

# Versionamento de Código com Git e GitHub

**Elidiana Andrade**

Desenvolvedora Front-end

**@elidianaandrade**

# Sobre Mim



Desenvolvedora Front-end  
[in](https://www.linkedin.com/in/elidianaandrade)/elidianaandrade  @elidianaandrade



Pós-Graduanda em Desenv. de Software  
Cursando Especialização em Desenvolvimento  
de Software com Metodologias Ágeis



Gosta de Compartilhar Conhecimento  
 users/elidianaandrade  @casafullstack

# Sobre Mim



@elicosmaker



# Sobre Mim



*We live in a society exquisitely dependent on science and technology, in which hardly anyone knows anything about science and technology.  
**This is a clear prescription for disaster.***

SAGAN, C. “*Why We Need To Understand Science*”. 1990.  
*The Skeptical Inquirer.*

# Objetivo Geral

**Introduzir ao versionamento de código com Git e GitHub.**



# Pré-requisitos

- ✓ Computador com acesso à internet
- ✓ Vontade de aprender
- ✓ Fechar as outras abas XD

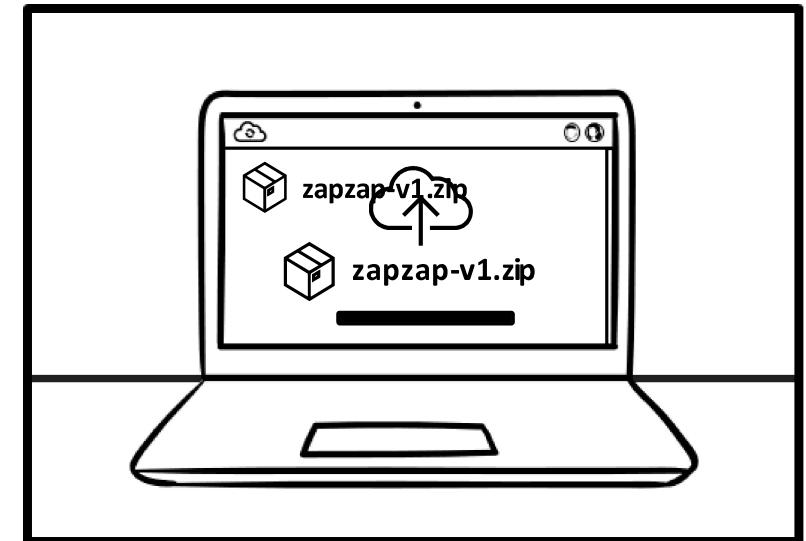
# Percorso

- ❑ Visão Geral do Curso e Ferramentas
- ❑ Instalação, Configuração e Autenticação
- ❑ Primeiros Passos com Git e GitHub
- ❑ Dicas e Materiais de Apoio

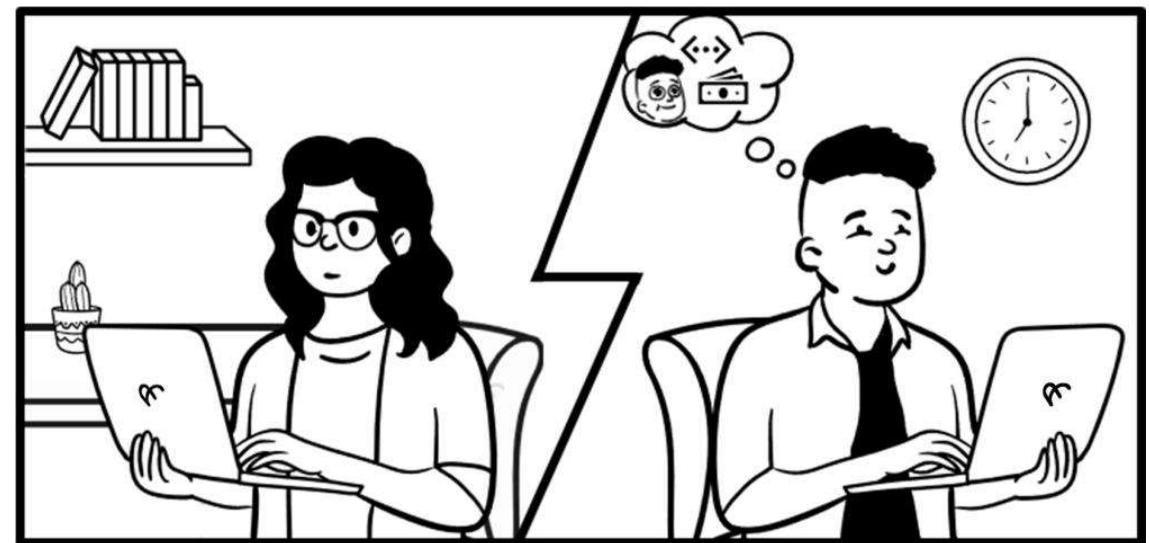
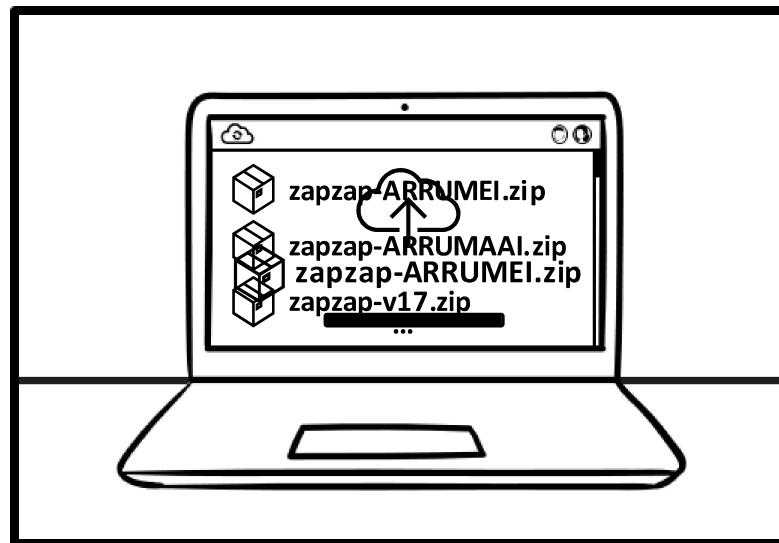
# Visão Geral do Curso e Ferramentas

Versionamento de Código, Git e GitHub

# O que é Versionamento de Código?



# O que é Versionamento de Código?



# O que é Versionamento de Código?



# Sistemas de Controle de Versão

**Controlam as versões de um arquivo ao longo do tempo.**



Registra o histórico de atualizações de um arquivo;



Gerencia quais foram as alterações, a data, autor, etc.;



Organização, controle e segurança.

# Tipos de Sistemas de Controle de Versão

Dentre os Sistemas de Controle de Versão (VCS), temos:



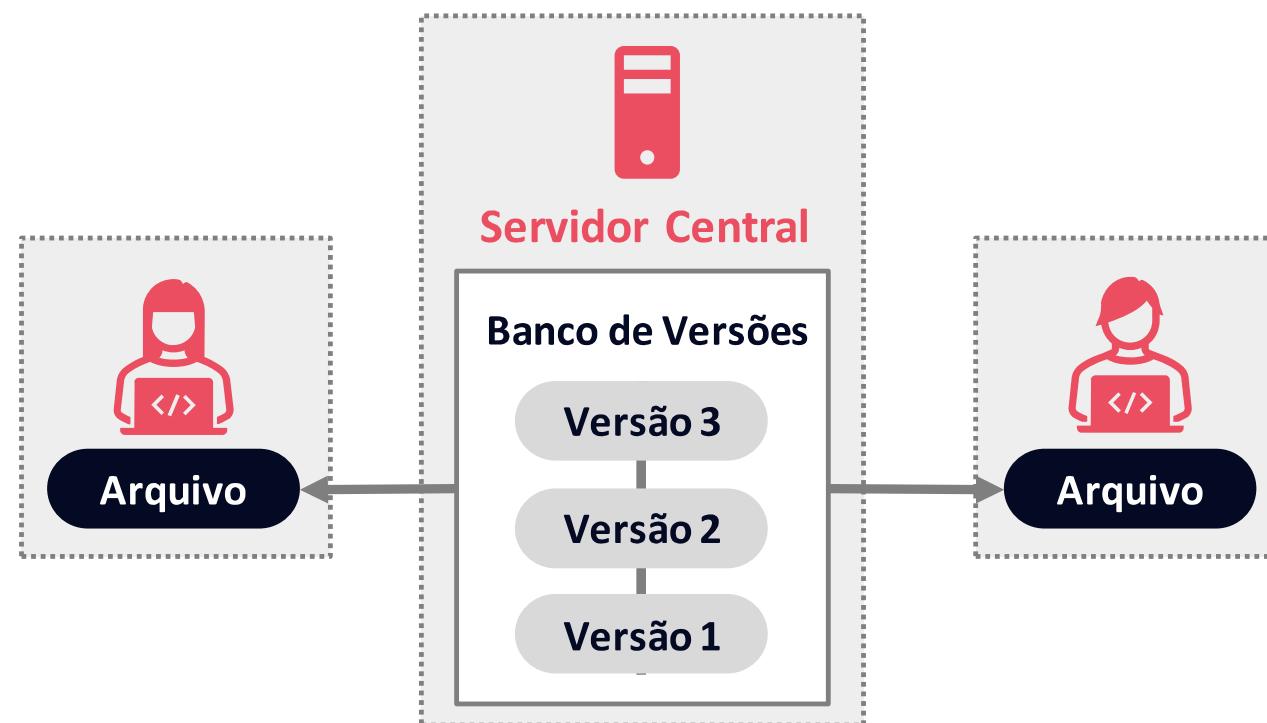
VCS Centralizado (CVCS)  
Ex.: **CVS**, Subversion.



