OBJECT-ORIENTED LANGUAGE AND THEORY

# 1-2. VERSION CONTROL

Nguyen Thi Thu Trang
trangntt@soict.hust.edu.vn

# Outline

1. **Introduction**
2. Models
3. Vocabulary
4. Tools

# Why version control? (1/2)

- Scenario 1:
  - Your program is working
  - You change “just one thing”
  - Your program breaks
  - You change it back
  - Your program is still broken--*why?*
- Has this ever happened to you?

# Why version control? (2/2)

- Your program worked well enough yesterday
- You made a lot of improvements last night...
  - ...but you haven't gotten them to work yet
- You need to turn in your program *now*

- Has this ever happened to you?

# Version control for teams (1/2)

- Scenario:
  - You change one part of a program--it works
  - Your co-worker changes another part--it works
  - You put them together--it doesn't work
  - Some change in one part must have broken something in the other part
  - What were all the changes?

# Version control for teams (2/2)

- Scenario:
  - You make a number of improvements to a class
  - Your co-worker makes a number of different improvements to the same class
- How can you merge these changes?

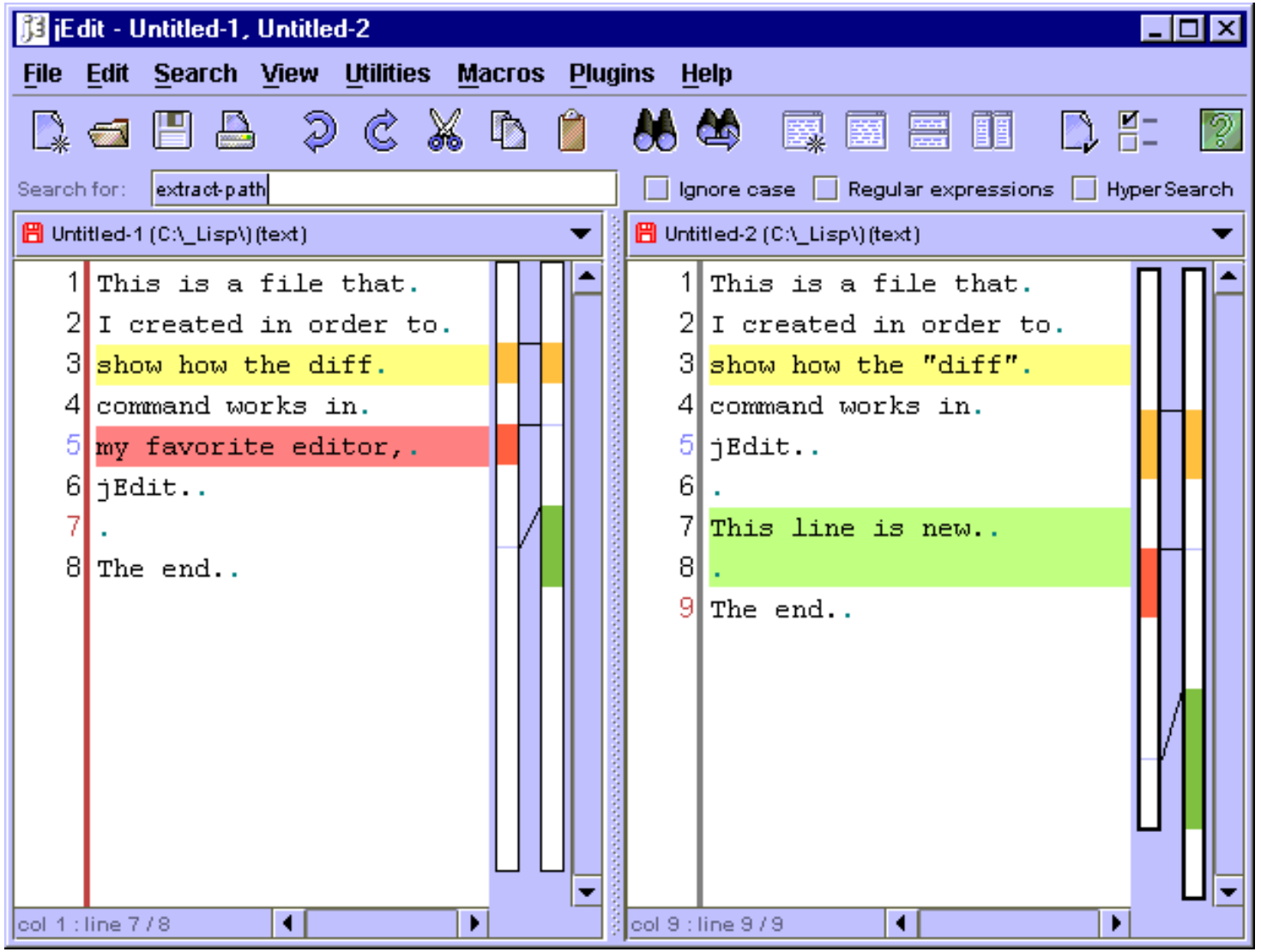
# Tools: diff

- There are a number of tools that help you spot changes (differences) between two files
- Tools include diff, rcsdiff, jDiff, etc.
- Of course, they won't help unless you kept a copy of the older version
- Differencing tools are useful for finding a *small* number of differences in a *few* files

# Tools: jDiff

- jDiff is a plugin for the jEdit editor
- Advantages:
  - Everything is color coded
  - Uses synchronized scrolling
  - It's inside an editor--you can make changes directly
- Disadvantages:
  - Not stand-alone, but must be used within jDiff
  - Just a diff tool, not a complete solution

# Tools: jDiff


jEdit - Untitled-1, Untitled-2
File Edit Search View Utilities Macros Plugins Help
Search for: extract-path
Ignore case
Regular expressions
HyperSearch
Untitled-1 (C:_Lisp\) (text)
1 This is a file that.
2 I created in order to.
3 show how the diff.
4 command works in.
5 my favorite editor,.
6 jEdit..
7 .
8 The end..
col 1 : line 7 / 8
Untitled-2 (C:_Lisp\) (text)
1 This is a file that.
2 I created in order to.
3 show how the "diff".
4 command works in.
5 jEdit..
6 .
7 This line is new..
8 .
9 The end..
col 9 : line 9 / 9

# Version control systems

- A version control system (often called a source code control system) does these things:
  - Keeps multiple (older and newer) versions of everything (not just source code)
  - Requests comments regarding every change
  - Allows “check in” and “check out” of files so you know which files someone else is working on
  - Displays differences between versions

