Version Control Introduction

Troy C. Haskin

University of Wisconsin–Madison

5/12/2014

1. Motivation
   - Paper Example
   - Other Examples
2. Version Control System
   - Introduction
   - How it Works
3. Git and GitHub
4. End

1 Motivation
Paper Example
Other Examples
2 Version Control System
Introduction
How it Works
3 Git and GitHub
4 End

Paper Workflow

A typical workflow of writing a paper:

- Start with an idea/outline
- Make a draft
- Proof
- Edit
- Repeat until done

Workflow Shortcomings

Individual paper:

- Forget what changes were and were not made
- Make big changes but like the way it was
- Want to use the material again but alter for a different audience, journal, etc.

Group paper:

- Don't know what changes were and were not made
- Not sure what version of the paper you or others have
- Not sure who or what was added to the version in-use

Solution

A common solution not using a version control system (VCS):

- Make a new directory and name appropriately
  - `NuclearEngineeringAndDesign`
  - `AnnalsOfNuclearEnergy`
- Make a new copy of the file and name it something different
  - `AwesomePaper-Draft1.docx`
  - `AwesomePaper-AdvisorsNotes.docx`
  - `AwesomePaper-Draft2NeedCitations.docx`
  - `AwesomePaper-Final.docx`
  - `AwesomePaper-FinalAdvisorNotes.docx`
  - `AwesomePaper-FinalFinal.docx`

Solution Shortcomings

- Proliferation of files and directories
- No automatic list of changes; "proper" naming attempts to correct this (e.g., `Draft2NeedCitations`)
- Ability to go back to an earlier version would complicate naming
- Collaboration issues still not addressed

1 Motivation
Paper Example
Other Examples
2 Version Control System
Introduction
How it Works
3 Git and GitHub
4 End

RELAP/MELCOR Inputs:

- Build the model by slowly adding control volumes and heat structures
- Adjust geometry input as more information becomes available
- Correct issues as they're discovered
- Might break things and need to find a older, working version
- **Re-use the input for multiple different simulations or numerical experiments**

Writing Programs

- Start simple and add more functionality
- Fix bugs as they're discovered
- Might break things and need to find a older, working version
- Someone else might want to leap off of the work already done but apply it differently

Same Problems

- These examples, and many more, all have the problems presented by the paper example.
- The problems only become worse as the work becomes larger or more people become involved.

What's the solution?

1 Motivation
Paper Example
Other Examples

2 Version Control System
Introduction
How it Works

3 Git and GitHub

4 End

What is it?

Definition A system that records changes to a file or set of files over time so that you can recall specific versions later. [src]

Features:

- Revert a file or an entire project back to a previous state
- Review changes made over time
- See who last modified something
- Create an off-shoot from a current project state (branching)
- Create a brand new project from a current project (forking)
- Work locally and save to an online system (distributed systems)

Advantages / Disadvantages

Advantages

- History of the project is automatically cataloged
- All versions of the project are saved and ID-ed automatically
- Line-by-line and person-by-person reviewable history.

Disadvantages:

- Can't see line-by-line changes for binary files (e.g., `docx` or image files)
- Not good for saving humongous files (large binary data files shouldn't be versioned)
- Requires discipline and effort to log and sync changes
- Becomes much, much more complicated for larger projects (not a worry for us)

1 Motion
Paper Example
Other Examples

2 Version Control System
Introduction
How it Works

3 Git and GitHub

4 End

Definitions (Examples to follow)

Repository A directory that holds all project files and VCS information.

Commit A submission of changes from the user to the VCS; this creates a new version and saves the previous state in the history

Commit Message A short/long description of the changes present in the commit.

