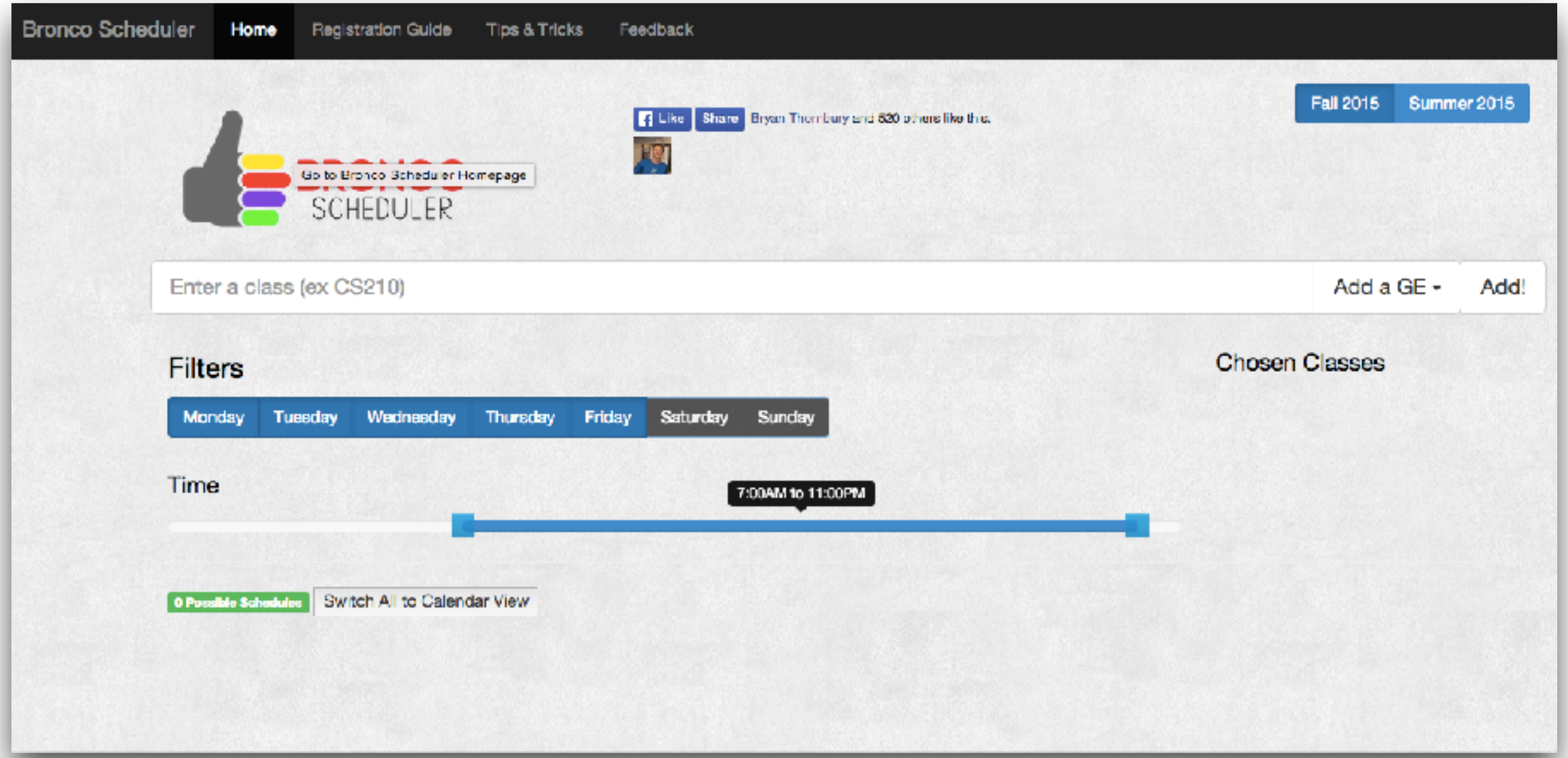
A good idea is worthless without impeccable execution and a commitment to iterate.

Startup Quote!

ZACH KLEIN

CO-FOUNDER, VIMEO



Bronco Scheduler

http://broncoscheduler.com/

Version Control
- A Brief Introduction to Git

CS480 Software Engineering

Yu Sun, Ph.D.
http://yusun.io
yusun@cpp.edu

CAL POLY POMONA

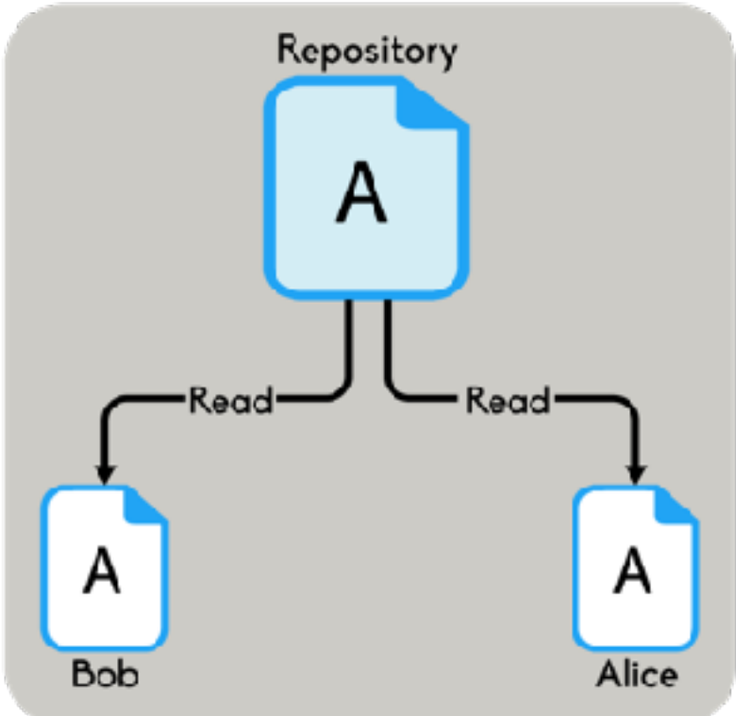
Why Version Control?

