Lecture 4 – Tools I

Software Engineering & Projects

Faculty of SET – School of Computer Science

make history.

THE UNIVERSITY of ADELAIDE

Overview

1. **Discussion Recap**
2. **Tools**
3. **Version Control Systems**
    1. Architecture
    2. GitHub
    3. Good practices
4. **IDEs**
5. **Git & GitHub Demo**
    1. basics (pull, add, commit, push)
    2. Branching, merging

THE UNIVERSITY of ADELAIDE

Discussion Recap

***Project brief & group sign up time are on home page!!**

2023 SEP Projects:

Format: <project name & link to brief> - <abbreviation> - <major> - <tutor>

- Shortest Path Algorithm for Material Transportation - PATH - AI/UG/PG - Dileepa Pitawela
- Block Model Compression Algorithm - BLOCK - General/UG/PG - Abdul Kareem & Nauman
- Building a dataset of real-world cyber-attacks with Attack Flow - ATTACK - Cyber/DS/UG/PG - Isuru Mahagani Arachchige
- Security Integration Impact Assessment Tool for Continuous Software Deployment Pipelines - CD - Cyber/UG/PG - Roshan Rajapakse

Group sign-up opening Friday 5:30pm

Communication:

Monday's lecture will be kickoff meeting by tutor, following same client order as Thursday.

Tools II will be on Thursday.

Q: Can I use Jira / Trello / other project management tools than Github Projects?

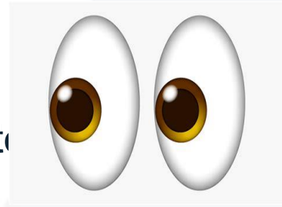
A: Generally, yes, but do let your tutor know & give them access.

Q: People are randomly joining / I want to switch group / group related queries!

A: Don't worry too much as once the projects are out, we will release official project groups which you must migrate over.

Still, good to reach out to random group members and see what's going on.

Q: What's the format for initial report? Any reference needed?



A: We didn't provide a format as we didn't want t
As long as all sections are included it is good.

No need for references, but I am in talk of making minor changes to the initial report structure. Stay tuned.

We are also talking about extending the deadline for initial report. Stay tuned!

THE UNIVERSITY of ADELAIDE

A Tool for Everything

- How many tools do you recognise from the image?
- What are some cool alternative for tools listed here?

THE UNIVERSITY of ADELAIDE

Why use tools if we can just write code for it?

Why are there so many tools??

How should we choose which tool to use???

How could 

help – or could it?

THE UNIVERSITY of ADELAIDE

Version Control Systems (VCS)

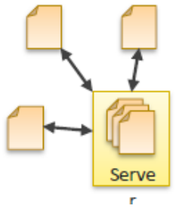
- = Systems that captures file changes
- Captures file content, timestamp, owner of change
- Manages version history & shared access to file
- Supports most file types
  - Diff available for plain text – e.g. code / HTML, not binary

- What are some VCS that you can think of?

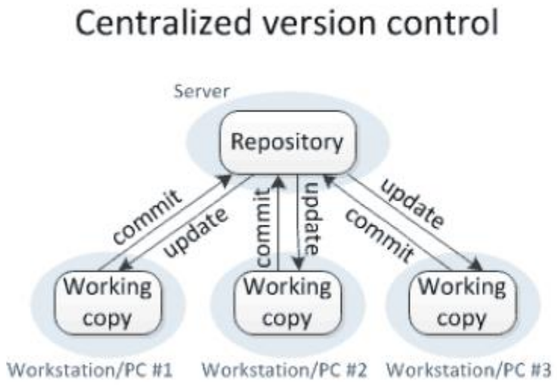
VCS Architecture

Central Systems

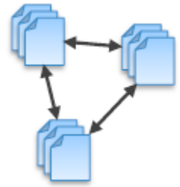
- Client-server architecture
- Complete repository on server
- Users have working copy



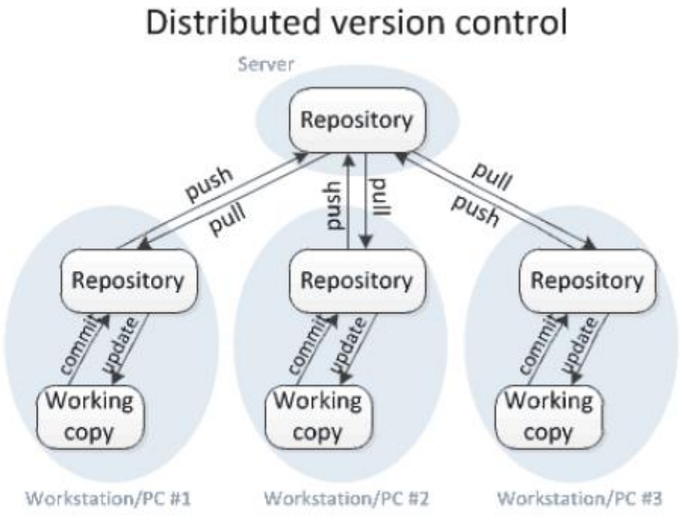
- Concurrent Versions System (CVS)
- **Subversion (SVN)**
- Team Foundation Server



