

Gerência de Versão e Configuração

Problema

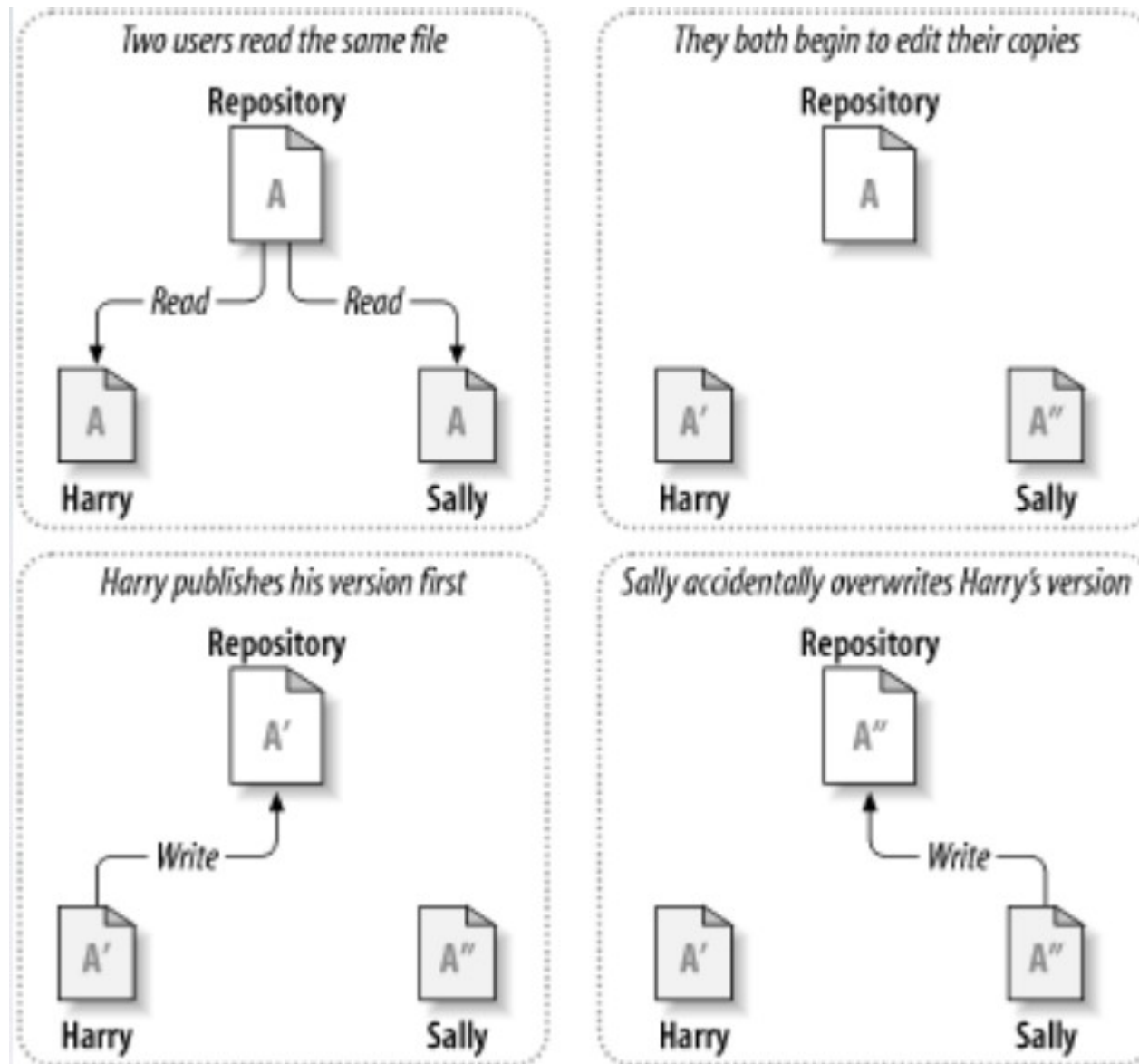
- Como desenvolver em equipe?
- Como compartilhar código e colaborar em um conjunto de artefatos comum?
- E se eu
 - precisar reverter uma alteração?
 - quiser saber como uma função estava antes da alteração?

Compartilhando código

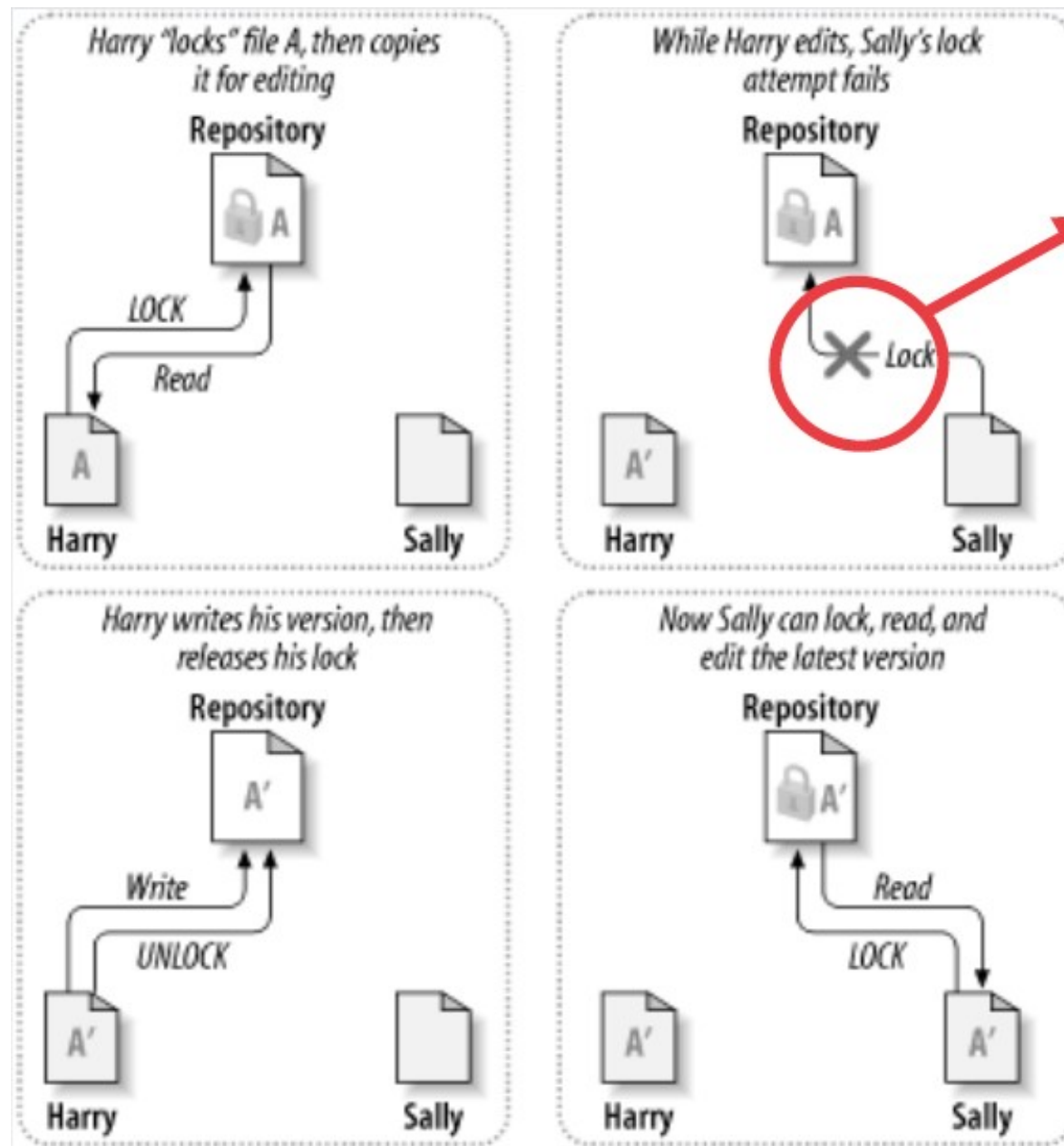


Dropbox

Problems...

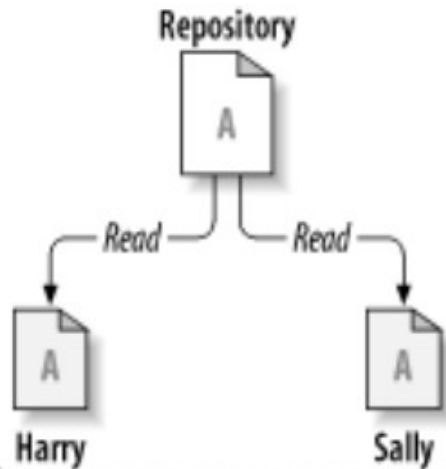


Problems...

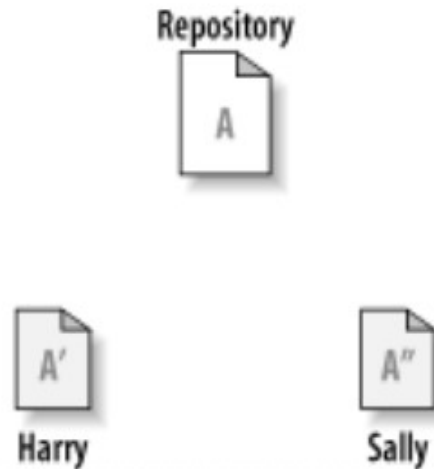


Problems...

