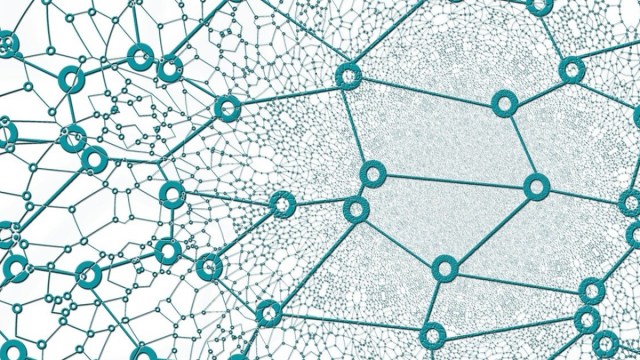


Licenciatura em Engenharia Multimédia

Computação na Nuvem

**A computação distribuída, princípios gerais do funcionamento**



Viorel Cojocari nº 21467

Vasco Sarrico nº 21459

Vanessa Brioso nº 21466

Professor Pedro Almeida

Turma Pós-Laboral

Ano letivo 2021 / 2022

Índice

[I. Introdução 3](#_Toc132108954)

[Definição de computação distribuída 4](#_Toc132108955)

[Importância da computação distribuída 5](#_Toc132108956)

[II. Arquiteturas de sistemas distribuídos 6](#_Toc132108957)

[Redes ponto a ponto 7](#_Toc132108958)

[Sistemas cliente-servidor 8](#_Toc132108959)

[Sistemas em cluster 9](#_Toc132108960)

[III. Compartilhamento de recursos em sistemas distribuídos 10](#_Toc132108961)

[Uso de vários computadores para processamento e armazenamento de dados 11](#_Toc132108962)

[Comunicação entre computadores 11](#_Toc132108963)

[IV. Segurança e confiabilidade em sistemas distribuídos 12](#_Toc132108964)

[Replicação de dados 13](#_Toc132108965)

[Deteção e tratamento de falhas 14](#_Toc132108966)

[V. Protocolos de comunicação em sistemas distribuídos 15](#_Toc132108967)

[Sincronização de Tarefas 16](#_Toc132108968)

[Comunicação entre computadores 18](#_Toc132108969)

[VI. Conclusão 20](#_Toc132108970)

[Benefícios da computação distribuída 21](#_Toc132108971)

[Desafios e oportunidades futuras 22](#_Toc132108972)

# I. Introdução

## Definição de computação distribuída

Computação distribuída é um modelo de processamento de dados em que um sistema computacional é composto por múltiplos componentes autónomos, como computadores, dispositivos móveis, servidores, entre outros, que trabalham juntos de forma coordenada para realizar uma tarefa computacional.

Neste modelo, cada componente do sistema é responsável por uma parte da computação, e a comunicação e coordenação entre eles é realizada por meio de uma rede de comunicação, como a internet. Esta abordagem permite que tarefas complexas sejam divididas em partes menores, que podem ser processadas em paralelo, aumentando assim a eficiência e a velocidade de processamento.

A computação distribuída é utilizada em diversas áreas, como processamento de grandes volumes de dados, inteligência artificial, sistemas de controle de tráfego aéreo, entre outras aplicações que requerem alta capacidade de processamento e coordenação entre múltiplos componentes.

## Importância da computação distribuída

Algumas das principais razões pelas quais a computação distribuída é importante são:

1.     Escalabilidade: Isto significa que é possível aumentar a capacidade de processamento de um sistema sem ter que substituí-lo completamente por um novo sistema, o que pode ser mais caro e complexo.

2.     Redundância: A computação distribuída também permite que o sistema tenha redundância em caso de falhas em algum componente. Isso significa que, se um componente falhar, outros componentes do sistema podem continuar processando a tarefa, garantindo assim a continuidade do processamento.