Distributed systems



- Mercurial
- **Git**
- BitKeeper

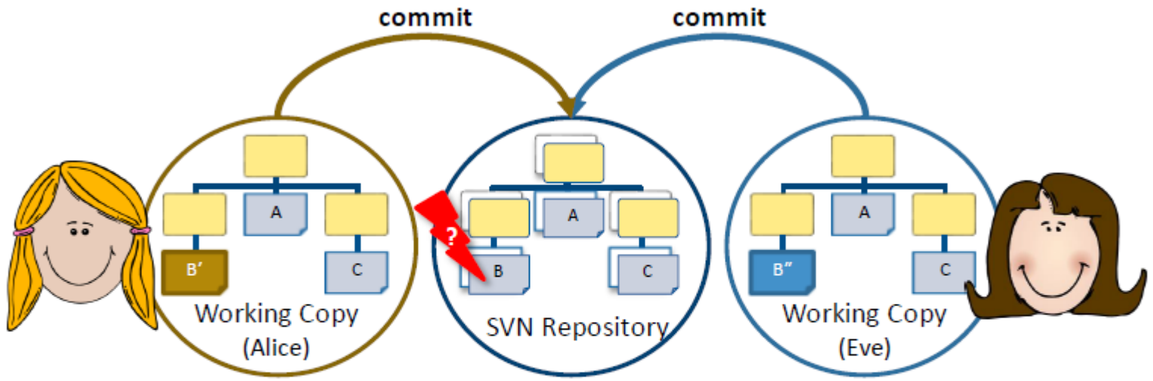


THE UNIVERSITY of ADELAIDE

Version Management

- = the process of keeping track of different versions of software components or configuration items and the systems in which these components are used
- ensuring that changes made by different developers to these versions do not interfere with each other
  - -> more in “configuration management”