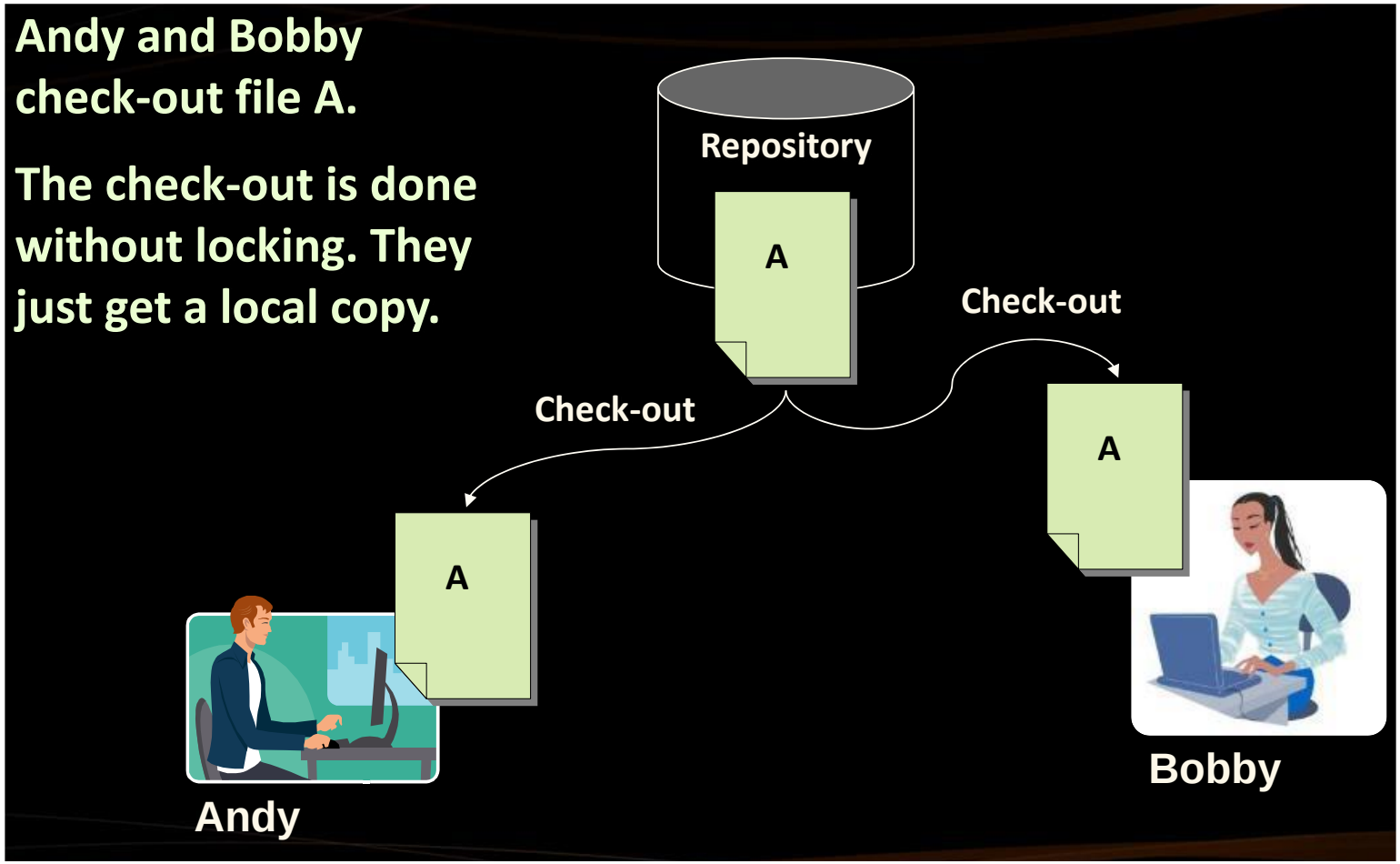
# Outline

1. Introduction
2. **Versioning Models**
3. Vocabulary
4. Tools

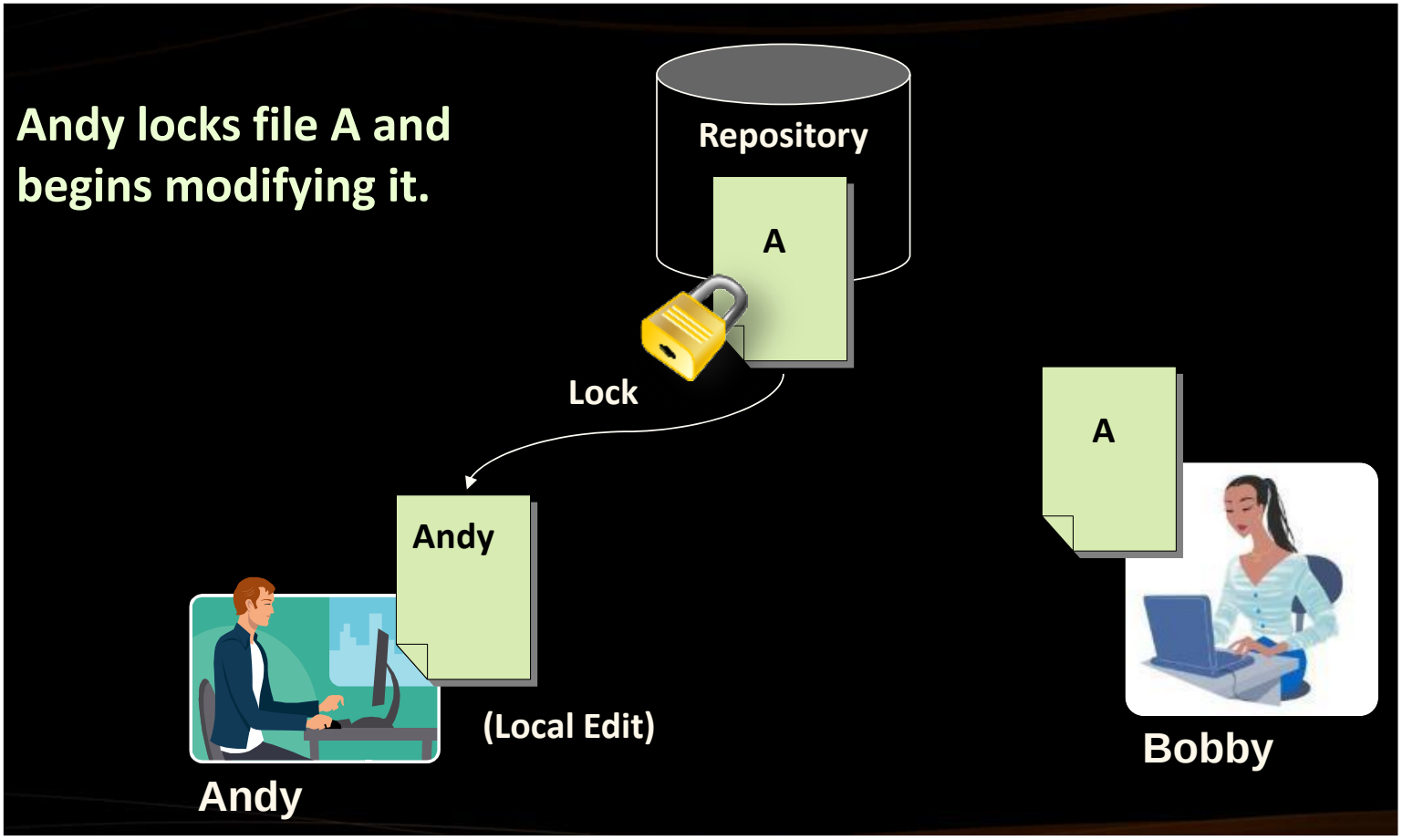
# 2. Versioning Models

- Lock-Modify-Unlock
- Copy-Modify-Merge
- Distributed Version Control

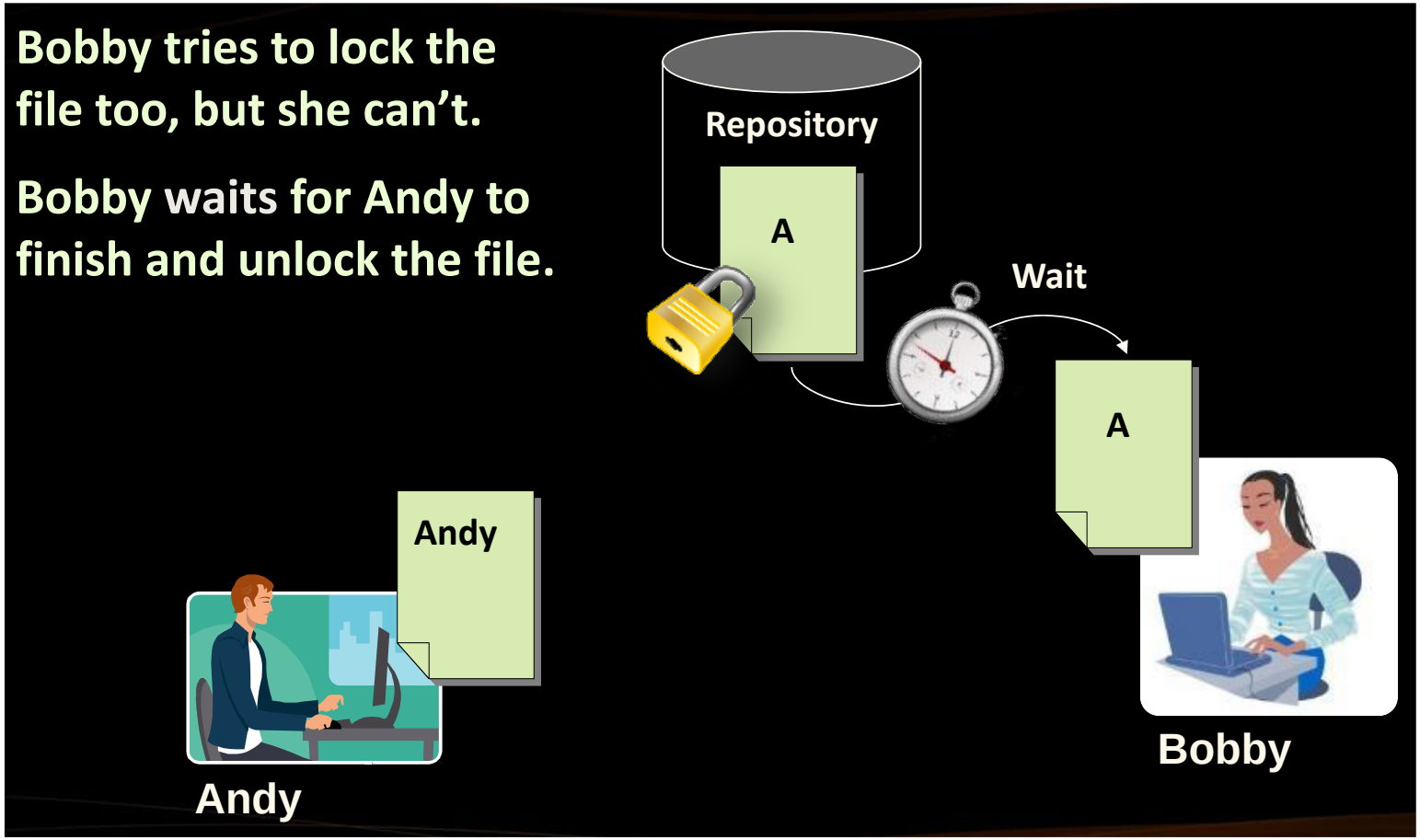
# 2.1. The Lock-Modify-Unlock Model



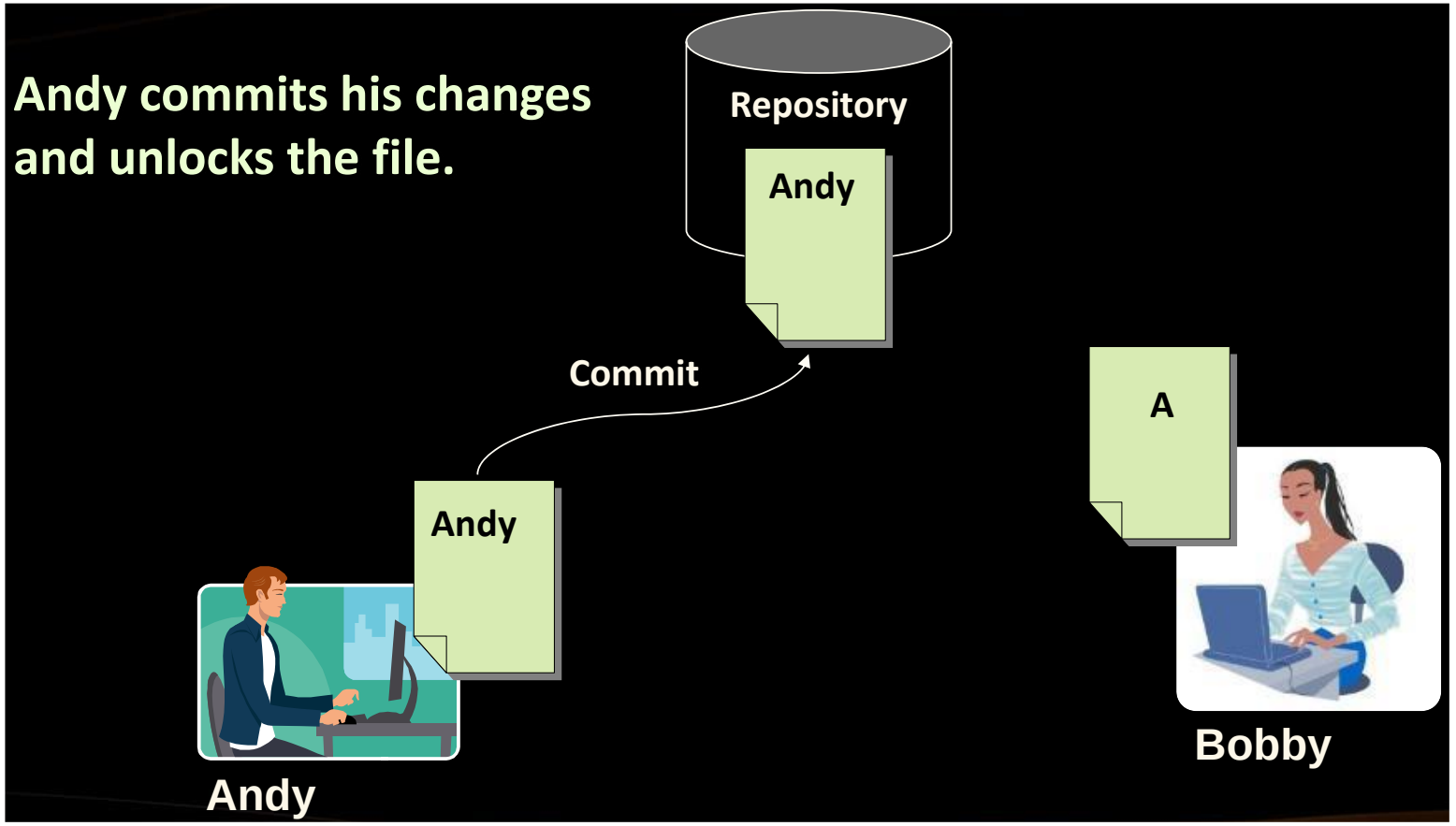
# 2.1. The Lock-Modify-Unlock Model (2)



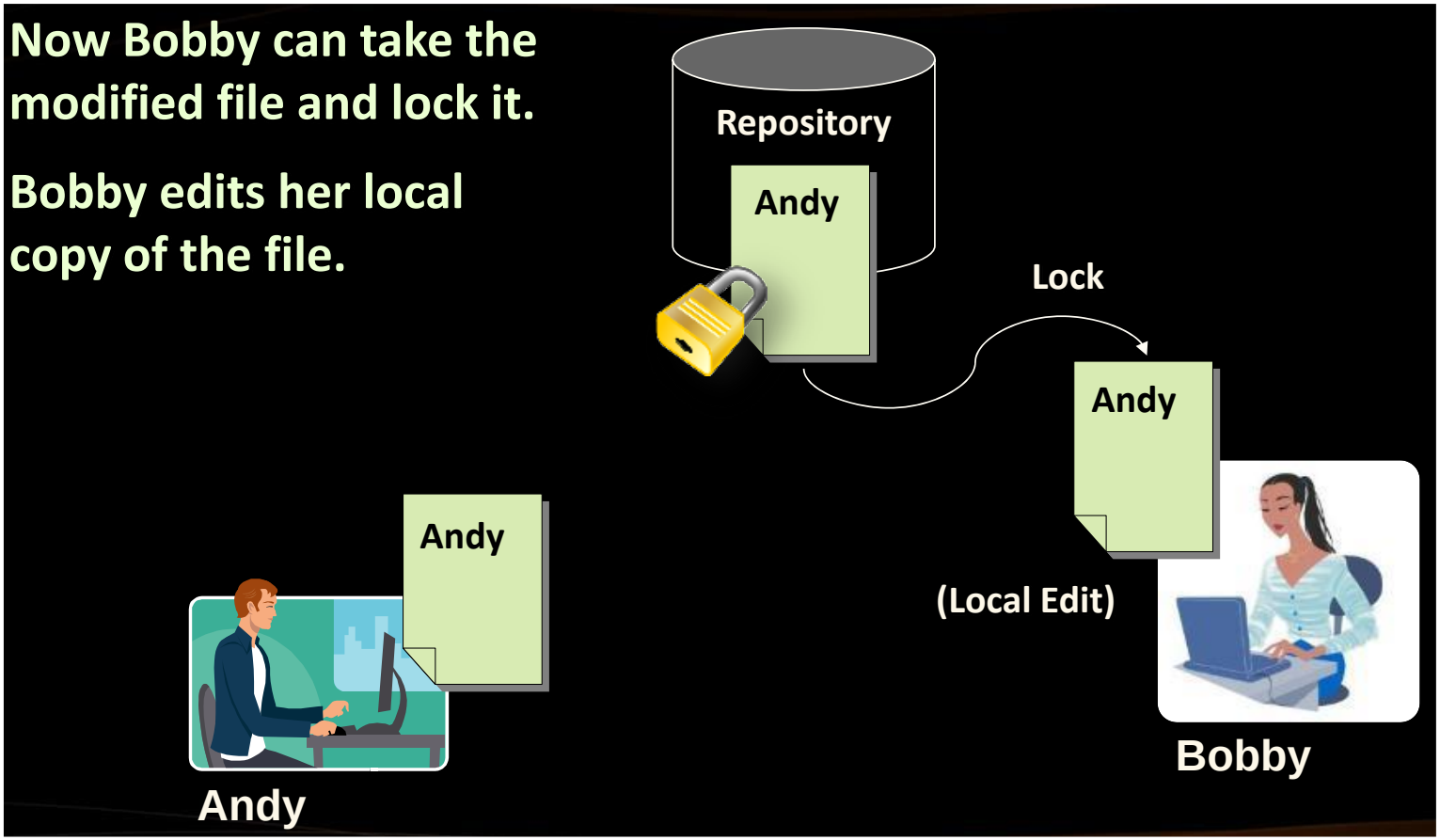
# 2.1. The Lock-Modify-Unlock Model (3)



