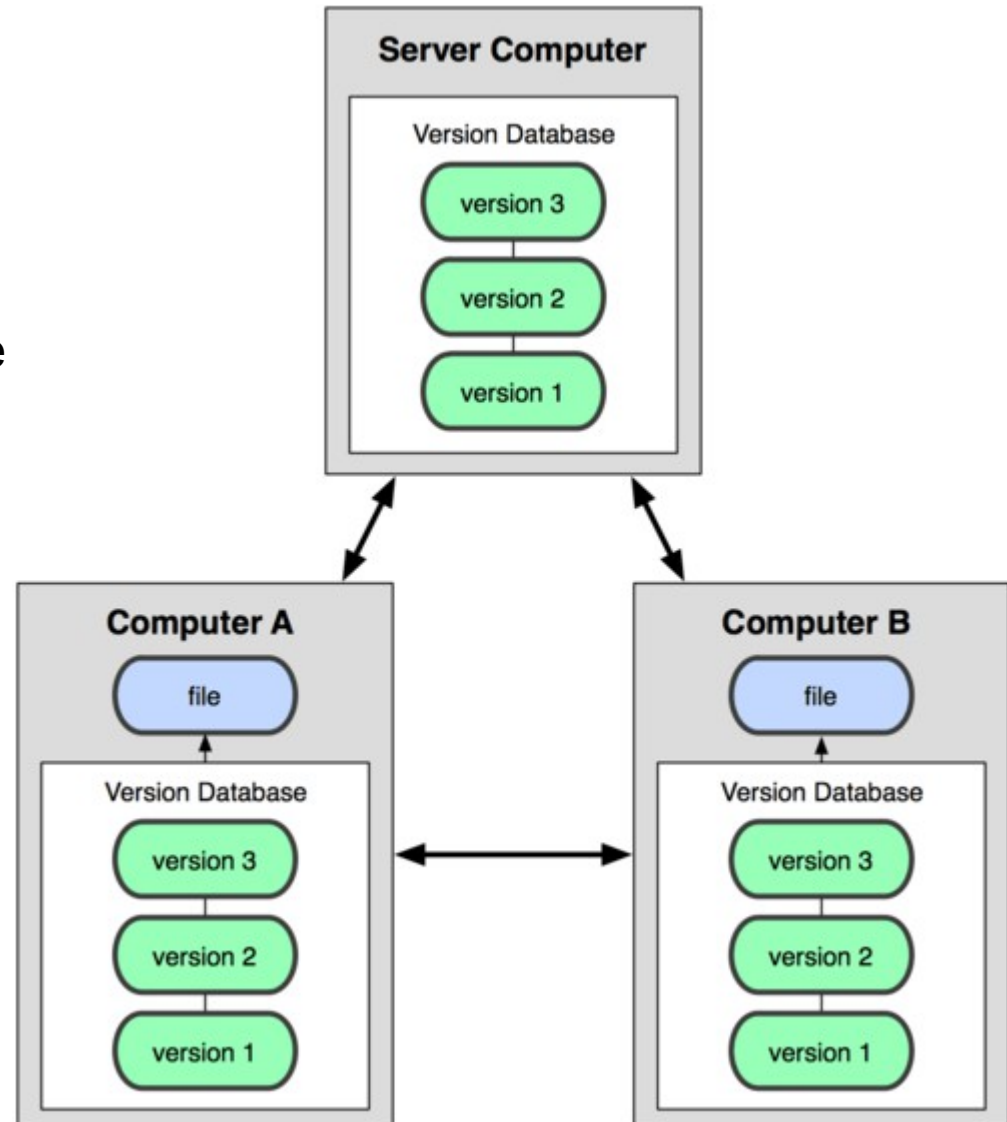


Version control: basic Git tutorial

Presented by
Victoria Rudakova

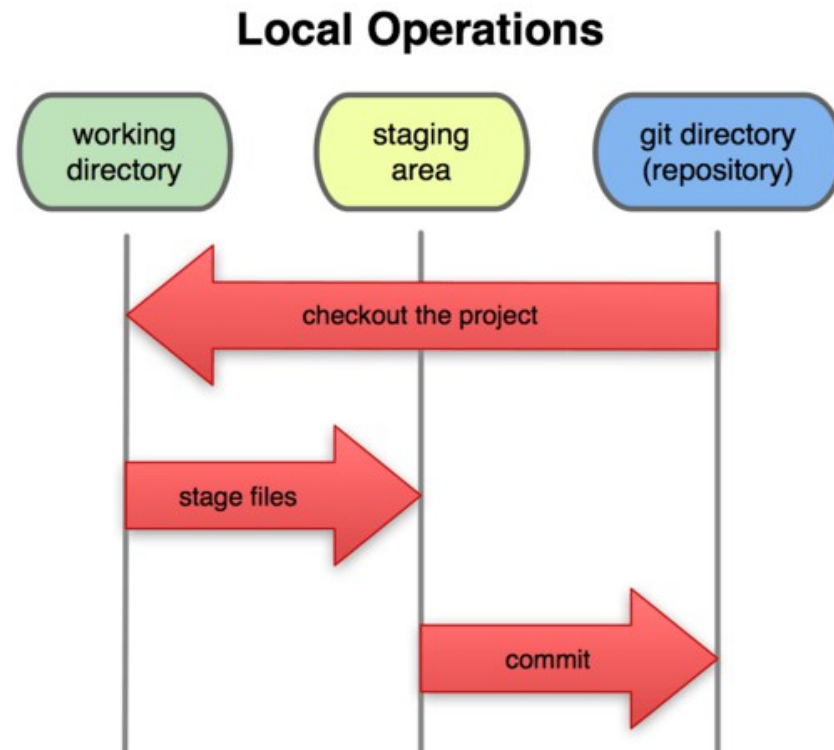
What is "version control"

Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later



Git: the three states

- Committed (stored in local database)
- Modified (file changed but not committed to database)
- Staged (modified file is marked to go into the next commit snapshot)



Git installation

- Windows: <http://www.git-scm.com>
- Linux:
 - apt-get install git
 - yum install git
- Already installed in cygwin

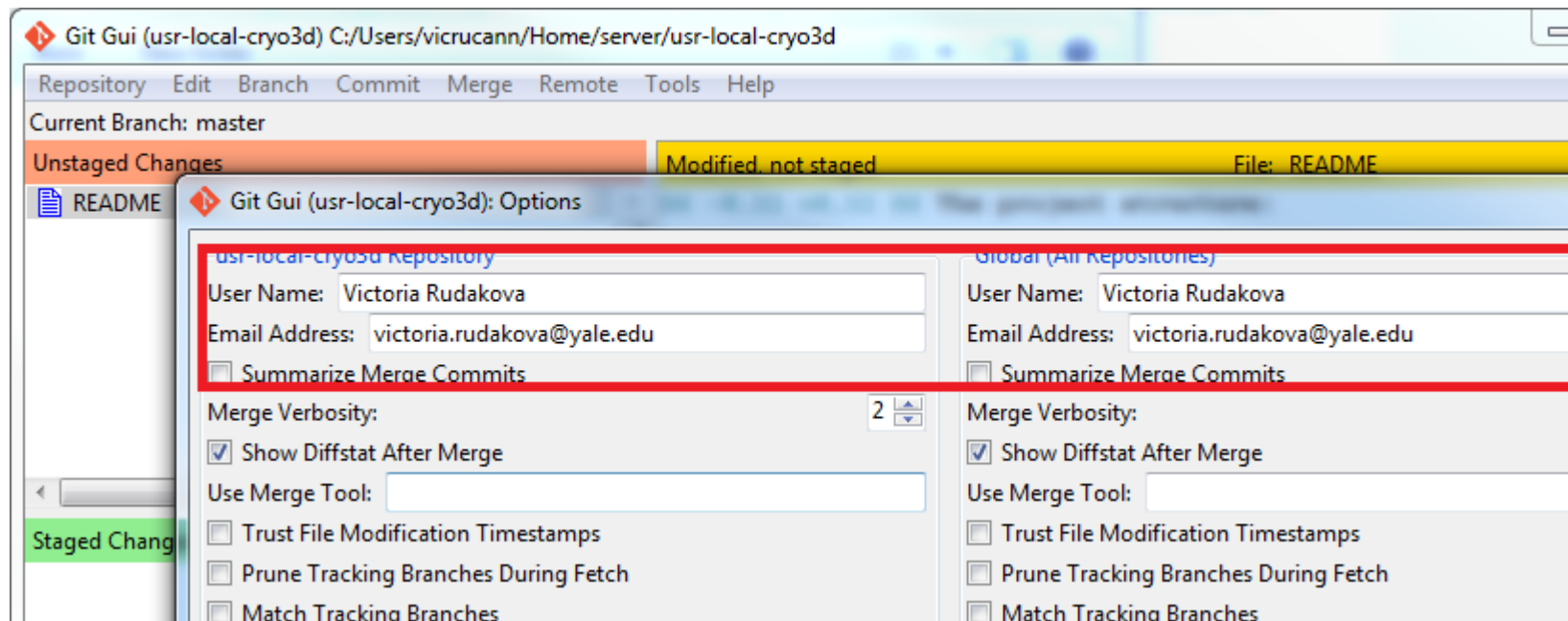
Git config

- Using Git Bash (command line):

```
$ git config --global user.name "Name Surname"
```

```
$ git config --global user.email name.surname@yale.edu
```

- Using Git GUI:



Getting git repository

- To clone existing repository from server2:

```
$ git clone username@172.23.5.77:/usr/local/cryo3d/cryo3d.git
```