Interfering versions

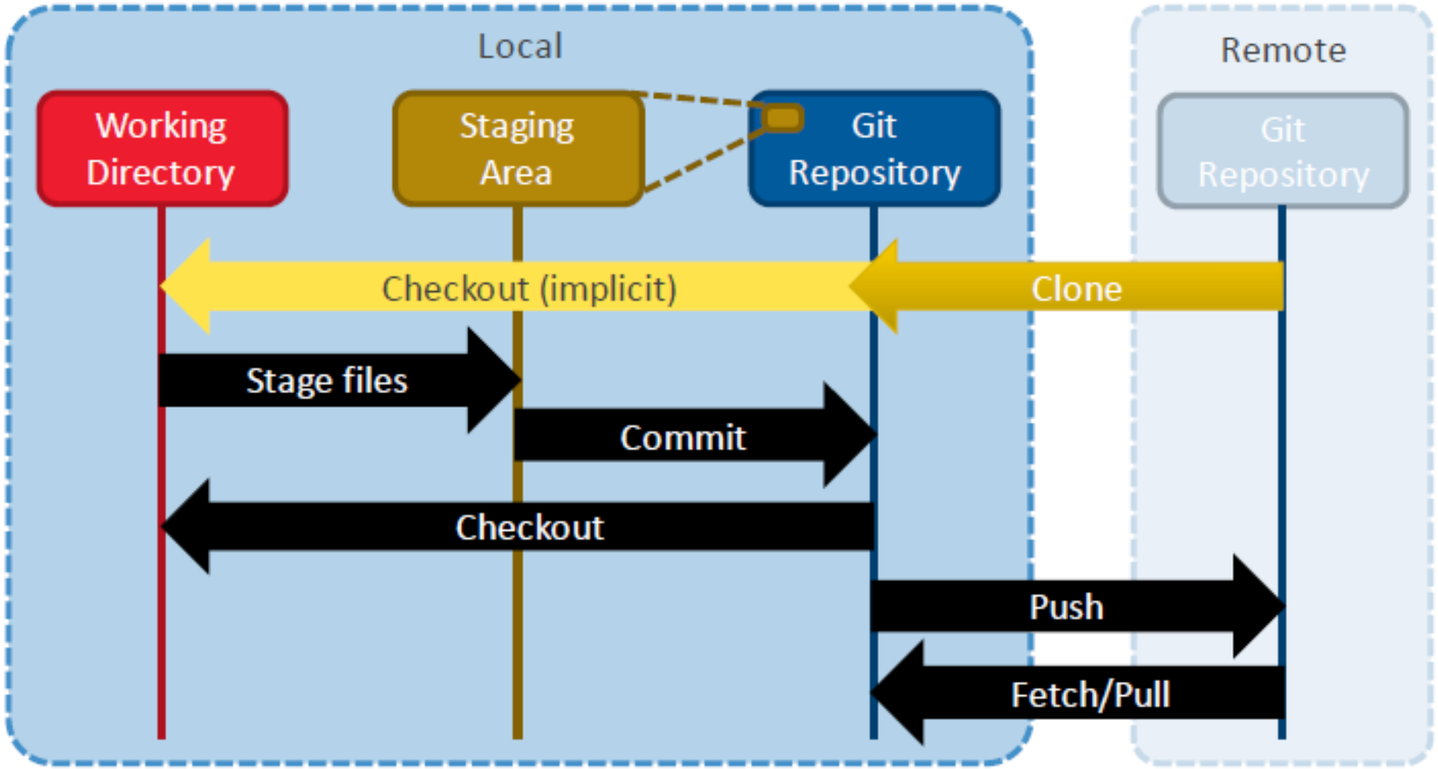


Problem: Multiple developers work on the same file during the same time (actually, they work on copies in their working directories)
Who's changes should be applied to the code in the repo?

Core problem: Centralised repository

Commit = remote action

Whereas Git…



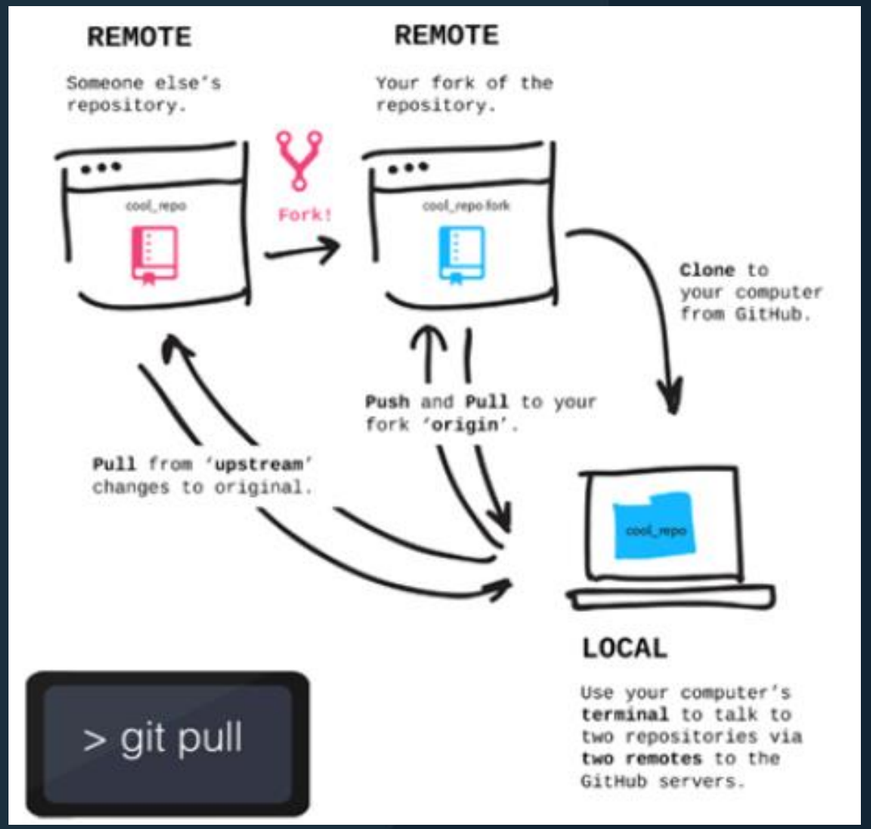
Commit = local action

Push = remote action

-> still need to resolve conflicts

Push / Pull?