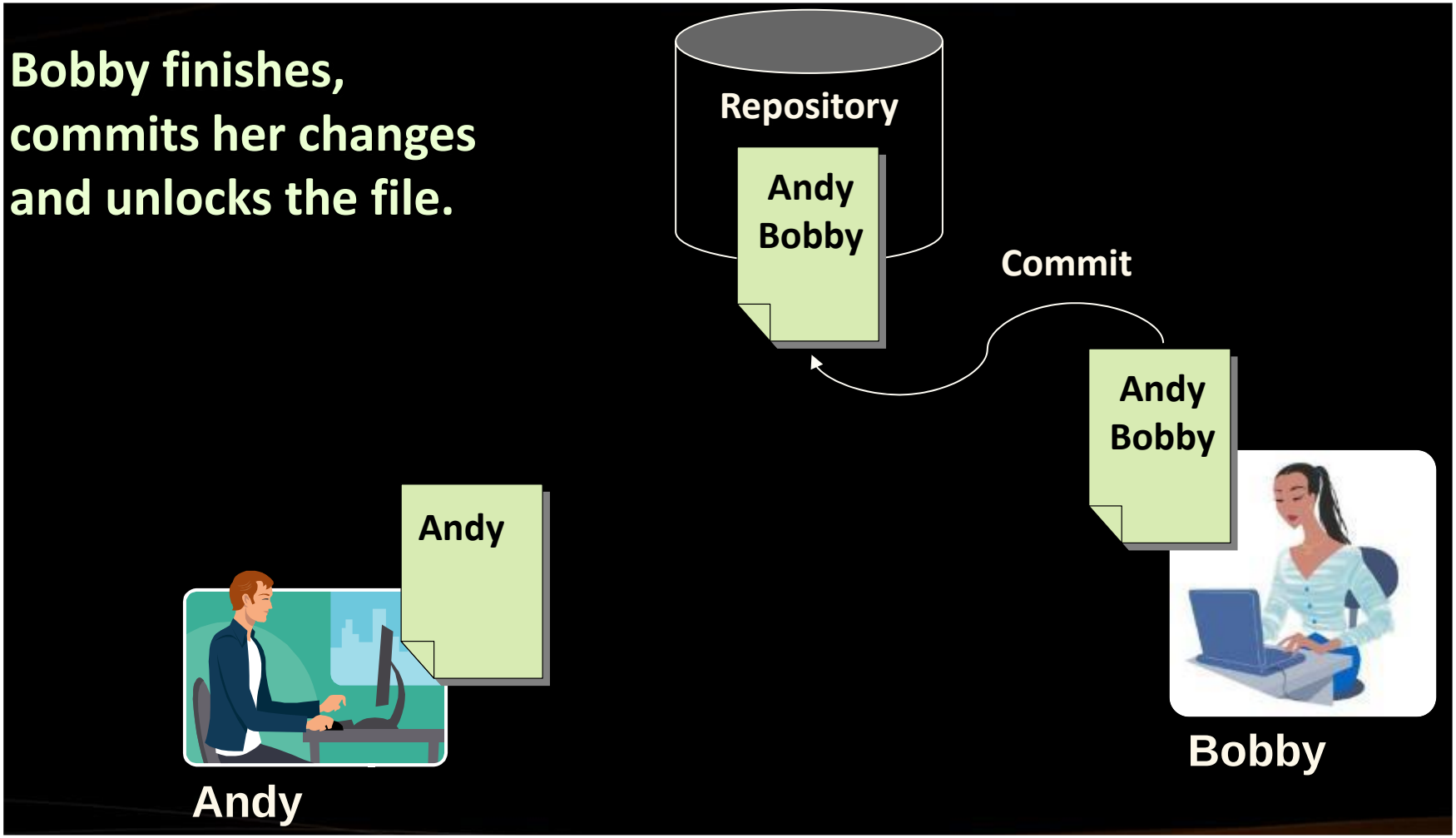
# 2.1. The Lock-Modify-Unlock Model (4)



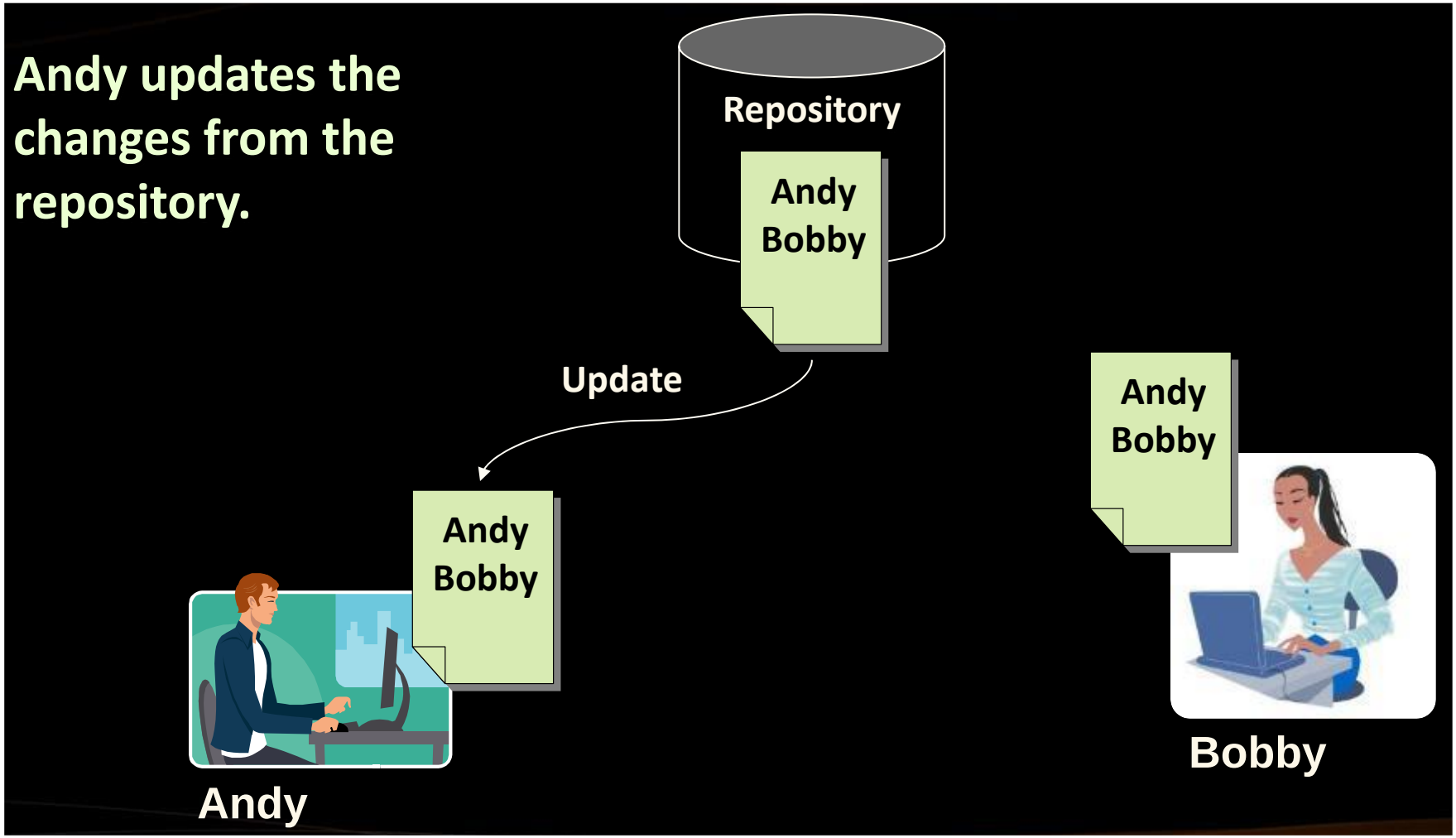
# 2.1. The Lock-Modify-Unlock Model (5)



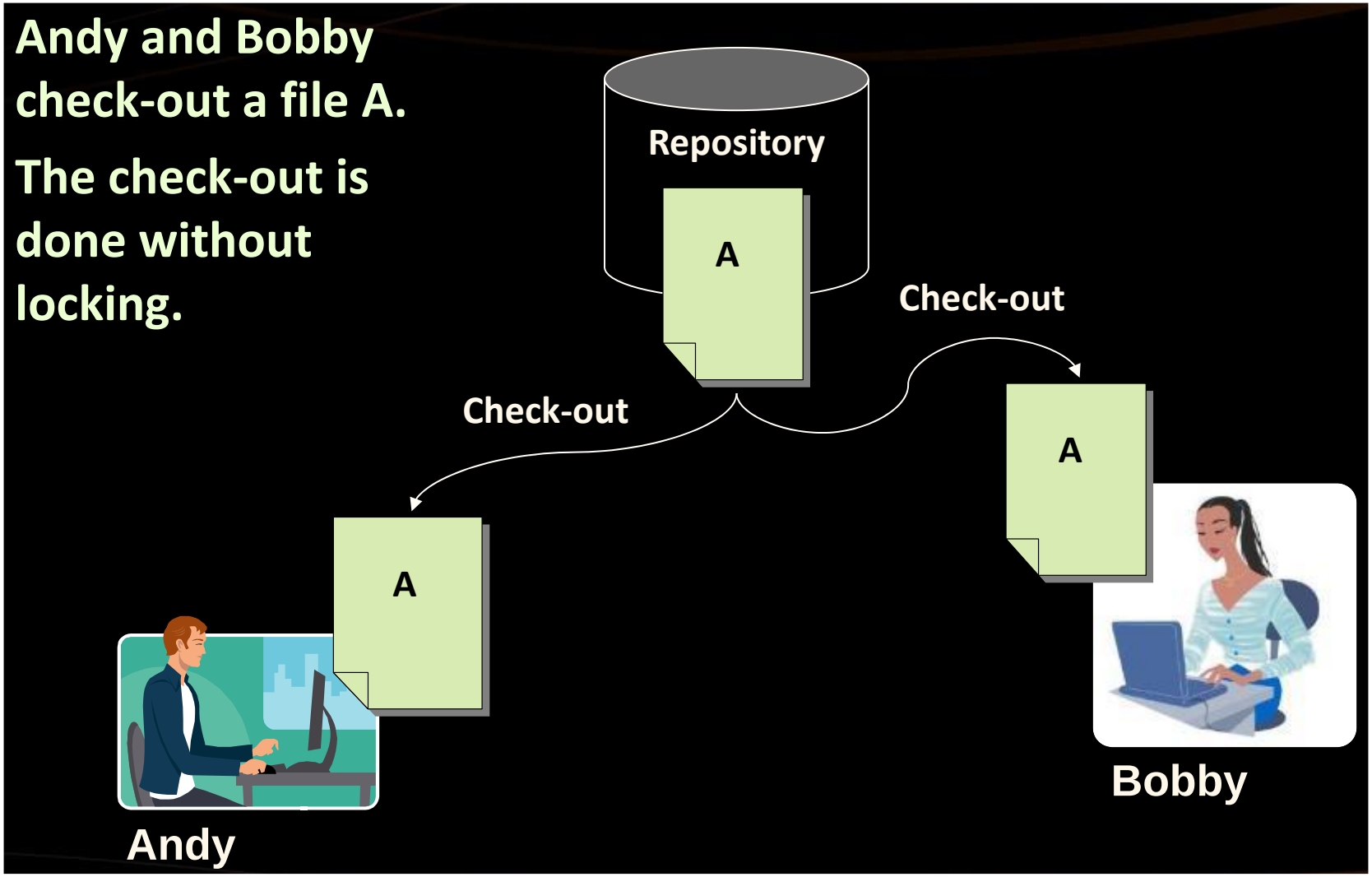
# 2.1. The Lock-Modify-Unlock Model (6)