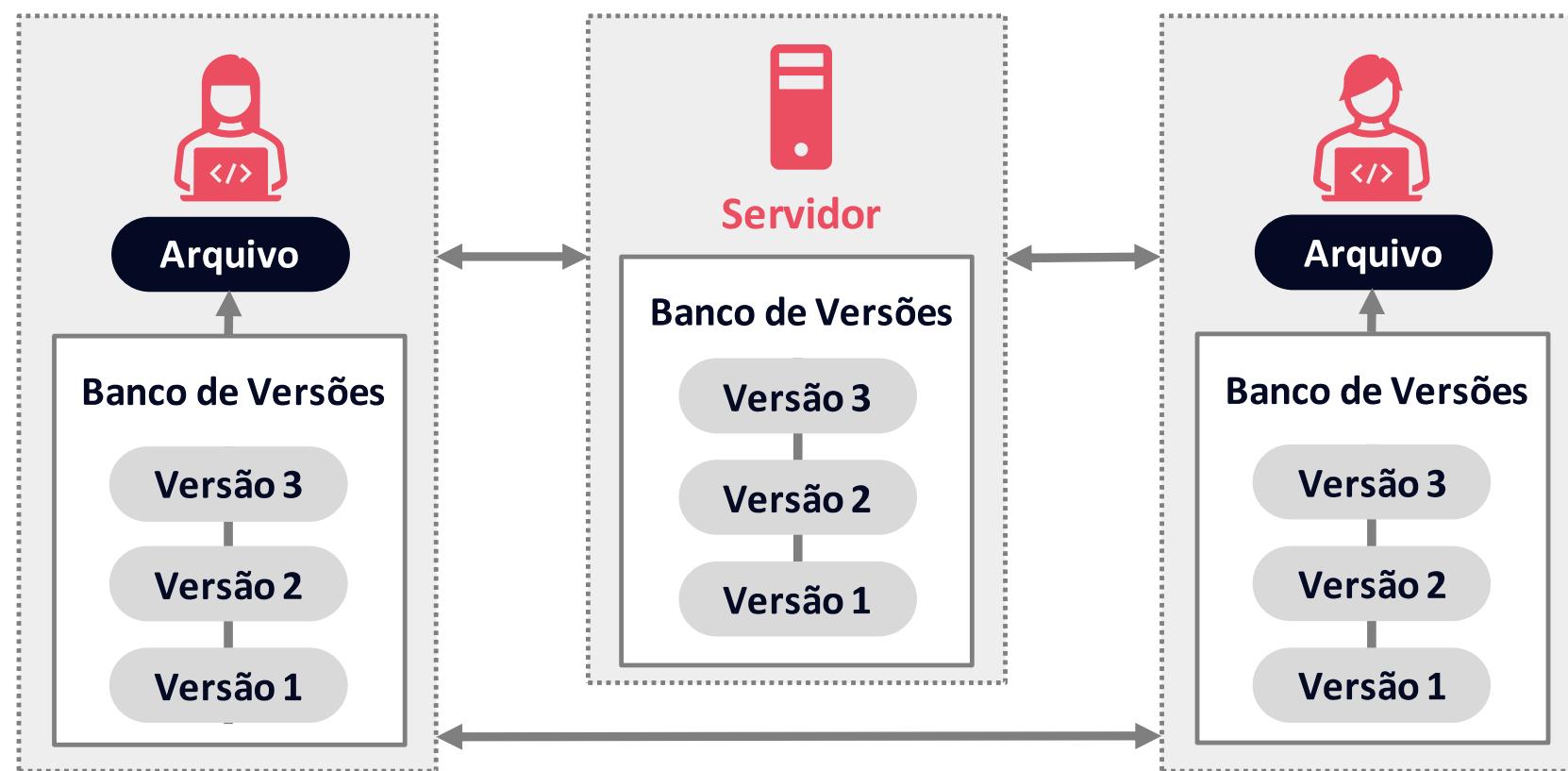
VCS Distribuído (DVCS)  
Ex.: **Git**, **Mercurial**.



# VCS Centralizado (CVCS)



# VCS Distribuído (DVCS)



# VCS Distribuído (DVCS)

**Clona o repositório completo, o que inclui o histórico de versões.**



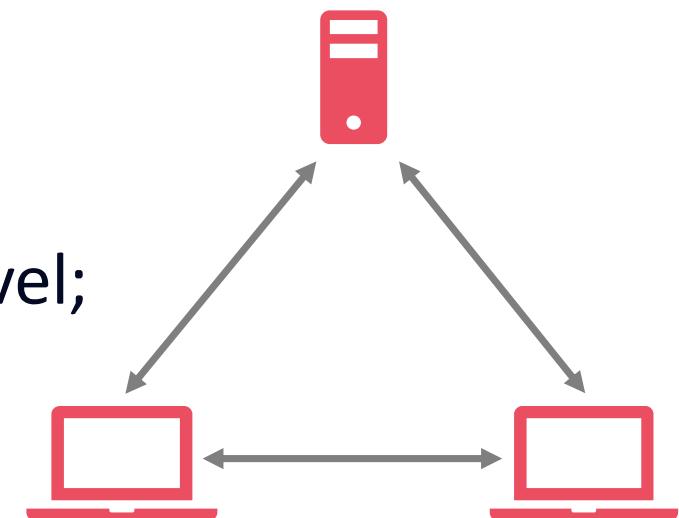
Cada clone é como um backup;



Possibilita um fluxo de trabalho flexível;



Possibilidade de trabalhar sem conexão à rede.



# O que é Git?

Sistema de Controle de Versão Distribuído.



- {...} Gratuito e Open Source (Código Aberto);
- ↑ Ramificações (branching) e fusões (merging) eficientes;
- ✓ Leve e rápido.

## Hands On!



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

# Breve Histórico do Git

**2002 →** O projeto do **núcleo (kernel) do Linux**, que é open source, começa a utilizar o **BitKeeper**, um **DVCS proprietário**;