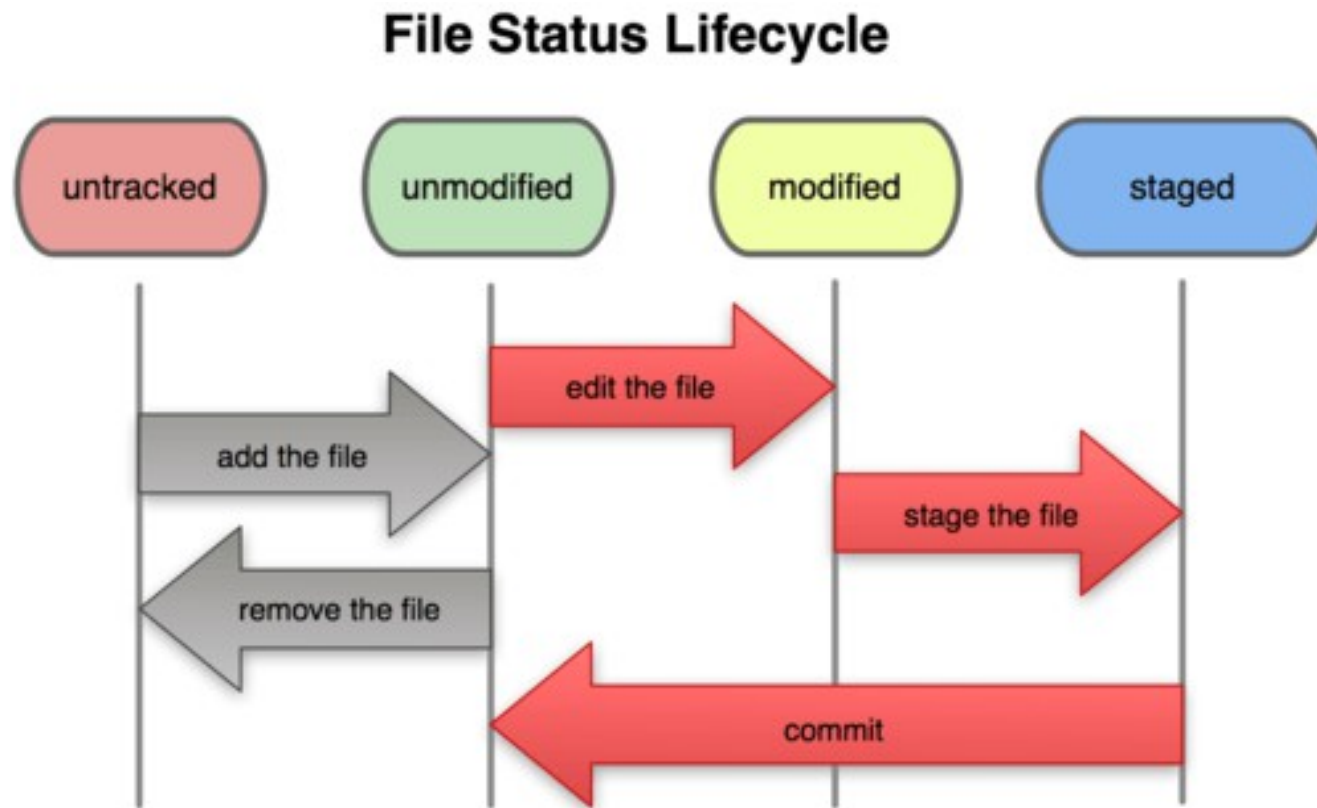
- To start version controlling edited existing (new) files (tracking and committing to local repository):

```
$ git add filename [start tracking new/edited filename]
```

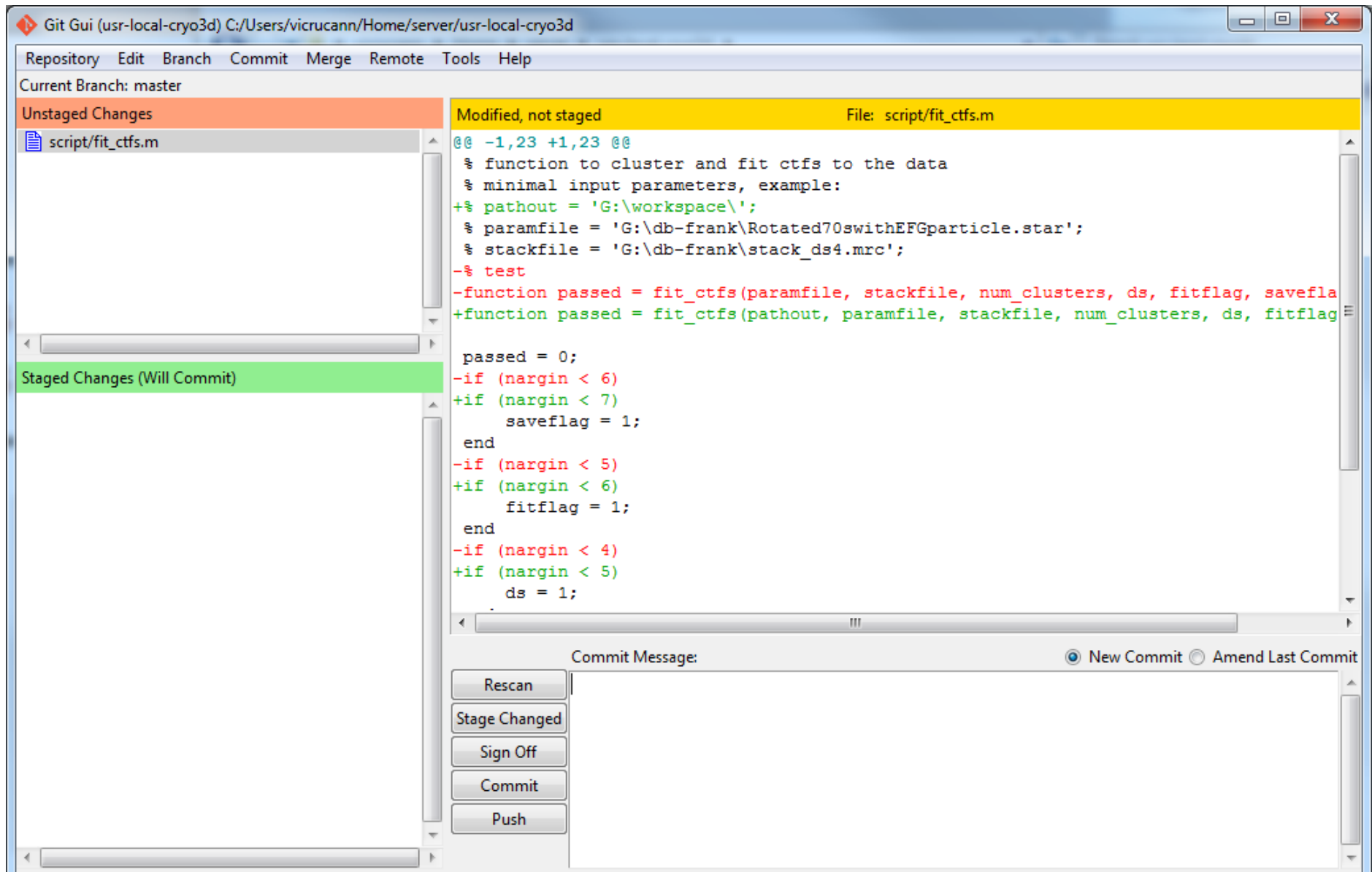
```
$ git add . (git add -A) [start tracking all changed/new files]
```

```
$ git commit -m 'Commit message: what changes were introduced'  
[save changes to the local repository]
```