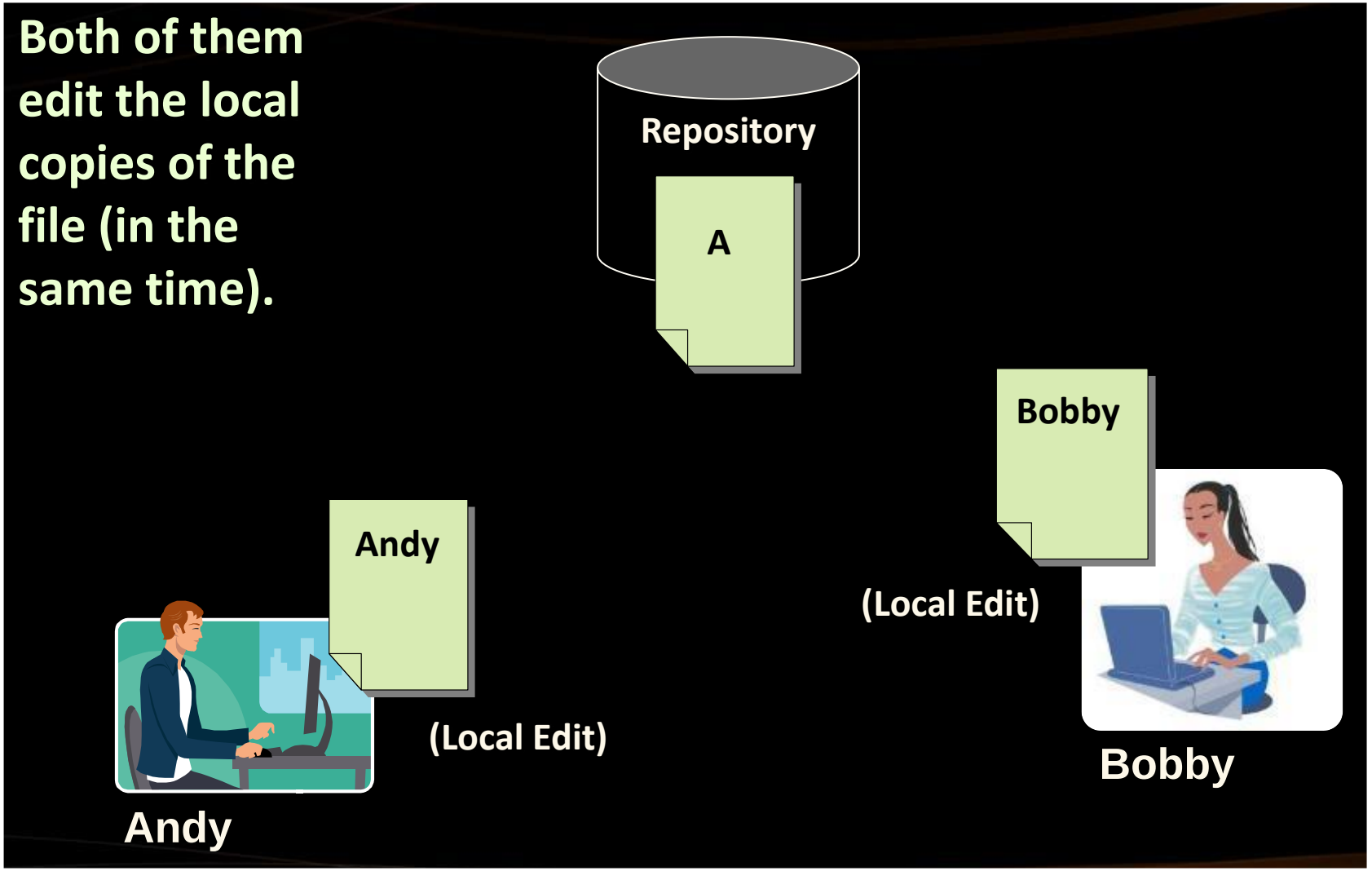
# 2.1. The Lock-Modify-Unlock Model (7)



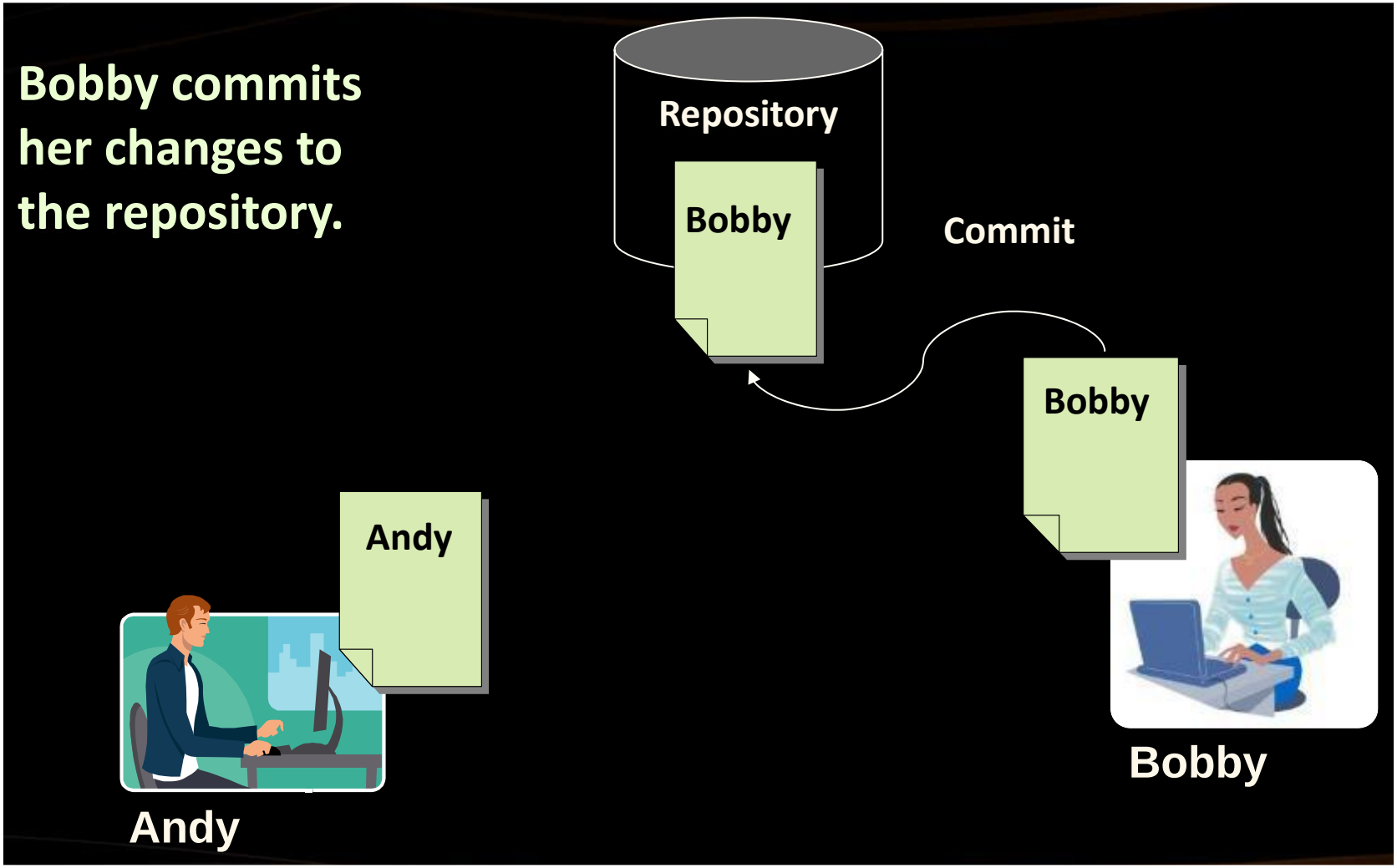
# 2.2. The Copy-Modify-Merge Model



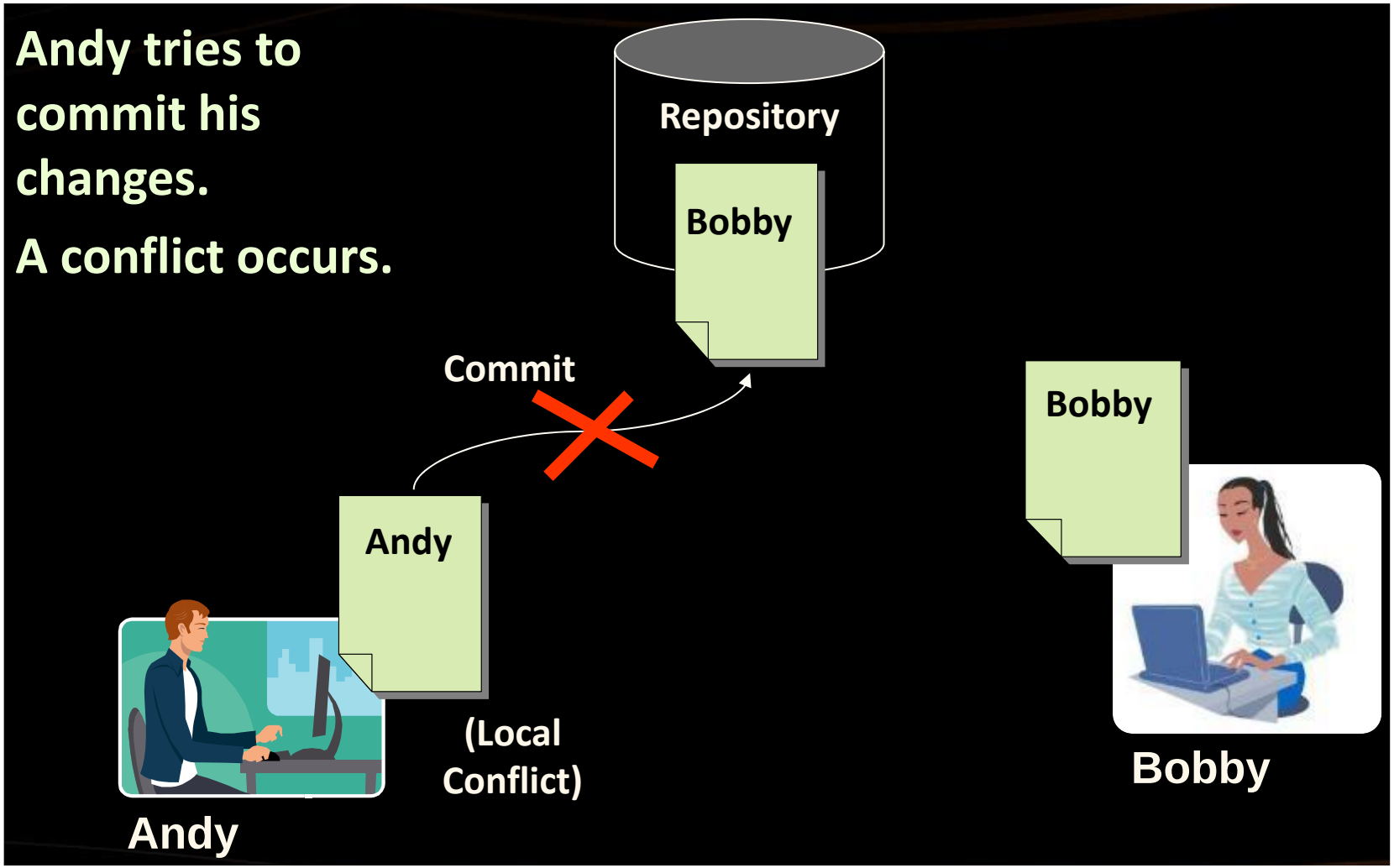
# 2.2. The Copy-Modify-Merge Model (2)



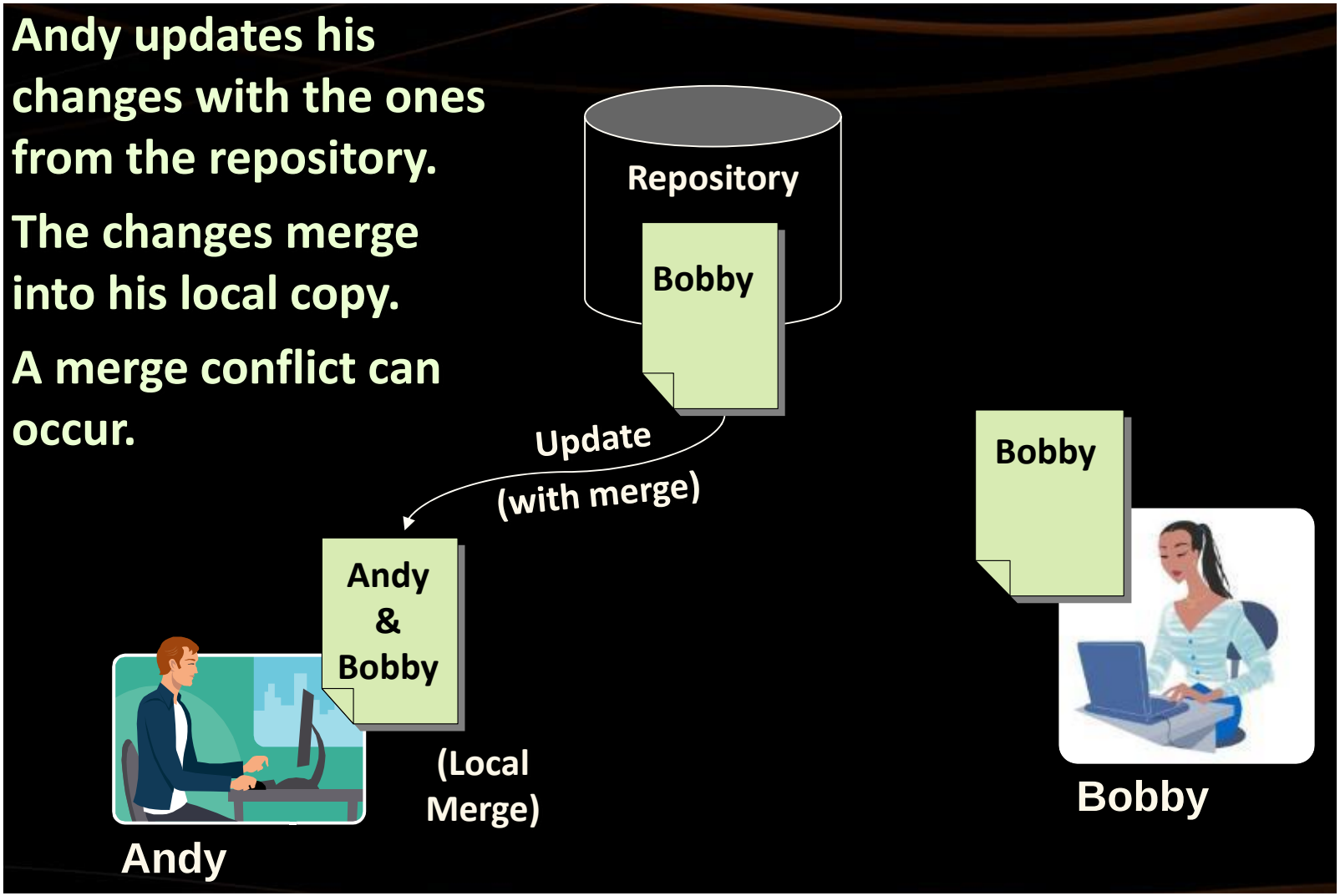
# 2.2. The Copy-Modify-Merge Model (3)