3.     Eficiência: A computação distribuída permite que tarefas sejam divididas em partes menores, que podem ser processadas em paralelo, aumentando a eficiência e a velocidade de processamento. Isso é especialmente importante em tarefas que requerem alto poder de processamento, como inteligência artificial, simulações complexas e análise de grandes volumes de dados.

4.     Flexibilidade: A computação distribuída permite que diferentes componentes do sistema tenham diferentes funções e responsabilidades. Isso significa que é possível criar sistemas mais flexíveis e adaptáveis, que podem ser ajustados para atender às necessidades específicas de uma determinada tarefa ou aplicação.

5.     Redução de custos: A computação distribuída pode ser uma opção mais econômica do que investir em um único sistema de alto desempenho. Isso porque, ao distribuir o processamento entre vários componentes, é possível utilizar recursos existentes e mais baratos, em vez de ter que investir em um único sistema de alto desempenho, que pode ser mais caro e difícil de manter.

# II. Arquiteturas de sistemas distribuídos

As arquiteturas de sistemas distribuídos podem ser classificadas de acordo com a forma como os componentes são organizados e como as comunicações entre eles são realizadas. As principais arquiteturas de sistemas distribuídos são:

1. Cliente-Servidor: Nesta arquitetura, os serviços são fornecidos por um servidor centralizado e os clientes fazem solicitações para aceder esses serviços. Esta arquitetura é comum em sistemas de banco de dados, onde os clientes acedem a informações armazenadas no servidor.
2. Peer-to-Peer: Nesta arquitetura, todos os componentes são iguais e colaboram para realizar a tarefa. Cada componente é tanto um cliente como um servidor, e comunicam uns com os outros diretamente para compartilhar recursos ou trocar informações. Esta arquitetura é comum em sistemas de compartilhamento de arquivos, como o BitTorrent.
3. Arquitetura em camadas: Nesta arquitetura, os componentes são organizados em camadas, onde cada camada tem uma função específica. A comunicação entre as camadas é realizada de forma estruturada e organizada, com cada camada a fornecer serviços à camada superior. Esta arquitetura é comum em sistemas de rede, onde as camadas podem incluir a camada de aplicação, a camada de transporte e a camada de rede.
4. Arquitetura orientada a serviços: Nesta arquitetura, os serviços são fornecidos como unidades independentes, que podem ser combinadas para formar sistemas mais complexos. Cada serviço é autónomo e pode ser acedido por meio de uma interface bem definida. Essa arquitetura é comum em sistemas de comércio eletrônico, onde serviços como pagamentos, envio de pedidos e gestão de estoque podem ser fornecidos por diferentes fornecedores.
5. Arquitetura de micro serviços: Nesta arquitetura, o sistema é dividido em micro serviços independentes, onde cada um executa uma tarefa específica. Cada micro serviço é autónomo e pode ser escalado independentemente. A comunicação entre os micro serviços é realizada por meio de APIs e protocolos bem definidos. Esta arquitetura é comum em sistemas modernos de computação em nuvem e em aplicações web escaláveis.

## Redes ponto a ponto

As redes ponto a ponto (peer-to-peer, ou P2P) são uma forma de conexão de computadores em que todos os dispositivos são considerados iguais, e cada um pode comunicar diretamente com qualquer outro dispositivo conectado à rede, sem a necessidade de um servidor central.

Essas redes são bastante utilizadas para compartilhamento de arquivos, pois permitem que os user’s possam baixar e enviar arquivos diretamente uns aos outros, sem a necessidade de um servidor intermediário. No entanto, as redes P2P também podem ser usadas para outras finalidades, como jogos online, mensagens instantâneas e compartilhamento de recursos, como impressoras e conexão à internet.

Existem vários tipos de redes P2P, cada uma com suas próprias características. Alguns exemplos incluem:

1. Rede P2P pura: Nesta rede, todos os dispositivos são iguais e se comunicam diretamente uns com os outros. Não há um servidor central que gira as conexões.
2. Rede P2P híbrida: Nesta rede, alguns dispositivos atuam como servidores centrais, mas apenas para coordenar as conexões entre os outros dispositivos. Esses servidores não armazenam conteúdo.
3. Rede P2P baseada em tracker: Nesta rede, um servidor central, chamado de tracker, é responsável por coordenar as conexões entre os dispositivos. Os dispositivos comunicam diretamente uns com os outros para compartilhar arquivos, mas usam o tracker para encontrar outros dispositivos na rede.

