Gerenciamento de Código com Git[Hub] 1

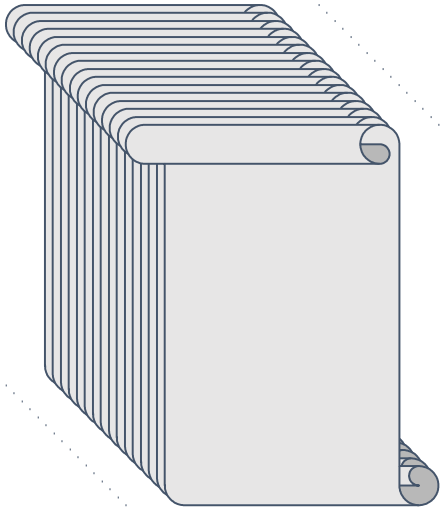
Prof. Davi da Silva Böger
dsboger@gmail.com

Tópicos

- Gerenciamento de Código
- Codificação Colaborativa
- Git
- Atividade

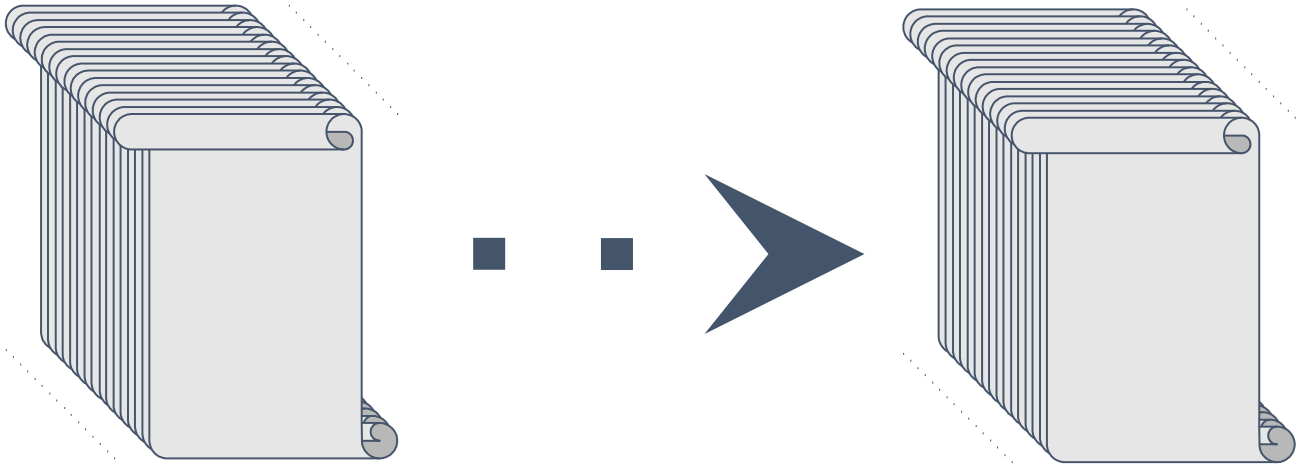
Gerenciamento de Código

- Códigos de programas podem ser bastante grandes e complexos
  - Centenas de classes
  - Milhões de linhas de código



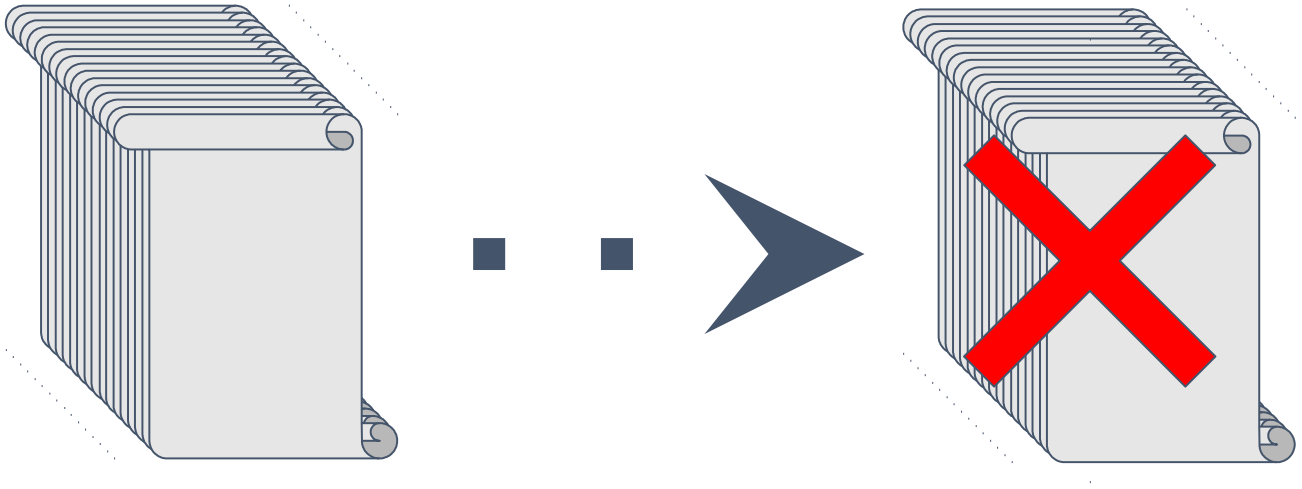
Gerenciamento de Código

- Modificações em partes do código podem ter implicações complexas e difíceis de rastrear
- Quando começamos uma modificação, não sabemos ao certo onde vai chegar!
- Nem se vai dar certo!



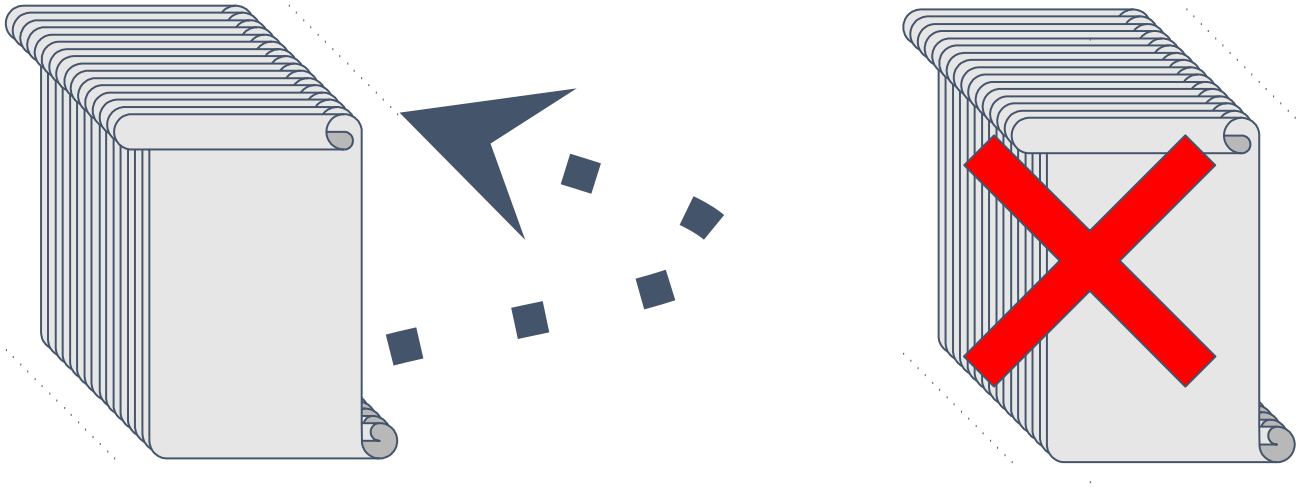
Gerenciamento de Código

- O que fazer se começamos uma modificação e depois decidimos que não deu certo?



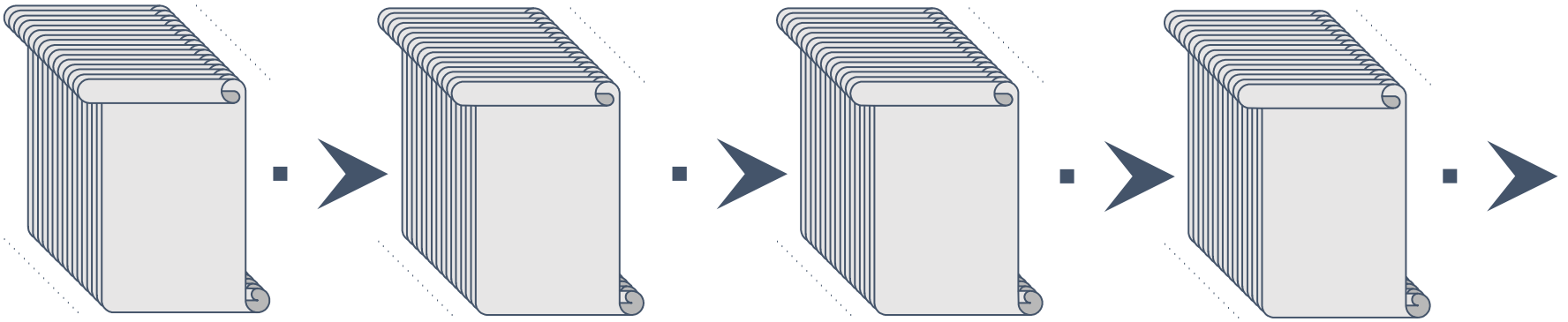
Gerenciamento de Código

- O que fazer se começamos uma modificação e depois decidimos que não deu certo?
- Precisamos desfazer o que foi feito.
- Para isso, guardamos uma cópia antes de cada modificação