Recording changes to the repository



Recording changes to the repository (GUI version)



Recording changes to the repository: status

\$ git status

[Check status of your project]

```
$ git status
On branch master
nothing to commit, working directory clean
```

```
$ echo 'My Project' > README
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

  README

nothing added to commit but untracked files present (use "git add" to track)
```

Recording changes to the repository: status

\$ git status

[Check status of your project]

```
$ git status
On branch master
nothing to commit, working directory clean
```

```
$ echo 'My Project' > README
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

  README

nothing added to commit but untracked files present (use "git add" to track)
```

Recording changes to the repository: tracking your files

\$ git add
(directory)]

[Begin tracking a new file

```
$ git add README
```

```
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

   new file:   README
```

Recording changes to the repository: staging modified files

\$ git add [Stage the file, add this content to the next commit]

```
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

   new file:   README

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

   modified:   CONTRIBUTING.md
```

```
$ git add CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

   new file:   README
   modified:   CONTRIBUTING.md
```

Recording changes to the repository: ignoring files

- If we do not want to track automatically generated files (e.g. Log files, build files etc)
- .gitignore file

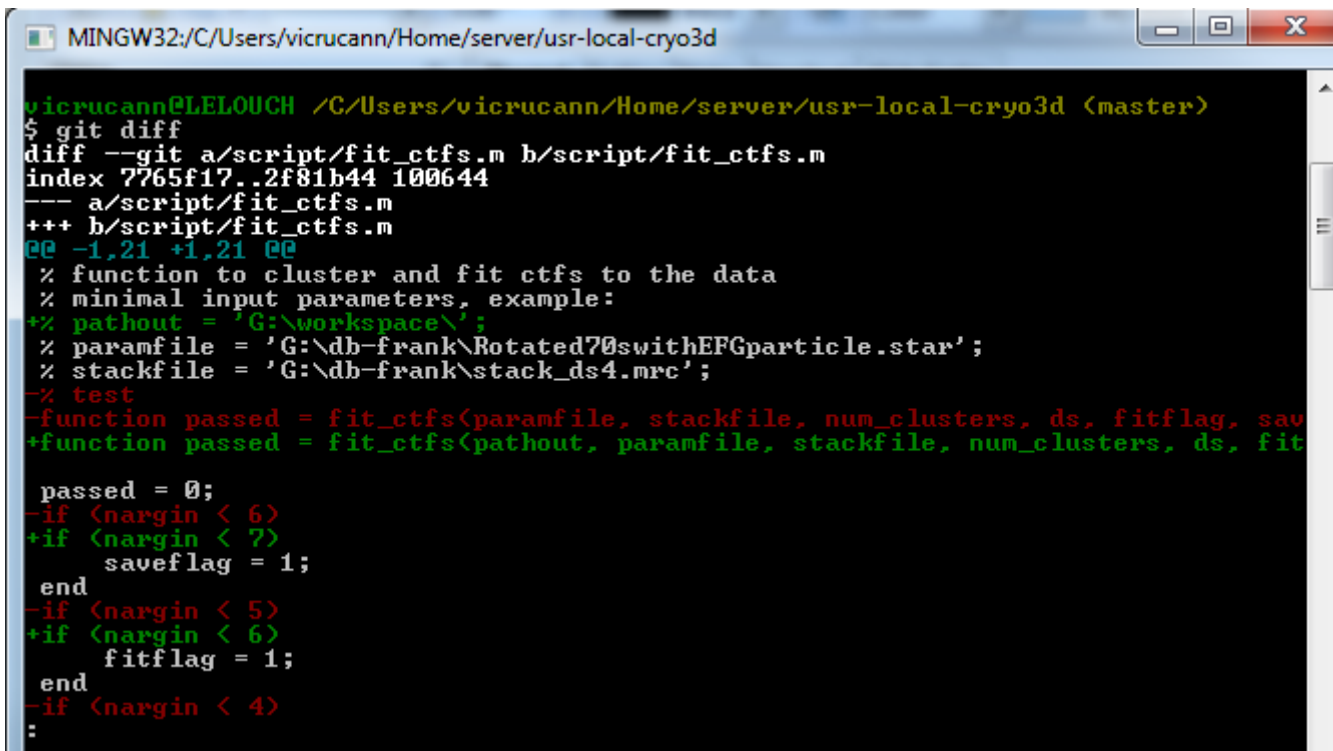
Name	Date modified
.git	2/16/2015 9:26
doc	2/6/2015 5:54
script	2/13/2015 4:04
src	2/11/2015 11:5
test	2/6/2015 5:52
.gitignore	2/10/2015 1:43
README	2/13/2015 1:26

```
.gitignore - Notepad
File Edit Format View Help
# Ignore any newly generated .mat and *.mrc files so that only source files are tracked
*.mat
*.mrc

# Ignore any temporary files
*.asv
```

Viewing staged and unstaged changes

\$ git diff [what changed but not yet staged]