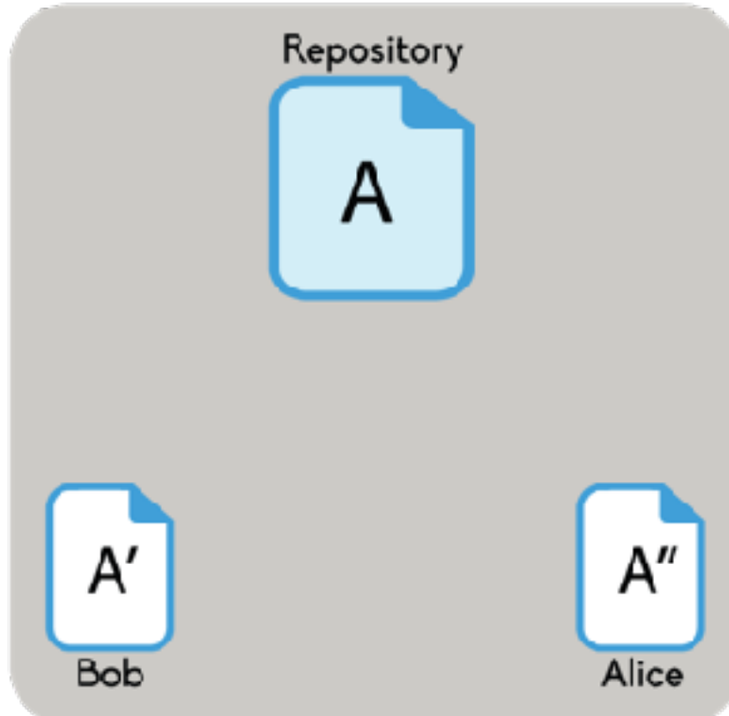
Maintain Multiple Versions

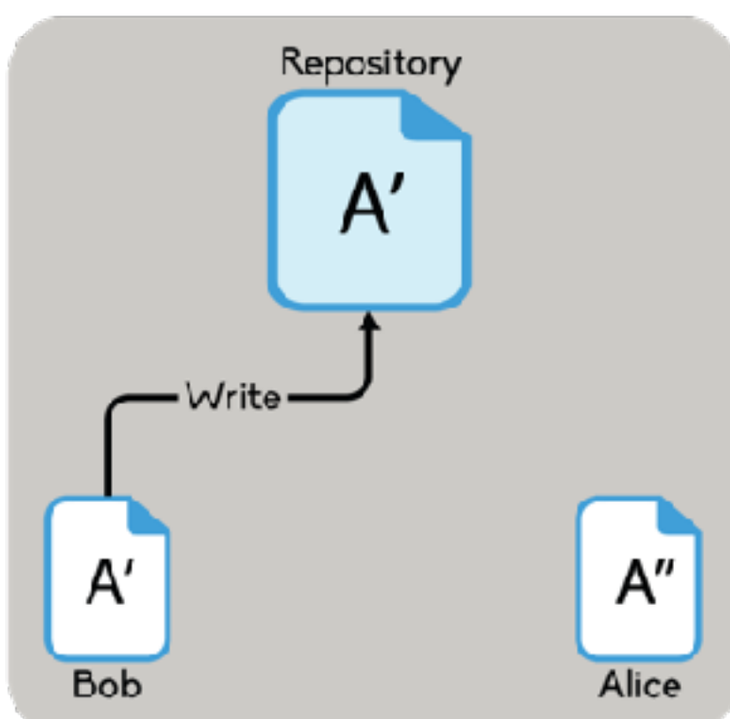
- Safe backup
- Change version

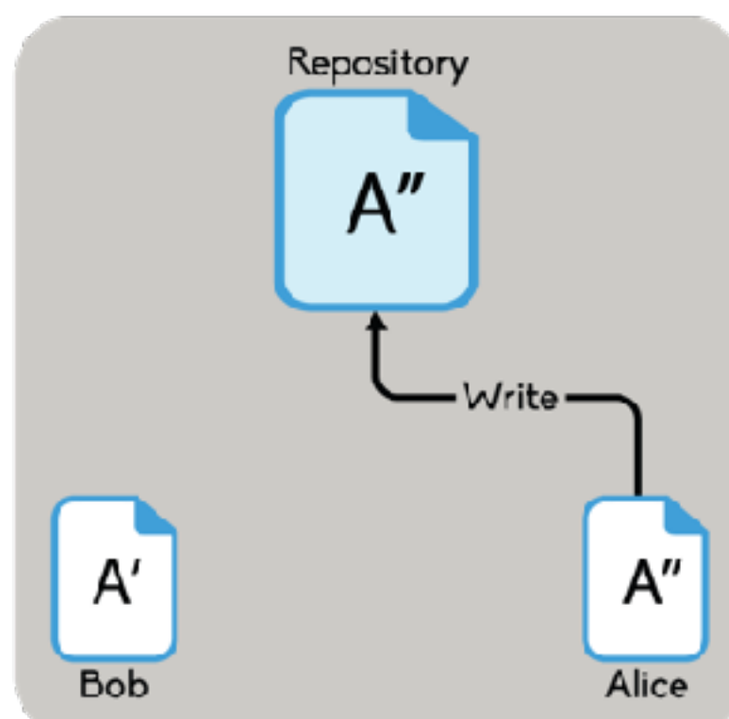
Collaboration

The problem to avoid



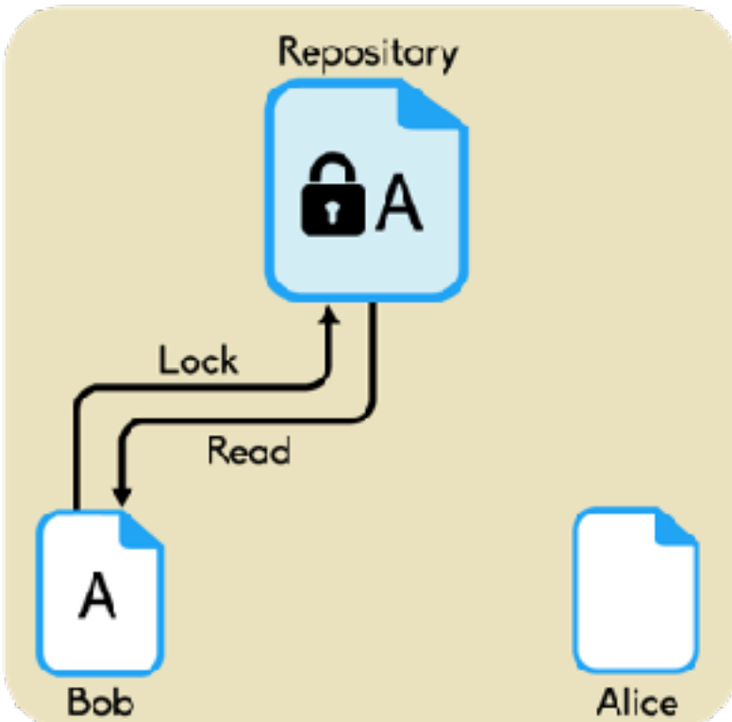