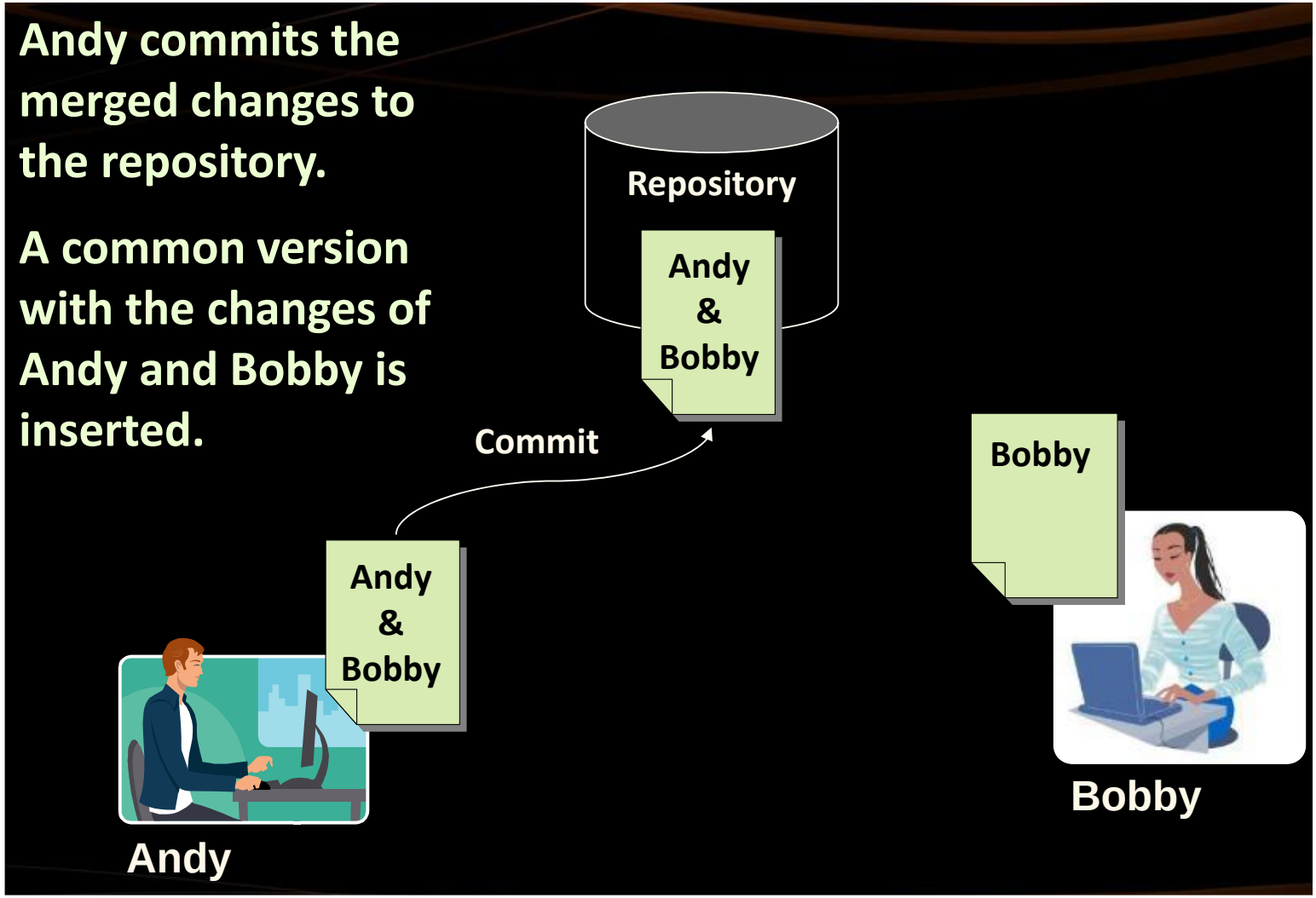
# 2.2. The Copy-Modify-Merge Model (4)



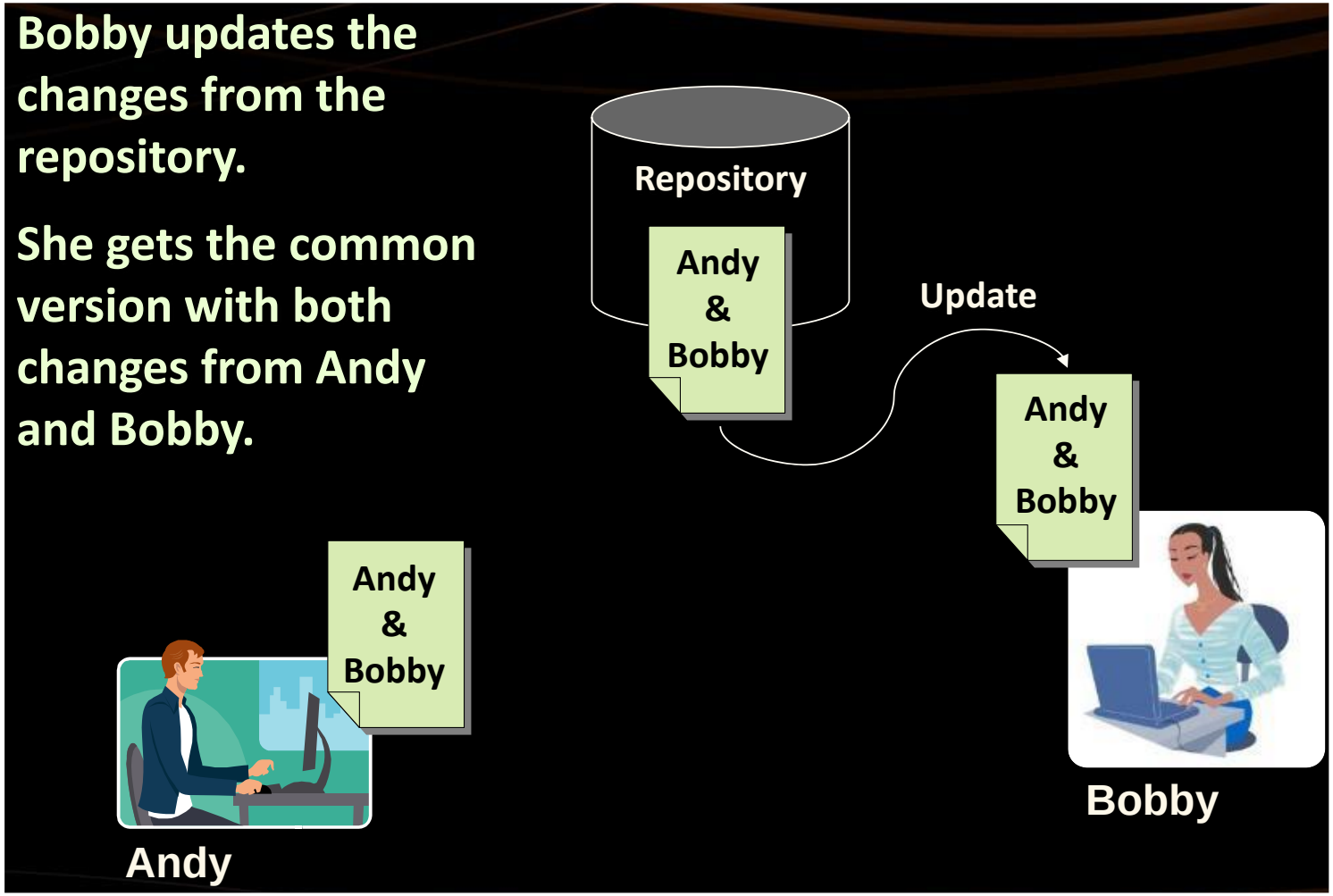
# 2.2. The Copy-Modify-Merge Model (5)