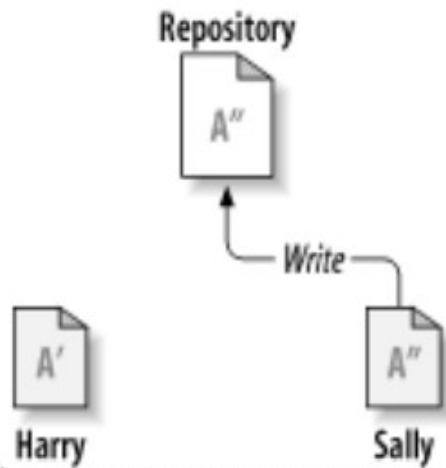
Two users copy the same file



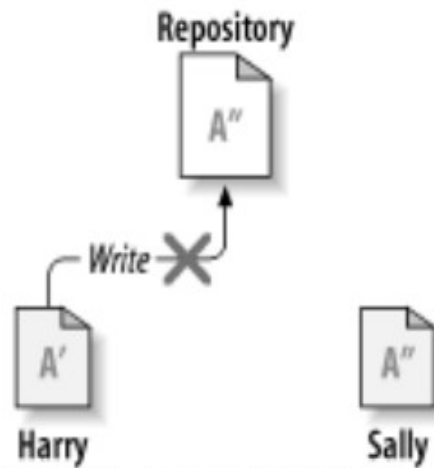
They both begin to edit their copies



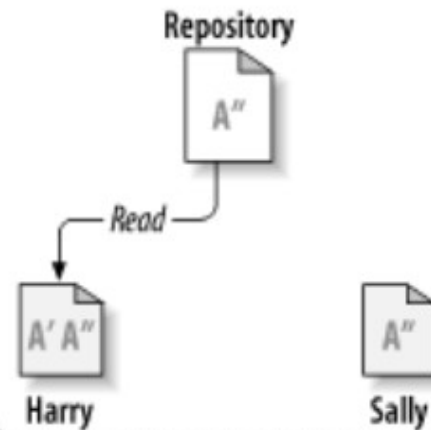
Sally publishes her version first



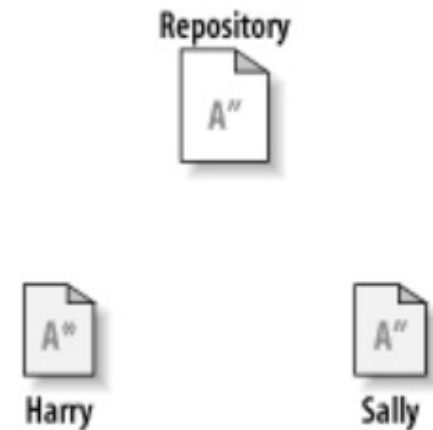
Harry gets an "out-of-date" error



Harry compares the latest version to his own



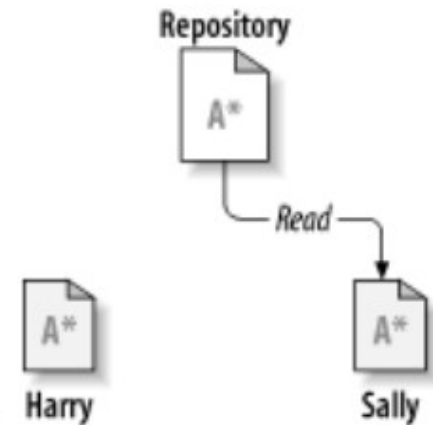
A new merged version is created



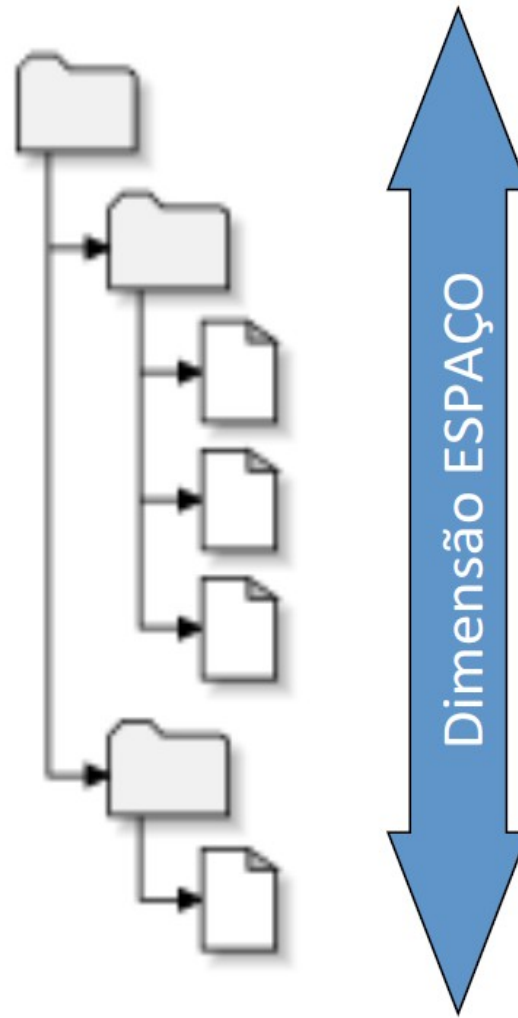
The merged version is published



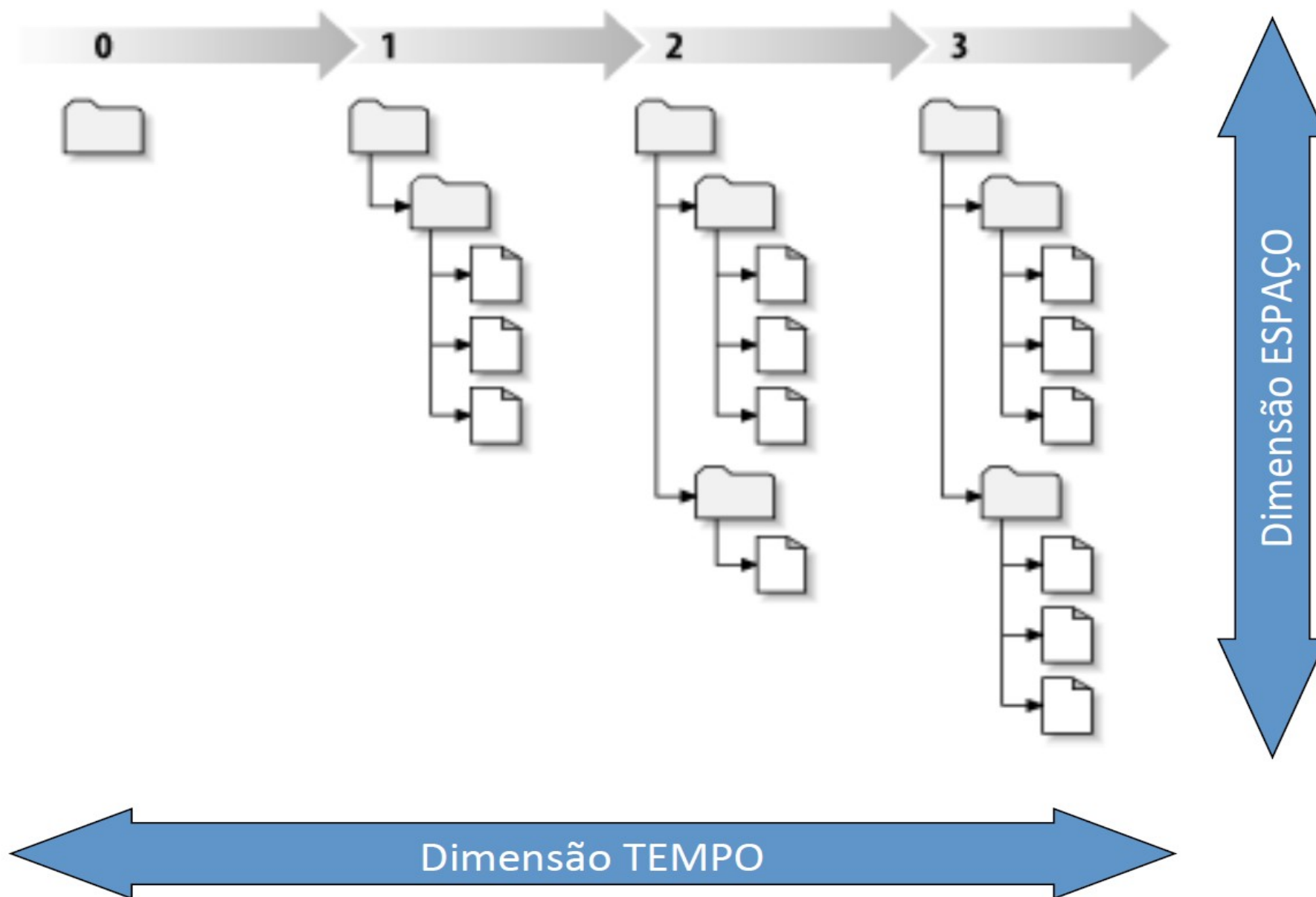
Now both users have each others' changes



Controle de Versão



Controle de Versão



Controle de Versão

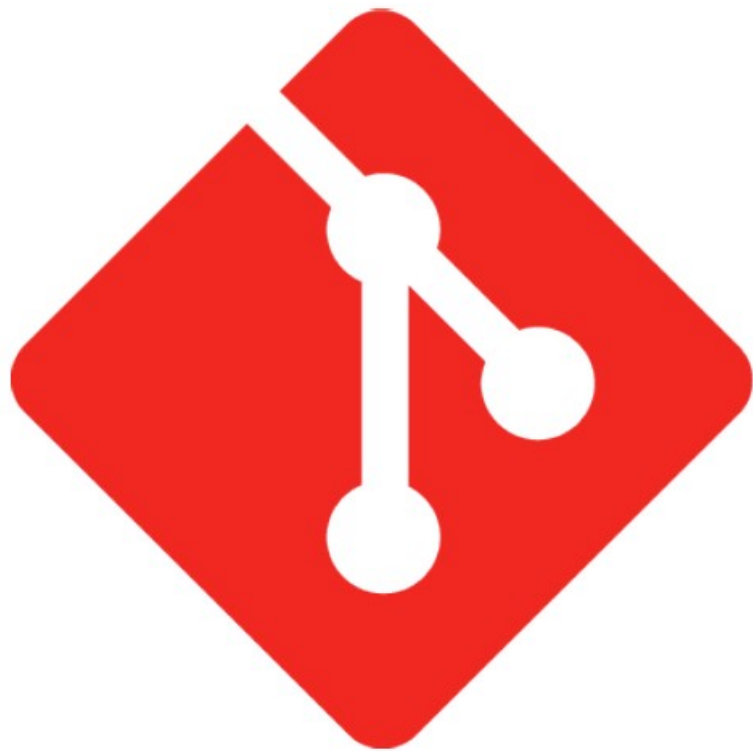


CVS



...