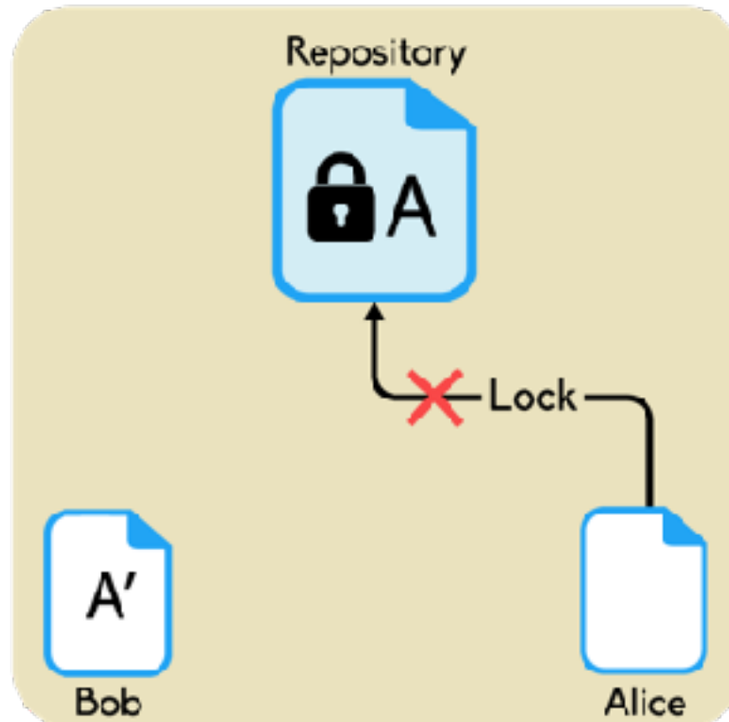


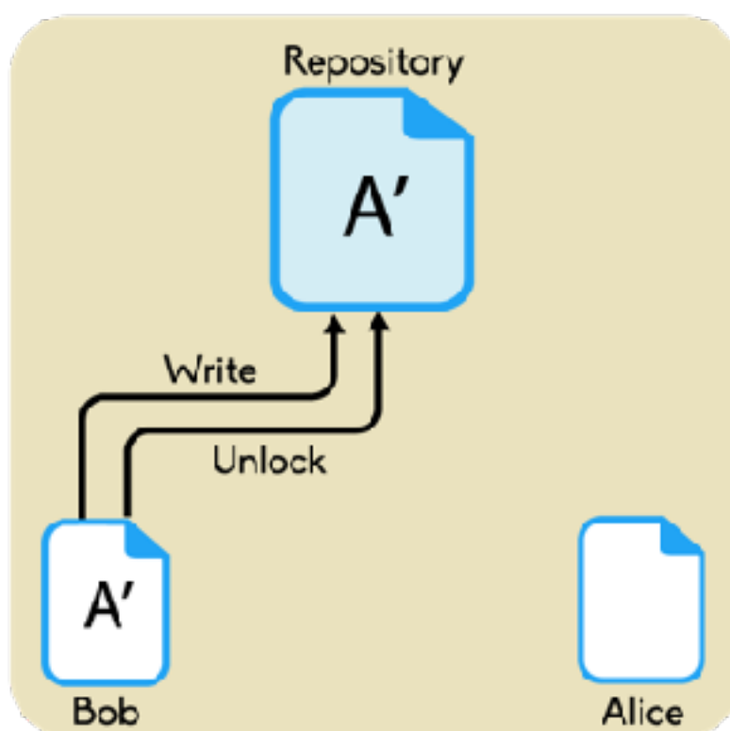


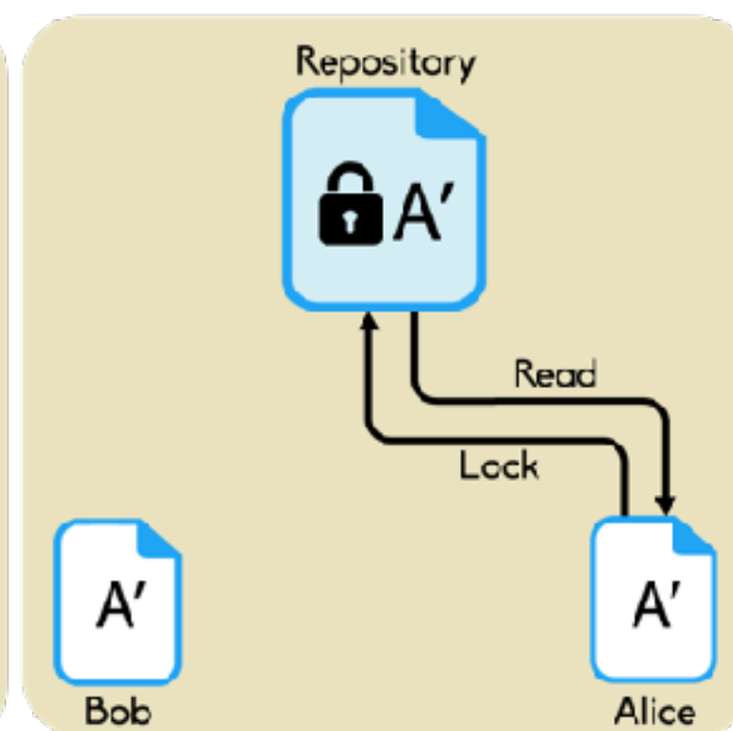
Collaboration

The lock-modify-unlock solution



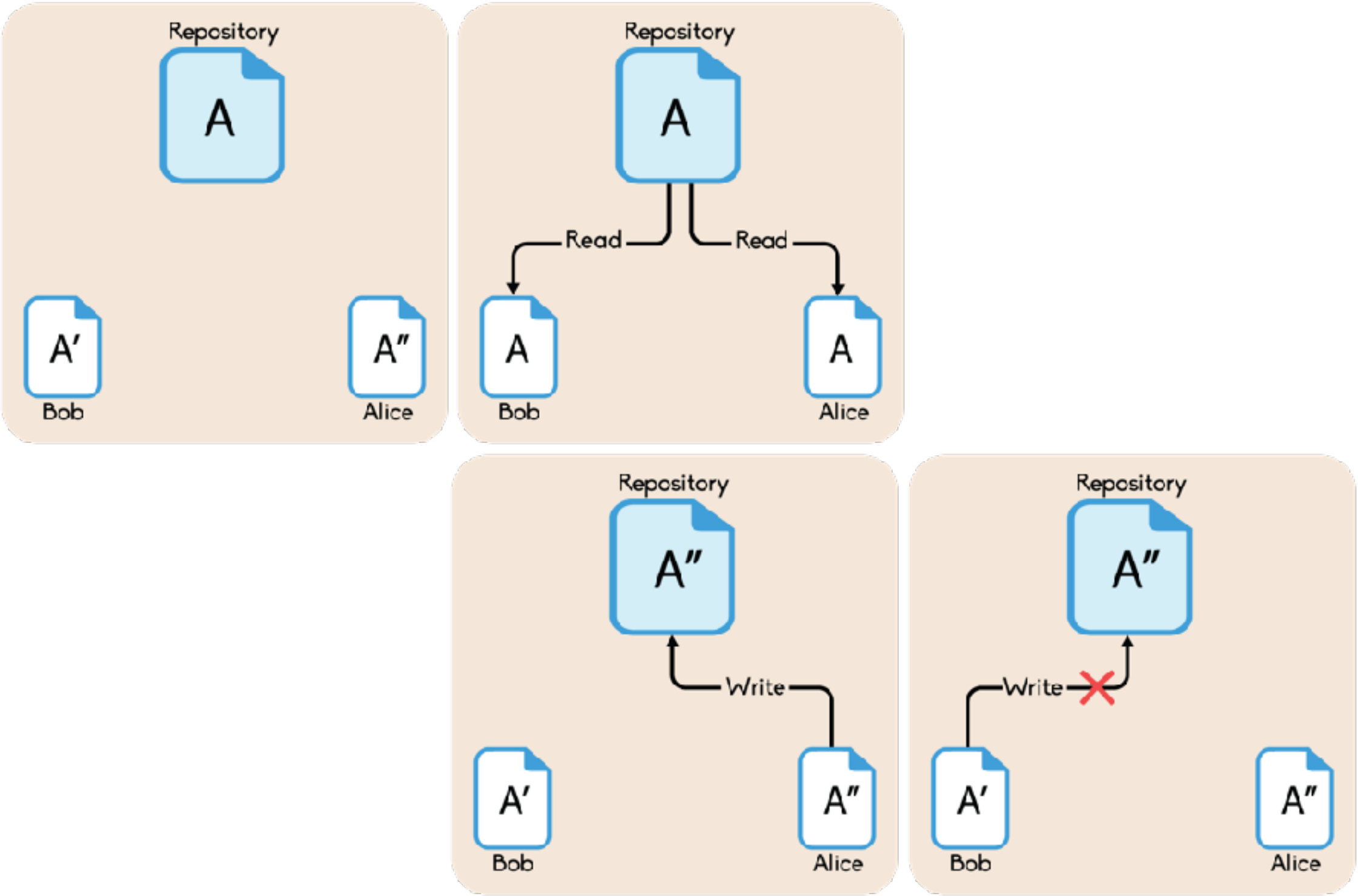






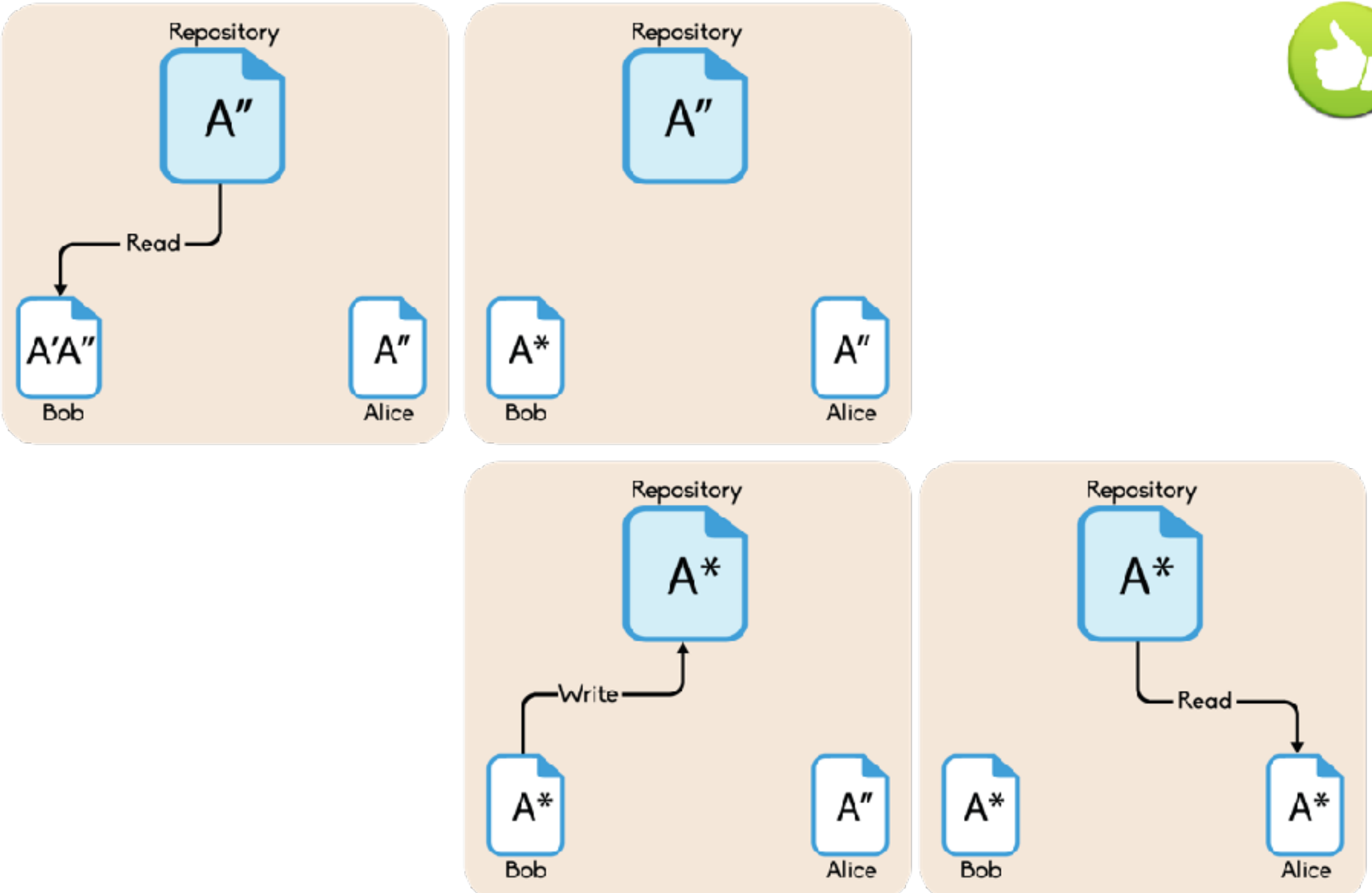
