Outline of today's session

- Introductions (again): students, assistants and instructor
- Questions?
  - Reading quizzes? Annotated input-compute-output example?
- Object, Type, Value, Variable and Assignment
  - Exercise in PyDev Console
- Calling functions
  - Exercises m0 through m4
- The input-compute-output pattern
  - Exercises m0 and m1
- Fixing syntax errors
  - Exercise m2
- Turning in your work: Version Control with SVN

Homework: Reading and exercises m5 and m6.

Roll Call & Introductions

- Name (nickname)
- Hometown
- Where you live on (or off) campus
- Something about you that most people in the room don't know

```
def main():
    """ Calls a function (chaos). """
    chaos()

def chaos():
    """
    Computes and prints a chaotic sequence of numbers,
    as a function of a number input from the user.
    """
    print('This function illustrates a (possibly) chaotic sequence.')
    input_string = input('Enter a number between 0 and 1: ')
    x = float(input_string)

    for k in range(10):
        x = 3.9 * x * (1 - x) # Try constants other than 3.9 if you like
        print(k, ':', x)

    print('Examine the sequence of numbers printed in the right column.')
    print('Does it appear chaotic?')
```

Questions:

- On the reading from the Syllabus?
- On Chapters 1 and 2?
- On the code you studied?
- On anything?

Object, *Type*, *Value*, *Variable* and *Assignment*

- Work through the handout on this topic
- Ask questions! Be curious!

Functions and function calls

- We'll talk through the handout on this topic
- Ask questions! Be curious!

The input-compute-output pattern

- **Checkout** today's project: **`Session02_InputCompute_Output`**
- **Study the m0** module briefly.
  - Ask questions as needed! Note especially **`math.cos(...)`**
- **Do the TODO's in the m1** module.
  - Refer back to the m0 module for examples.
- **Do the TODO's in the m2** module
  - You will very likely have questions here!
- Continue to the other modules if you get ahead.

Software Engineering Tools

- The computer is a powerful tool
- We can use it to make software development easier and less error prone!
- Some software engineering tools:
  - IDEs, like Eclipse and IDLE
  - Version Control Systems, like Subversion
  - Testing frameworks, like JUnit
  - Diagramming applications, like UMLet, Violet and Visio
  - Modeling languages, like Alloy, Z, and JML
  - Task management trackers like TRAC

Version Control Systems

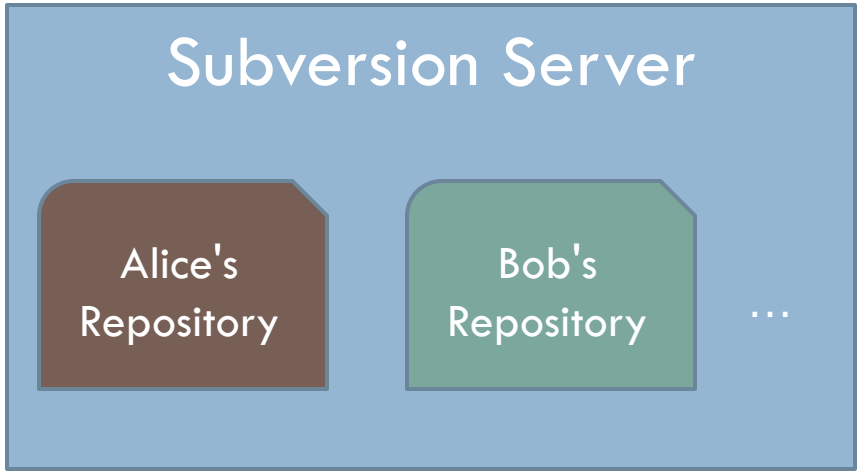
- Store "snapshots" of all the changes to a project over time
- Benefits:
  - Multiple users
    - Multiple users can share work on a project
    - Record who made what changes to a project
    - Provide help in resolving conflicts between what the multiple users do
    - Maintain multiple different versions of a project simultaneously
  - Logging and Backups
    - Act as a "global undo" to whatever version you want to go back to
    - Maintain a log of the changes made
    - Can simplify debugging
  - Drop boxes are history!
    - Turn in programming projects
    - Get it back with comments from the grader embedded in the code

