



Version Control Systems & Git

Steven Lamerton

steven.lamerton@stfc.ac.uk

Version Control Systems

“Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.” – Pro Git Book

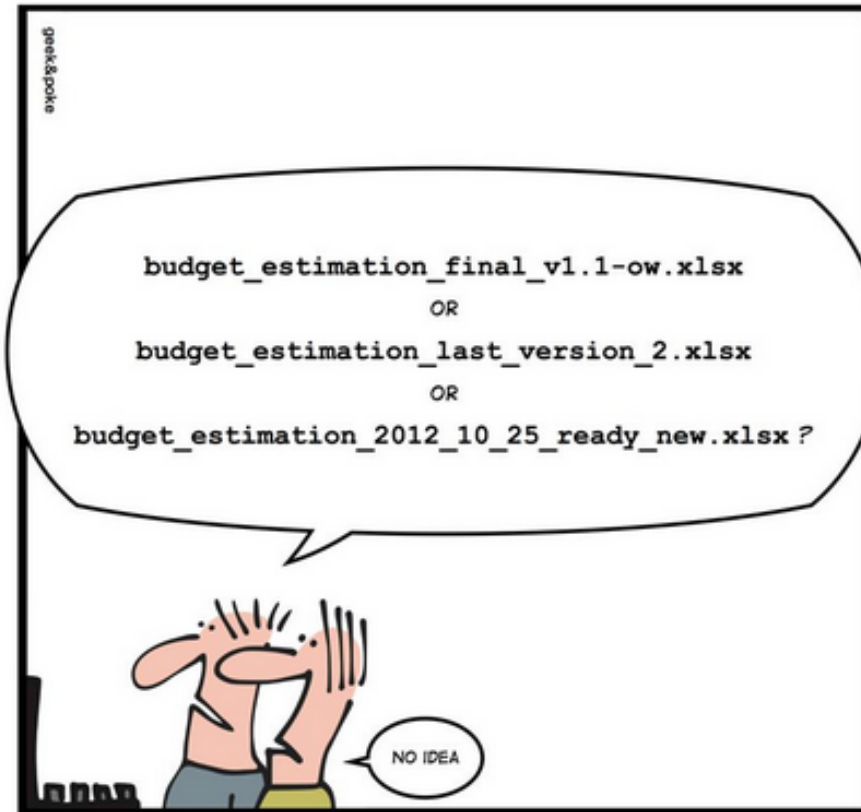


Version Control Systems

- No longer need to copy files around to keep previous versions
 - No more mazes of folders
 - No more copying files to USB drives
- Find out why changes were made, who made them and when they happened
- Makes collaboration easier
 - No fear about overwriting work
 - No copying to network drives / emailing files around
 - Support for better workflows