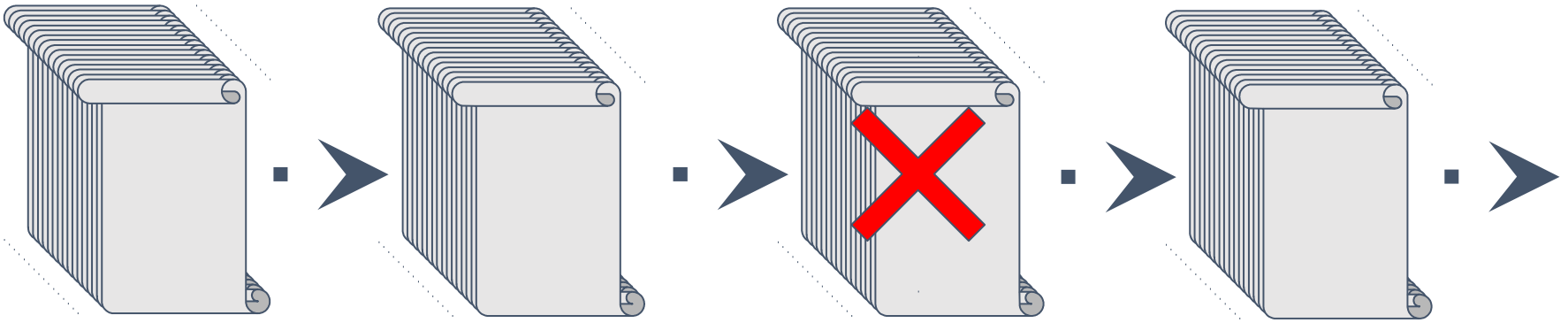
Gerenciamento de Código

- Cada cópia do código é chamada de **versão**
- Guardamos todas as versões passadas do código como uma forma de registro histórico



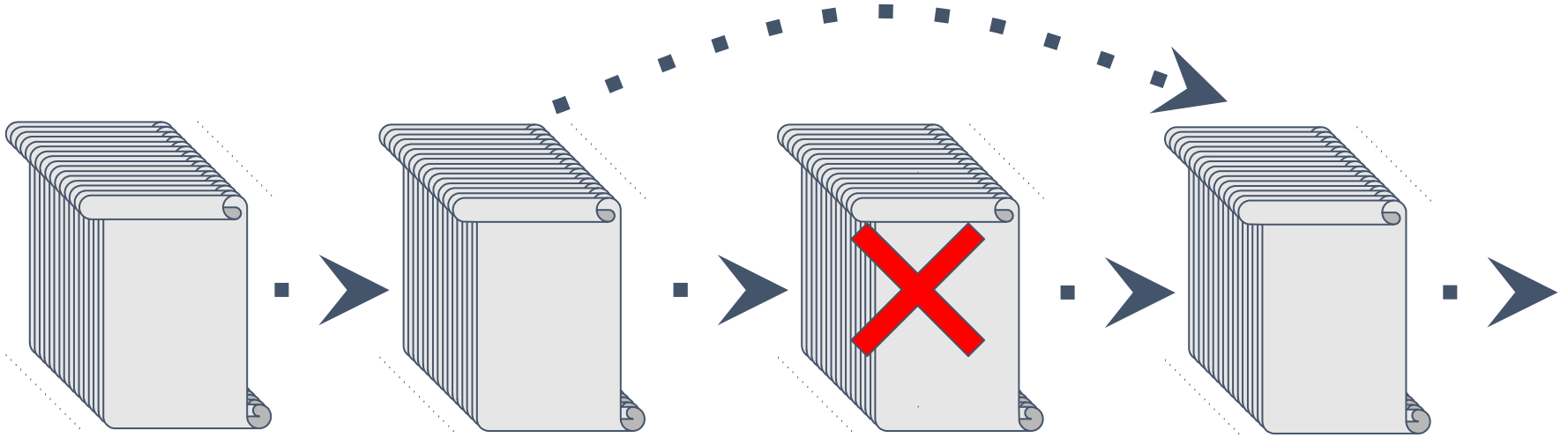
Gerenciamento de Código

- E se somente descobrirmos que uma modificação precisa ser desfeita no futuro?



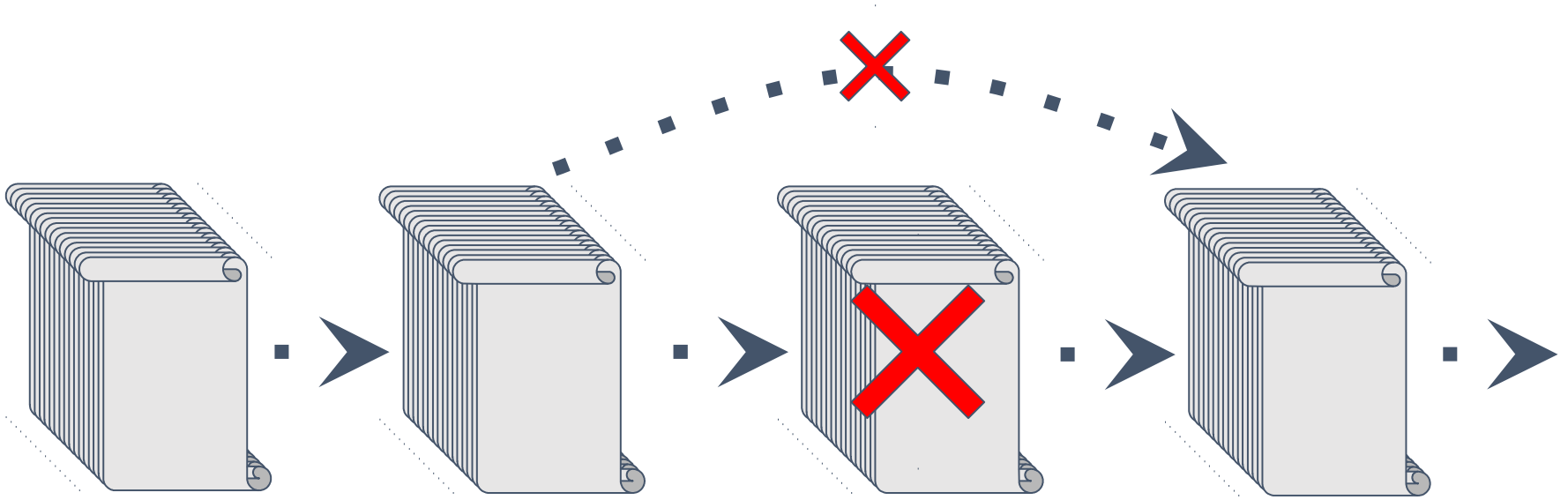
Gerenciamento de Código

- Podemos mudar a história do código?



Gerenciamento de Código

- Podemos mudar a história do código? **NÃO!**
-



Gerenciamento de Código

- Agora imagine um software desenvolvido por vários anos!
- Milhares de versões! Fazer cópias “na mão” não é uma opção viável!
- Mas existem ferramentas!

Gerenciamento de Código

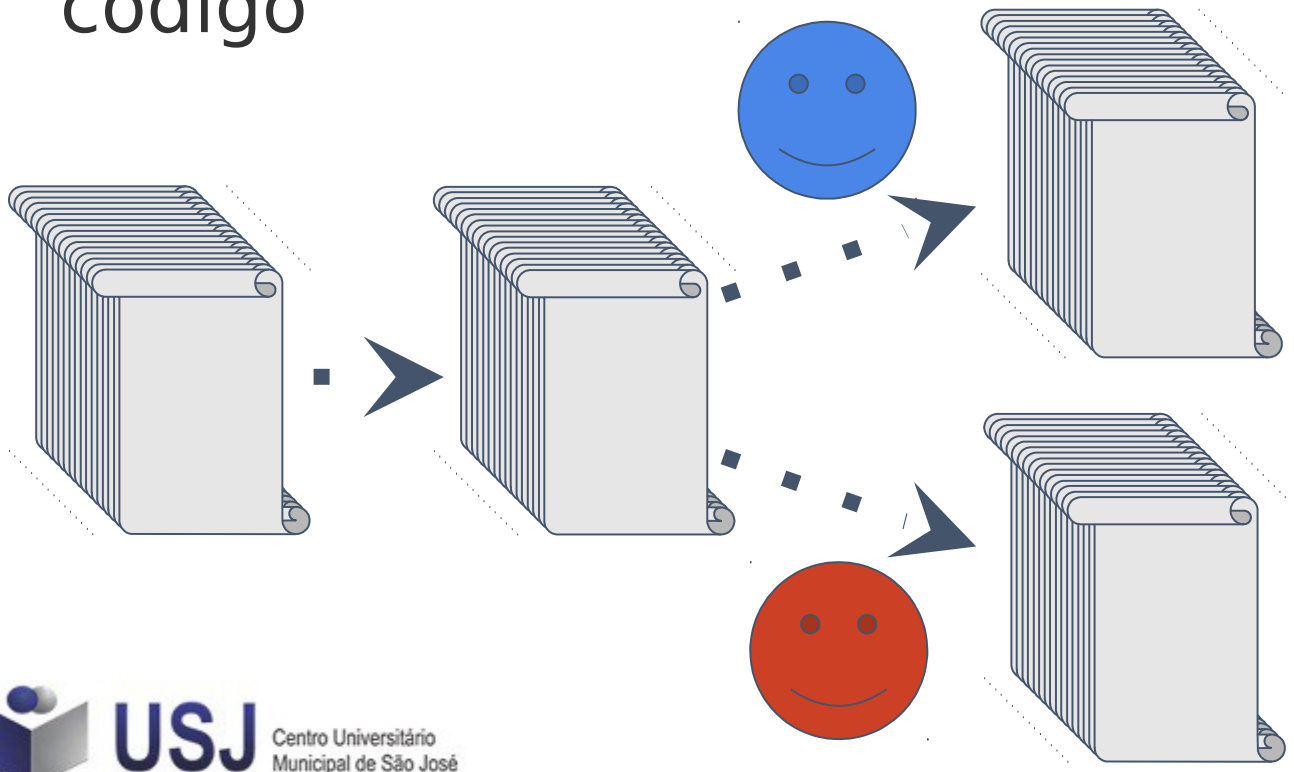
- Além das modificações do código, o gerenciamento de versões também registra:
  - Quando uma alteração foi feita
  - Quem fez a alteração
  - Uma mensagem explicando o que e, principalmente, o porquê da modificação