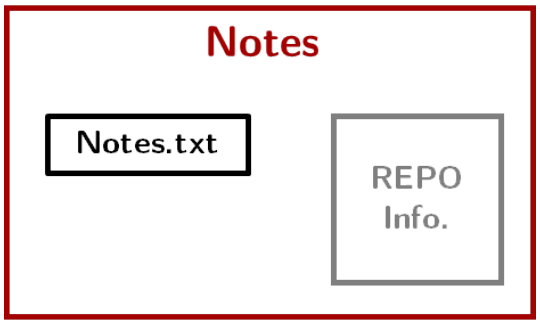
Branch A new, separate line of history starting from a certain version; changes can be made to a branch without affecting what it was branched from

Diff A comparison of two files with line-by-line differences highlighted

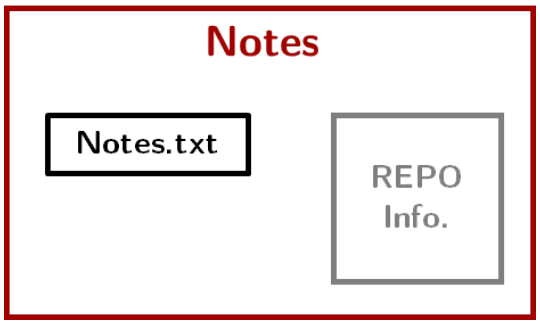
Sync/Push A synchronization of a local repository with a non-local one

History/Commit Example

Create repository called "Notes".



Add a new file Notes.txt

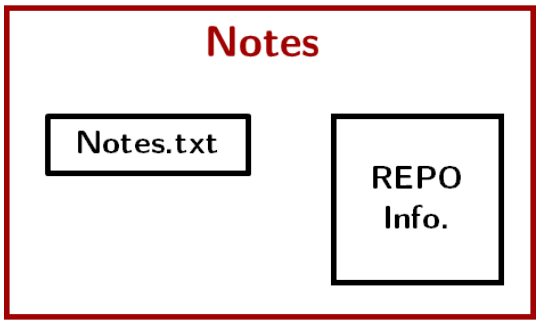


History:

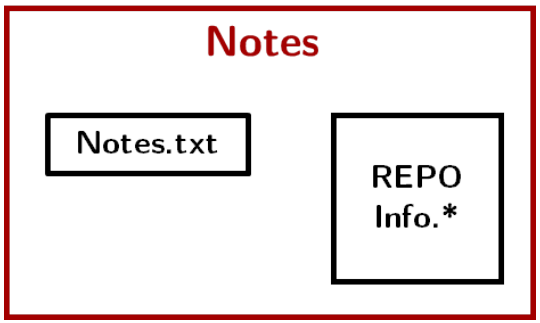
REPO Info. initially empty

History/Commit Example

Commit new file to VCS.



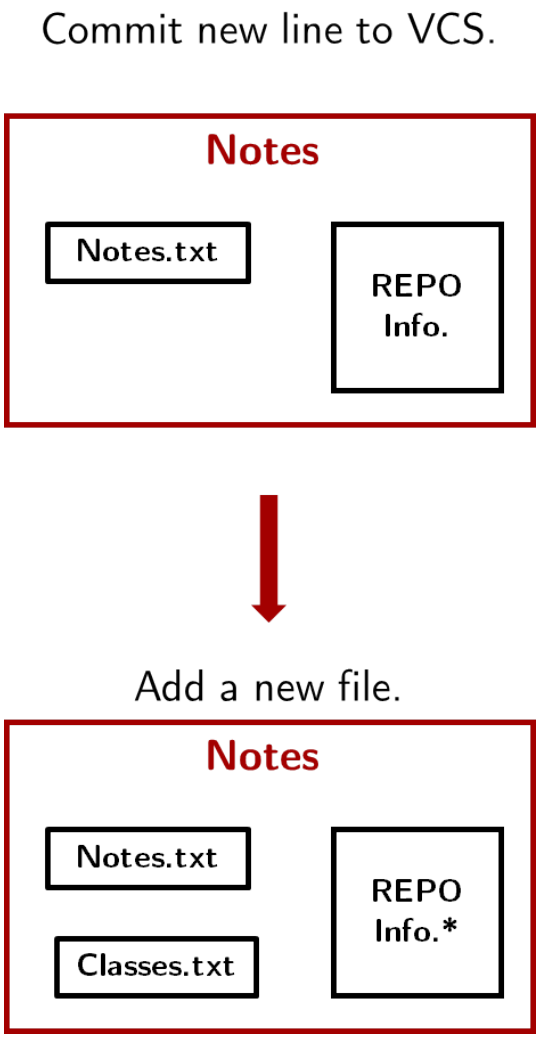
Add a new note to file.