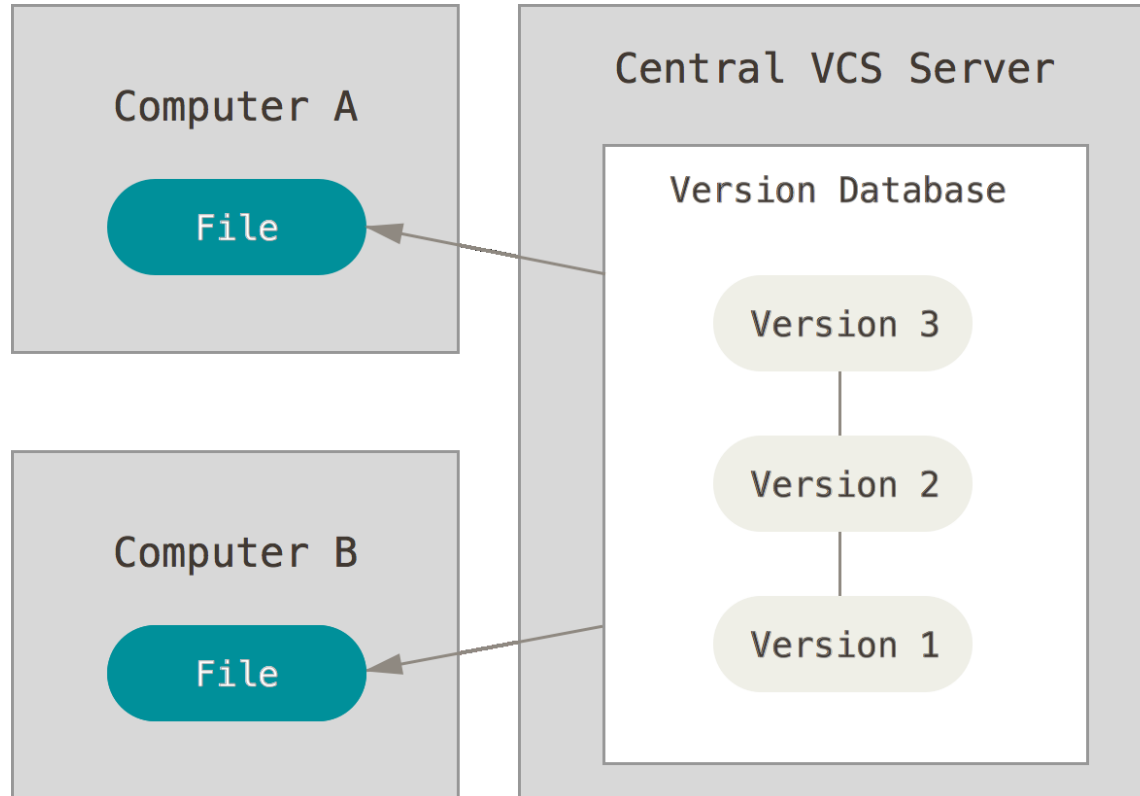
SIMPLY EXPLAINED



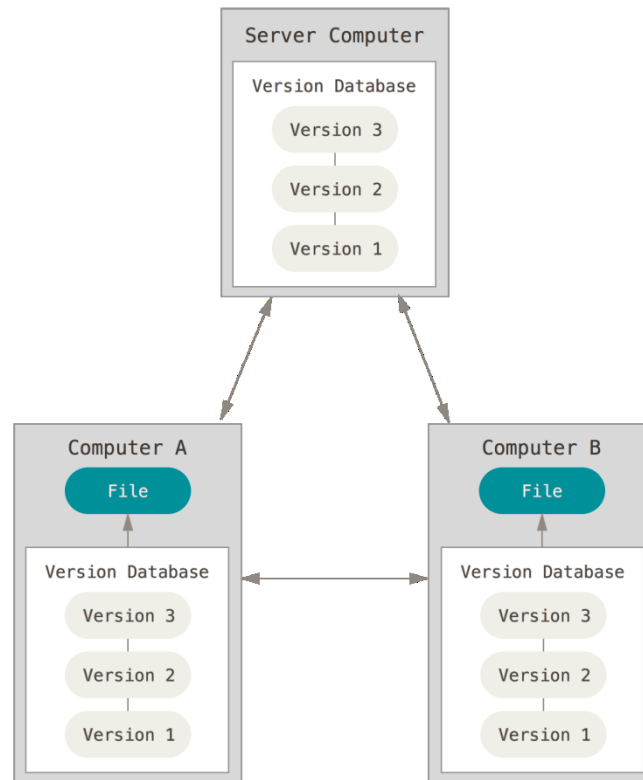
VERSION CONTROL



Centralised Version Control



Distributed Version Control



Version Control Systems

- Centralised
 - CVS
 - SVN
 - Perforce
 - Team Foundation Server
- Distributed:
 - *Git*
 - Mercurial
 - Bazaar
 - BitKeeper
 - Darcs

Git has been chosen for the CCP-WSI code repository



Git

- Started in 2005 by Linus Torvalds, the founder of Linux, to manage the code for the Linux kernel
- Features:
 - Fully distributed
 - Fast
 - Widely supported: hosting, GUIs, tools & documentation
- Widely used by other projects including:
 - Android
 - GCC
 - OpenFOAM