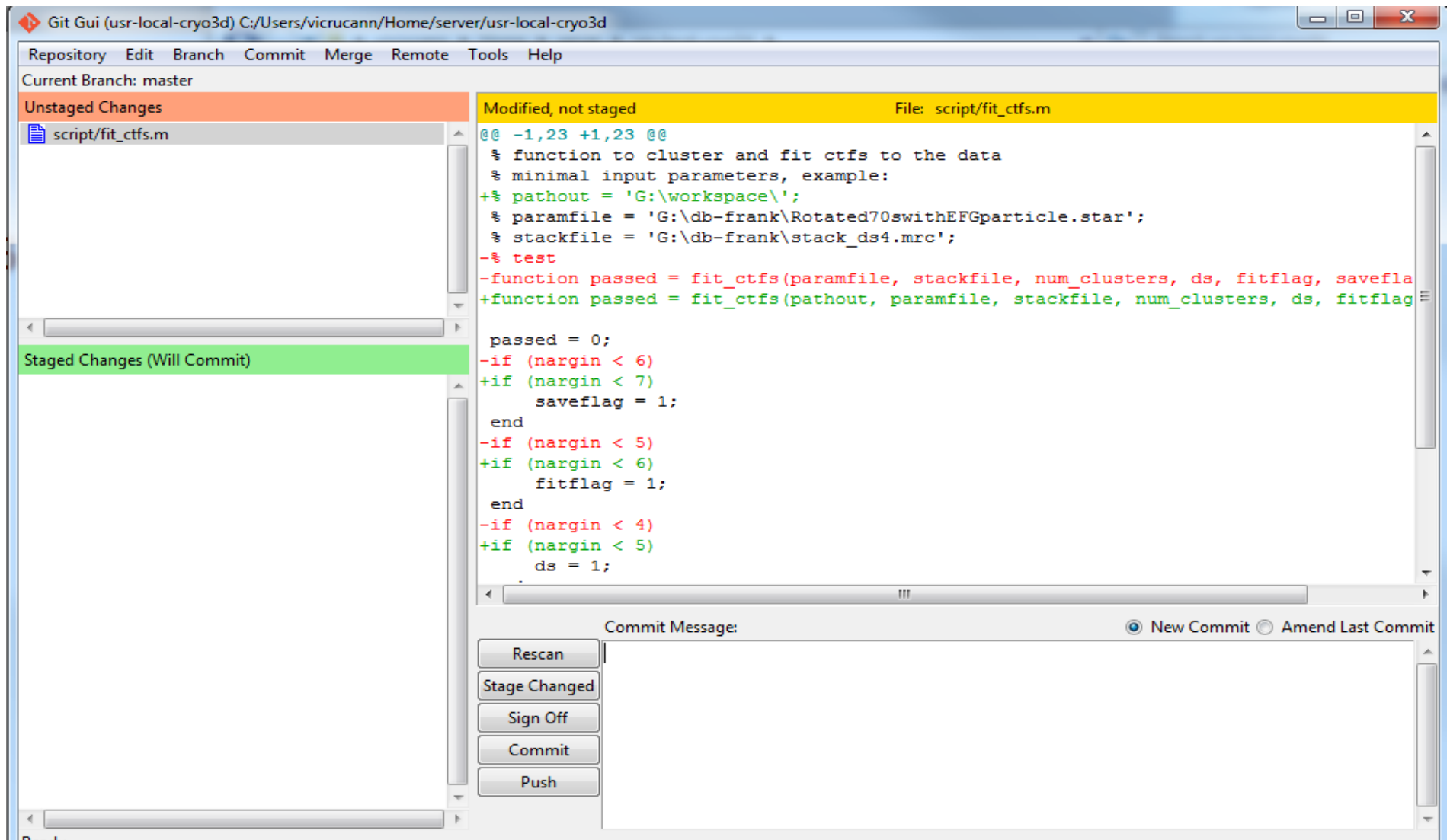


```
MINGW32:/C:/Users/vicrucann/Home/server/usr-local-cryo3d

vicrucann@LELOUCH /C:/Users/vicrucann/Home/server/usr-local-cryo3d (master)
$ git diff
diff --git a/script/fit_ctfs.m b/script/fit_ctfs.m
index 7765f17..2f81b44 100644
--- a/script/fit_ctfs.m
+++ b/script/fit_ctfs.m
@@ -1,21 +1,21 @@
 % function to cluster and fit ctfs to the data
 % minimal input parameters, example:
+% pathout = 'G:\workspace\';
 % paramfile = 'G:\db-frank\Rotated70swithEFGparticle.star';
 % stackfile = 'G:\db-frank\stack_ds4.mrc';
-% test
-function passed = fit_ctfs(paramfile, stackfile, num_clusters, ds, fitflag, sav
+function passed = fit_ctfs(pathout, paramfile, stackfile, num_clusters, ds, fit

passed = 0;
-if (nargin < 6)
+if (nargin < 7)
    saveflag = 1;
end
-if (nargin < 5)
+if (nargin < 6)
    fitflag = 1;
end
-if (nargin < 4)
:
:
```

