INTRODUCATION TO SOFTWARE ENGINEERING

5. SW CONFIGURATION MANAGEMENT

Bui Thi Mai Anh
anhbtm@soict.hust.edu.vn

Content

1. Changes and Software Configuration
2. Software Configuration Management
3. SCM Process
4. Version Control

Content

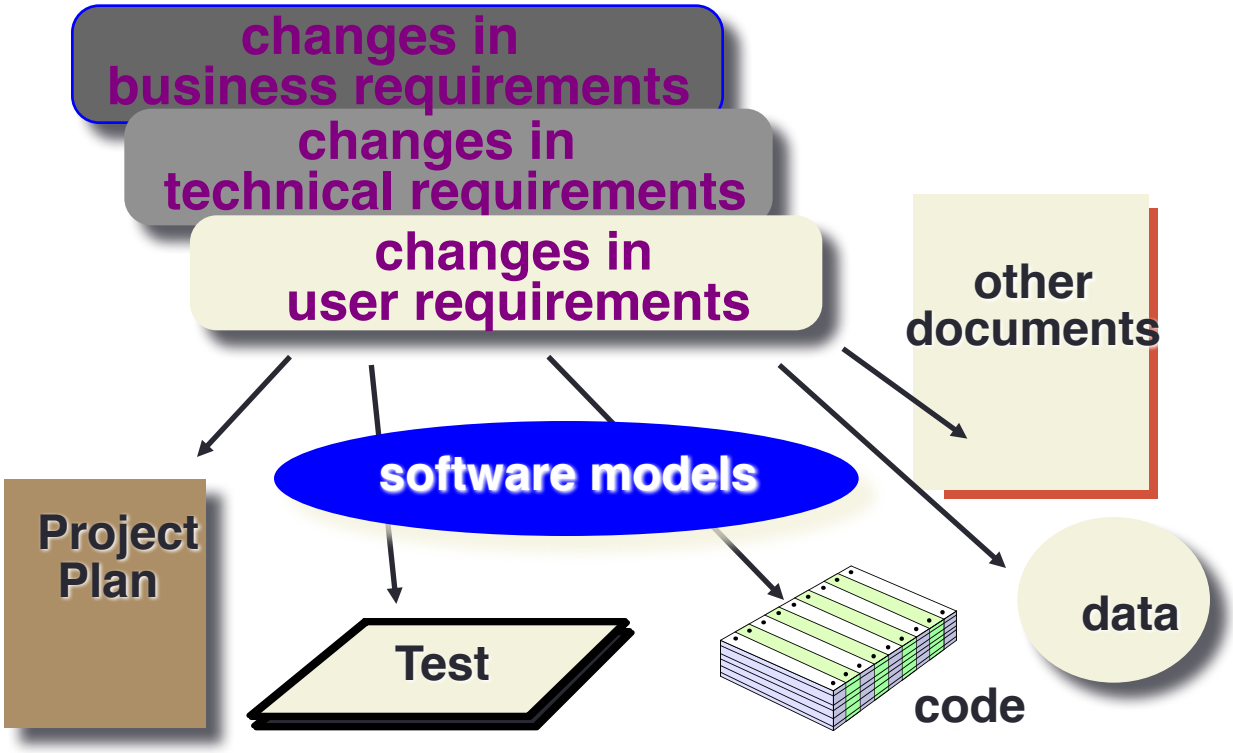
1. Changes and Software Configuration
2. Software Configuration Management
3. SCM Process
4. Version Control

The "First Law"

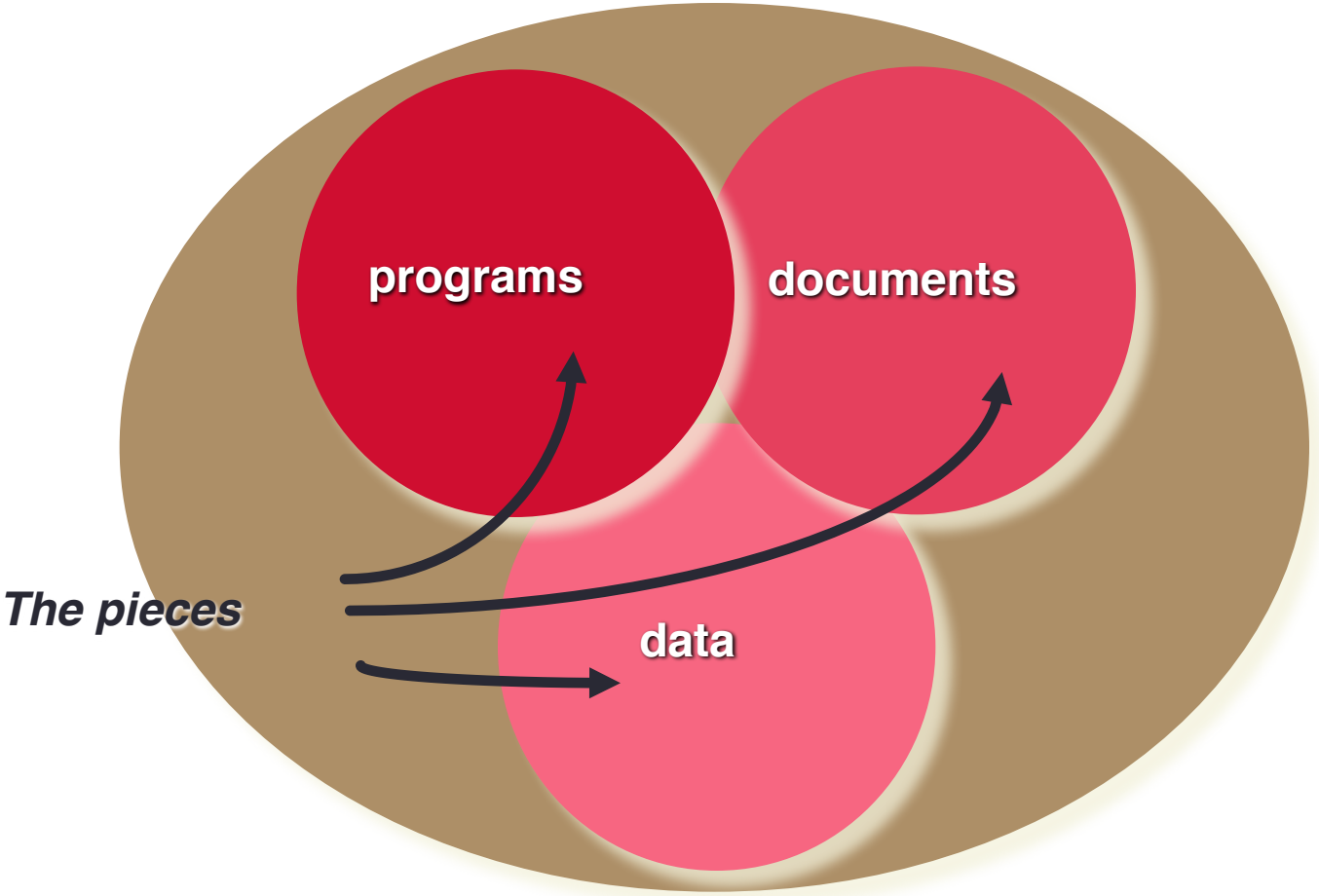
No matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle.

Bersoff, et al, 1980

What Are These Changes?



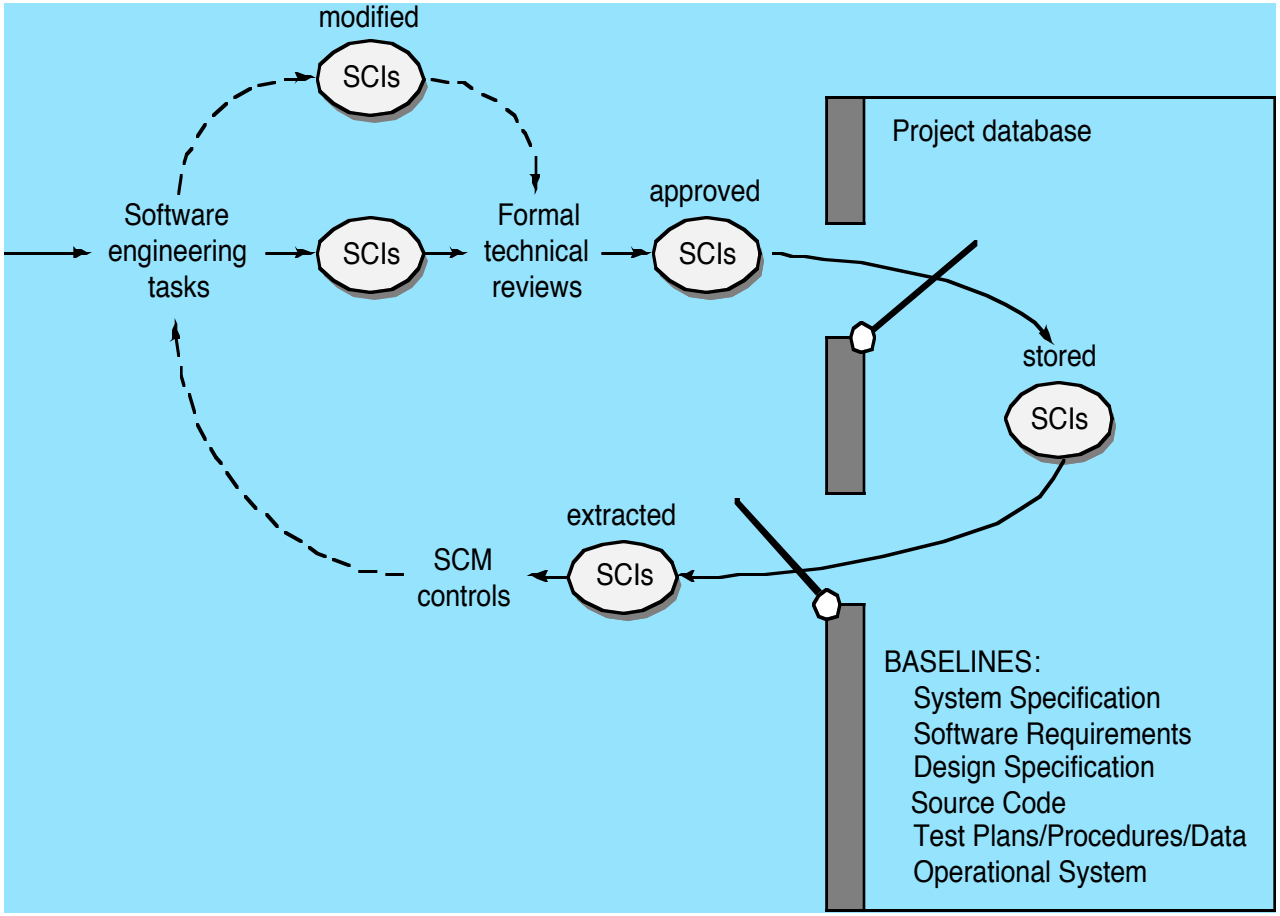
The Software Configuration



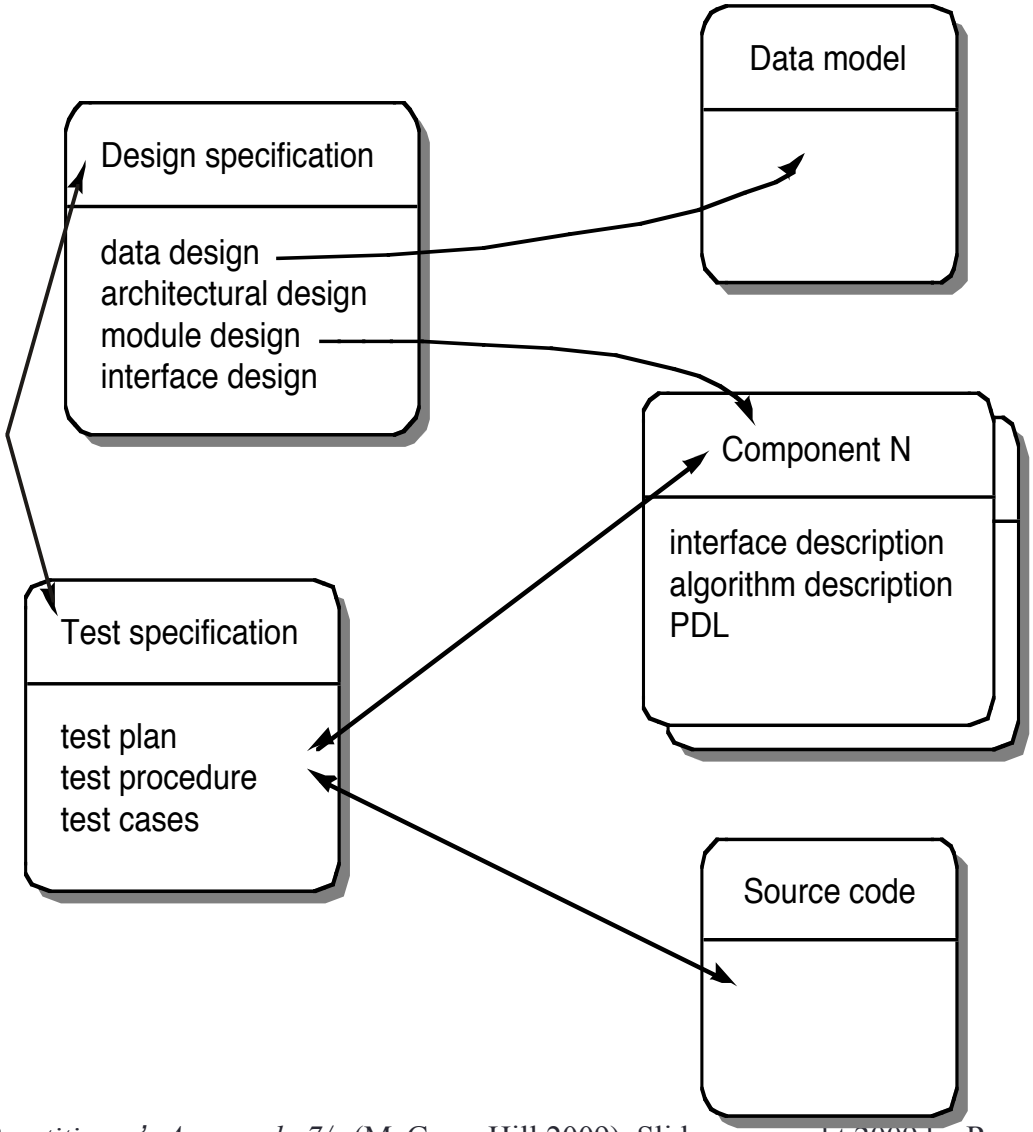
Baselines

- The IEEE (IEEE Std. No. 610.12-1990) defines a baseline as:
  - A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures.
- a baseline is a milestone in the development of software that is marked by the delivery of one or more software configuration items and the approval of these SCIs that is obtained through a formal technical review

Baselines


modified
SCIs
Project database
Software engineering tasks
SCIs
Formal technical reviews
approved
SCIs
stored
SCIs
extracted
SCIs
SCM controls
BASELINES:
System Specification
Software Requirements
Design Specification
Source Code
Test Plans/Procedures/Data
Operational System

Software Configuration Objects



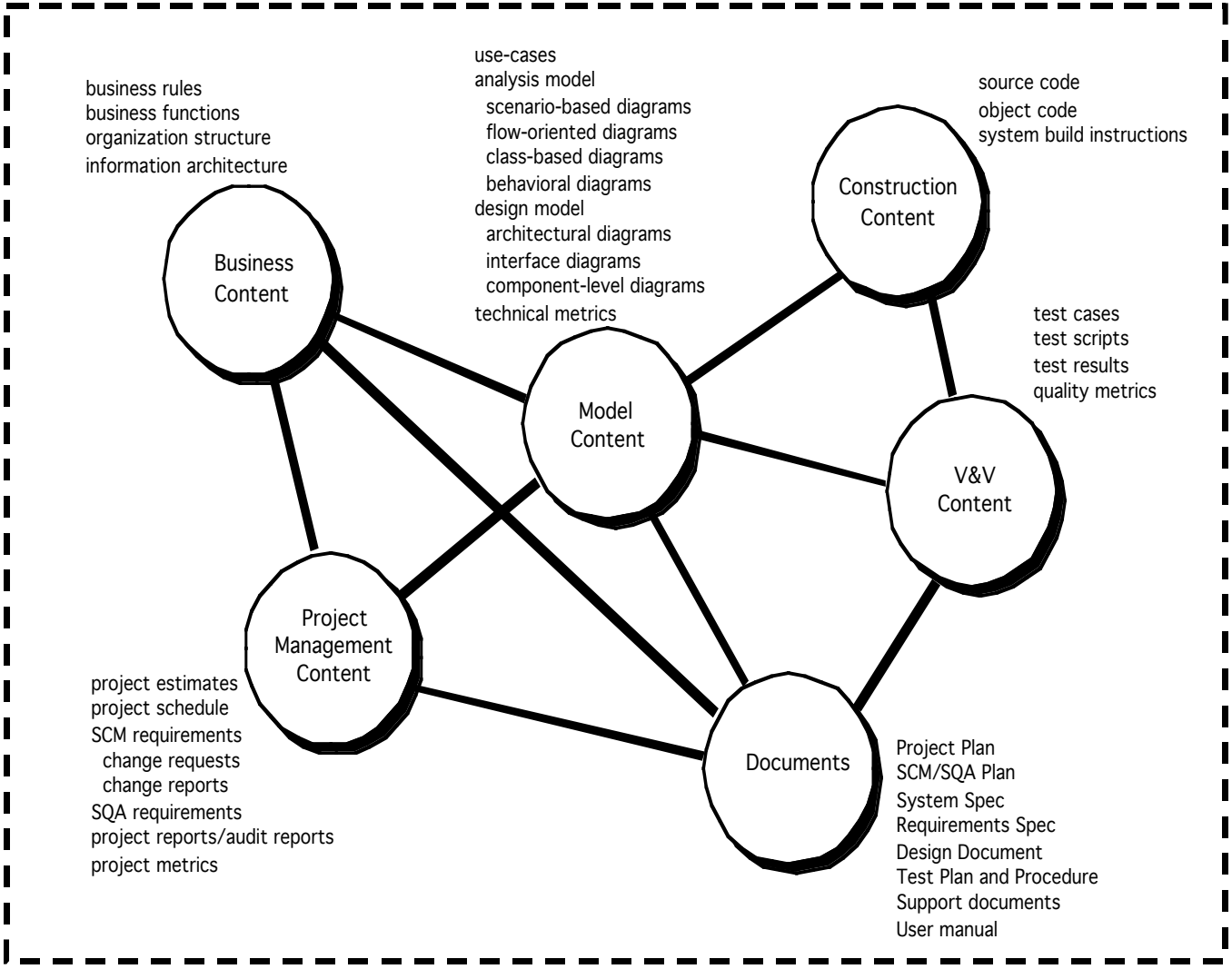
Content

1. Changes and Software Configuration
2. Software Configuration Management
3. SCM Process
4. Version Control

SCM Repository

- The SCM repository is the set of mechanisms and data structures that allow a software team to manage change in an effective manner
- The repository performs or precipitates the following functions [For89]:
  - Data integrity
  - Information sharing
  - Tool integration
  - Data integration
  - Methodology enforcement
  - Document standardization

Repository Content



Repository Features

- **Versioning.**
  - saves all of these versions to enable effective management of product releases and to permit developers to go back to previous versions
- **Dependency tracking and change management.**
  - The repository manages a wide variety of relationships among the data elements stored in it.
- **Requirements tracing.**
  - Provides the ability to track all the design and construction components and deliverables that result from a specific requirement specification
- **Configuration management.**
  - Keeps track of a series of configurations representing specific project milestones or production releases. Version management provides the needed versions, and link management keeps track of interdependencies.
- **Audit trails.**
  - establishes additional information about when, why, and by whom changes are made.