Collaboration

The copy-modify-merge solution

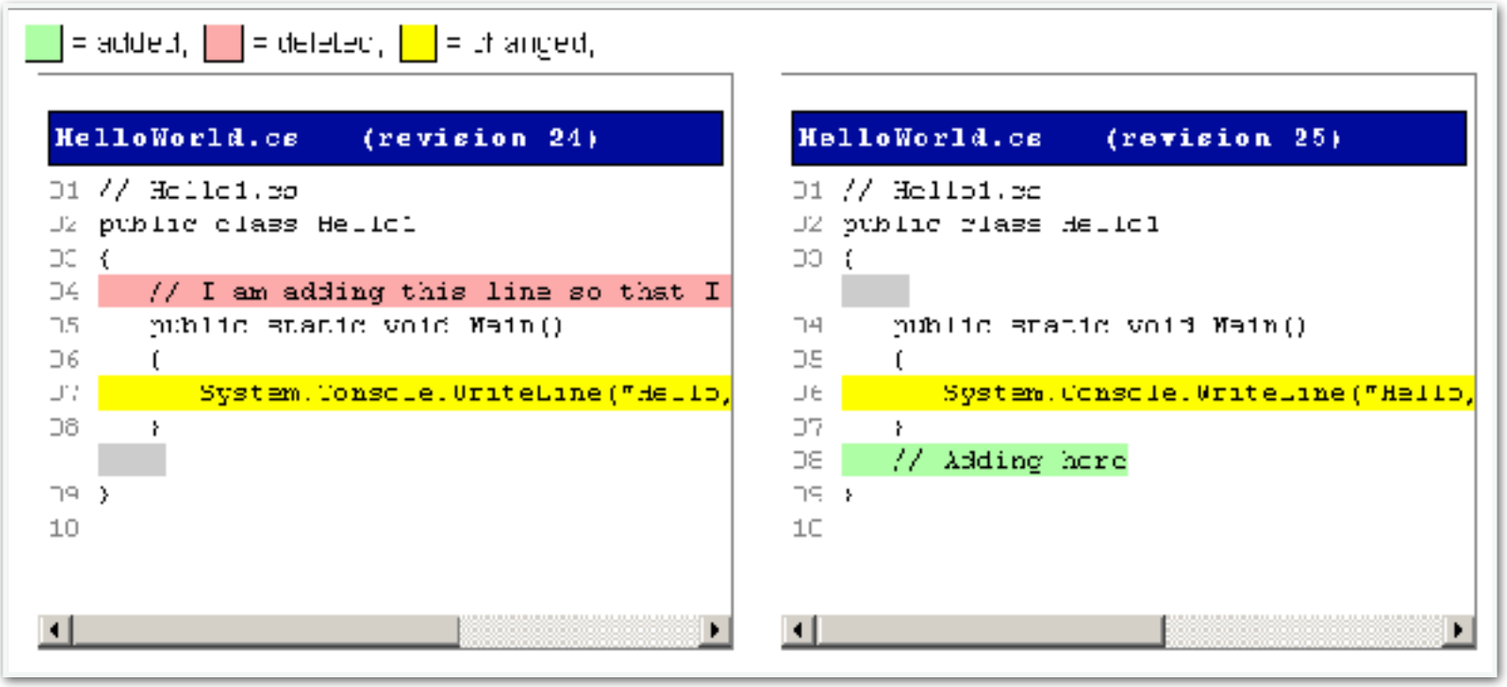


Collaboration

The copy-modify-merge solution

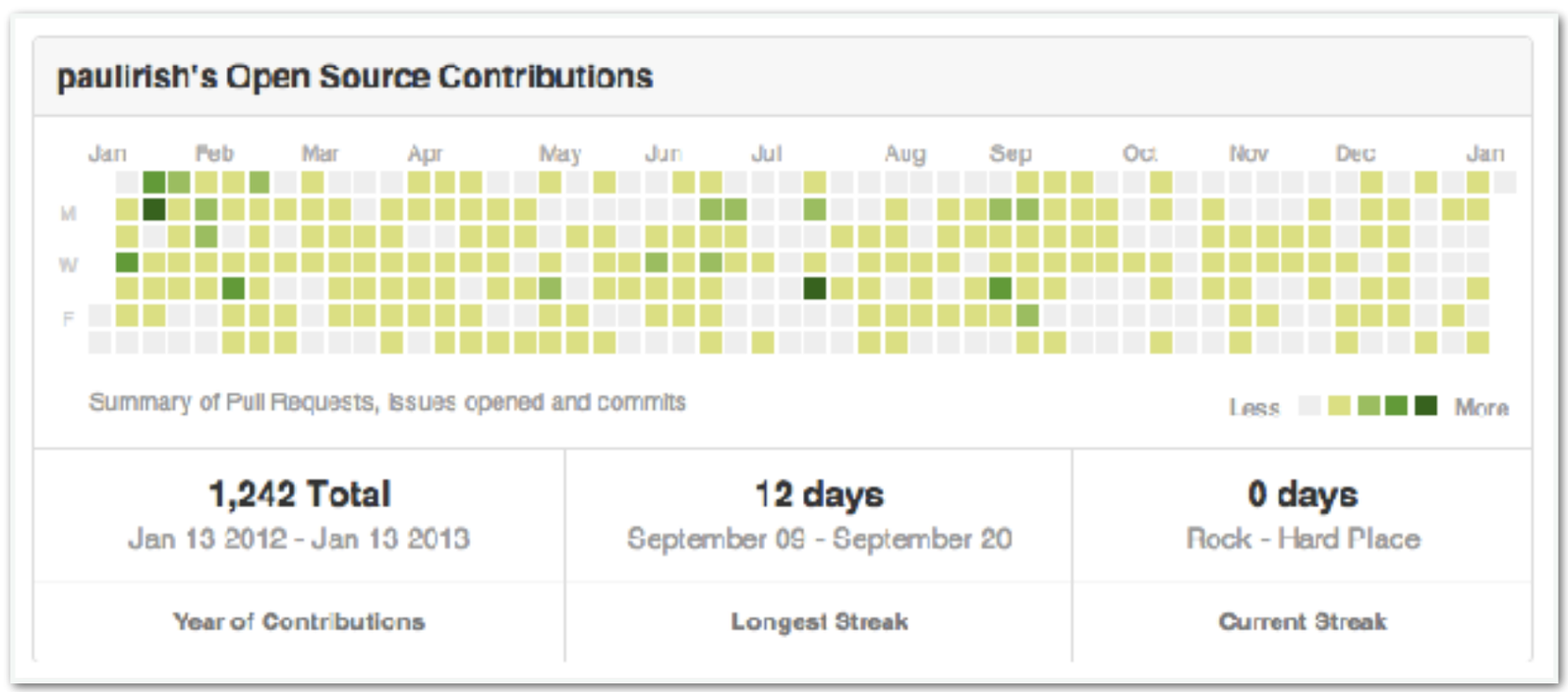


Monitor and Track Progress



Code Difference