Viewing staged and unstaged changes - GUI



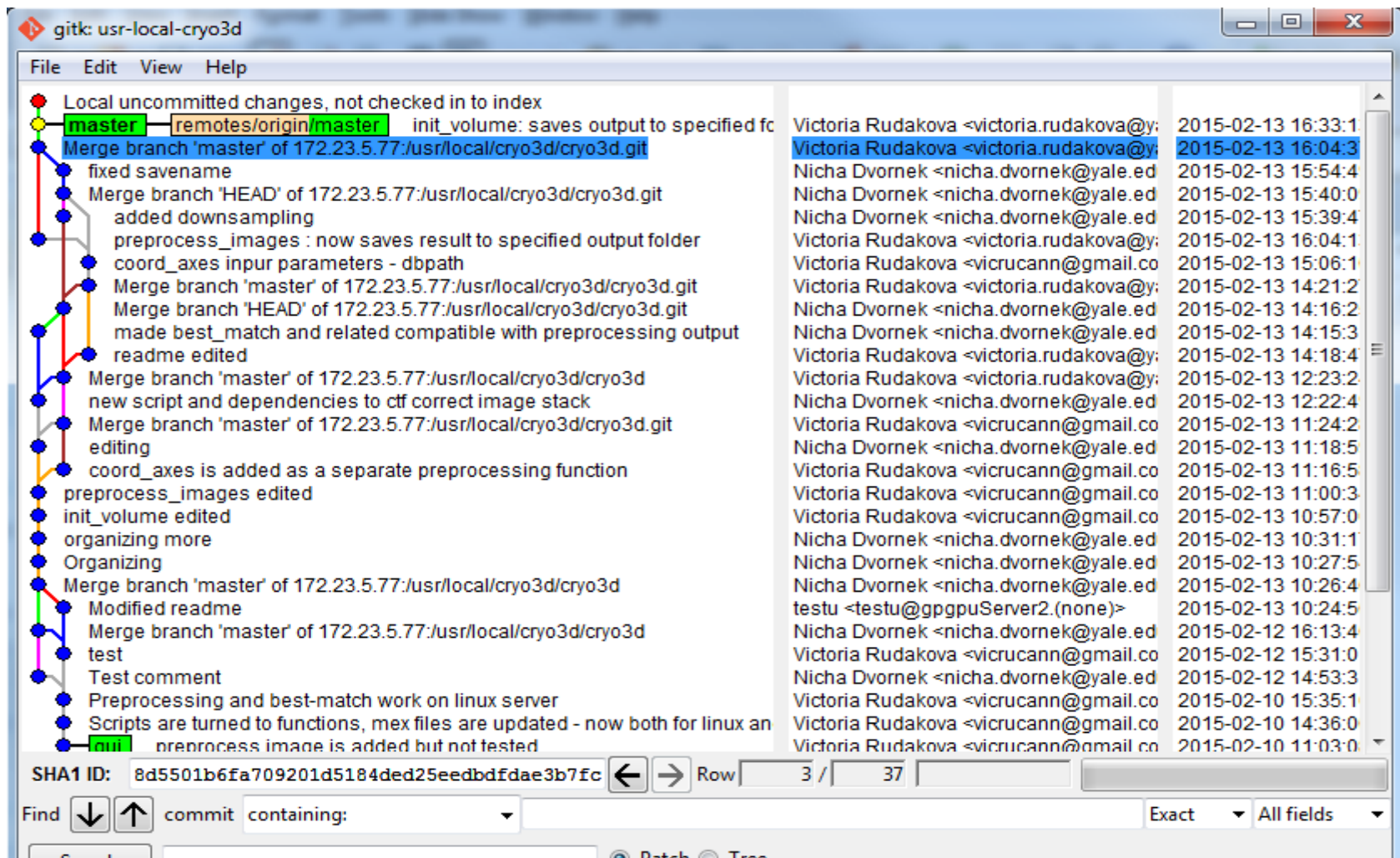
Recording changes to the repository: committing your changes

`$ git commit` [commit your changes to the local repository]

```
$ git commit -m "Story 182: Fix benchmarks for speed"
[master 463dc4f] Story 182: Fix benchmarks for speed
2 files changed, 2 insertions(+)
create mode 100644 README
```


Viewing the commit history

\$ git log



The screenshot shows a git log window with a graph on the left and a list of commits on the right. The graph shows the commit history with branches and merges. The list of commits includes the following information:

Commit Message	Author	Date
Local uncommitted changes, not checked in to index		
init_volume: saves output to specified folder	Victoria Rudakova <victoria.rudakova@yale.edu>	2015-02-13 16:33:1
Merge branch 'master' of 172.23.5.77:/usr/local/cryo3d/cryo3d.git	Victoria Rudakova <victoria.rudakova@yale.edu>	2015-02-13 16:04:3
fixed savename	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 15:54:4
Merge branch 'HEAD' of 172.23.5.77:/usr/local/cryo3d/cryo3d.git	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 15:40:0
added downsampling	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 15:39:4
preprocess_images : now saves result to specified output folder	Victoria Rudakova <victoria.rudakova@yale.edu>	2015-02-13 16:04:1
coord_axes input parameters - dbpath	Victoria Rudakova <vicrucann@gmail.com>	2015-02-13 15:06:1
Merge branch 'master' of 172.23.5.77:/usr/local/cryo3d/cryo3d.git	Victoria Rudakova <victoria.rudakova@yale.edu>	2015-02-13 14:21:2
Merge branch 'HEAD' of 172.23.5.77:/usr/local/cryo3d/cryo3d.git	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 14:16:2
made best_match and related compatible with preprocessing output	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 14:15:3
readme edited	Victoria Rudakova <victoria.rudakova@yale.edu>	2015-02-13 14:18:4
Merge branch 'master' of 172.23.5.77:/usr/local/cryo3d/cryo3d	Victoria Rudakova <victoria.rudakova@yale.edu>	2015-02-13 12:23:2
new script and dependencies to ctf correct image stack	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 12:22:4
Merge branch 'master' of 172.23.5.77:/usr/local/cryo3d/cryo3d.git	Victoria Rudakova <vicrucann@gmail.com>	2015-02-13 11:24:2
editing	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 11:18:5
coord_axes is added as a separate preprocessing function	Victoria Rudakova <vicrucann@gmail.com>	2015-02-13 11:16:5
preprocess_images edited	Victoria Rudakova <vicrucann@gmail.com>	2015-02-13 11:00:3
init_volume edited	Victoria Rudakova <vicrucann@gmail.com>	2015-02-13 10:57:0
organizing more	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 10:31:1
Organizing	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 10:27:5
Merge branch 'master' of 172.23.5.77:/usr/local/cryo3d/cryo3d	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-13 10:26:4
Modified readme	testu <testu@gpgpuServer2.(none)>	2015-02-13 10:24:5
Merge branch 'master' of 172.23.5.77:/usr/local/cryo3d/cryo3d	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-12 16:13:4
test	Victoria Rudakova <vicrucann@gmail.com>	2015-02-12 15:31:0
Test comment	Nicha Dvornek <nicha.dvornek@yale.edu>	2015-02-12 14:53:3
Preprocessing and best-match work on linux server	Victoria Rudakova <vicrucann@gmail.com>	2015-02-10 15:35:1
Scripts are turned to functions, mex files are updated - now both for linux and mac	Victoria Rudakova <vicrucann@gmail.com>	2015-02-10 14:36:0
preprocess image is added but not tested	Victoria Rudakova <vicrucann@gmail.com>	2015-02-10 11:03:0