Nosso foco

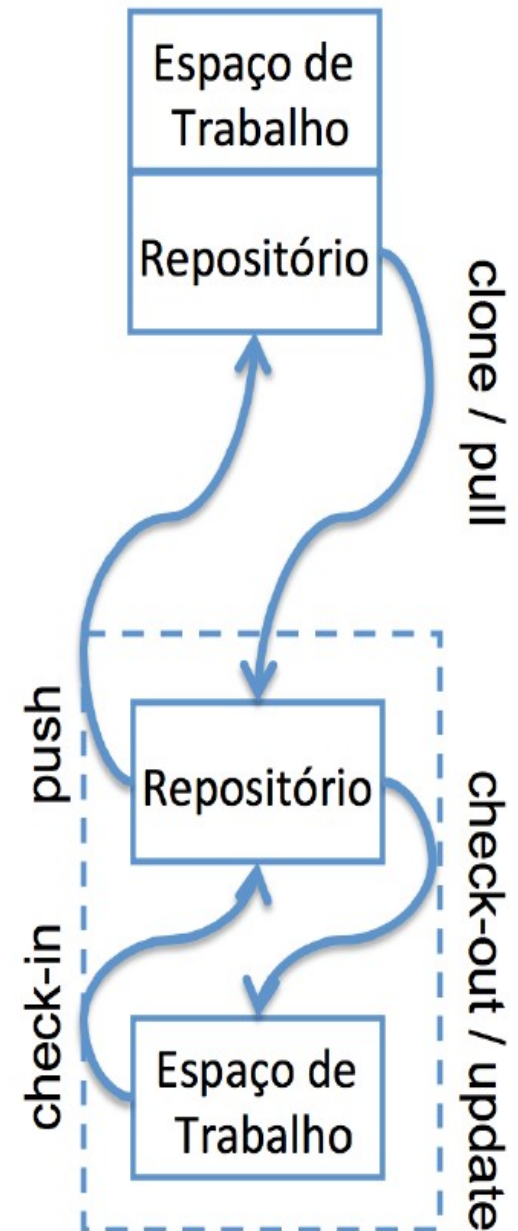
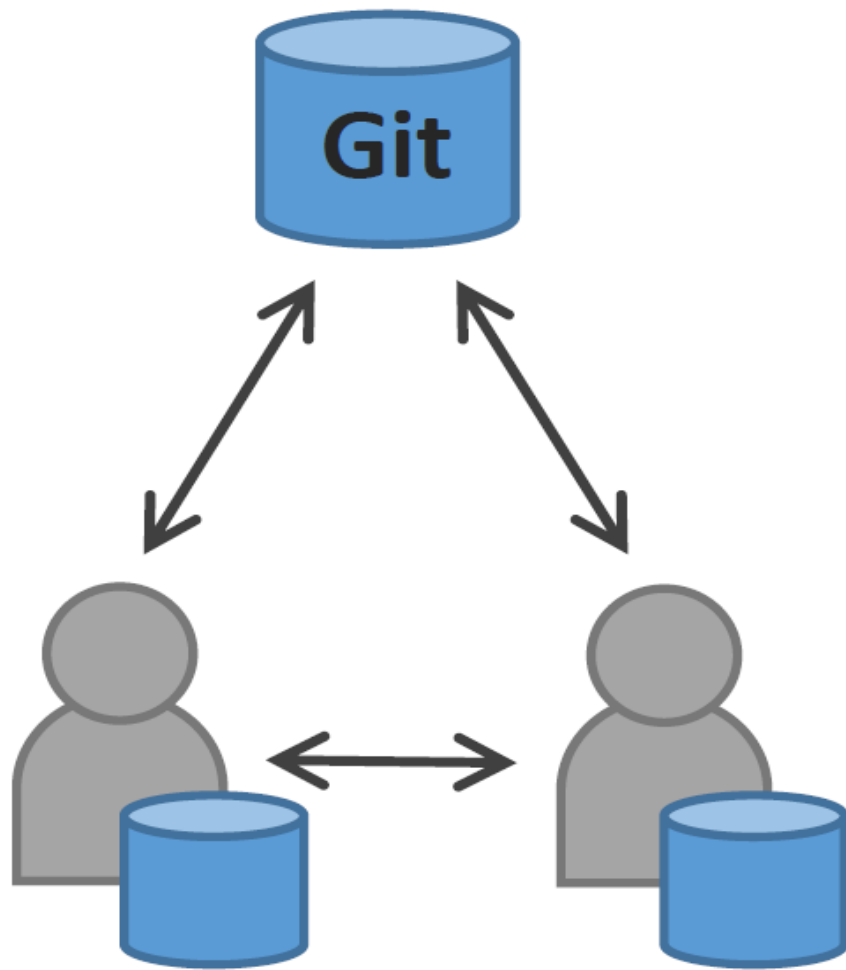


git

Quem oferece o serviço?

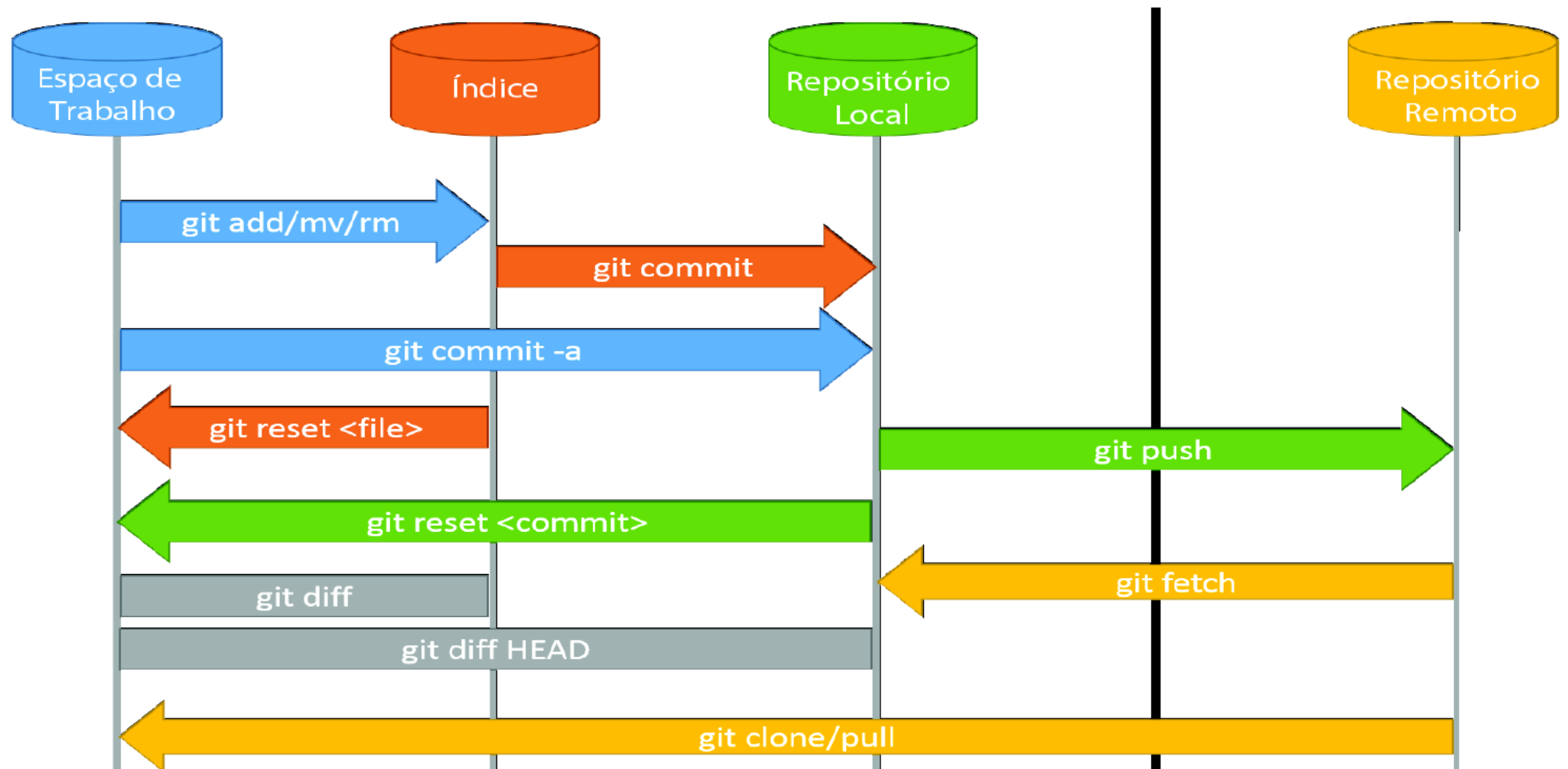
- Sua máquina?
- GitHub
- GitLab
- SourceForge.net
- BitBucket
- ...