# 2.2. The Copy-Modify-Merge Model (6)

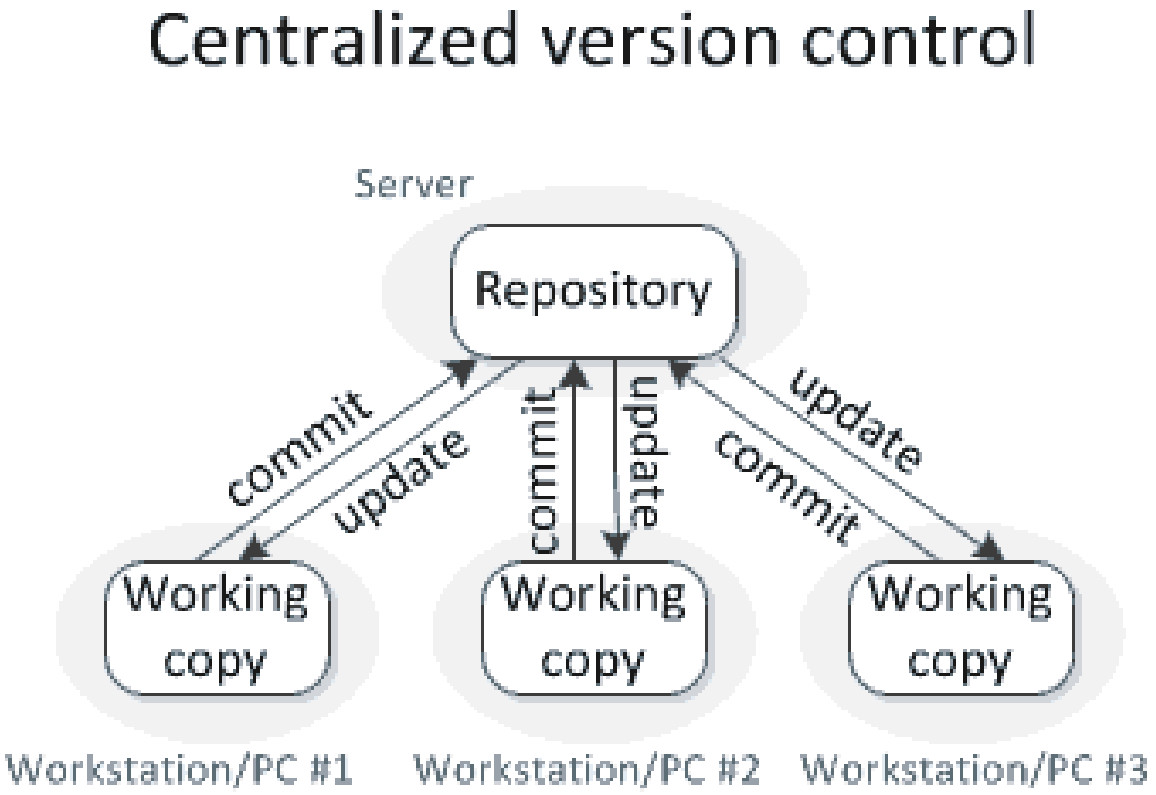


# 2.2. The Copy-Modify-Merge Model (7)

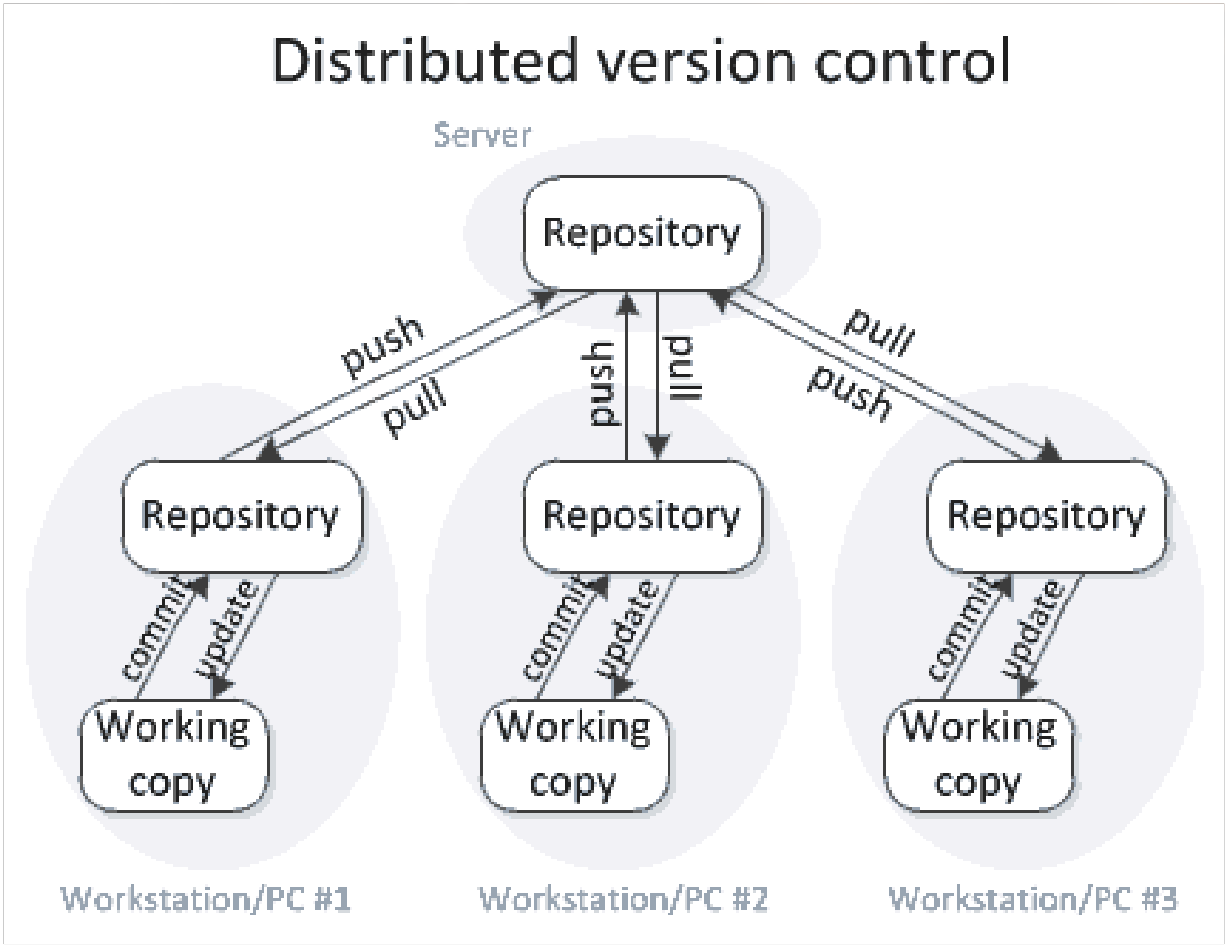


# 2.3. Distributed version control

- Compared to Central version control
  - Only one repository



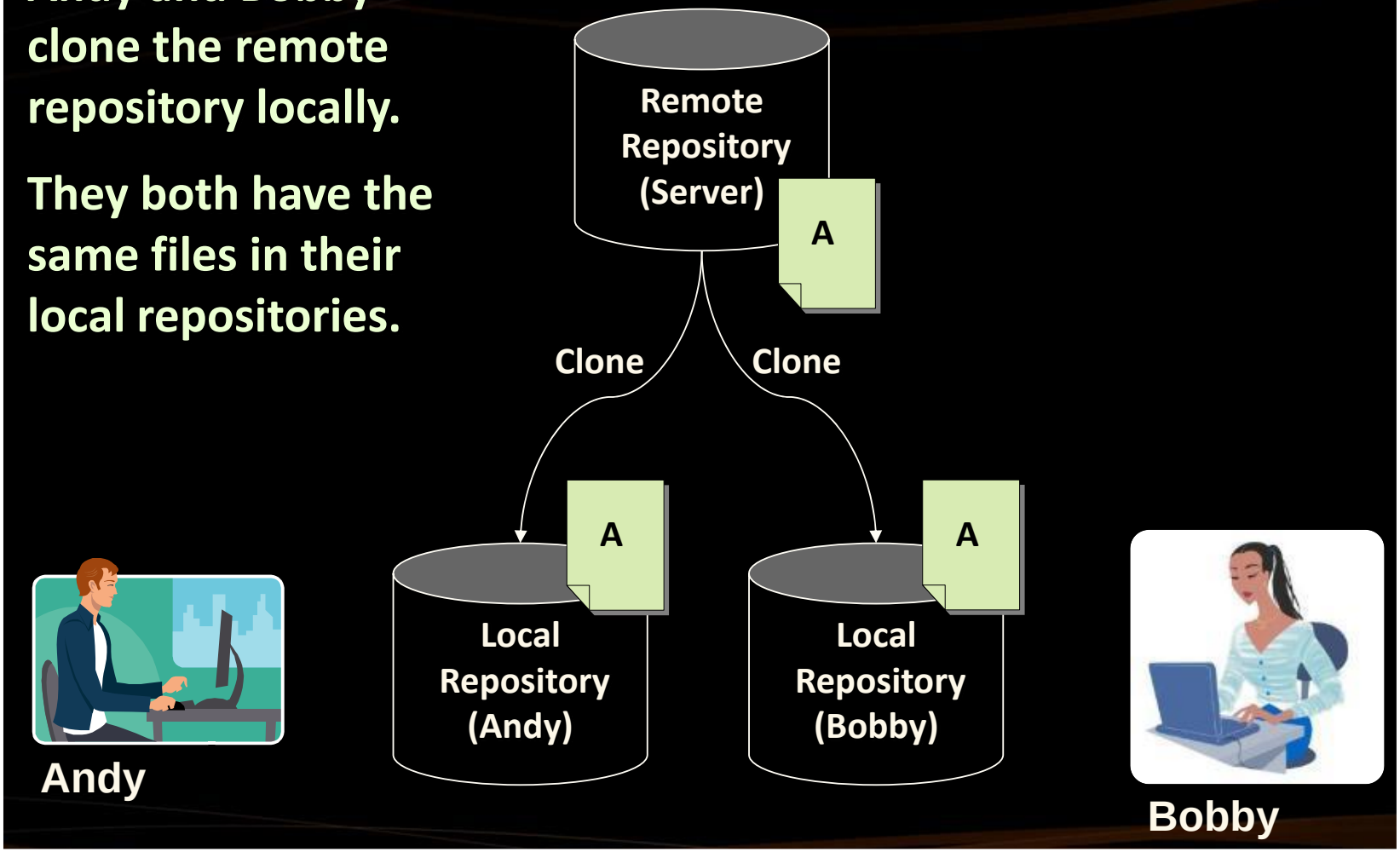
# Distributed Version Control



# Distributed Version Control (1)

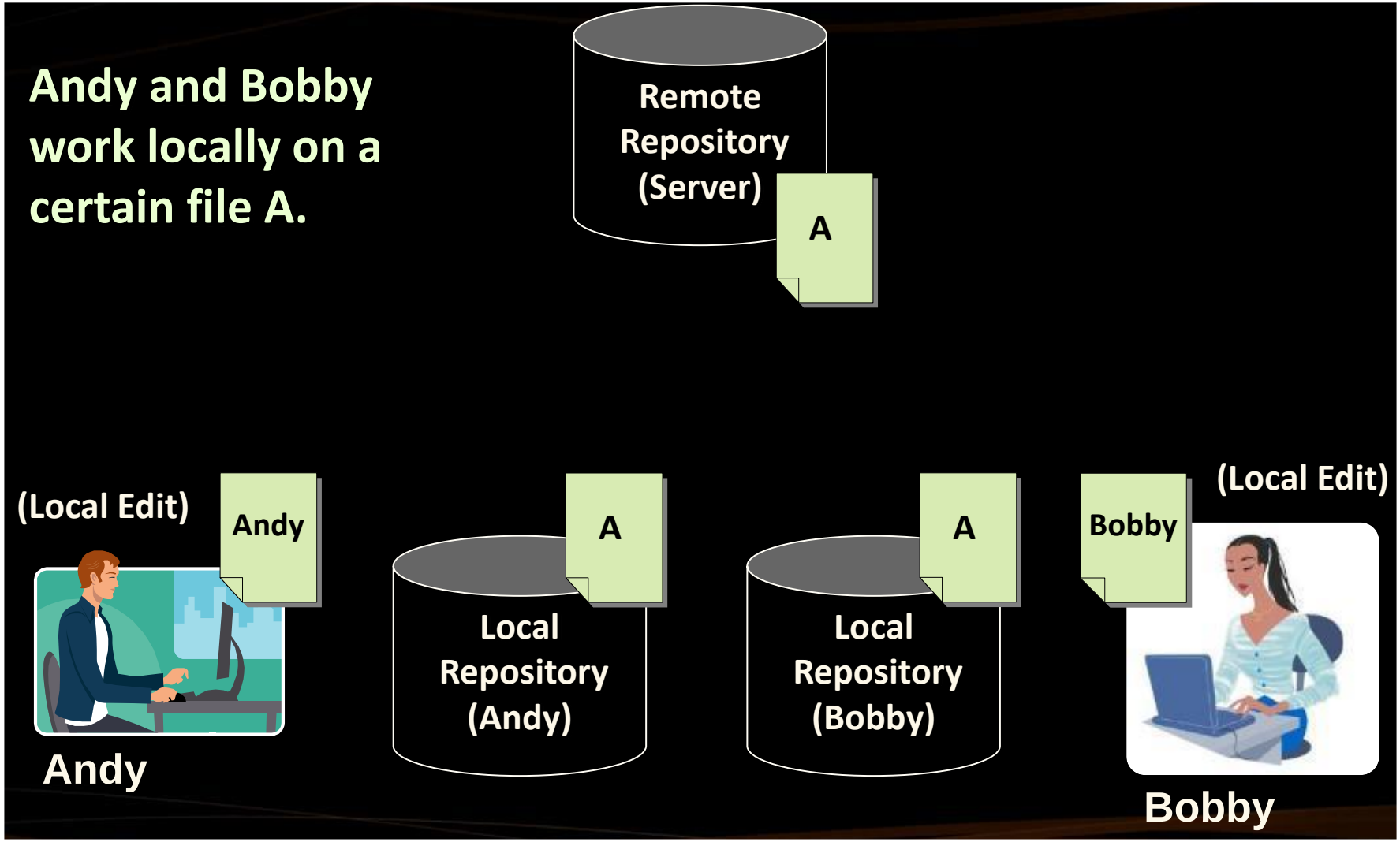
Andy and Bobby clone the remote repository locally.

They both have the same files in their local repositories.

