ESCOLA DE TECNOLOGIA DA INFORMAÇÃO

INTRODUÇÃO AO GIT

Versionamento de códigos

Geucimar Briatore
geucimar@up.edu.br

Atualizado em 08/2024

Servidores com sistema de versionamento Git

O que é o Git?

- Git é um software de versionamento de linha de comando usado para criar versões de códigos ou documentos e voltar versões anteriores facilmente, assim como analisar o histórico;
- Git não é uma abreviação. Inicialmente o Git foi chamado *de "The stupid content tracker"* e posteriormente alterado para *"Fast, scalable, distributed revision control system"*).
- Antes do Git haviam outros sistemas de versionamento como o CVS (Concurrent Versions System, 1986) e SVN (Apache Subversion, 2000).

Instalação (https://git-scm.com)

git --everything-is-local

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.

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.

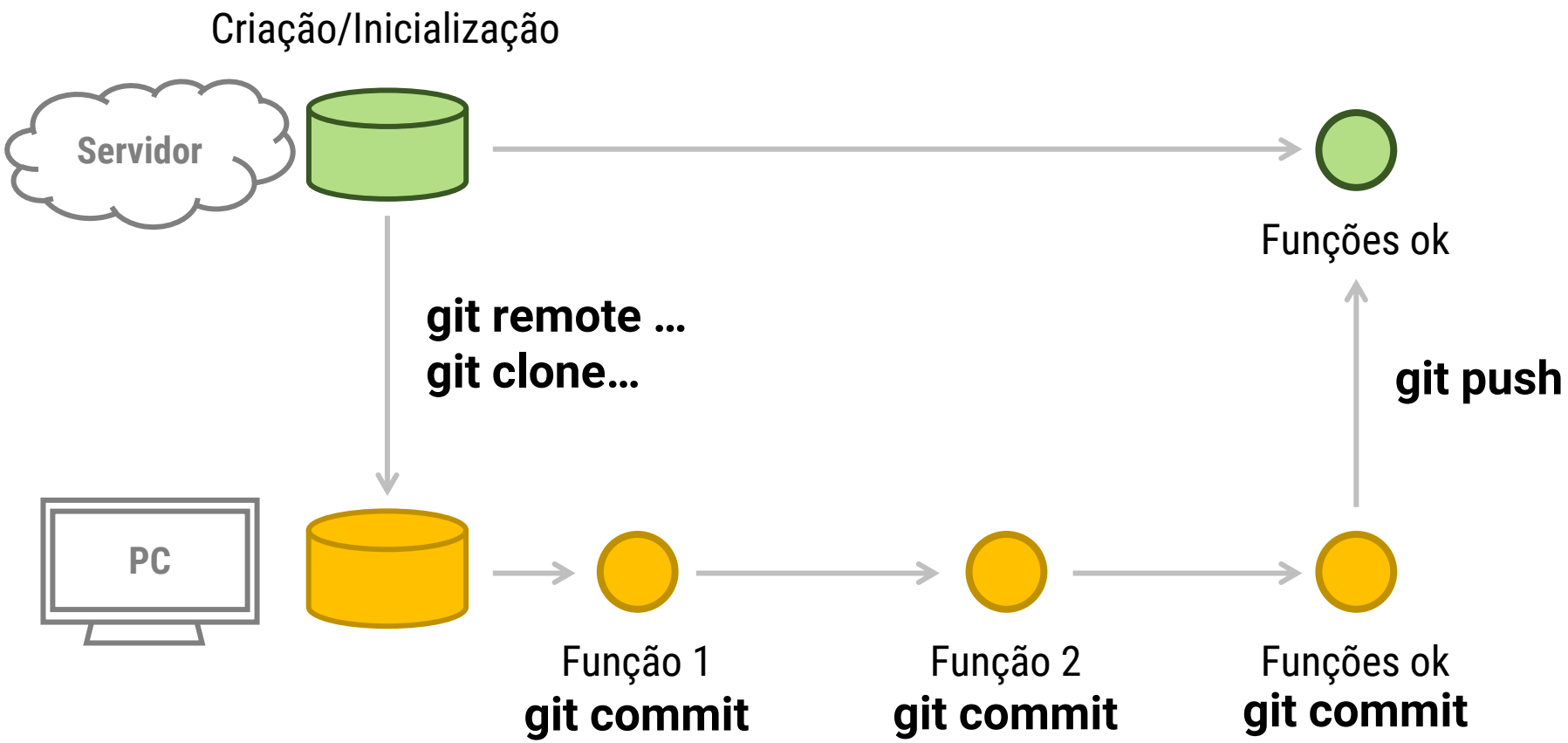
View GUI Clients →

Logos

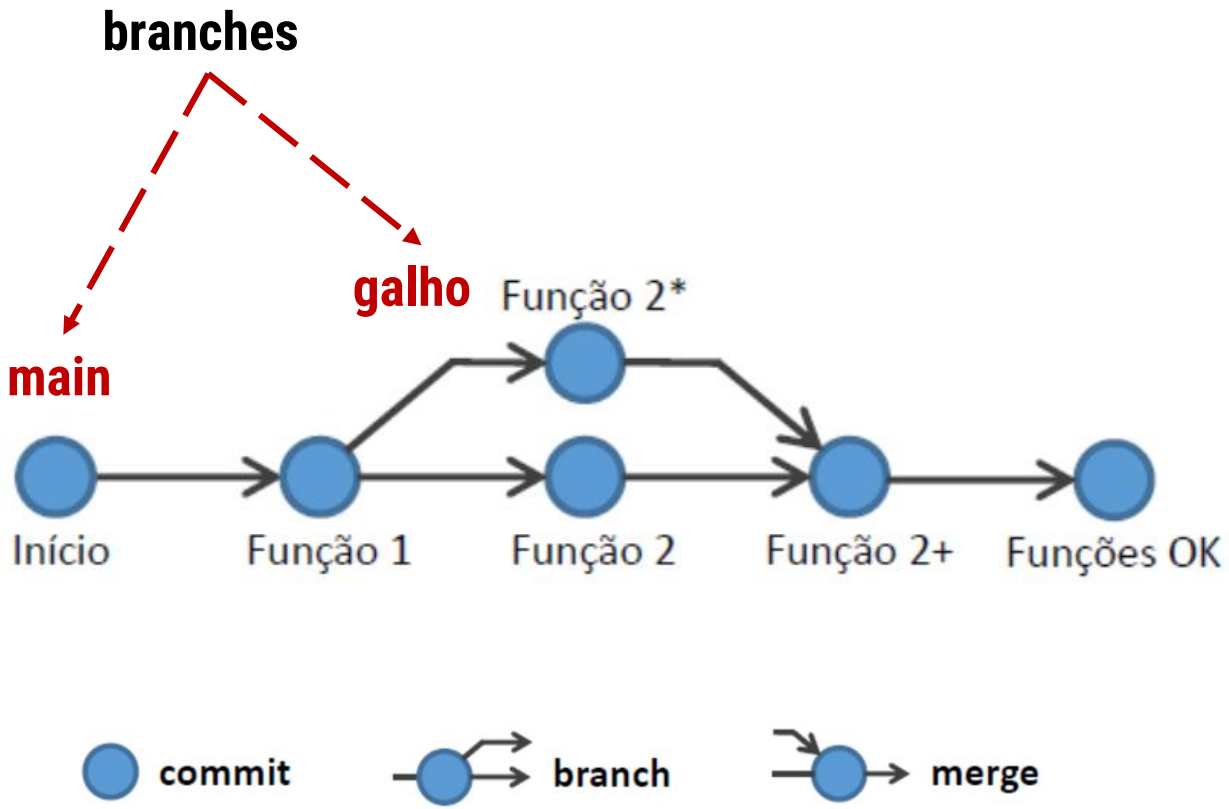
Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

