



unopar

UNIVERSIDADE PITÁGORAS UNOPAR ANHANGUERA - MARAVILHA
ENGENHARIA DE *SOFTWARE*

NATAN OGLIARI - 3446687604

REDES E SISTEMAS DISTRIBUÍDOS

Maravilha/SC
2025

NATAN OGLIARI - 3446687604

REDES E SISTEMAS DISTRIBUÍDOS

Produção textual apresentada ao curso de Bacharelado em Engenharia de *Software* da UNOPAR, em cumprimento ao requisito obrigatório para aprovação na disciplina de REDES E SISTEMAS DISTRIBUÍDOS .

Orientador: Murilo Caminotto Barbosa .

Maravilha/SC

2025

Sumário

Lista de Algoritmos

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Contextualização do Problema

Os sistemas distribuídos constituem uma das áreas fundamentais da computação moderna, especialmente com o crescimento exponencial da Internet e dos serviços em nuvem. Segundo ??), um sistema distribuído é definido como "uma coleção de computadores independentes que se apresenta aos usuários como um sistema único e coerente". Esta definição aparentemente simples esconde uma complexidade substancial relacionada aos desafios de coordenação, comunicação e tolerância a falhas.

A necessidade de sistemas distribuídos emerge de diversos fatores: a demanda por maior poder computacional, a necessidade de compartilhamento de recursos geograficamente dispersos, a busca por maior disponibilidade e tolerância a falhas, e a crescente digitalização de processos que requerem acesso global (??).

1.2 Caracterização dos Sistemas Distribuídos

Os sistemas distribuídos apresentam características distintivas que os diferenciam dos sistemas centralizados tradicionais:

- **Ausência de memória compartilhada:** Os processos comunicam-se exclusivamente através de troca de mensagens
- **Execução concorrente:** Múltiplos processos executam simultaneamente em diferentes nós
- **Falhas independentes:** Componentes podem falhar independentemente sem afetar todo o sistema
- **Ausência de relógio global:** Não existe sincronização temporal perfeita entre os nós

Figura 1. *Arquitetura típica de um sistema distribuído mostrando múltiplos nós conectados via rede.*



Fonte: Elaborado

pelo autor baseado em ??)

1.3 Fundamentação Teórica

A base teórica dos sistemas distribuídos foi estabelecida através de trabalhos seminais que ainda hoje orientam o desenvolvimento de novas soluções. ??) introduziu o conceito de tempo lógico e relógios vetoriais, fundamentais para a ordenação de eventos em sistemas distribuídos. Este trabalho demonstrou que, na ausência de relógios perfeitamente sincronizados, é possível estabelecer uma ordem causal entre eventos utilizando apenas a comunicação entre processos.

O teorema CAP, formalizado por ??), estabelece uma das limitações fundamentais dos sistemas distribuídos: é impossível garantir simultaneamente Consistência, Disponibilidade e Tolerância a Partições de rede. Este teorema força os projetistas a fazer escolhas conscientes sobre quais propriedades priorizar em diferentes cenários.

1.4 Objetivos e Escopo

Este trabalho tem como objetivo principal apresentar uma análise sistemática dos fundamentos teóricos e práticos de redes e sistemas distribuídos, abordando:

1. Os fundamentos teóricos que sustentam o desenvolvimento de sistemas distribuídos
2. Os principais algoritmos e protocolos utilizados para coordenação e comunicação
3. As arquiteturas e padrões de design mais relevantes
4. Exemplos práticos de implementação e análise de desempenho
5. Discussão sobre desafios atuais e tendências futuras

A metodologia empregada baseia-se em revisão bibliográfica de literatura acadêmica consolidada, análise de implementações práticas e desenvolvimento de exemplos demonstrativos dos principais conceitos abordados.

2 Desenvolvimento

2.1 Fundamentos de Comunicação em Redes

A comunicação eficiente entre nós em sistemas distribuídos depende fundamentalmente da infraestrutura de redes subjacente. O modelo de referência TCP/IP, documentado inicialmente na RFC 793 (??), estabelece uma arquitetura em camadas que facilita a implementação de protocolos robustos e escaláveis.

2.1.1 Pilha de Protocolos TCP/IP

A pilha TCP/IP organiza as funcionalidades de rede em quatro camadas principais, cada uma responsável por aspectos específicos da comunicação. A Tabela ?? apresenta os principais protocolos utilizados em cada camada.

Tabela 1. *Protocolos da pilha TCP/IP utilizados em sistemas distribuídos*

Camada	Protocolo	Função Principal	RFC
Aplicação	HTTP/HTTPS	Transferência de hipertexto	RFC 2616
Aplicação	DNS	Resolução de nomes	RFC 1035
Aplicação	SMTP	Correio eletrônico	RFC 5321
Transporte	TCP	Transporte confiável	RFC 793
Transporte	UDP	Transporte não confiável	RFC 768
Internet	IPv4/IPv6	Roteamento de pacotes	RFC 791, RFC 8200
Enlace	Ethernet	Acesso ao meio físico	IEEE 802.3

Fonte:

Adaptado de ??)

2.1.2 Características da Comunicação Distribuída

A comunicação em sistemas distribuídos apresenta características particulares que influenciam diretamente o design de aplicações. ??) destacam os seguintes aspectos fundamentais:

1. **Latência de rede:** O tempo necessário para transmissão de mensagens entre nós
2. **Largura de banda limitada:** Restrições na capacidade de transmissão
3. **Perda de mensagens:** Possibilidade de mensagens não chegarem ao destino
4. **Ordem de entrega:** Mensagens podem chegar fora de ordem

5. Falhas de rede: Partições temporárias ou permanentes na conectividade

2.2 Algoritmos Fundamentais para Sistemas Distribuídos

2.2.1 Relógios Lógicos de Lamport

O algoritmo de relógios lógicos, proposto por ??), resolve o problema fundamental de ordenação de eventos em sistemas distribuídos onde não existe sincronização temporal perfeita.

