SE2250 – Software Construction

Lecture #3

Software Engineering Tools

January 21, 2019

Outline

- Change Management
- Tools for software construction

Change management

- The **change management** process is the process of requesting, determining attainability, planning, implementing, and evaluating of changes to a system.

Change management

- Software change is inevitable
  - New requirements emerge
  - The business environment changes
  - Errors must be repaired
  - New computers and equipment are added to the system
  - The performance or reliability of the system may have to be improved

Change management

- The majority of the software budget is devoted to changing and evolving existing software rather than developing new software.

Software tools

Some examples (design tools not included):

- Code editor
- Complier
- Linker
- Source Control System (Revision Control System, Configuration Management Tools)
- Source Code Merge Tools
- Continuous integration tools

Software tools

More tools:

- Issue Management System
- Collaboration tools
- Collaborative code review tools
- Unit test libraries
- Automated testing
- Debuggers
- Integrated Development Environments
- …

Software tools

- Code editor creates source code
- Complier translate source code from a high-level programming language to a lower level language (e.g., assembly language, object code, or machine code) to create an executable program
  - Text files are code (C; C++; Java etc.)
  - Binary files contain machine code for a specific CPU(s)
  - E.g. Gnu CC; MSVC; Clang
- Linker takes files produced by a compiler and creates a one binary file (executable)

Software tools

- <u>Source Control System (Revision Control System/Management, Configuration Management)</u>
  - Manages source code files and projects
  - Keeps track of the history of changes in files
  - Usually keeps track of user comments for each change
  - Interface may be dialog based or command line
  - Git, Subversion and Mercurial are widely used

Software tools

- Centralized Source Control Management
  - Files are stored on a central server
  - Typical workflow:
    - Copy files to local machine ("*check out*")
    - Modify files
    - Copy changed files back to server ("*check in*")
  - Others must be able to "*check in*" their changes
  - Need to periodically update local copy
  - Good for situations which need central control

Software tools

- Centralized Source Control Management - problems
  - Merging multiple changes to the same file
  - Access control: Who can modify what files? Need to trust the person. Not the changes.
  - Access to the repository: Depends on being connected to the server
  - Resolving merge conflicts
  - Complex merges with many files and many changes are difficult

Software tools

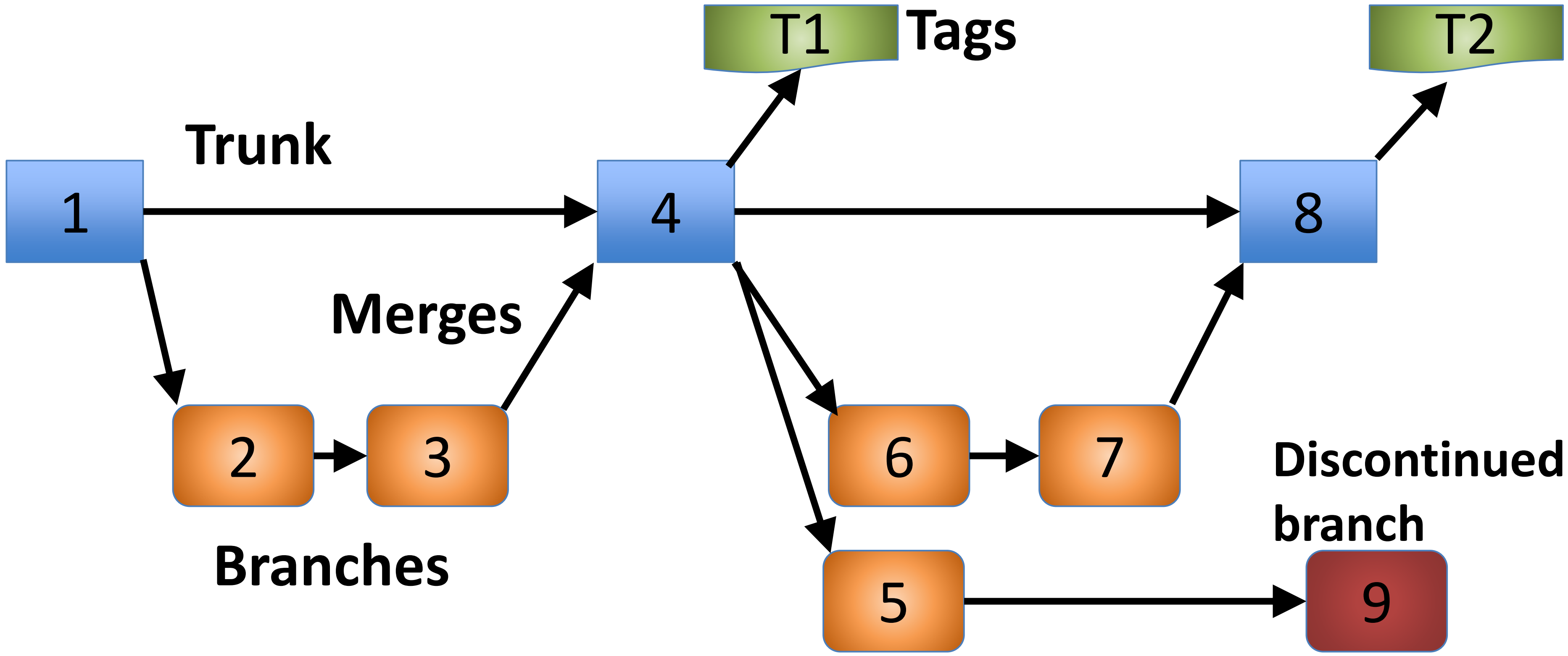
- Distributed Source Control Management (DSCM)
  - Each author has the complete repository locally (with history) and has complete control
  - Each author becomes the owner of his/her copy and can selectively accept changes made by others
  - Instead of having to trust a person, one has to trust the changes by deciding to incorporate them

Software tools

- Distributed Source Control Management (DSCM)
  - There isn't a central entity in charge of the work's history
  - Anyone can sync with any other team member
  - Open source project tend to use this type of versioning
  - Developer can work without connection to the centralized server
  - GitHub is a web-based hosting service for version control using Git