View Logos →

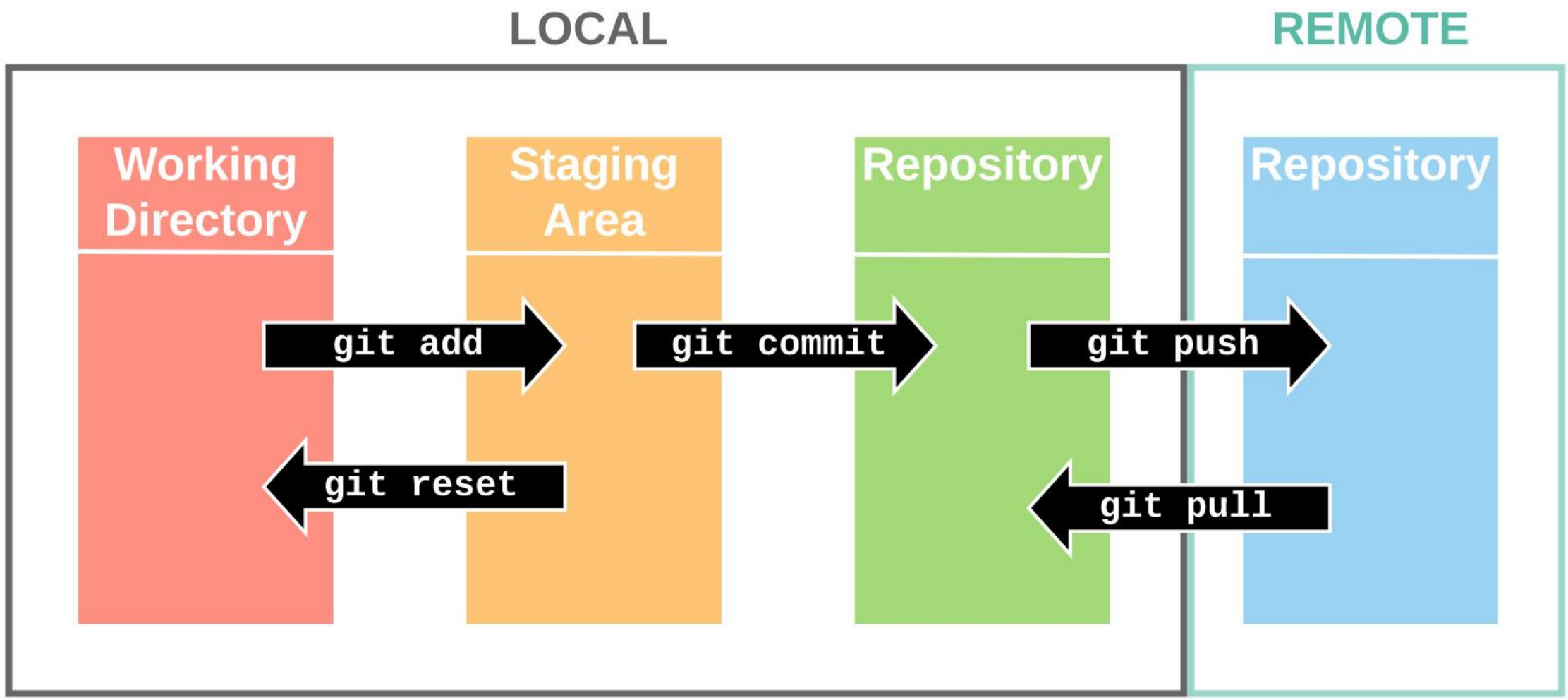
Como funciona o Git? Repositório remoto e local



Como funciona o Git? Branches

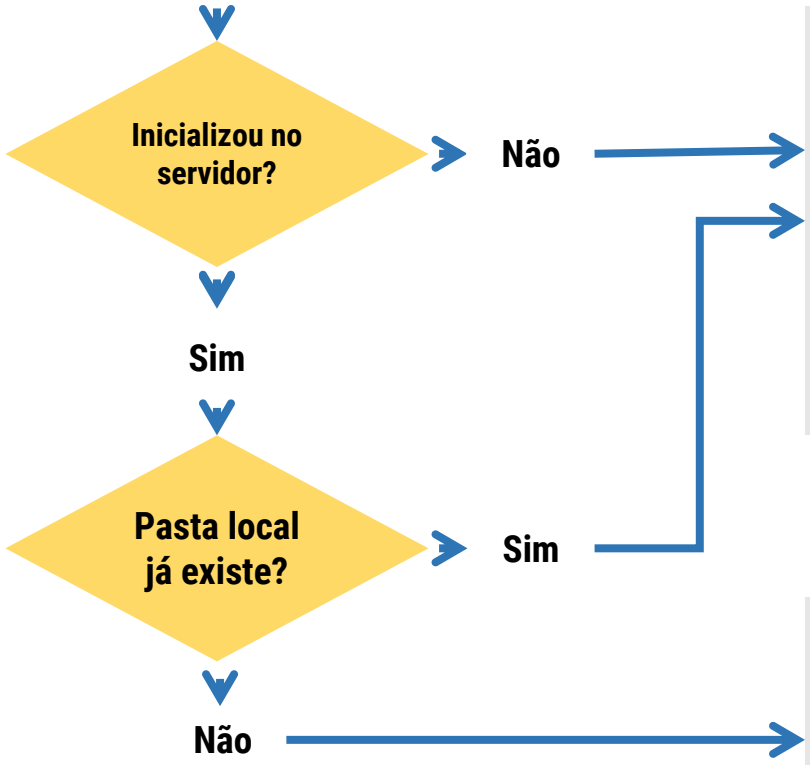


Estágios do versionamento...