SCM Elements

- *Component elements*—a set of tools coupled within a file management system (e.g., a database) that enables access to and management of each software configuration item.
- *Process elements*—a collection of procedures and tasks that define an effective approach to change management (and related activities) for all constituencies involved in the management, engineering and use of computer software.
- *Construction elements*—a set of tools that automate the construction of software by ensuring that the proper set of validated components (i.e., the correct version) have been assembled.
- *Human elements*—to implement effective SCM, the software team uses a set of tools and process features (encompassing other CM elements)

Content

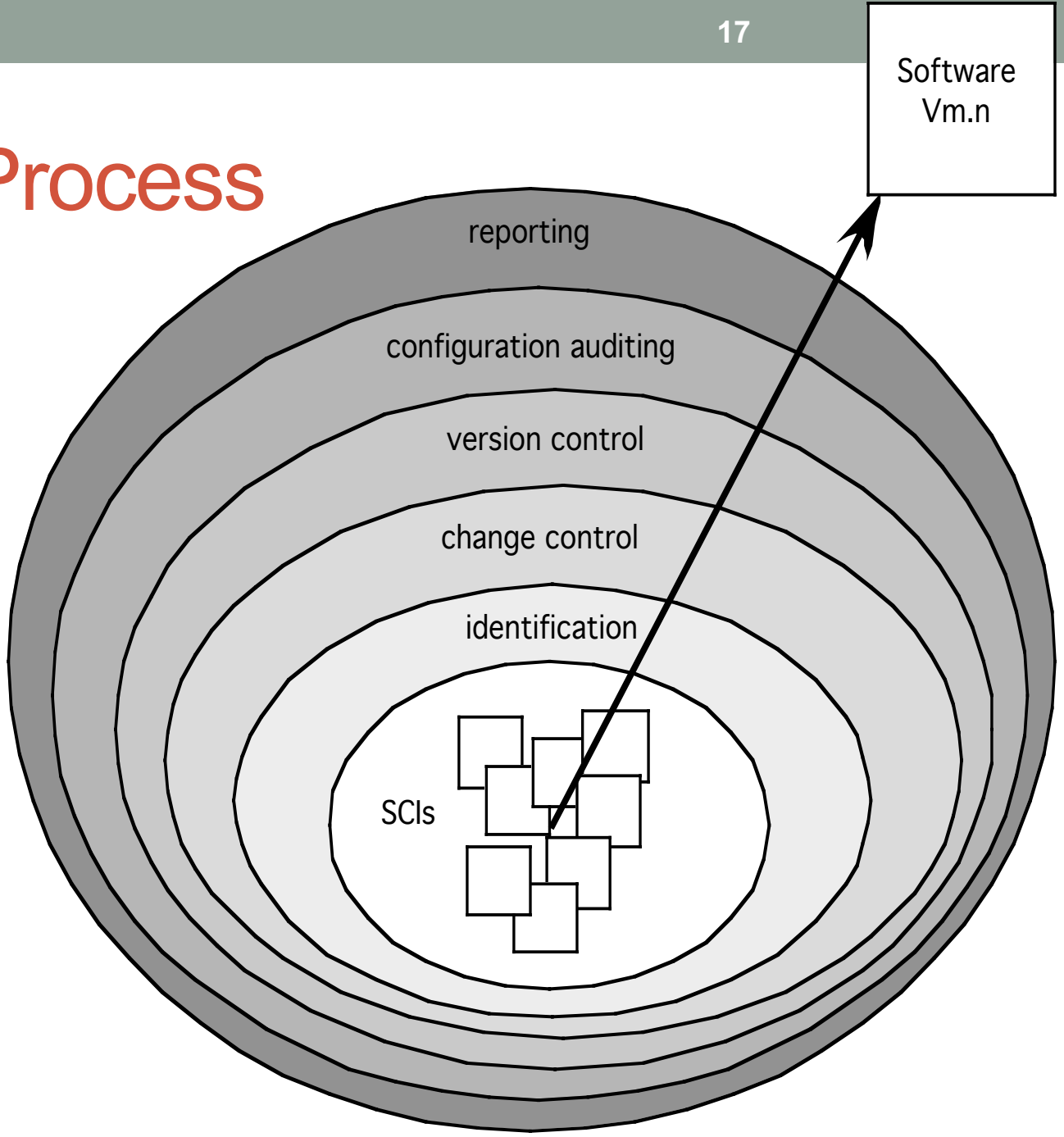
1. Changes and Software Configuration
2. Software Configuration Management
3. SCM Process
4. Version Control

The SCM Process

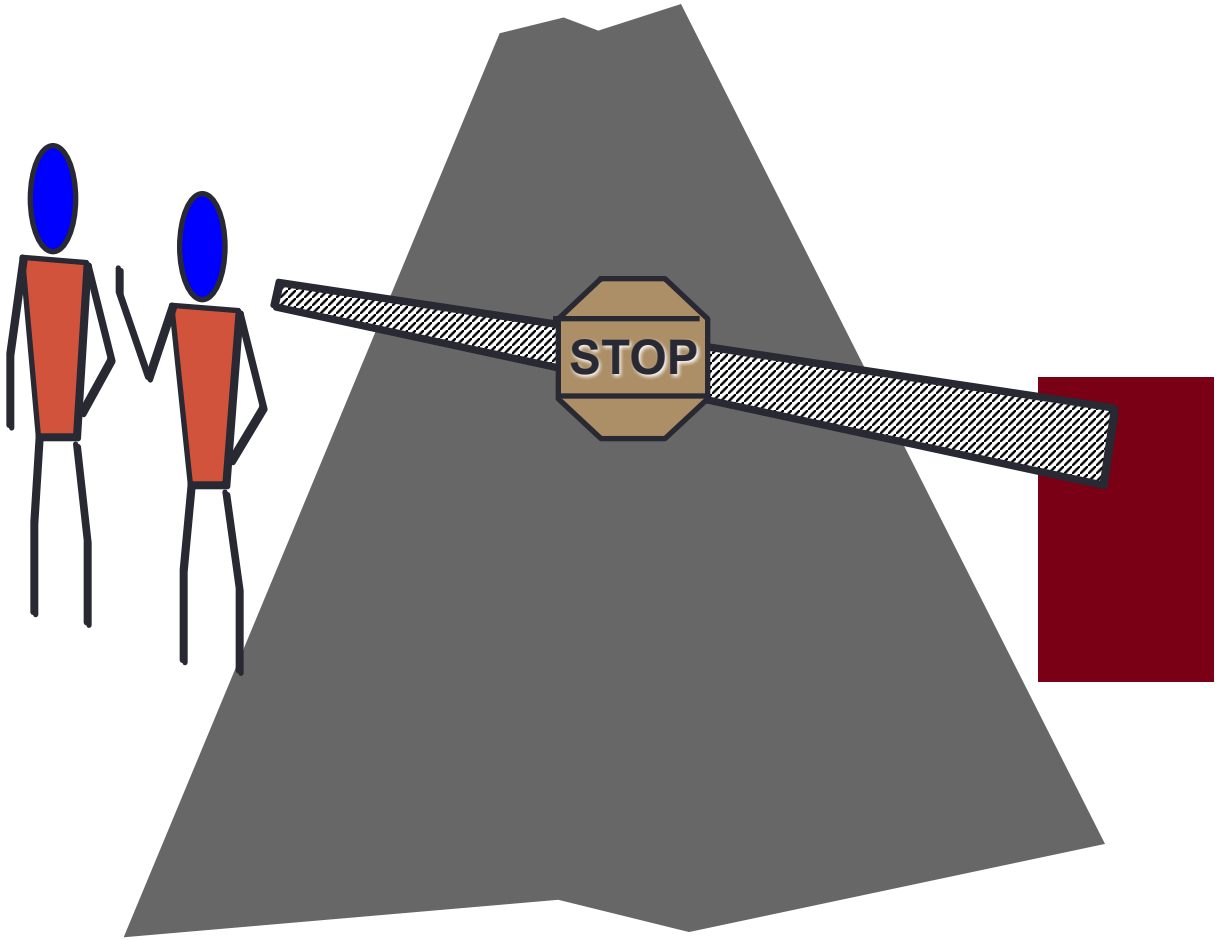
Addresses the following questions …

- How does a software team identify the discrete elements of a software configuration?
- How does an organization manage the many existing versions of a program (and its documentation) in a manner that will enable change to be accommodated efficiently?
- How does an organization control changes before and after software is released to a customer?
- Who has responsibility for approving and ranking changes?
- How can we ensure that changes have been made properly?
- What mechanism is used to appraise others of changes that are made?