Software tools

- DSCM – Directed Acyclic Graph

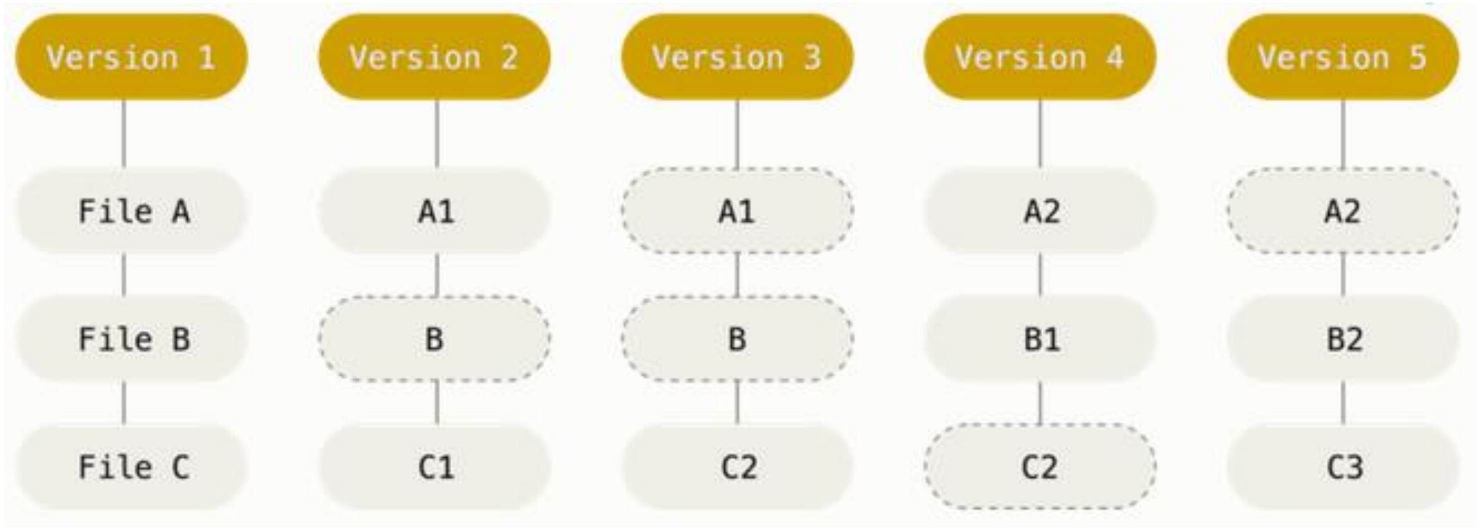


Software tools

- **Git** - states
  - ***Committed*** - the data is stored in your local database.
  - ***Modified*** - you have changed the file but have not committed it to your database yet.
  - ***Staged*** - you have marked a modified file in its current version to go into your next commit snapshot.