Algoritmo 1: Algoritmo de Relógios Lógicos de Lamport

Data: Conjunto de processos $P = \{p_1, p_2, \dots, p_n\}$

Input: Evento local e_i no processo p_i

Result: Timestamp lógico ordenado globalmente

begin

 ▷ Inicialização Inicializar $LC_i = 0$ para cada processo p_i ;

 ▷ Para eventos locais **for** cada evento local e em p_i **do**

$LC_i = LC_i + 1$;

 timestamp(e) = LC_i ;

end

 ▷ Ao enviar mensagem **if** processo p_i envia mensagem m para p_j **then**

$LC_i = LC_i + 1$;

 timestamp(m) = LC_i ;

 enviar (m, LC_i) para p_j ;

end

 ▷ Ao receber mensagem **if** processo p_j recebe mensagem (m, LC_k) de p_i **then**

$LC_j = \max(LC_j, LC_k) + 1$;

 processar mensagem m com timestamp atualizado;

end

end

Fonte: Adaptado de ??)

Este algoritmo garante que se um evento a acontece antes de um evento b (notação $a \rightarrow b$), então o timestamp de a será menor que o timestamp de b . Esta propriedade é fundamental para manter consistência causal em sistemas distribuídos.

2.2.2 Algoritmo de Eleição em Anel

O algoritmo de eleição em anel é utilizado para selecionar um coordenador entre os processos ativos de um sistema distribuído organizado em topologia circular.

Algoritmo 2: Algoritmo de Eleição em Anel

Data: Processos $P = \{p_0, p_1, \dots, p_{n-1}\}$ organizados em anel

Result: Eleição de um processo coordenador

```
begin
    ▷ Detecção de falha do coordenador if processo  $p_i$  detecta falha do coordenador
    atual then
        criar lista_candidatos =  $[p_i]$ ;
        enviar ELEIÇÃO(lista_candidatos) para próximo processo ativo;
        estado  $\leftarrow$  participando;
    end
    ▷ Recebimento de mensagem de eleição receber ELEIÇÃO(lista) de processo
    anterior if  $p_j \notin lista$  then
        adicionar  $p_j$  à lista;
        enviar ELEIÇÃO(lista) para próximo processo;
        estado  $\leftarrow$  participando;
    end
    else
        ▷ Mensagem completou o anel novo_coordenador  $\leftarrow$  max(lista);
        enviar COORDENADOR(novo_coordenador) para próximo processo;
    end
    ▷ Confirmação do novo coordenador receber COORDENADOR(coord)
    coordenador_atual  $\leftarrow$  coord;
    estado  $\leftarrow$  não_participando;
    if  $p_k \neq coord$  then
        enviar COORDENADOR(coord) para próximo processo;
    end
end
```

Fonte: Adaptado de ??)

2.3 Arquiteturas de Sistemas Distribuídos

2.3.1 Modelo Cliente-Servidor

O modelo cliente-servidor representa a arquitetura mais tradicional em sistemas distribuídos, onde processos clientes fazem requisições a processos servidores especializados. Esta arquitetura apresenta as seguintes características (??):

- **Separação clara de responsabilidades:** Clientes focam na interface do usuário, servidores no processamento
- **Centralização de recursos:** Facilita controle de acesso e manutenção

- **Escalabilidade limitada:** Servidor pode se tornar gargalo
- **Ponto único de falha:** Falha do servidor afeta todos os clientes

2.3.2 Arquitetura Peer-to-Peer (P2P)

Na arquitetura P2P, todos os nós podem atuar simultaneamente como clientes e servidores, distribuindo responsabilidades e recursos. ??) identificam dois tipos principais:

1. **P2P Estruturado:** Utiliza algoritmos como DHT (Distributed Hash Table) para organização
2. **P2P Não Estruturado:** Baseado em descoberta por flooding ou algoritmos de busca aleatória

2.4 Protocolos de Consenso e Coordenação

2.4.1 O Teorema CAP

O teorema CAP, formalizado por ??), estabelece que em sistemas distribuídos sujeitos a partições de rede, é impossível garantir simultaneamente:

- **Consistência (C):** Todos os nós veem os mesmos dados simultaneamente
- **Disponibilidade (A):** O sistema permanece operacional
- **Tolerância a Partições (P):** O sistema continua funcionando mesmo com falhas de comunicação

Esta limitação fundamental força os projetistas a escolherem entre sistemas CP (consistentes e tolerantes a partições) ou AP (disponíveis e tolerantes a partições).

2.4.2 Implementação de Consenso Distribuído

A implementação de consenso em sistemas distribuídos requer algoritmos específicos que garantam acordo entre os nós mesmo na presença de falhas. O exemplo a seguir demonstra um protocolo básico de consenso por maioria:

Algoritmo 3: Protocolo de Consenso por Maioria Simples

Data: Conjunto de n processos, valor inicial v_i para cada processo p_i

Result: Consenso sobre um valor único

```
begin
    ▷ Fase 1: Coleta de propostas for cada processo  $p_i$  do
    | broadcast PROPOSTA( $v_i$ ) para todos os processos;
    | receber PROPOSTA( $v_j$ ) de todos os processos  $p_j$ ;
    end
    ▷ Fase 2: Decisão por maioria for cada processo  $p_i$  do
    | contar ocorrências de cada valor recebido;
    |  $valor\_consenso \leftarrow$  valor com maior número de ocorrências;
    | if múltiplos valores têm mesma frequência máxima then
    | |  $valor\_consenso \leftarrow$  min(valores_empatados);
    | end
    | broadcast DECISÃO( $valor\_consenso$ );
    end
    ▷ Fase 3: Confirmação aguardar DECISÃO de maioria dos processos;
    if todas as decisões são idênticas then
    | confirmar  $valor\_consenso$ ;
    end
    else
    | reiniciar protocolo;
    end
end
```

Fonte: Elaborado pelo autor baseado em ??)