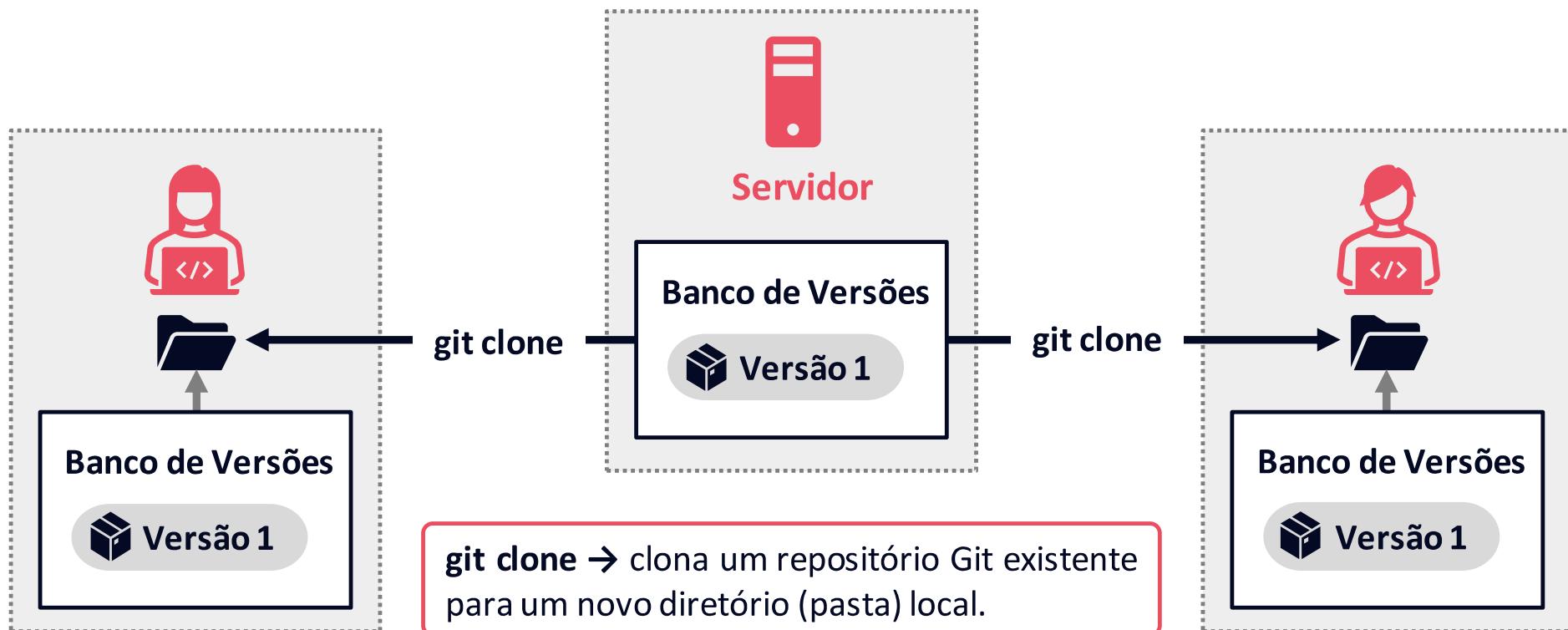
**2005 →** Após conflitos com a comunidade, o BitKeeper **rescinde a licença gratuita**. O que leva a **Linus Torvalds**, o criador do Linux, e sua equipe, a desenvolverem sua própria ferramenta, o **Git**.