SHA1 ID: 8d5501b6fa709201d5184ded25eedbdfdae3b7fc Row 3 / 37

Find commit containing: Exact All fields

Working with remotes

`$ git fetch` [fetch all the info you don't have from remote repository, no automatic merging]

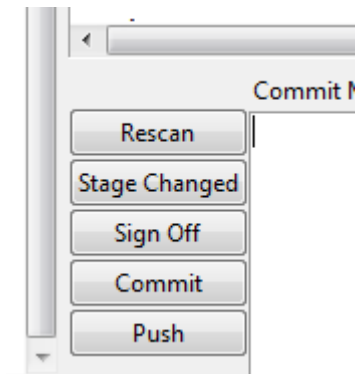
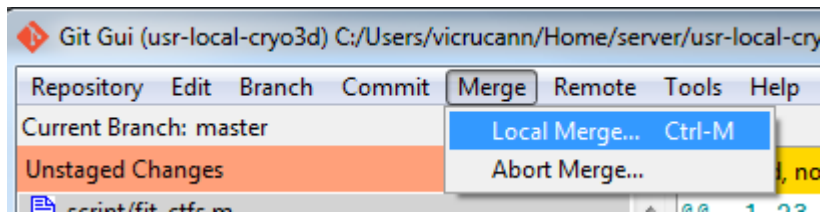
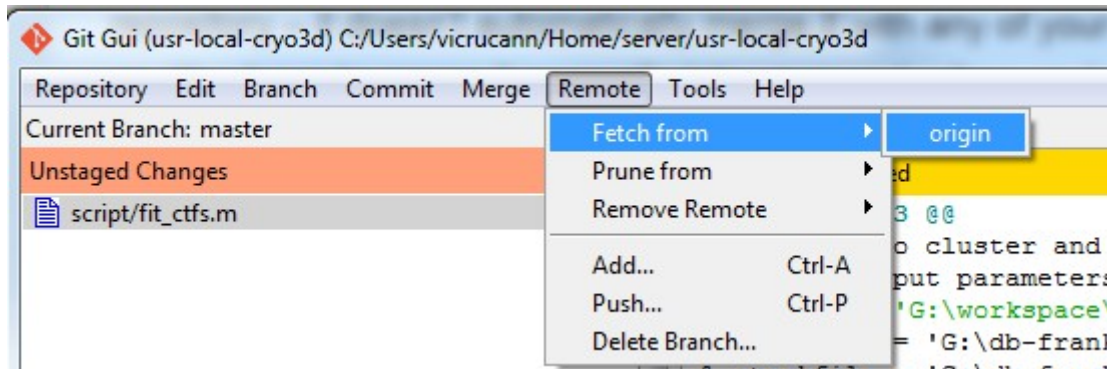
`$ git merge` [automatically merge data from remote with the your repository data]

`$ git pull` [fetch and merge automatically]

`pull = fetch + merge`

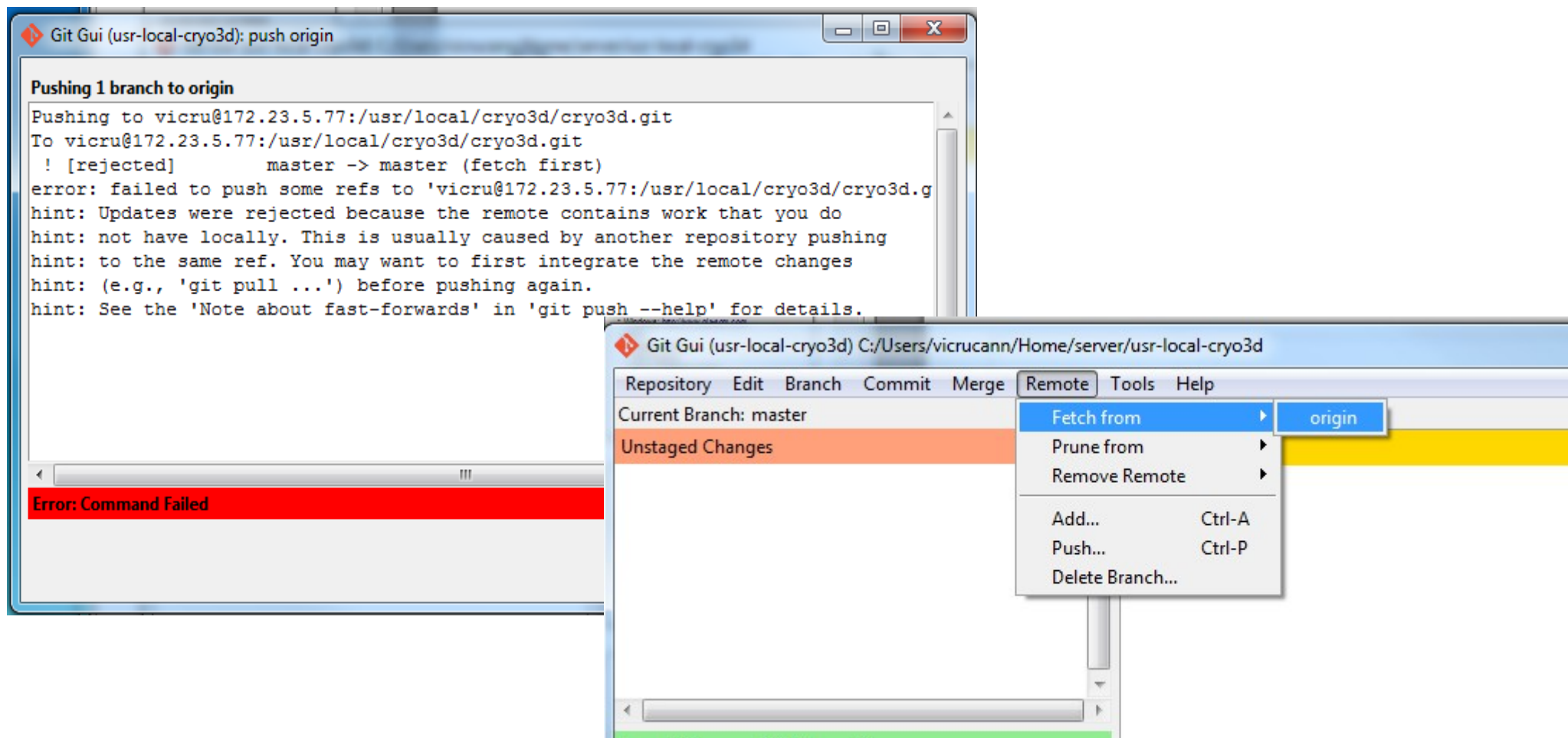
`$ git push origin master` [push your version to the server]

