CISC 3120

Class Projects: Project 2 and Version Control Systems

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Project 1 Evaluation

- Requirements
  - 4 basic requirements
    - Input validation, maximum guess, game level, command line arguments
  - 2 bonus requirements
    - Game board area and game statistics, 3rd party library
- Grading scheme
  - Accepted or unaccepted for each requirement
  - Basic requirements: 4 accepted = A; 3 accepted = B; 2 accepted = C; 1 accepted = D
  - Bonus requirement: 1 – 2 accepted = one letter grade upgrade (A is upgraded to A+)

Project 1 Survey

- Two surveys
  - Peer evaluation
    - Mandatory: yon won't get a grade until you complete it
  - Course feedback
    - Voluntary: you are encouraged to provide feedback

What did we learn from our experience?

What you might have learned

- Project logic and resources should be organized in a manageable fashion
  - https://github.com/trending/c++
  - https://github.com/trending/java
  - …
- Application design
  - How do you decompose the project logic and resources into multiple components?
  - How do these components interact?
  - Is there a better design?

Project 2: Enhancing the Target Game

- Design with inheritance and polymorphism
- Group discussion (after a review and discussion on Version Control Systems)

Tool Support for Team Work

- Version control system
- Summary of your experience

Version Control System (VCS)

- Why do we need it?
  - https://stackoverflow.com/questions/1408450/

"

Have you ever:

Made a change to code, realised it was a mistake and wanted to revert back?
Lost code or had a backup that was too old?
Had to maintain multiple versions of a product?
Wanted to see the difference between two (or more) versions of your code?
Wanted to prove that a particular change broke or fixed a piece of code?
Wanted to review the history of some code?
Wanted to submit a change to someone else's code?
Wanted to share your code, or let other people work on your code?
Wanted to see how much work is being done, and where, when and by whom?
Wanted to experiment with a new feature without interfering with working code?

Team Support with VCS

- VCS provides a “centralized“ location to store project files
  - Versioned code, configuration files, build scripts …
- VCS tracks each contributors’ individual changes
- VCS helps prevent concurrent work from conflicting

Benefits of VCS

- Branching & merging.
  - Example workflow: branching for each feature, branching for each release.
- Traceability
  - Example use scenarios: track changes between revisions of a project, documented history of who did what and when
- Complete history of changes
  - Example use scenarios: help in root cause analysis for bugs, fix problems in older versions of software that has been released, roll back to an older version without newly introduced bugs