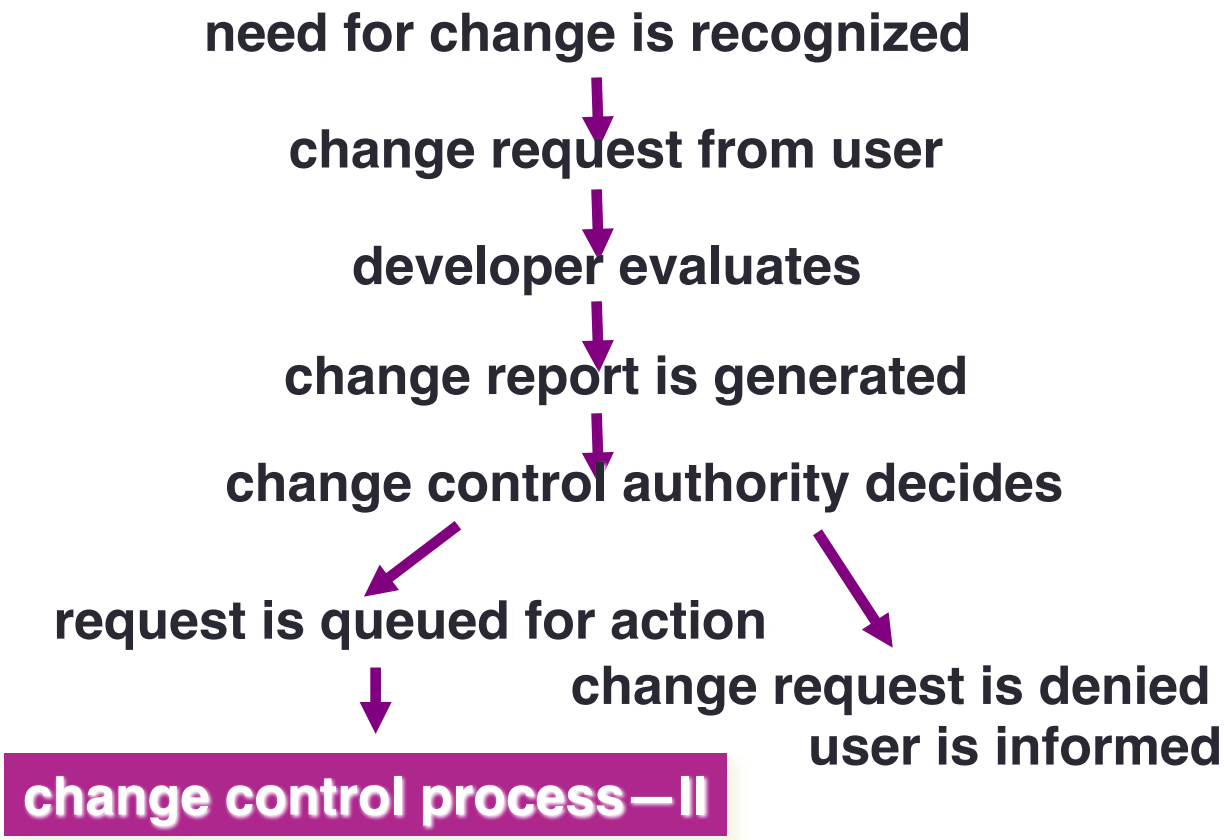
The SCM Process



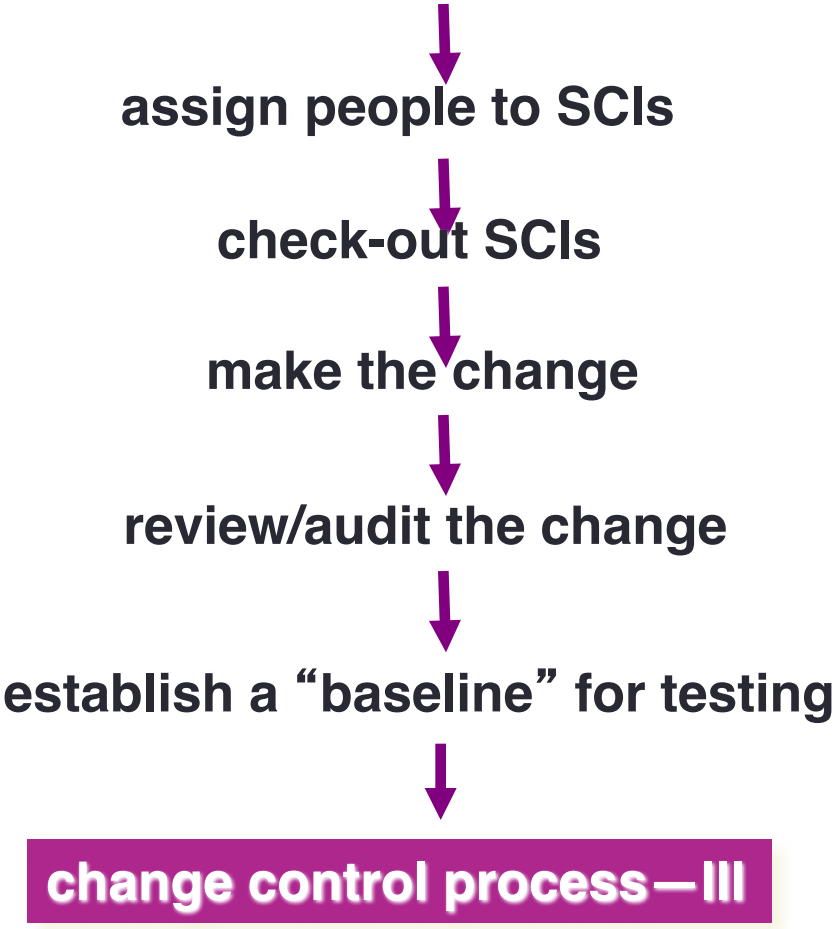
Change Control



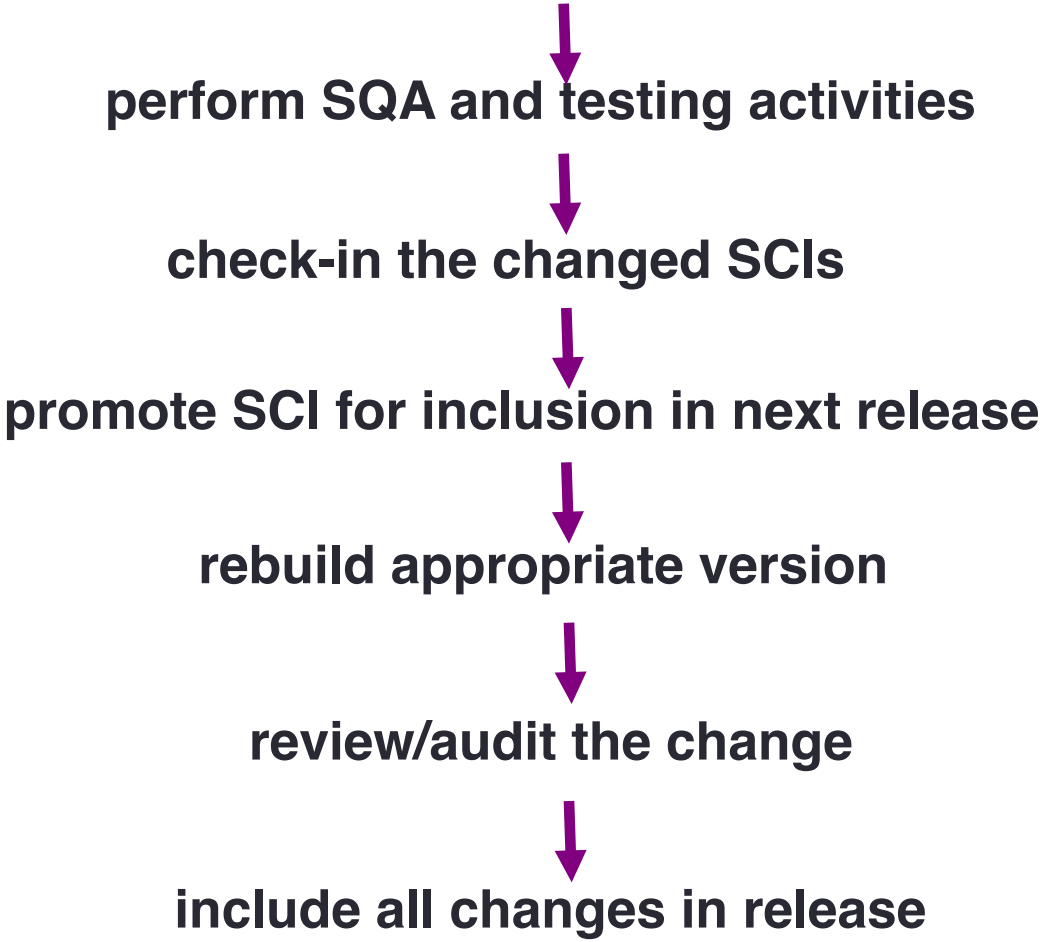
Change Control Process—I



Change Control Process-II



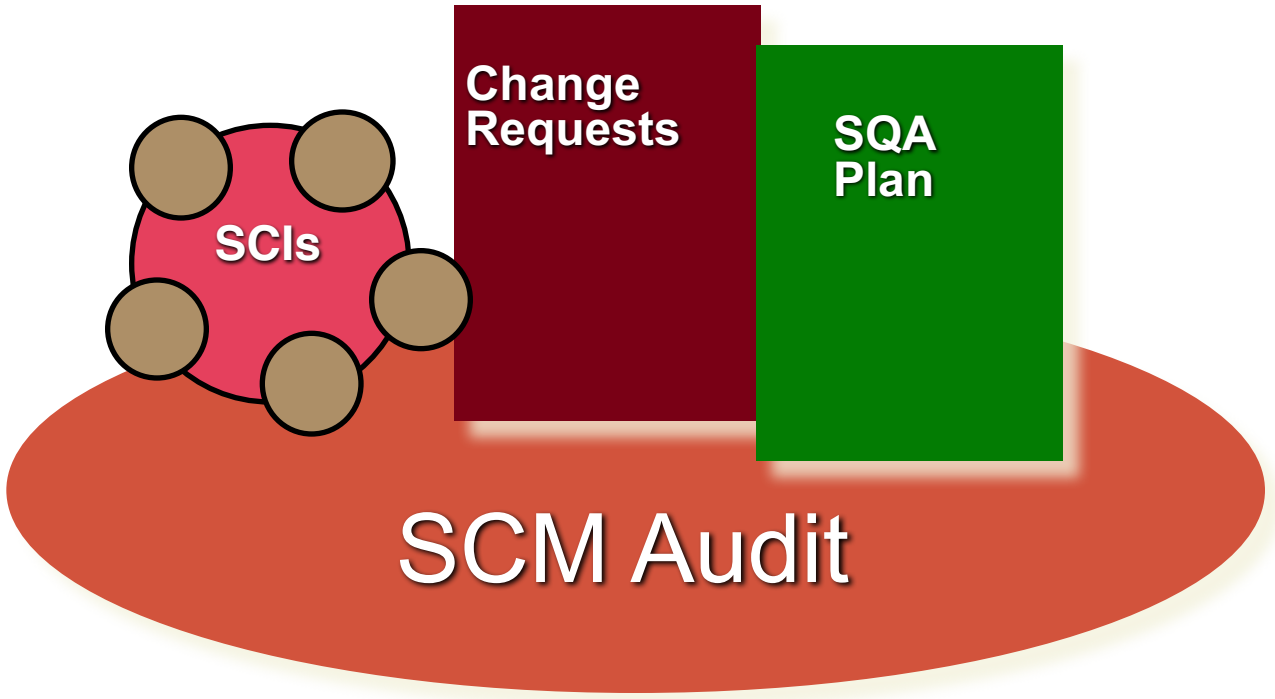
Change Control Process-III



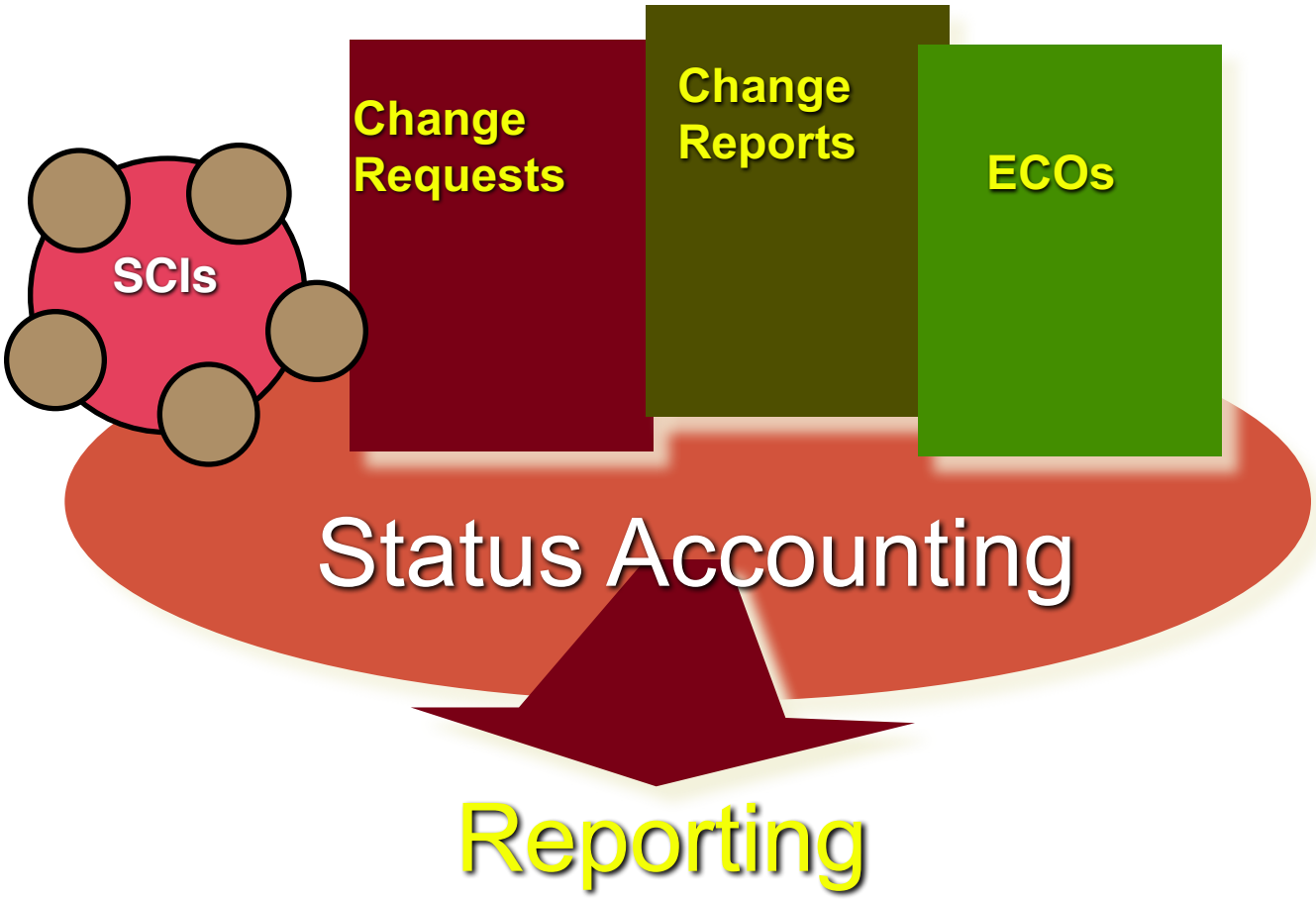
Version Control

- Version control combines procedures and tools to manage different versions of configuration objects that are created during the software process
- A version control system implements or is directly integrated with four major capabilities:
  - a *project database (repository)* that stores all relevant configuration objects
  - a *version management* capability that stores all versions of a configuration object (or enables any version to be constructed using differences from past versions);
  - a *make facility* that enables the software engineer to collect all relevant configuration objects and construct a specific version of the software.
  - an *issues tracking* (also called *bug tracking*) capability that enables the team to record and track the status of all outstanding issues associated with each configuration object.

Configuration Auditing



Status Accounting



SCM for Web Engineering-I

- **Content**
  - A typical WebApp contains a vast array of content—text, graphics, applets, scripts, audio/video files, forms, active page elements, tables, streaming data, and many others.
  - The challenge is to organize this sea of content into a rational set of configuration objects (Section 27.1.4) and then establish appropriate configuration control mechanisms for these objects.
- **People**
  - Because a significant percentage of WebApp development continues to be conducted in an ad hoc manner, any person involved in the WebApp can (and often does) create content.

SCM for Web Engineering-II

- **Scalability**
  - As size and complexity grow, small changes can have far-reaching and unintended affects that can be problematic. Therefore, the rigor of configuration control mechanisms should be directly proportional to application scale.
- **Politics**
  - Who ‘owns’ a WebApp?
  - Who assumes responsibility for the accuracy of the information on the Web site?
  - Who assures that quality control processes have been followed before information is published to the site?
  - Who is responsible for making changes?
  - Who assumes the cost of change?

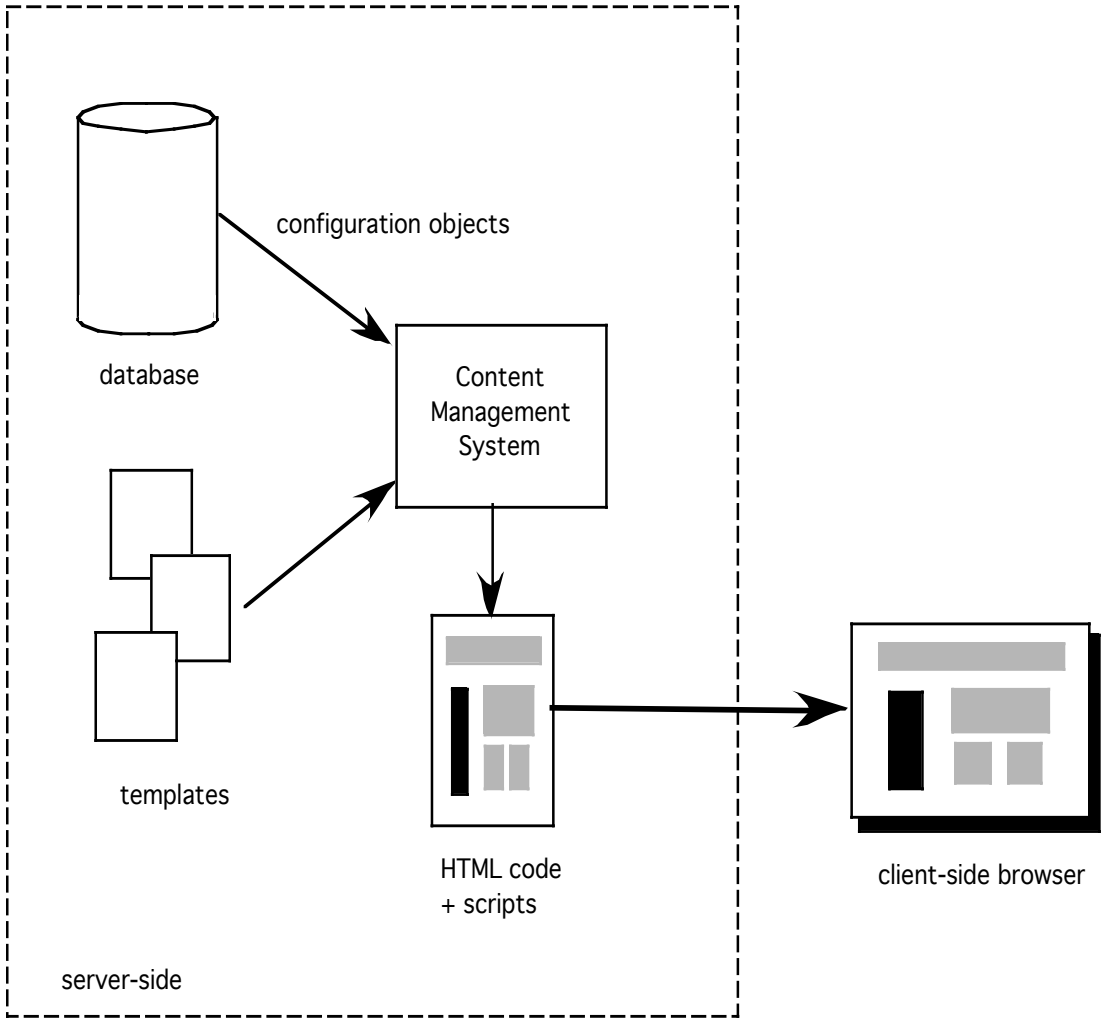
Content Management-I

- **The collection subsystem** encompasses all actions required to create and/or acquire content, and the technical functions that are necessary to
  - convert content into a form that can be represented by a mark-up language (e.g., HTML, XML
  - organize content into packets that can be displayed effectively on the client-side.
- **The management subsystem** implements a repository that encompasses the following elements:
  - *Content database*—the information structure that has been established to store all content objects
  - *Database capabilities*—functions that enable the CMS to search for specific content objects (or categories of objects), store and retrieve objects, and manage the file structure that has been established for the content
  - *Configuration management functions*—the functional elements and associated workflow that support content object identification, version control, change management, change auditing, and reporting.

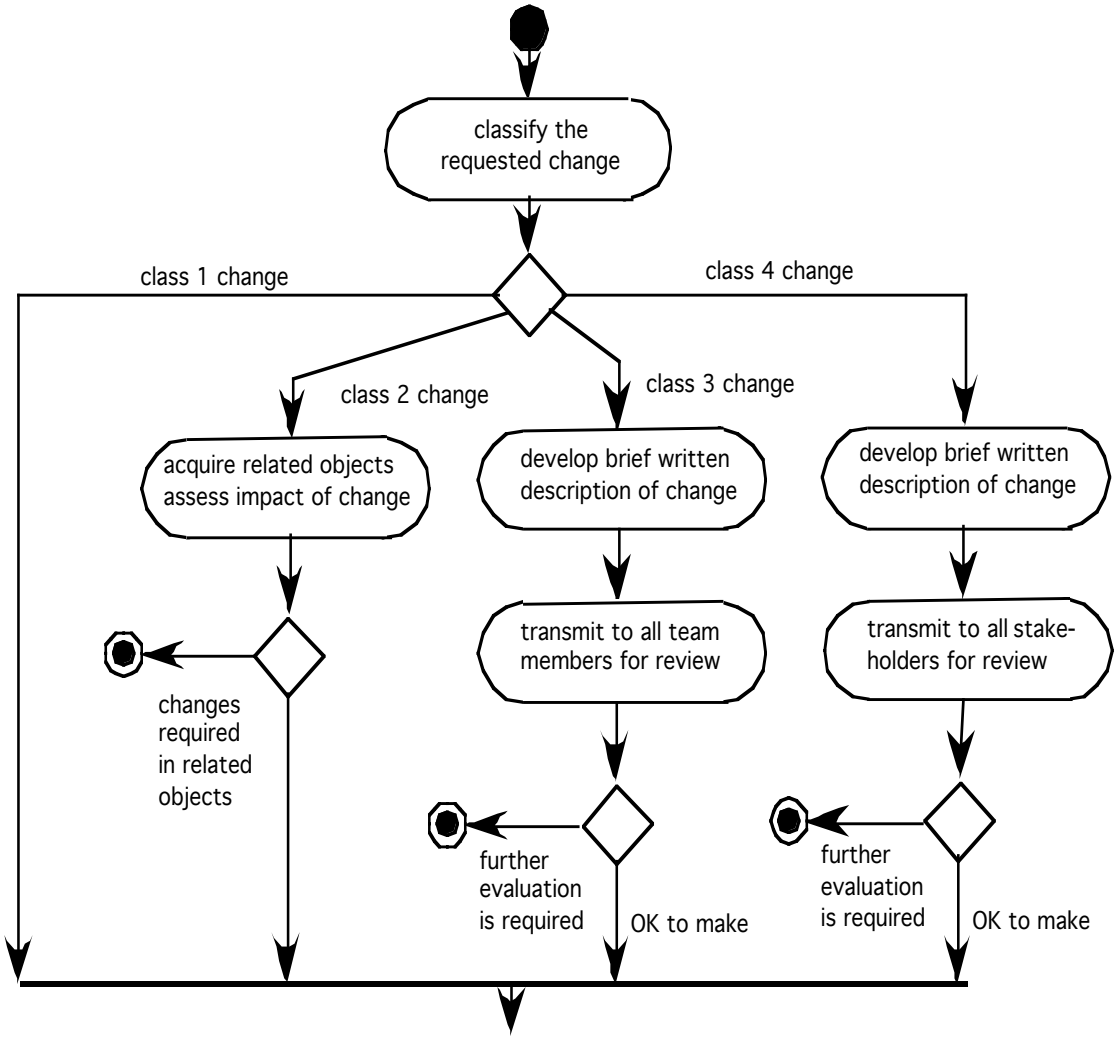
Content Management-II

- The **publishing subsystem** extracts from the repository, converts it to a form that is amenable to publication, and formats it so that it can be transmitted to client-side browsers. The publishing subsystem accomplishes these tasks using a series of templates.
- Each *template* is a function that builds a publication using one of three different components [BOI02]:
  - *Static elements*—text, graphics, media, and scripts that require no further processing are transmitted directly to the client-side
  - *Publication services*—function calls to specific retrieval and formatting services that personalize content (using predefined rules), perform data conversion, and build appropriate navigation links.
  - *External services*—provide access to external corporate information infrastructure such as enterprise data or "back-room" applications.

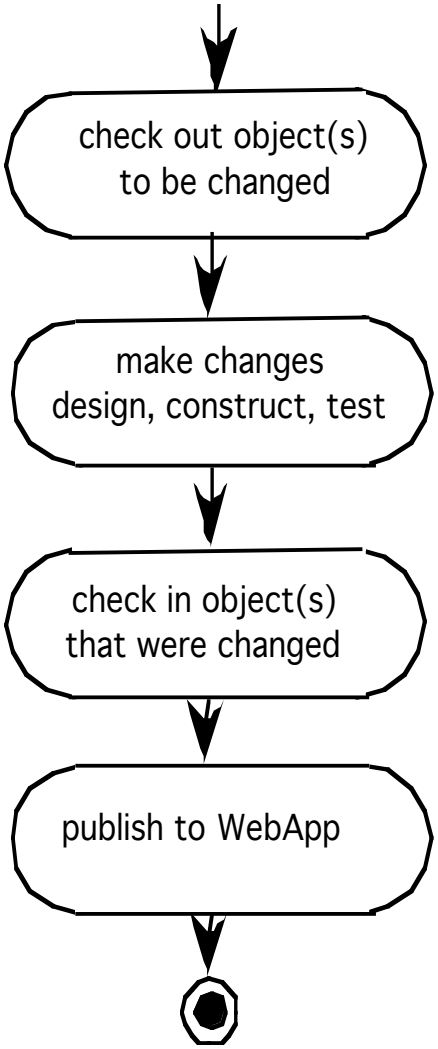
Content Management



Change Management for WebApps-I



Change Management for WebApps-II



Content

1. Changes and Software Configuration
2. Software Configuration Management
3. SCM Process
4. Version Control

4.1. Why version control?

- 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?

- 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

- 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

- 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?