Codificação Colaborativa

- As ferramentas de gerenciamento de código modernas facilitam que vários programadores modifiquem um mesmo código

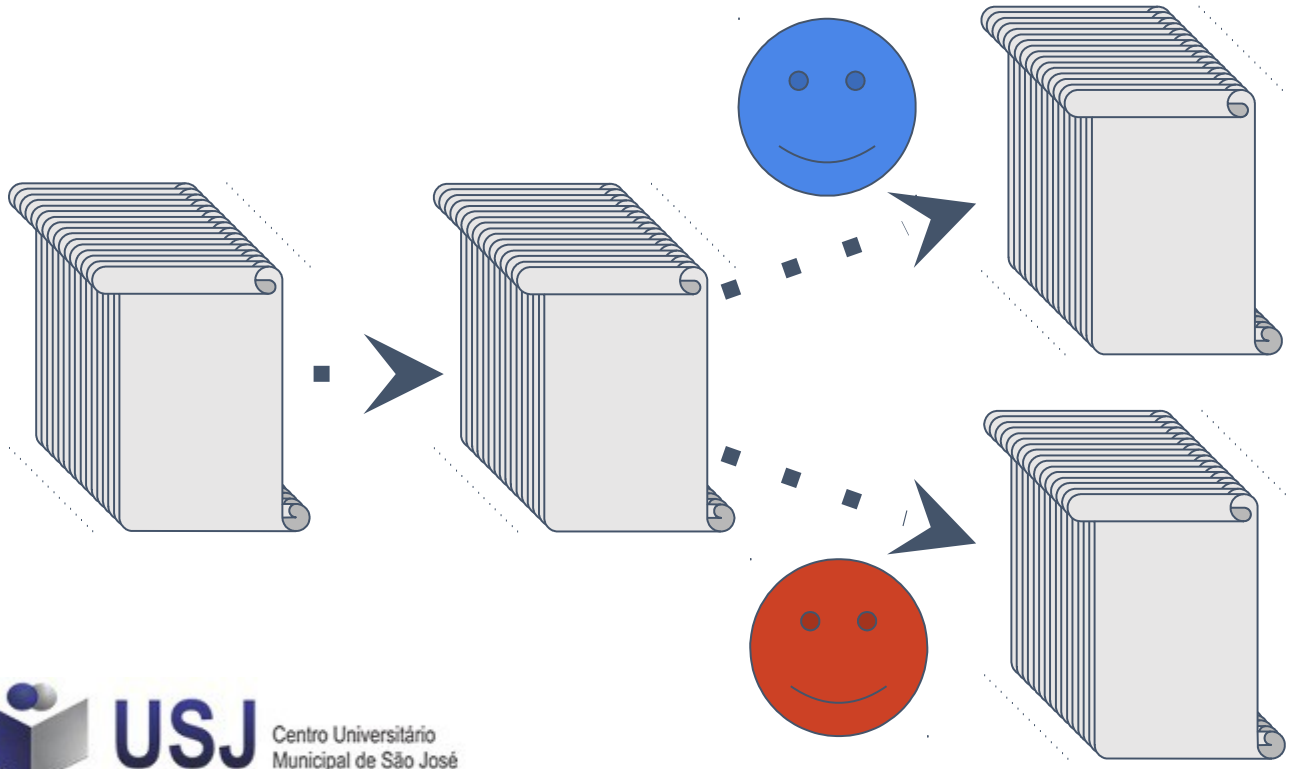
Codificação Colaborativa

- As ferramentas de gerenciamento de código modernas facilitam que vários programadores modifiquem um mesmo código



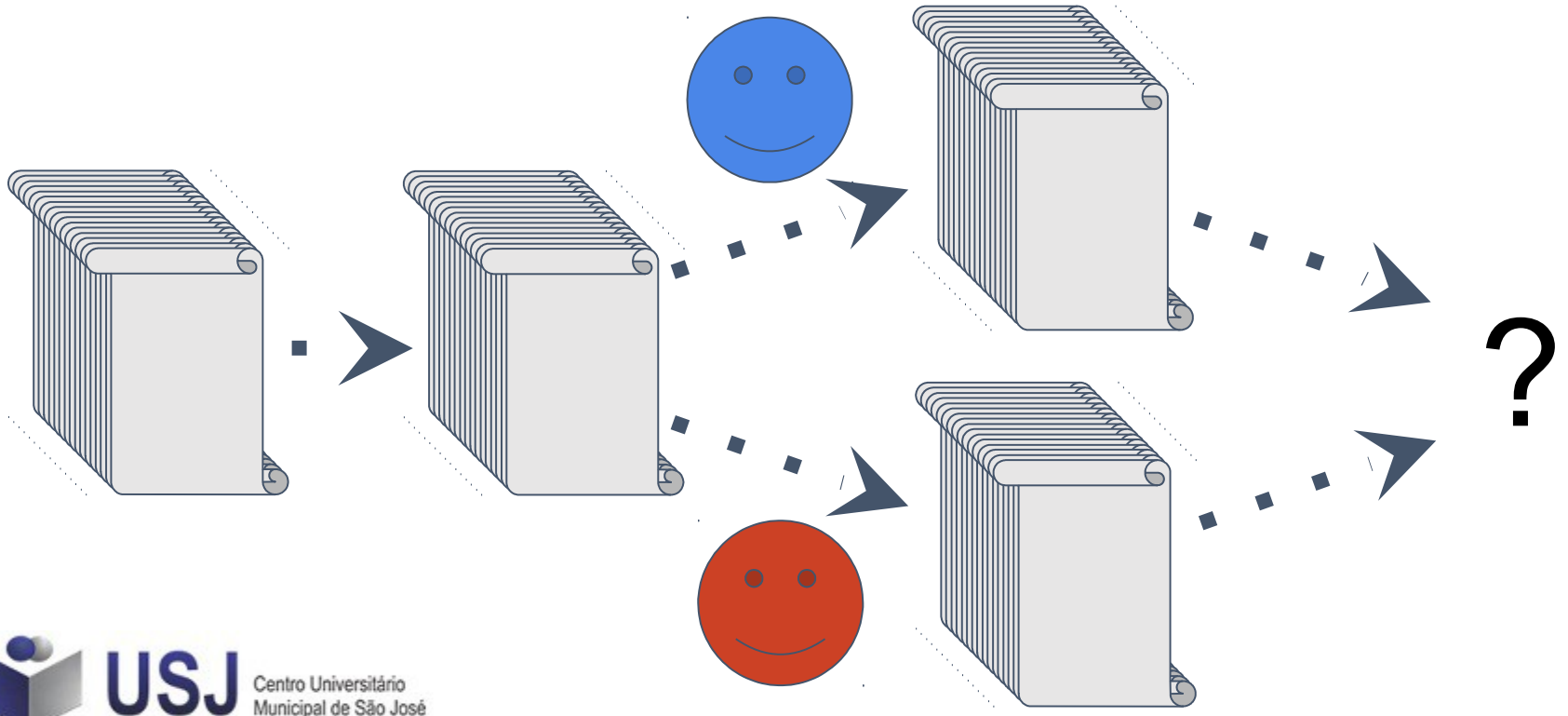
Codificação Colaborativa

- Quando isso ocorre, a história se ramifica, cada programador cria seu ramo (branch, em inglês)



