

Conceitos, Funções e Tipos de Sistemas Operacionais

Prof. Me. Deivison S. Takatu

deivison.takatu@fatec.sp.gov.br

Sumário

- Introdução aos Sistemas Operacionais;
- Tipos de Sistemas Operacionais;
- Controle de Versão com Git;
- Boas Práticas de Versionamento;
- Utilização de IDE;
- Atividade.

Sistemas de Grande Porte (Mainframes)

Projetados para alta capacidade de E/S e carga massiva de transações. Operam com processamento em lote e tempo compartilhado para muitos usuários simultâneos.



Características principais

- Alta confiabilidade e disponibilidade
- Processamento de lotes e transações (TPS)
- Segurança e integridade de dados
- Uso típico: bancos, grandes varejistas, processamento massivo

Sistemas de Grande Porte (Mainframes)

- **Exemplos**

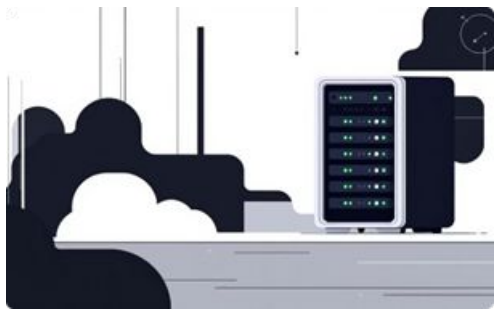
- OS/360
- OS/390
- Linux (em ambientes de mainframe)
- Variantes UNIX

- **Mainframes continuam sendo utilizados, especialmente como:**

- - Servidores web de grande escala
- - Plataformas para e-commerce
- - Sistemas bancários de alta criticidade

Sistemas Operacionais de Servidor

Focados em atender múltiplos usuários e serviços via rede: web, arquivos, bancos de dados e autenticação. Projetados para estabilidade, escalabilidade e compartilhamento de recursos.



Linux

Ampla adoção em servidores, flexibilidade e vasto ecossistema de serviços.



Windows Server

Integração com Active Directory e serviços empresariais.

Sistemas de Multiprocessadores

Suportam múltiplas CPUs ou muitos núcleos para processamento paralelo. Requerem mecanismos avançados de escalonamento e sincronização para aproveitar o paralelismo e evitar condições de corrida.



Desafios e soluções

- Escalonamento: balancear carga entre núcleos
- Sincronização: locks, semáforos, algoritmos lock-free
- Coerência de cache e comunicação entre núcleos
- Aplicações: servidores de alto desempenho, scientific computing

Sistemas de Computadores Pessoais

Orientados a um único usuário com interface gráfica, suporte a multiprogramação e ampla compatibilidade de aplicações. Foco em usabilidade e suporte multimídia.

Windows

Ampla compatibilidade com software de produtividade e jogos.

macOS

Integração hardware-software e foco em experiência do usuário.

Linux

Opção para power users e desenvolvimento; personalizável.



Sistemas Operacionais Portáteis

Principais aspectos

- Gerenciamento agressivo de energia
- API para sensores (GPS, acelerômetro, câmera)
- Segurança por permissões
- Exemplos: Android, iOS

Forte integração com hardware; Distribuição de aplicativos via lojas oficiais;
Grande foco em segurança e sandboxing



Sistemas Embarcados

Executam em dispositivos dedicados com recursos limitados; software frequentemente armazenado em ROM/flash. Usuário normalmente não instala ou altera software.



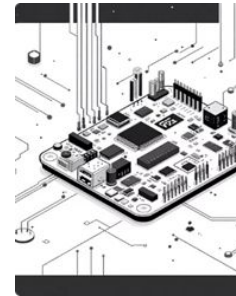
Aplicações domésticas

Micro-ondas, Smart TVs
resposta imediata e
interface simples.



Automotivo

Controle de motor,
infotainment; requisitos
de segurança críticos.



Sistemas embarcados sofisticados

Embedded Linux, QNX,
VxWorks — quando é
necessário mais
flexibilidade.

Sistemas de Nós Sensores

Dispositivos muito pequenos e com bateria limitada, comunicando-se sem fio e orientados a eventos. Projetos priorizam baixo consumo e protocolos leves.

Uso: monitoramento ambiental, vigilância militar, agricultura de precisão. Exemplos de SO: TinyOS, Contiki — otimizados para energia e eventos.