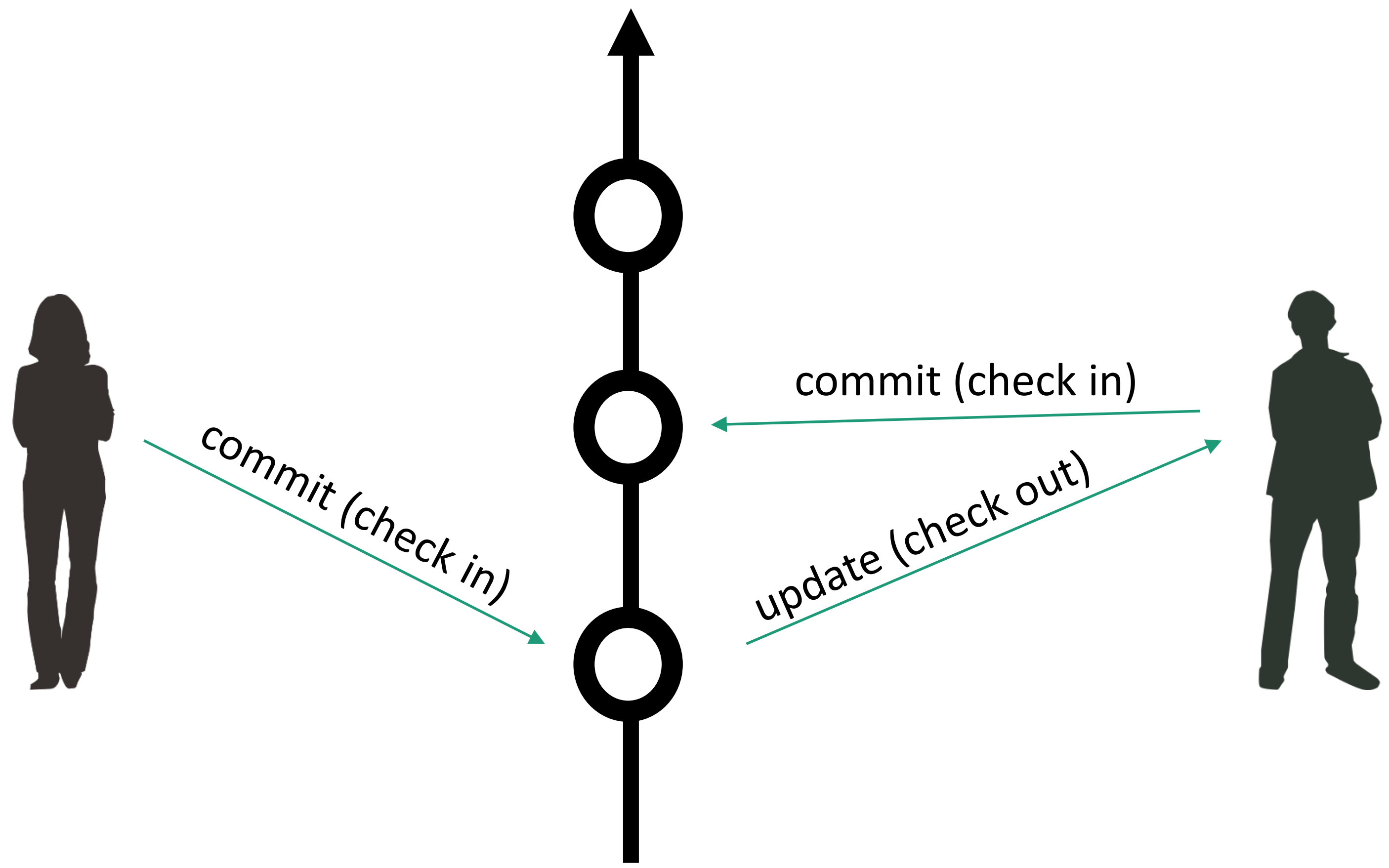
Centralized vs. Distributed

- Centralized VCS
    - Examples: Revision Control System (RCS), Concurrent versions systems (CVS), Subversion (SVN)
- Distributed VCS
    - Examples: Git, Mercurial (hg)

Basic VCS Operations

- Check out/update: copying the repository to the machine you are working at
- Check in/Commit: copying the changes you made to the repository and creating a new version
- Branch: create a new “child” development from a state of the repository

Example: Centralized Workflow

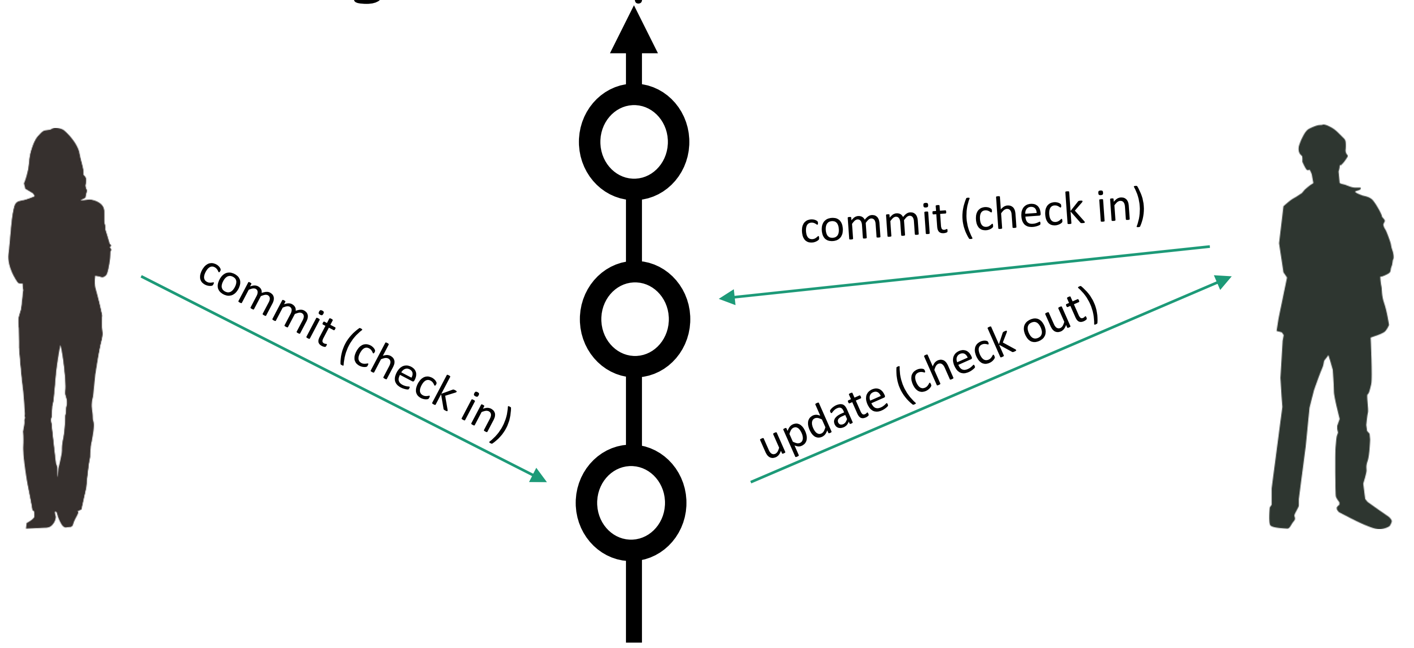


Merge Conflicts

- A conflict may occur when two developers edit the same file
- Merge
  - The developer that tries to commit the file ***last*** will have to combine her changes with those of the prior developer
  - Many VCS's (e.g., git) may automatically combine the changes
  - Developers may have to merge the changes by hand

Question: A Merge?

