

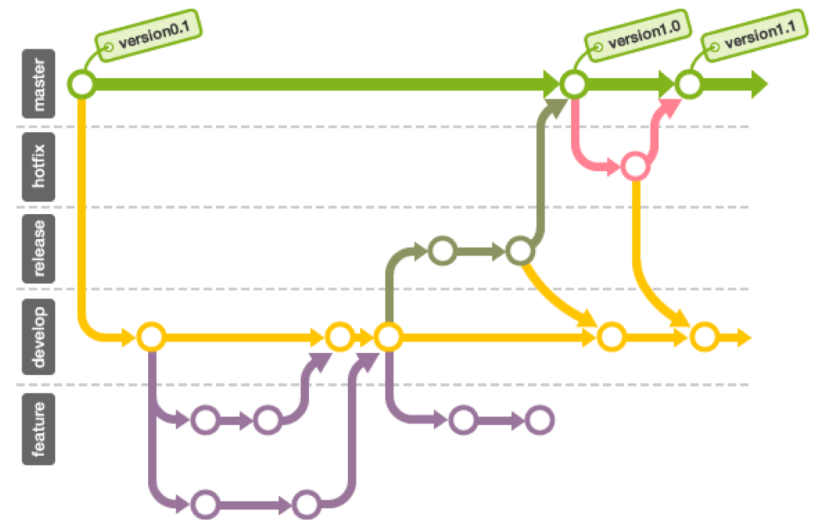
# Controle de Versão



Prof. Matheus Sousa Faria

# Sistema de Controle de Versão (VCS)

- Mantém registrado
  - O que foi alterado
  - Quando foi alterado
  - Quem alterou
  - Porque alterou
- Histórico de mudanças
- Permite voltar e avançar nas modificações
  - Ver diferentes estados
  - Desfazer mudanças
- Previnem perda de informação
- Comparação de estados
  - Ajuda em resolução de bugs

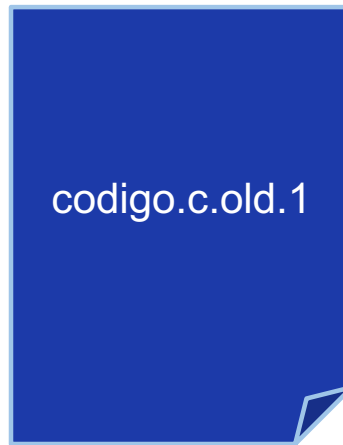


Só isso? Então vou  
fazer o meu próprio

# MyVCS



Atual



Cópias para  
histórico

MyVCS

Funciona!

codigo.c

codigo.c.old.1

codigo.c.old.2

codigo.c.old.3

Atual

Cópias para  
histórico

# MyVCS

Funciona!

codigo.c

codigo.c.old.1

codigo.c.old.2

codigo.c.old.3

Atual

4x Mais espaço

Quem alterou?