As redes P2P têm algumas vantagens em relação às redes centralizadas, como maior eficiência no uso de recursos e maior resistência a falhas. No entanto, também podem ser mais difíceis de gerir e podem apresentar problemas de segurança e privacidade, como a possibilidade de compartilhamento de conteúdo ilegal e a exposição de informações pessoais.

## Sistemas cliente-servidor

O sistema cliente-servidor é uma arquitetura de rede em que um ou mais dispositivos, chamados de clientes, se conectam a um ou mais dispositivos, chamados de servidores, para compartilhar recursos, dados e serviços.

Os clientes enviam solicitações aos servidores, que processam as solicitações e enviam as respostas de volta aos clientes. Esta arquitetura é amplamente utilizada em redes de computadores, em que muitos dispositivos precisam de se conectar a um ou mais servidores para aceder a recursos compartilhados, como arquivos, bases de dados, impressoras e serviços de aplicações.

Algumas das principais características do modelo cliente-servidor incluem:

1. Centralização: O servidor é o ponto central da rede, responsável por gerir os recursos e os serviços que são compartilhados com os clientes.
2. Escalabilidade: A arquitetura cliente-servidor permite que mais clientes sejam adicionados à rede conforme a procura aumenta, sem afetar o desempenho ou a disponibilidade dos recursos.
3. Segurança: Os servidores podem ser configurados para controlar o acesso aos recursos compartilhados, permitindo que apenas user’s autorizados tenham acesso a esses recursos.
4. Desempenho: A arquitetura cliente-servidor pode melhorar o desempenho da rede, pois o servidor pode executar operações complexas e pesadas em recursos, deixando os clientes livres para realizar tarefas mais leves.

No entanto, o modelo cliente-servidor também pode apresentar alguns desafios, como a necessidade de manter e atualizar o software do servidor, a complexidade da configuração e a possível sobrecarga do servidor quando muitos clientes se conectam simultaneamente.

## Sistemas em cluster

Um sistema em cluster é um conjunto de computadores interconectados que trabalham juntos para fornecer alta disponibilidade, escalabilidade e desempenho para aplicações e serviços. Esses sistemas são usados em serviços web, bases de dados, renderização de imagens, processamento de dados e simulações científicas.

Num cluster, cada computador é um “nó” e possui um sistema operativo e recursos próprios, como processadores, memória e armazenamento.

Existem vários tipos de clusters, como:

1. Cluster de alto desempenho (HPC): usado para computação científica, engenharia e outros tipos de aplicativos que exigem grande poder de processamento e memória. Estes clusters podem ter centenas ou até milhares de nós.
2. Cluster de Load Balancer: usado para distribuir o tráfego de rede entre vários servidores, a fim de garantir que a carga seja equilibrada e os recursos não sejam sobrecarregados.
3. Cluster de alta disponibilidade (HA): usado para fornecer serviços ininterruptos, mesmo em caso de falha de um ou mais nós do cluster.
4. Cluster de armazenamento: usado para fornecer armazenamento compartilhado numa rede de computadores.

Os sistemas em cluster oferecem várias vantagens, incluindo:

* Alta disponibilidade: se um nó falhar, o serviço ainda pode ser executado noutros nós do cluster.
* Escalabilidade: o cluster pode ser facilmente expandido adicionando mais nós.
* Desempenho: a capacidade de processamento e memória pode ser compartilhada entre os nós do cluster, melhorando o desempenho do sistema.

No entanto, a configuração e manutenção de um sistema em cluster podem ser complexas e exigir um conhecimento técnico avançado. Além disso, os custos de hardware e software podem ser significativos, especialmente em clusters de alta performance.