History:

1. 'Notes.txt' created.

History/Commit Example

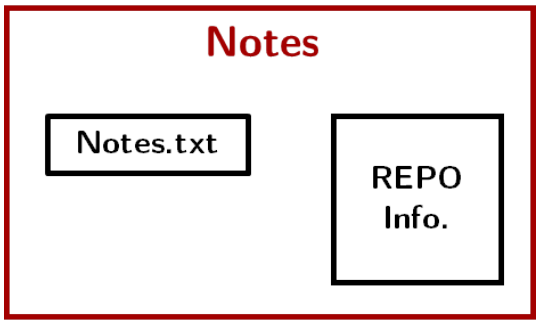


History:

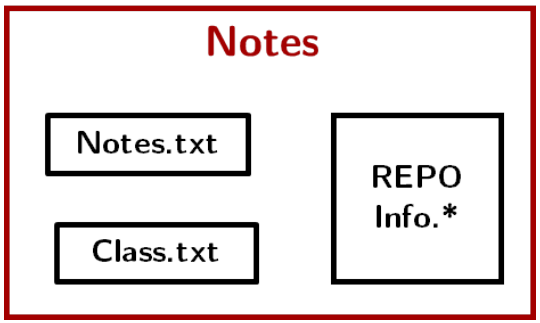
1. 'Notes.txt' created.
2. Added new note to 'Notes.txt'.

History/Commit Example

Commit new file to VCS.



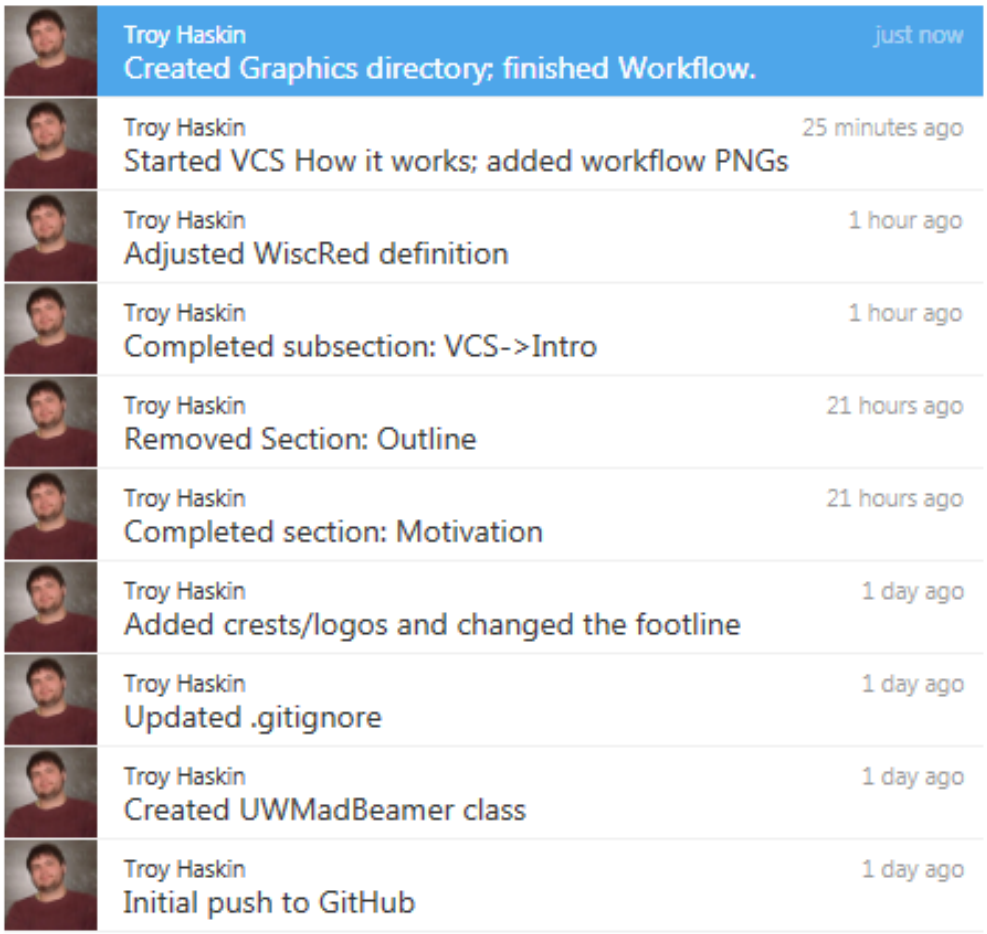
And so on ...