Sistemas de Tempo Real

01

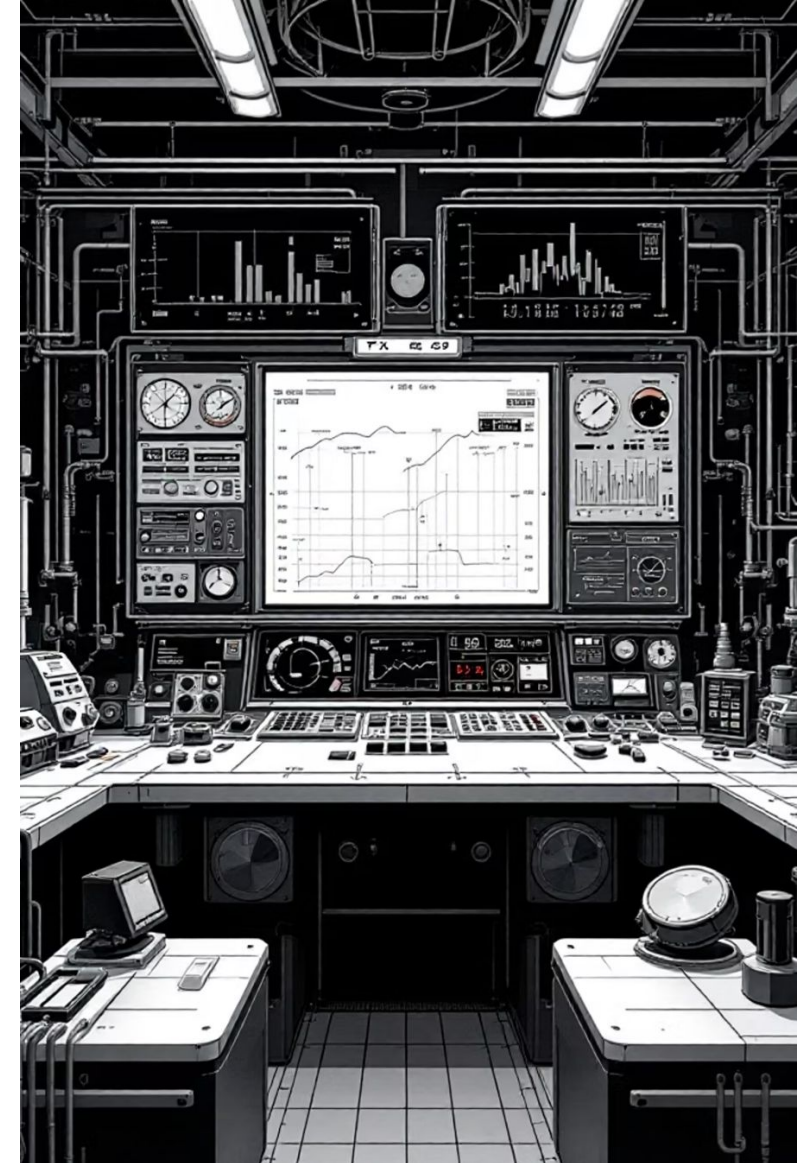
Hard Real-Time

Falha ao perder prazo pode causar desastre (ex.: controle de voo, segurança)

02

Soft Real-Time

Degradação aceitável (ex.: streaming de mídia, interações multimídia)



Sistemas de Cartões Inteligentes

Desafios e características

- Recursos restritos → gerenciamento fino de memória
- Segurança: criptografia, autenticação e resistência a ataques físicos
- Multiprogramação limitada: isolamento por applets

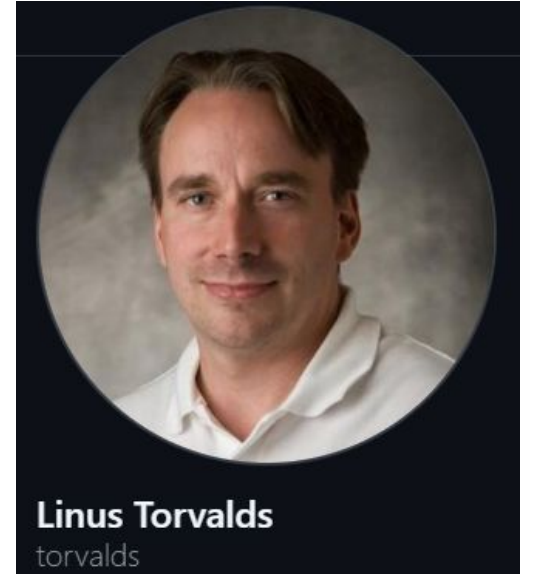


Ao final: diferenciar tipos de SO, reconhecer aplicações e entender exigências de projeto.