# III. Compartilhamento de recursos em sistemas distribuídos

## Uso de vários computadores para processamento e armazenamento de dados

## Comunicação entre computadores

# IV. Segurança e confiabilidade em sistemas distribuídos

## Replicação de dados

## Deteção e tratamento de falhas

# V. Protocolos de comunicação em sistemas distribuídos

*“Os protocolos de comunicação são* ***conjuntos de regras, procedimentos e formatos de mensagens*** *que regem a comunicação entre dispositivos num sistema distribuído”*

## Sincronização de Tarefas

“Permite que diferentes computadores trabalhem em conjunto e de forma coordenada para realizar uma tarefa específica”.

A sincronização de tarefas é um aspeto importante dos protocolos de comunicação. Em sistemas distribuídos, as tarefas são normalmente divididas em partes menores e executadas em computadores diferentes, que precisam de ser sincronizadas para assegurar que a tarefa é concluída com sucesso.  
Existem vários **protocolos de sincronização de tarefas** que podem ser utilizados em sistemas distribuídos, incluindo:  
1. **Protocolos de bloqueio**: Estes protocolos são utilizados para controlar o acesso a recursos partilhados em sistemas distribuídos. Permitem que apenas um computador aceda a um recurso partilhado de cada vez, evitando conflitos e assegurando que o recurso é utilizado corretamente.

\*Two-Phase Locking (2PL) numa base de dados:

* Fase de crescimento, cada processo solicita um bloqueio, naquilo que está a fazer (X-lock). O primeiro a solicitar, obtém o bloqueio.
* Assim que o primeiro conclui, atualiza e liberta para o segundo. O segundo obtém o bloqueio.

2. **Protocolos de operações**: fornece um conjunto de propriedades que garantem que as transações sejam executadas de forma consistente e confiável em um sistema distribuído. Asseguram que as transações são concluídas com sucesso ou completamente desfeitas se ocorrer um erro. \*ACID (Atomicidade, Consistência, Isolamento e Durabilidade) A: ou todo o trabalho é feito, ou nada feito.

3. **Protocolos eleitorais**: Estes protocolos são utilizados para selecionar um líder ou coordenador num grupo de computadores de um sistema distribuído. O líder é responsável por coordenar **os processos** e assegurar que todos os computadores estão a trabalhar em conjunto de forma coordenada. \*Líder de Bully, cada nó do sistema tem um ID único e um com o ID mais alto é considerado como o nó mais poderoso.

Nó 1: ID = 100

Nó 2: ID = 50

Nó 3: ID = 200 (o mais poderoso do sistema)

Nó 4: ID = 150 (caso o 3 falhe, este será o novo eleito para garantir a tarefa que o servidor anterior estava a executar)

4. **Protocolos de sincronização dos relógios**: Estes protocolos são utilizados para sincronizar os relógios de diferentes computadores num sistema distribuído, permitindo-lhes coordenar com precisão as suas ações e eventos.  
\* (Network Time Protocol - NTP): faz a comparação entre o tempo do relógio do dispositivo com o tempo do relógio do servidor e sincroniza.

A sincronização de tarefas é essencial para assegurar que as tarefas em sistemas distribuídos sejam concluídas de forma eficaz e eficiente. Os protocolos de sincronização de tarefas ajudam a assegurar que os diferentes computadores de um sistema distribuído possam trabalhar em conjunto de forma coordenada para concluir uma tarefa com sucesso.