Git: Sistema de controle de versão distribuído



Locais de Operação

- Espaço de trabalho
- Índice
- Repositório Local
- Repositório Remoto

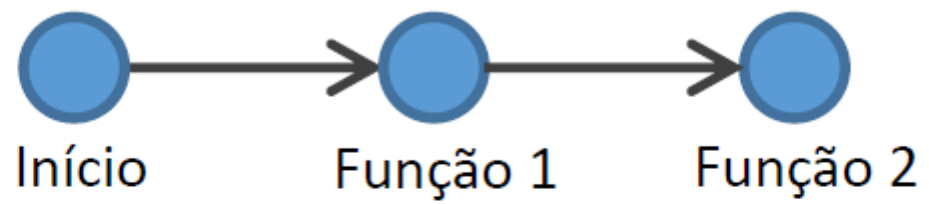





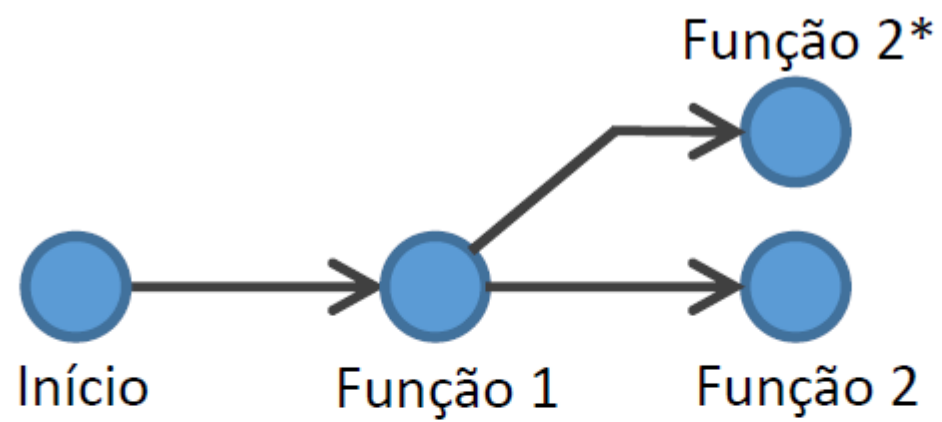
Início

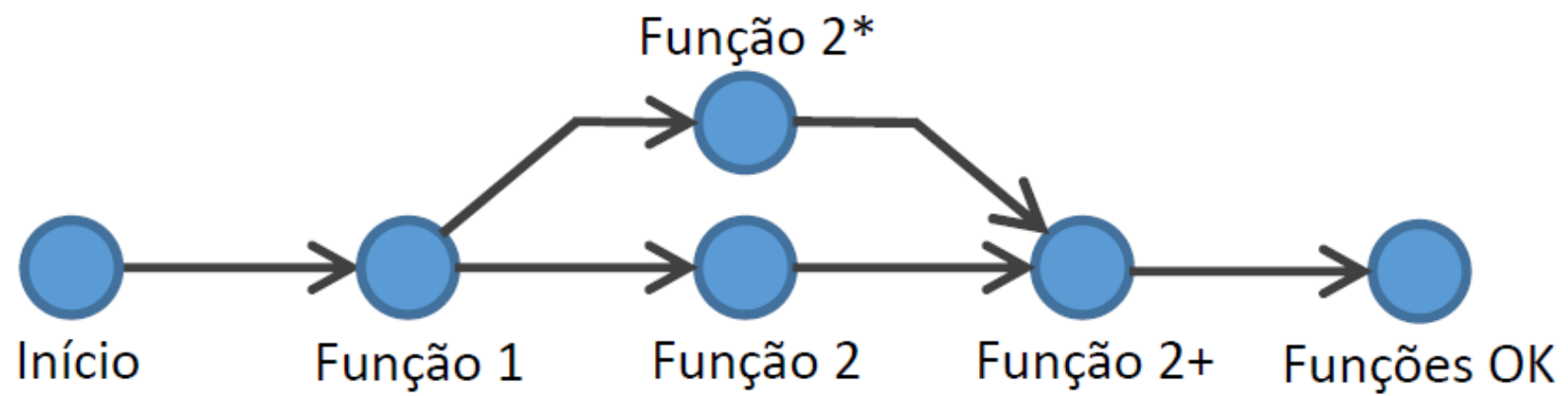


commit




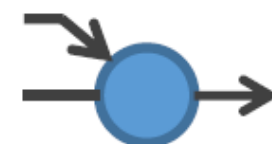
 **commit**

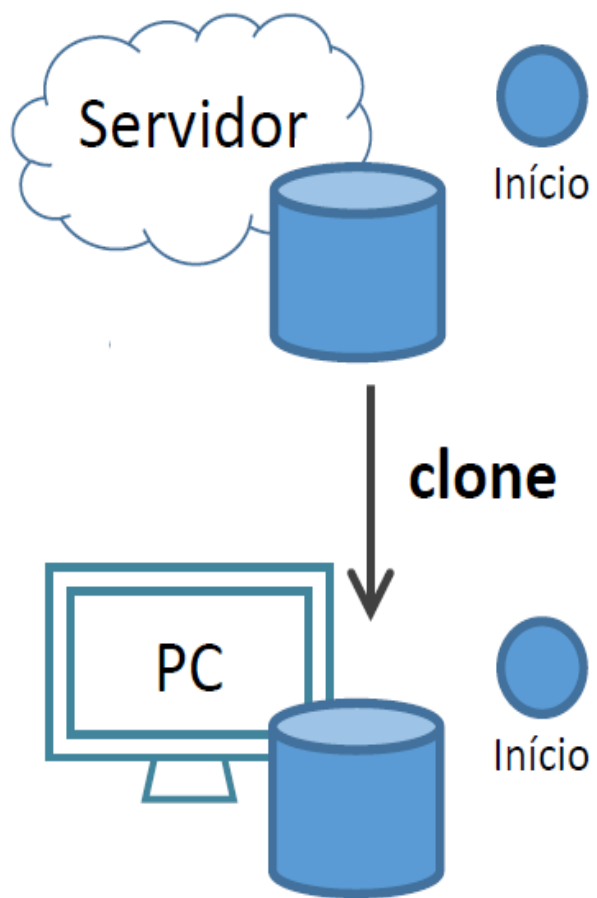


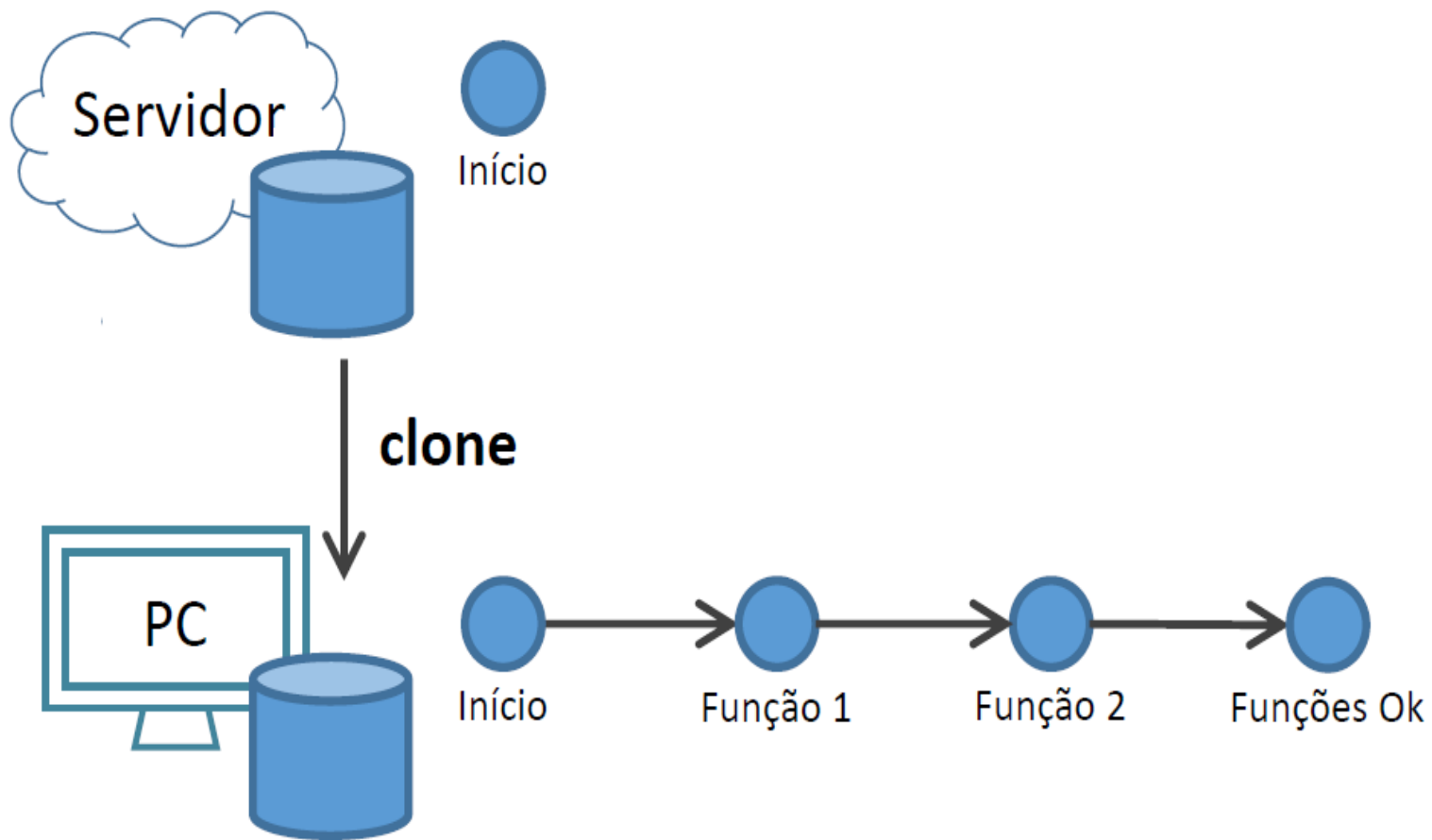


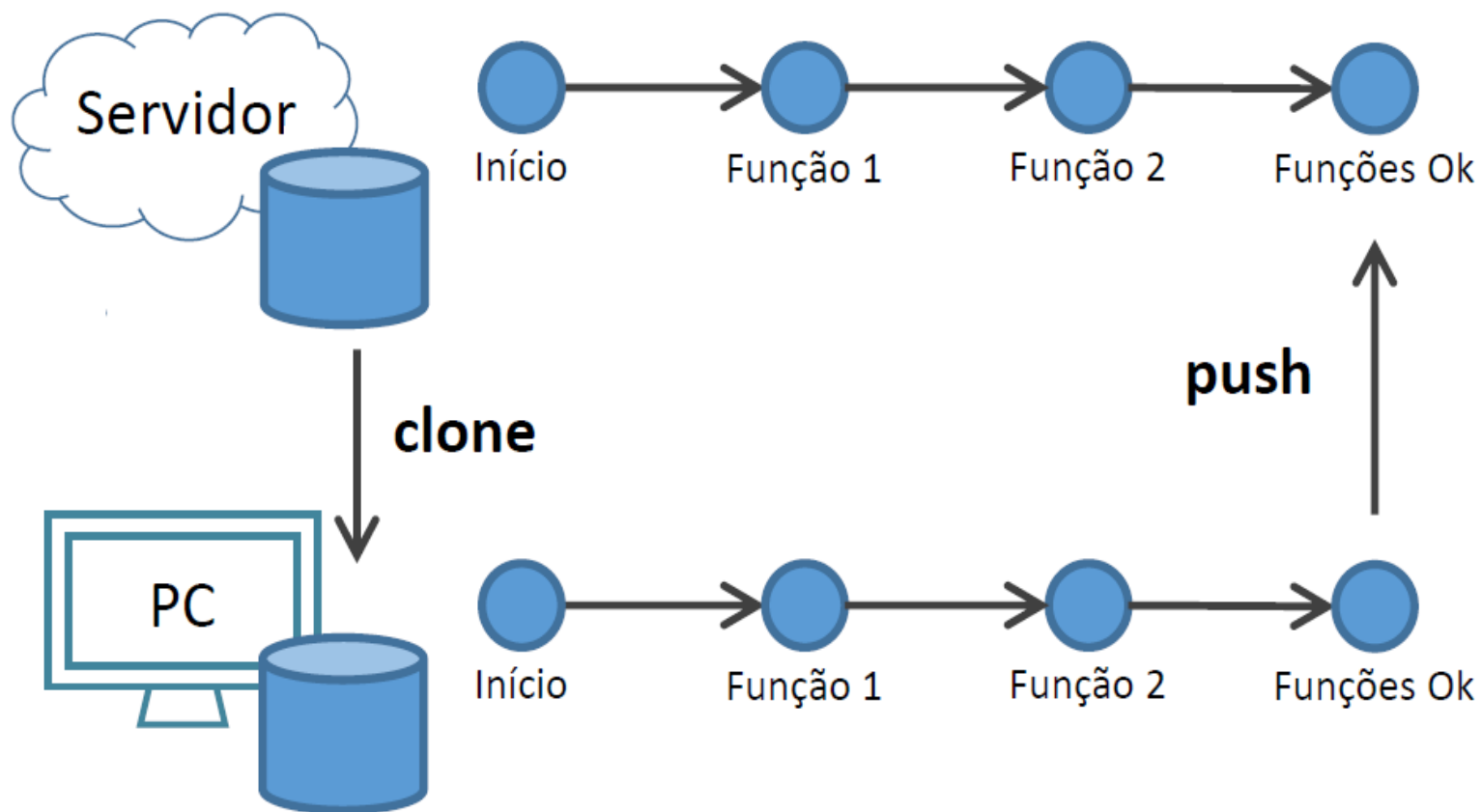
 **commit**

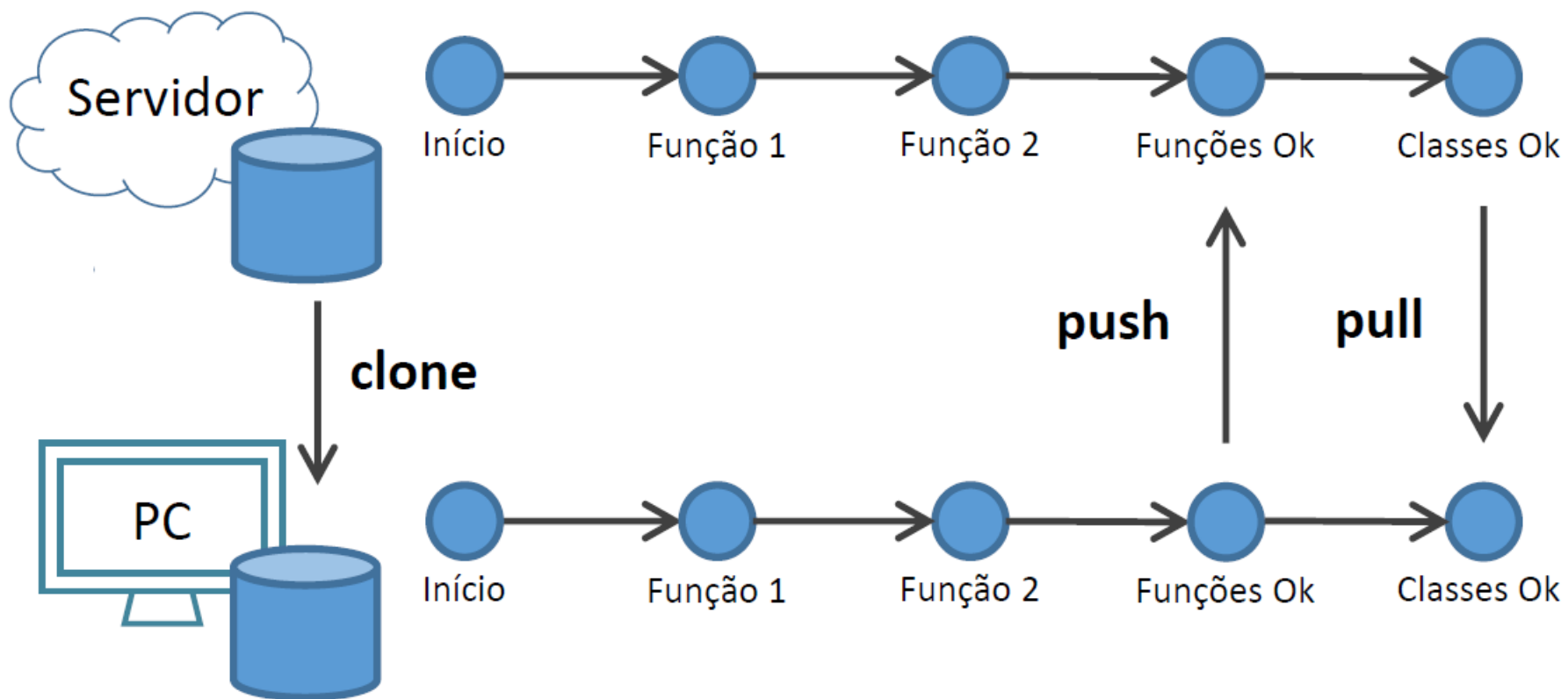
 **branch**

 **merge**









Bora praticar?!?

- Teremos 2 momentos
- Momento 1
 - Manipulação local com adições, commits, branches, remoções, conflitos
- Momento 2
 - Repositório remoto com pushes e pulls

Hands on!

- `git config --global user.name "Igor Steinmacher"`
- `git config --global user.email "igorfs@utfpr.edu.br"`
- Criar um diretório
- Entrar no diretório
- Transformar em um repositório git
 - `git init`

Hands on!

- Temos um repositório!!!
- Vamos criar um arquivo no repositório
 - `touch teste.txt`
- Verificando o status atual dos arquivos
 - `git status`
- Precisamos adicionar o arquivo
 - `git add teste.txt`
 - `git status`

Hands on!

- Fazendo um 'commit' para o repositório local
 - `git commit -a -m "Commit: primeiro!"`
 - -a: todos os arquivos
 - -m adicionar uma mensagem ao commit
- Vamos listar os últimos commits realizados
 - `git log`
- Ou ver o que foi feito no último commit
 - `git show`

Hands on!

- Mudem o arquivo teste.txt inserindo “Olá” na primeira linha
- Verifiquem o status do repositório
- Commitem o arquivo
- E verifiquem o status novamente

Hands on: Branches

- Você pode criar ramos de seu repositório
- Todo o repositório tem um branch padrão chamado master
- Listando os branches
 - `git branch`
- E dá para criar outros branches
 - `git branch novo`
- E para usar o branch novo
 - `git checkout novo`

Hands on!

- Vamos alterar o arquivo teste.txt nesse branch adicionando na segunda linha:
 - Meu nome é Peter
- E vamos commitar as alterações nesse branch
 - `git commit -a -m "Saudacao do Peter no teste.txt"`
- Vejam o conteúdo do arquivo
 - `cat teste.txt`
- Mudem para o branch master
- Deem um `cat teste.txt`