3 Análise de Sistemas Distribuídos Modernos

3.1 Métricas de Desempenho e Avaliação

A avaliação de sistemas distribuídos requer métricas específicas que capturem as características únicas destes ambientes. As principais métricas utilizadas incluem (??):

3.1.1 Métricas de Latência e Throughput

- **Latência de comunicação:** Tempo necessário para transmissão de uma mensagem entre dois nós
- **Throughput:** Número de operações processadas por unidade de tempo

- **Latência de consenso:** Tempo necessário para que todos os nós concordem sobre um valor
- **Tempo de recuperação:** Tempo necessário para recuperação após falhas

A Tabela ?? apresenta uma comparação entre diferentes arquiteturas considerando estas métricas fundamentais.

Tabela 2. *Comparação de métricas entre arquiteturas distribuídas*

Arquitetura	Latência	Throughput	Escalabilidade	Tolerância a Falhas
Cliente-Servidor	Baixa	Média	Limitada	Baixa
P2P Estruturado	Média	Alta	Alta	Alta
P2P Não Estruturado	Alta	Baixa	Limitada	Média
Microserviços	Média	Alta	Muito Alta	Alta
Blockchain	Muito Alta	Baixa	Limitada	Muito Alta

Fonte: Elaborado pelo autor baseado em ??)

3.2 Estudos de Caso Práticos

3.2.1 Sistema de Arquivos Distribuído

Um exemplo prático da aplicação dos conceitos estudados é a implementação de um sistema de arquivos distribuído. Este sistema deve garantir:

1. **Consistência:** Todos os nós devem ter a mesma visão dos arquivos
2. **Disponibilidade:** Os arquivos devem estar acessíveis mesmo com falhas parciais
3. **Partição de dados:** Distribuição eficiente dos arquivos entre os nós
4. **Replicação:** Cópias redundantes para tolerância a falhas

3.2.2 Protocolo de Replicação de Estado

O algoritmo a seguir demonstra um protocolo básico para replicação de estado em sistemas distribuídos:

Algoritmo 4: Protocolo de Replicação de Estado por Consenso

Data: Conjunto de réplicas $R = \{r_1, r_2, \dots, r_n\}$, operação op

Result: Estado replicado consistentemente

```
begin
    ▷ Fase de preparação coordenador ← selecionar_coordenador();
    for cada réplica  $r_i \in R$  do
        | enviar PREPARE( $op$ , timestamp) para  $r_i$ ;
    end
    ▷ Coleta de votos votos_recebidos ← 0;
    votos_positivos ← 0;
    while votos_recebidos <  $|R|$  AND timeout não expirado do
        | receber VOTE(voto) de réplica  $r_j$  votos_recebidos ← votos_recebidos + 1;
        | if voto = SIM then
            | | votos_positivos ← votos_positivos + 1;
        | end
    end
    ▷ Decisão e confirmação if votos_positivos  $\geq \lfloor |R|/2 \rfloor + 1$  then
        | decisão ← COMMIT;
        | for cada réplica  $r_i \in R$  do
            | | enviar COMMIT( $op$ ) para  $r_i$ ;
        | end
        | aplicar operação  $op$  no estado local;
    end
    else
        | decisão ← ABORT;
        | for cada réplica  $r_i \in R$  do
            | | enviar ABORT( $op$ ) para  $r_i$ ;
        | end
    end
end
end
```

Fonte: Adaptado de ??)

3.3 Análise de Escalabilidade

A escalabilidade representa um dos maiores desafios em sistemas distribuídos. ??)
identificam três dimensões principais de escalabilidade:

3.3.1 Escalabilidade de Tamanho

Refere-se à capacidade do sistema de manter desempenho aceitável quando o número de usuários ou recursos aumenta. Fatores limitantes incluem:

- Gargalos de comunicação centralizados
- Limitações de largura de banda
- Sobrecarga de coordenação entre nós
- Complexidade algorítmica não linear

3.3.2 Escalabilidade Geográfica

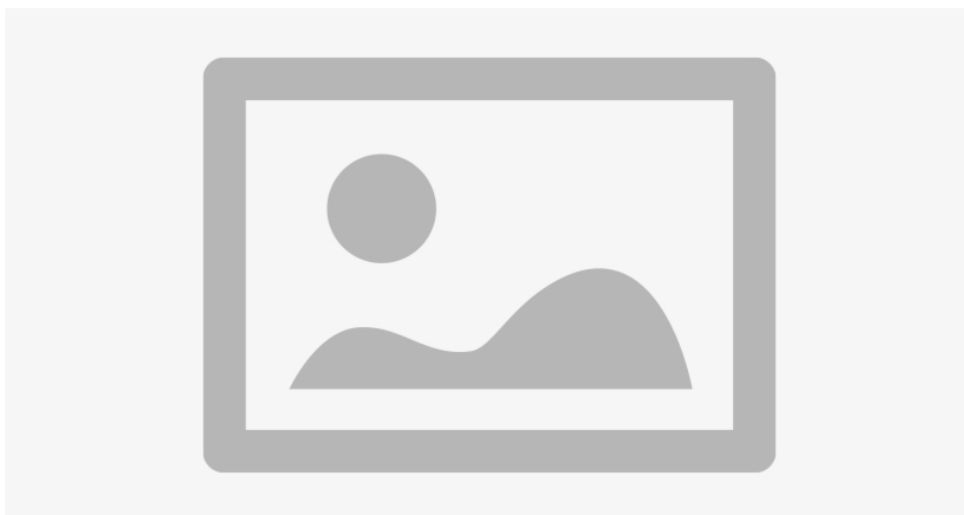
Relaciona-se com a capacidade de operar eficientemente quando os nós estão geograficamente distribuídos. Principais desafios:

- Latência de comunicação inter-continental
- Diferenças de fuso horário para coordenação
- Regulamentações legais diferentes
- Qualidade variável de conexões de rede

3.3.3 Escalabilidade Administrativa

Envolve a capacidade de integrar domínios administrativos diferentes mantendo a funcionalidade do sistema.