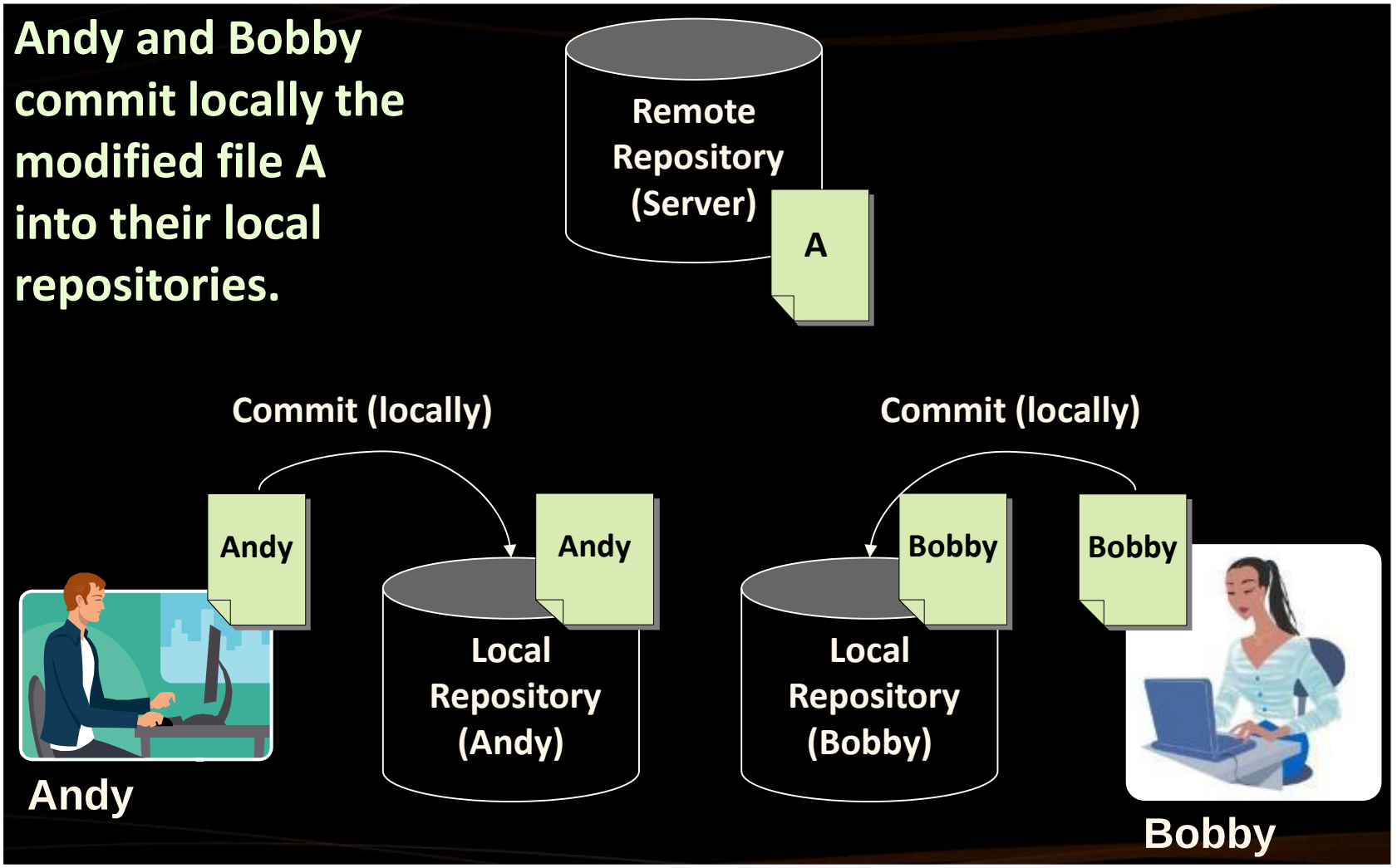


# Distributed Version Control (2)

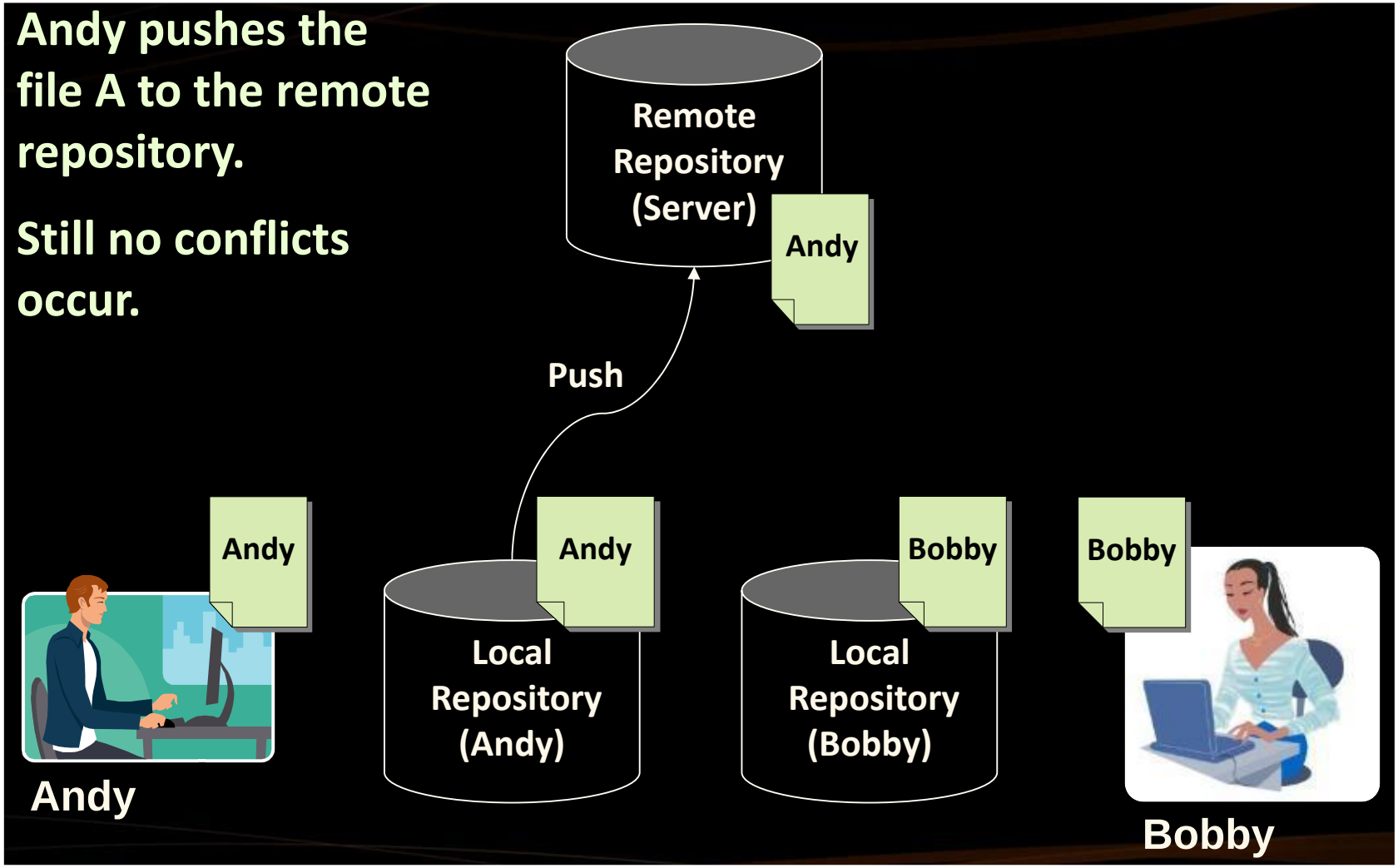
Andy and Bobby work locally on a certain file A.



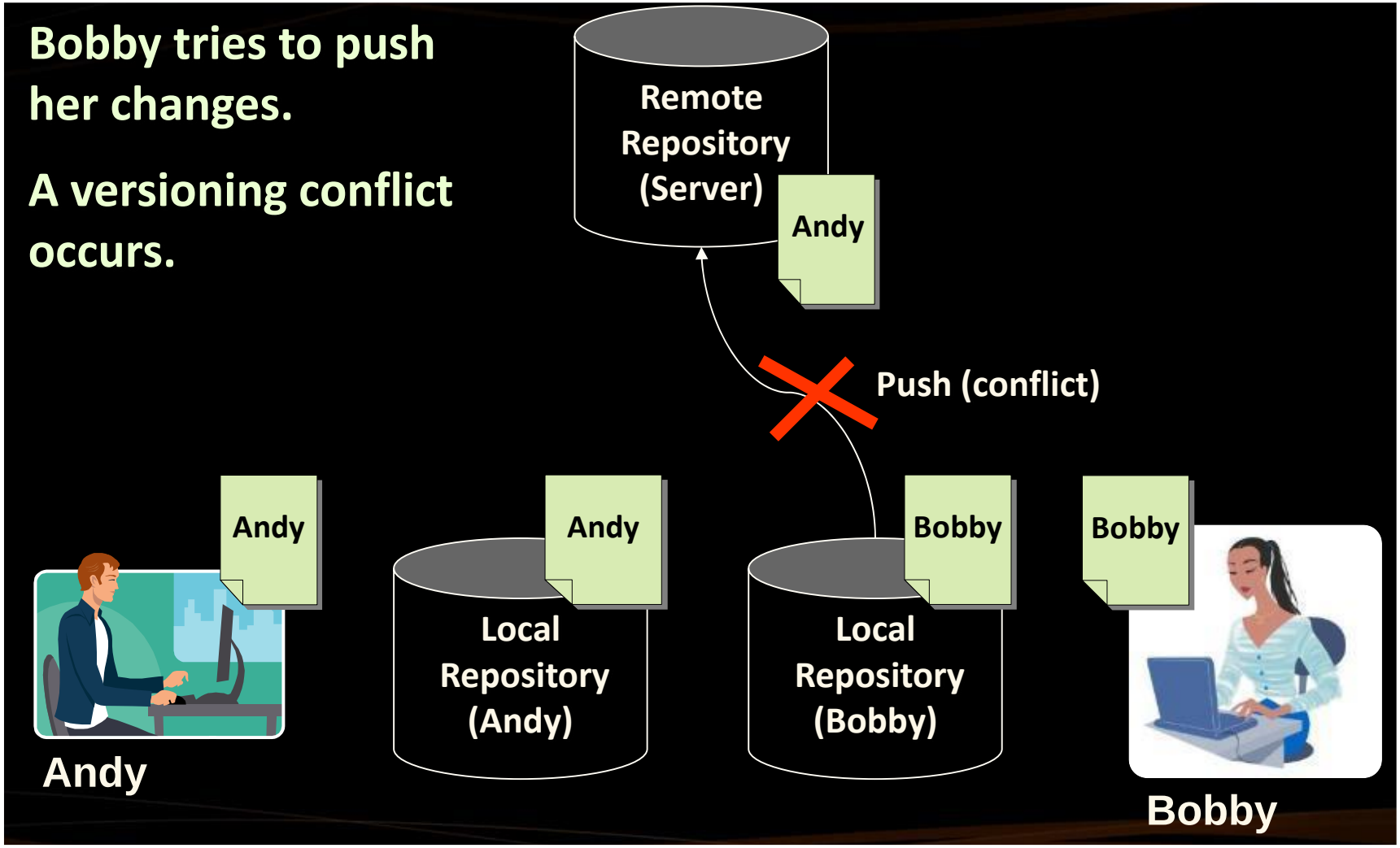
# Distributed Version Control (3)



# Distributed Version Control (4)



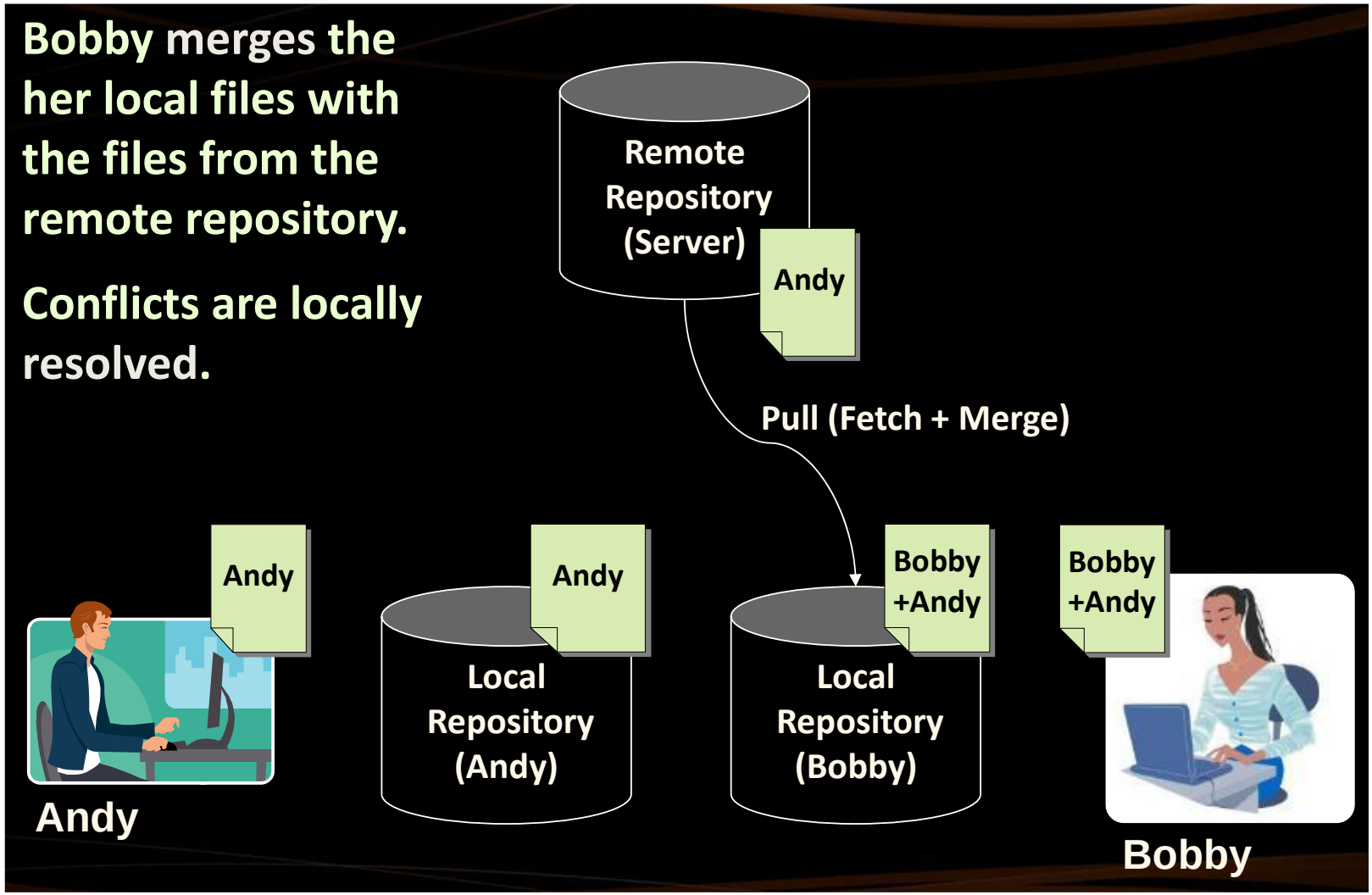
# Distributed Version Control (5)