Our Version Control System

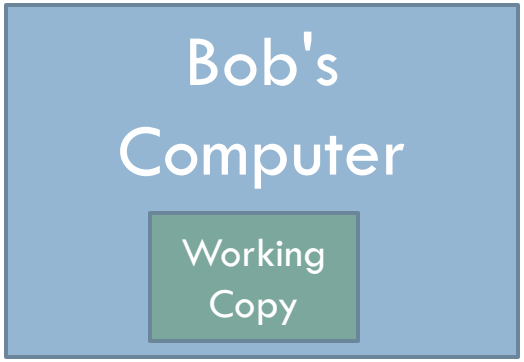
- Subversion, sometimes called SVN
- A free, open-source application
- Lots of tool support available
  - Works on all major computing platforms
  - **TortoiseSVN** for version control in Windows Explorer
  - **Subclipse** for version control inside Eclipse

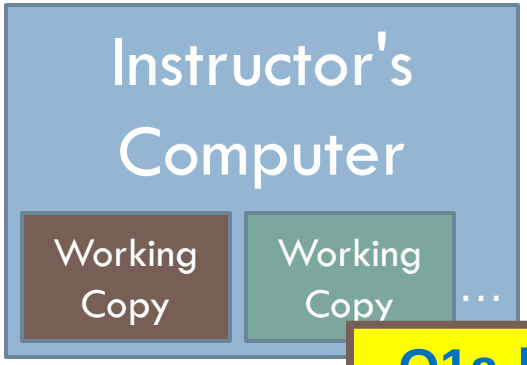
Version Control Terms

Repository: the copy of your data on the server, includes ***all*** past versions



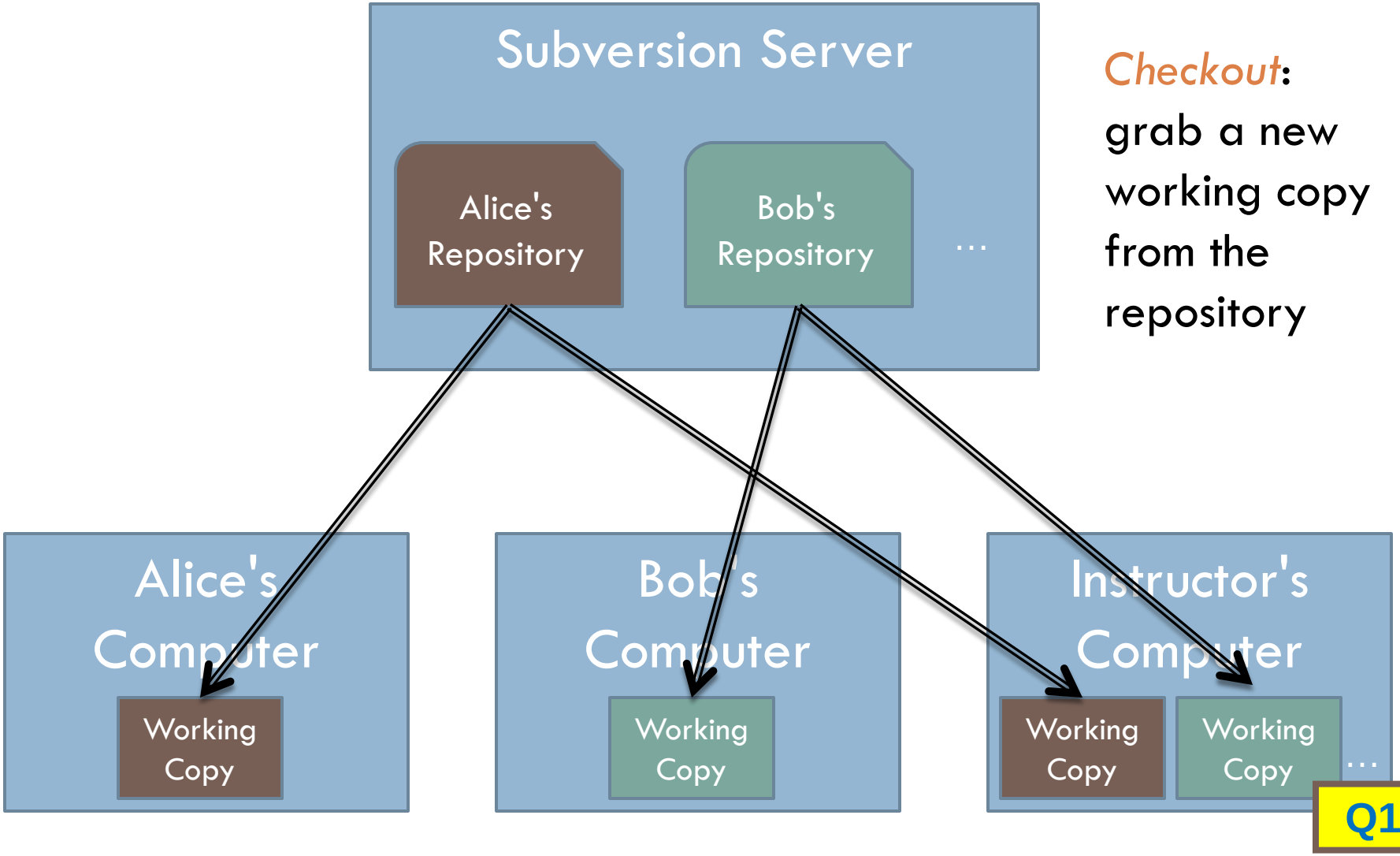
Working copy: the version of your data on ***your computer***