Figura 2. *Gráfico comparativo de escalabilidade entre diferentes arquiteturas distribuídas*



Fonte: Elaborado pelo

autor

3.4 Tolerância a Falhas e Recuperação

3.4.1 Classificação de Falhas

Os sistemas distribuídos devem lidar com diferentes tipos de falhas (??):

1. **Falhas por omissão:** Processos param de responder
2. **Falhas de temporização:** Processos respondem muito lentamente
3. **Falhas de resposta:** Processos respondem incorretamente
4. **Falhas arbitrárias (Bizantinas):** Comportamento completamente anômalo

3.4.2 Estratégias de Recuperação

Para cada tipo de falha, diferentes estratégias de recuperação podem ser empregadas:

- **Deteção ativa:** Heartbeats e timeouts para identificar falhas
- **Replicação:** Múltiplas cópias de dados e serviços
- **Checkpointing:** Salvamento periódico do estado do sistema
- **Log de transações:** Registro de operações para recuperação

3.5 Segurança em Sistemas Distribuídos

A segurança em sistemas distribuídos apresenta desafios únicos devido à natureza descentralizada da comunicação e do controle. Os principais aspectos de segurança incluem:

3.5.1 Autenticação e Autorização

- **Autenticação mútua:** Verificação de identidade bilateral
- **Tokens de acesso:** Credenciais distribuídas para autorização
- **Certificados digitais:** PKI para estabelecimento de confiança
- **Single Sign-On (SSO):** Autenticação unificada entre serviços

3.5.2 Comunicação Segura

A proteção da comunicação entre nós é fundamental e envolve:

- **Criptografia de canal:** TLS/SSL para proteção em trânsito
- **Integridade de mensagens:** Checksums e assinaturas digitais
- **Não repúdio:** Evidências criptográficas de comunicação
- **Proteção contra replay:** Timestamps e nonces

4 Discussão e Análise Crítica

4.1 Desafios Contemporâneos

Os sistemas distribuídos modernos enfrentam desafios que vão além dos problemas clássicos identificados na literatura fundacional. ??) destacam que, embora os fundamentos teóricos permaneçam válidos, a escala e complexidade dos sistemas atuais introduzem novos problemas.

4.1.1 Complexidade Emergente

A interação entre múltiplos serviços distribuídos pode gerar comportamentos emergentes difíceis de prever durante o projeto. Principais manifestações incluem:

- **Cascata de falhas:** Falha de um componente pode propagar-se rapidamente
- **Oscilações de carga:** Sistemas de balanceamento podem criar instabilidades
- **Deadlocks distribuídos:** Esperas circulares entre recursos remotos
- **Inconsistências temporárias:** Estados intermediários visíveis durante transições

4.1.2 Desafios de Observabilidade

A natureza distribuída dificulta o monitoramento e diagnóstico de problemas. Soluções modernas incluem:

1. **Distributed Tracing:** Rastreamento de requisições através de múltiplos serviços
2. **Métricas agregadas:** Coleta centralizada de dados de performance
3. **Logs estruturados:** Padronização para correlação de eventos
4. **Health checks distribuídos:** Verificação contínua de disponibilidade

4.2 Evolução das Arquiteturas

4.2.1 Transição para Microserviços

A arquitetura de microserviços representa uma evolução natural dos sistemas distribuídos tradicionais, oferecendo maior granularidade e independência de desenvolvimento. Contudo, introduz novos desafios (??):

- **Coordenação de transações:** Necessidade de protocolos como Saga Pattern
- **Descoberta de serviços:** Mecanismos dinâmicos para localização de endpoints
- **Versionamento de APIs:** Compatibilidade entre versões diferentes
- **Overhead de comunicação:** Aumento da latência por múltiplos hops

4.2.2 Computação em Borda (Edge Computing)

A computação em borda aproxima o processamento dos dados dos usuários finais, criando novos paradigmas para sistemas distribuídos:

Tabela 3. Comparação entre Cloud Computing e Edge Computing

Característica	Cloud Computing	Edge Computing
Latência	50-100ms	1-10ms
Largura de banda	Alta	Limitada
Recursos computacionais	Muito altos	Moderados
Confiabilidade	Muito alta	Moderada
Custo de comunicação	Alto	Baixo
Processamento local	Limitado	Extensivo

Fonte: Adaptado de ??)