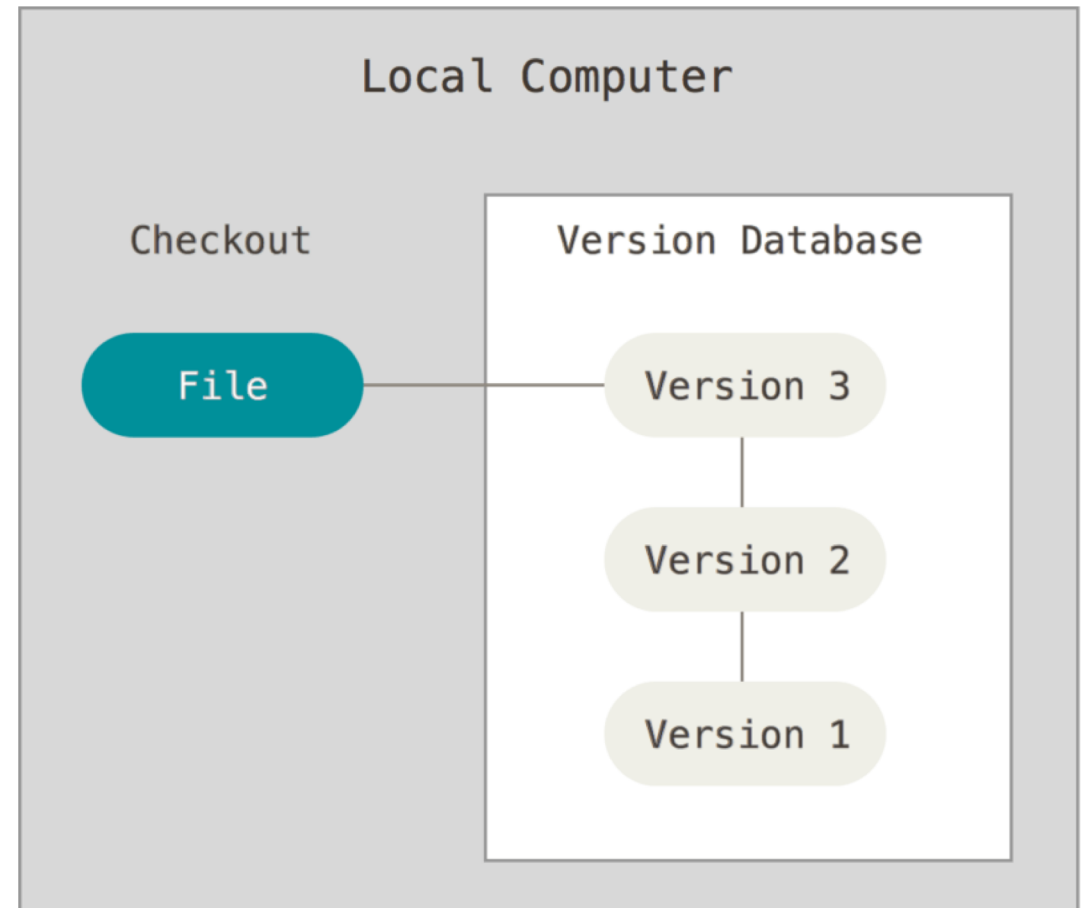
Onde esta a  
feature X?

Trabalhar em  
grupo?

Cópias para  
histórico

# Revision Version Control (RCS)

- Bacon de dados local
- Guarda informações das alterações
- Diferenças são guardadas em patches
- Trabalho em grupo?