History:

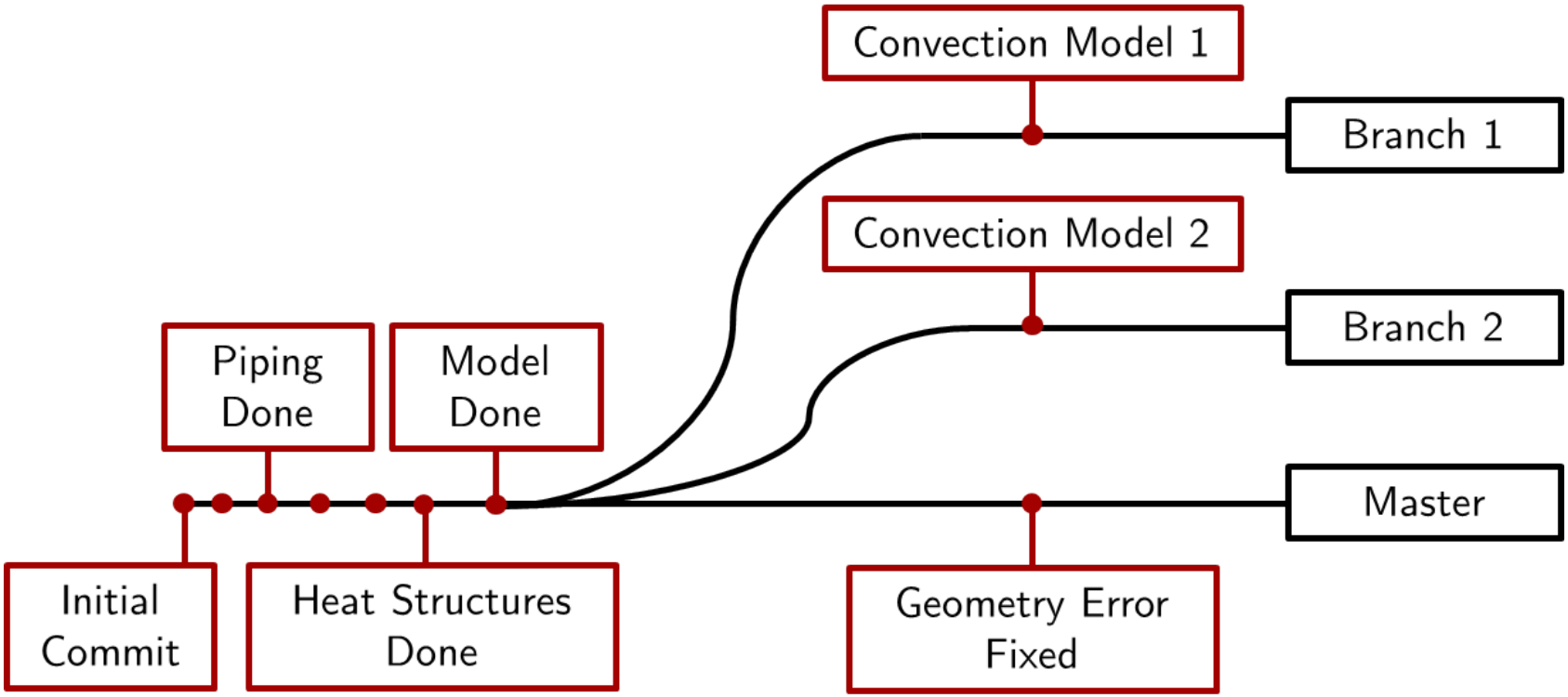
1. 'Notes.txt' created.
2. Added new note to 'Notes.txt'.
3. Added new file 'Classes.txt'.