Decisões antes de criar um repositório...

Criar repositório no servidor




```
git init
git remote add origin ...
git branch -M main
git rebase... (se inicializou no servidor)
git add .
git commit -m "Primeiro commit"
git push -u origin main
```

- só na incialização do repositório local e primeiro push.

```
git clone ...
git add .
git commit -m "Mensagem commit"
git push
```

Abordagens de criação de um repositório

A) Criar uma pasta para o repositório e incluir a pasta do projeto e outros arquivos dentro da pasta do repositório:

B) Criar o repositório e projeto com o mesmo nome e mesma pasta:

MeuRepositorioProjeto / Arquivos

Abordagem A

1. Criar o repositório remoto (inicializado);
2. Clonar o repositório no computador local:

```
git clone (url do repositório)
```

3. Criar o projeto ou arquivos dentro da pasta do repositório;
4. Versionar as alterações:

```
git add .
git commit -m "Nome do commit"
git push
```

Abordagem B (Sem conflitos)

Criar o projeto local e inicializar o repositório local na pasta do projeto...

```
git init
git branch -M main
git add .
git commit -m "Primeiro commit"
```

Criar o repositório remoto (inicializado) e vincular ao repositório local...

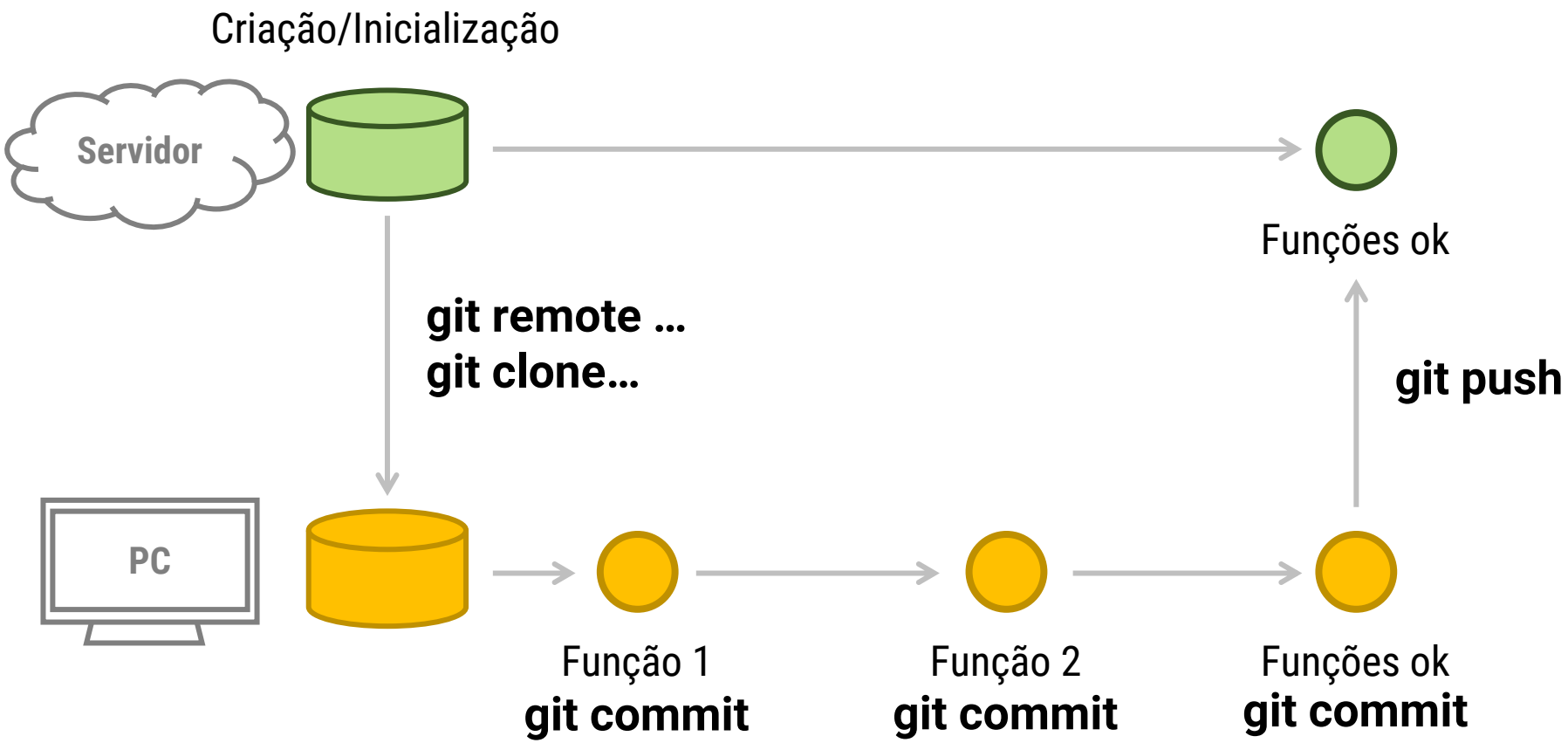
```
git remote add origin https://github.com/xxx/xxxx.git
git fetch origin
git rebase origin/main ou git merge origin/main
git push -u origin main
```