## Comunicação entre computadores

*“Existem protocolos concebidos para permitir que os computadores comuniquem entre si, transmitindo e recebendo dados em diferentes formatos”*

São essenciais a existência de protocolos para assegurar que os diferentes computadores de um sistema distribuído possam comunicar eficazmente e de forma coordenada para realizar uma tarefa ou partilhar dados. Os protocolos de comunicação em sistemas distribuídos incluem várias etapas que permitem que os **computadores comuniquem** entre si. Estas **etapas** podem incluir:  
1. **Estabelecimento de ligação**: o protocolo define como um computador estabelece uma ligação com outro.  
2. **Transmissão de dados**: o protocolo define a forma como os dados são transmitidos de um computador para outro.  
3. **Receção de dados**: O protocolo define como um computador recebe dados enviados de outro computador.  
4. **Tratamento de erros**: O protocolo define a forma como os erros de comunicação são tratados.  
5. **Fim da ligação**: o protocolo define como é terminada uma ligação após a conclusão da tarefa ou transmissão de dados.

O protocolo **TCP** (Transmission Control Protocol) garante que os dados são entregues e sem erros.

Para iniciar uma conversa:

* O cliente envia a mensagem SYN (sincronização) para o servidor, “quero iniciar sessão”;
* O servidor responde SYN-ACK (sincronização e reconhecimento) “estou pronto”
* O cliente responde com a mensagem ACK (reconhecimento) “ligação está ok”

Já pode trocar dados de forma confiável. O protocolo TCP também incluí mecanismos para controlar o fluxo de dados e deteção /correção de erros na transmissão.

Para terminar uma conversa:

* A aplicação que deseja encerrar a conexão envia uma mensagem FIN (finalizar) “quero terminar a ligação”;
* A aplicação recetora confirma a receção e envia ACK “recebido”;
* A aplicação recetora envia a mensagem FIN “estou pronto para terminar”;
* A aplicação que deseja encerrar confirma a receção e envia ACK “terminado”.

O protocolo **HTTP** é usado para transferência de informações na World Wide Web. É baseado numa arquitetura cliente-servidor e usa um modelo de requisição-resposta, onde o cliente envia uma solicitação HTTP para o servidor e o servidor responde com uma mensagem HTTP com o recurso solicitado. Existem vários métodos de solicitação HTTP que permitem diferentes tipos de operações, como:

GET: solicitar um recurso ao servidor \*página HTML ou Imagem.

GET /articles/123 HTTP/1.1

Host: example.com

POST: enviar dados do cliente para o servidor \*informações do formulário no HTML

POST /api/posts HTTP/1.1

Host: example.com

Content-Type: application/json

{

"title": "Novo Post",

"content": "Este é o conteúdo do meu novo post.",

"author": "João"

}

PUT: atualizar um recurso existente no servidor.

PUT /api/tasks/1 HTTP/1.1

Host: example.com

Content-Type: application/json

{

"title": "Atualizar exemplo de PUT",

"description": "Adicionar um exemplo para PUT",

"status": "Em andamento"

}

DELETE: excluir um recurso existente no servidor.

DELETE /api/tasks/1 HTTP/1.1

Host: example.com

Existem também o protocolo cliente-servidor, como **FTP** (File Transfer Protocol) que serve para enviar arquivo e o **SMTP** (Simple Mail Transfer Protocolo) para enviar e-mails entre servidores de e-mail.

# VI. Conclusão

A computação distribuída é uma abordagem à construção de sistemas informáticos que permite que vários computadores trabalhem em conjunto de forma coordenada para executar tarefas complexas. Esta abordagem oferece várias vantagens sobre sistemas centralizados, incluindo maior escalabilidade, maior disponibilidade, melhor desempenho e maior resiliência a falhas.  
Contudo, a computação distribuída também apresenta desafios significativos em termos de gestão de recursos, sincronização de tarefas, transmissão e receção de dados, segurança, e outros aspetos críticos. Para lidar com estes desafios, são necessários protocolos e algoritmos específicos que permitam aos computadores de um sistema distribuído trabalhar em conjunto de uma forma coordenada e eficiente.  
Globalmente, a computação distribuída é uma abordagem importante para a construção de sistemas informáticos complexos e de grande escala. À medida que a procura de recursos informáticos continua a crescer, é provável que a computação distribuída se torne cada vez mais importante para a construção de sistemas eficientes e escaláveis que possam lidar com as necessidades do mundo moderno.

