a carga de processamento é distribuída em máquinas que possam ter maior poder de processamento juntas do que uma única máquina.



# Confiabilidade e Maior Disponibilidade do Sistema

## Confiabilidade e Maior Disponibilidade do Sistema:

Sistemas distribuídos são potencialmente mais confiáveis, pois desde que os processadores são autônomos, uma falha em um não afeta o funcionamento correto dos demais. Temos o benefício da redundância não só de processadores mas de qualquer recurso que seja necessário.

## Confiabilidade e Maior Disponibilidade do Sistema:

Sistemas distribuídos são potencialmente mais confiáveis, pois desde que os processadores são autônomos, uma falha em um não afeta o funcionamento correto dos demais. Temos o benefício da redundância não só de processadores mas de qualquer recurso que seja necessário.

Portanto, a confiabilidade do sistema pode ser aumentada ao se replicar funções e/ou dados da aplicação nos vários processadores e/ou recursos disponíveis.

# Confiabilidade e Maior Disponibilidade

## Confiabilidade e Maior Disponibilidade:

Desse modo, se alguns elementos processadores falharem, os demais poderão continuar o serviço. Exemplos clássicos de aplicações Tolerantes a Falhas são: Controle de Aviões e Controle de Chão de Fábrica Automatizado.

# Confiabilidade e Maior Disponibilidade

## Confiabilidade e Maior Disponibilidade:

Desse modo, se alguns elementos processadores falharem, os demais poderão continuar o serviço. Exemplos clássicos de aplicações Tolerantes a Falhas são: Controle de Aviões e Controle de Chão de Fábrica Automatizado.

Em aplicações de missão crítica que não podem parar, sob risco de haver prejuízos sérios ou até perda de vidas é altamente recomendável o uso de uma arquitetura distribuída e Sistemas de Tempo-Real.

# Transparência em Sistemas Distribuídos

Transparência	Descrição
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 uma 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

# Transparência em Sistemas Distribuídos

## Transparência de Acesso:

Oculta a diferença na representação de dados e o modo como os recursos podem ser acessados pelos usuários. Por exemplo, quando estamos num nível mais básico, deseja-se ocultar diferenças na nomeação de arquivos [TANENBAUM e STEEN 2007].

# Transparência em Sistemas Distribuídos

## Transparência de Acesso:

Oculta a diferença na representação de dados e o modo como os recursos podem ser acessados pelos usuários. Por exemplo, quando estamos num nível mais básico, deseja-se ocultar diferenças na nomeação de arquivos [TANENBAUM e STEEN 2007].

## Transparência de Localização:

Refere-se a transparência da localização física dos recursos (componentes e arquivos) do sistema. A nomeação desempenha um papel importante na transparência de localização. Por exemplo, pode-se apenas atribuir nomes lógicos para os recursos, como URLs, que não dá pistas sobre a localização dos servidores, além de prover transparência de migração e relocação.



# Transparência em Sistemas Distribuídos

## Transparência de Migração:

Sistemas Distribuídos nos quais recursos podem ser movimentados sem afetar o modo como esses sistemas podem ser acessados, provêm transparência de migração.

# Transparência em Sistemas Distribuídos

## Transparência de Migração:

Sistemas Distribuídos nos quais recursos podem ser movimentados sem afetar o modo como esses sistemas podem ser acessados, provêm transparência de migração.

## Transparência de Relocação:

Sistemas Distribuídos que permitem que recursos possam ser relocados enquanto estão sendo acessados, sem que os usuários ou as aplicações percebam qualquer coisa provêm transparência de relocação. Um exemplo de transparência de relocação é o uso de *notebooks* móveis, onde o usuário se move, trocando de redes, sem ao menos perceber essa troca constante.

# Transparência em Sistemas Distribuídos

## Transparência de Replicação:

A **replicação** é um recurso muito importante e muito utilizado nos SD. A transparência de replicação é provida através da nomeação igual das réplicas e da substituição automática dos recursos em caso de falha. Para se ter transparência de replicação, obrigatoriamente, necessita-se de transparência de localização.

# Transparência em Sistemas Distribuídos

## Transparência de Concorrência:

Uma importante meta dos SD é prover compartilhamento de recursos. Para tanto, necessita-se permitir que os processos concorram por esses recursos compartilhados de forma transparente.

# Transparência em Sistemas Distribuídos

## Transparência de Concorrência:

Uma importante meta dos SD é prover compartilhamento de recursos. Para tanto, necessita-se permitir que os processos concorram por esses recursos compartilhados de forma transparente.

Eles devem compartilhar esses recursos sem saber que estão compartilhando e com quem estão compartilhando. Isto se chama transparência de concorrência. A transparência de concorrência tem que existir sem interferir no estado consistente do recurso.

# Transparência em Sistemas Distribuídos

## Transparência a Falha:

O Sistema Distribuído é Transparente a Falha, quando o usuário não percebe que um componente do sistema falhou, e possivelmente se recuperou, durante o tempo em que ele usou o sistema.

# Transparência em Sistemas Distribuídos

## Transparência a Falha:

O Sistema Distribuído é Transparente a Falha, quando o usuário não percebe que um componente do sistema falhou, e possivelmente se recuperou, durante o tempo em que ele usou o sistema.

A Tolerância a Falhas é um dos itens mais discutidos e desafiadores da implementação de Sistemas Distribuídos. Diversos aspectos são difíceis e até impossíveis de tratar, como por exemplo: como diferenciar um nó morto na rede de um incrivelmente lento?

# Desafios em Sistemas Distribuídos

Quais são os principais desafios a serem vencidos para desenvolvermos sistemas distribuídos robustos e confiáveis??





# Desafios em Sistemas Distribuídos

Quais são os principais desafios a serem vencidos para desenvolvermos sistemas distribuídos robustos e confiáveis??

- 1 Prover Tolerância a Falhas
- 2 Sincronização de Relógios
- 3 Tratar Concorrência
- 4 Prover Transparência ao usuário
- 5 Prover Confiabilidade e Disponibilidade do Sistema
- 6 Escalabilidade

# Referências

# Referências

-  COULOURIS, G. et al. *Sistemas Distribuídos: Conceitos e Projetos*. 5. ed. [S.l.]: Bookman, 2013. v. 1.
-  TANENBAUM, A.; STEEN, M. V. *Sistemas Distribuídos - Princípios e Paradigmas*. 2. ed. [S.l.]: Prentice Hall, 2007. v. 1.