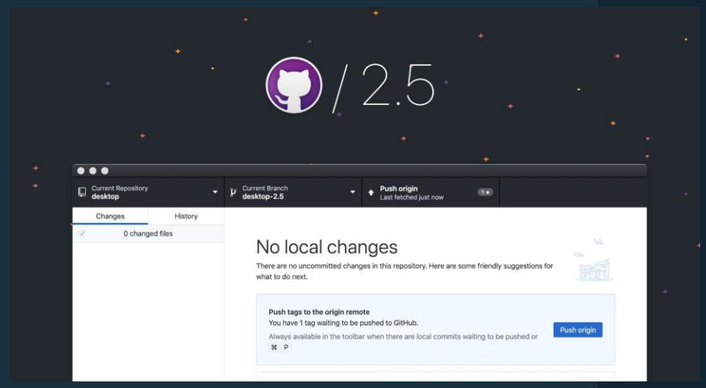
- Git allows developers to synchronize changes between repositories
  - There is no “one true repository”
- Git can “emulate” a central VCS if one central repo is used for synchronization (e.g. hosted on GitHub)



THE UNIVERSITY of ADELAIDE

GitHub

- = Git but more…
  - Web-based access
  - Advanced access control
  - Social features (e.g., user profiles, organizations, stars, followers, gamification)
  - Collaborative features (e.g., bug tracking, features requests, task management, wikis)
- Student Developer Pack: https://education.github.com/pack
- Internal UoASchool of CS GitHub: https://github.cs.adelaide.edu.au/login



- GitHub desktop
  - GUI Git
- GitHub Web
  - Drag-and-drop
- No restriction on which to be used

THE UNIVERSITY of ADELAIDE

Commit messages

- help yourself & other developers to understand…
  - Why is this commit required?
  - How is it implemented?
  - What implications does the change have?

Three important parts:

- Title/summary
- Detailed Description
- Ticket/bug/issue ID (if related to a ticket/bug/issue)

Example:

```
Fixed bug where user can't signup.

Users were unable to register if they hadn't
visited the plans and pricing page because we
expected that tracking information to exist for
the logs we create after a user signs up. I
fixed this by adding a check to the logger to
ensure that if that information was not
available we weren't trying to write it.

[Fixes #2873942]
```

Me writing Computer System svn submission messages – if you know, you know…

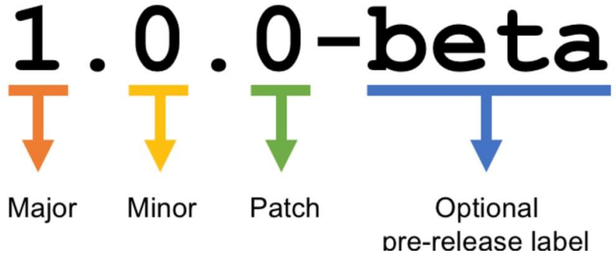
Additional info:
https://www.freecodecamp.org/news/how-to-write-better-git-commit-messages/
https://who-t.blogspot.com/2009/12/on-commit-messages.html

THE UNIVERSITY of ADELAIDE

Version Identification

- systematically and unambiguously talk about a version
- Should have version numbering + change log

Version numbering + semantic versioning:

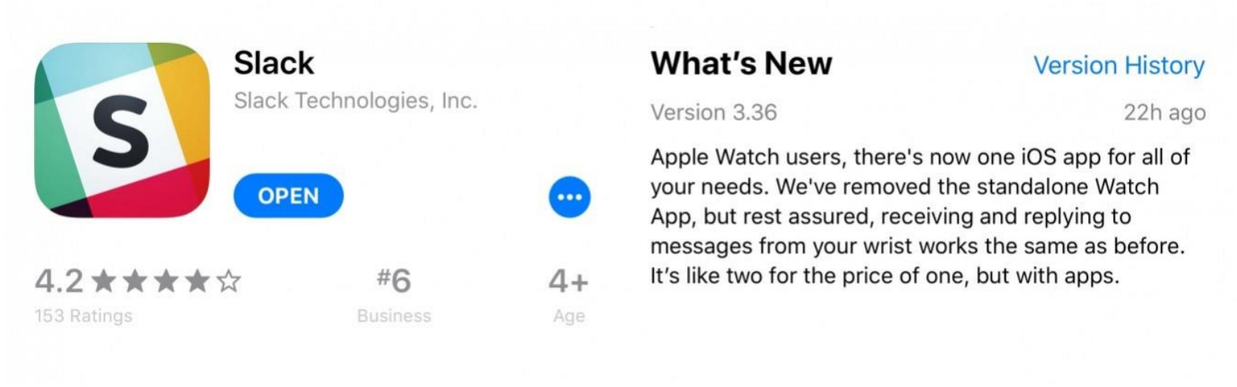


- MAJOR = when you make incompatible API changes
- MINOR = when you add functionality in a backwards compatible manner
- PATCH = when you make backwards compatible bug fixes

THE UNIVERSITY of ADELAIDE

Changelog

- a file which contains a curated, chronologically ordered list of notable changes for each version of a project.
  - -> make it easier for users and contributors to see precisely what notable changes have been made between each release (or version) of the project.
  - human beings care about what's in the software!