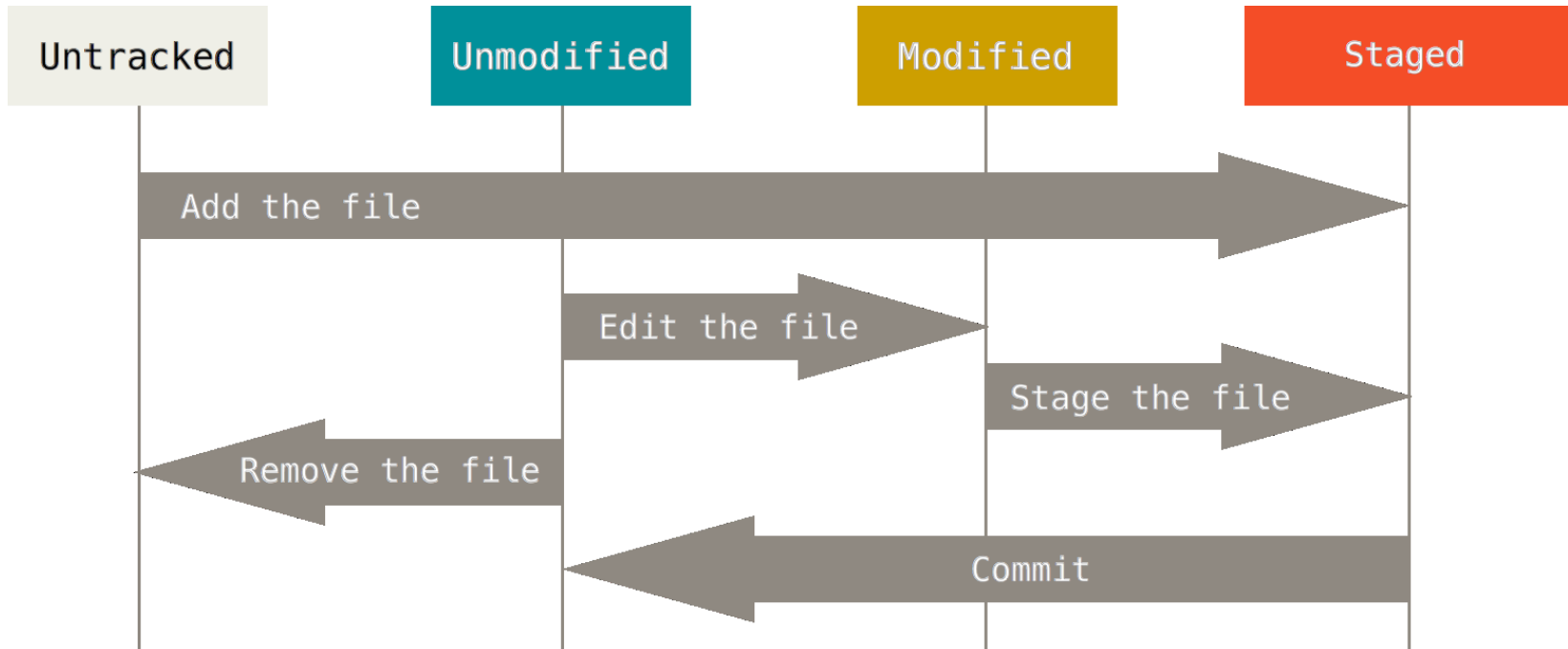


Overview

- Files can be in one of four states
 - Untracked – not managed by Git
 - Unmodified – managed by Git, no changes
 - Modified – managed by Git, has changes since the last version
 - Staged – managed by Git, has changes which are marked to be part of the next commit
- Once you are happy with the staged changes you commit them, adding a descriptive message