History of This Presentation



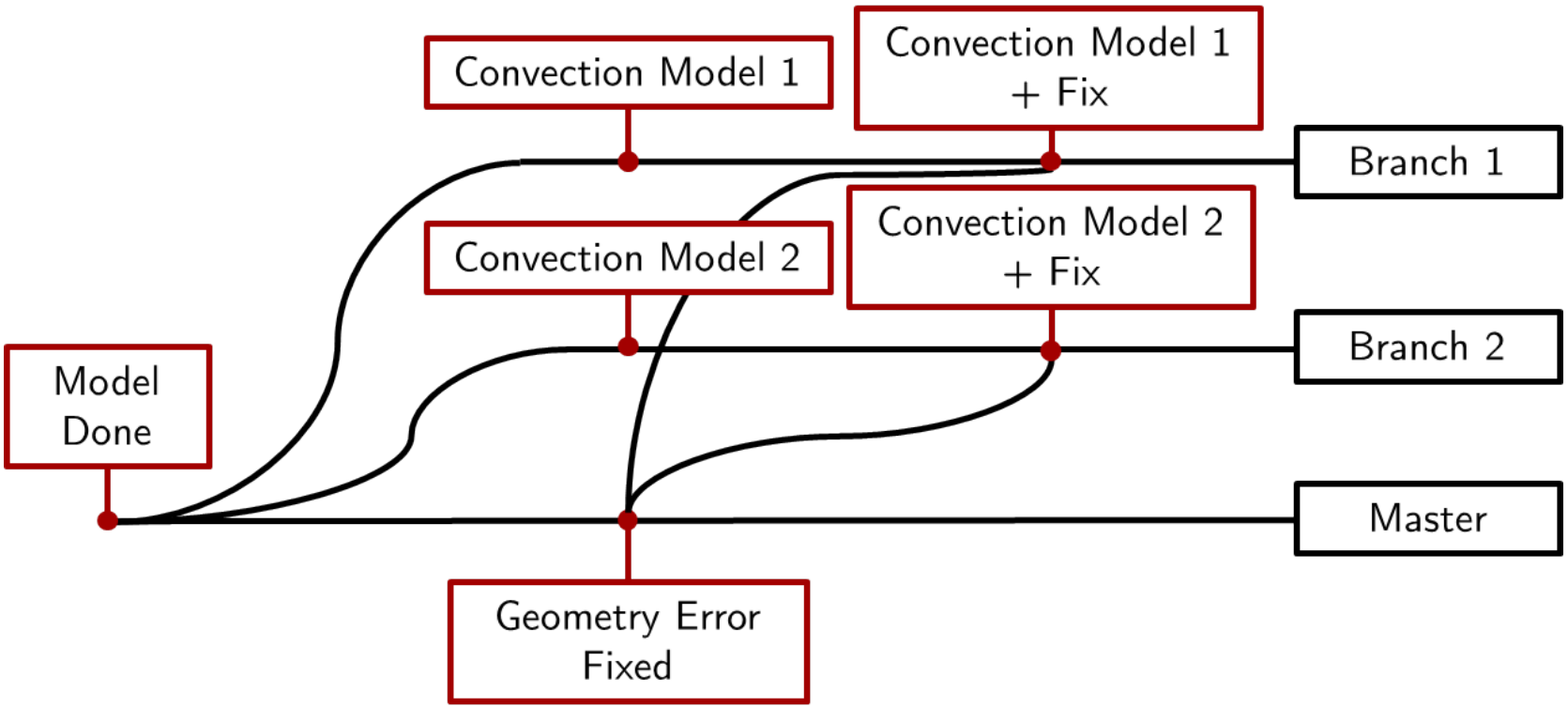
Branching Example

Branches share a common ancestor but can have different histories after the branch commit.



Merge Example

It is possible to merge histories in branches but can lead to conflicts (mismatched or ambiguous histories that require resolution).



Ignoring Files

As was stated before, versioning large binary files is not good practice. Large restart or plot files should be stored elsewhere.

In order to accomplish this, all VCSs have a manner of ignoring files.

For Git, what we will cover next, it involves editing the `.gitignore` file.