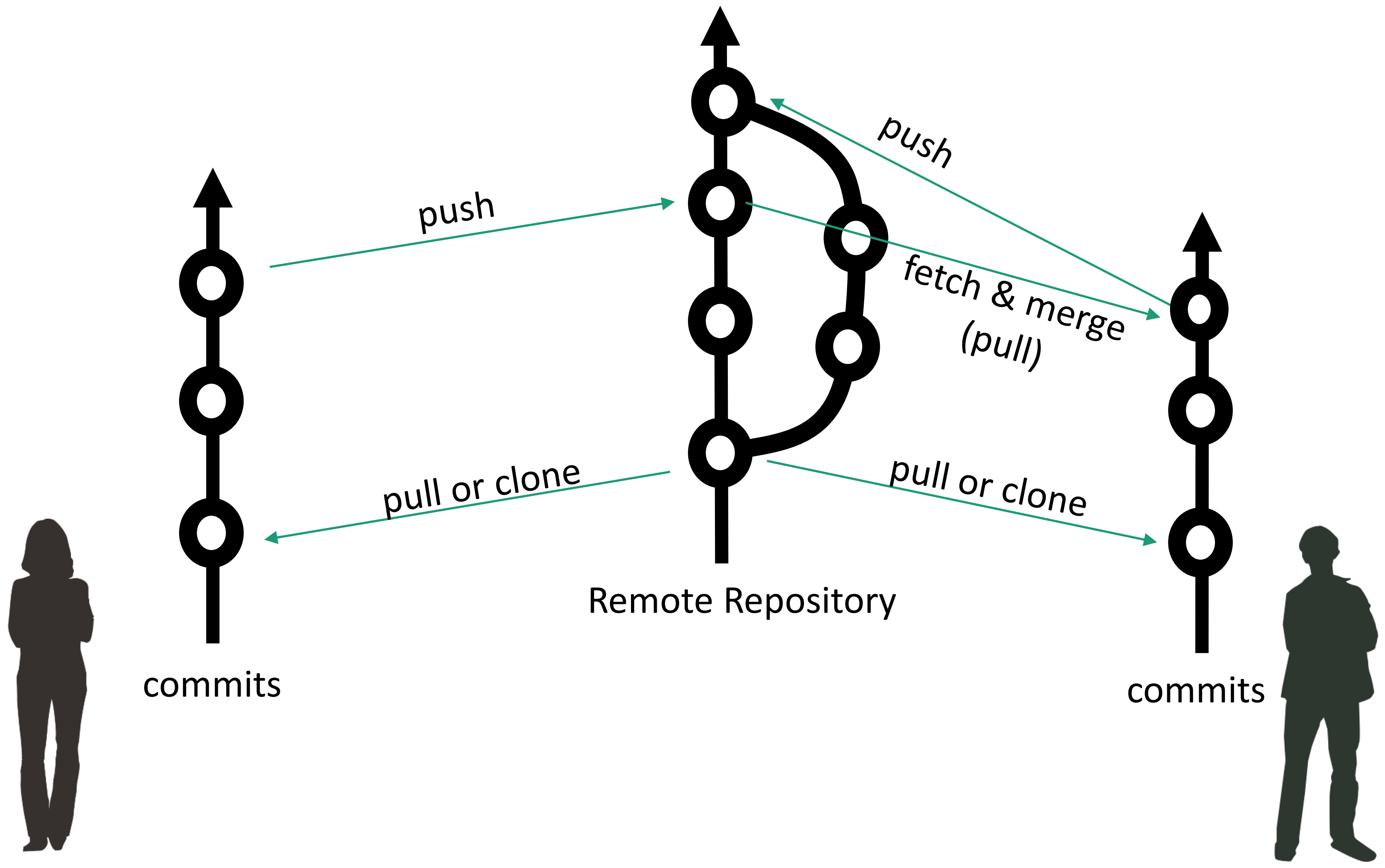
- How would you revise this graph to illustrate when a merge is required?



Distributed Version Control

- Possible to commit locally without upsetting the others
- Allow more flexibility and support different kinds of workflow

Example: Distributed Workflow



Final Thought

- *"There are no winners on a losing team, and no losers on a winning team."*

— Frederick Brooks Jr.

Questions

- Projects 1 & 2
- Learning from experience: Version Control Systems