Abordagem B (com conflitos)

Criar o repositório remoto (inicializado) e vincular ao repositório local...

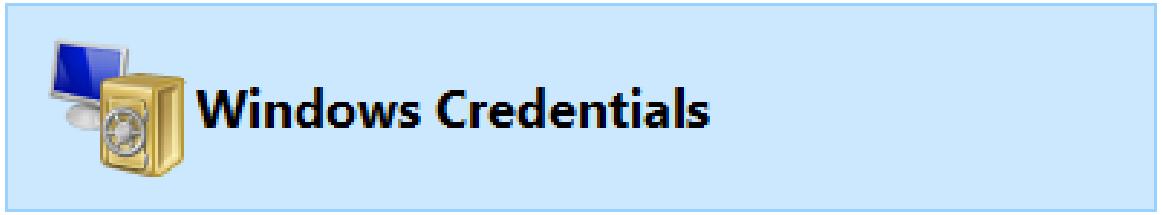
```
//em caso de conflito fazer merge manual e...
git add .
git rebase --continue
git push -u origin main
ou
git add .
git merge origin/main --allow-unrelated-histories
git push -u origin main
```

Versionamento de códigos (individual)



1. Configuração do usuário local e da credencial

```
git config --global user.name "nome de usuario"
git config --global user.email "meu_email@exemplo.com"
git config --global credential.username "usuario"
```



2. Inicialização local e vinculação do repositório remoto

```
git init
git remote add origin https://github.com/xxx/xxxx.git
git branch -M main
git add .
git commit -m "Primeiro commit"
git push -u origin main
```

- só na inicialização do repositório local e primeiro push.

3. Commits quando trabalhando individualmente

```
git add .

git commit -m "Outro commit"

git push
```

4. Clonagem do repositório em outro computador

```
git clone https://github.com/xxx/xxxx.git
git add .
git commit -m "Outro commit"
git push
```

- só na clonagem do repositório.

5. Commits quando trabalhando em equipe

```
git add .

git commit -m "Outro commit"

git pull (realizar o merge manual quando necessário)

git push
```