Q1a-b

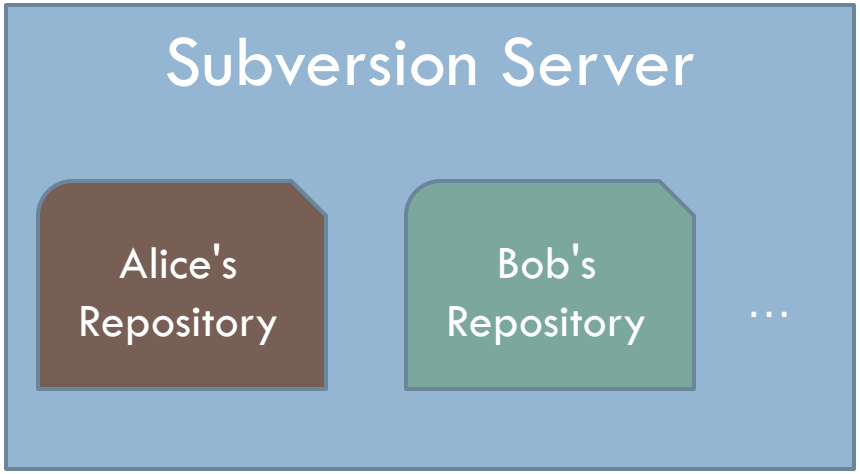
Version Control Steps—Checkout



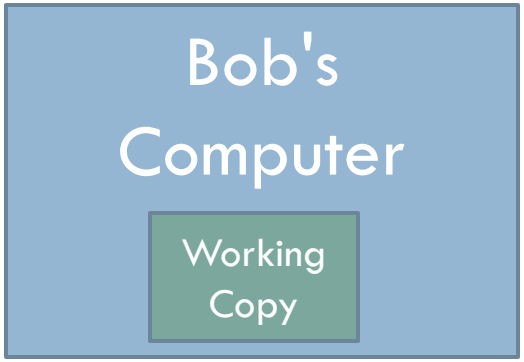
Checkout: grab a new working copy from the repository