# Políticas de Compartilhamento

## **Cópia de trabalho**

Trava-Modifica-Destrava

Copia-Modifica-Resolve

---



# Cópia de trabalho

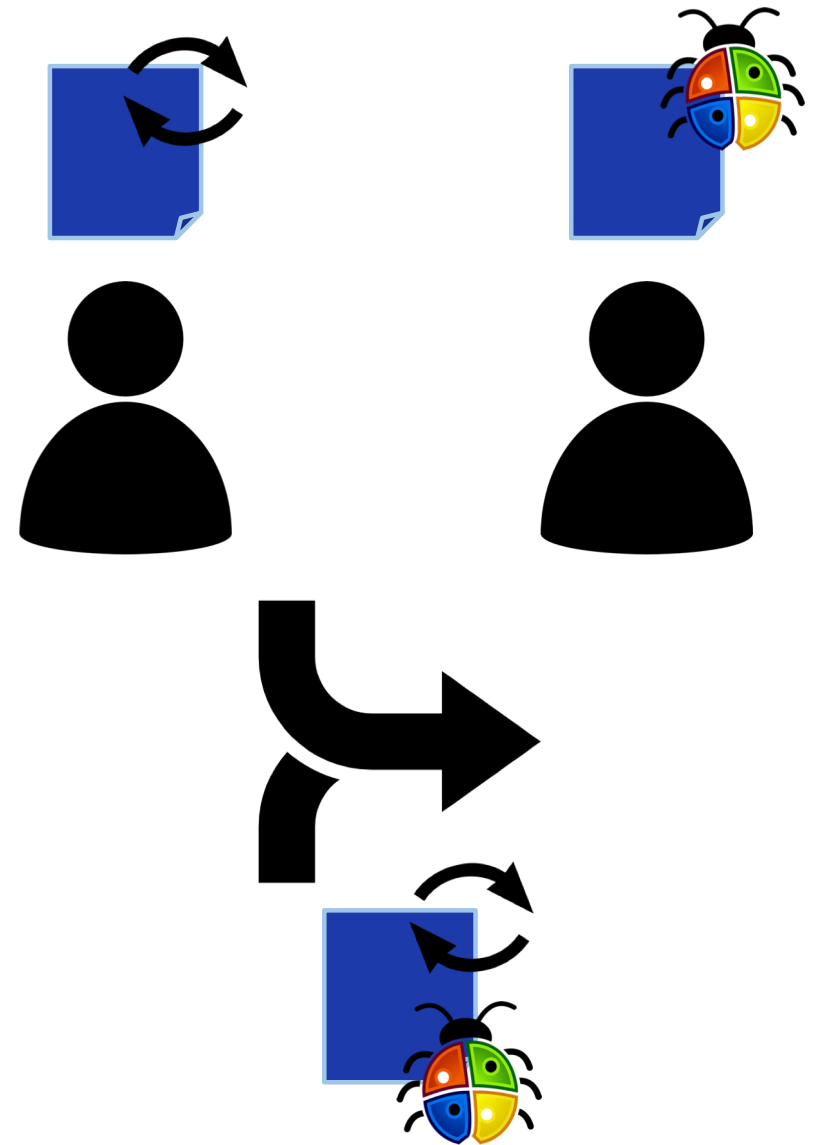
- Cada um tem uma cópia
- Modelo descentralizado
- Sem cópia central remota
- RCS ou “MyVCS”

## Vantagens:

- Não requer ferramentas

## Desvantagens:

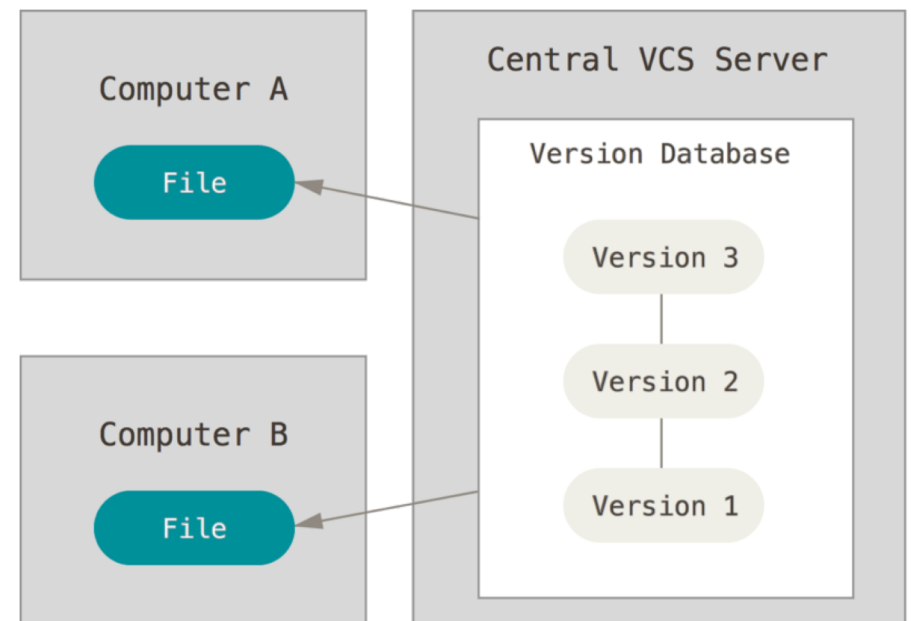
- Díficil de sincronizar
- Muito trabalho manual



“Precisamos trabalhar  
em grupo de uma  
maneira melhor”

# Sistemas de Controle de Versão Centralizados

- Cópia remota em algum servidor
- Todos podem saber quem está trabalhando no que
- Mais poderes para o administrador
  - Controle de acesso
- E se o server cair?