Hands on!

- PROBLEMA!!! Eu alterei no meu branch de trabalho...
 - Meu master está desatualizado :(
- Precisamos fazer uma mesclagem das alterações no master
 - Mudamos para o branch master
 - E fazemos a mesclagem
 - git merge working

Hands on!

- Imaginem um problema maior!!!
 - O arquivo já tinha sido alterado no branch master...
 - Tentando fazer igualzinho:

Auto-merging teste.txt

CONFLICT (content): Merge conflict in teste.txt

Automatic merge failed; fix conflicts and then commit the result.

- cat teste.txt

Olá!

Meu nome é Peter

<<<<<<< HEAD

Eu moro em SP

=====

Eu moro lá

>>>>>>> working

Hands on!

- E agora?!?! Precisamos fazer o merge manual...
 - Editar o arquivo
 - E fazer um commit local
- Mas, e o outro branch?
 - Está desatualizado!!!
 - Atualizando
 - `git rebase master`

Hands on: Repositórios Remotos

- Similar ao local, mas o repositório remoto está...
 - Remoto
- Vamos utilizar o github nessa investida
 - Criem suas contas



github
SOCIAL CODING

Hands on: Repositórios Remotos

- Para criar um clone (cópia de todo o projeto, incluindo todos commits) devemos utilizar o seguinte comando:
 - git clone <https://github.com/Usuario/Projeto>
- Vamos clonar o repositório que eu criei, bem bonito
 - <https://github.com/igorsteinmacher/githandson.git>

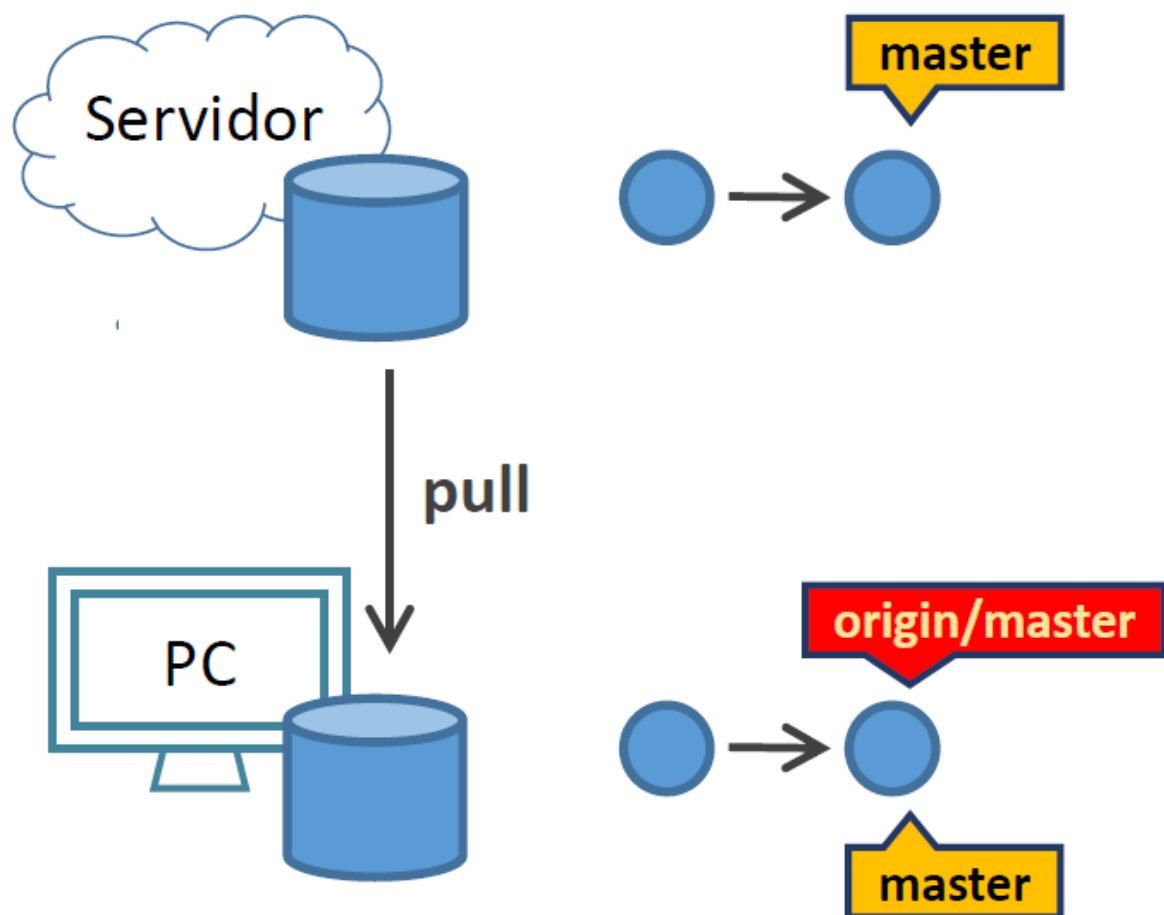
Hands on: Repositórios Remotos

- Entremos no diretório e vamos trabalhar
 - `cd githandson`
 - `git branch -r` //para ver os branches remotos
- Para manipular as coisas do repositório remoto:
 - `git pull` //traz tudo e atualiza o repositório local
 - `git fetch` //baixa os dados do repo remoto
 - `git push` //manda coisas pro repo remoto
 - `git push <repo> :<branch>` //remove branch do repo remoto
 - `git push <repo> <branch>` //envia um branch específico para o repo

Hands on: Repositórios Remotos

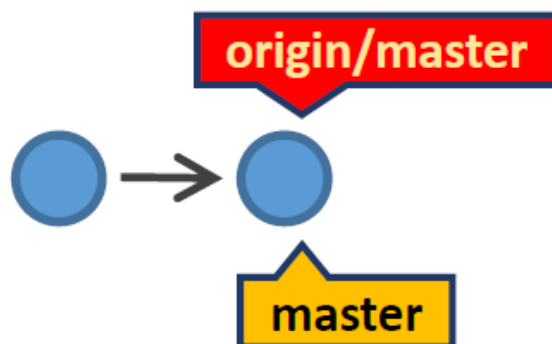
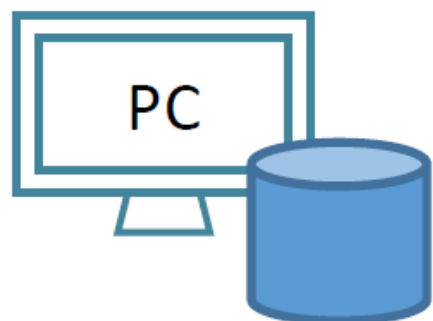
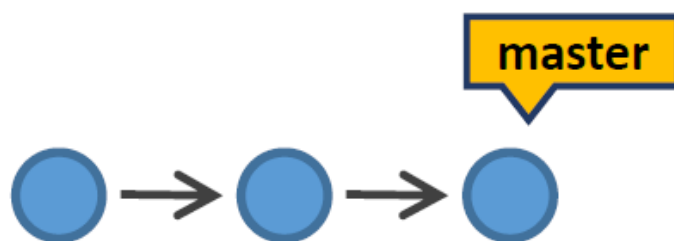
- No github, em geral trabalhamos com o esquema “fork + pull-request”
 - Criação de uma cópia do projeto + envio das mudanças de sua cópia para o projeto principal
- Para isso, vamos fazer um fork do meu projeto do github
 - NA TELA!