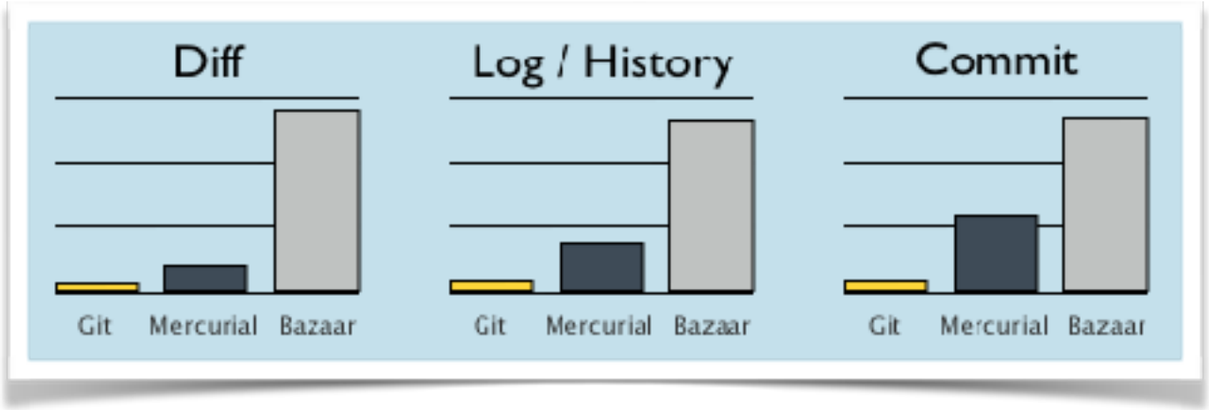
Code Contribution



Why Git?

Why Git?

- Performance
- Github
- Popular



Git History

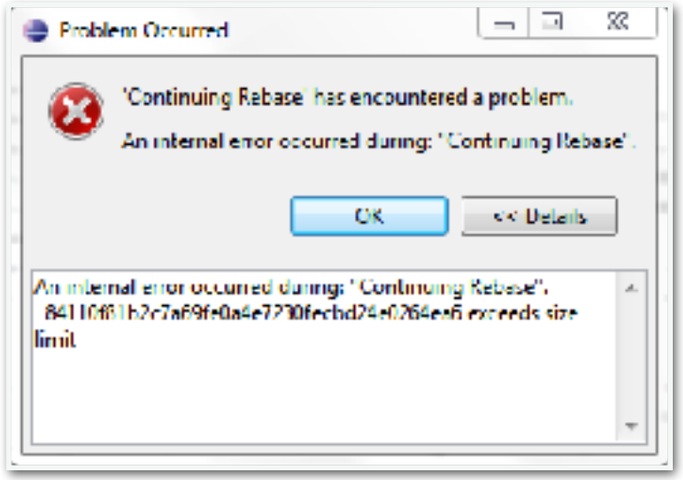
"I'm an egotistical bastard, and I name all my projects after myself.
First Linux, now git."