That's it

And that covers the broad introduction.

There is more, of course, but that will wait for later.

The Program and the Website

Git My VCS program of choice; created to manage one of the largest collaborative projects in history — the Linux Kernel.

GitHub A website the allows online hosting of repositories and uses Git as its VCS. Public repositories are completely free; private repositories cost money.

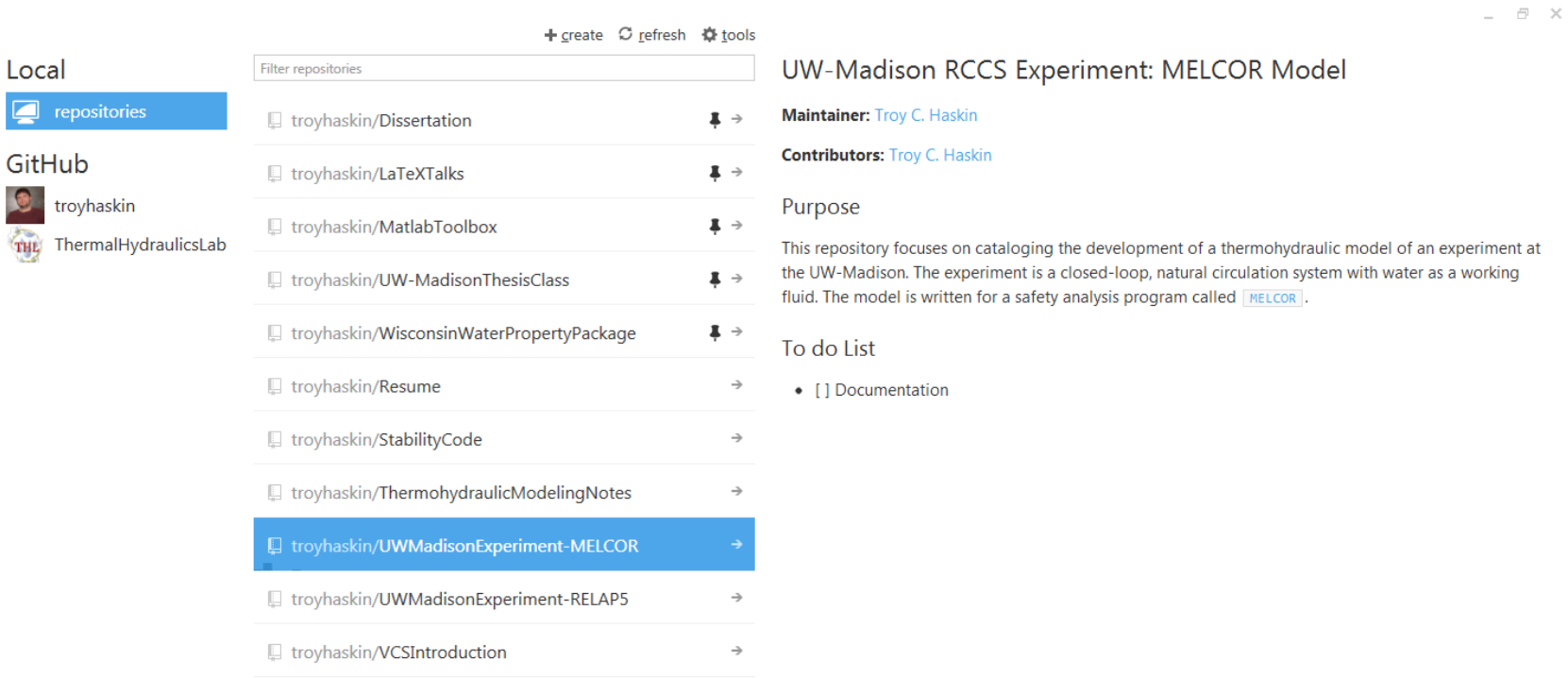
GitHub Applications

GitHubWindows An application for Windows 7/8 that syncs repositories between a computer and GitHub

