

LECTURE 3: VERSION CONTROL SYSTEMS

Computer Systems and Networks

Dr. Pallipuram

(vpallipuramkrishnamani@pacific.edu)

University of the Pacific

Lab Schedule

Today

- **Lab 2 – Version Control**

Next Week

- C programming
- **Lab 3 – Build Tools**



Before Version Control

1. <Report.doc>
2. <Report.doc.bak>
3. <Report-1.doc>
4. Email off to partner...
5. <Report-2.doc>
6. Partner responds with doc (that is missing the changes you just made)
7. <Report-2a.doc>
8. <Report-2a-WITH-REFERENCES.doc>
9. Email off to partner... Partner responds with new doc <Report-3.doc>
10. <Report-3-FINAL.doc>
11. <Report-3-FINAL_v1.doc>
12. Report-3-FINAL_v2.doc>
13. Report-3-FINAL_forsure.doc>
14. Report-3-FINAL_forsure_v1.doc>

Motivation for Version Control

Why would a single programmer (working alone) use version control?

- Backup files
- Roll-back to earlier (working) version
- See changes made between current (broken) code and earlier (working) code
- Experiment with a new feature
 - Try a risky change in a “sandbox”
 - If it works, you can merge it into the regular code.
If it fails, you can throw it away.

Motivation for Version Control

Why would a small group of developers use version control?

- All the reasons a single programmer would, plus...
- Merging different changes made by different developers into the same file
 - Git keeps track of it using the concept of *changesets*
 - What changed? where? Moved where in the file?

Motivation for Version Control

Why would a large group of developers use version control?

Different question: Could you develop the Linux kernel, Adobe Photoshop, Google Chrome, etc... using:

- **A single shared “folder of code”?**
- **Emailing code snippets between developers?**
- **Everyone sits around and shares one keyboard?**

Version Control Basics

What kind of files should I keep in version control?

- Program source code (*obviously*)
- VHDL / Verilog files (from digital design class)
- Matlab scripts (from DSP and Image Processing)
- HTML files
- Server configuration files
 - Imagine you work at Livermore National Labs, and your job is to manage Linux cluster computers with 100,000+ machines (nodes)...
- **Anything that is plain text!**

Version Control Basics

What kind of files should I not keep in version control?

- *These aren't "rules", so much as "guidelines"...*
- **Binary data**
 - How do you *merge* two different binary files together? No general-purpose way to do this
- **Anything auto-generated by the compiler**
 - Object files or executable file
 - Wastes space on useless junk that can be re-created automatically
- **Text editor temp files** (e.g. `main.c~`)

Problem 1: How are these VCSs different? Google 'em!