— Linus Torvalds

Why Command-Line?

Why Git Command-Line?

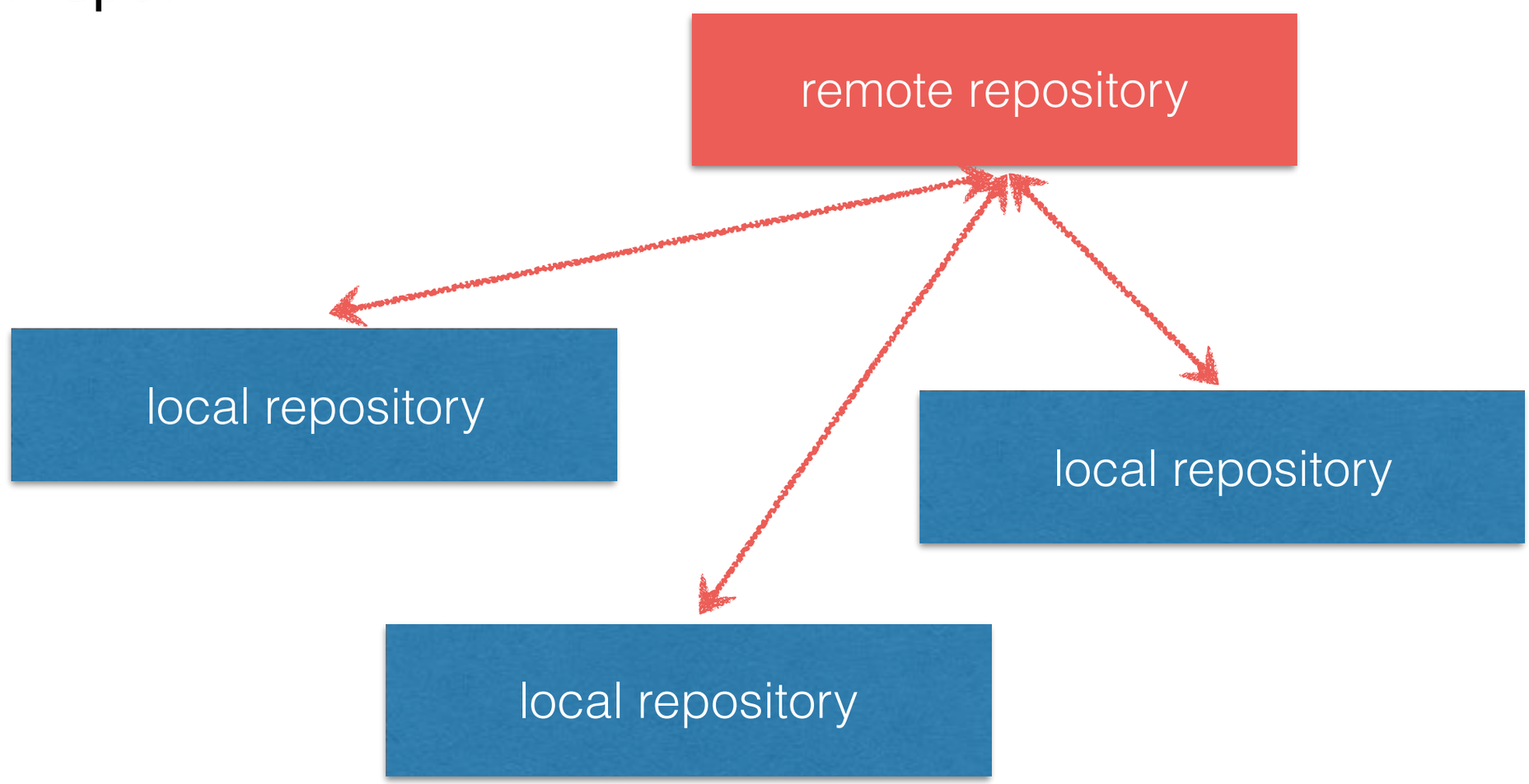
- Graphical clients are based on CLT
- Graphical clients could cause problems
- Integrated with shell scripts
- Graphical clients not always available