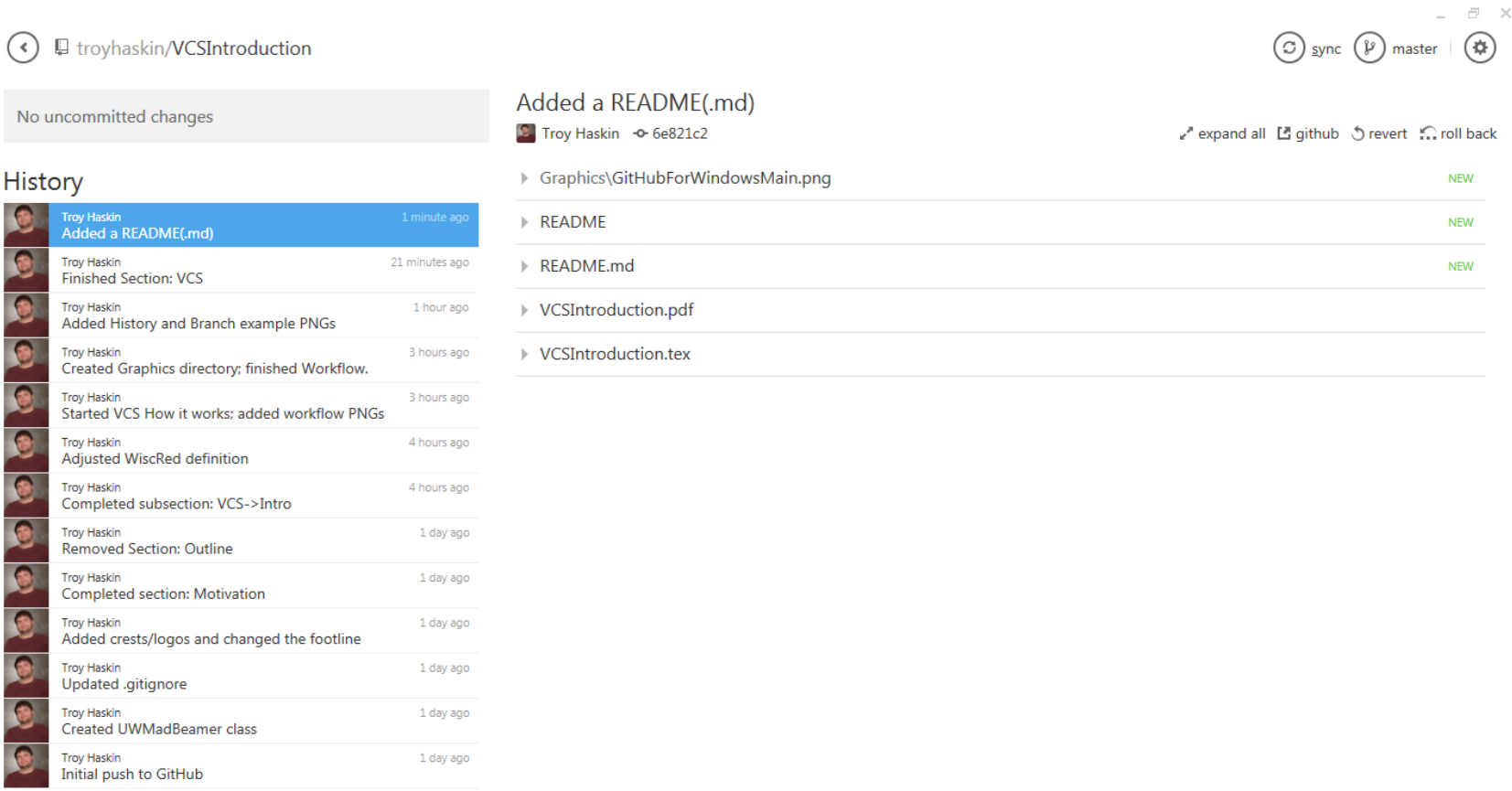
GitHubMac An application for OSX 10.7+ that syncs repositories between a computer and GitHub

Both programs allow for creation, commits, branching, merging, and more.

My GitHub For Windows



My GitHub For Windows



Thank You!

Links:

- Troy's GitHub Page
- THL's GitHub Page
- GitHubWindows
- GitHubMac
- RELAP/MELCOR `.gitignore` file