Basic Commands

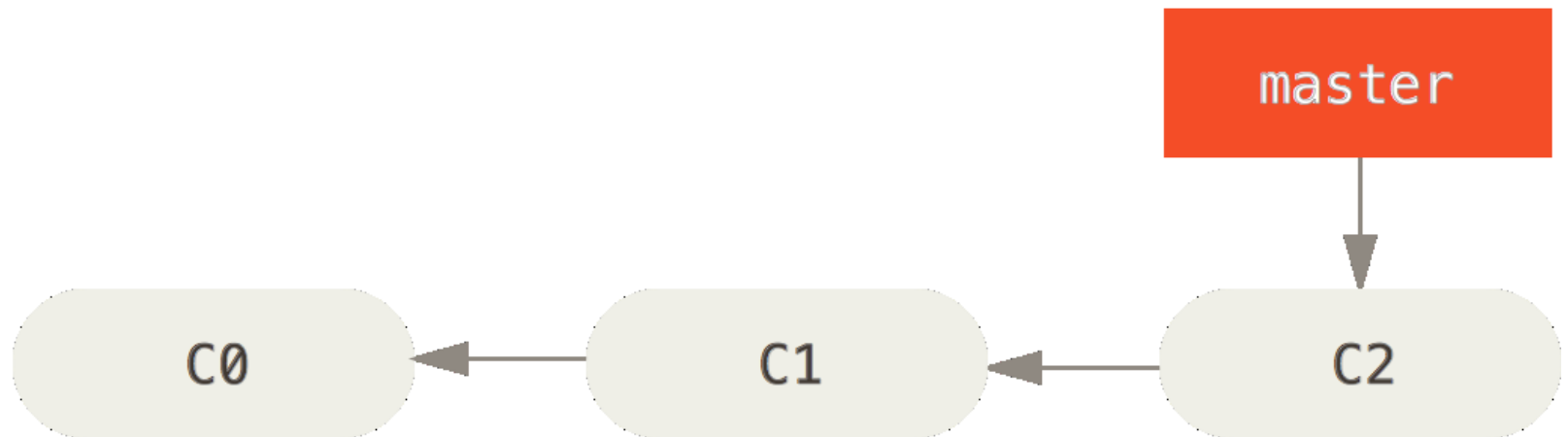


Branching & Merging

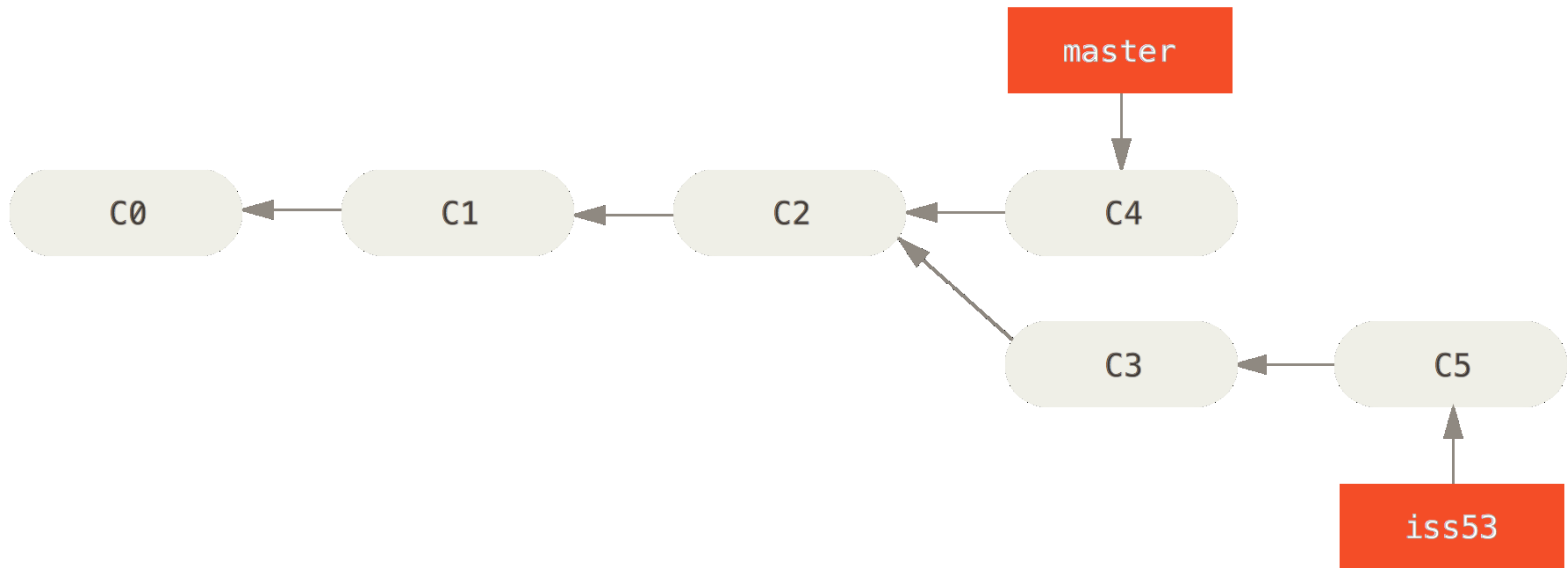
- Branches simplify development
 - Allows splitting from the mainline so as not to break it
 - Useful to separate development of different features and bug fixes
 - Especially useful with many collaborators
- Once work is finished it can be merged back into the mainline again
 - Possibility causing conflicts if others have worked on the same areas of the files