6. Manipulação de Branchs

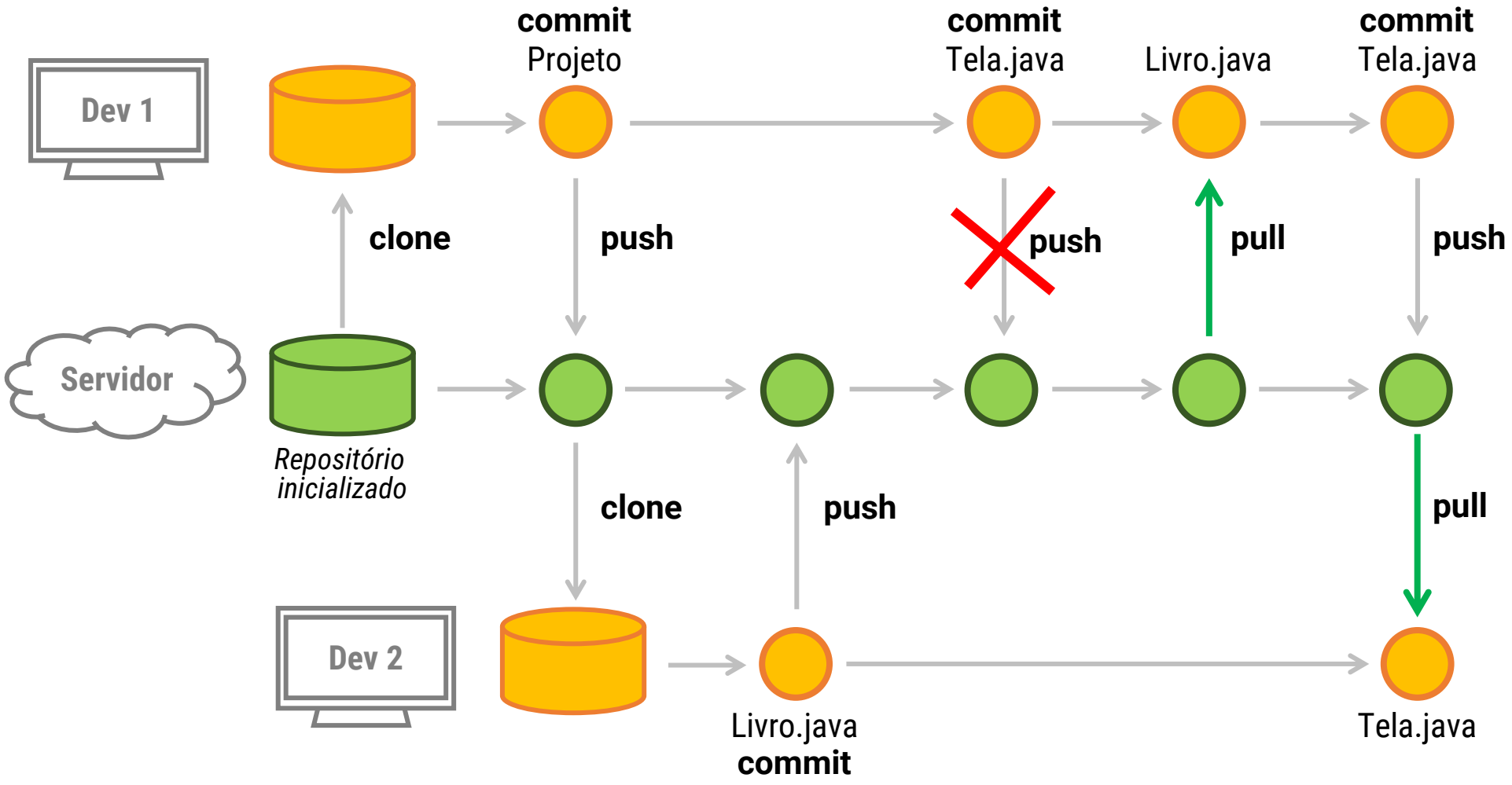
```
git branch --list //Listar

git branch <nova branch> //Criar

git branch -d <branch> //Apagar

git checkout <branch> //Alternar
```

https://www.atlassian.com/git/glossary

Exemplo de versionamento com mesclagem automática

1. **Dev1**: Cria e inicializa o repositório no servidor;
2. **Dev1**: Clona o repositório no computador local, cria o projeto e faz push;
3. **Dev2**: Clona o repositório no computador local, cria a classe Livro.java e faz push;
4. **Dev1**: Cria a classe Tela.java e tenta fazer push sem pull; **(erro)**
5. **Dev1**: Faz a sequência incluindo pull antes do push; **(automerge)**
6. **Dev2**: Faz pull. **(automerge)**