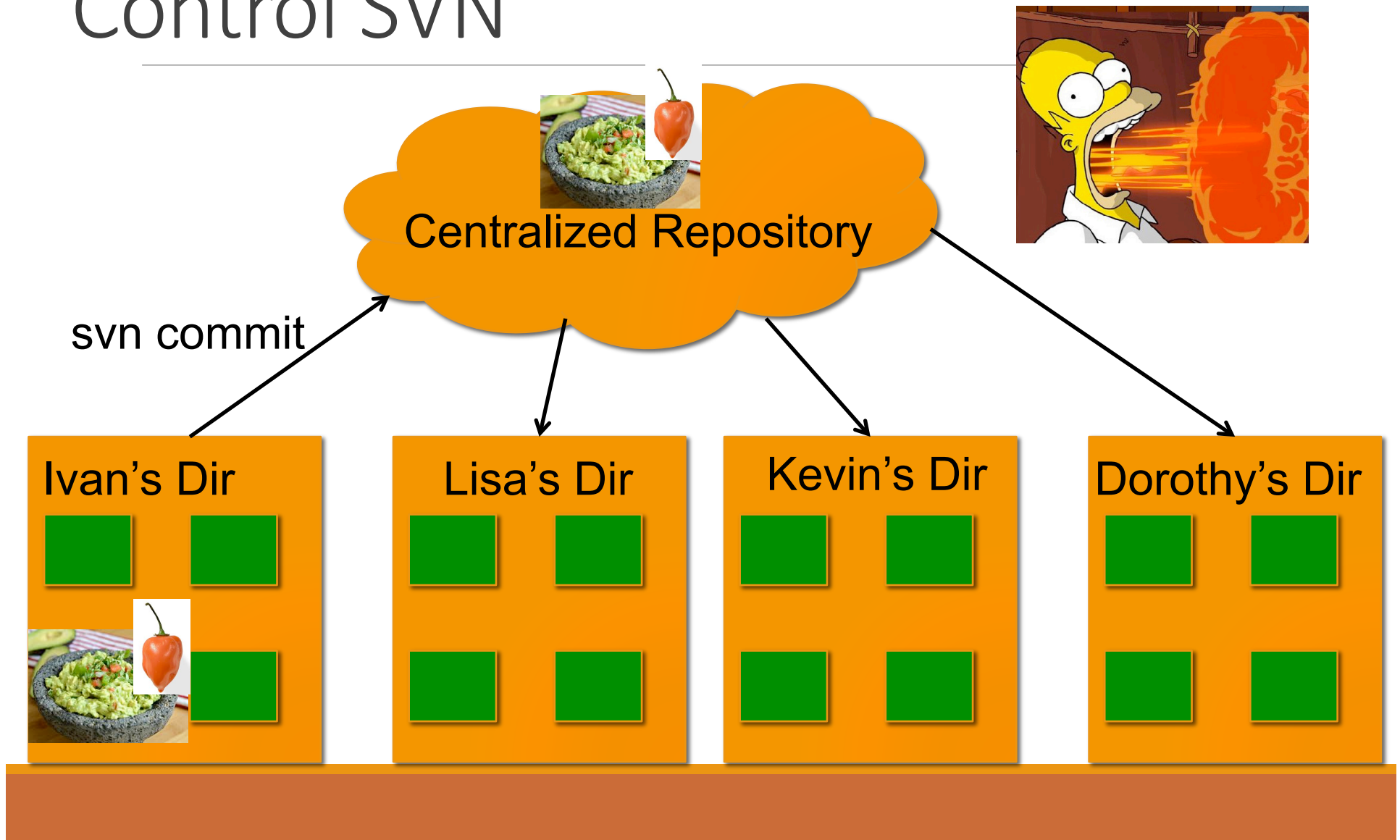
Git

Mercurial

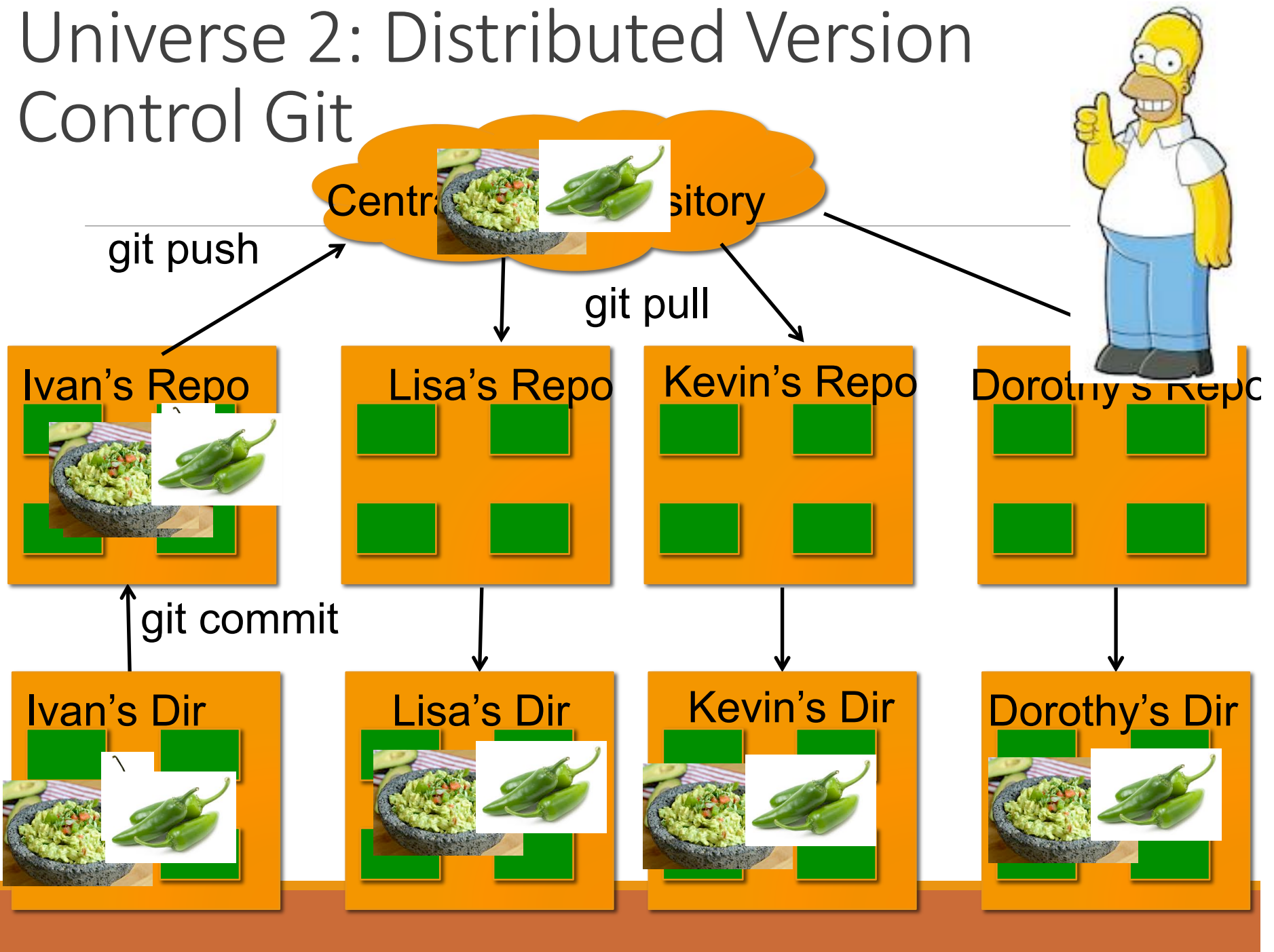
SVN



Universe 1: Centralized Version Control SVN



Universe 2: Distributed Version Control Git



Git Command Flow (usually)

1. `git clone <repository address>`
 - a. #get repo on your desktop
2. `git add <filenames>` #always specify a filename to add
 - a. #add new files and make changes
3. `git commit -m <meaningful commit message>`
 - a. #commit to your repo. Also use `-a` to commit changed files
 - b. Make changes and repeat 3
4. `git push` #All done? Let everyone see

Problem 2: Google Search for Git Cheat Sheet(s)




Version Control in ECPE 170

Version control **required** for this class

- Used to distribute boilerplate code for labs
- Used to turn in assignments when finished

If you only do one check-in at the very end of your project, you've missed the whole point of version control, and turned a valuable tool into an obstacle to completing the assignment

**Check-in code on a
regular basis!**



In case of fire



THIS IS GIT. IT TRACKS COLLABORATIVE WORK
ON PROJECTS THROUGH A BEAUTIFUL
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL
COMMANDS AND TYPE THEM TO SYNC UP.
IF YOU GET ERRORS, SAVE YOUR WORK
ELSEWHERE, DELETE THE PROJECT,
AND DOWNLOAD A FRESH COPY.



[https://dzone.com/
articles/top-20-git-
commands-with-
examples](https://dzone.com/articles/top-20-git-commands-with-examples)



In-class Participation

Collaborate with only one class member to answer in-class participation questions