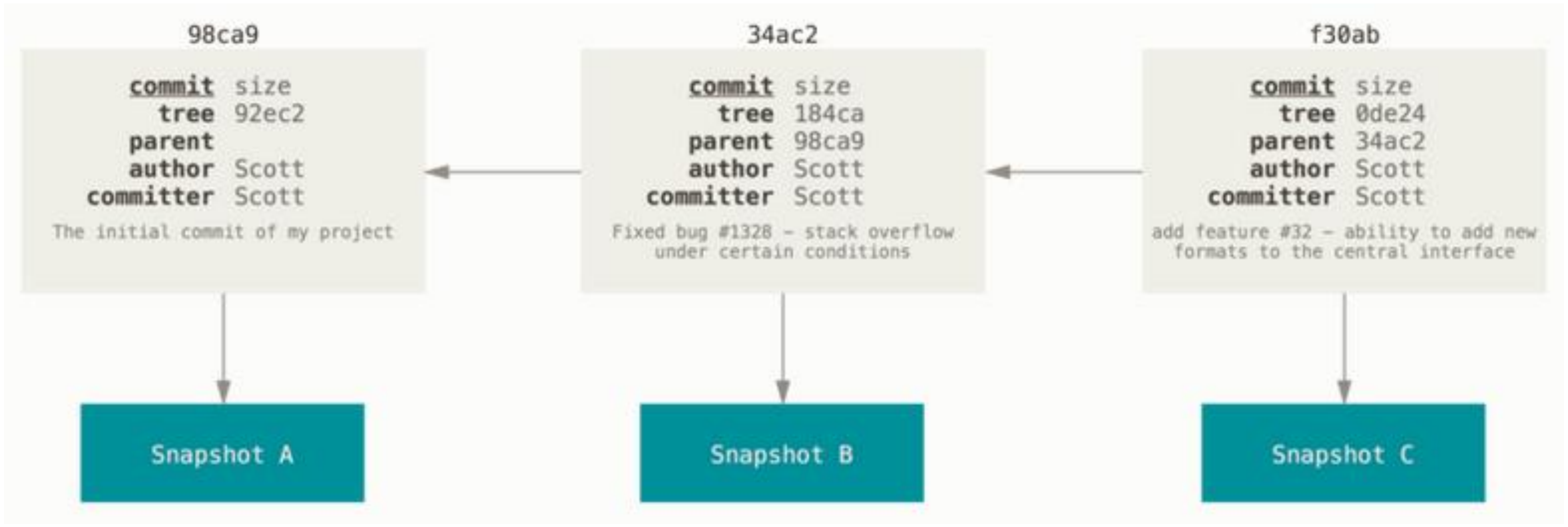
Software tools

- Git – snapshots, not differences



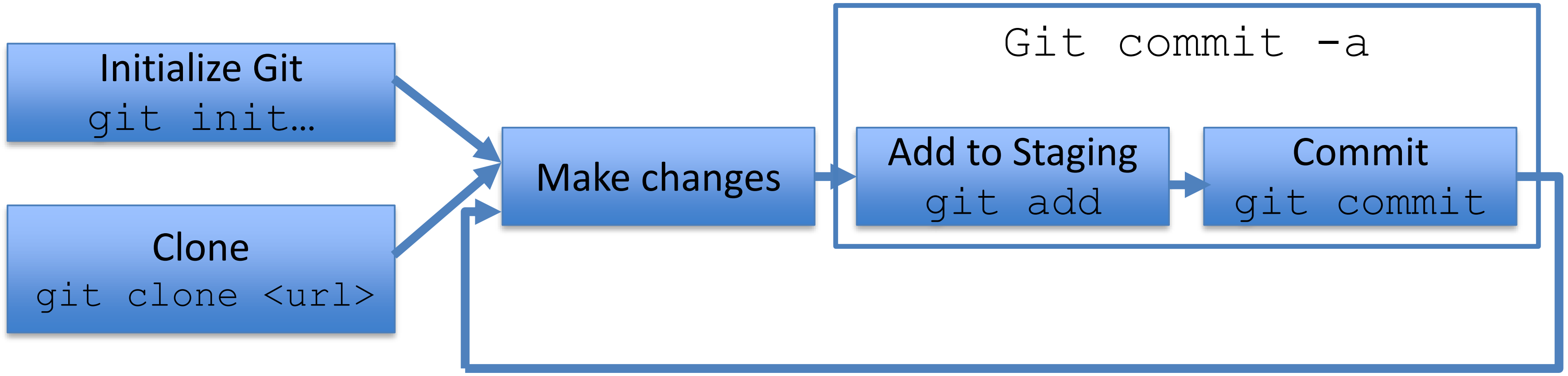
Software tools

- Git – Commits and their parents



Software tools

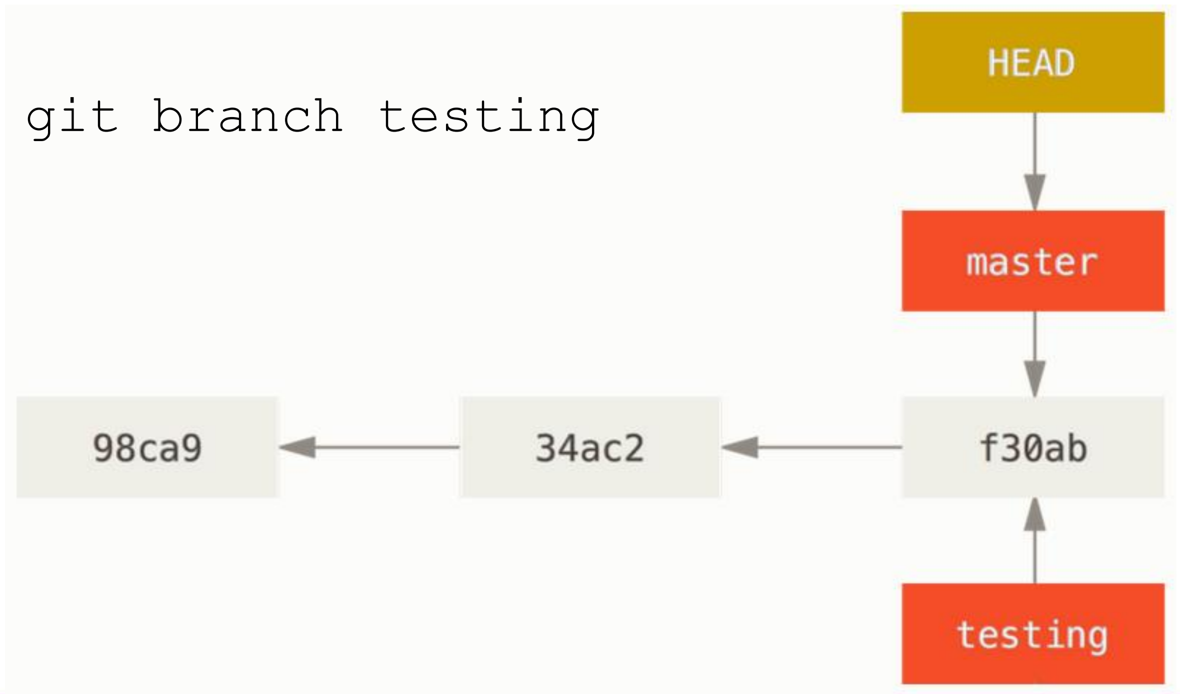
- Git - process



Only commit working, tested code

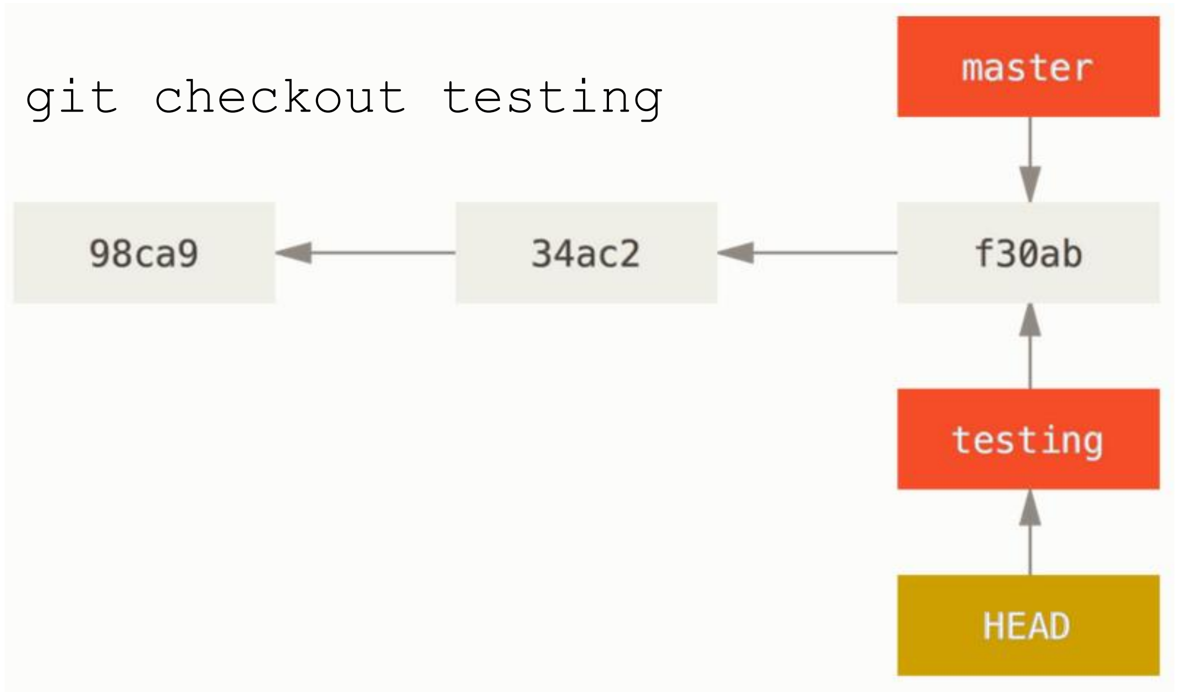
Software tools

- Git – Creating a New Branch
- Two branches pointing into the same series of commits
- **HEAD** - a pointer to the local branch you're currently on.