Exemplo de Aplicação



```
$ git pull #Atualiza o repositório local
```

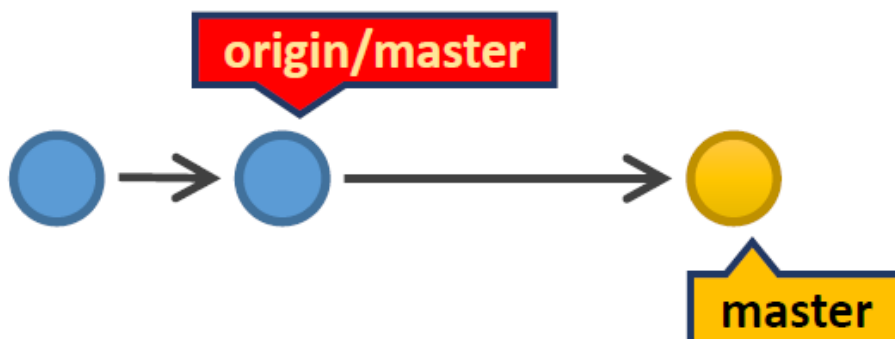
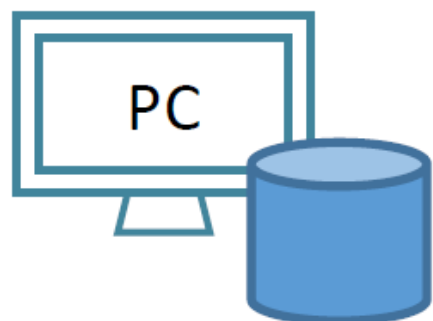
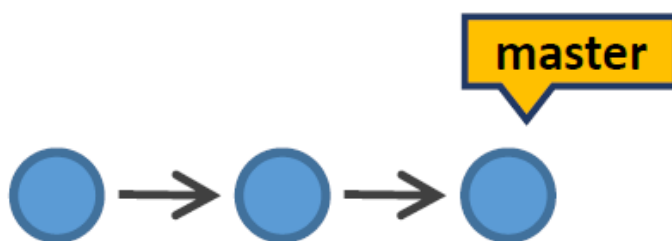
Exemplo de Aplicação



#Modifica o repositório remoto

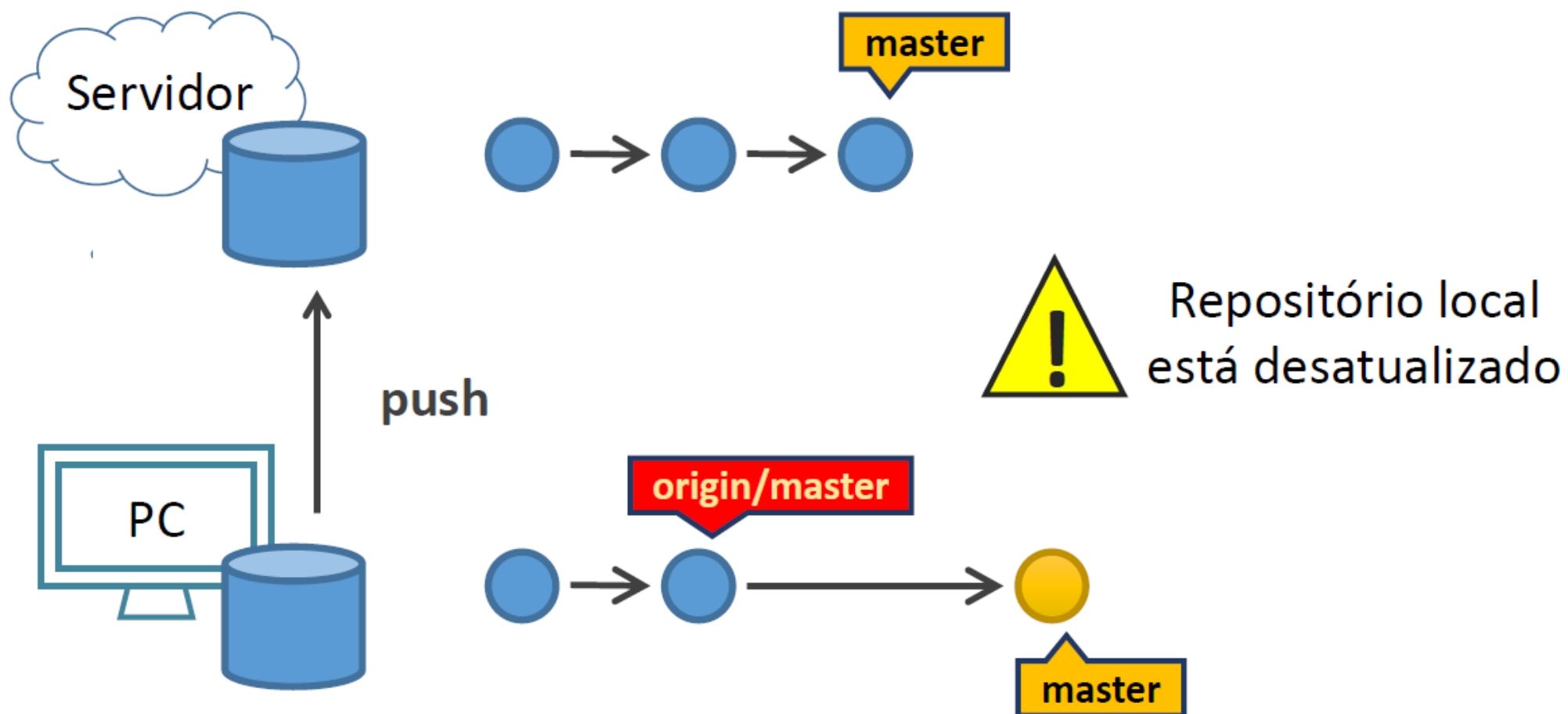


Exemplo de Aplicação



```
$ git commit -a #Altera o repositório local
```

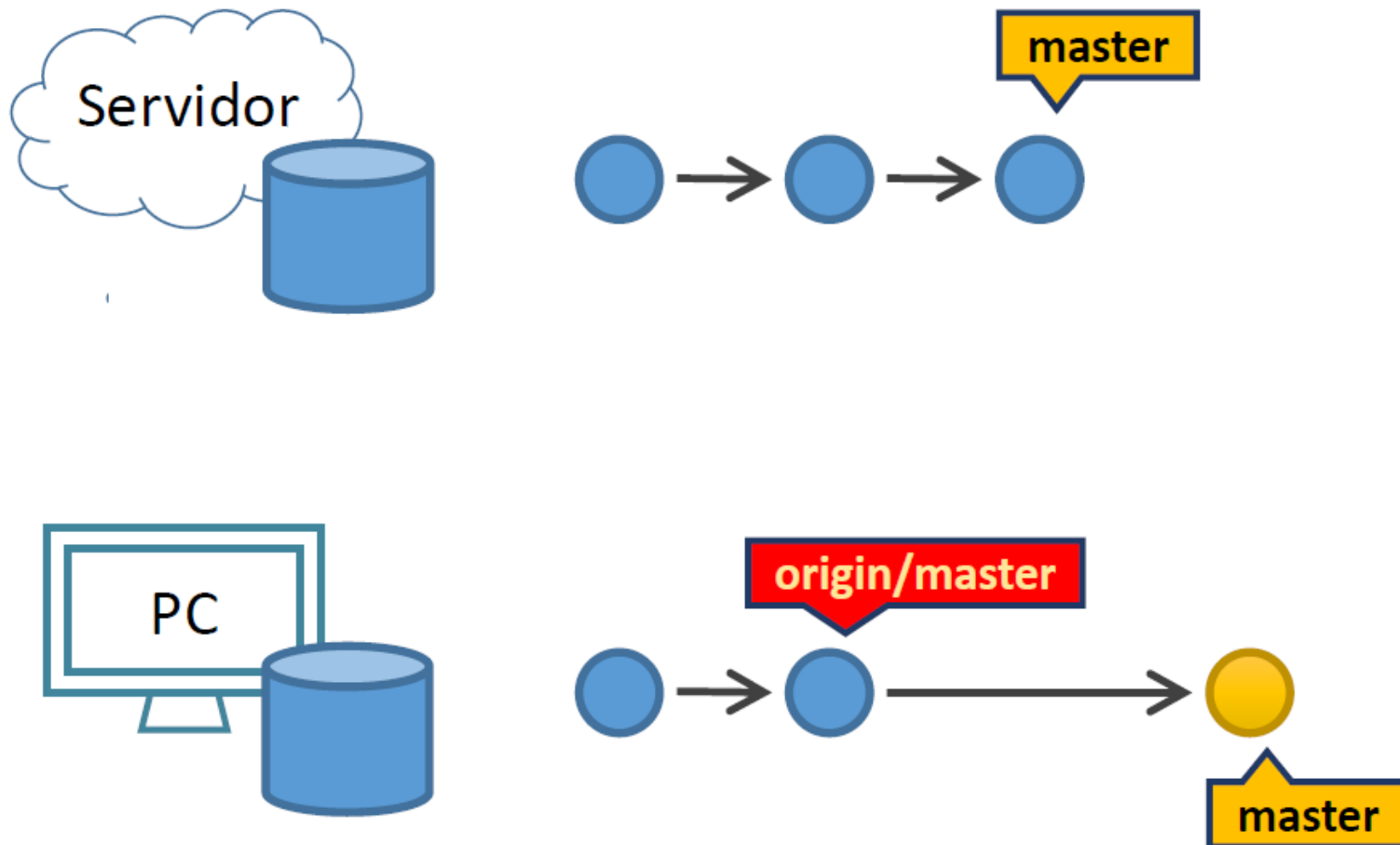
Exemplo de Aplicação



```
$ git push #Tenta atualizar o servidor
```