THE UNIVERSITY of ADELAIDE

IDEs

= Integrated development environment

- Source editor + build environment + tester + refactor helper + debugging + tools + ……

Visual Studio Code

Everything IDE

Atom

Everything IDE

PyCharm

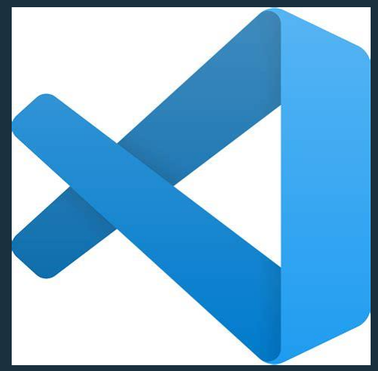
Python IDE

THE UNIVERSITY of ADELAIDE

Takeaway

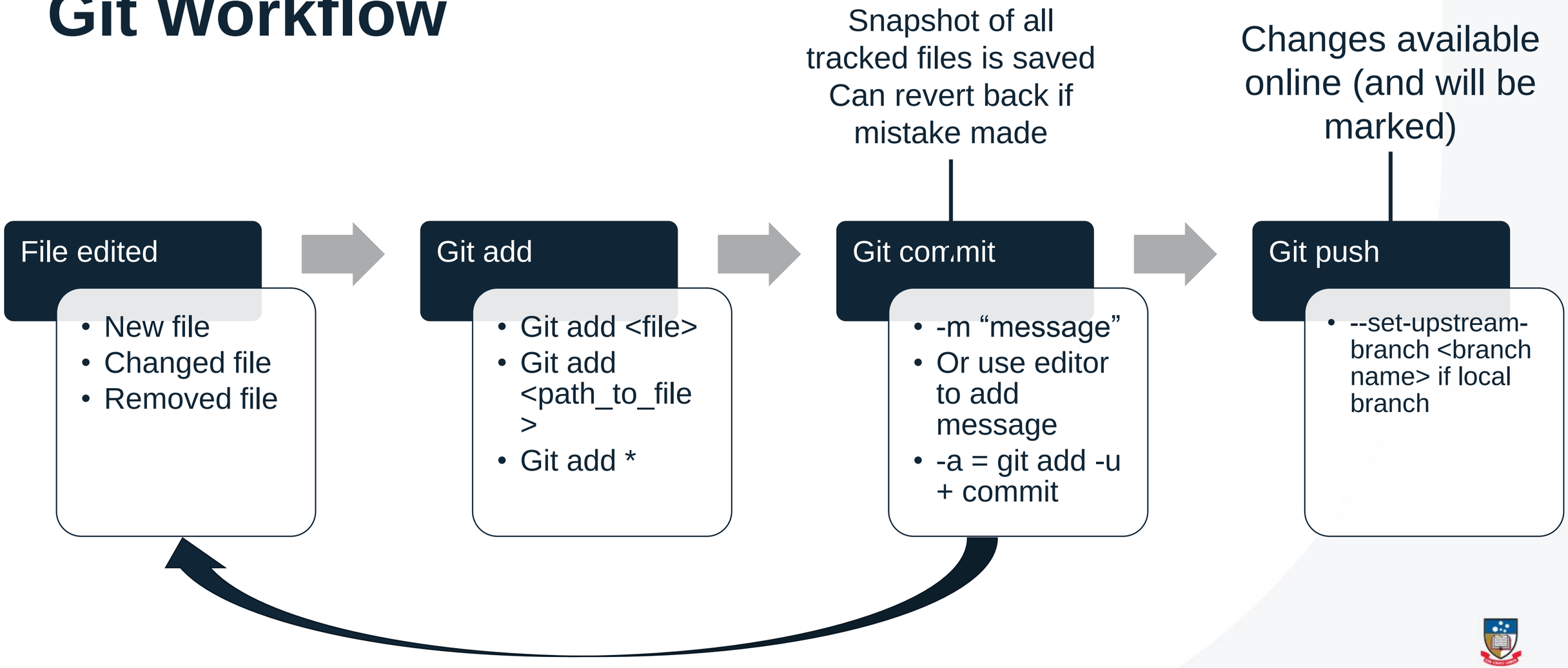
1. **Tools help, but only to some extend**
2. **Git is great for version control**
    1. Pro git: https://git-scm.com/book/en/v2
3. **IDEs saves set-up time**

My favourite:



THE UNIVERSITY of ADELAIDE

Git Workflow



THE UNIVERSITY of ADELAIDE