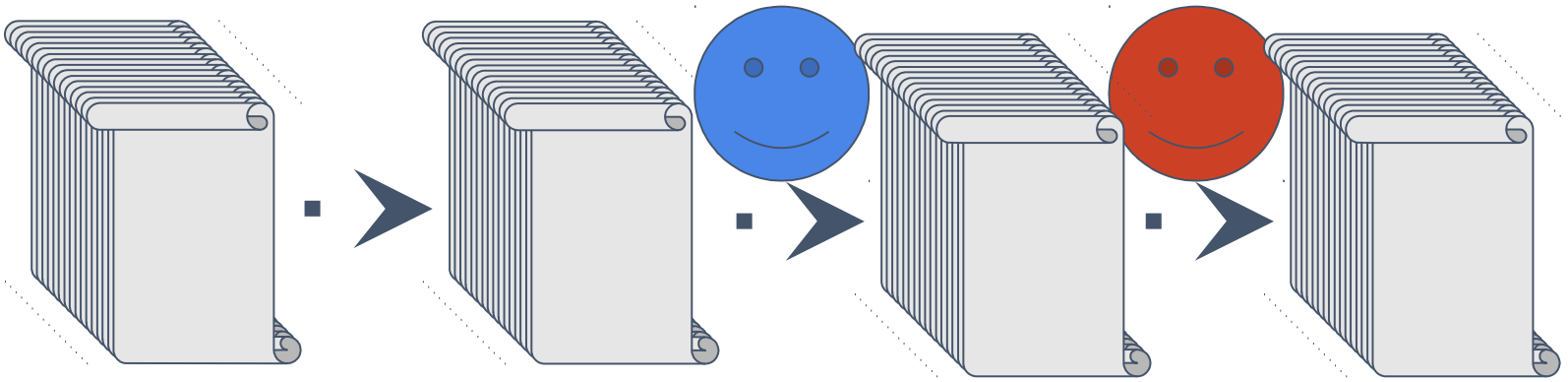
Codificação Colaborativa

- Mas o código é um só, ou seja, os ramos precisam ser unidos. Mas como?



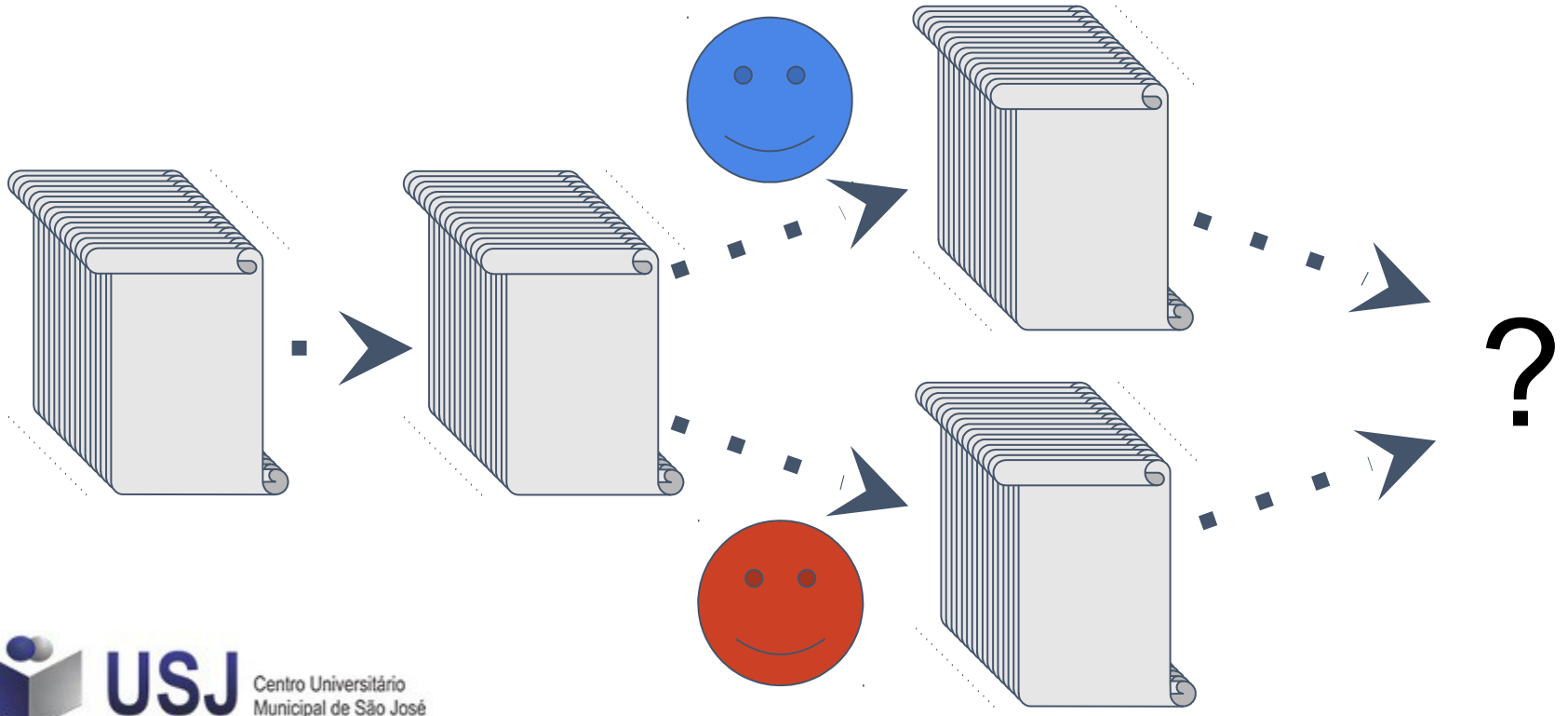
Codificação Colaborativa

- Se as modificações forem independentes (i.e. em arquivos diferentes), pode ser assim:



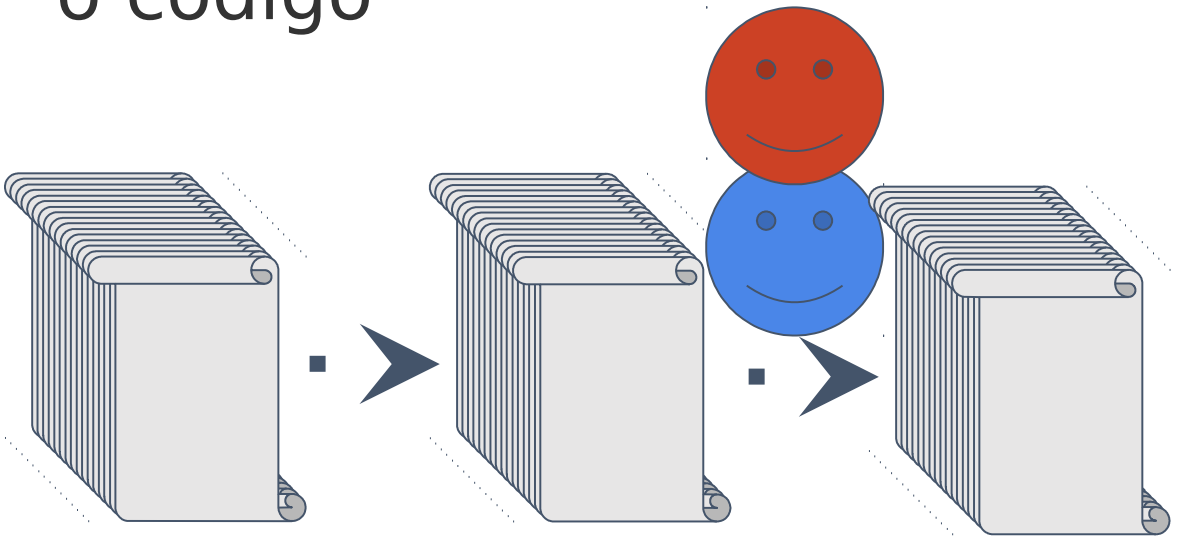
Codificação Colaborativa

- Mas se as modificações tiverem arquivos em comum? Qual vale?