Git Exercises

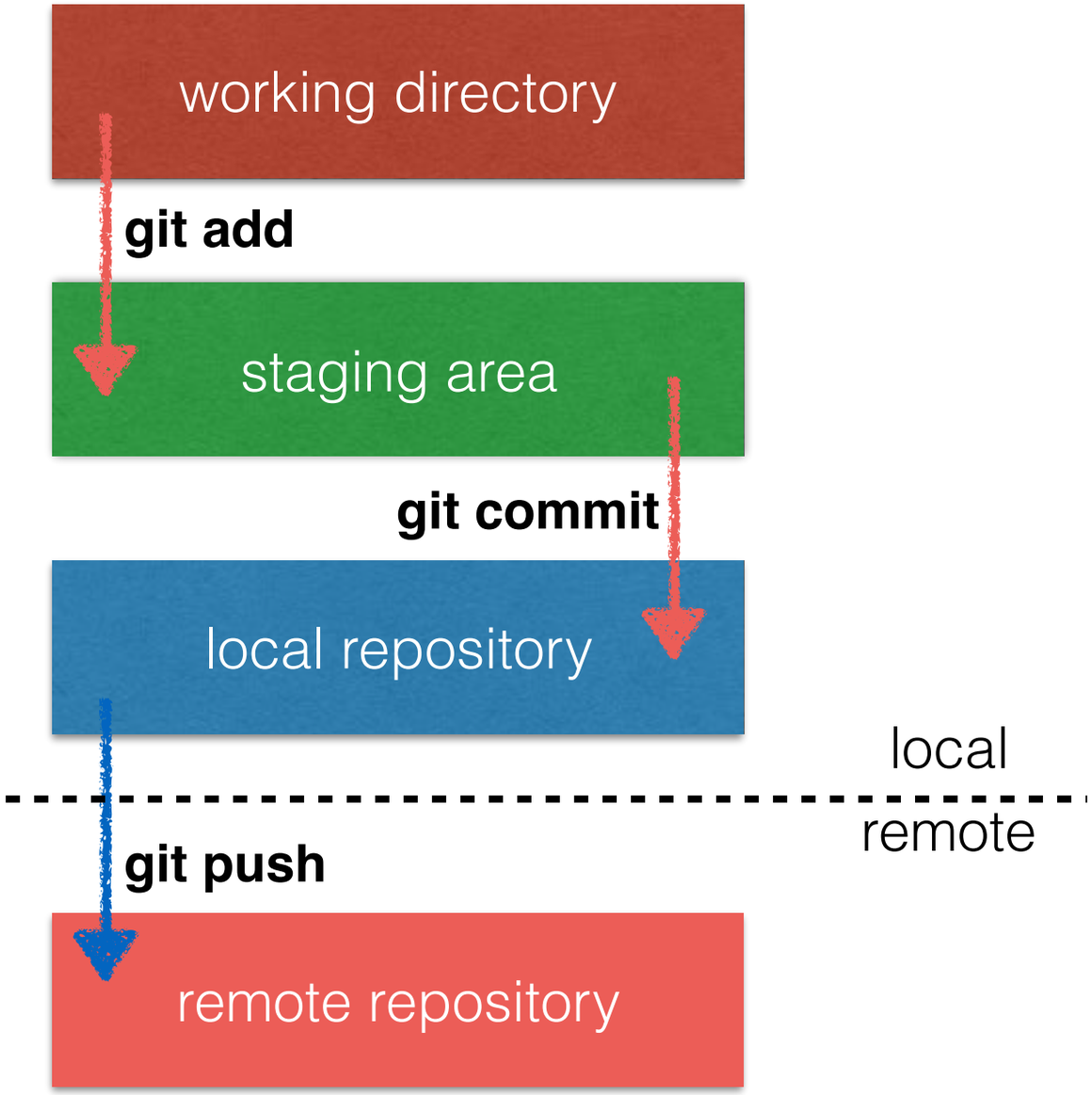
1. Create Git Repositories

- **`git clone <repo>`**
- `git init <repo>`



2. Add/Commit/Push

- `git add <path>`
- `git commit`
- `git push`



3. Check Status

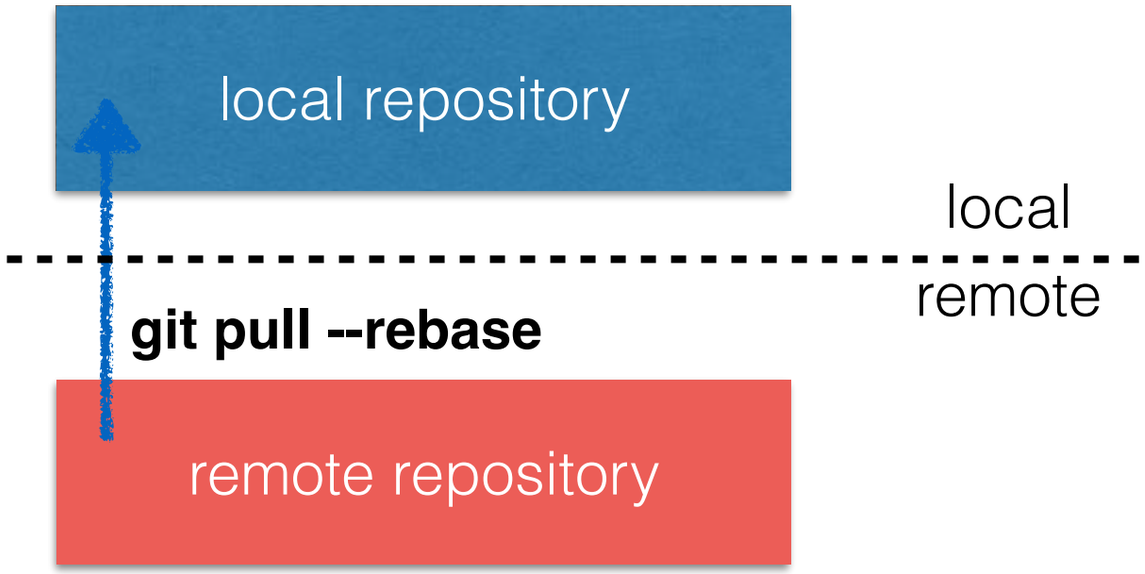
- **`git status`**
- **`git log`**
- **`git branch`**

```
|gh-pages ✗| → git status
On branch gh-pages
Changes to be committed:
  (use "git reset HEAD <file>..." to uns

      new file:   images/boxes.png
      new file:   images/empty.png
      new file:   images/ignored.png
      new file:   images/pallet.png
      new file:   images/push.png
      new file:   images/truck.png
      new file:   images/untracked.png

Changes not staged for commit:
  (use "git add <file>..." to update v                    ed)