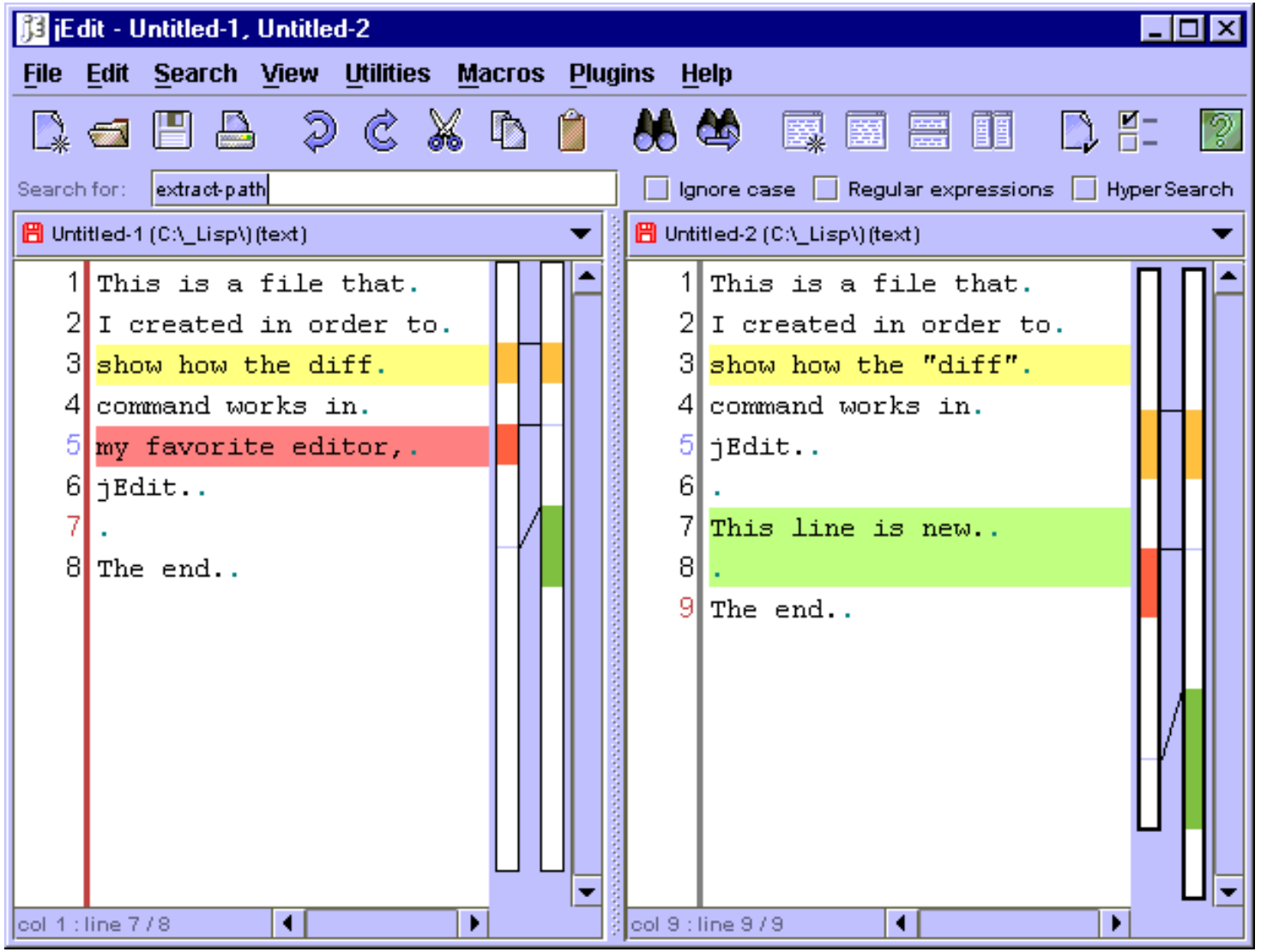
Tools: diff

- 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

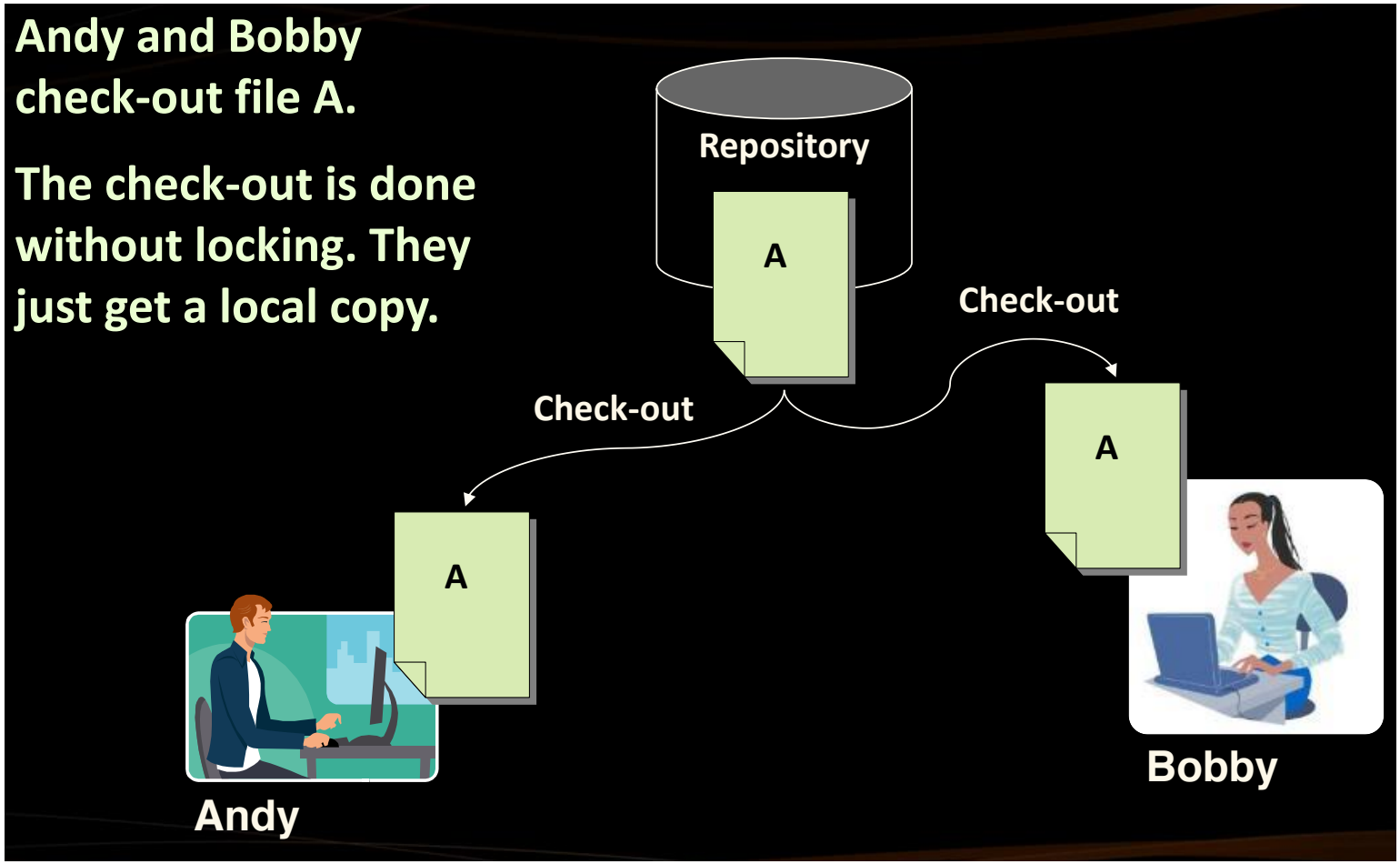
Version control systems

- 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

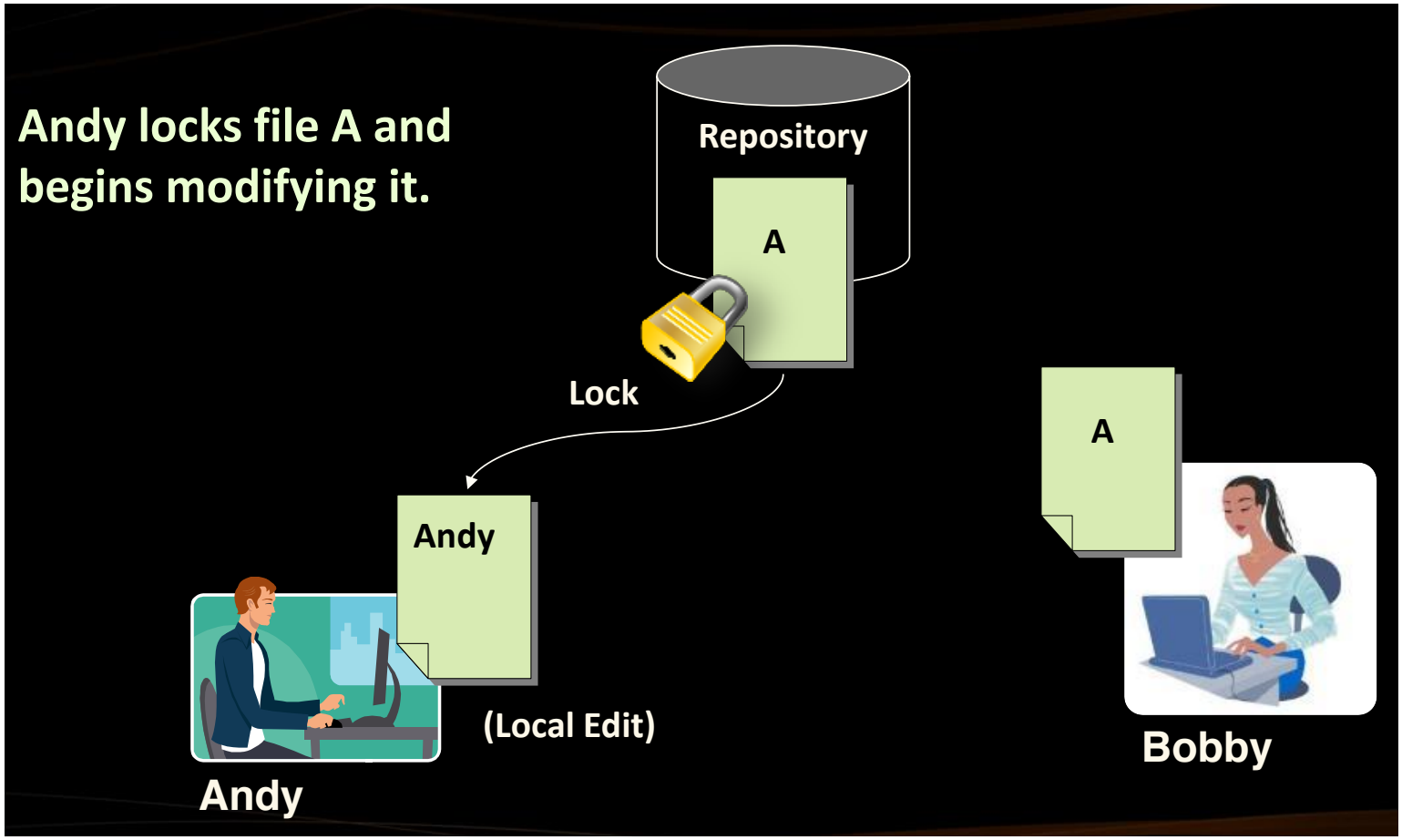
4.2. Versioning Models

- Lock-Modify-Unlock
- Copy-Modify-Merge

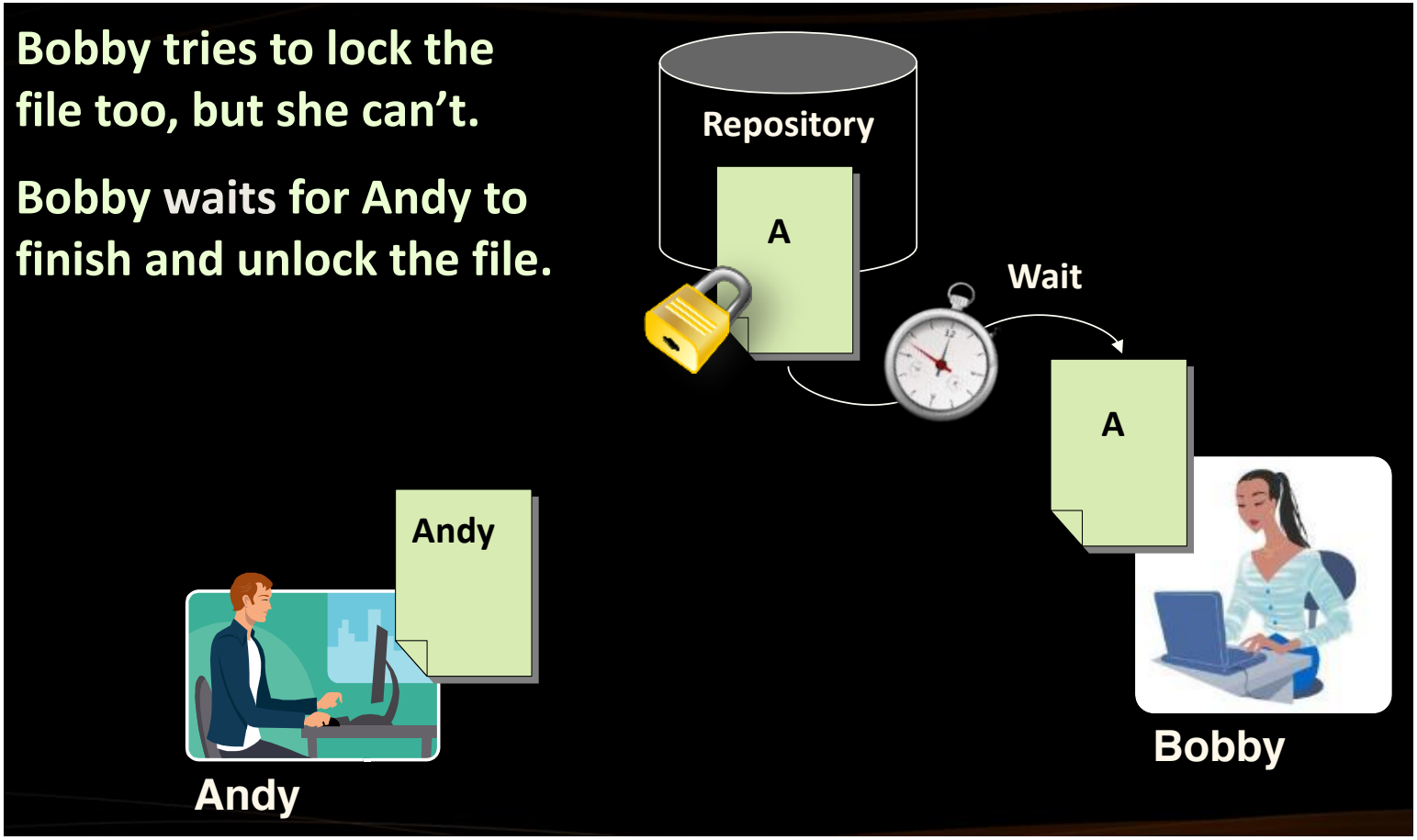
a. The Lock-Modify-Unlock Model



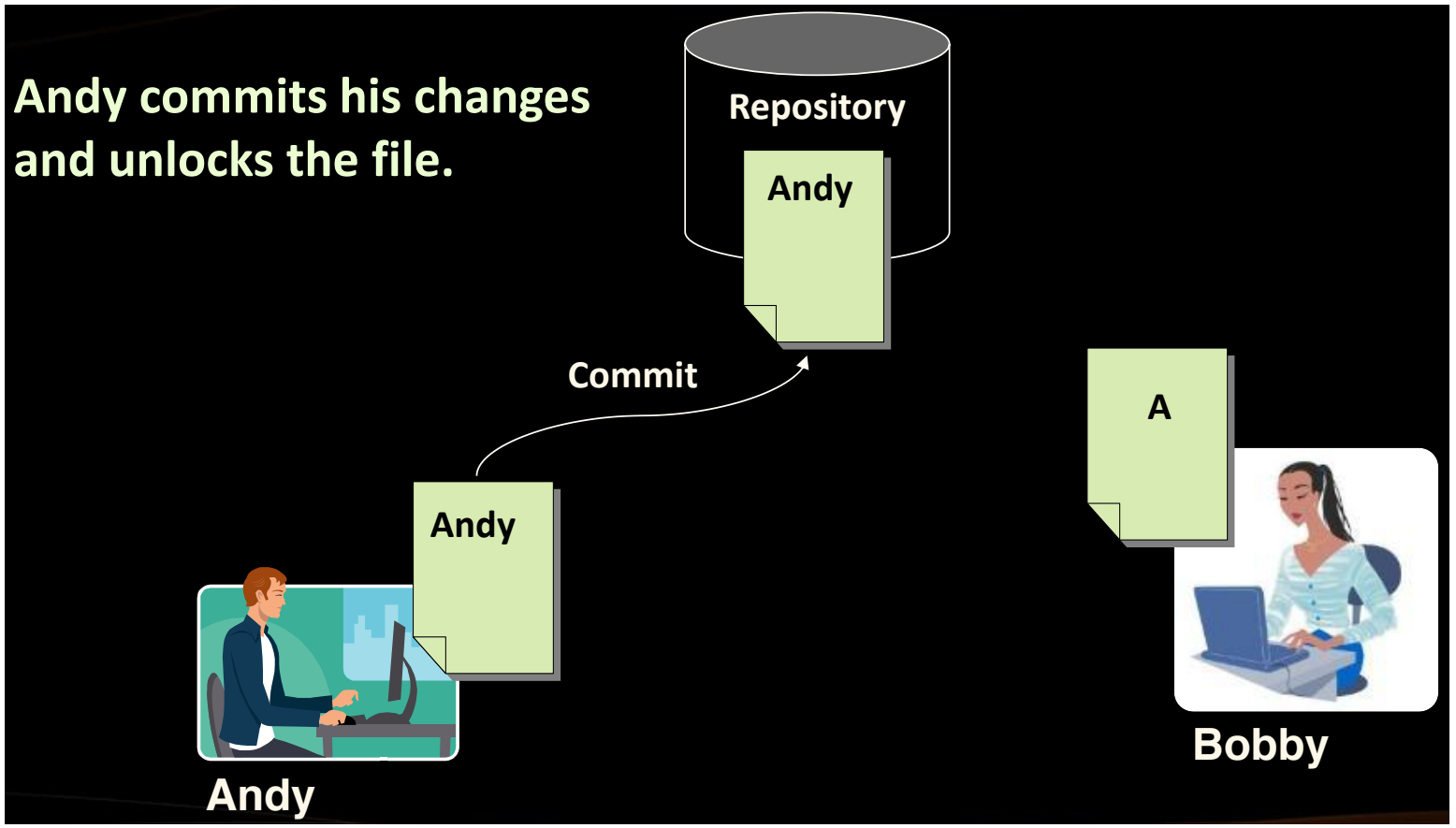
a. The Lock-Modify-Unlock Model (2)



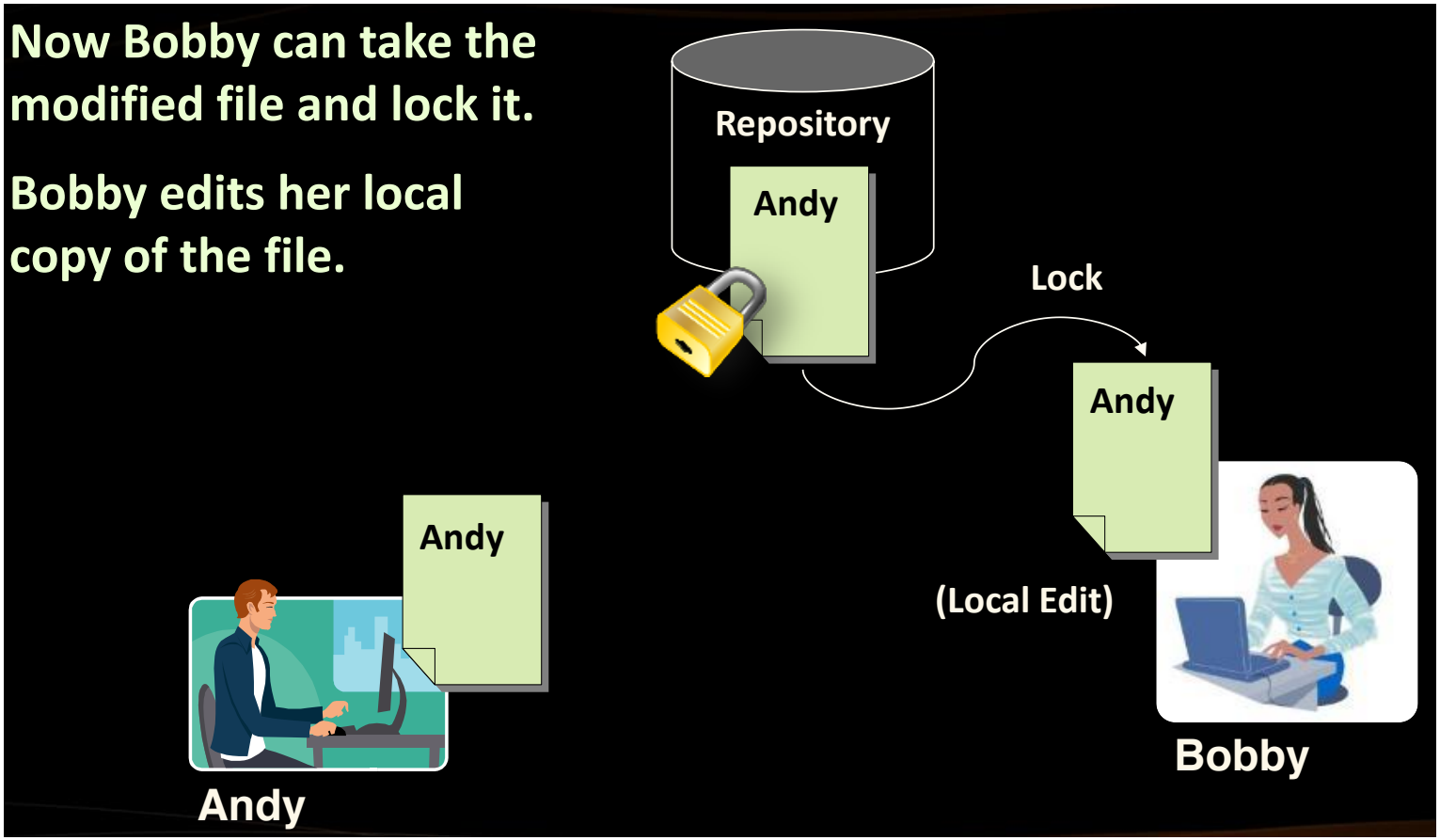
a. The Lock-Modify-Unlock Model (3)