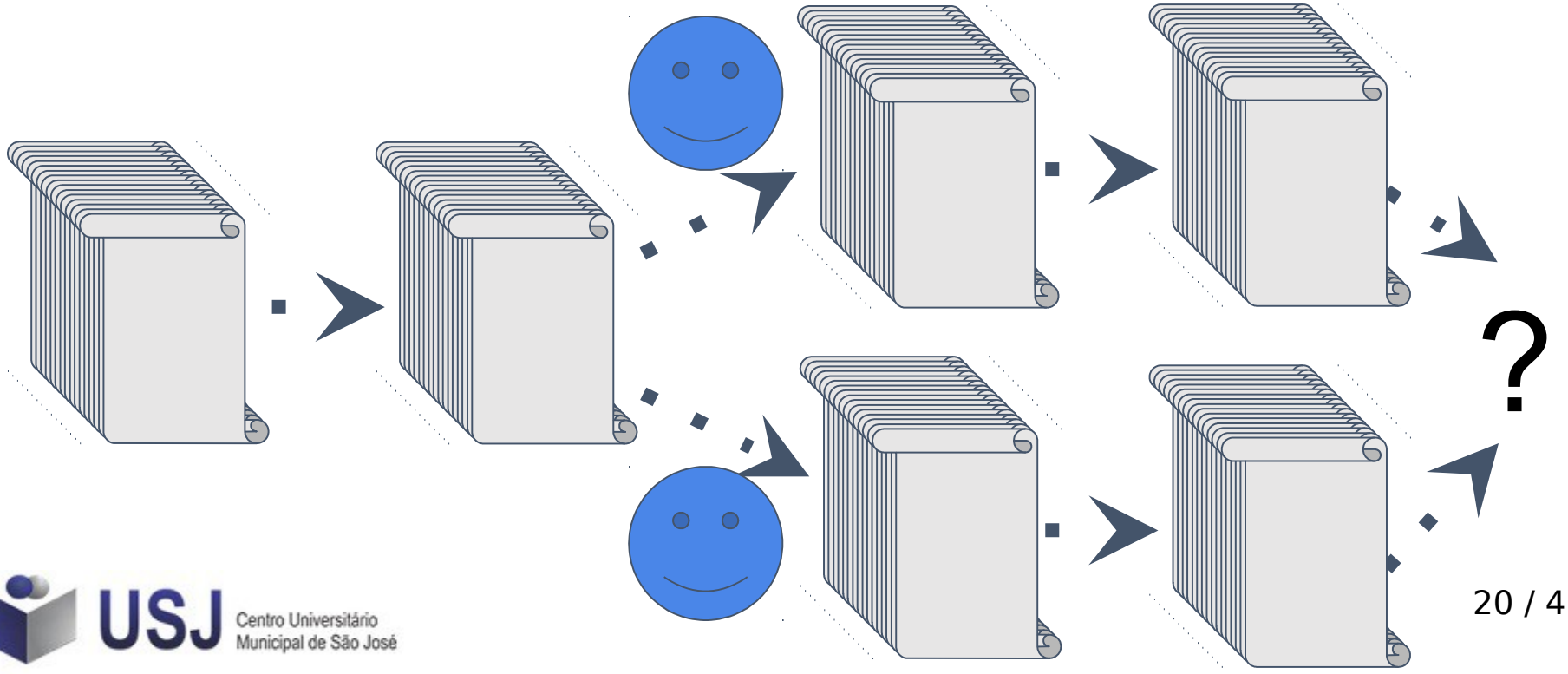
Codificação Colaborativa

- Nesse caso é preciso fazer uma mistura das duas. Não dá pra fazer de maneira automática, precisa um programador arrumar o código



Codificação Colaborativa

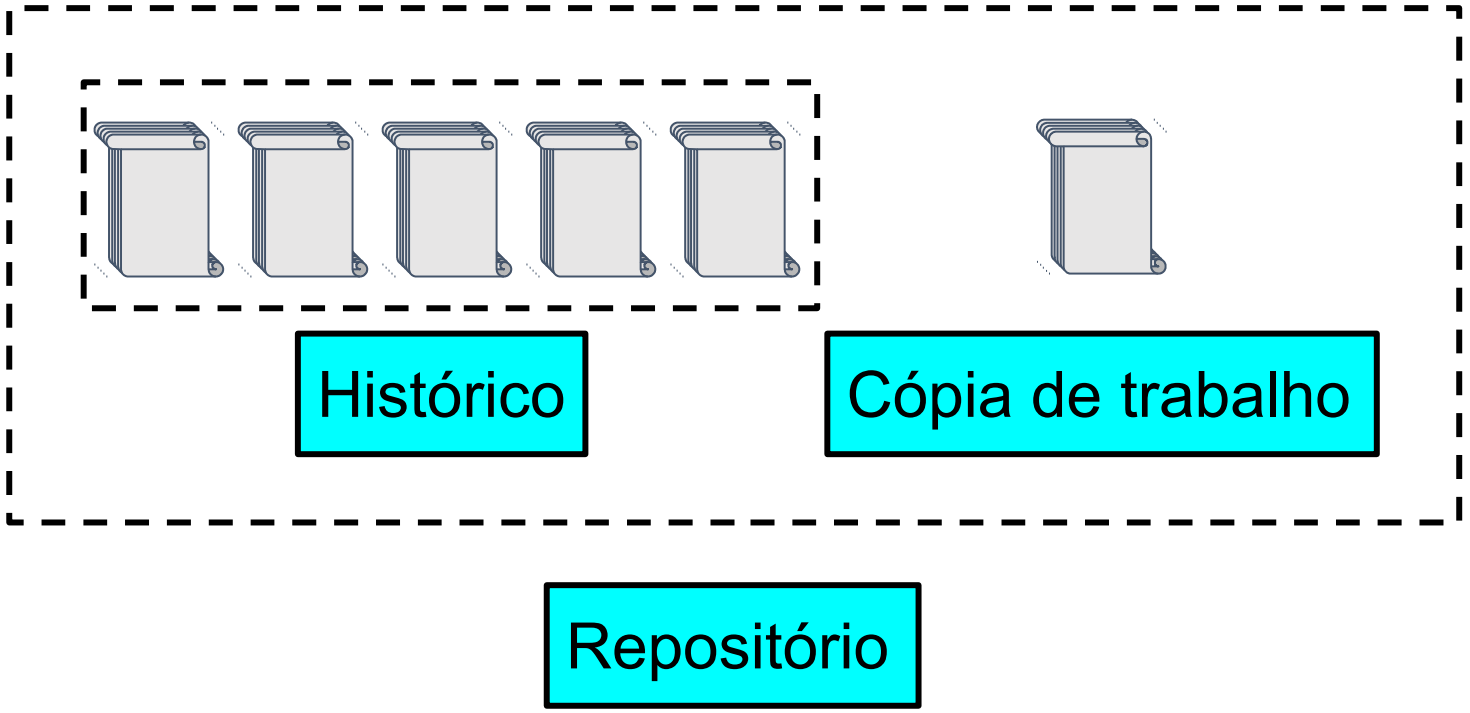
- Também podemos usar a ramificação para trabalhar em duas modificações grandes ao mesmo tempo



Git

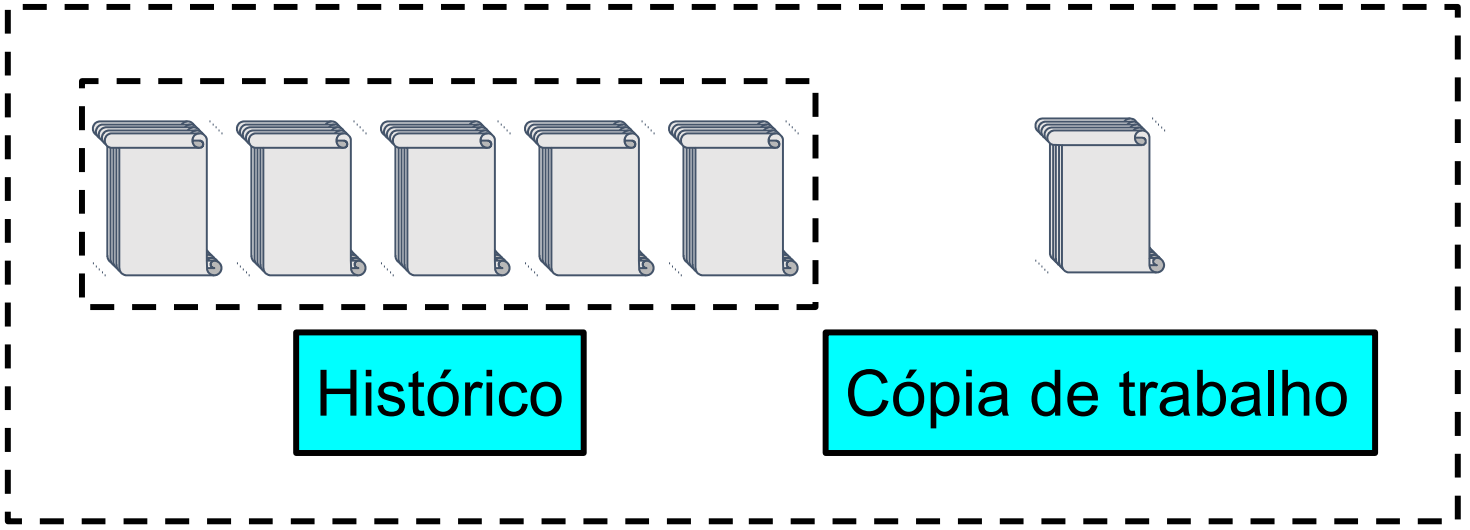
- Git é a ferramenta de gerenciamento de código mais usada atualmente
- Git organiza o código e armazena as versões e o histórico de modificações como um **repositório**
- A versão mais atual do código, incluindo as modificações em andamento, é chamada de **cópia de trabalho**

Git



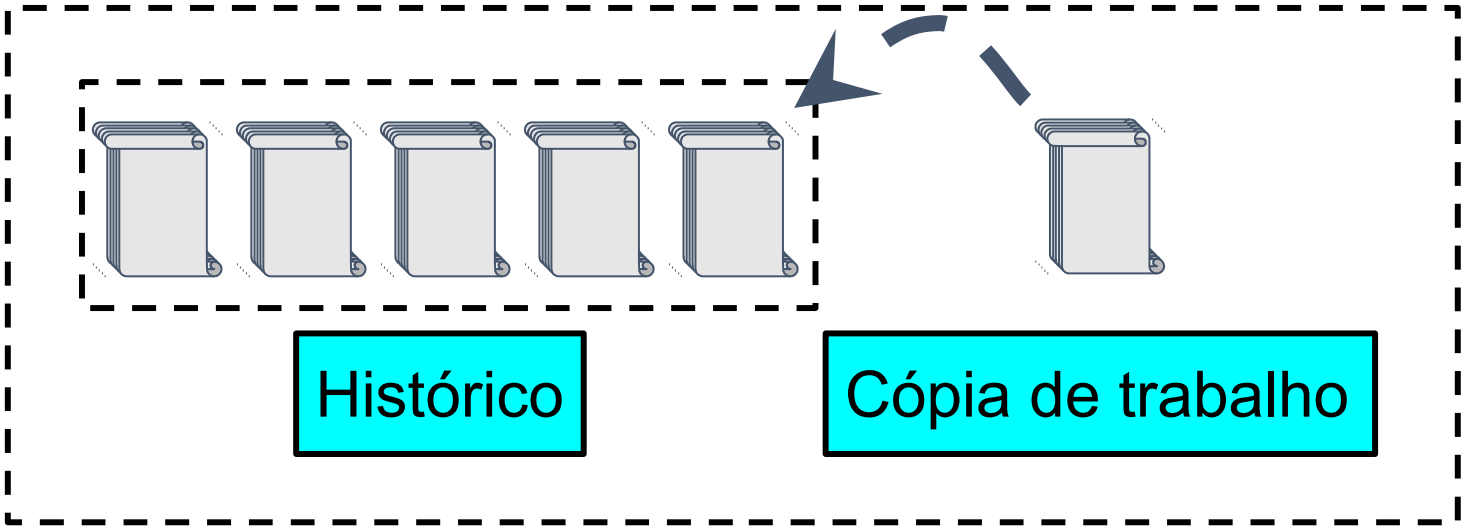
Git

- Quando estamos contentes com as modificações que queremos guardar uma nova versão, executamos uma ação `git commit`