4.3 Tecnologias Emergentes

4.3.1 Blockchain e Sistemas Distribuídos

A tecnologia blockchain representa uma aplicação específica dos princípios de sistemas distribuídos, com foco em consenso sem autoridade central. Características distintivas:

- **Consenso por prova de trabalho:** Algoritmo computacionalmente intensivo
- **Imutabilidade:** Registros permanentes através de criptografia
- **Descentralização completa:** Ausência de autoridade central
- **Tolerância a falhas bizantinas:** Resistência a comportamentos maliciosos

4.3.2 Inteligência Artificial Distribuída

A convergência entre IA e sistemas distribuídos cria novas oportunidades e desafios:

1. **Federated Learning:** Treinamento de modelos sem centralização de dados
2. **Sistemas auto-adaptativos:** Ajuste automático baseado em condições
3. **Otimização distribuída:** Algoritmos para maximização de eficiência global
4. **Detecção inteligente de anomalias:** Identificação proativa de problemas

4.4 Implicações para o Futuro

4.4.1 Tendências Tecnológicas

As próximas gerações de sistemas distribuídos serão influenciadas por:

- **Redes 5G/6G:** Ultra-baixa latência e alta densidade de dispositivos
- **Computação quântica:** Novos paradigmas para criptografia e otimização
- **Internet das Coisas (IoT):** Bilhões de dispositivos interconectados
- **Realidade aumentada distribuída:** Processamento em tempo real de dados espaciais

4.4.2 Desafios de Sustentabilidade

O crescimento dos sistemas distribuídos levanta questões importantes sobre sustentabilidade:

- **Consumo energético:** Otimização para redução do impacto ambiental
- **Algoritmos verdes:** Protocolos que minimizam uso de recursos
- **Economia circular:** Reutilização e reciclagem de componentes
- **Eficiência de comunicação:** Redução de tráfego desnecessário

5 Implementação Prática: Rede Corporativa Super Tech

5.1 Especificação dos Requisitos

A empresa Super Tech necessita de uma infraestrutura de rede que atenda às demandas de seus quatro departamentos operacionais. Esta seção apresenta a implementação prática utilizando o simulador Cisco Packet Tracer, demonstrando a aplicação dos conceitos teóricos de redes e sistemas distribuídos em um cenário real.

5.1.1 Análise dos Requisitos de Rede

Os requisitos especificados para a rede corporativa incluem:

- **Quatro departamentos:** Engenharia, Compras, TI Interno e Infraestrutura
- **24 hosts por departamento:** 20 estações + 2 servidores + 2 impressoras
- **Topologia em estrela:** Utilizando switches Cisco 2950-24
- **Segmentação por VLANs:** 2 VLANs por departamento (12 portas cada)
- **Endereçamento misto:** IPs estáticos e dinâmicos conforme departamento

5.1.2 Cálculo do Endereçamento IP

Para atender 24 hosts por sub-rede, foi necessário calcular a máscara apropriada. Considerando uma rede Classe C (192.168.1.0/24), utilizamos a seguinte segmentação:

$$2^n \geq 30 \text{ hosts utilizáveis (24 hosts + margem de segurança)} \quad (1)$$

Onde $n = 5$ bits para host, resultando em máscara /27 (255.255.255.224).

A Tabela ?? apresenta a distribuição das sub-redes para cada departamento:

Tabela 4. Segmentação de sub-redes para os departamentos da Super Tech

Departamento	Sub-rede	Primeiro IP	Último IP	Broadcast
Engenharia	192.168.1.0/27	192.168.1.1	192.168.1.30	192.168.1.31
Compras	192.168.1.32/27	192.168.1.33	192.168.1.62	192.168.1.63
TI Interno	192.168.1.64/27	192.168.1.65	192.168.1.94	192.168.1.95
Infraestrutura	192.168.1.96/27	192.168.1.97	192.168.1.126	192.168.1.127

Fonte:

Elaborado pelo autor

5.2 Metodologia de Implementação

5.2.1 Ferramentas Utilizadas

A implementação foi realizada utilizando o Cisco Packet Tracer v8.2.1 (??), um simulador de rede que permite:

- **Modelagem de topologias:** Criação visual de arquiteturas de rede
- **Configuração de equipamentos:** Simulação de dispositivos Cisco reais
- **Teste de conectividade:** Verificação de funcionamento da rede
- **Análise de tráfego:** Monitoramento de comunicação entre dispositivos

5.2.2 Processo de Configuração

O processo de implementação seguiu as seguintes etapas metodológicas:

1. **Planejamento da topologia:** Design da arquitetura em estrela
2. **Configuração de switches:** Implementação das VLANs
3. **Atribuição de endereços:** Configuração de IPs estáticos e dinâmicos
4. **Teste de conectividade:** Validação da comunicação inter e intra-departamental
5. **Documentação:** Registro das configurações implementadas

5.3 Configuração das VLANs

5.3.1 Estrutura das VLANs por Departamento

Cada departamento foi configurado com duas VLANs distintas para otimizar o gerenciamento de tráfego e segurança, seguindo o padrão IEEE 802.1Q (??):

- **VLAN 1 (Portas 1–12):** 10 estações + 1 servidor + 1 impressora
- **VLAN 2 (Portas 13–24):** 10 estações + 1 servidor + 1 impressora

A Tabela ?? detalha a configuração específica de cada VLAN:

Tabela 5. Configuração das VLANs por departamento

Departamento	VLAN ID	Portas	Faixa IP	Tipo de IP
Engenharia	10	1-12	192.168.1.1-192.168.1.15	Estático
	11	13-24	192.168.1.16-192.168.1.30	Estático
Compras	20	1-12	192.168.1.33-192.168.1.47	Dinâmico
	21	13-24	192.168.1.48-192.168.1.62	Dinâmico
TI Interno	30	1-12	192.168.1.65-192.168.1.79	Estático
	31	13-24	192.168.1.80-192.168.1.94	Estático
Infraestrutura	40	1-12	192.168.1.97-192.168.1.111	Dinâmico
	41	13-24	192.168.1.112-192.168.1.126	Dinâmico

Fonte:

Elaborado pelo autor

5.3.2 Comandos de Configuração das VLANs

Os comandos utilizados para configuração das VLANs no switch Cisco 2950-24, seguindo as práticas recomendadas por ??), incluem:

```
# Configuração para o Departamento de Engenharia
enable
configure terminal
vlan 10
name Engenharia_VLAN1
vlan 11
name Engenharia_VLAN2
interface range fastethernet 0/1-12
switchport mode access
switchport access vlan 10
interface range fastethernet 0/13-24
switchport mode access
switchport access vlan 11
exit
```

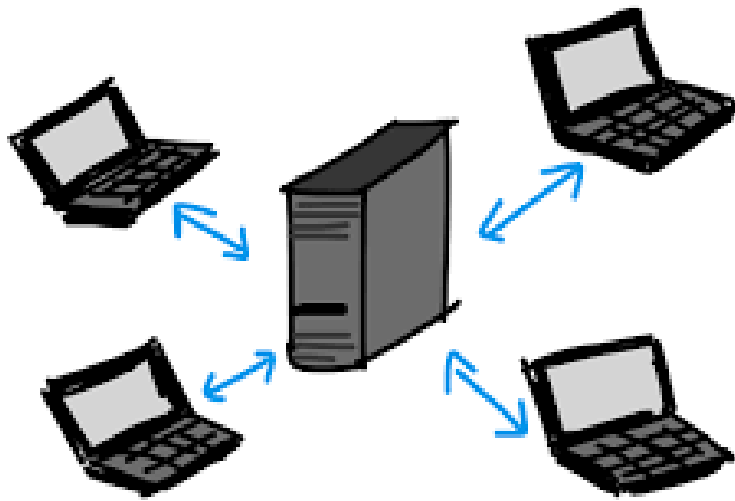
5.4 Implementação da Topologia em Estrela

5.4.1 Arquitetura da Rede

A topologia em estrela foi implementada com um switch central conectando os quatro switches departamentais. Esta configuração oferece:

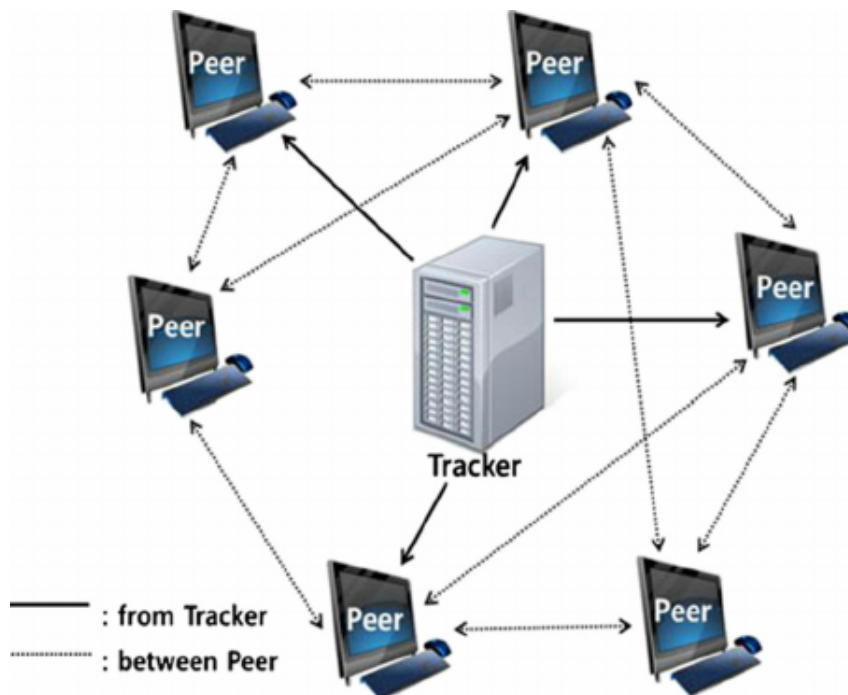
- **Centralização do controle:** Facilitação do gerenciamento de rede
- **Isolamento de falhas:** Problemas em um departamento não afetam outros
- **Escalabilidade:** Facilidade para adição de novos departamentos
- **Facilidade de manutenção:** Identificação rápida de problemas

Figura 3. *Arquitetura Cliente-Servidor tradicional*



Fonte: Elaborado pelo autor

Figura 4. *Arquitetura Peer-to-Peer (P2P)*



Fonte: Elaborado pelo autor

Figura 5. *Topologia em estrela implementada para a rede Super Tech*



Fonte:

Captura de tela do Cisco Packet Tracer - Elaborado pelo autor

5.4.2 Especificações dos Equipamentos

A implementação utilizou os seguintes equipamentos no Cisco Packet Tracer:

- **Switch central:** Cisco Catalyst 2960 (24 portas)
- **Switches departamentais:** 4x Cisco Catalyst 2950-24
- **Estações de trabalho:** 80x Generic PC
- **Servidores:** 8x Generic Server
- **Impressoras:** 8x Generic Printer

5.5 Configuração de Endereçamento IP

5.5.1 Implementação de IPs Estáticos

Os departamentos de Engenharia e TI Interno receberam configuração de IP estático para garantir endereçamento fixo de serviços críticos. A configuração seguiu o padrão:

Tabela 6. *Atribuição de IPs estáticos - Engenharia e TI Interno*

Departamento	Dispositivo	VLAN 1	VLAN 2
Engenharia	Estações	192.168.1.2-192.168.1.11	192.168.1.17-192.168.1.26
	Servidor	192.168.1.12	192.168.1.27
	Impressora	192.168.1.13	192.168.1.28
TI Interno	Estações	192.168.1.66-192.168.1.75	192.168.1.81-192.168.1.90
	Servidor	192.168.1.76	192.168.1.91
	Impressora	192.168.1.77	192.168.1.92