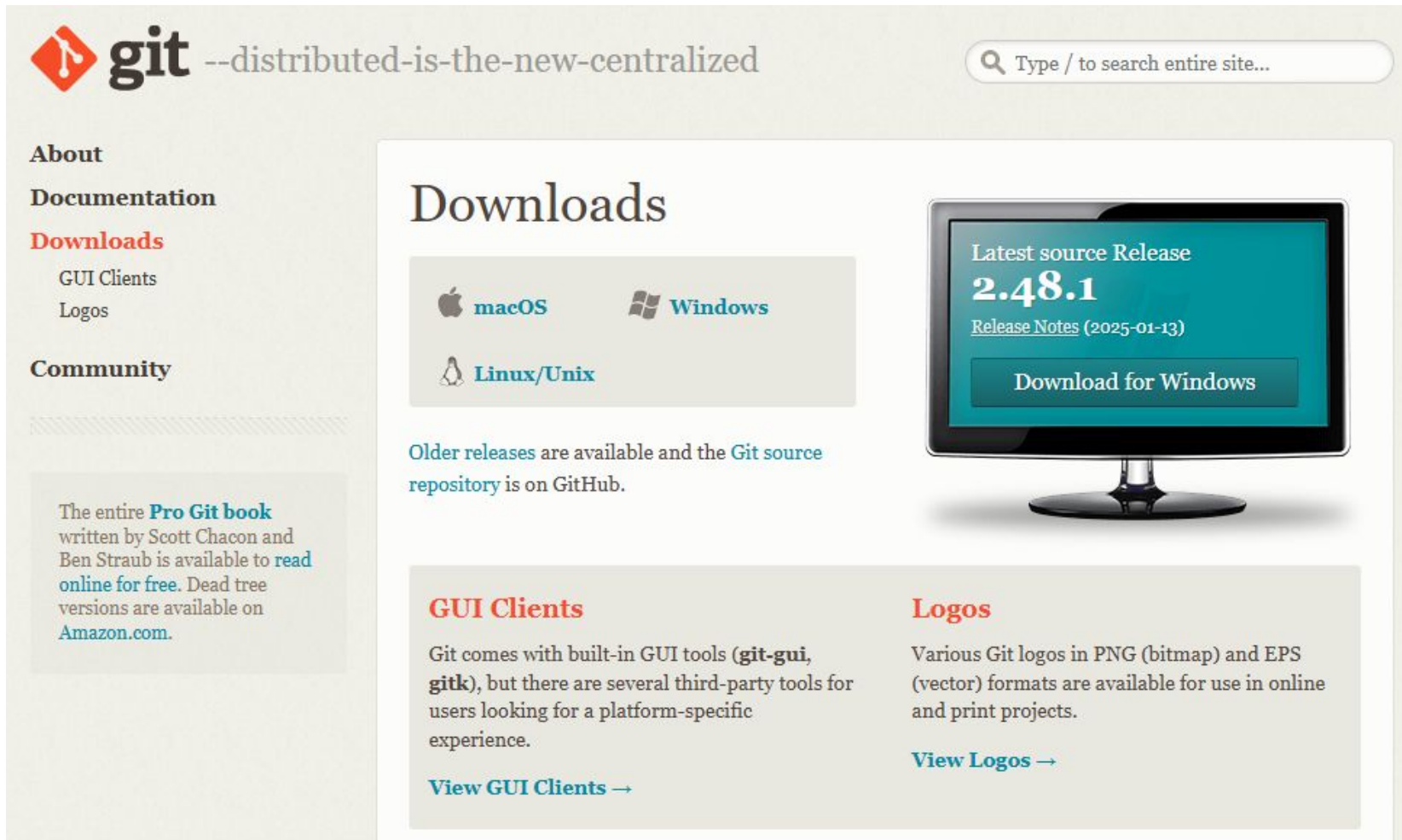
Introdução ao Git

- O que é Git?
 - Sistema de controle de versão de arquivos.
 - Instalado no computador e utilizado via linha de comando.
- O que o Git pode fazer?
 - Sincroniza com repositórios online, permitindo baixar e enviar código.
 - Registra versões do projeto, possibilitando acompanhar mudanças e restaurar versões anteriores.



Fonte: <https://github.com/torvalds>

Instalando o Git



The screenshot shows the Git website's Downloads page. At the top left is the Git logo and the tagline "--distributed-is-the-new-centralized". A search bar is at the top right. The left sidebar contains links for "About", "Documentation", "Downloads" (highlighted), "GUI Clients", "Logos", and "Community". Below these is a box about the "Pro Git book". The main content area has a "Downloads" heading, followed by buttons for macOS, Windows, and Linux/Unix. A monitor graphic displays the "Latest source Release 2.48.1" with a "Download for Windows" button. Below the download buttons, text states that older releases are available and the source repository is on GitHub. At the bottom, there are sections for "GUI Clients" and "Logos", each with a "View" link.

git --distributed-is-the-new-centralized

Search: Type / to search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloads

macOS Windows Linux/Unix

Older releases are available and the Git source repository is on GitHub.