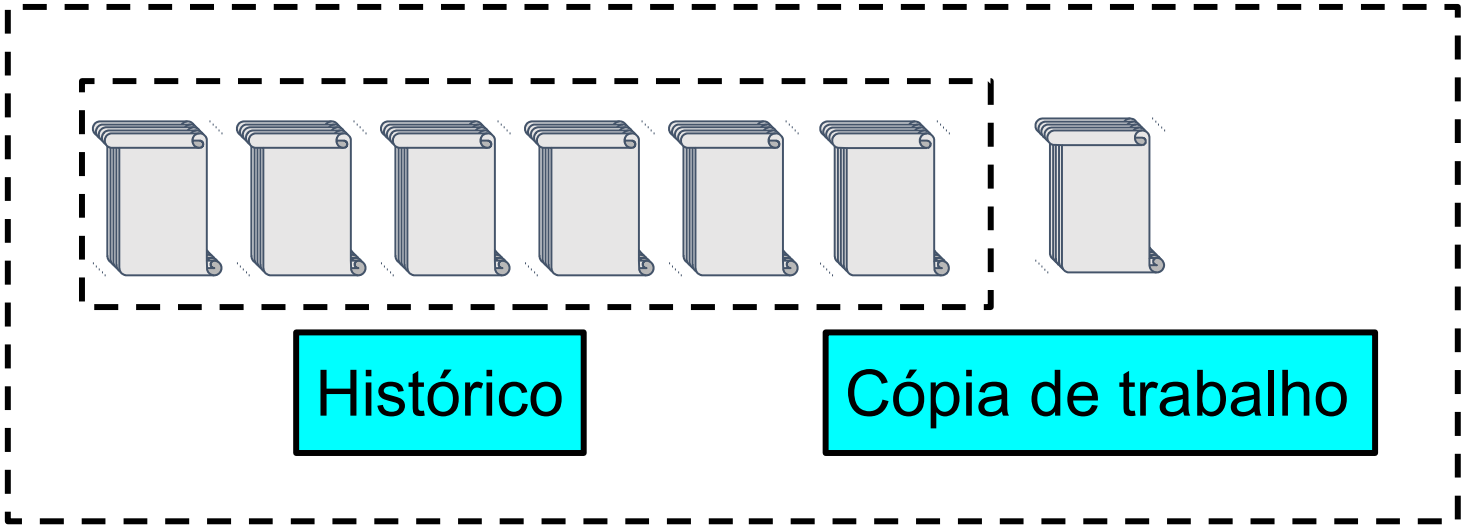
Git

- `git commit`: registra a cópia de trabalho atual como uma nova versão, junto com uma mensagem explicativa



Git

- `git commit`: registra a cópia de trabalho atual como uma nova versão, junto com uma mensagem explicativa

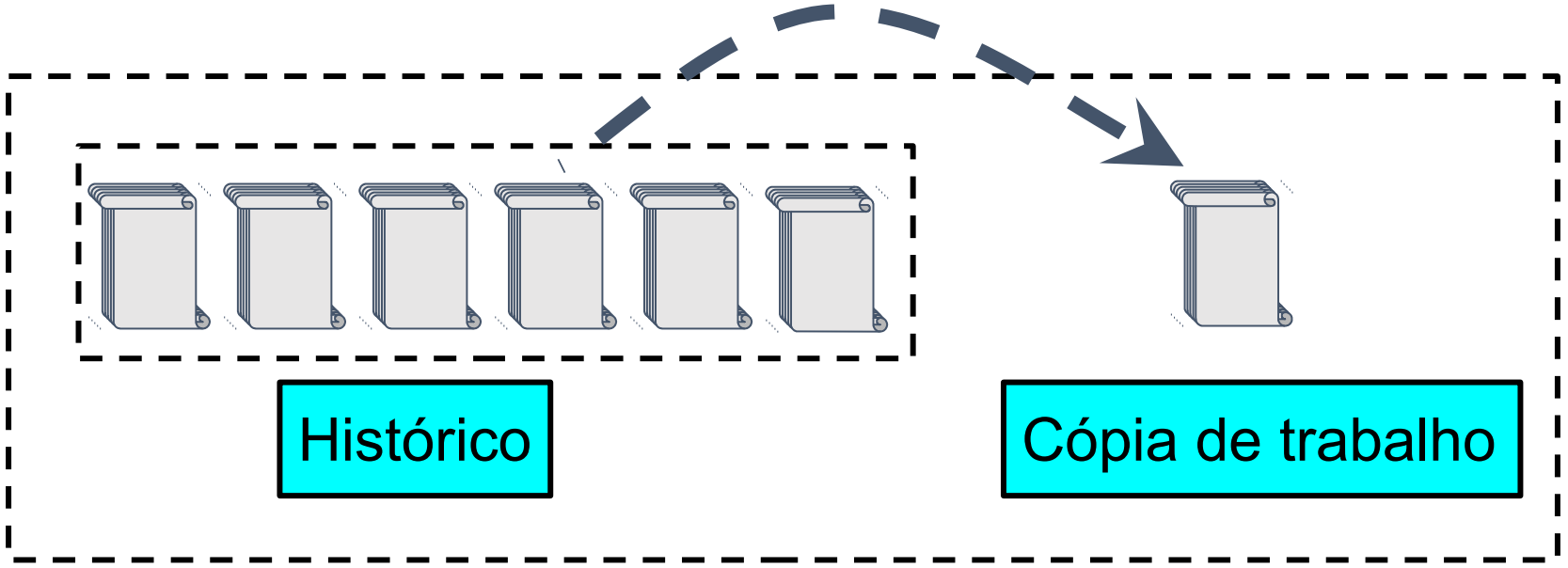


Git

- Cada versão nova registrada no repositório possui um identificador alfanumérico gerado automaticamente pelo Git
- O Git garante que nenhum identificador será repetido na história do código
- Podemos usar esse identificador para nos referir a uma versão específica e copiar a mesma para a cópia de trabalho