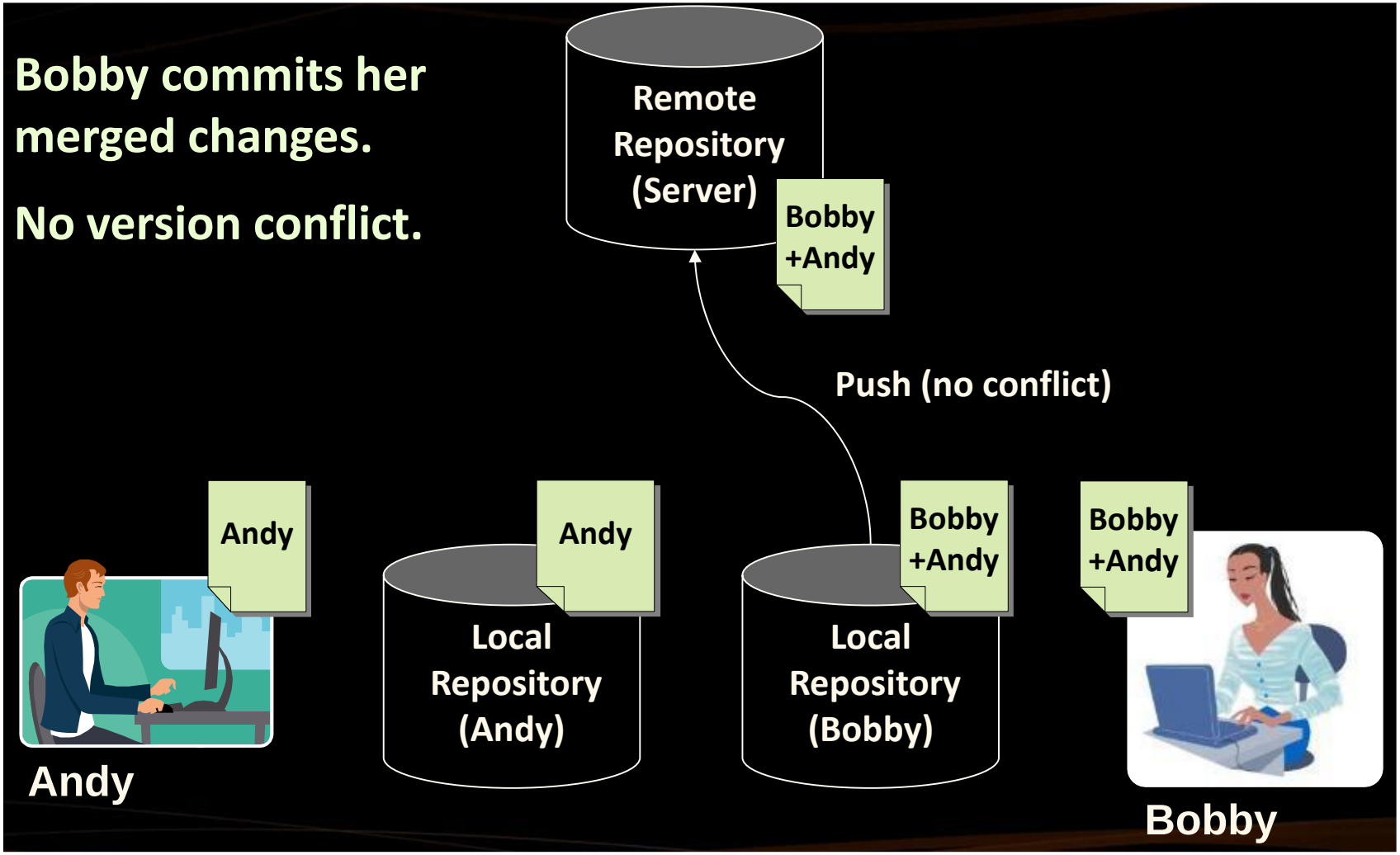
# Distributed Version Control (6)

Bobby merges the her local files with the files from the remote repository.

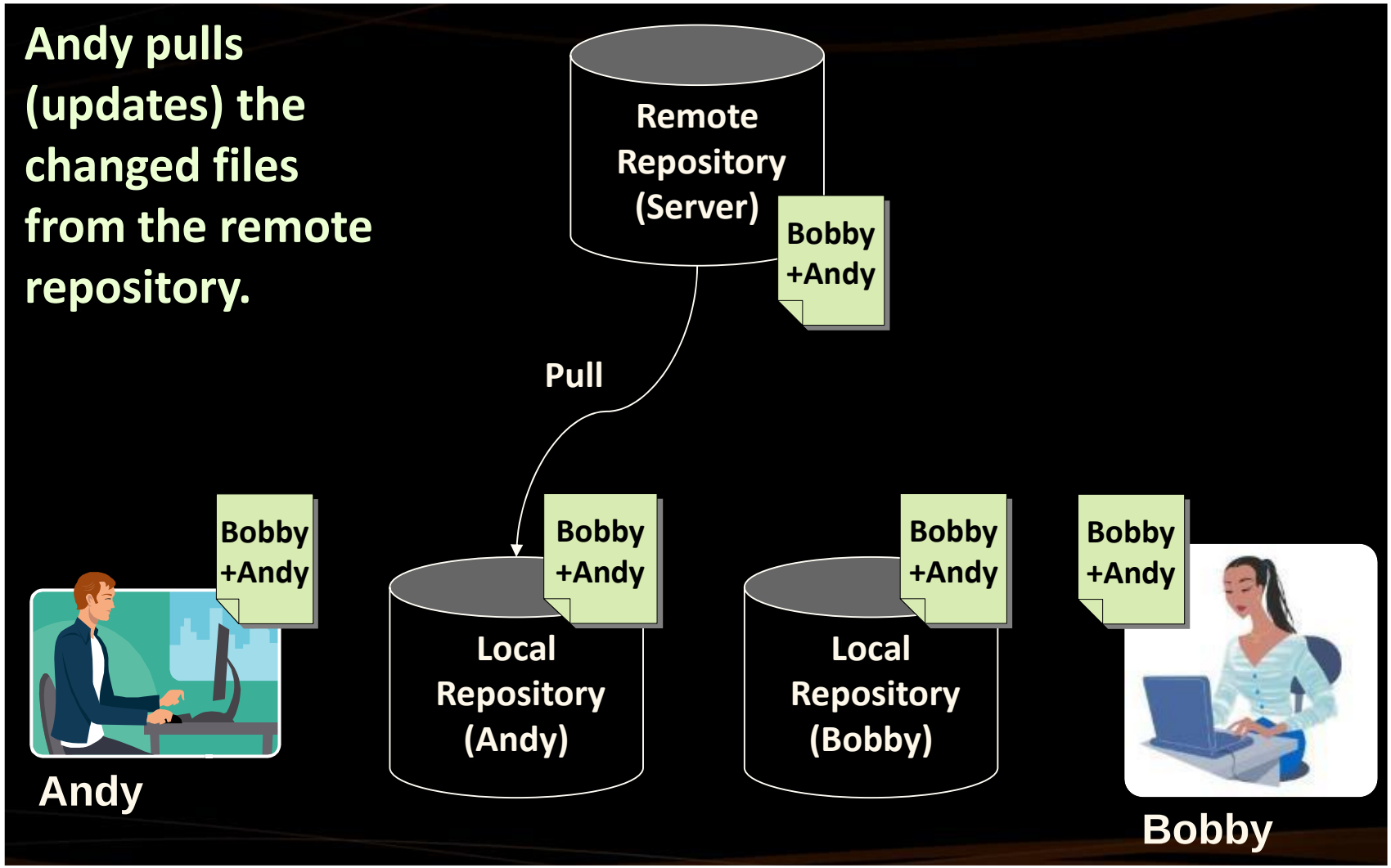
Conflicts are locally resolved.



# Distributed Version Control (7)



# Distributed Version Control (8)



# Outline

1. Introduction
2. Versioning Models
3. **Vocabulary**
4. Tools

# 3. Vocabulary

- Repository (source control repository)
  - A server that stores the files (documents)
  - Keeps a change log
- Revision, Version
  - Individual version (state) of a document that is a result of multiple changes
- Check-Out, Clone
  - Retrieves a working copy of the files from a remote repository into a local directory
  - It is possible to lock the files

# Vocabulary

- Change
  - A modification to a local file (document) that is under version control
- Change Set / Change List
  - A set of changes to multiple files that are going to be committed at the same time
- Commit, Check-In
  - Submits the changes made from the local working copy to the repository
  - Automatically creates a new version
  - Conflicts may occur!

# Vocabulary

- Conflict
  - The simultaneous change to a certain file by multiple users
  - Can be solved automatically and manually
- Update, Get Latest Version, Fetch / Pull
  - Download the latest version of the files from the repository to a local working directory + merge conflicting files
- Undo Check-Out, Revert / Undo Changes
  - Cancels the local changes
  - Restores their state from the repository

# Vocabulary

- Merge
  - Combines the changes to a file changed locally and simultaneously in the repository
  - Can be automated in most cases
- Label / Tag
  - Labels mark with a name a group of files in a given version
  - For example a release
- Branch / Branching
  - Division of the repositories in a number of separate workflows

# Outline

1. Introduction
2. Versioning Models
3. Vocabulary
4. **Tools**

# Tools

- Central version control
  - SVN (Subversion)
  - TFS
  - Source safe (commercial)
- Distributed version control
  - Git
  - Mercurial

# What is Git?

- Git
  - Distributed source-control system
  - Work with local and remote repositories
  - Git bash – command line interface for Git
  - Free, open-source
  - Has Windows version (msysGit)
    - http://msysgit.github.io
    - https://www.atlassian.com/git/tutorials/setting-up-a-repository

# Installing Git

- msysGit Installation on Windows
  - Download Git for Windows from: http://msysgit.github.io
  - “Next, Next, Next” does the trick
  - Options to select (they should be selected by default)
    - “Use Git Bash only”
    - “Checkout Windows-style, commit Unix-style endings”
- Git installation on Linux:

  sudo apt-get install git

# Basic Git Commands

- Cloning an existing Git repository

  git clone [remote url]

- Fetch and merge the latest changes from the remote repository

  git pull

- Preparing (adding / selecting) files for a commit

  git add [filename] ("git add ."adds everything)

- Committing to the local repository

  git commit –m "[your message here]"

# Basic Git Commands

- Check the status of your local repository (see the local changes)

  git status

- Creating a new local repository (in the current directory)

  git init

- Creating a remote (assign a short name for remote Git URL)

  git remote add [remote name] [remote url]

- Pushing to a remote (send changes to the remote repository)

  git push [remote name] [local name]

# Using Git: Example

```
mkdir work
cd work
git clone https://github.com/SoftUni/test.git dir
cd test
dir
git status
(edit some file)
git status
git add .
git commit -m "changes"
git push
```

# Project Hosting Sites

- GitHub – https://github.com
  - The #1 project hosting site in the world
  - Free for open-source projects
  - Paid plans for private projects
- GitHub provides own Windows client
  - GitHub for Windows
  - http://windows.github.com
  - Dramatically simplifies Git
  - For beginners only

# Project Hosting Sites

- Google Code – http://code.google.com/projecthosting/
  - Source control (SVN), file release, wiki, tracker
  - Very simple, basic functions only, not feature-rich
  - Free, all projects are public and open source
  - 1-minute signup, without heavy approval process
- SourceForge – http://www.sourceforge.net
  - Source control (SVN, Git, ...), web hosting, tracker, wiki, blog, mailing lists, file release, statistics, etc.
  - Free, all projects are public and open source

# Project Hosting Sites

- CodePlex – http://www.codeplex.com
  - Microsoft's open source projects site
  - Team Foundation Server (TFS) infrastructure
  - Source control (TFS), issue tracker, downloads, discussions, wiki, etc.
  - Free, all projects are public and open source
- Bitbucket – http://bitbucket.org
  - Source control (Mercurial), issue tracker, wiki, management tools
  - Private projects, free and paid editions