# Fluxo Básico no Git

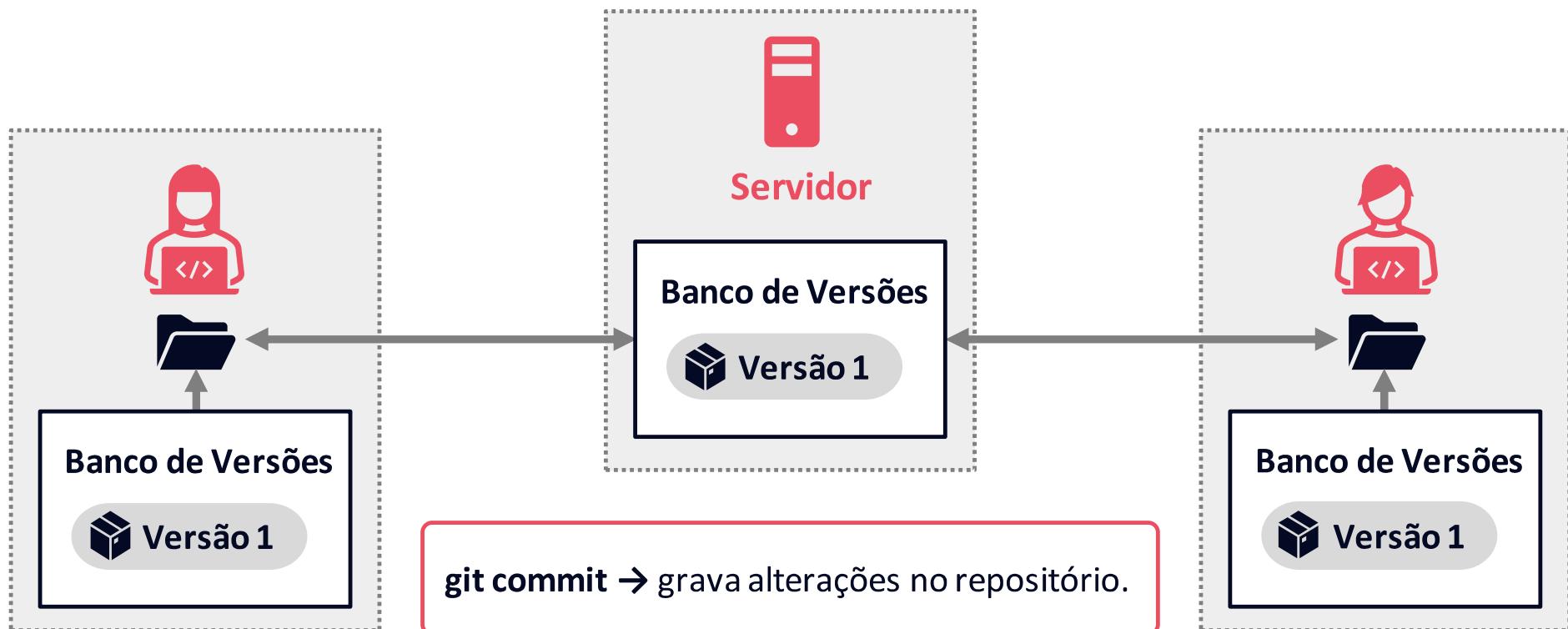


**git clone** → clona um repositório Git existente para um novo diretório (pasta) local.

# Fluxo Básico no Git



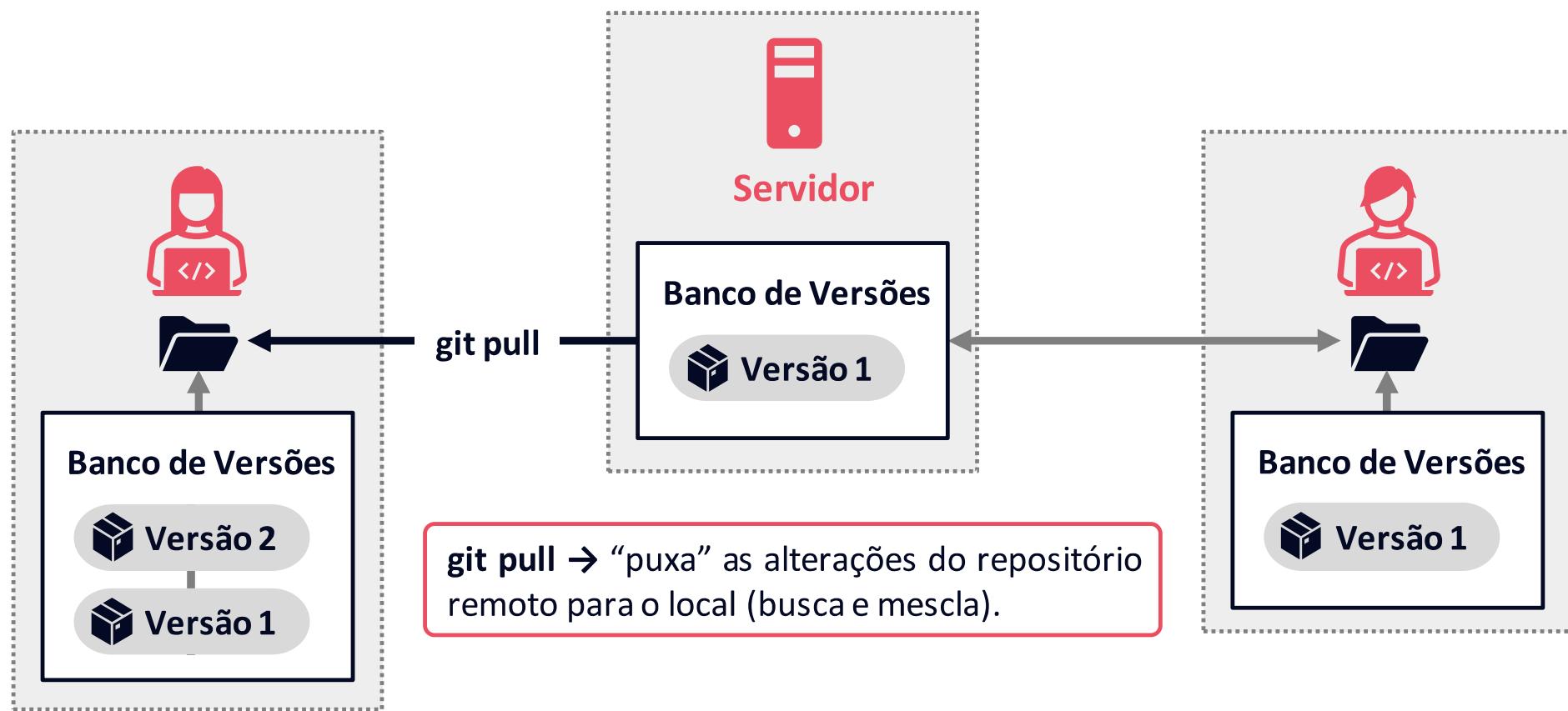
# Fluxo Básico no Git



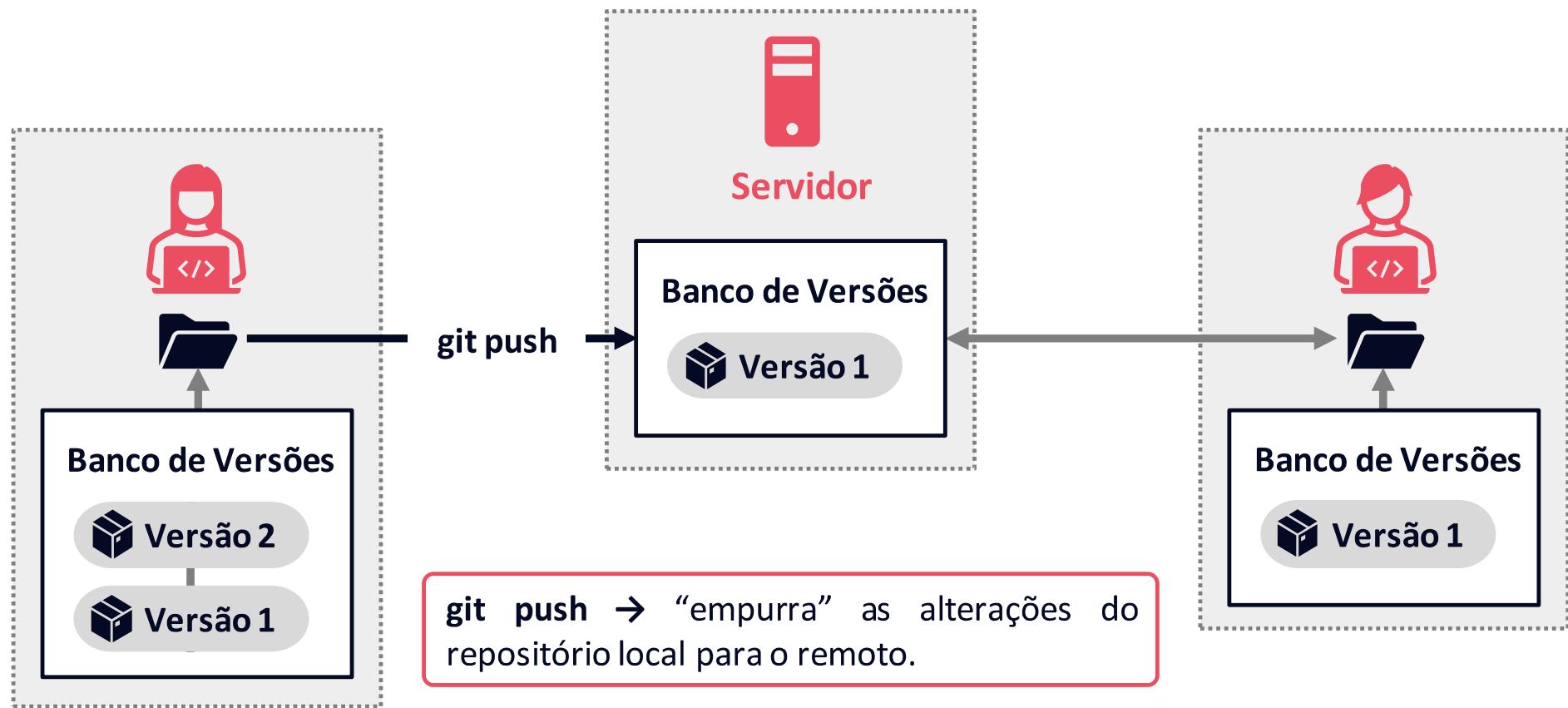
# Fluxo Básico no Git



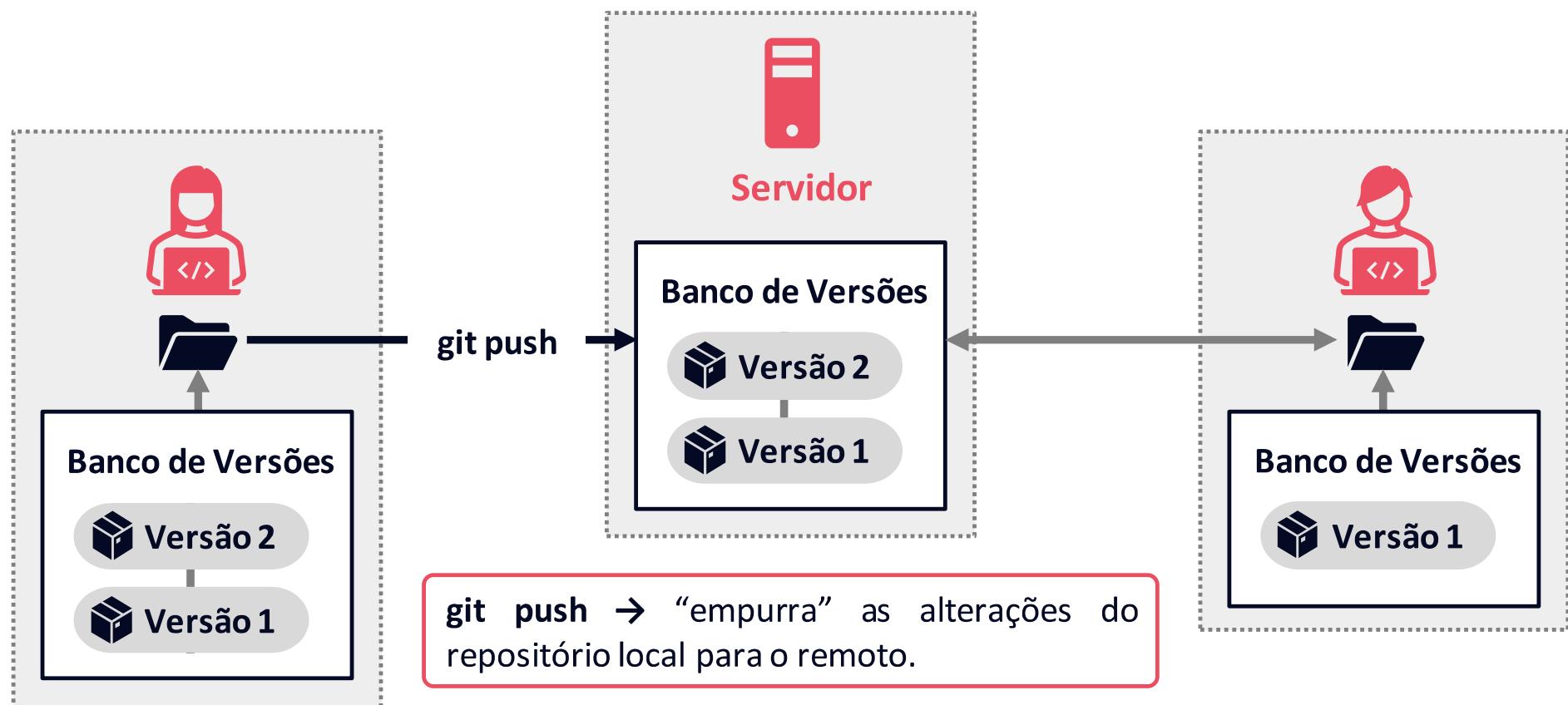
# Fluxo Básico no Git



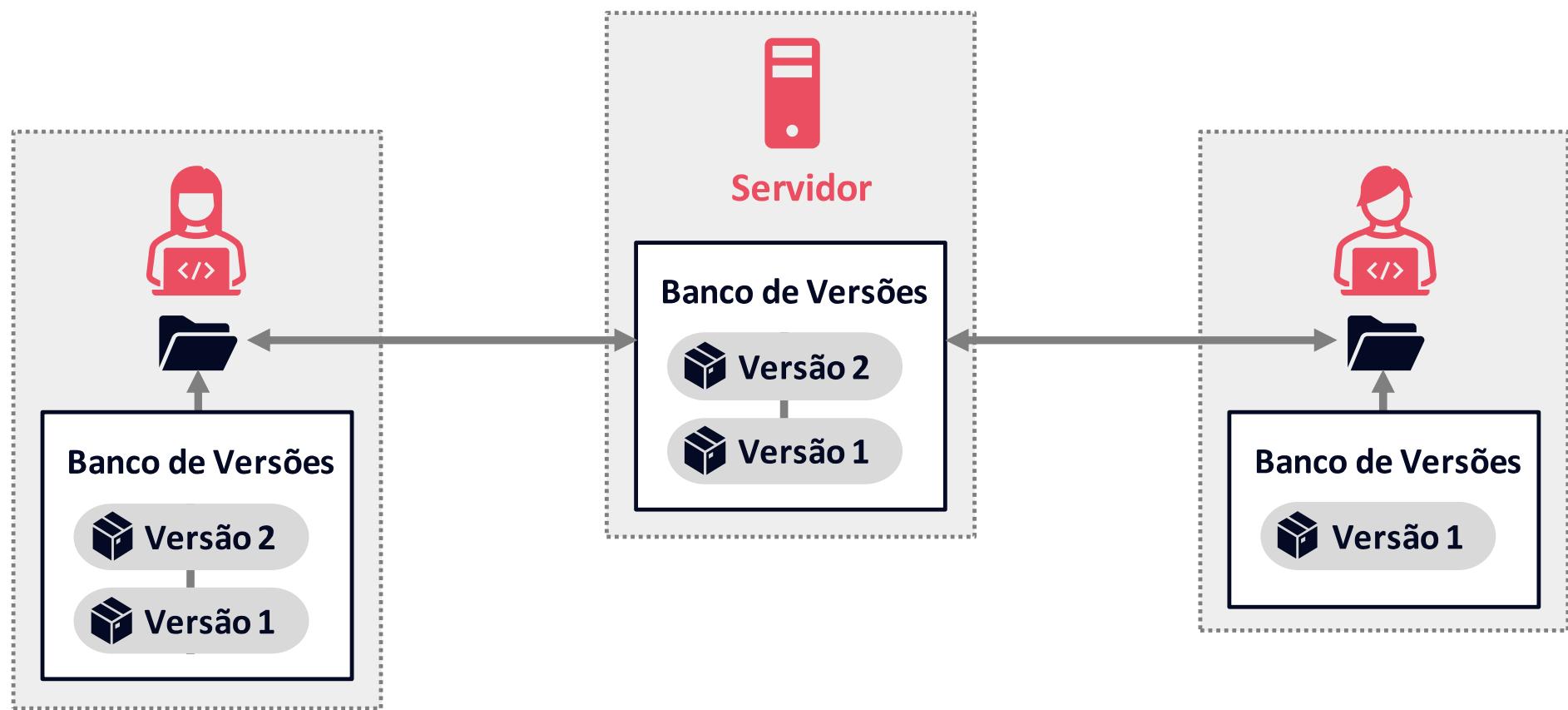