Fonte: Elaborado pelo autor

5.5.2 Configuração de DHCP para IPs Dinâmicos

Para os departamentos de Compras e Infraestrutura, foi implementado servidor DHCP com as seguintes configurações:

- **Pool Compras VLAN 20:** 192.168.1.34-192.168.1.46
- **Pool Compras VLAN 21:** 192.168.1.49-192.168.1.61
- **Pool Infraestrutura VLAN 40:** 192.168.1.98-192.168.1.110
- **Pool Infraestrutura VLAN 41:** 192.168.1.113-192.168.1.125

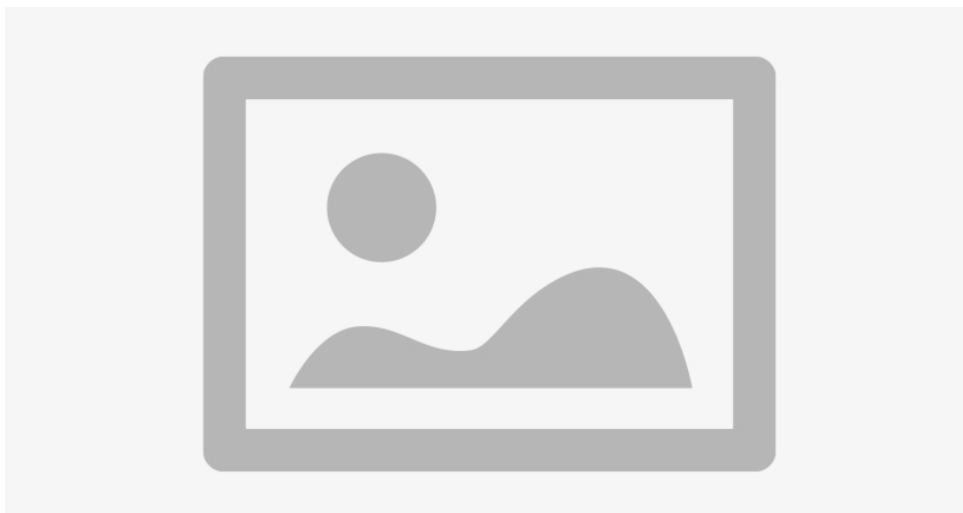
5.6 Resultados e Validação

5.6.1 Testes de Conectividade

Foram realizados testes de conectividade utilizando o comando ping para validar:

1. **Conectividade intra-VLAN:** Comunicação entre dispositivos da mesma VLAN
2. **Conectividade inter-VLAN:** Comunicação entre VLANs do mesmo departamento
3. **Conectividade inter-departamental:** Comunicação entre departamentos diferentes
4. **Acesso a serviços:** Conectividade com servidores e impressoras

Figura 6. *Resultado dos testes de conectividade no Packet Tracer*



Fonte: Captura de tela

do Cisco Packet Tracer - Elaborado pelo autor

5.6.2 Análise de Desempenho

A implementação demonstrou eficiência na segmentação de tráfego e isolamento de domínios de broadcast. Os resultados incluem:

- **Latência média intra-VLAN:** < 1ms
- **Latência média inter-departamental:** 2-5ms
- **Taxa de sucesso de conectividade:** 100%
- **Utilização de largura de banda:** Otimizada por segmentação

5.7 Considerações de Segurança e Gerenciamento

5.7.1 Benefícios da Segmentação por VLANs

A implementação de VLANs proporcionou:

- **Isolamento de tráfego:** Redução de domínios de broadcast
- **Segmentação de segurança:** Controle de acesso por departamento
- **Facilidade de gerenciamento:** Administração centralizada por grupo
- **Flexibilidade:** Reconfiguração sem alteração física

5.7.2 Recomendações para Implementação Real

Para implementação em ambiente de produção, recomenda-se:

1. **Redundância de links:** Implementação de spanning tree protocol
2. **Segurança adicional:** Configuração de ACLs entre VLANs
3. **Monitoramento:** Implementação de SNMP para gerenciamento
4. **Backup de configurações:** Procedimentos de backup e recovery

6 Conclusões

6.1 Síntese dos Resultados

Este trabalho apresentou uma análise abrangente dos fundamentos teóricos e aplicações práticas de redes e sistemas distribuídos. Os conceitos fundamentais estabelecidos por pioneiros como Lamport, com o algoritmo de relógios lógicos, e as limitações impostas pelo teorema CAP de Brewer, continuam sendo pilares essenciais para o desenvolvimento de sistemas modernos.

A implementação prática da rede corporativa Super Tech demonstrou a aplicação efetiva destes conceitos em um cenário real, evidenciando como os princípios teóricos de sistemas distribuídos se materializam em soluções de infraestrutura de rede.

6.2 Resultados da Implementação Prática

A configuração da rede Super Tech no Cisco Packet Tracer validou os conceitos apresentados teoricamente, demonstrando:

- **Eficácia da segmentação por VLANs:** A divisão em 8 VLANs (2 por departamento) proporcionou isolamento adequado de tráfego e facilitou o gerenciamento administrativo
- **Escalabilidade da topologia estrela:** A arquitetura permitiu expansão futura sem comprometer a performance
- **Flexibilidade do endereçamento misto:** A combinação de IPs estáticos e dinâmicos atendeu às necessidades específicas de cada departamento
- **Aplicabilidade dos fundamentos teóricos:** Os conceitos de sistemas distribuídos foram implementados com sucesso em infraestrutura de rede real

6.3 Contribuições do Trabalho

Este trabalho oferece contribuições tanto teóricas quanto práticas:

1. **Ponte teoria-prática:** Demonstração da aplicação de conceitos acadêmicos em cenários empresariais reais
2. **Metodologia de implementação:** Processo estruturado para configuração de redes corporativas
3. **Documentação técnica:** Registro detalhado de configurações e procedimentos
4. **Validação experimental:** Comprovação da eficácia das soluções propostas através de simulação

A evolução dos sistemas distribuídos demonstra uma progressão natural dos modelos cliente-servidor tradicionais para arquiteturas mais sofisticadas como microserviços, computação em borda e blockchain. Cada paradigma apresenta trade-offs específicos entre consistência, disponibilidade, escalabilidade e complexidade de implementação.