  (use "git checkout -- <file>..." to                     working dire

      modified:   index.html
      modified:   init.md

Untracked files:
  (use "git add <file>..." to inclu                       committed)

      fork.md
```

4. Sync Changes & Resolve Conflicts

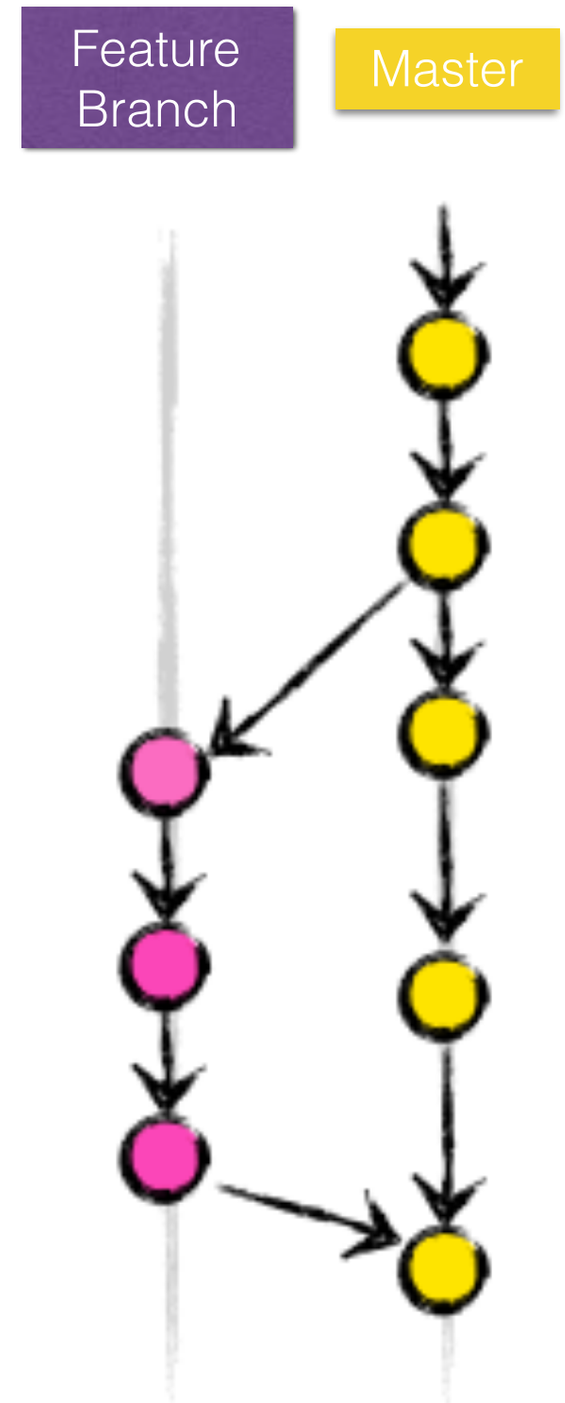
- **`git pull --rebase`** (referred approach)
- `git pull`

- `git status`
- `// clean up conflicts`
- `git add <conflicted file>`
- `git rebase --continue`
- `git push`



5. Branches

- `git checkout -b <branch>`
- `git checkout <branch>`
- `git merge <branch>`

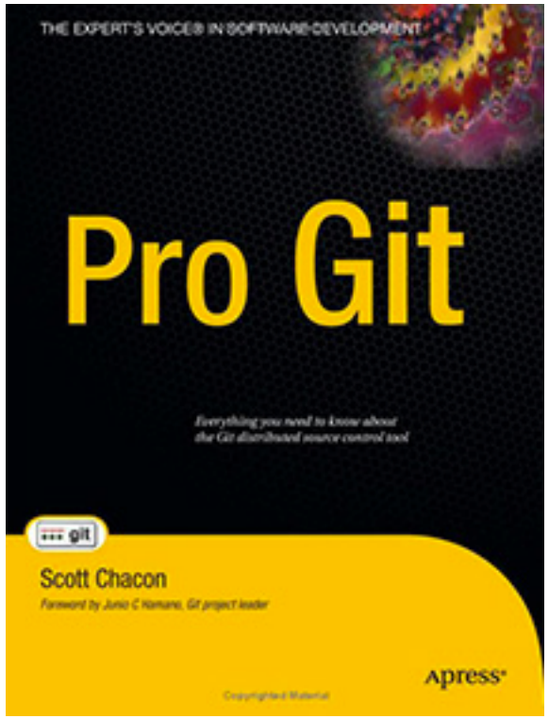


6. Git Undo

- Always backup first
- Google the solution

Git Learning Resources

- http://git-scm.com/
- https://www.youtube.com/user/GitHubGuides
- **Google it!**



Git Basics Overview