Software tools

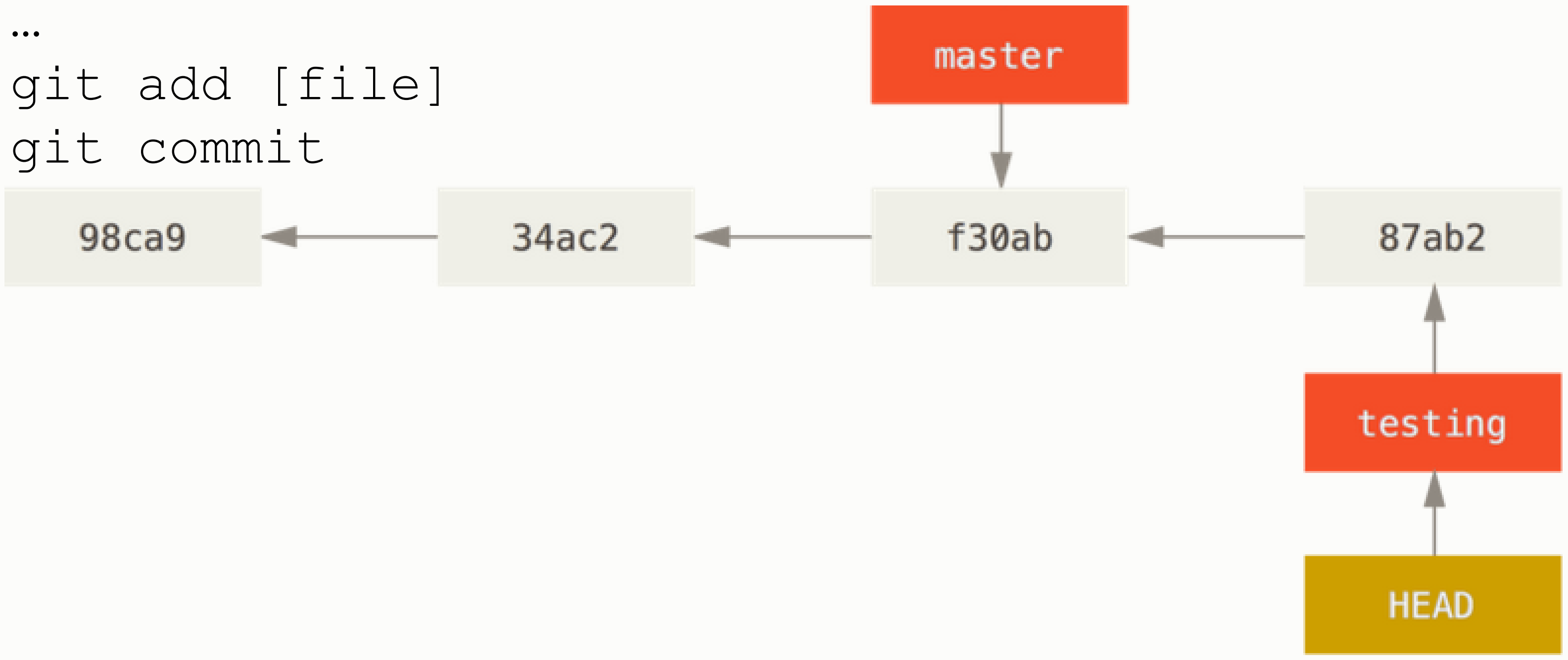
- Git – Switching Branches



Software tools

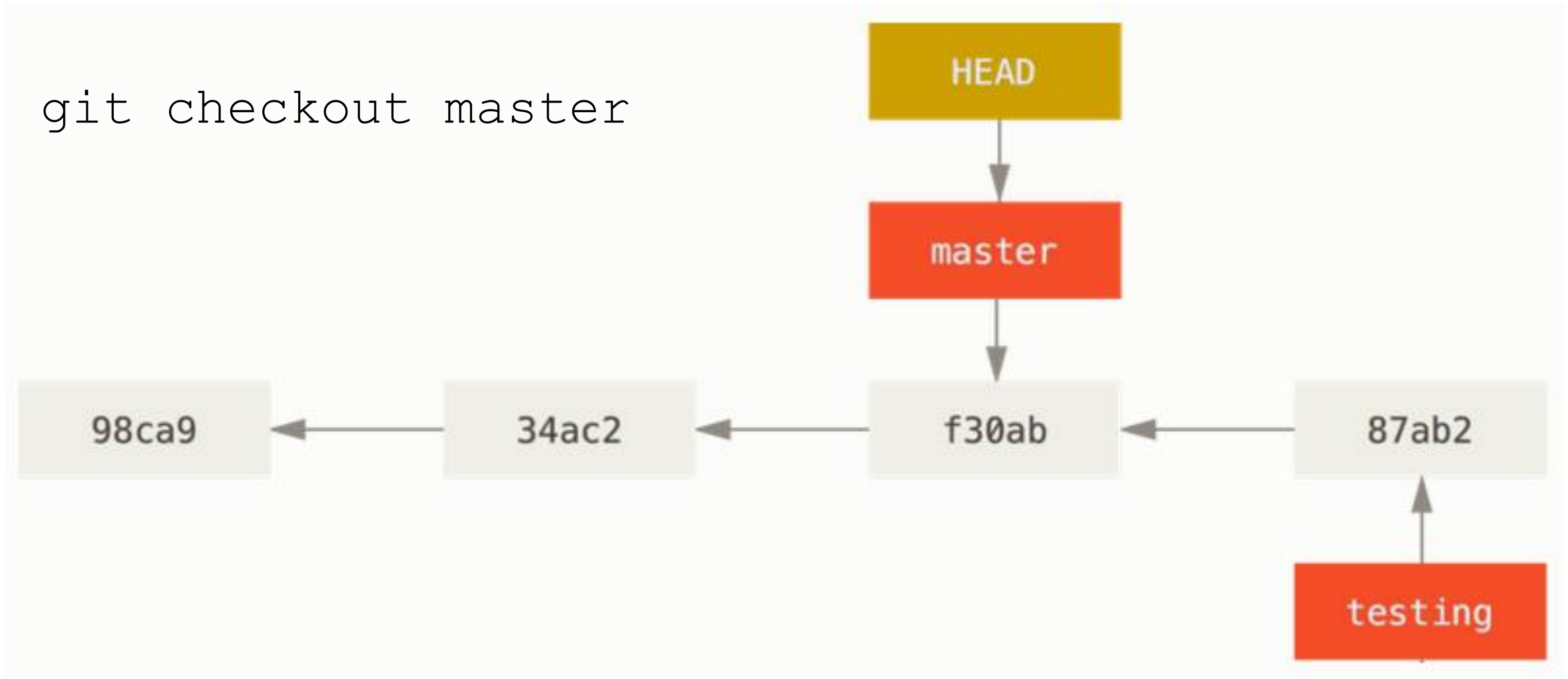
- Git – Changing ‘testing branch’

```
…
git add [file]
git commit
```