# Fluxo Básico no Git



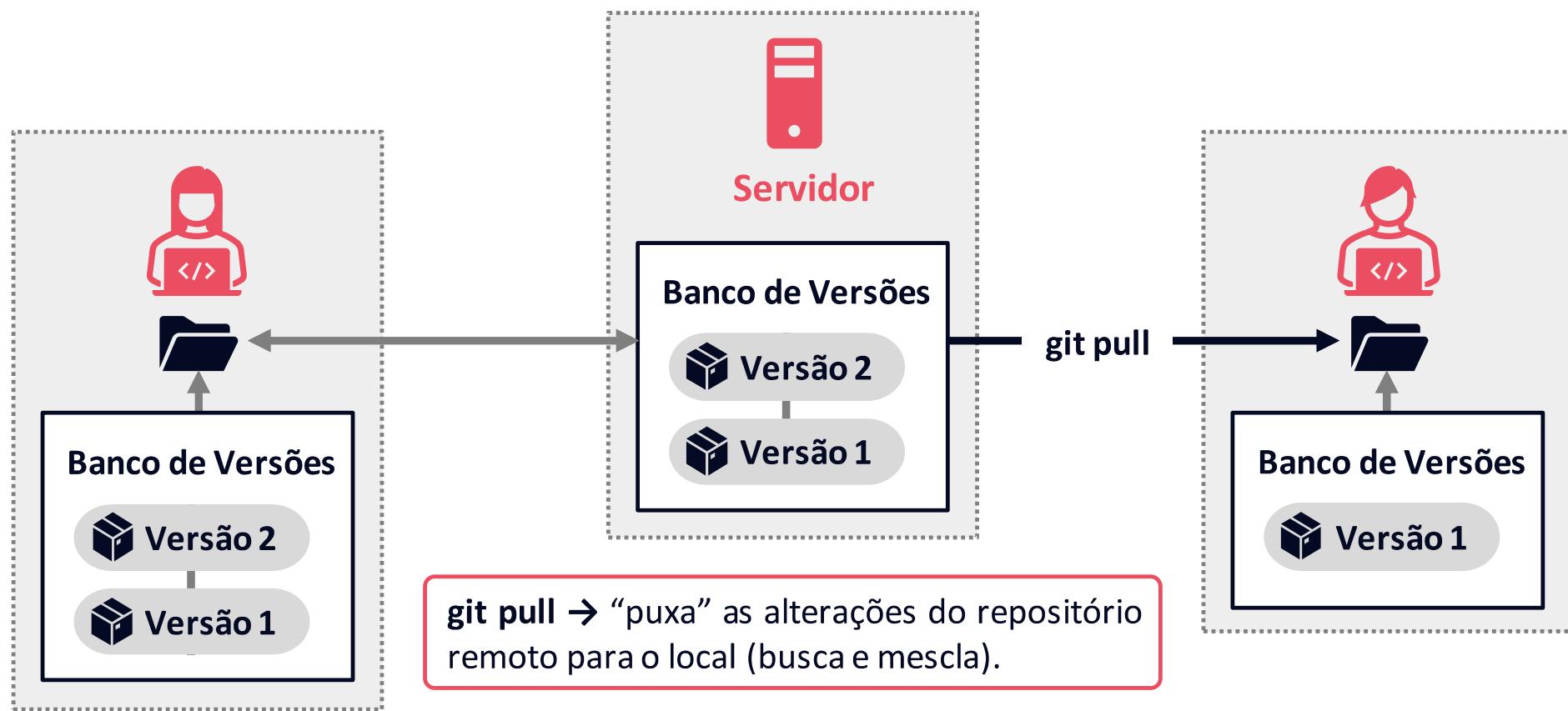
# Fluxo Básico no Git



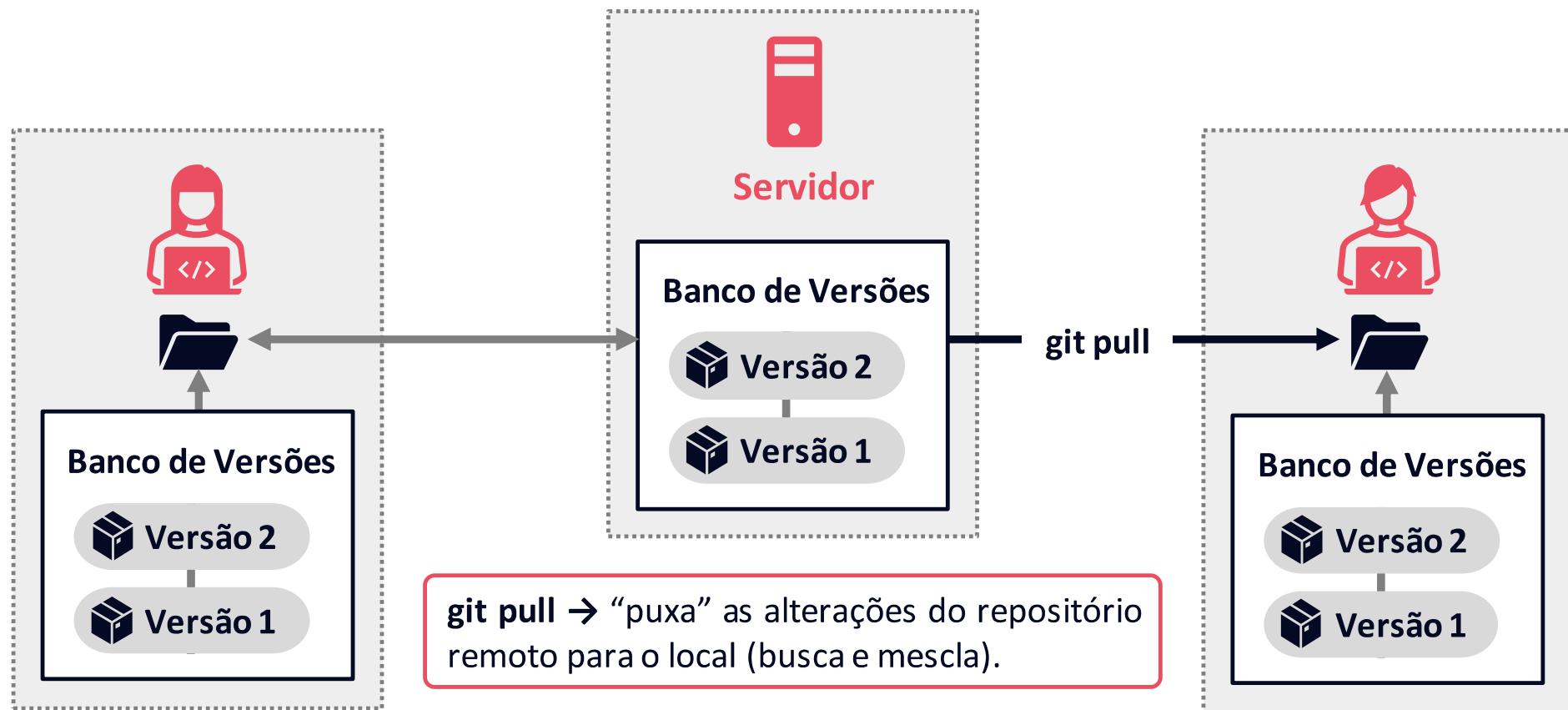
# Fluxo Básico no Git



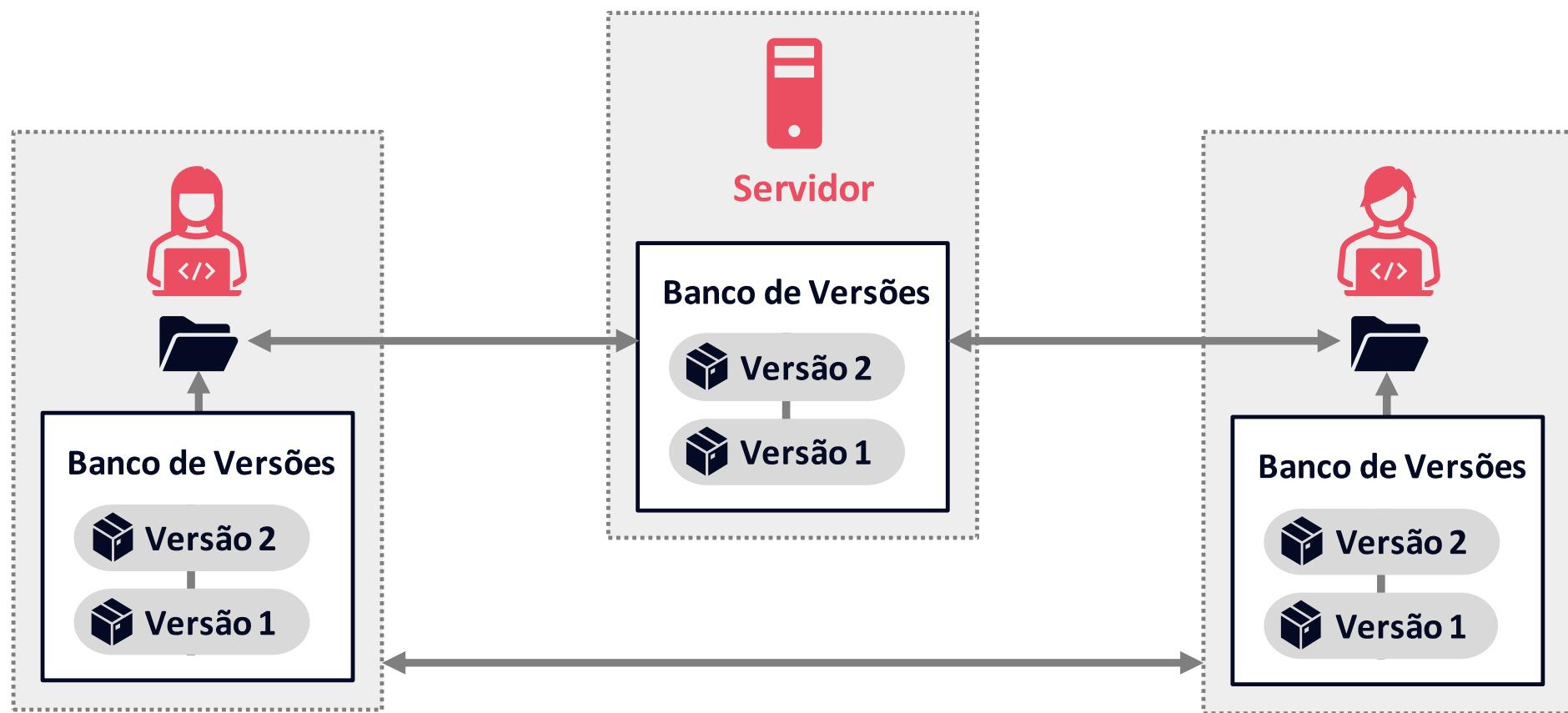
# Fluxo Básico no Git



# Fluxo Básico no Git



# Fluxo Básico no Git



# O que é GitHub?

Plataforma de hospedagem de código para controle de versão com Git, e colaboração.