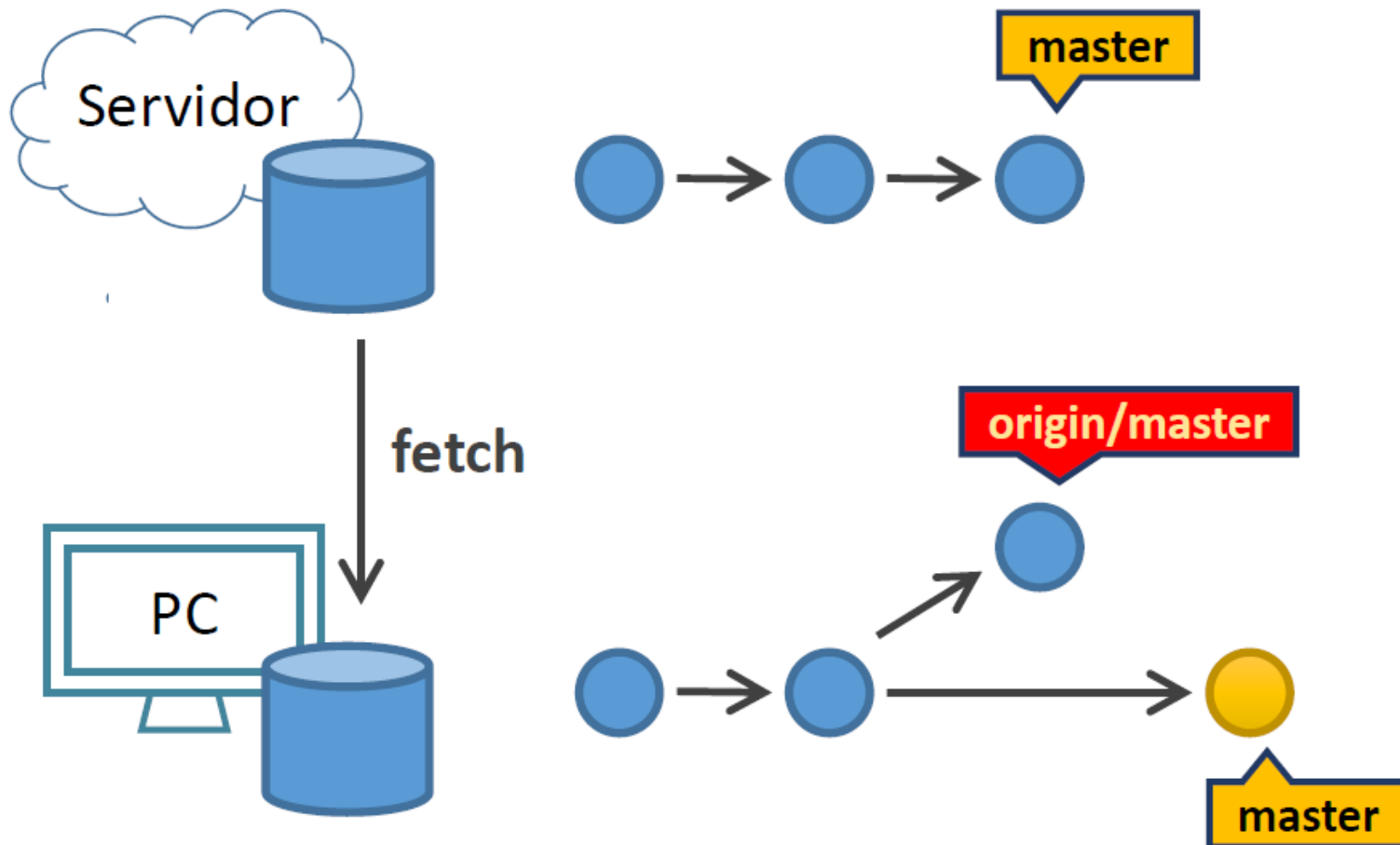



Solução 1: *fetch + rebase + push*





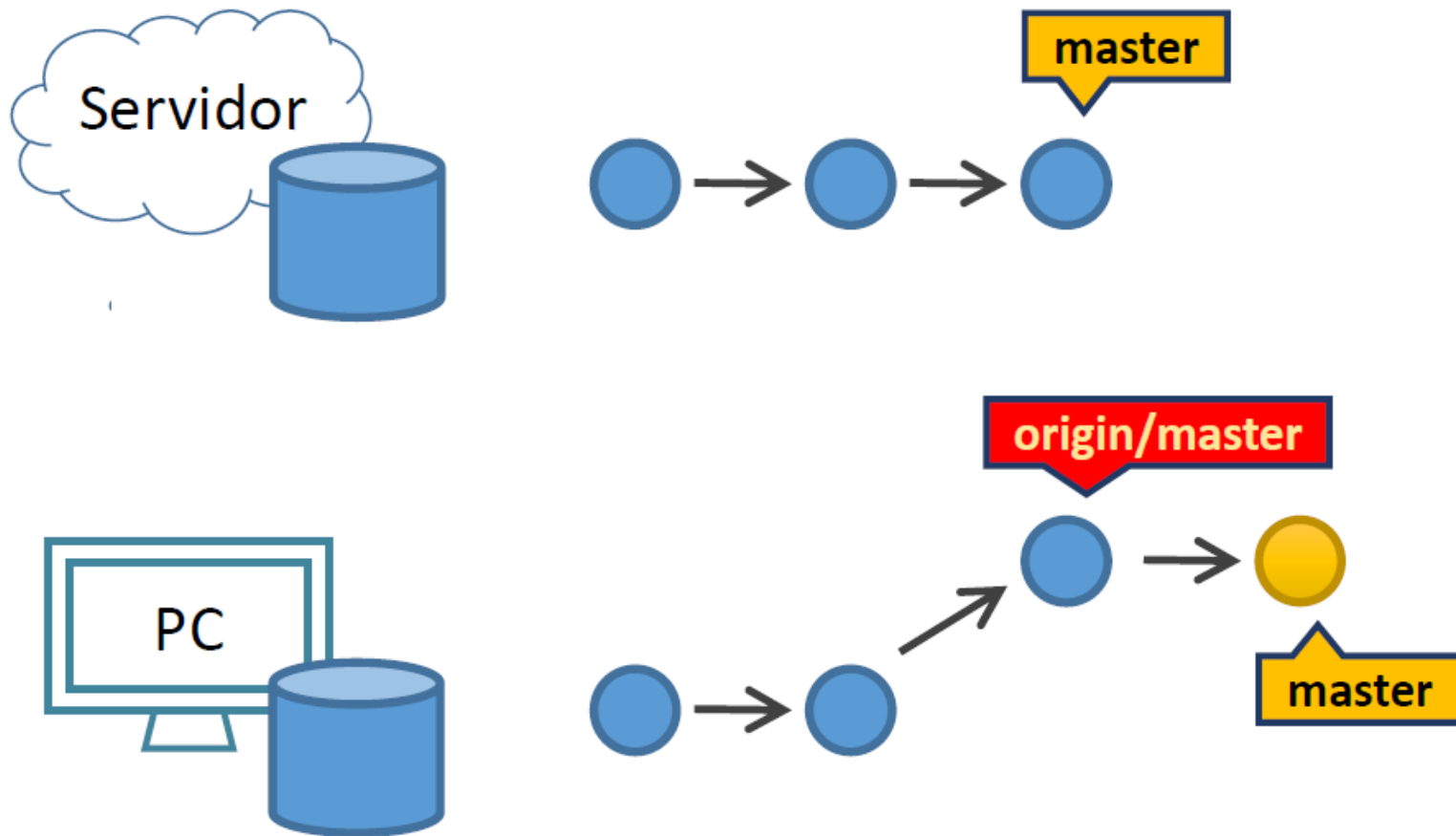
Solução 1: *fetch* + *rebase* + *push*



```
$ git fetch #Baixa os dados do servidor
```



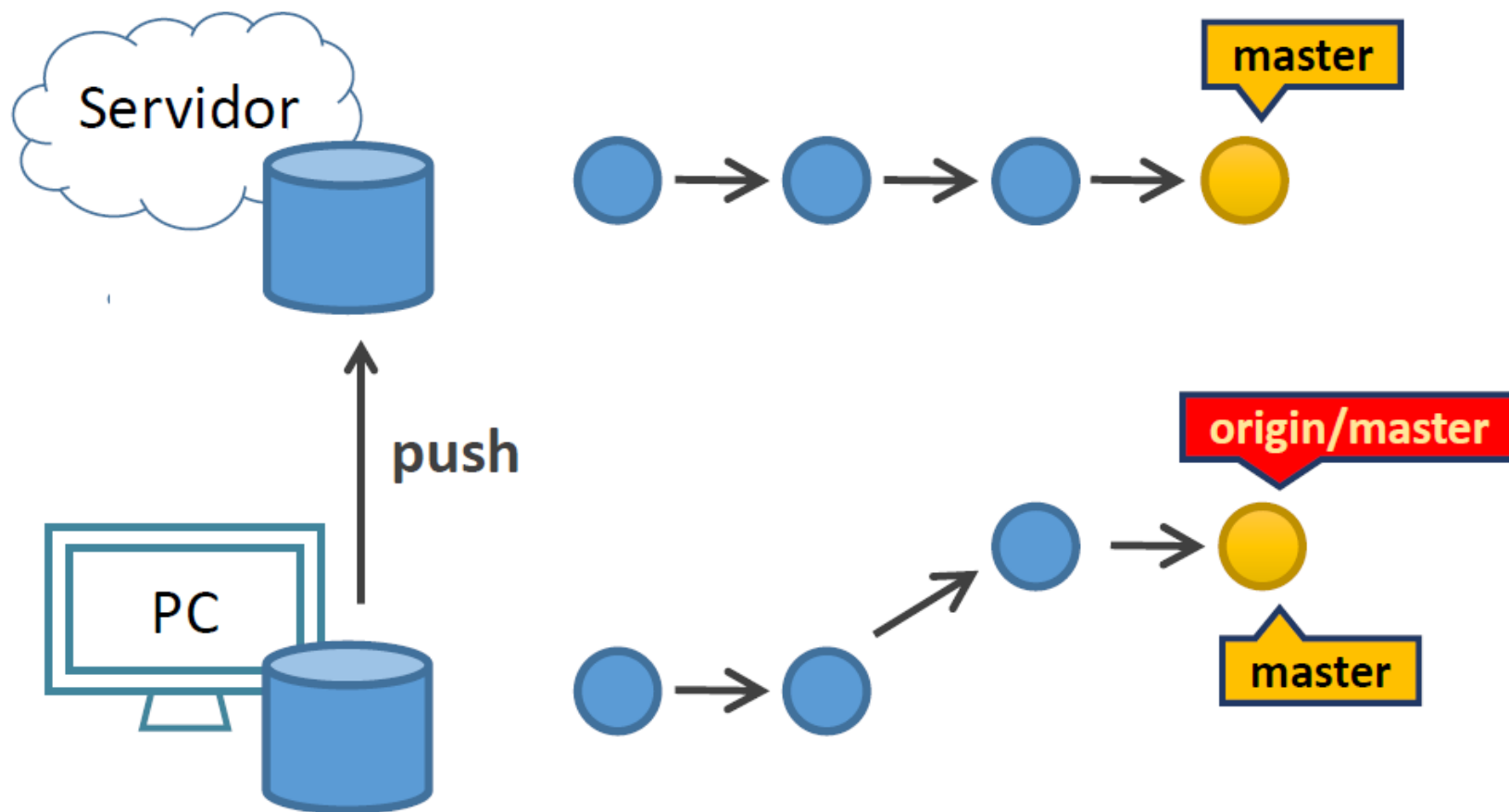
Solução 1: *fetch + rebase + push*



```
$ git rebase origin/master #Realiza o rebase
```