Software tools

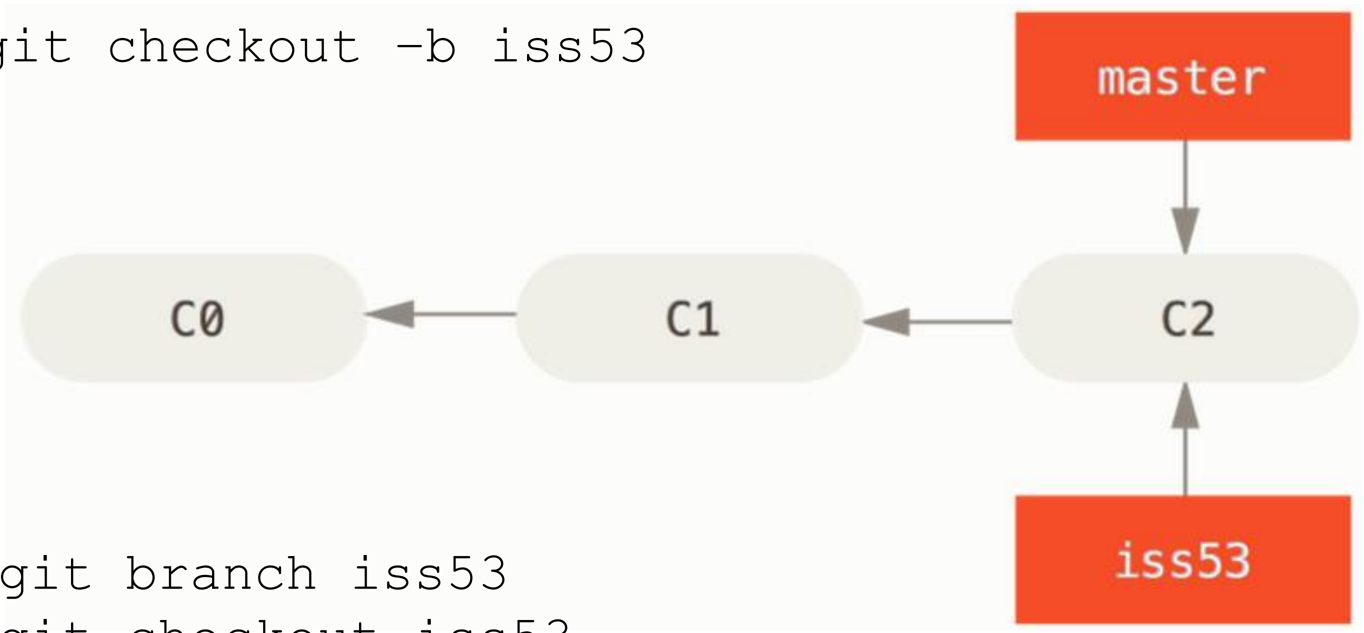
- Git – back to the master



Software tools

- Git – branching and merging

```
git checkout -b iss53
```



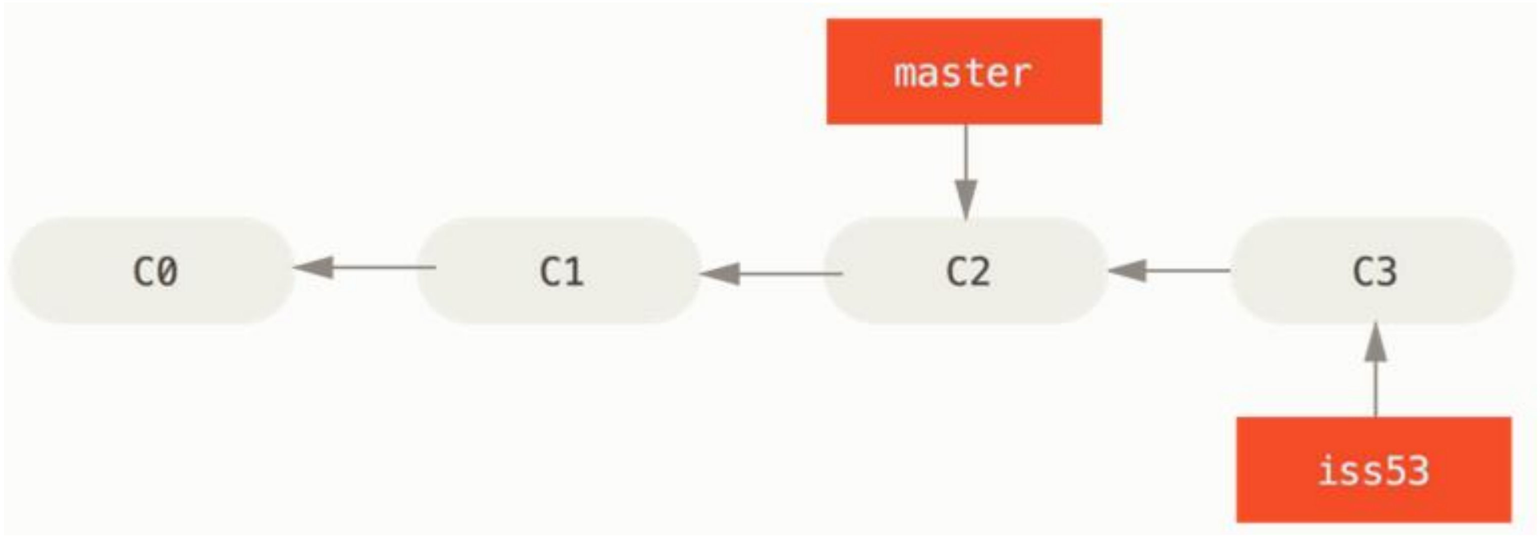
```
git branch iss53
git checkout iss53
```