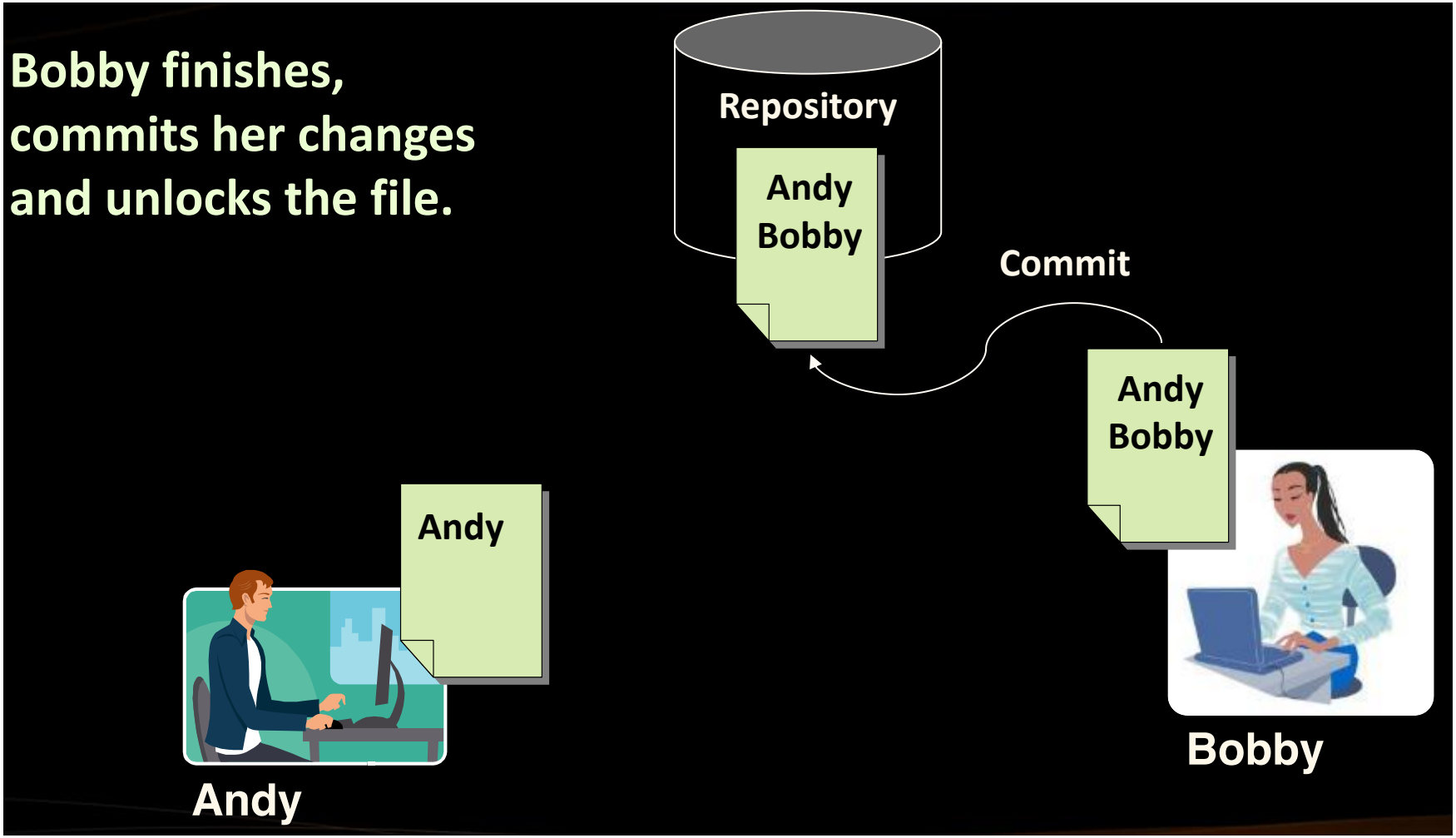
a. The Lock-Modify-Unlock Model (4)



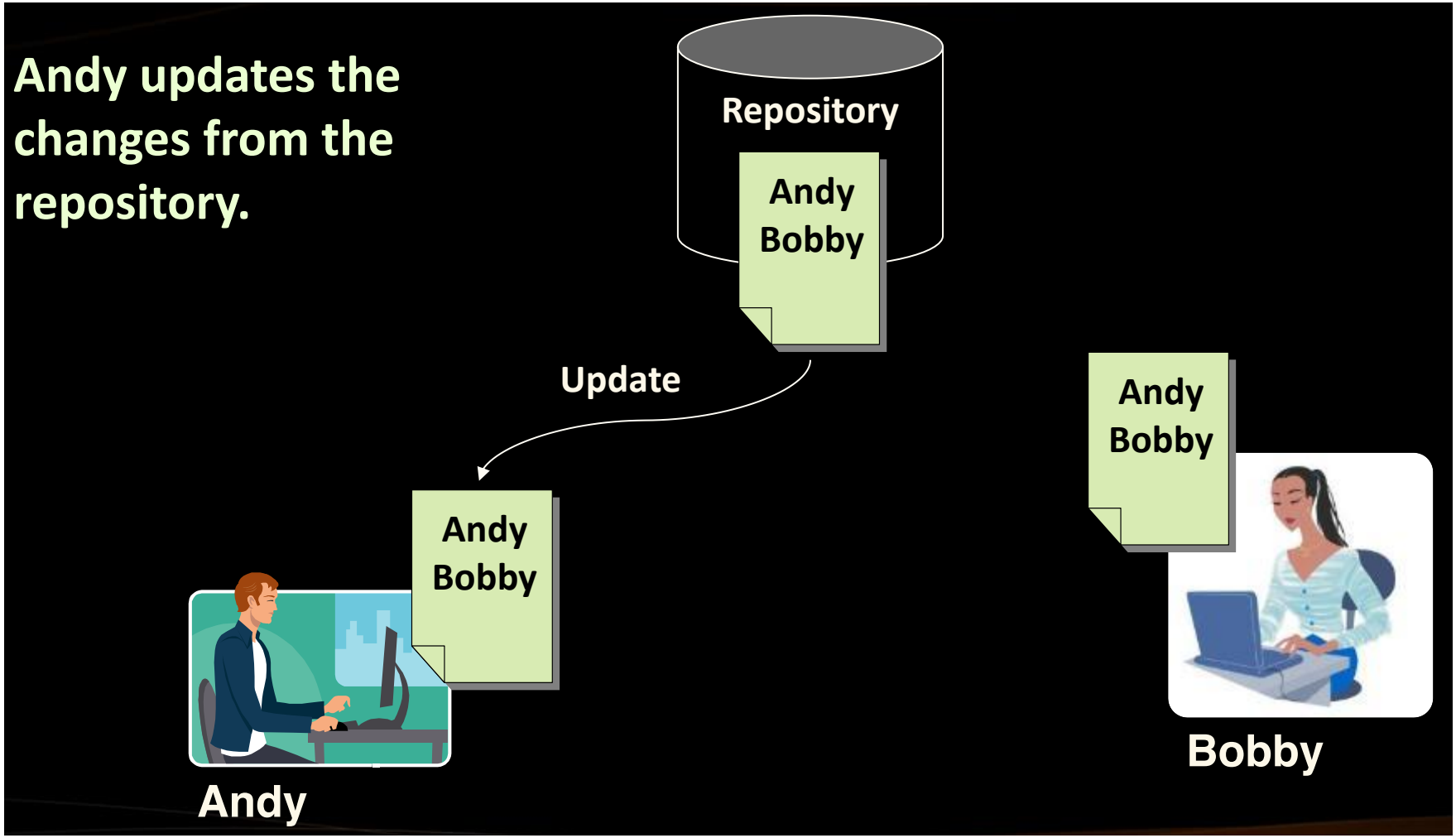
a. The Lock-Modify-Unlock Model (5)



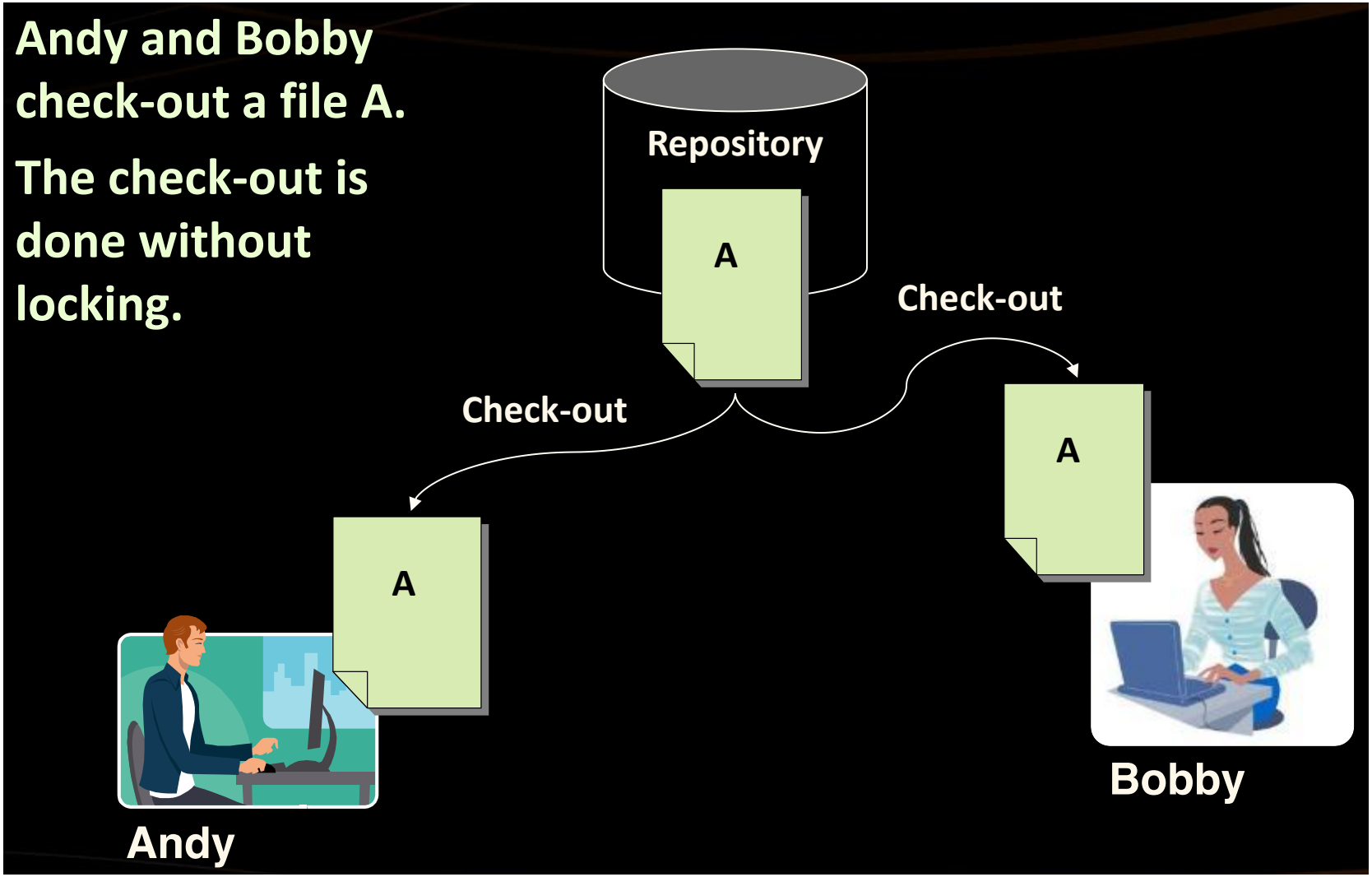
a. The Lock-Modify-Unlock Model (6)



a. The Lock-Modify-Unlock Model (7)

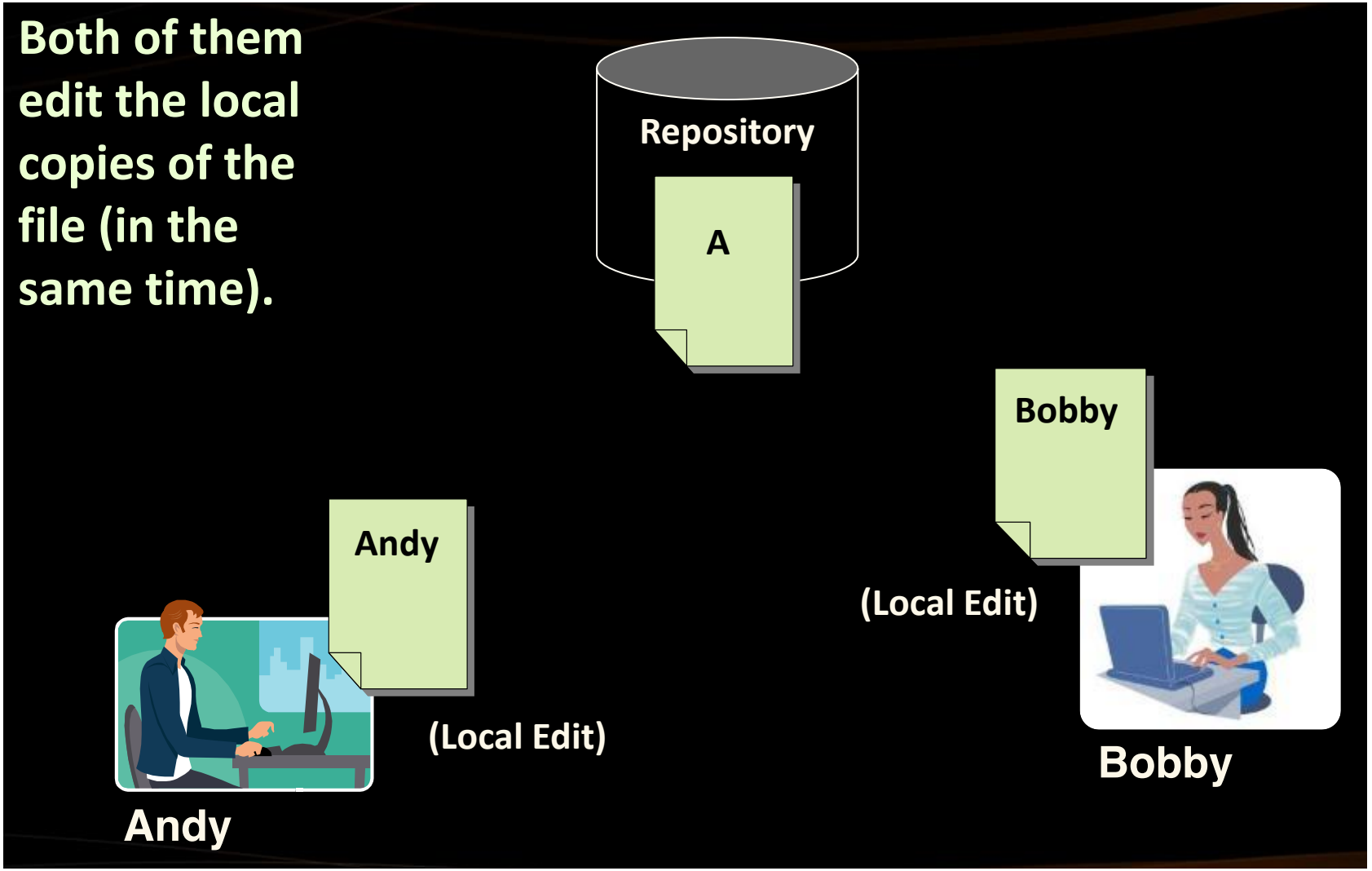


b. The Copy-Modify-Merge Model

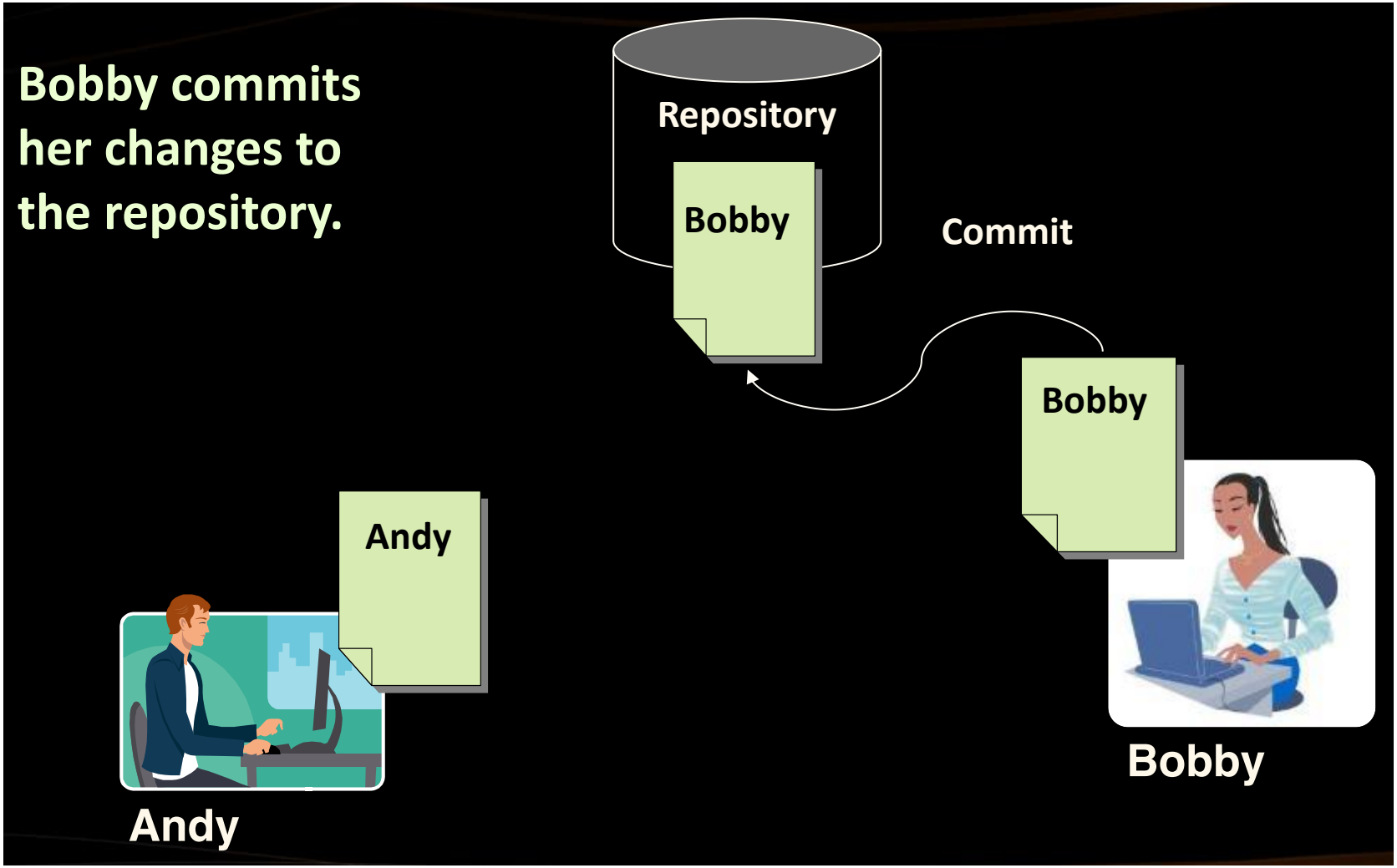


b. The Copy-Modify-Merge Model (2)