# Políticas de Compartilhamento

Cópia de trabalho

**Trava-Modifica-Destrava**

Copia-Modifica-Resolve

---

# Trava-Modifica-Destrava

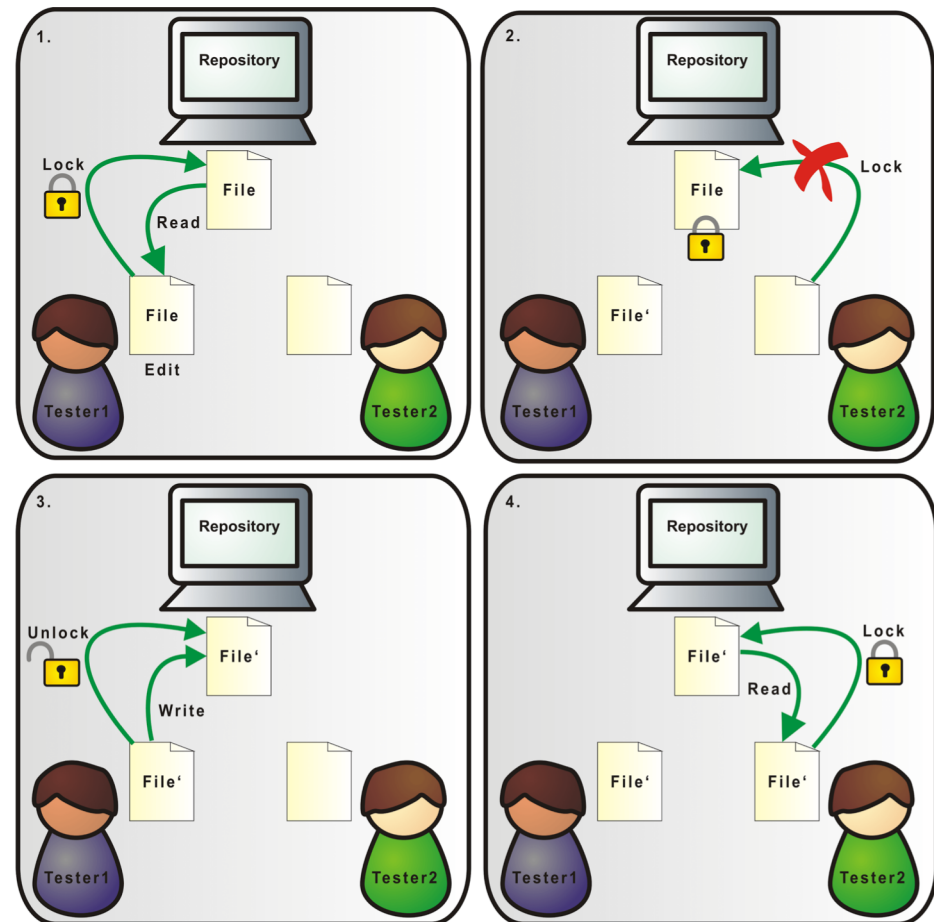
- Bloqueia o acesso a um arquivo
- Não permite o trabalho paralelo
- Fácil de mesclar os trabalhos
- CVS, SVN