Software tools

- Git – branching and merging

…

```
git commit -a 'changed something[issue 53]'
```



Software tools

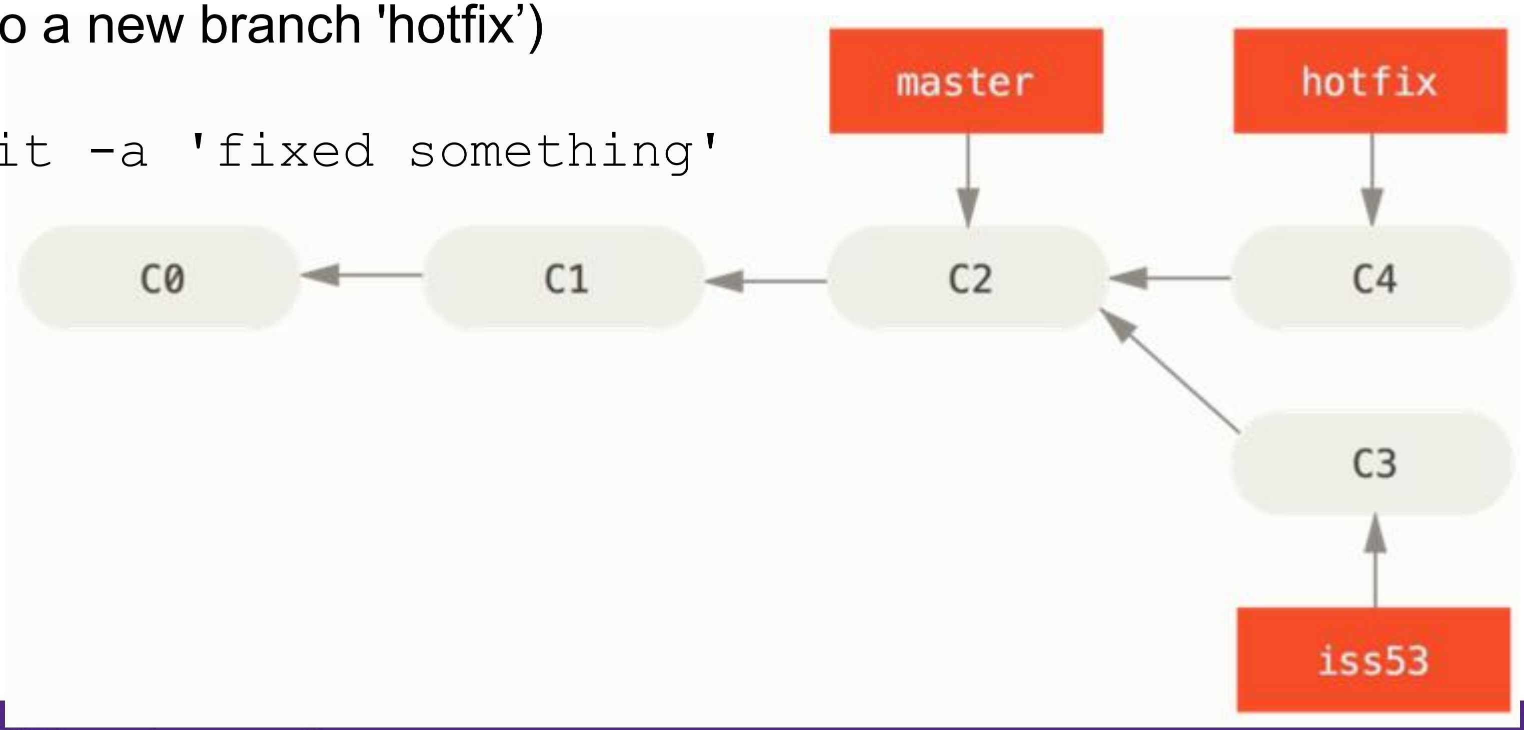
- Git – branching and merging

```
git checkout master
git checkout -b hotfix
```

(Switched to a new branch 'hotfix’)