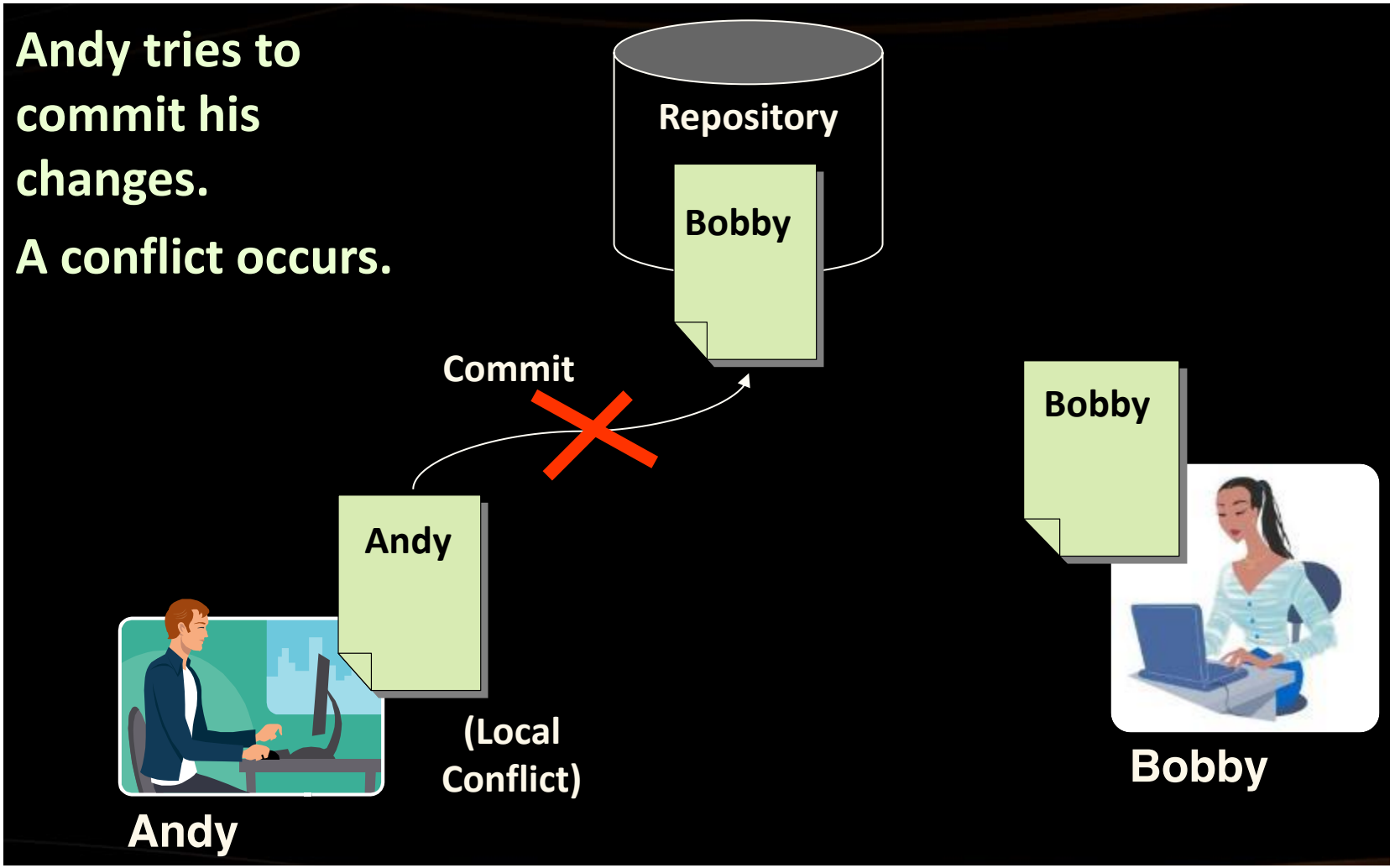
Both of them edit the local copies of the file (in the same time).



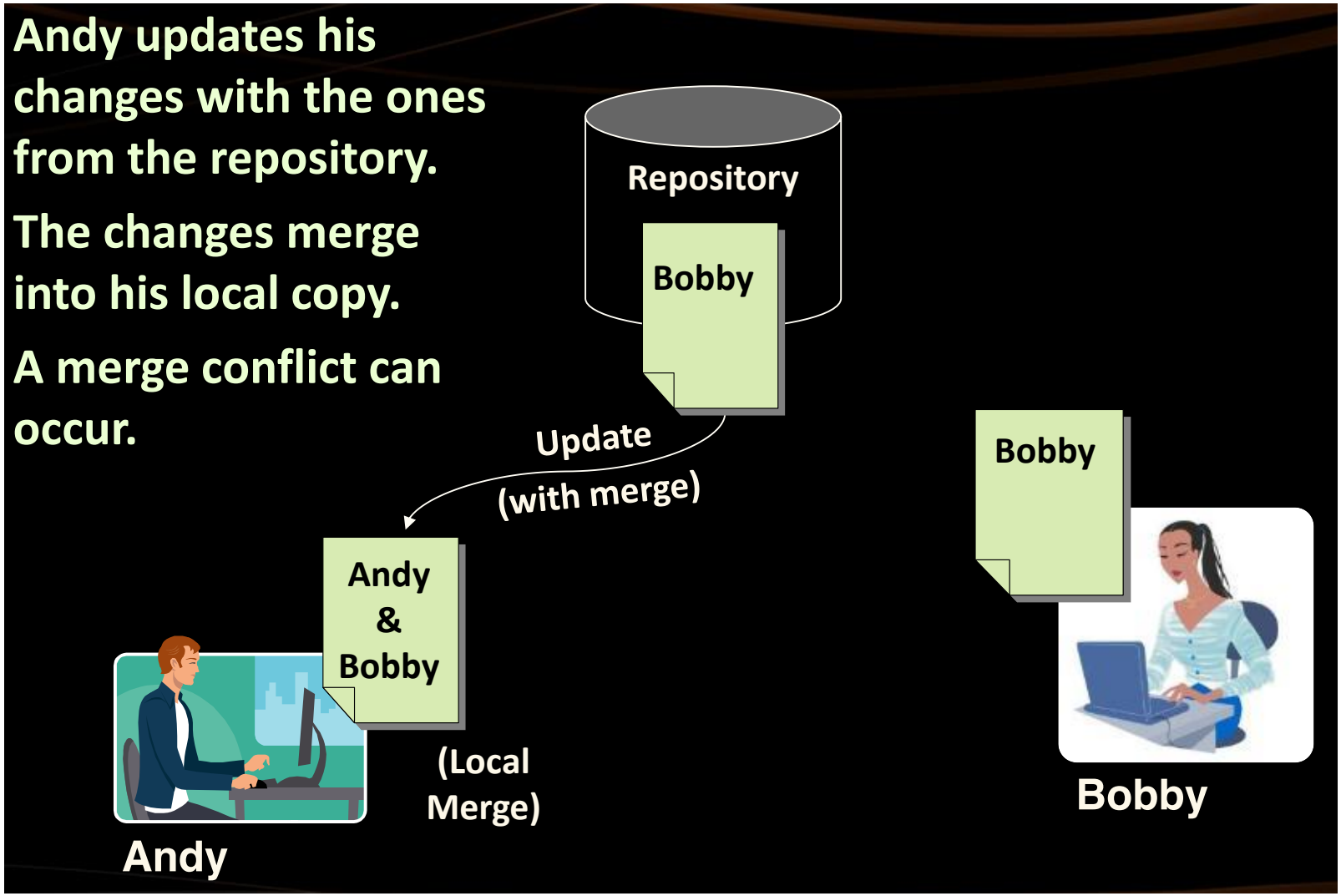
b. The Copy-Modify-Merge Model (3)



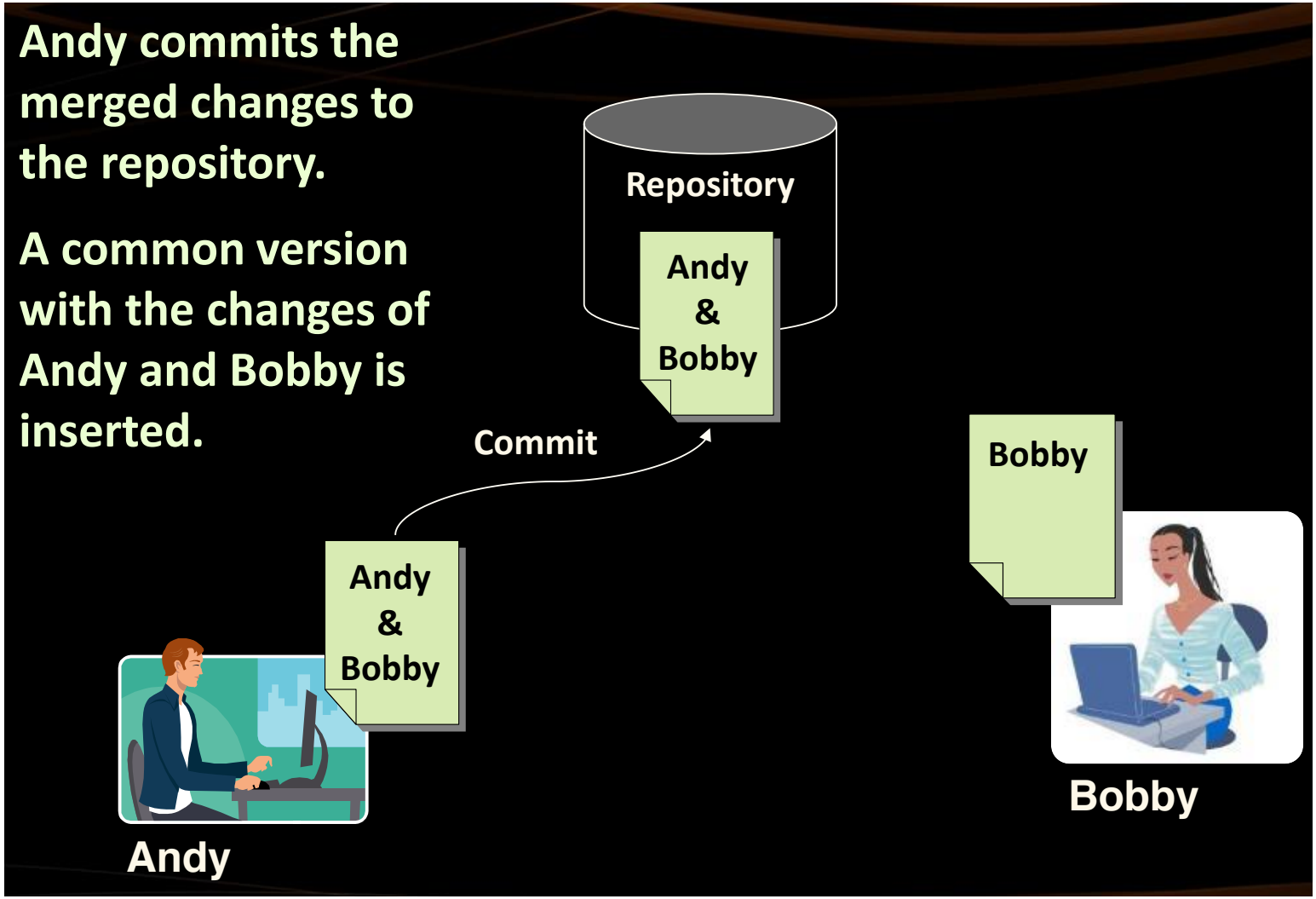
b. The Copy-Modify-Merge Model (4)



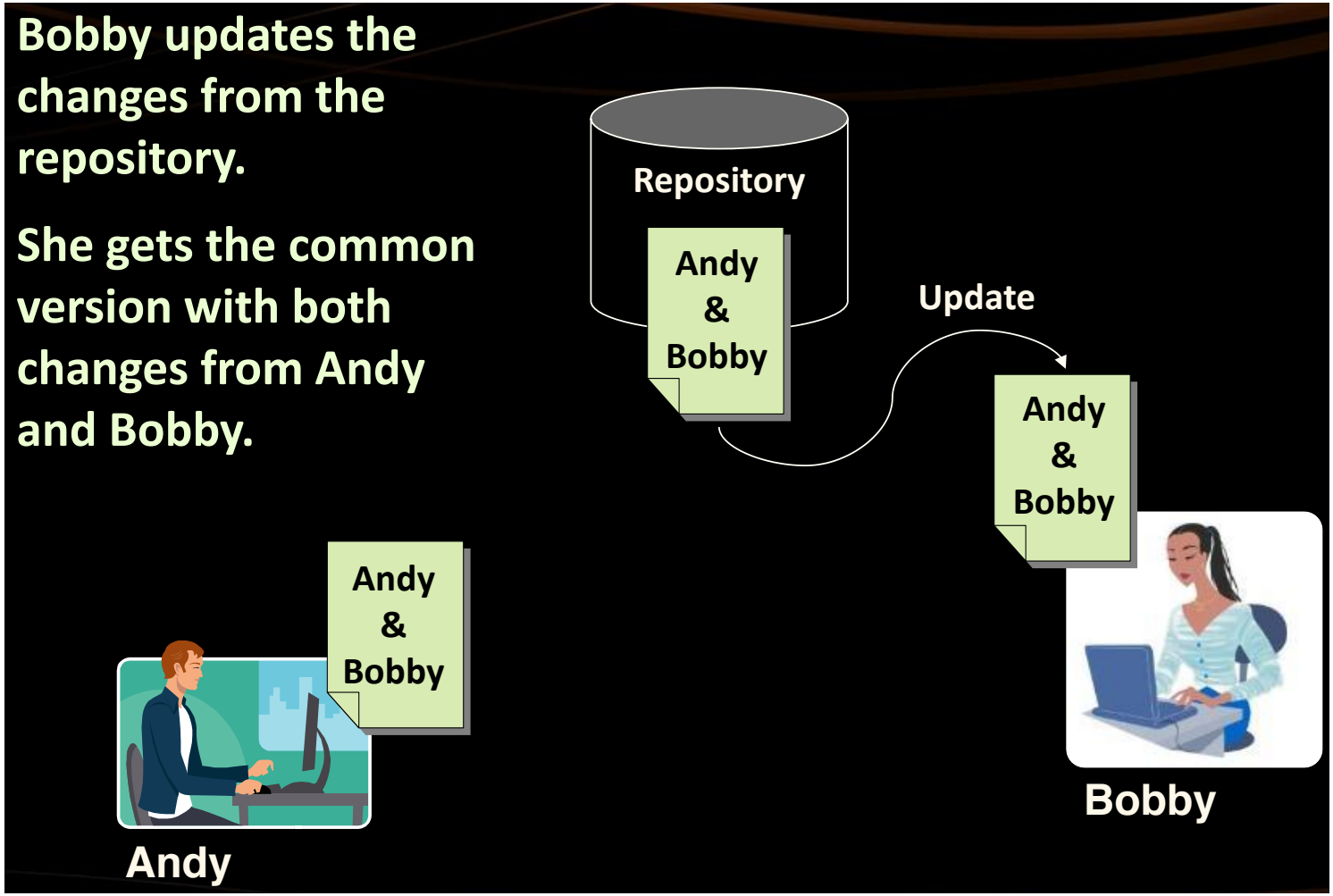
b. The Copy-Modify-Merge Model (5)



b. The Copy-Modify-Merge Model (6)

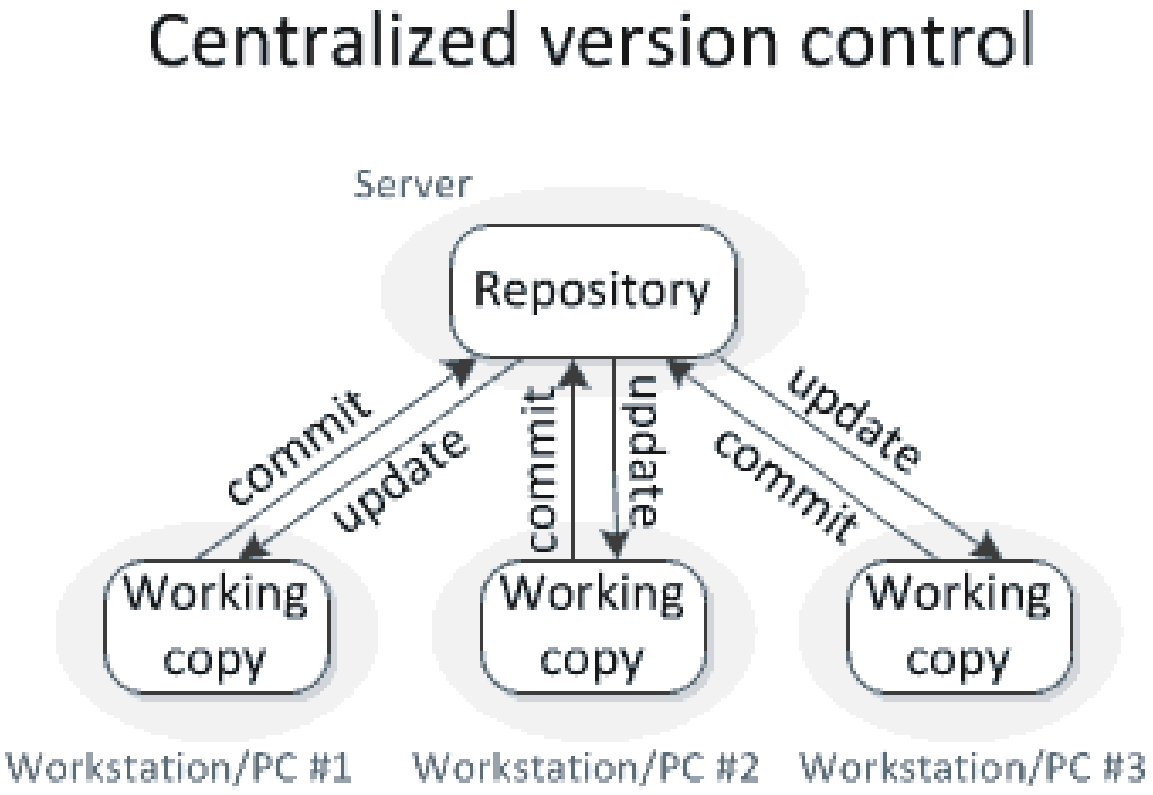


b. The Copy-Modify-Merge Model (7)