Git

- `git checkout`: copia uma versão do histórico para a cópia de trabalho

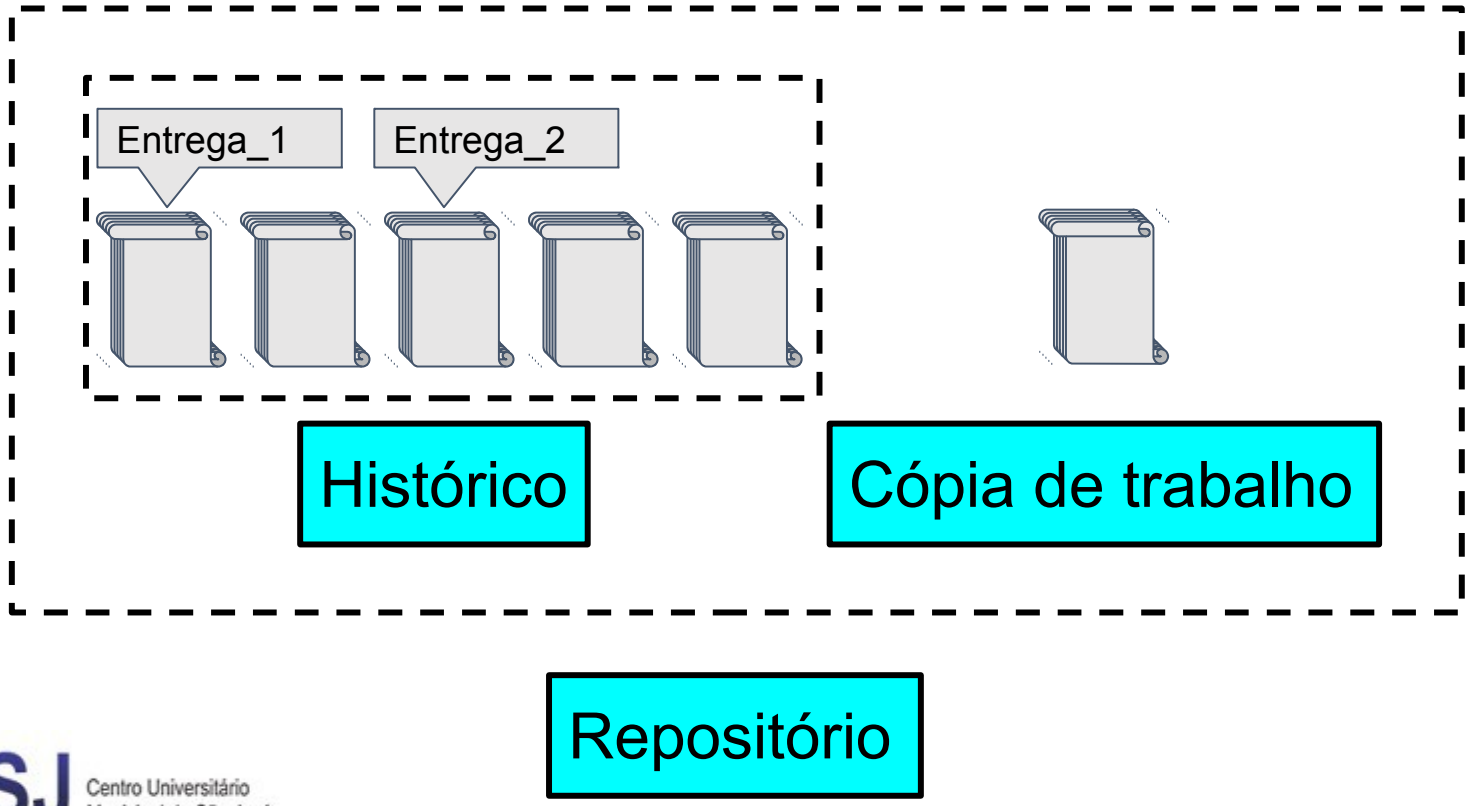


Git

- Git permite associar uma etiqueta com uma versão específica do código
- Etiquetas são anotações que podemos colocar para indicar alguma coisa sobre aquela versão
- Exemplo: Podemos adicionar uma etiqueta “Entrega_2” para dizer que aquela versão foi usada como a 2ª entrega do software para o cliente

Git

- `git tag`: adiciona uma etiqueta a uma versão



Git

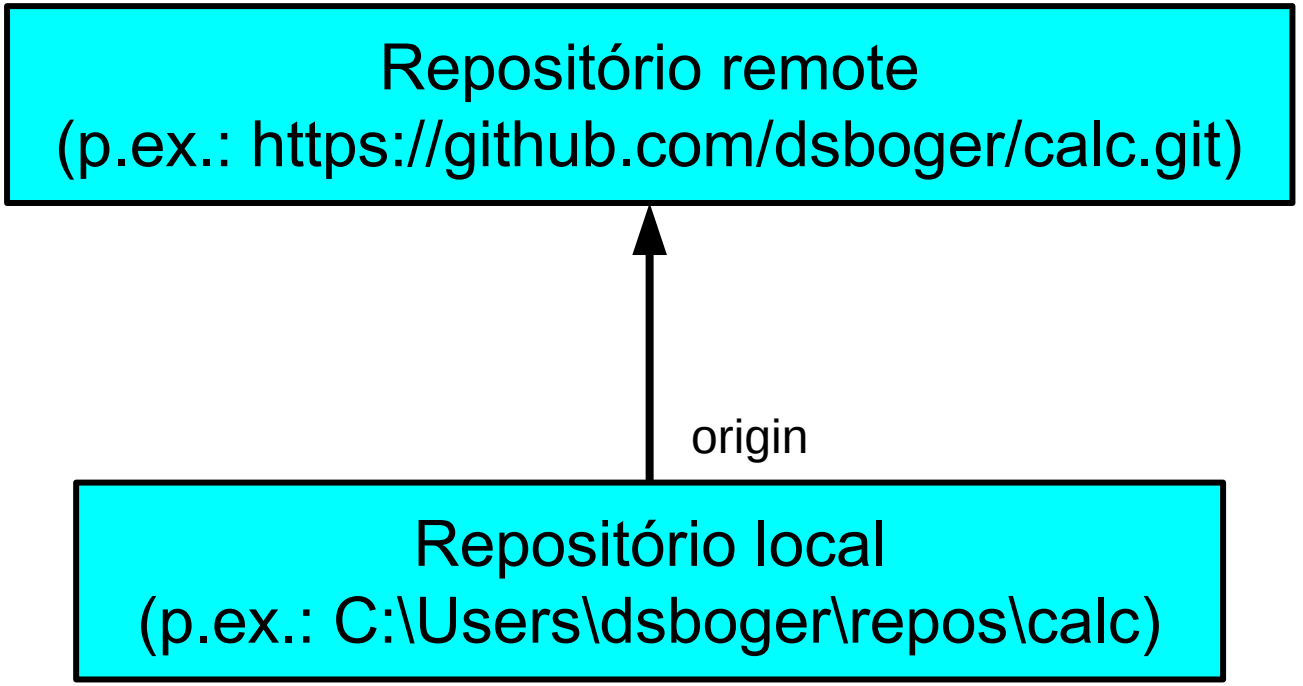
- Alguns outros comandos git:
  - `git branch`: cria um ramo no histórico
  - `git merge/rebase`: junta as modificações de dois ramos
  - `git show`: mostra os detalhes uma modificação do histórico

Git

- Colaboração com git é baseada em “Remotes”
- Um repositório pode estar conectado a outros repositórios distribuídos, chamados de remotes

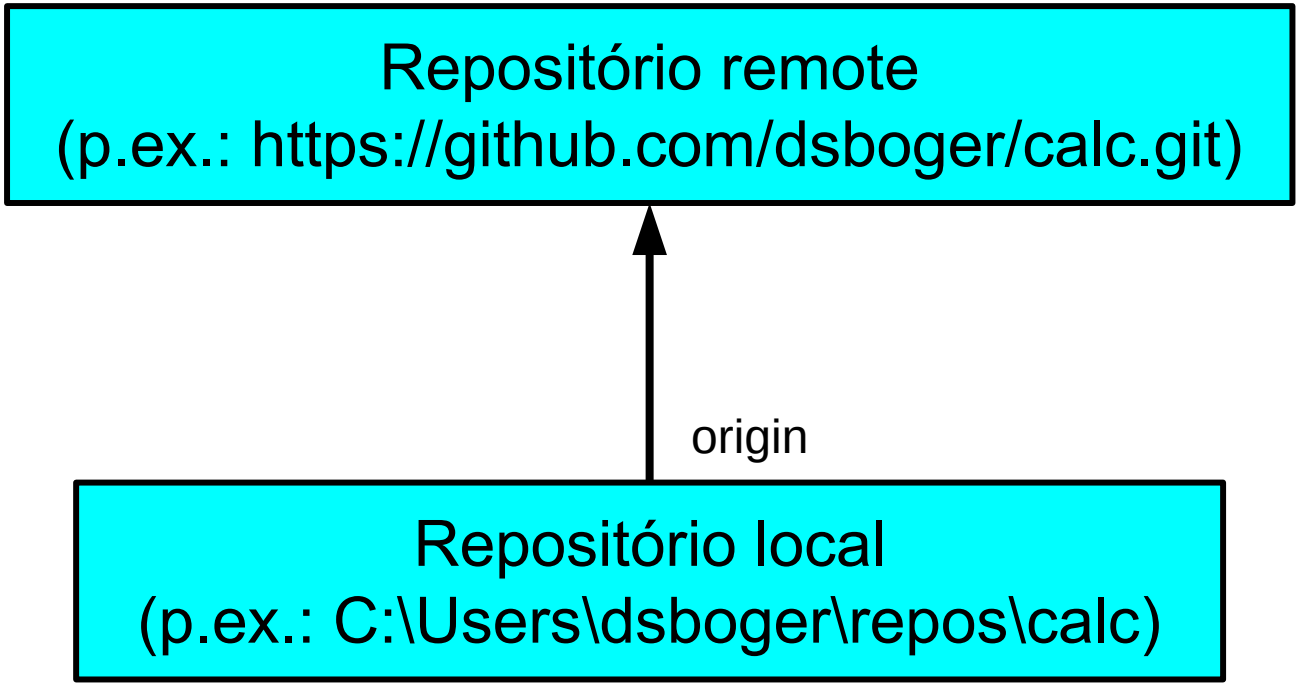
Git

- Quando copiamos um repositório inteiro (i.e. `git clone`), o repositório copiado é configurado como remote automaticamente