## Benefícios da computação distribuída

A computação distribuída oferece uma série de benefícios sobre sistemas centralizados, incluindo:

1. **Escalabilidade**: permite adicionar ou remover facilmente recursos informáticos do sistema, o que permite que o sistema cresça ou encolha conforme necessário.

2. **Melhor desempenho**: executa tarefas mais rapidamente do que os sistemas centralizados, porque as tarefas podem ser distribuídas por vários computadores e executadas em paralelo.

3. **Disponibilidade**: proporciona uma maior disponibilidade do sistema porque se um computador falhar, outros computadores podem assumir as tarefas que estavam a ser executadas pelo computador falhado.

4. **Redundância**: utilizada para criar sistemas redundantes, que podem ser utilizados para garantir que a informação e os dados estejam sempre disponíveis.

5. **Flexibilidade**: utilizada para criar sistemas flexíveis que podem ser adaptados às necessidades específicas de diferentes utilizadores ou aplicações.

6. **Relação custo-eficácia**: mais rentável do que os sistemas centralizados porque permite a partilha e utilização mais eficiente dos recursos informáticos.

7. **Resiliência a falhas**: resistente a falhas do que os sistemas centralizados porque os dados e tarefas podem ser distribuídos por vários computadores, e podem ser criadas redundâncias para assegurar que o sistema continua a funcionar mesmo quando ocorrem falhas.

No geral, a computação distribuída pode ser uma solução eficaz para enfrentar os desafios da computação moderna, oferecendo maior escalabilidade, desempenho, disponibilidade, flexibilidade, e resiliência a falhas.

## Desafios e oportunidades futuras

Embora a computação distribuída tenha muitos benefícios, existem também desafios e oportunidades futuras a considerar. Alguns destes desafios e oportunidades incluem:

1. **A gestão de recursos**: A gestão de recursos distribuídos através de múltiplos computadores pode ser complexa e desafiante. As futuras soluções de gestão de recursos devem ser desenvolvidas para assegurar que os recursos sejam atribuídos de forma eficiente e otimizada.

2. **Segurança**: Com tantos dispositivos ligados a um sistema distribuído, a segurança é uma grande preocupação. A computação distribuída deve ser protegida de intrusões e ciberataques, o que pode ser difícil devido à natureza distribuída da computação.

3. **Sincronização de tarefas**: Assegurar que as tarefas são executadas na ordem correta e de forma sincronizada pode ser um desafio nos sistemas distribuídos. Novos protocolos e algoritmos devem ser desenvolvidos para assegurar que as tarefas sejam sincronizadas de forma eficiente.

4. **Envio e receção de dados**: O envio e a receção de dados em sistemas distribuídos podem ser problemáticos. As soluções futuras devem concentrar-se em melhorar a eficiência da transmissão e assegurar que os dados cheguem ao seu destino de forma correta e segura.

5. **Análise de dados em tempo real**: a computação distribuída pode ser utilizada para a análise de dados em tempo real, mas pode ser um desafio assegurar que a análise seja suficientemente precisa e rápida. As soluções futuras devem concentrar-se em melhorar a eficiência da análise de dados em tempo real.

6. **Oportunidades para o IoT**: A computação distribuída é fundamental para a Internet das Coisas (IdC), permitindo que os dispositivos comuniquem uns com os outros e realizem tarefas complexas. A computação distribuída pode ser utilizada para melhorar a eficiência da Internet das Coisas e permitir novas aplicações e serviços.

7. **Aumento do poder de processamento**: Com a computação distribuída, é possível aumentar significativamente o poder de processamento disponível para a execução de tarefas complexas. Isto pode levar a novas aplicações e serviços que antes não eram possíveis.

Em geral, a computação distribuída oferece muitas oportunidades para melhorar a eficiência e o poder de processamento dos sistemas informáticos modernos, mas também apresenta desafios que precisam de ser ultrapassados. O futuro da computação distribuída dependerá da capacidade de enfrentar estes desafios e de tirar partido das oportunidades apresentadas.