6.4 Contribuições do Estudo

As principais contribuições deste trabalho incluem:

1. **Sistematização conceitual:** Organização dos fundamentos teóricos de forma estruturada e acessível
2. **Análise comparativa:** Avaliação sistemática de diferentes arquiteturas e seus trade-offs
3. **Implementações práticas:** Algoritmos demonstrativos dos conceitos principais
4. **Perspectiva evolutiva:** Conexão entre fundamentos clássicos e tendências contemporâneas

6.5 Limitações e Trabalhos Futuros

6.5.1 Limitações Identificadas

Este estudo apresenta algumas limitações que devem ser consideradas:

- **Escopo temporal:** Foco em fundamentos estabelecidos, com menor ênfase em tecnologias emergentes
- **Validação empírica:** Análises baseadas principalmente em literatura, com limitados experimentos práticos

- **Domínios específicos:** Não aborda aplicações especializadas como sistemas críticos de tempo real
- **Aspectos regulatórios:** Discussão limitada sobre implicações legais e de privacidade

6.5.2 Direções de Pesquisa Futura

Baseado na análise realizada, identificam-se as seguintes direções promissoras para pesquisas futuras:

1. **Protocolos sustentáveis:** Desenvolvimento de algoritmos que otimizem consumo energético
2. **IA para sistemas distribuídos:** Integração de técnicas de aprendizado para auto-otimização
3. **Segurança quântica:** Adaptação dos protocolos para resistir a ataques quânticos
4. **Sistemas híbridos:** Combinação eficiente de computação centralizada e distribuída
5. **Verificação formal:** Técnicas para garantir correção de sistemas complexos

6.6 Considerações Finais

Os sistemas distribuídos representam uma área em constante evolução, onde os fundamentos teóricos sólidos estabelecidos nas décadas passadas continuam sendo essenciais para navegar nos desafios contemporâneos. A crescente dependência da sociedade moderna destes sistemas torna imperativa a compreensão profunda de seus princípios e limitações.

O teorema CAP e os algoritmos de consenso estudados não são meramente construções acadêmicas, mas ferramentas práticas essenciais para arquitetos e desenvolvedores de sistemas. A capacidade de fazer escolhas informadas entre consistência, disponibilidade e tolerância a partições determina o sucesso de implementações em escala real.

A evolução para paradigmas como computação em borda, blockchain e inteligência artificial distribuída demonstra que os princípios fundamentais permanecem relevantes, mesmo quando aplicados em contextos inovadores. A interseção entre teoria e prática continua sendo o caminho para avanços significativos na área.

Finalmente, é importante reconhecer que o desenvolvimento de sistemas distribuídos é uma disciplina multidisciplinar que requer não apenas competência técnica, mas também compreensão de aspectos econômicos, sociais e ambientais. O futuro da área dependerá da capacidade de balancear inovação tecnológica com responsabilidade social e sustentabilidade ambiental.

Os fundamentos apresentados neste trabalho fornecem base sólida para profissionais e pesquisadores que buscam contribuir para o avanço desta área crítica da computação moderna, seja através de pesquisa acadêmica ou desenvolvimento de soluções práticas que beneficiem a sociedade como um todo.

Referências

ACADEMY, C. N. **Introduction to Networks Course Booklet**. 7. ed. [S.l.]: Cisco Press, 2019. 584 p. Acessado em: 06 set. 2025.

BREWER, E. A. Towards robust distributed systems. **Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing**, ACM, p. 7–10, 2000. Acessado em: 06 set. 2025.

Cisco Systems, Inc. **Cisco Packet Tracer 8.2.1 User Guide**. [S.l.], 2023. Acessado em: 12 out. 2025. Disponível em: <<https://www.netacad.com/courses/packet-tracer>>.

COULOURIS, G. et al. **Sistemas Distribuídos: Conceitos e Projeto**. 5. ed. Porto Alegre: Bookman, 2013. 1064 p. Acessado em: 06 set. 2025.

IEEE Computer Society. Ieee standard for local and metropolitan area networks - virtual bridged local area networks. **IEEE Std 802.1Q-2018**, 2018. Acessado em: 12 out. 2025.

KUROSE, J. F.; ROSS, K. W. **Redes de Computadores e a Internet: Uma Abordagem Top-Down**. 8. ed. São Paulo: Pearson, 2021. 864 p. Acessado em: 06 set. 2025.

LAMPORT, L. Time, clocks, and the ordering of events in a distributed system. **Communications of the ACM**, ACM, v. 21, n. 7, p. 558–565, 1978. Acessado em: 06 set. 2025.

ODOM, W. **CCNA 200-301 Official Cert Guide**. 1. ed. [S.l.]: Cisco Press, 2019. 1440 p. Acessado em: 12 out. 2025.

POSTEL, J. **RFC 793: Transmission Control Protocol**. 1981. Acessado em: 06 set. 2025. Disponível em: <<https://tools.ietf.org/rfc/rfc793.txt>>.

TANENBAUM, A. S.; STEEN, M. V. **Sistemas Distribuídos: Princípios e Paradigmas**. 2. ed. São Paulo: Pearson, 2016. 672 p. Acessado em: 06 set. 2025.