## Vantagens

- Fácil de compartilhar o trabalho
- Não gera conflitos

## Desvantagens

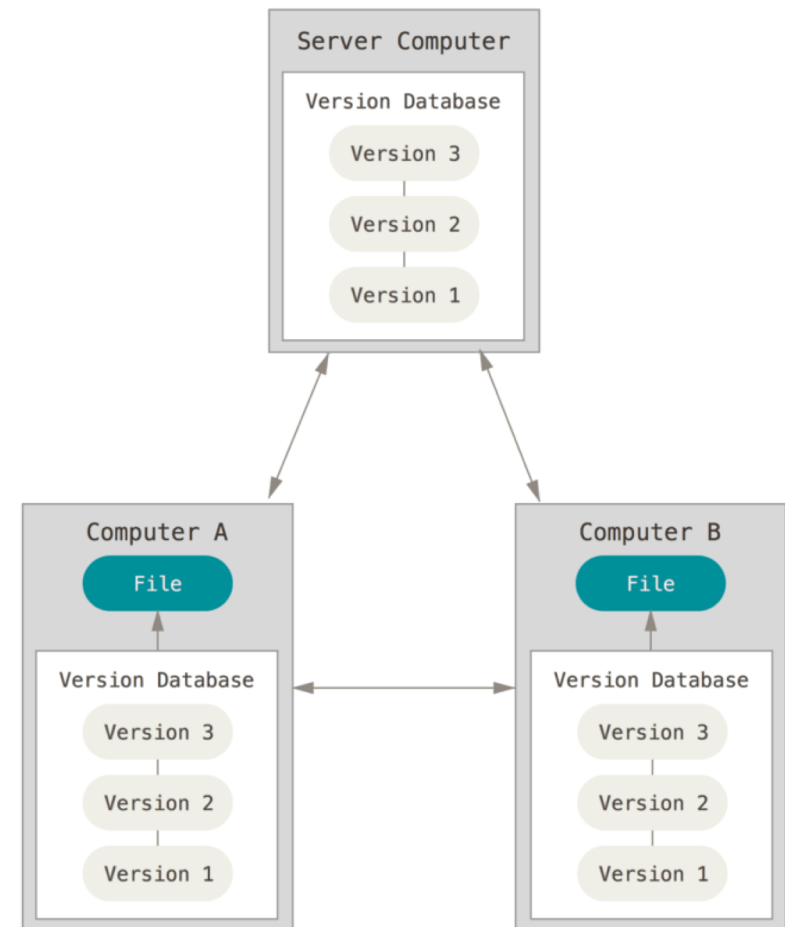
- Não há trabalho paralelo no mesmo arquivo



Precisamos evitar perder  
todo o trabalho por  
causa de falha no  
servidor

# Sistemas de Controle de Versão Descentralizados

- O servidor tem uma cópia, e cada desenvolvedor tem a sua
  - Cópias idênticas
- Contribuições e trabalho simultâneo
- Se o servidor cair? Não tem problema



# Políticas de Compartilhamento

Cópia de trabalho

Trava-Modifica-Destrava

**Copia-Modifica-Resolve**

---



# Copia-Modifica-Resolve

- Você obtém uma cópia, modifica, e envia ela para o servidor central
  - Se houver conflitos, você resolve
- Git, Mercurial, Bazaar

## Vantagens

- Permite trabalho em paralelo no mesmo arquivo
- Mais tolerante a falhas

## Desvantagens

- Maior complexidade em mesclar as modificações





**git**

# Git



# git

- Sistema de Controle de Versão Distribuido
- O sistema mais utilizado em projetos open-source
- Criado em 2005 por Linus Torvalds
- Substituto ao BitKeeper no projeto kernel do linux
  - Bitkeeper era proprietário
  - Antes do Bitkeeper, utilizava-se tar.gz
- Git foi desenhado para:
  - Aplicar um patch/diff em 3 segundos ou menos
  - Seguir o workflow parecido ao do BitKeeper
  - Suporte a desenvolvimento não linear (milhares de branches)
  - Distribuído
  - Ser o oposto ao CVS (Concurrent Versions System)
- Nome escolhido “aleatoriamente”