Latest source Release
2.48.1
[Release Notes](#) (2025-01-13)
[Download for Windows](#)

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients](#) →

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos](#) →

Fonte: <https://git-scm.com/downloads>

Instalando o Git

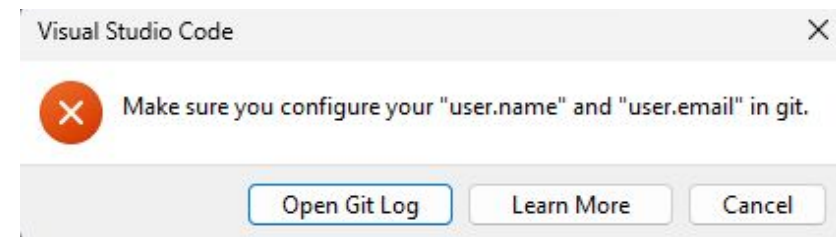
- Escolha a versão do seu sistema operacional
- Baixe e execute o instalador
- Siga as etapas "Next > Next > Install"
- Para testar a instalação:
 - Abra o Prompt de Comando
 - Digite `git --version`
 - Se aparecer a versão instalada, está correto



Fonte: Elaboração própria.

Instalando o Git

- Se solicitado, configure o usuário e e-mail no CMD com os comandos:
 - `git config --global user.name "<Nome>"`
 - `git config --global user.email "<Email>"`
- Abra o VS Code
- Vá até a aba de Controle de Código-Fonte (terceiro ícone à esquerda)
- Se solicitado, instale o Git
- Feche e reabra o VS Code



Fonte: Elaboração própria.

Criando um Repositório no VS Code

- Crie uma pasta no seu computador
- Abra a pasta no VS Code
- Crie um arquivo .md
- Vá até a aba de Controle de Código-Fonte
- Clique em Inicializar Repositório
- Adicione uma mensagem de commit e clique em Confirmar (Commit)

Publicando no GitHub

- Clique em Publicar Branch
- Faça login no GitHub
- Escolha entre repositório público ou privado
- Após a publicação, o código estará acessível no GitHub
- Acesse seu perfil no GitHub
- Vá até "Repositórios"
- Seu projeto deve estar listado
- Clique no repositório e verifique os arquivos

Boas Práticas Git

- Commits pequenos e frequentes:
 - Facilita a identificação de problemas e a reversão de mudanças.
- Mensagens de commit claras:
 - Descreva o que foi alterado e por quê.
 - Uso de branches:
 - Mantenha a branch principal estável.
 - Use branches para novas funcionalidades e correções.
- Testes automatizados:
 - Garanta que o código funcione corretamente antes de fazer merge.

Atividade

1. No computador, deverá ser realizada a configuração de integração entre a IDE instalada no sistema operacional e a conta pessoal no GitHub, utilizando o Git como ferramenta de versionamento. O aluno deverá garantir que a autenticação esteja funcional e que a IDE reconheça corretamente o repositório remoto, permitindo operações de commit, push e pull.

Atividade

2. Em seguida, deverá ser criado um repositório de teste contendo ao menos um arquivo inicial do projeto. Após realizar alterações no arquivo, execute commit e sincronize com o GitHub. Depois, apague a pasta local do projeto no computador e utilize o comando `git clone` para clonar novamente o repositório remoto. Por fim, valide se o projeto foi restaurado corretamente.

Atividade

3. Por fim, realize uma pesquisa no Github e encontre 5 projetos de outros usuários. Utilizem o comando `git clone` para copiar os projetos nas máquinas pessoais e analisem o conteúdo contido em cada repositório.

Referências

- TANENBAUM, Andrew S.; BOS, Herbert. Sistemas Operacionais Modernos. 4. ed. São Paulo: Pearson, 2016.
- SILBERSCHATZ, Abraham; GALVIN, Peter B.; GAGNE, Greg. Fundamentos de Sistemas Operacionais. 9. ed. Rio de Janeiro: LTC, 2015.
- STALLINGS, William. Sistemas Operacionais: Conceitos e Projetos. 8. ed. São Paulo: Pearson, 2015.
- DENARDIN, G. W.; BARRIQUELLO, C. H. Sistemas Operacionais de Tempo Real e sua Aplicação em Sistemas Embarcados. Porto Alegre: Editora da UFRGS, 2014.
- AWASTHI, A.; RAWAT, V. Ramificação e Tarefas do Sistema Operacional. Edições Nosso Conhecimento, 2023.
- DOWNEY, Allen B. Think OS: A Brief Introduction to Operating Systems. Green Tea Press, 2015.
- RED HAT. Red Hat Enterprise Linux – System Administration Guide. Documentação Oficial.
- DOCKER INC. Docker Documentation. Documentação Oficial. Disponível em: <https://docs.docker.com>

Conceitos, Funções e Tipos de Sistemas Operacionais

Prof. Me. Deivison S. Takatu

deivison.takatu@fatec.sp.gov.br