…

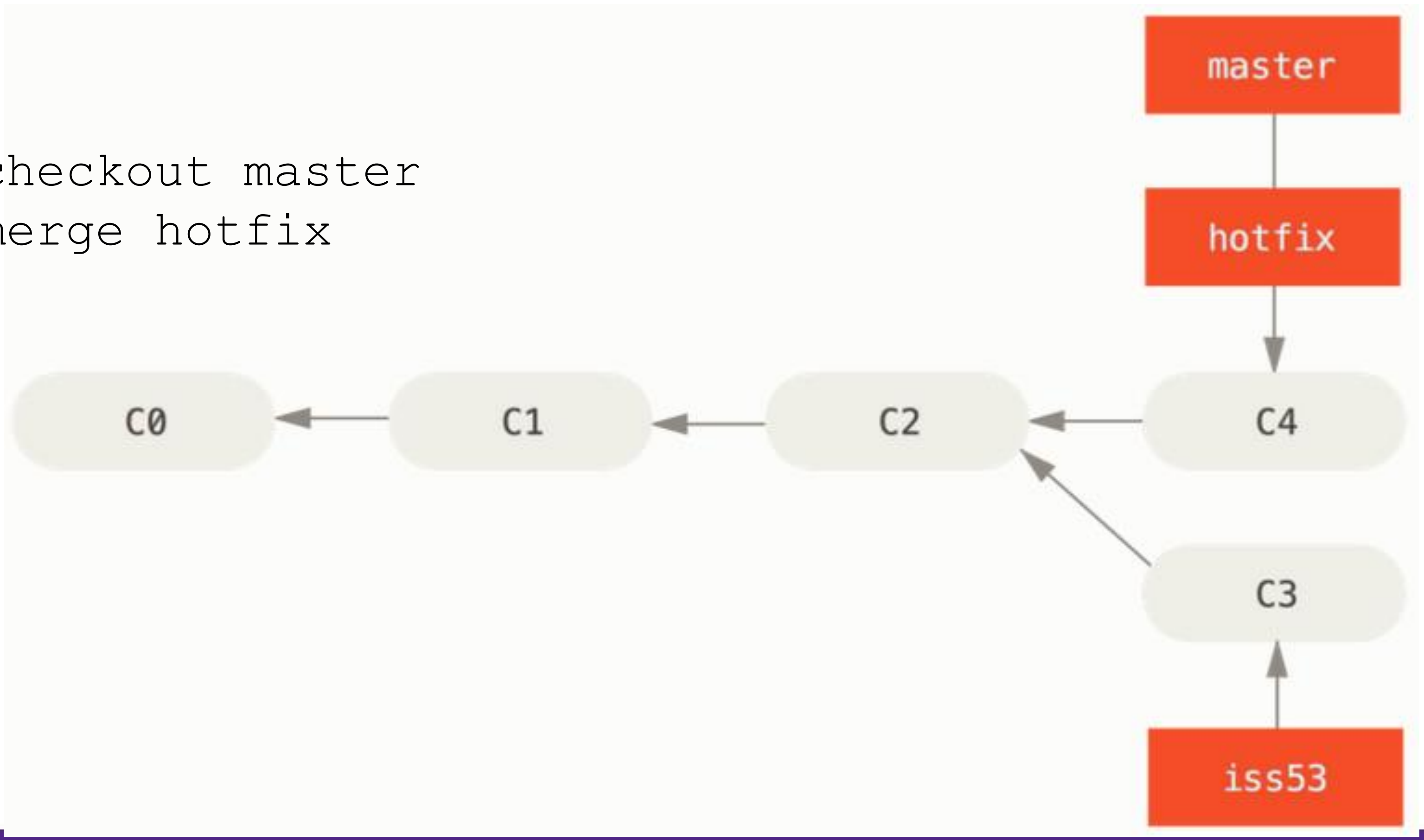
```
git commit -a 'fixed something'
```



Software tools

- Git – branching and merging

```
git checkout master
git merge hotfix
```



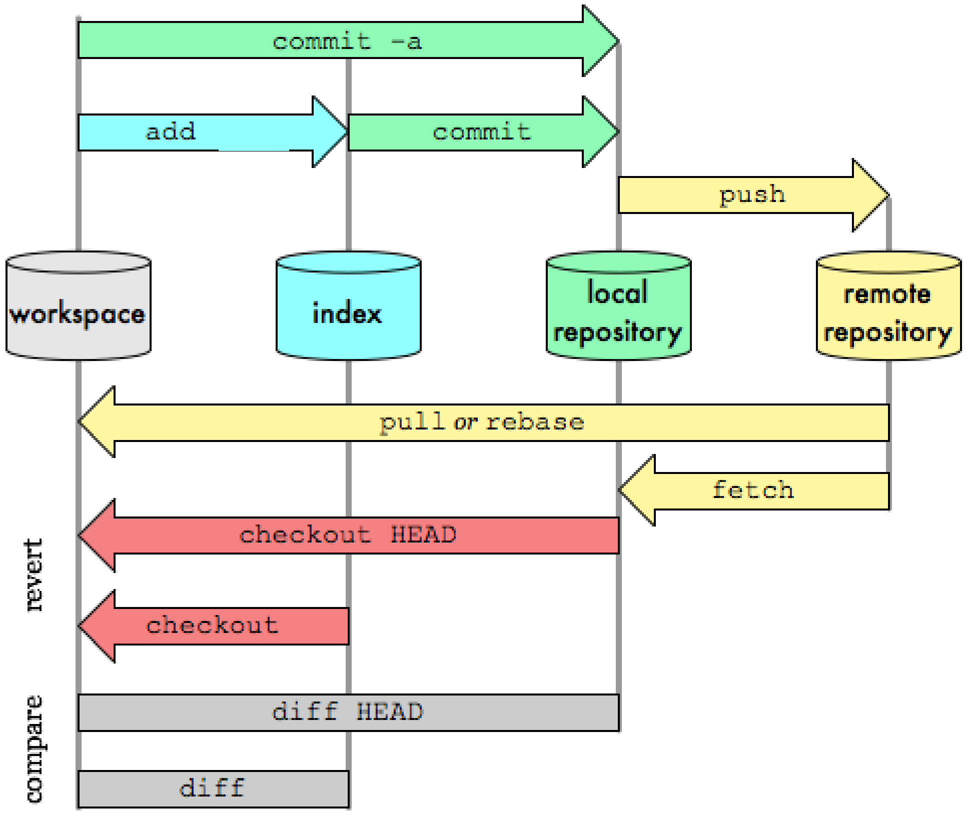
Software tools

- Git – commit and push

  `git commit` – commits changes to your local repository
  `git push` – pushes changes to a remote repository