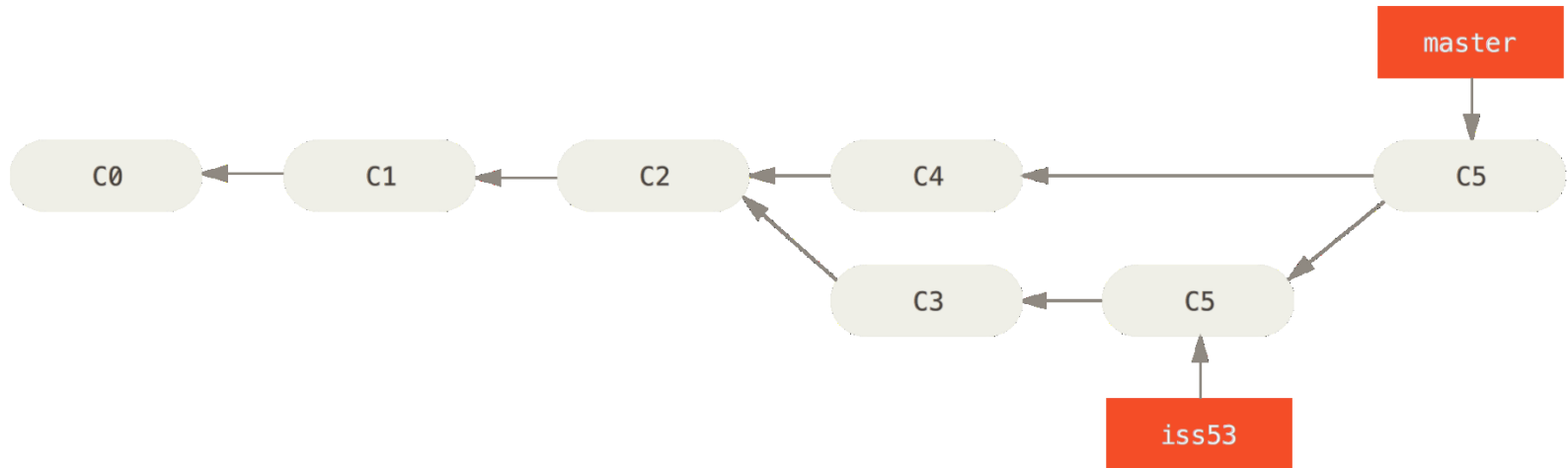
Basic History



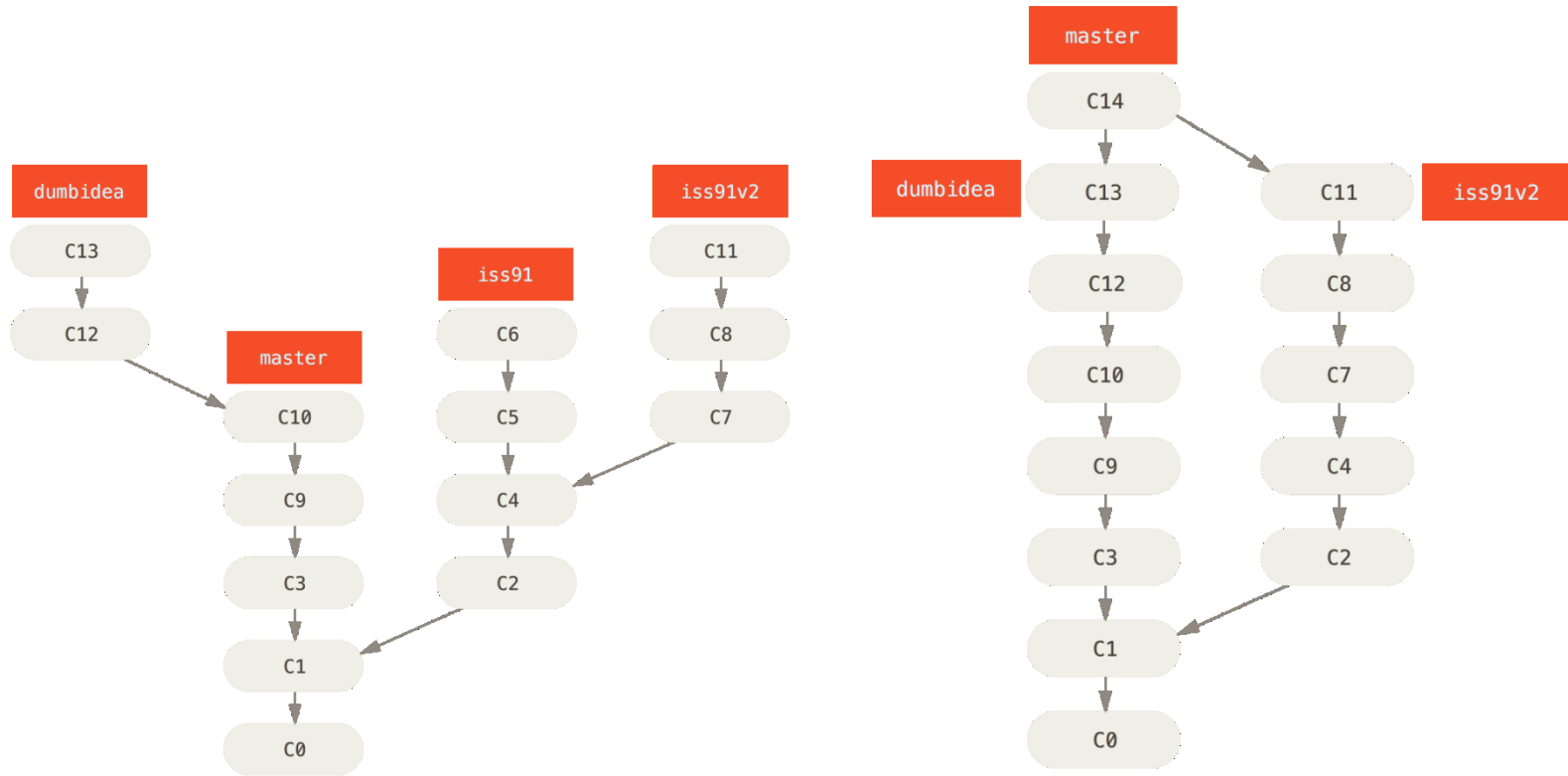
Basic Branch



Basic Merge



Complex History

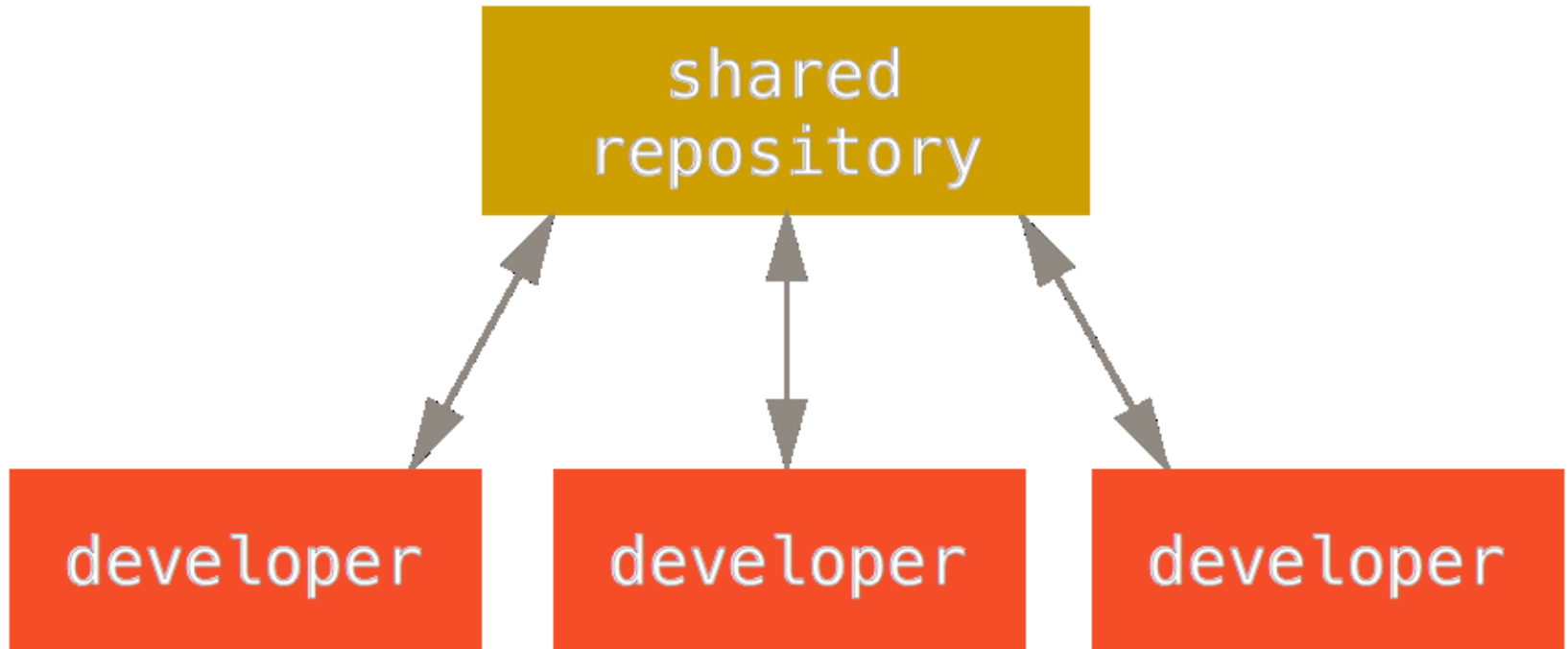


Distributed Working

- Examples so far have been for a local repository, the concepts are exactly the same for working as a group
- Can add other repositories as *remotes*
 - For example the repository on CCPForge
 - Could also be another developers repository
- Code can be pushed and pulled between repositories
 - Essentially branches, which can be merged into the local copy



Distributed Workflow



Don't Panic!

- Lots of new concepts in this presentation
 - Two long practical sessions later today
 - Plenty of time to discuss over coffee / lunch
- By the end of today you should be able to:
 - Get the CCP-WSI repository
 - Make your own changes on a branch
 - Merge that branch into the development branch
 - Push the changes back to the repository



Acknowledgements

- All diagrams and the leading quote are from the Pro Git book and licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 License.
- The Version Control comic is from Geek & Poke and licensed under the Creative Commons Attribution 3.0 License



Pre-Coffee

- Please have Git and OpenFOAM installed
- Git installation
 - `sudo yum install git-all` (Fedora / Scientific Linux)
 - `sudo apt-get install git-all` (Ubuntu / Debian)
- OpenFOAM installation
 - <http://www.openfoam.org/download/ubuntu.php>
- We will come and help if needed
- Register for CCPForge if you are not a member
 - <https://ccpforge.cse.rl.ac.uk>