Comunidade ativa;



Utilizado mundialmente;



Mascote “Octocat”.

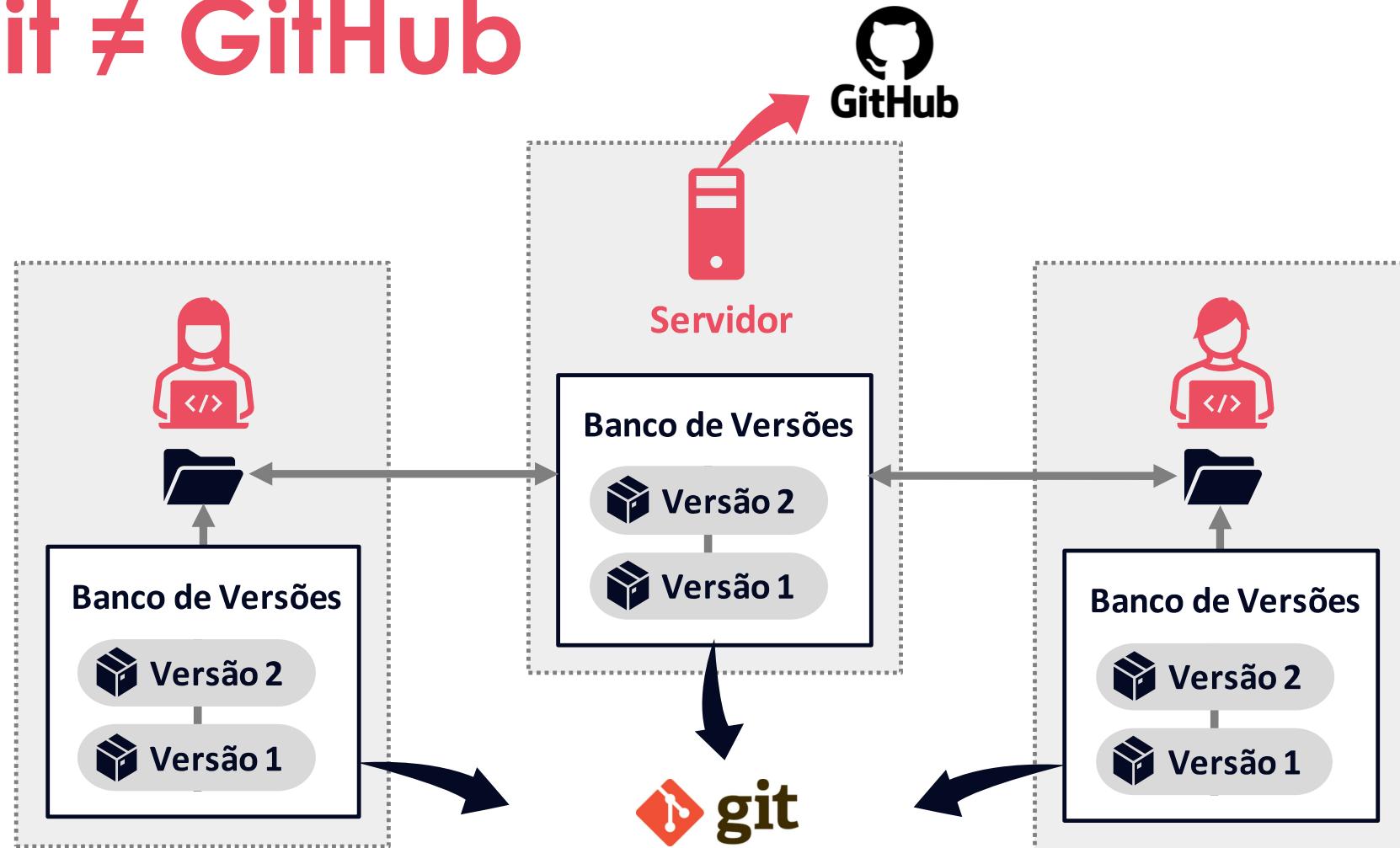


# Breve Histórico do GitHub

**2008 →** Desenvolvido por Chris Wanstrath, J. Hyett, Tom Preston-Werner e Scott Chacon.

**2018 →** Vítima de um dos maiores **ataques de DDoS** (ataque distribuído de negação de serviço); Comprado pela **Microsoft Corporation** por **US \$ 7,5 bilhões**.