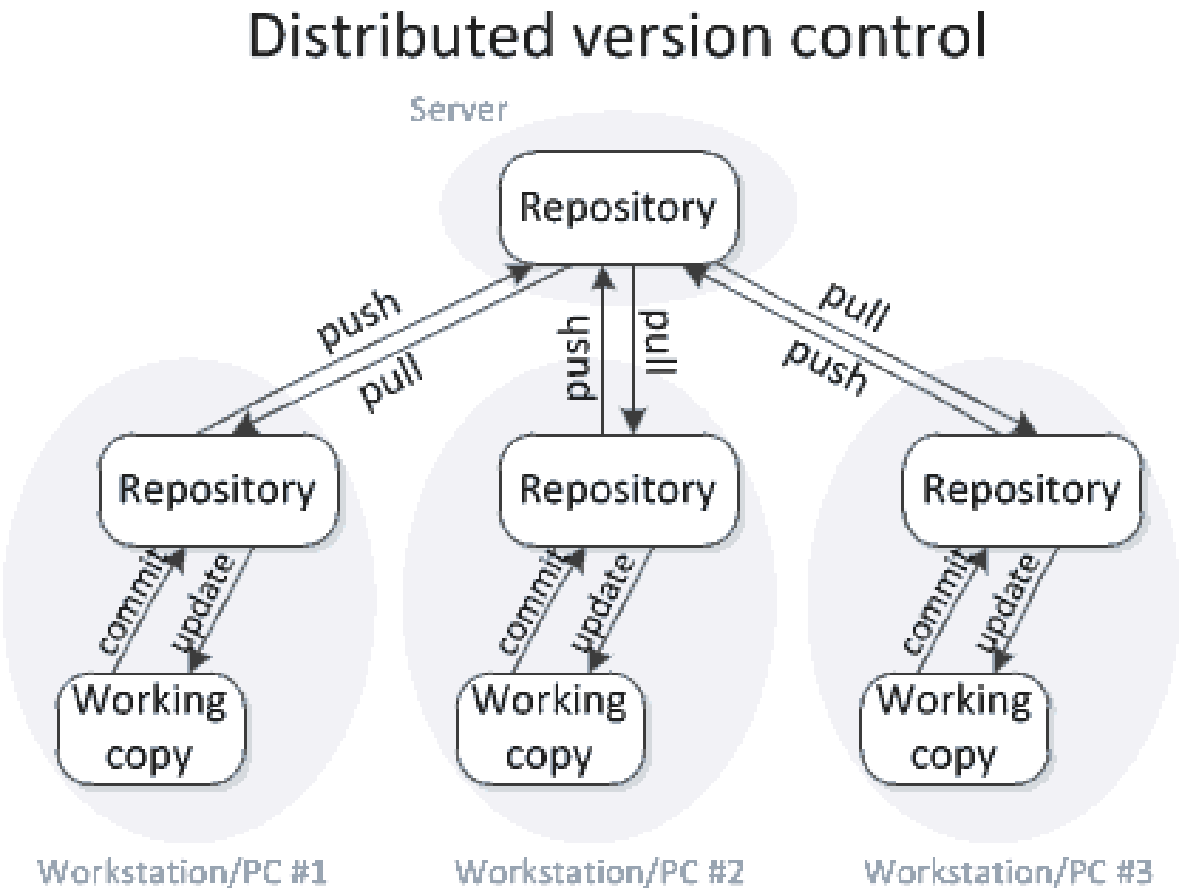


4.3. Central vs. Distributed version control

- Central version control
  - Only one repository



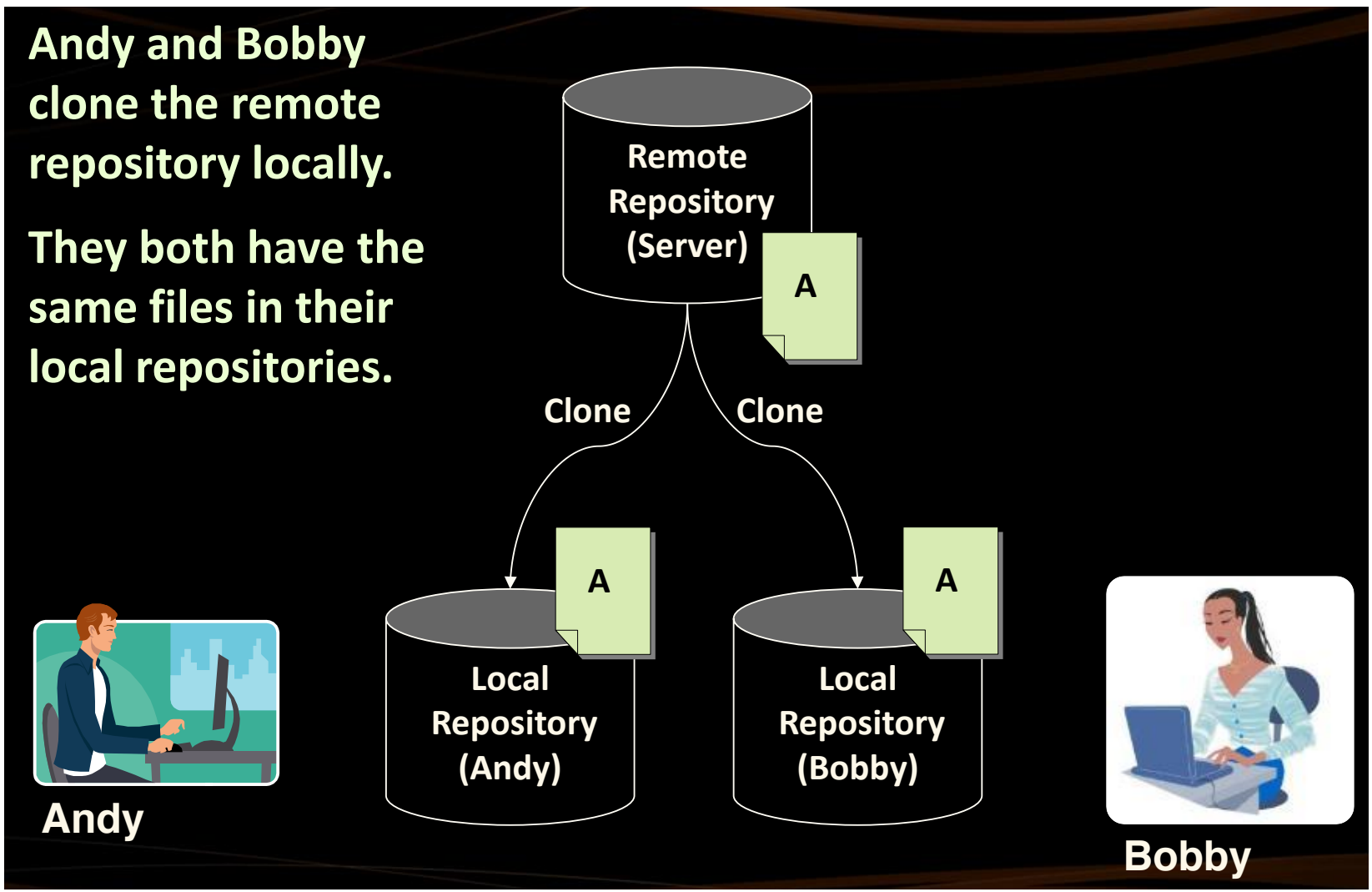
Distributed Version Control



Distributed Version Control (1)

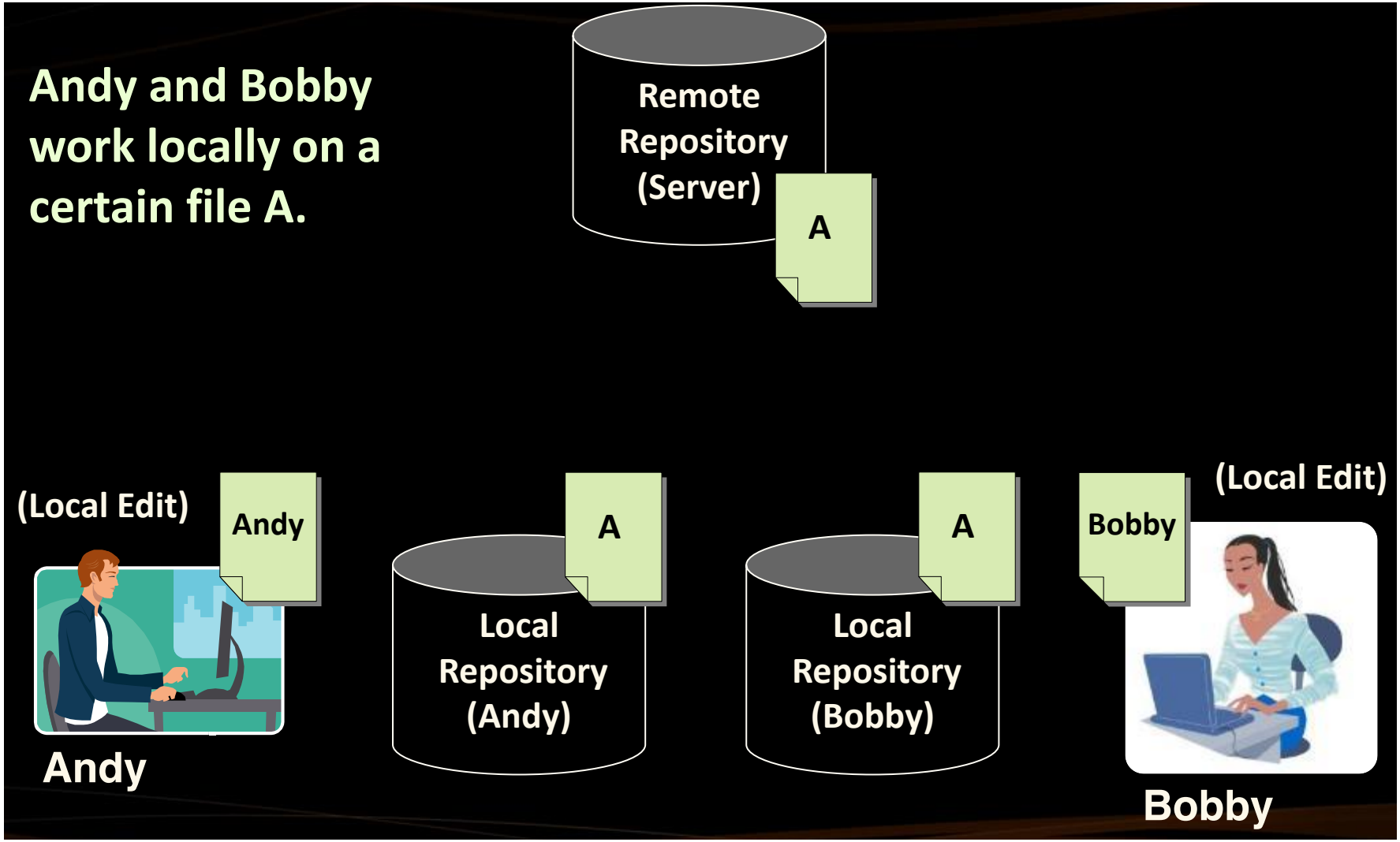
Andy and Bobby clone the remote repository locally.

They both have the same files in their local repositories.

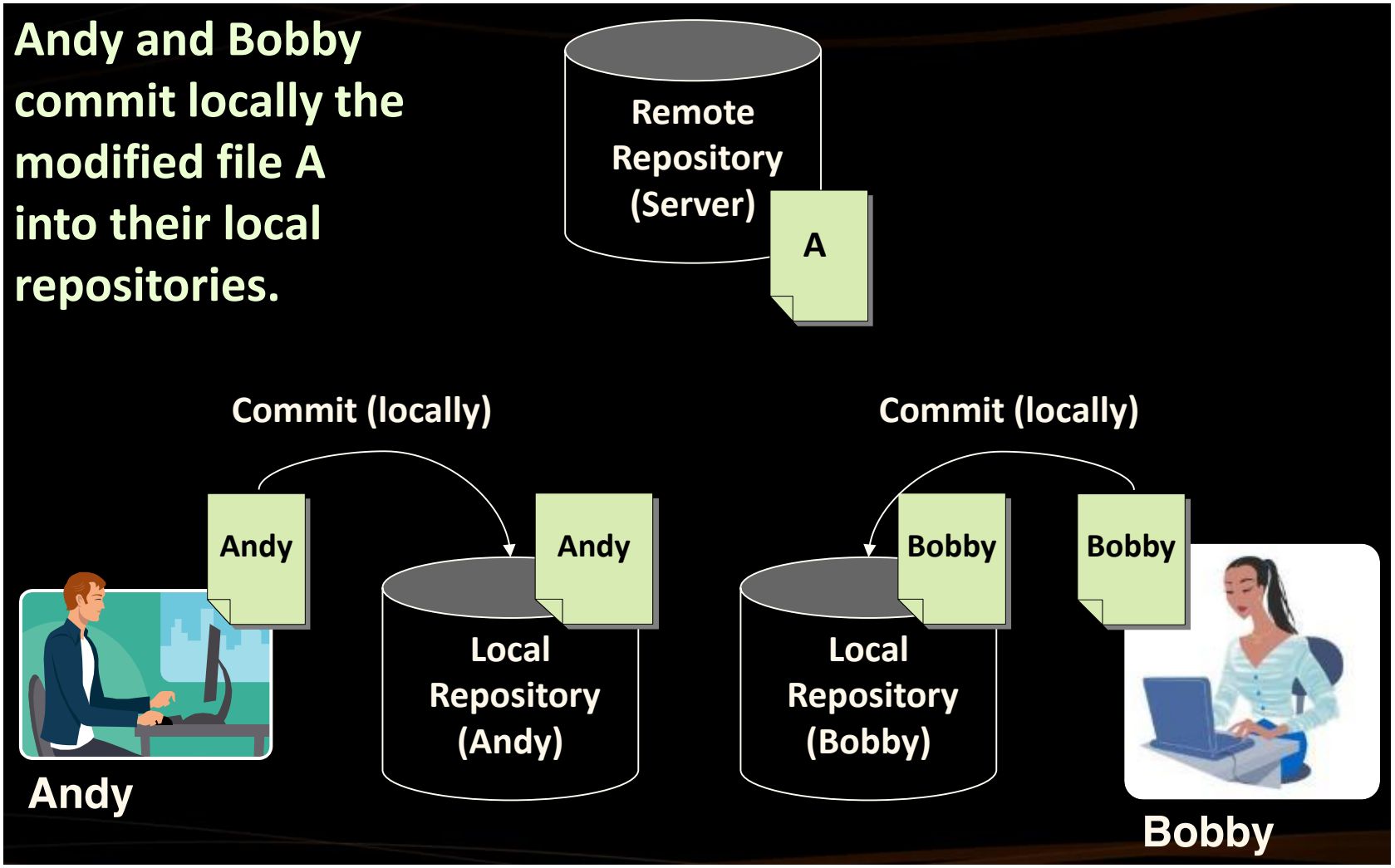


Distributed Version Control (2)

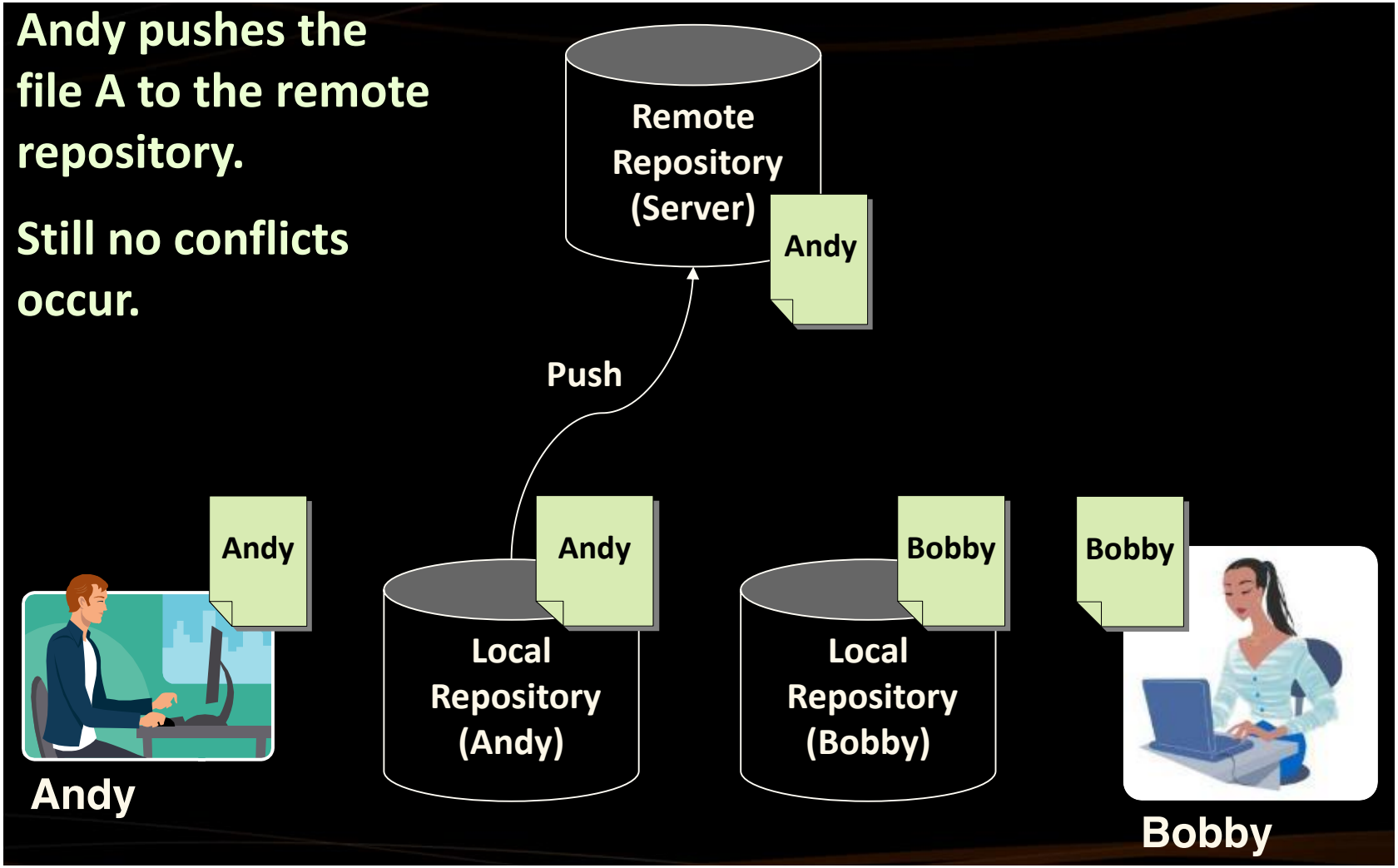
Andy and Bobby work locally on a certain file A.