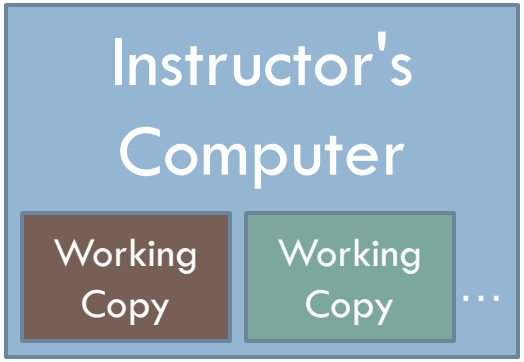
Q1c

Version Control Steps—Edit

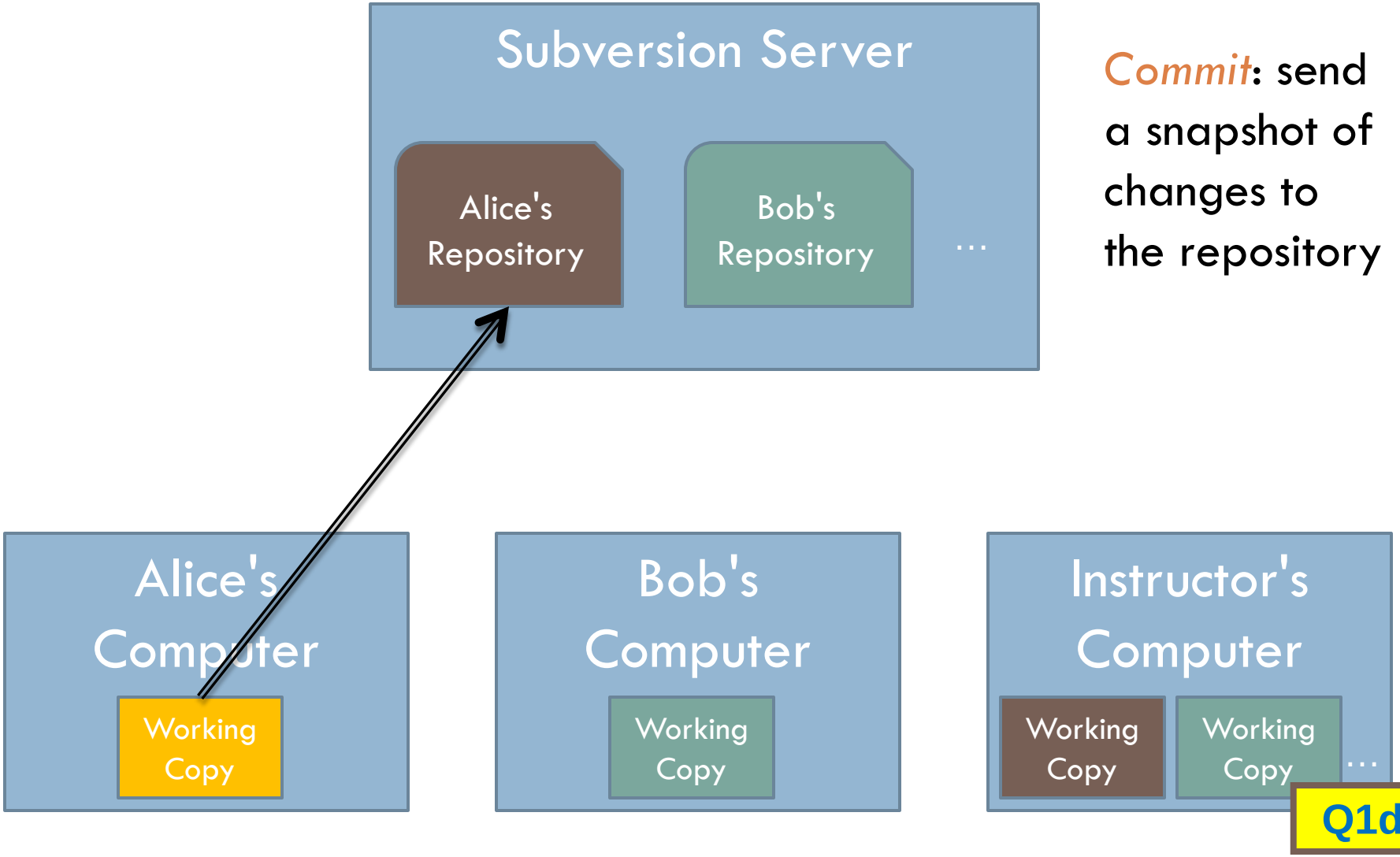


Edit: make ***independent*** changes to a working copy





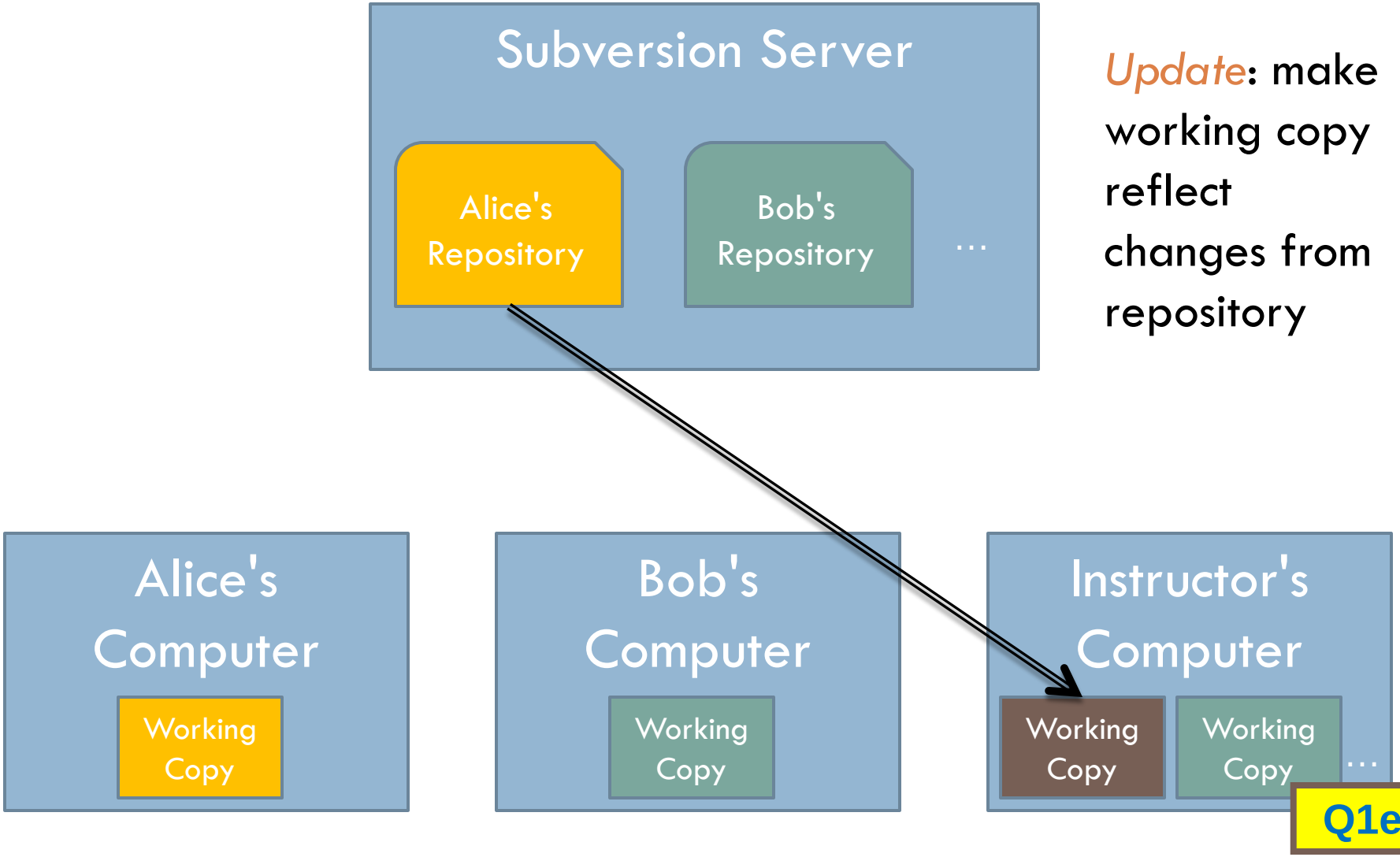
Version Control Steps—Commit



Commit: send a snapshot of changes to the repository

Q1d

Version Control Steps—Update



Update: make working copy reflect changes from repository

Rest of Session

- ***Finish the in-class quiz and turn it in. Ask questions!***
- ***Continue working on the TODO's in the modules***
  - Ask questions as needed!
- **Sources of help after class:**
  - **Assistants in the CSSE lab**
    - And other times as well (see link on the course home page)
  - **Email** `csse120-staff@rose-hulman.edu`
    - You get faster response from the above than from just your instructor

```
Moench D-219 (and F-217)
7 to 9 p.m.
Sundays thru Thursdays
```