# Git ≠ GitHub



# Autenticação de Dois Fatores

Acesse sua conta do GitHub e vá em **Settings > Password and authentication > Two-factor authentication > Authenticator app**

- 1** Leia o QR Code através do aplicativo autenticador (ex.: Microsoft Authenticator) e insira o código no GitHub;
- 2** Salve os códigos de recuperação;
- 3** Autenticação ativada! ;D

## Hands On!



<https://github.com/> 

# Instalação, Configuração e Autenticação

Instalando e configurando o Git, e  
autenticando o GitHub via Token e Chave SSH

## Hands On!



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



# Instalando o Git no Windows

- 1** Acesse <<https://git-scm.com/download/win>>;
- 2** Faça o download do instalador e execute;
- 3** Aceite a licença e clique em “Next”, e siga configurando como desejar<sup>1</sup> e clicando em “Next”;
- 4** Finalize clicando em “Install”, e “Finish”.

<sup>1</sup>Em "Select Components", deixe as opções "Git Bash Here" e "Git GUI Here" marcadas.

## Hands On!



<https://git-scm.com/download/win>



# Instalando o Git no Linux (Ubuntu)

- 1 Confira a doc.: <<https://git-scm.com/download/linux>>;
- 2 Instale a última versão estável do Git:

```
# add-apt-repository ppa:git-core/ppa
```

```
# apt update
```

```
# apt install git
```

## Hands On!



<https://git-scm.com/download/linux>



# Instalando o Git no macOS

- 1 Confira a doc.: <<https://git-scm.com/download/mac>>;
- 2 Instale o Homebrew: <<https://brew.sh/>>;
- 3 Instale o Git:

```
$ brew install git
```

# Configurando o Git

```
$ git config --list
```

1

Configurando seu nome de usuário e e-mail (globalmente):

```
$ git config --global user.name "Nome Sobrenome"  
$ git config --global user.email seuemail@email.com
```

2

Configurando o nome da Branch Padrão:

```
$ git config --global init.defaultBranch main
```

# Autenticando via Token



Para gerar um Token, acesse sua conta no GitHub, e no menu superior direito clique em ***Settings > Developer settings > Tokens (classic) > Generate new token.***

The screenshot shows the GitHub developer settings interface. The top navigation bar includes the GitHub logo, a search bar, and links for Pull requests, Issues, Codespaces, Marketplace, and Explore. Below this, the 'Settings / Developer settings' path is visible. On the left, a sidebar lists GitHub Apps, OAuth Apps, Personal access tokens (classic), Fine-grained tokens (Beta), and Tokens (classic). The 'Personal access tokens (classic)' section is the active tab, displaying instructions to generate a token for API access. A 'Generate new token' button is located in the top right corner of this section.

# Armazenando Credenciais

**Você pode armazenar suas credenciais para reduzir o número de vezes que você deve digitar seu nome de usuário ou senha:**



Salvando no cache:

```
$ git config --global credential.helper cache
```



Ou permanentemente:

```
$ git config --global credential.helper store
```

Veja mais na doc.: <https://git-scm.com/book/pt-br/v2/Git-Tools-Credential-Storage>

# Autenticando via Chave SSH



Para adicionar uma Chave SSH, acesse sua conta no GitHub, e no menu superior direito clique em **Settings** > **SSH and GPG keys** > **New SSH key**.

The screenshot shows the GitHub settings interface. On the left, there's a sidebar with various account management options like Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and plans, Emails, Password and authentication, Sessions, and SSH and GPG keys. The SSH and GPG keys option is highlighted with a blue bar at the bottom. The main content area has three sections: 'SSH keys' (with a 'New SSH key' button), 'GPG keys' (with a 'New GPG key' button), and 'Vigilant mode' (with a checkbox for 'Flag unsigned commits as unverified').

# Primeiros Passos com Git e GitHub

Do primeiro repositório à manipulação de  
branches

# Criando e Clonando Repositórios

Existem duas formas de obter um repositório Git na sua máquina:

- 1 Transformando um diretório local que não está sob controle de versão, num repositório Git;
- 2 Clonando um repositório Git existente.

# Criando um Repositório Local

Acesse a pasta que deseja transformar em um repositório Git pelo terminal ou clique no atalho em “Git Bash Here”:

1

Inicie o Git no diretório escolhido:

```
$ git init
```

2

Conecte o repositório local com o repositório remoto:

```
$ git remote add origin  
https://github.com/username/nome-do-repositorio.git
```

# Clonando um Repositório

Para clonar um repositório no Git, acesse seu repositório no GitHub e siga os próximos passos:

- 1 Em “Code”, copie o código HTTPS ou SSH (a depender de como autenticou sua conta) do repositório no GitHub;
- 2 Abra o GitBash, insira o comando git clone e cole o conteúdo copiado para cloná-lo:

```
$ git clone https://github.com/username/nome-do-repositorio
```

# Criando um Repositório Remoto

Acesse a sua conta do GitHub, clique no “+” no canto superior direito, e em “New repository”:

- 1** Insira um nome (obrigatório), e a descrição (opcional);
- 2** Coloque uma breve descrição sobre o projeto, essa etapa é opcional;
- 3** Defina se o acesso será público ou privado;

# Criando um Repositório Remoto

Acesse a sua conta do GitHub, clique no “+” no canto superior direito, e em “New repository”:

4

Escolha como deseja inicializar seu repositório (se quiser vazio, deixe as opções desmarcadas)

5

Clique em “Create repository”, e pronto!

# Salvando Alterações no Repositório Local

## 1) Como criar um commit:

1

Adicione o conteúdo que deseja inserir no commit:

```
$ git add
```

2

Crie um commit e adicione uma mensagem descritiva:

```
$ git commit -m "message"
```

# Desfazendo Alterações no Repositório Local

## 1) Como alterar a mensagem do último commit:

```
$ git commit --amend
```

Alterando a mensagem sem abrir o editor:

```
$ git commit --amend -m "nova mensagem"
```

# Desfazendo Alterações no Repositório Local

## 2) Como desfazer um commit:

```
$ git reset
```

```
$ git reset --soft
```

```
$ git reset --mixed
```

```
$ git reset --hard
```

# Enviando Alterações para o Repositório Remoto

Como enviar as alterações do repositório local para o remoto:

```
$ git push
```

“Puxar” as alterações do repositório remoto para o local (busca e mescla).

```
$ git pull
```

# Trabalhando com Branches

De maneira simplista, uma Branch (em tradução, “Ramo”), é uma ramificação do seu projeto.

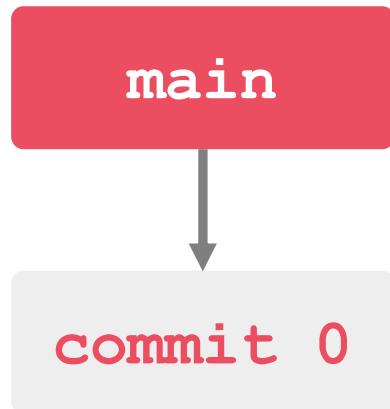


É um ponteiro móvel para um commit no histórico do repositório;

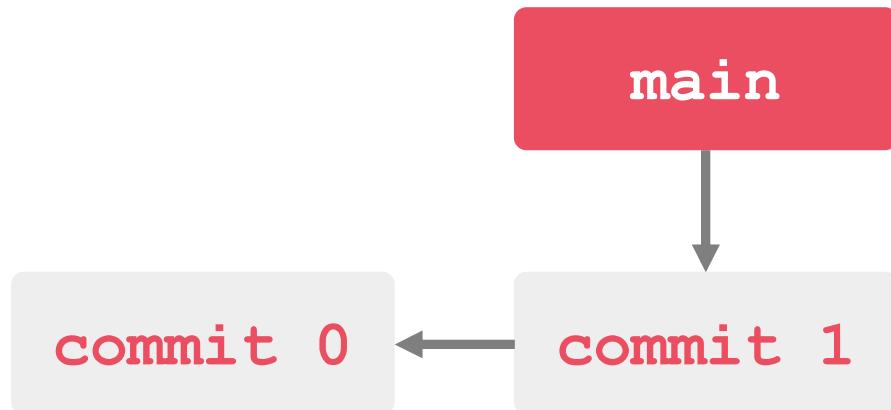


Quando você cria uma nova Branch a partir de outra existente, a nova se inicia apontando para o mesmo commit da Branch que estava quando foi criada.