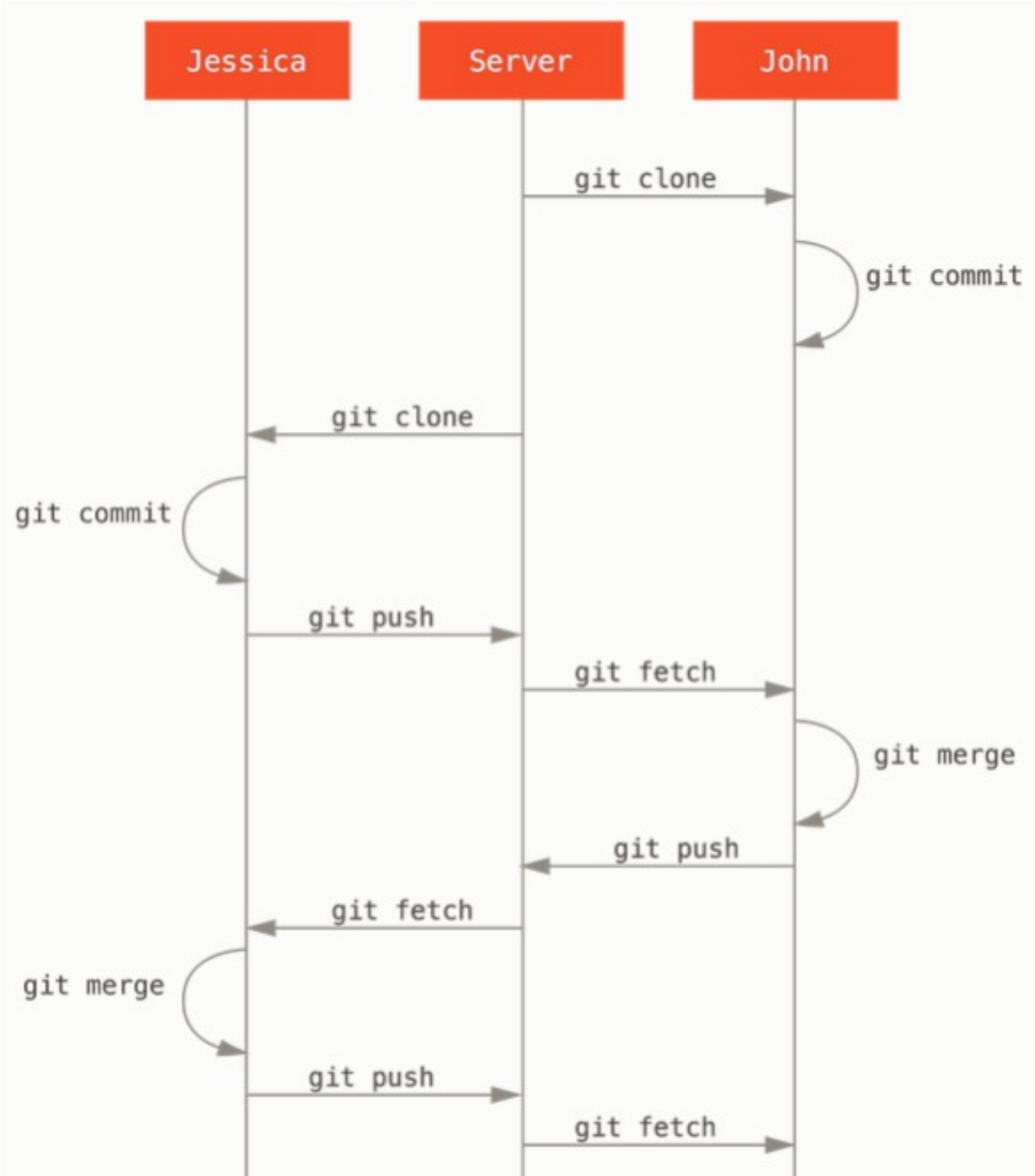
Working with remotes - GUI



Pushing to already changed remote

- Git won't allow to push to the remote which is ahead of your version: first need to fetch





To know more

- <http://www.git-scm.com/book/en/v2>