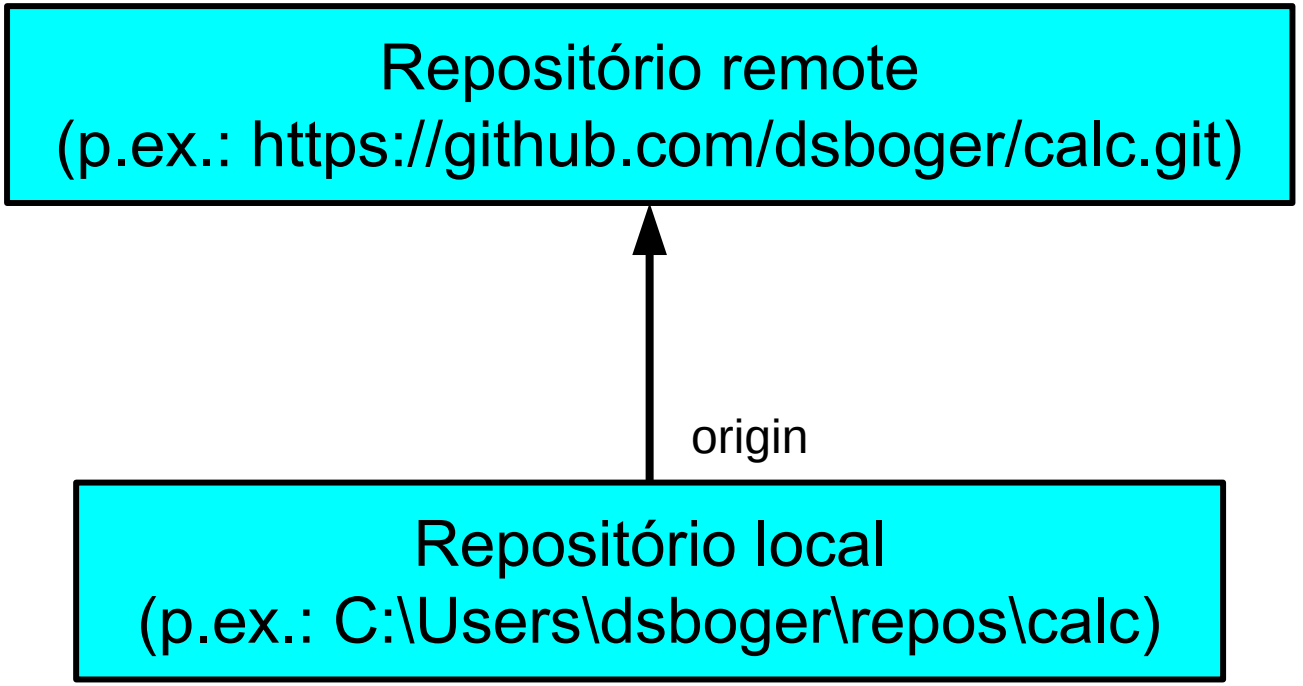
Git

- O repositório local mantém uma cópia do remote. Podemos atualizar a cópia do remote com `git fetch`



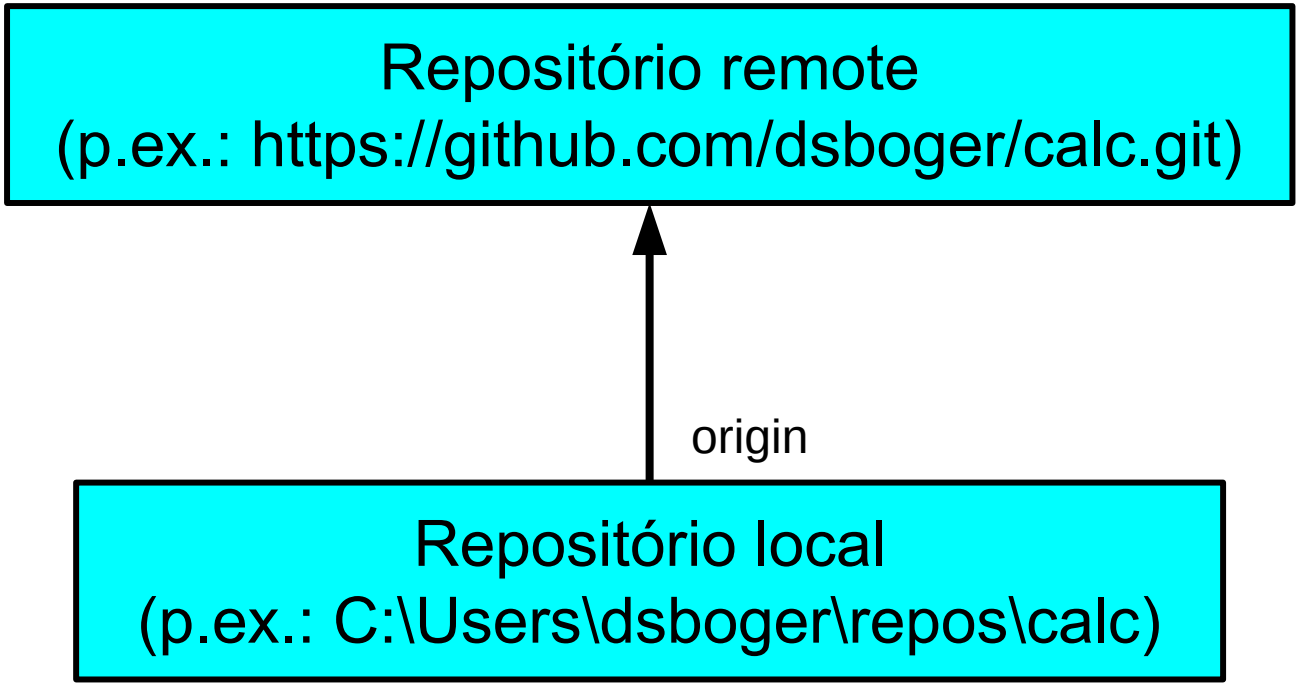
Git

- Podemos fazer `git merge/rebase` entre um ramo local e um ramo do remote, para incluir modificações feitas por outros colaboradores



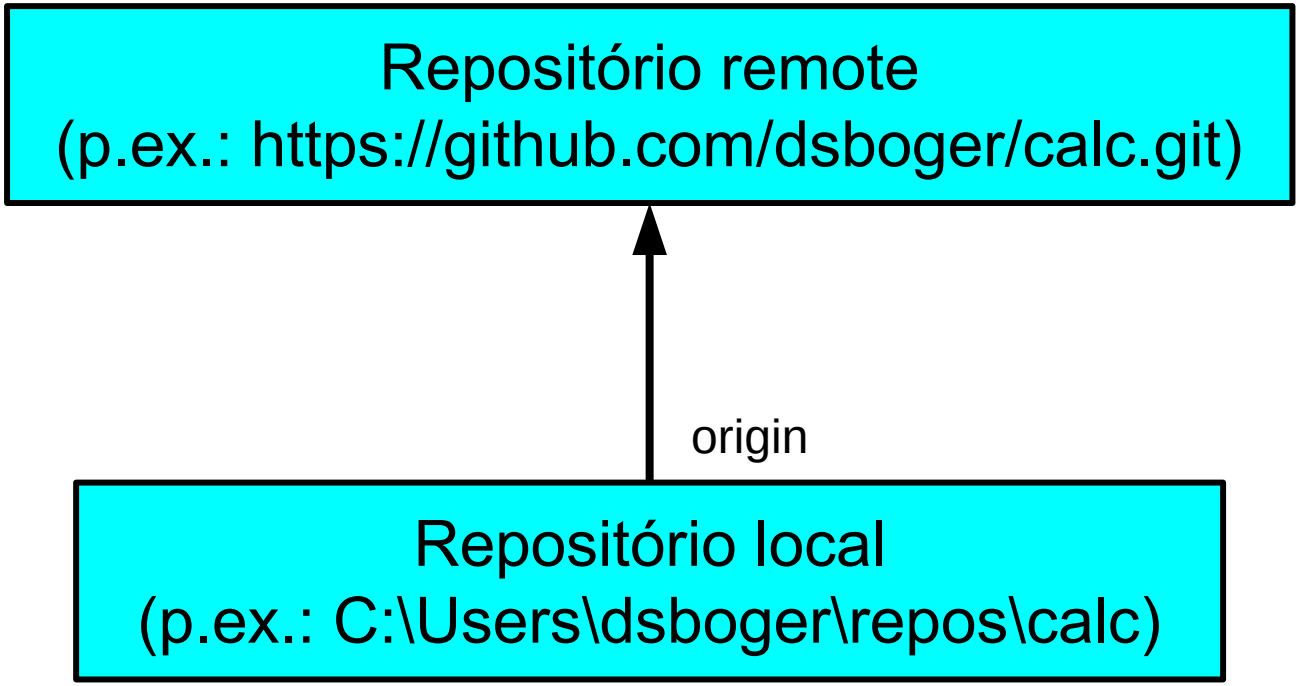
Git

- O comando `git pull` combina `git fetch` com `git merge`: baixa as atualizações do remote e aplica no repositório local



Git

- Podemos copiar modificações do repositório local para o remote usando `git push`



Git

- O que é GitHub?

Git

- O que é GitHub?
  - Pode ser visto como uma rede social de compartilhamento de código
  - Tecnicamente, é uma hospedagem de remotes, junto com uma interface Web para facilitar o gerenciamento e colaboração
  - Inclui outras funcionalidades não relacionadas com Git (p.ex. tickets)

Atividade

- Crie uma conta no GitHub
- Instale um cliente Git (pode ser a aplicação de linha de comando ou uma IDE como Eclipse ou AndroidStudio)
- Crie um repositório no GitHub
- Clone esse repositório localmente
  - git clone https://github.com/user/repo.git

Atividade

- Modifique um arquivo já existente no repositório
- Crie um arquivo no espaço de trabalho do repositório
- Crie uma versão no repositório local contendo o arquivo criado
  - git add README.md arquivo.html
  - git commit

Atividade

- Envie o repositório local para sincronização do remote do GitHub
  - git push