Software tools

- Git – transport commands with remote repository

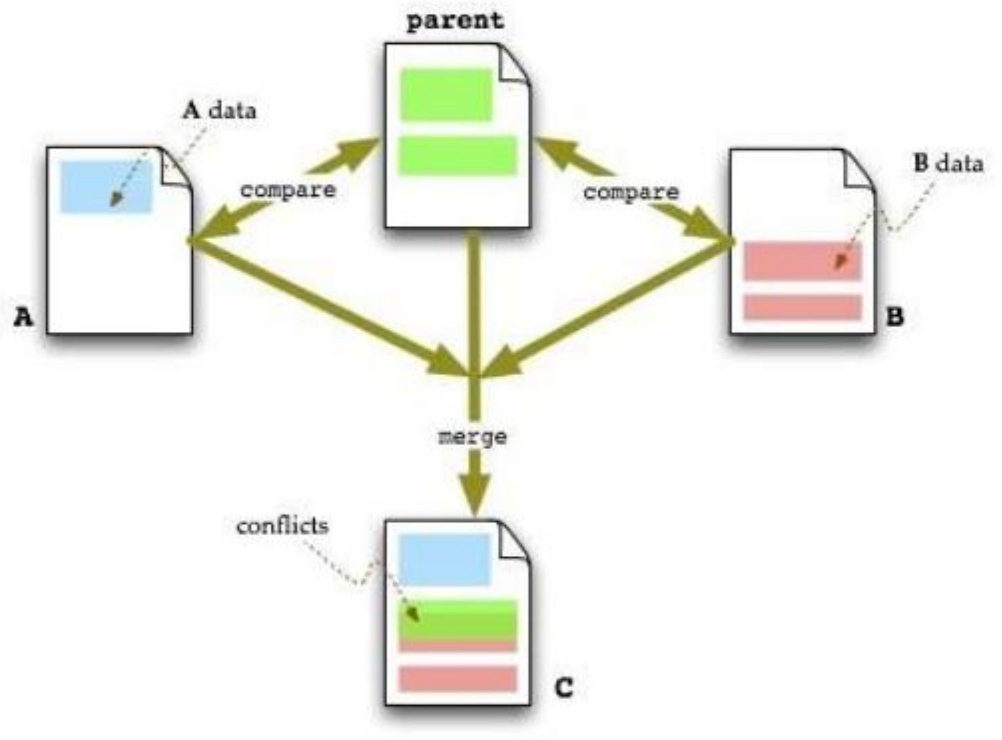


Software tools

- Git
  - Documentation https://git-scm.com/doc
  - Git cheat sheet
    - GitHub: https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf
    - Atlassian: https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet

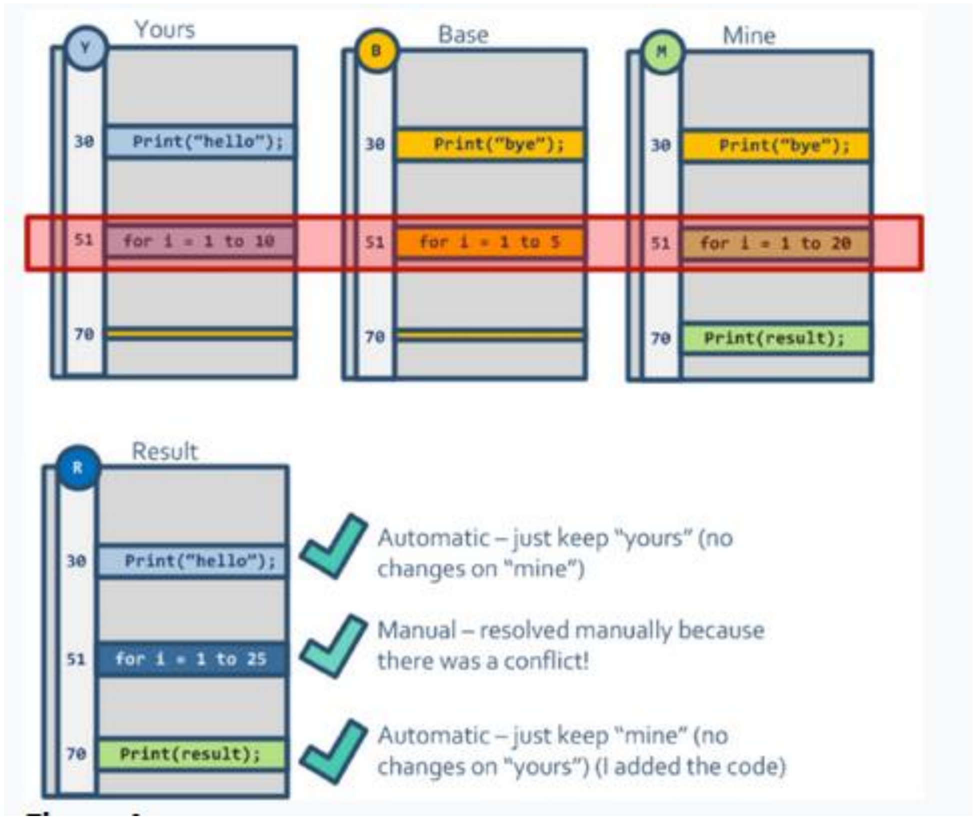
Software tools

- Source Code Merge Tool
  - Helps merge code and resolve conflicts
  - Helps compare files and version controlled projects (diff)
  - Two- and three-way comparison



Software tools

- Source Code Merge Tool
  - Three-way comparison



Software tools

- <u>Continuous integration tools</u>
  - Continuous integration – members of a team integrated their work at least daily
  - Each integration is verified by an automated build
  - An automated build includes testing to detect integration errors as quickly as possible
  - Automate build, test and deploy processes
  - Examples: Jenkins, TeamCity

Software tools

- Issue Management System (Issue tracking system)
  - An *issue* represents a concrete problem, such as a requirement, a design, or a management problem
  - An issue does not necessary indicate that there is a problem in developer's code

  - Issue management system - a software that maintains and manages lists of issues
  - Used to create, update, and resolve issues

Software tools

- <u>Issue Management System </u>(Issue tracking system)
  - Often provide project management functions
  - Workflow support
  - Can support planning (user stories, plan sprints)
  - Can support prioritization and distribute/assign tasks
  - Some have integrated time-tracking
  - Examples: Jira
    - Apache example: https://issues.apache.org/jira/secure/Dashboard.jspa
    - UWO WTS Service desk

Software tools

- Collaboration tools
  - Help teams collaborate - create, share, and discover content fast
  - Example features:
    - All content is searchable
    - Persistent chat rooms organized by topics
    - Private groups
    - Feedback in context
  - Examples: Confluence, Slack

Software tools

- Collaborative code review tools
  - Automates the code review process
  - Reviewer can see code changes, provide feedback (even on specific lines), approve or return changes
  - Example: Collaborator

Software tools

- <u>Unit test libraries/frameworks</u>
  - Support for unit testing
- <u>Automated testing (test automation)</u>
  - Use software to execute tests and identify defects
  - Examples: Selenium, TestingWhiz

Software tools

- Debuggers
  - Software programs used to test and debug other programs
  - Running programs step-by step, breakpoints, variable tracking
  - Some debuggers – ability to modify state while the program is running
  - Often part of IDE
  - …

Software tools

- Integrated Development Environments (IDE)
  - IDE is a software application that provides comprehensive facilities for software development.
  - Normally includes a source code editor, build automation tools, and a debugger.
  - Examples: Visual Studio, Eclipse
  - SE2250 IDE -> Unity