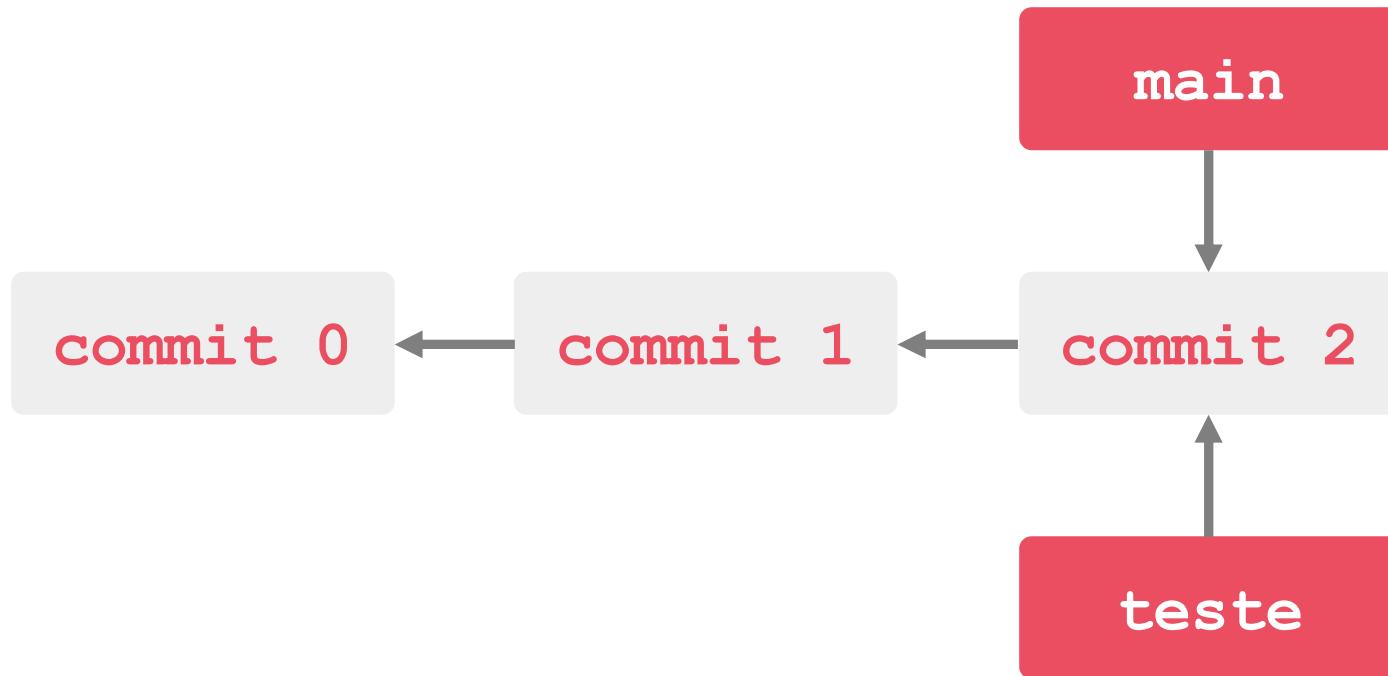
# Trabalhando com Branches



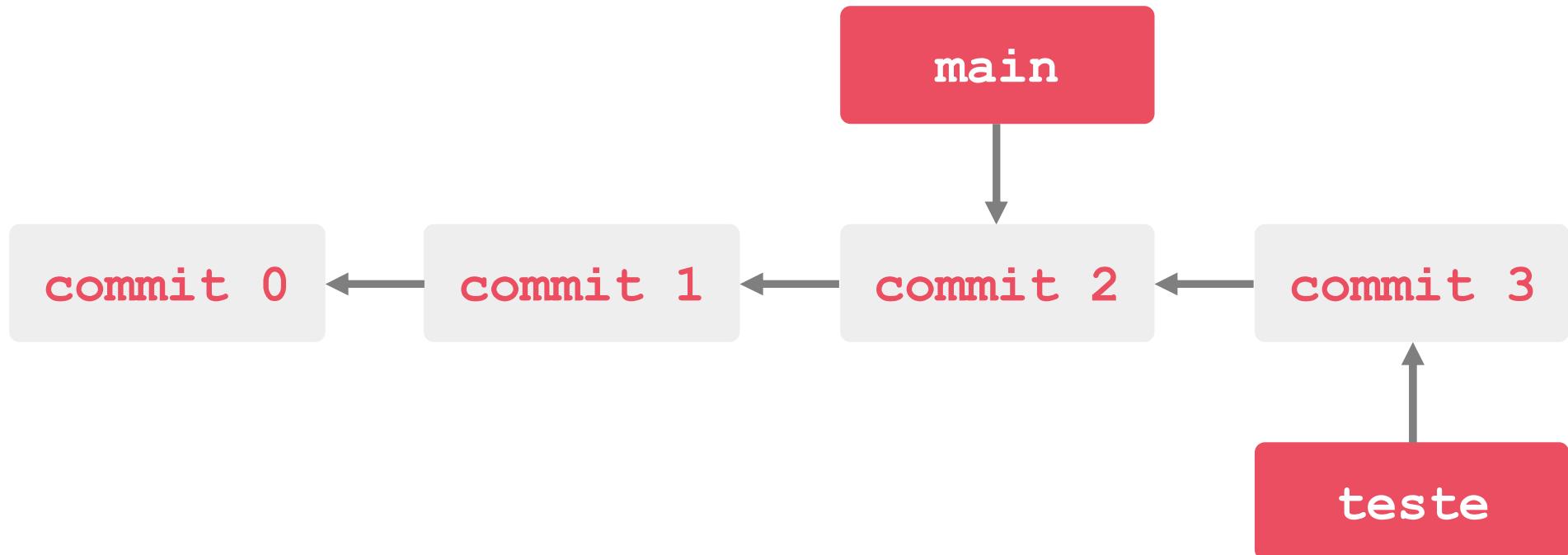
# Trabalhando com Branches



# Trabalhando com Branches



# Trabalhando com Branches



# Trabalhando com Branches

```
$ git branch
```

- Trocar de Branch e criar uma nova:

```
$ git checkout -b nova-branch
```

- Deletar uma Branch

```
$ git branch -d nome-da-branch
```

- Ver o último commit de cada Branch:

```
$ git branch -v
```



# Dicas e Materiais de Apoio

Links Úteis e Para Saber Mais

# Links Úteis

- Repositório no GitHub
- Documentação Git | Documentação GitHub
- Referências:
  - <https://git-scm.com/>
  - <https://docs.github.com/>
  - <https://github.blog/>

# Links Úteis

- Referências:
  - <https://github.blog/2020-12-15-token-authentication-requirements-for-git-operations/>
  - <https://github.blog/2018-03-01-ddos-incident-report/>
  - <https://news.microsoft.com/2018/06/04/microsoft-to-acquire-github-for-7-5-billion/>
  - <https://github.blog/2023-03-09-raising-the-bar-for-software-security-github-2fa-begins-march-13/>

# Para saber mais

- Tech Talk: Linus Torvalds on git: <https://youtu.be/4XpnKHJAok8>
- ProGit: <https://git-scm.com/book/en/v2>
- Markdown: <https://docs.github.com/pt/get-started/writing-on-github>
- Conventional Commits: <https://github.com/conventional-commits/conventionalcommits.org>

# Para saber mais

- Chocolatey: <https://community.chocolatey.org/packages/git>
- GitHub Desktop: <https://desktop.github.com/>
- GitFluence: <https://gitfluence.com/>
- My Octocat: <https://myoctocat.com/>
- GitHub Pages: <https://docs.github.com/en/pages/getting-started-with-github-pages>

# Dúvidas?

- > Fórum/Artigos
- > Comunidade Online ([Discord](#))