Alguns outros  
conceitos

# Repositório

Guarda arquivos

Guarda Configurações

Guarda todas as versões

---

# Plataforma de Colaboração (Forge)

Repositório remoto em um servidor

Compartilhada entre vários desenvolvedores

Facilita a comunicação

Rastreo de bugs e funcionalidades

---



# GitLab

## Plataforma de Colaboração (Forge)



Repositório remoto em um servidor

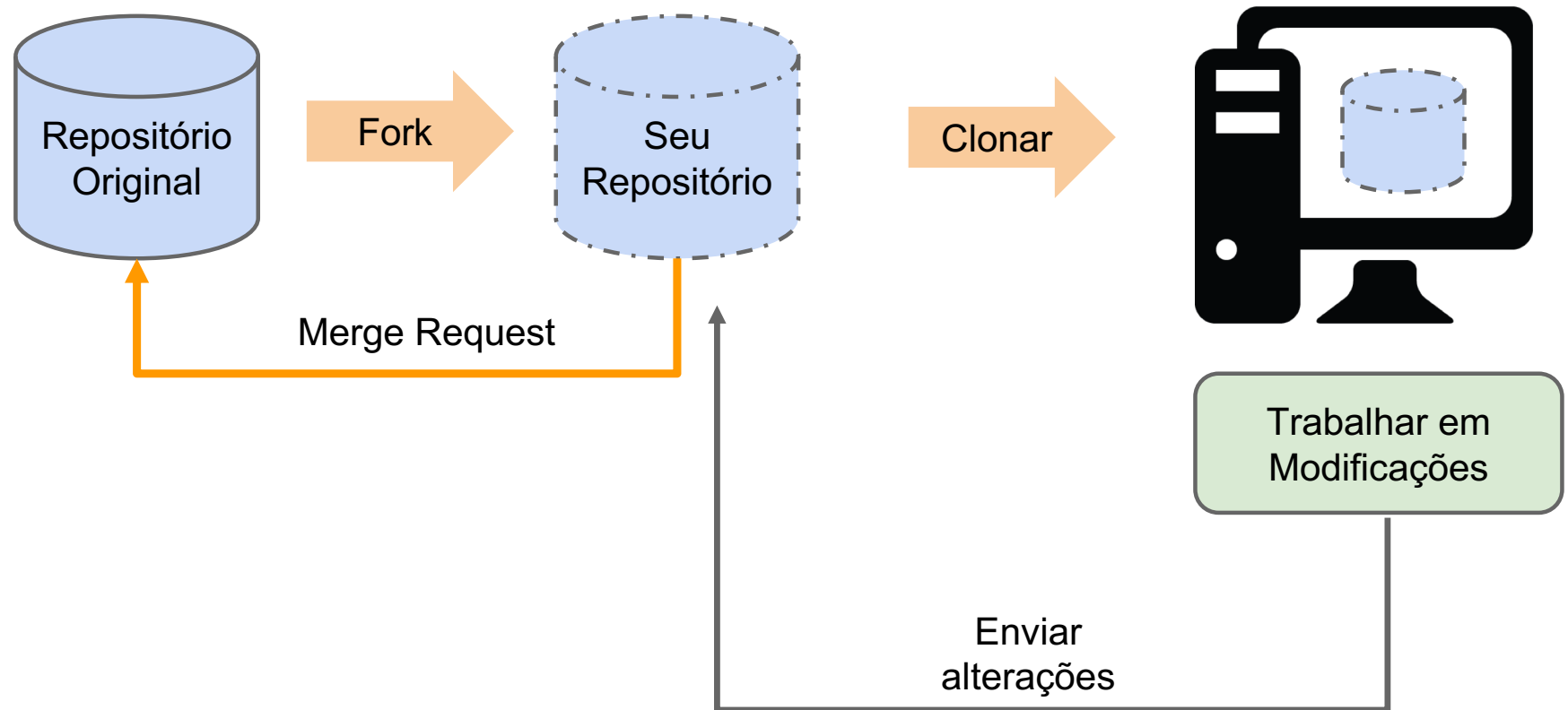
Compartilhada entre vários  
desenvolvedores

Facilita a comunicação

Rastreamento de bugs e funcionalidades

---

# Fluxo de trabalho em um Forge





# Organização do Repositório

## Política de Branches

- Estabelecido pelo time

Um modelo conhecido:

- Master é uma branch instável, a mais recente
  - Pode estar quebrada ou não
- Versões estáveis são **tags**
  - Exemplos: v1.0
- Branches v1.0.x : são fix para bugs que surgem na versão estável
- Outras branches: trabalho atual

# Perguntas?