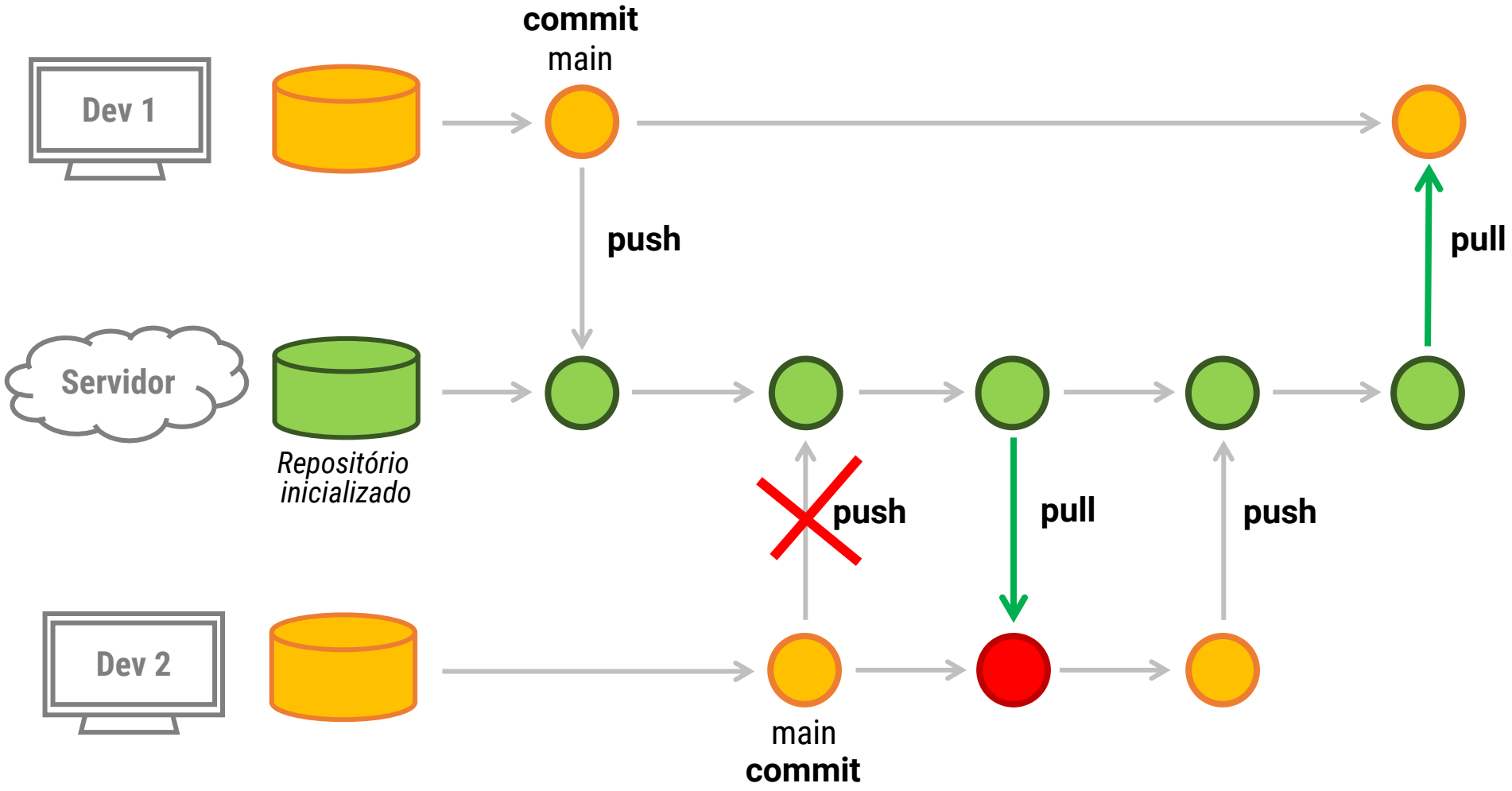
Versionamento de códigos em equipe (automerge)



Exemplo de versionamento com mesclagem manual

1. **Dev1**: Faz alteração no método main e faz push;
2. **Dev2**: Faz alteração no mesmo ponto e tenta fazer push; **(erro)**
3. **Dev2**: Faz a sequência com pull e mas não ocorre automerge; **(conflito)**
4. **Dev2**: Faz a correção dos conflitos no código; **(merge manual)**
5. **Dev2**: Faz push;
6. **Dev1**: Faz pull.

Versionamento de códigos em equipe (merge manual)



Miscelânea...

git status
verifica a situação do versionamento

git reset HEAD nome_do_arquivo
volta ao estágio anterior do adicionamento do arquivo

git merge nome_do_branch
mescla as alterações de uma branch com a branch que estiver ativa

git revert
defaz o último commit