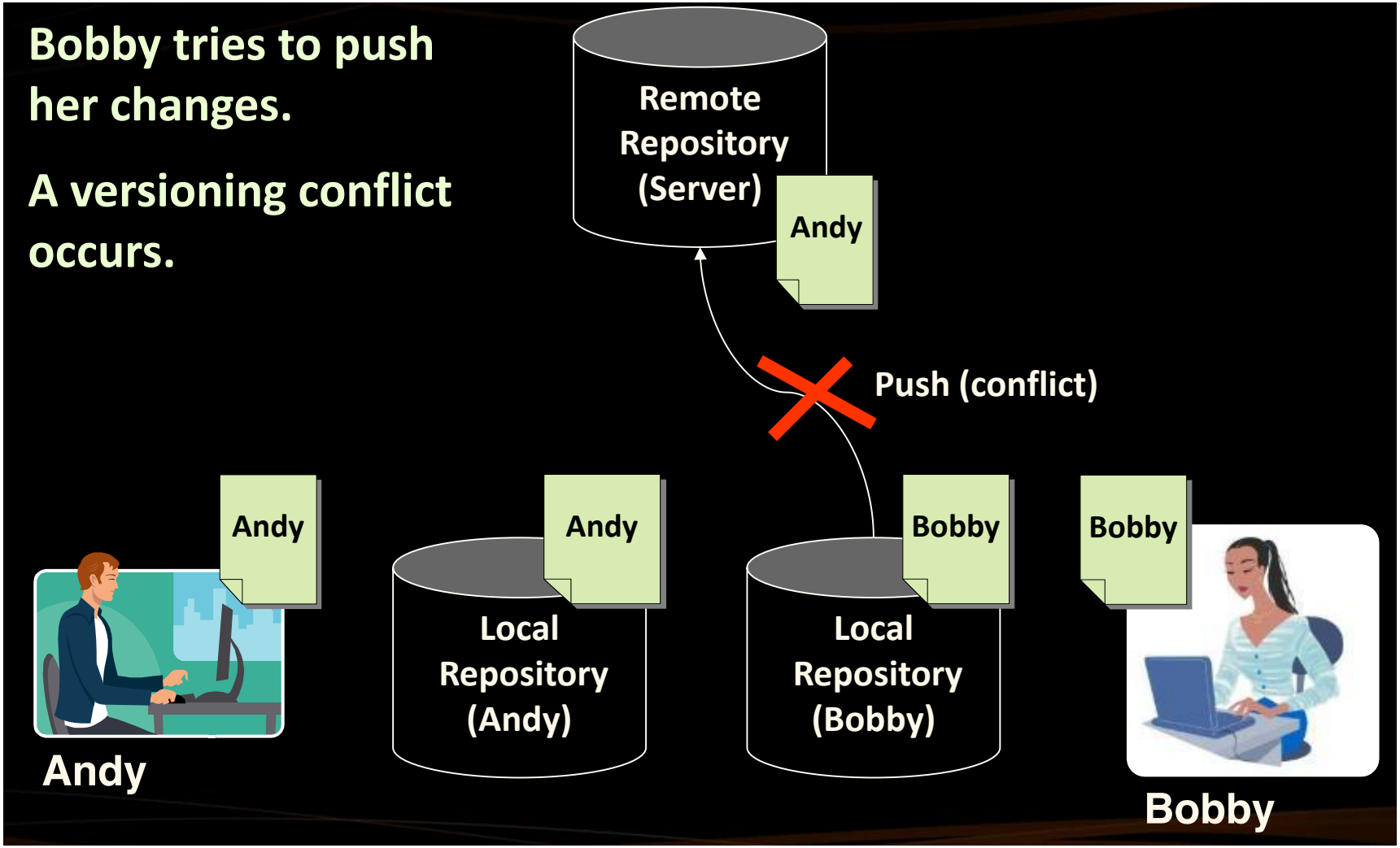
Distributed Version Control (3)



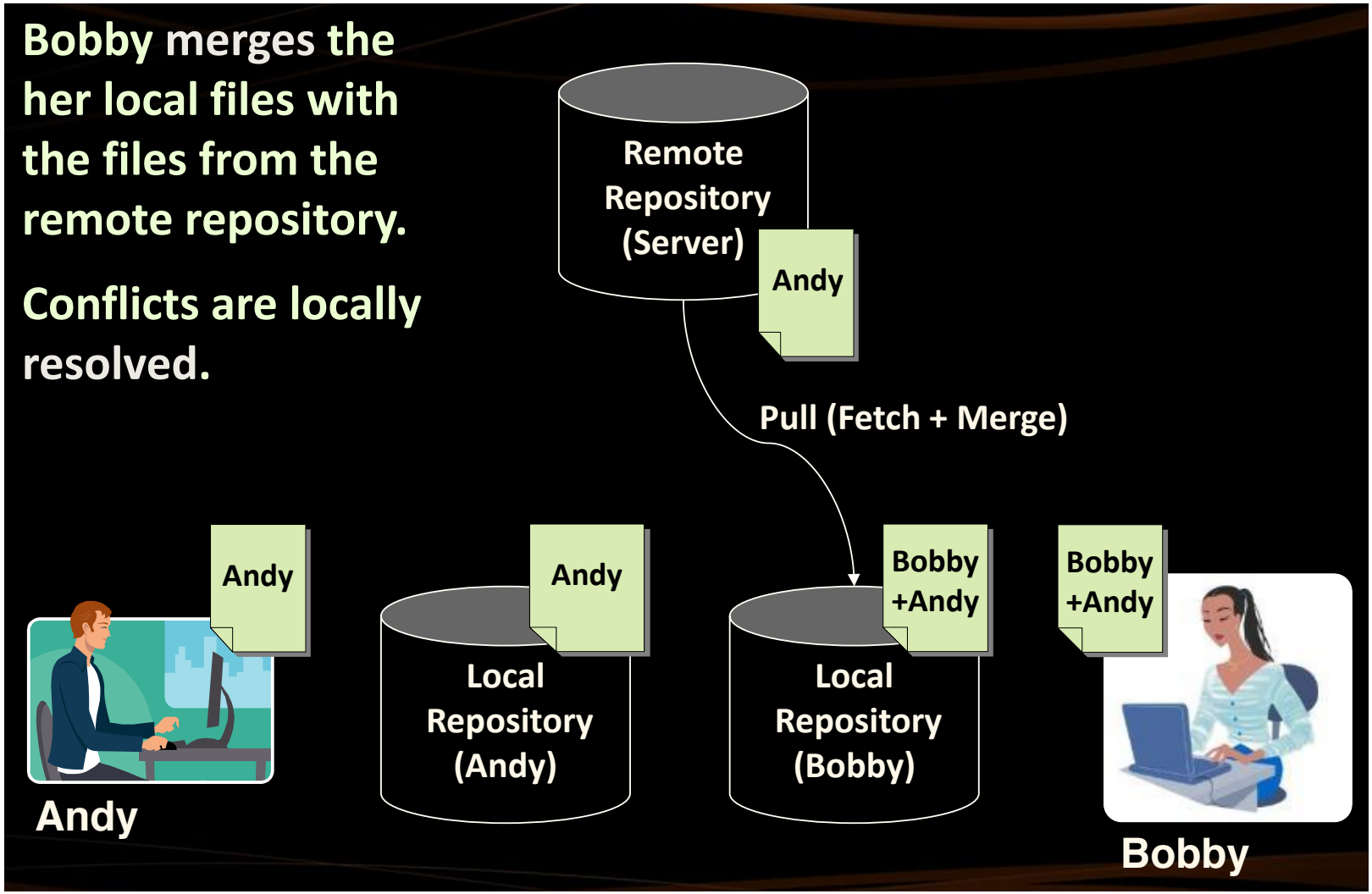
Distributed Version Control (4)



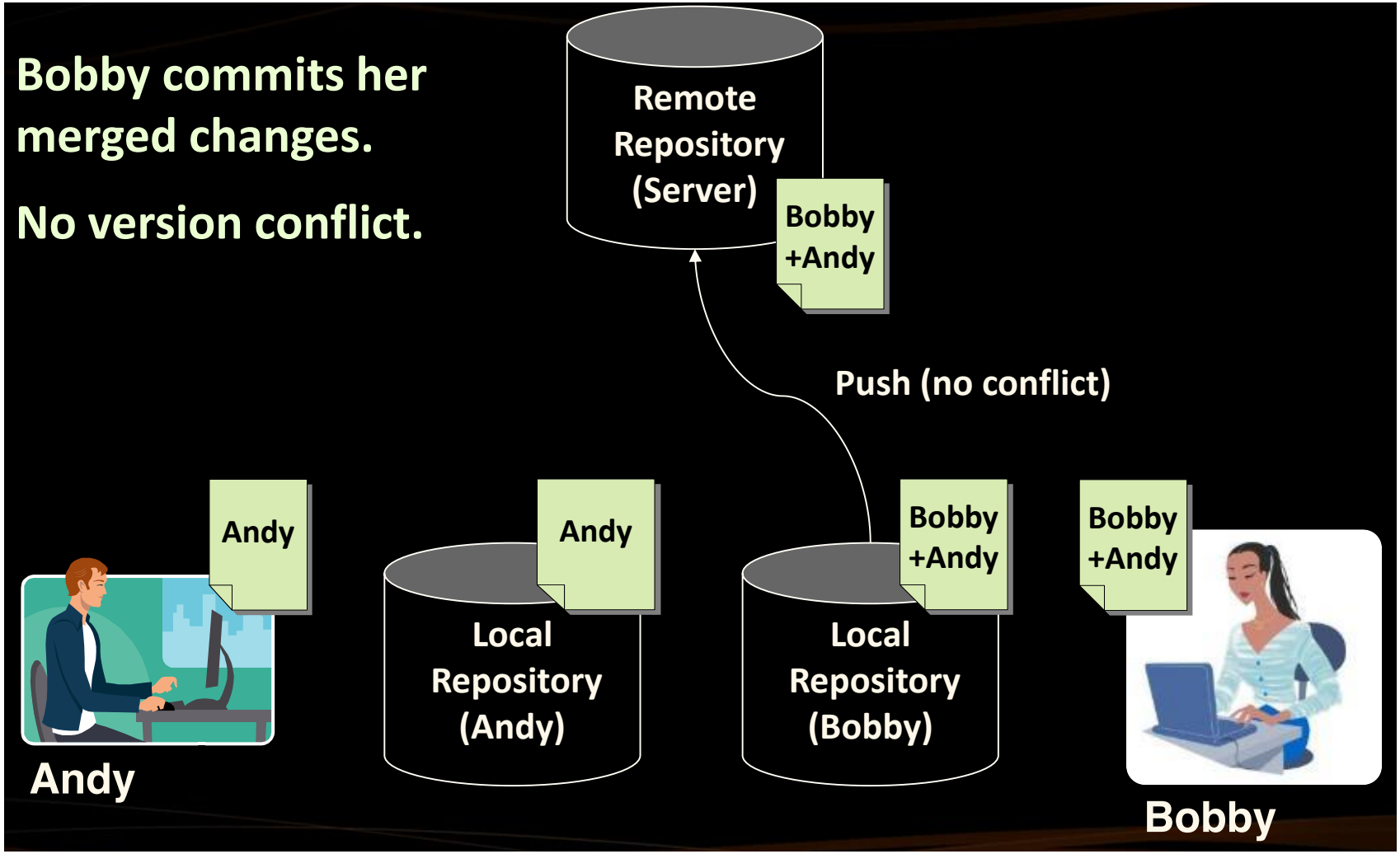
Distributed Version Control (5)



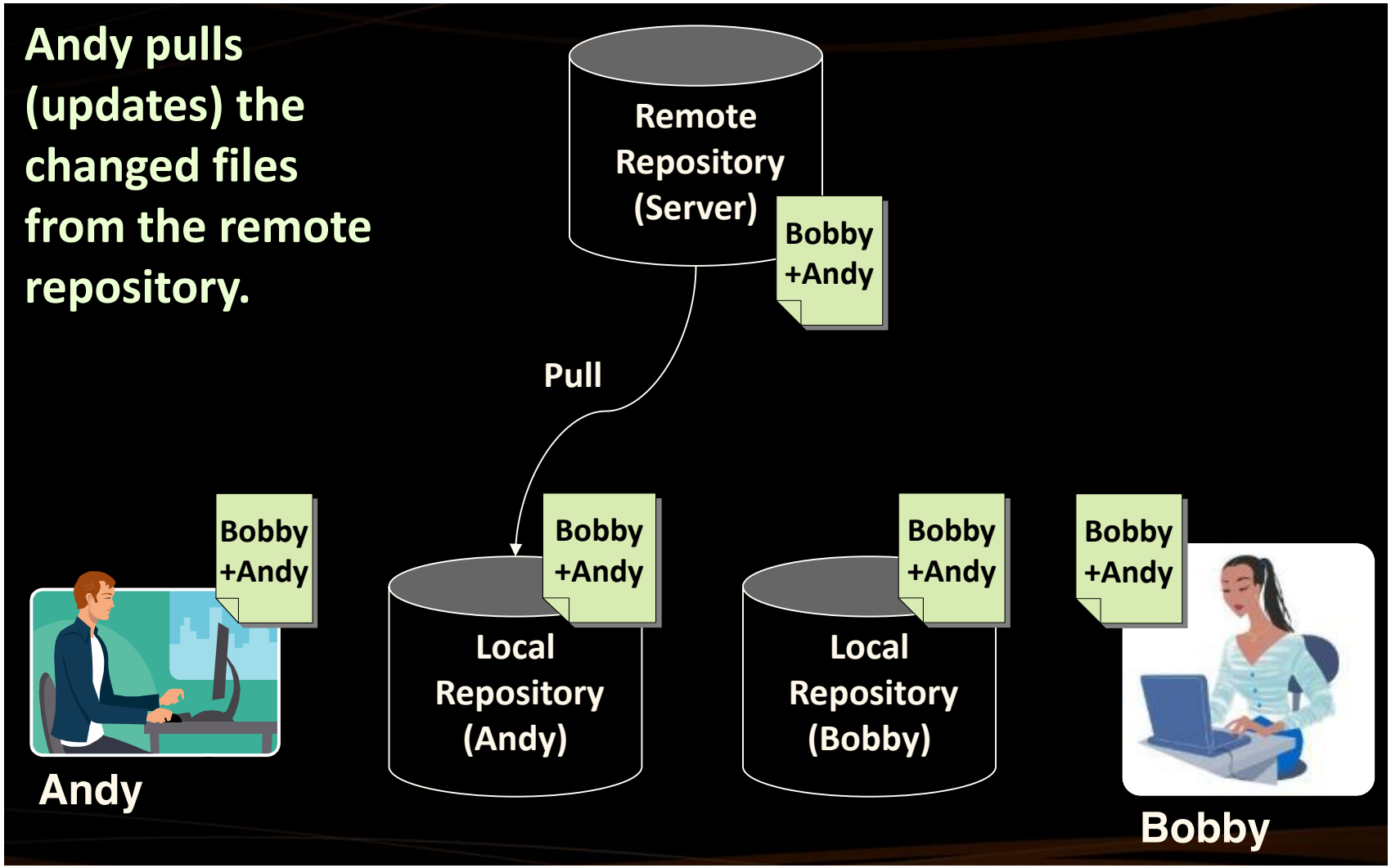
Distributed Version Control (6)



Distributed Version Control (7)



Distributed Version Control (8)



4.3. 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!

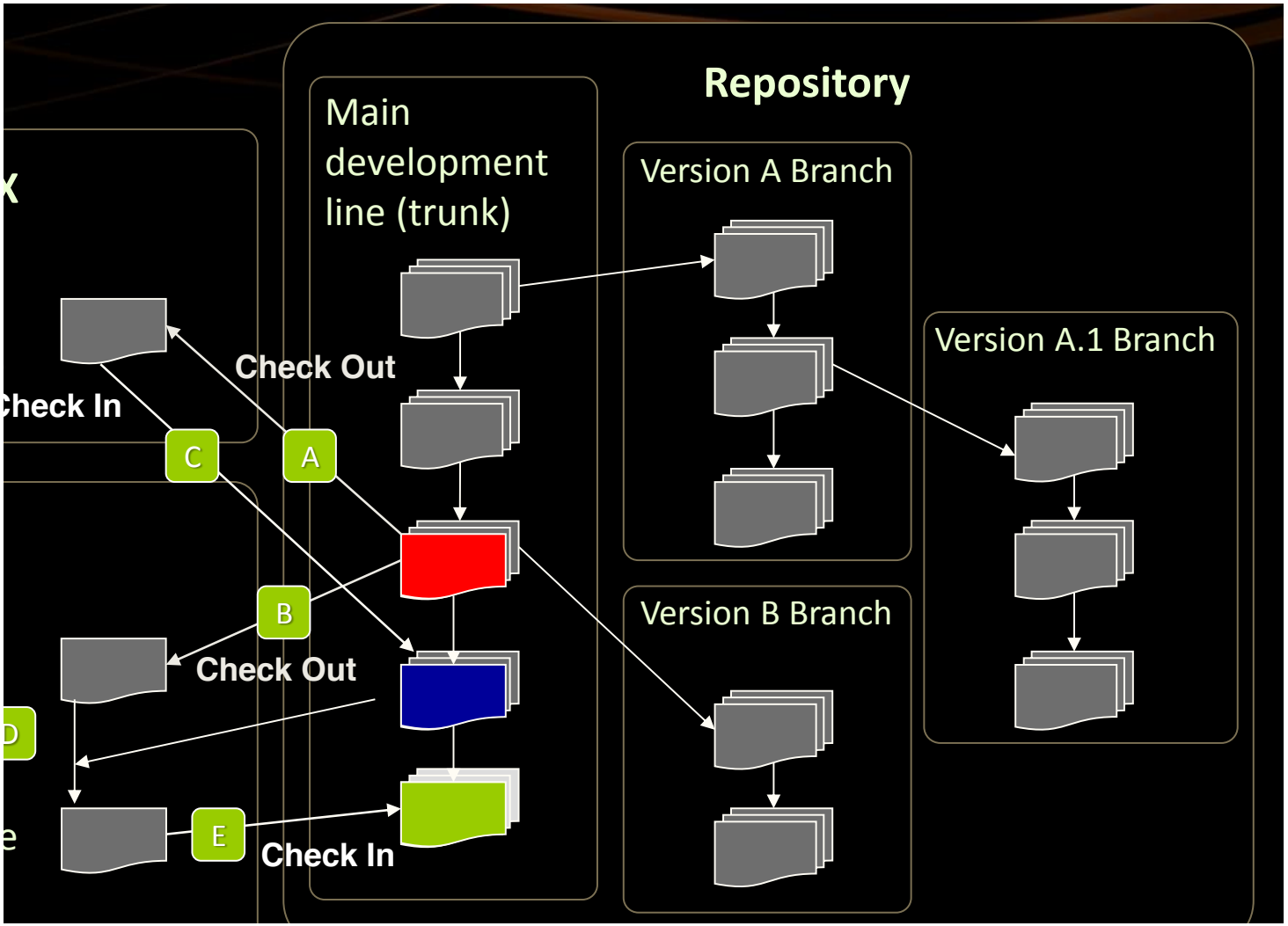
Vocabulary

- 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

Version Control: Typical Scenario



4.4. 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

Bitbucket demo

- Introduction to version control
  - https://www.youtube.com/watch?v=gY2JwRfin1M
  - See Episode 1-> 5
- Bitbucket
  - https://www.youtube.com/watch?v=BtEvnE79jxY&list=PL57RkP_325rLuT3EmFTf3lkoLw93Iw1_8

Homework

- Create an account in bitbucket.org
- Create a repository in bitbucket, join with all members in your group
  - Naming ~ the project name (NMCNPM.20171-06)
  - Add trangntt-student as a member in your project
- Tutorials
  - git-scm:
    - https://git-scm.com/book/vi/v1/
    - https://git-scm.com/book/en/v2
  - Framgia:
    - https://github.com/framgia/coding-standards/blob/master/vn/git/flow.md
    - https://github.com/framgia/coding-standards/blob/master/eng/git/flow.md

Homework

- Use Eclipse (eGit) and terminal (command line) to do at least the following tasks with bitbucket
  - Clone your repository to your local system
  - Add a file to your local repository and put it on Bitbucket
  - Create a file in Bitbucket
  - Pull changes from a remote repository
  - Create a branch and make a change
  - Merge your branch: fast-forward merging
  - Push your change to Bitbucket