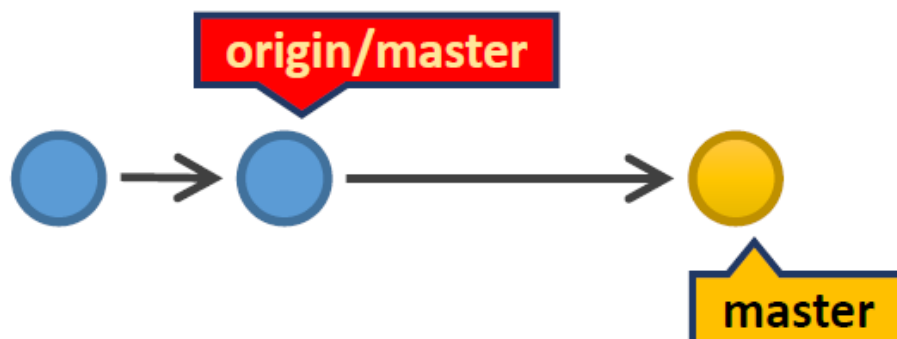
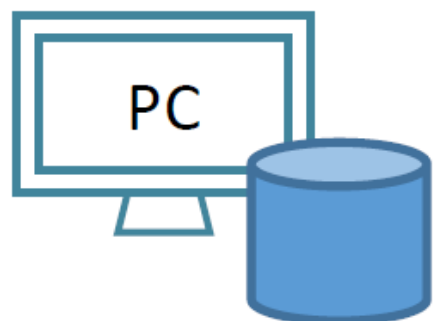
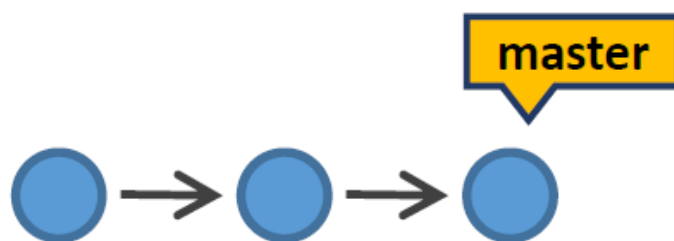


Solução 1: *fetch + rebase + push*



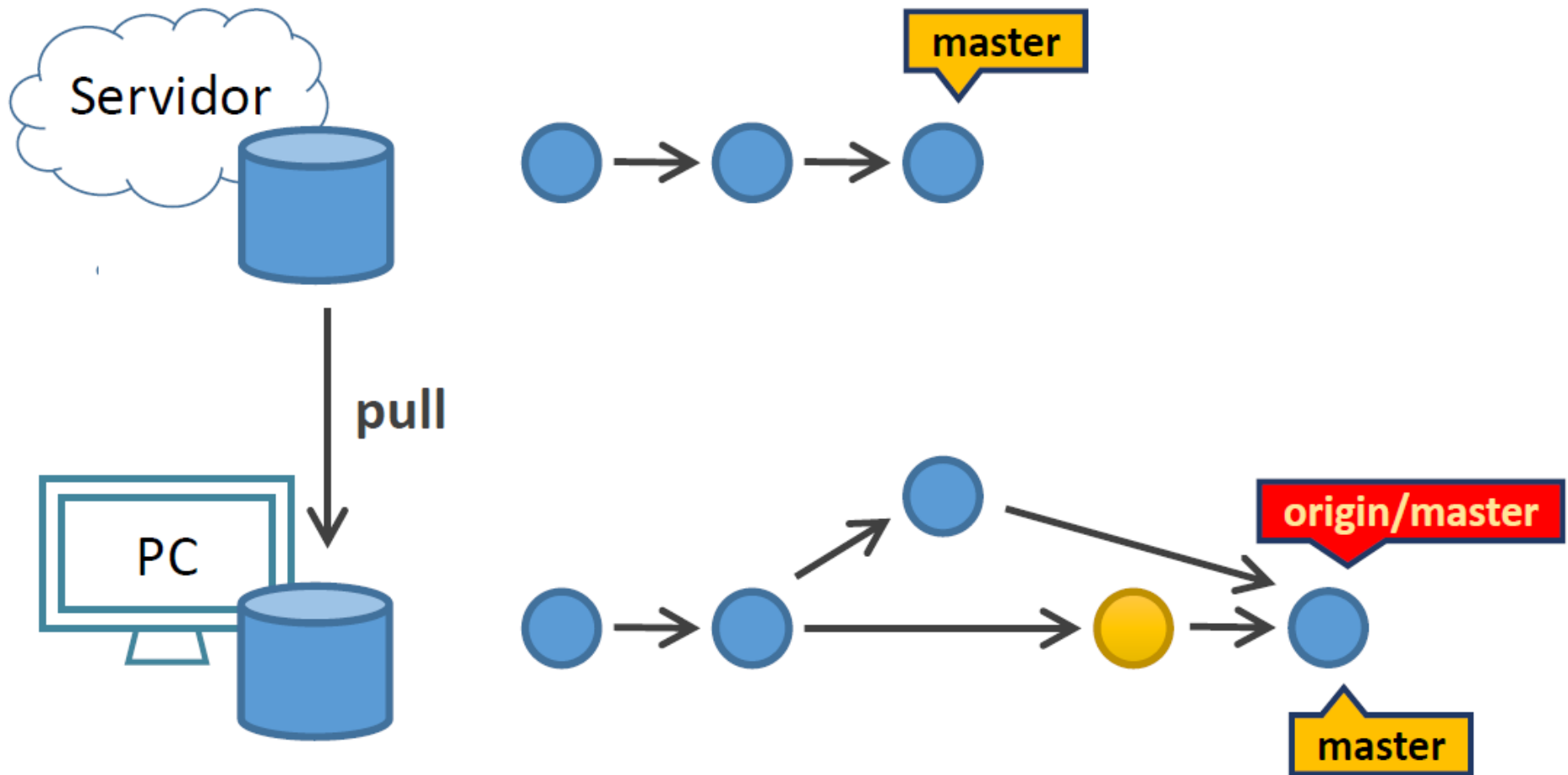
```
$ git push #Envia para o servidor
```

Solução 2: *pull + push*



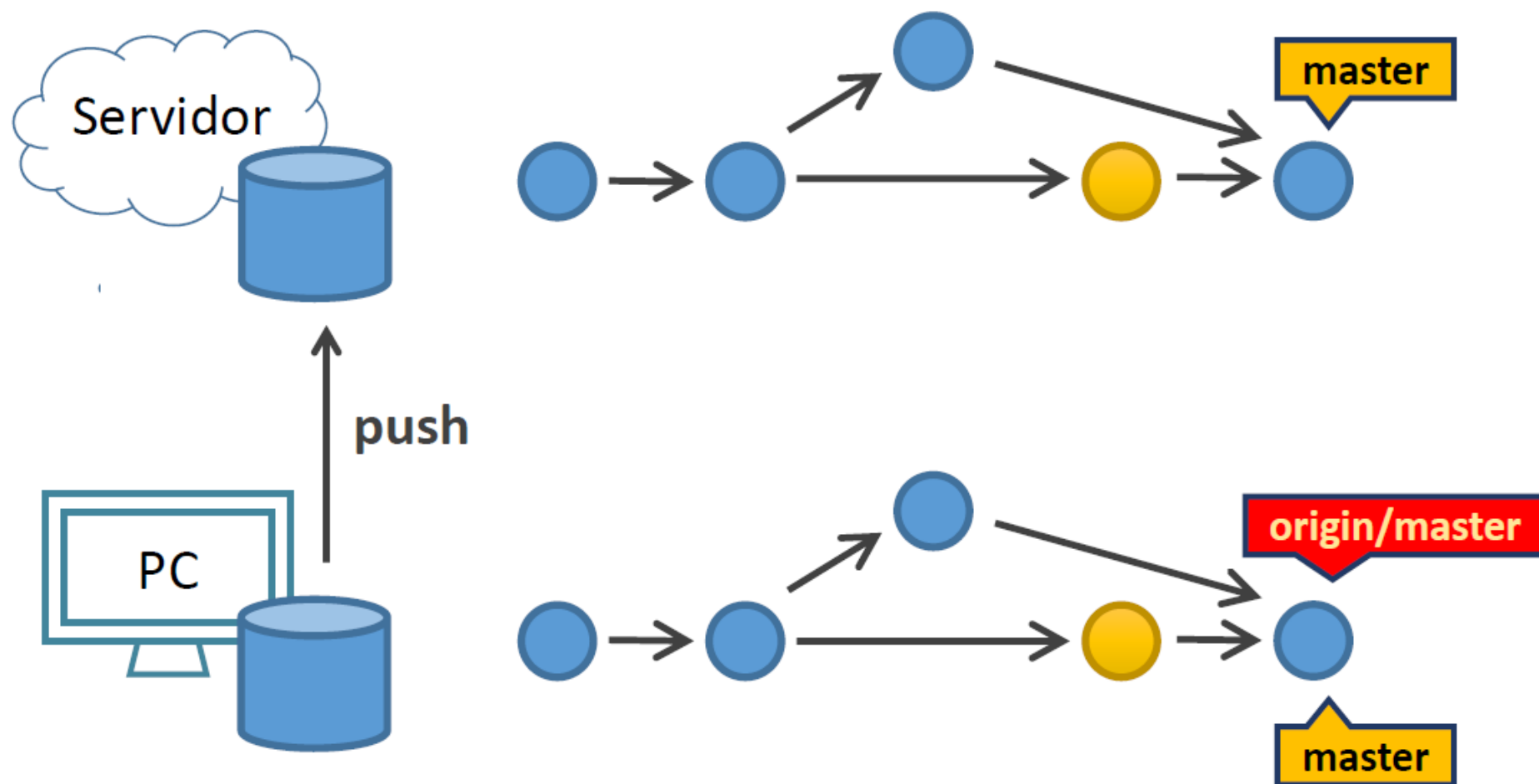


Solução 2: *pull* + *push*



```
$ git pull #Atualiza o repositório local
```

Solução 2: *pull + push*



```
$ git push #Envia para o servidor
```