Using git to develop AMT

March 19, 2014

1 Rationale

Git is a version control system developed by Linus Thorvalds to maintain the Linux kernel. It is a distributed version control system scalable to extremely large projects, because there is not central repository. In Git, every developer can have his/her own repository.

Git has been choosen for the AM toolbox, because *branching* in Git is very easy. This allows each developer to work without disturbing the others, because each sub-project is developed in its own branch.

Even though each developer has his own repository, there *is* a main one, namely the one of sourceforge. This means that we don't using Git in the intended way, using multiple repositories, but instead use a little like SVN with properly working branches!

2 Binary files

Don't put binary files into the toolbox without a clear confirmation from P. Søndergaard

It is very difficult to remove a file once it has been commited: It can easily be deleted, but it is still kept in the Git history. This means that all other developers will have to download you binary file, only to have it be deleted again.

To completely remove a file, you must alter the history to remove it from history. If the branch has been pushed to Sourceforge, you must forcibly push the modified history to Sourceforge overwriting the usual permission, and then all other developers that checkout out your changed must delete their working trees and start over. Alternatively, you must delete your working branch and start a new branch with only the correct files.

If you have not pushed to Sourceforge, it is much easier, you can just throw away you commit.

3 Structure of the AMTOOLBOX Sourceforge repository

The sourceforge repository has several branches. The idea is that you only work on specific things in specific branches.

- The main branch is called *master*. Files in this branch are the ones that gets uploaded by a file release. Don't do *any* new development work on this branch, use it only to fix bugs.
- Development is done in other branches, and when the development is finished you ask on the mailling list to have it merged into *master*.

Git works by having both *local* and *remote* branches. You must connect your local branch to a remote branch in order to track it. For the sake of less confusion, I suggest to use the same names for the local and remote branch.

4 Git on Windows

To use Git on windows install `msysgit` and `TortoiseGit`

5 Getting the code

To get the reposity, you need to clone the Sourceforge one:

```
git clone ssh://soender@amtoolbox.git.sourceforge.net/gitroot/amtoolbox/amtoolbox amtoo
```

On Windows, right-click in where you want the directory, choose "Git clone ..." and then enter the URL above.

This will create a repository `amtoolbox`, which is related remotely to the Sourceforge repository. After this command, change to the `amtoolbox` directory for all further operations.

After this operation, you will only have obtained the `master` branch. To see the other remote branches, type

```
git branch -r
```

To see you local ones, type just

```
git branch
```

On Windows, both these task can be done in the right-click menu TortoiseGit -> Checkout/Branch.

To get the code from any of the remote branches, type

```
git branch --track sti origin/sti
```

This will connect the local branch to the remote branch, both named "sti".

On Windows, again use the "TortoiseGit -> Checkout/Branch" menu, but remember to check the "Create New Branch" box if you are getting a remote branch for the first time. Choose the same name (i.e. `devel`) as the remote branch.

To switch to this branch use

```
git checkout sti
```

The checkout command is always used to switch between branches.

6 Working with the code

To commit, use `git commit`, this is similar to SVN or CVS, only difference is that you only commit to your local tree, and not to Sourceforge. You must *add* your changes before they can be commited, then command `git commit -a` is very usefull for this.

To upload your changes, use `git push`, and to get new updates, use `git pull`.

6.1 Windows tips

- Use `sync`, not `push`, it is easier to handle. If you can not push: `http://code.google.com/p/tortoisegit/issues/detail?id=593`.
- If you don't see the overlay icons in the Explorer: `http://abdullin.com/journal/2009/10/26/fixing-icon-overlays-for-dropbox-tortoisesvn.html` and then restart Explorer

7 End-of-line conversions

Unix and Windows uses different standards to terminate the end of lines in text files. This creates a mess in a version control system. Files are kept in the repository using the Unix standard.

See discussion here:

`http://stackoverflow.com/questions/3206843/how-line-ending-conversions-work-with-git-cor`
`//stackoverflow.com/questions/10418975/how-to-change-line-ending-settings`

- In Git-Bash, "git global –site –list" must result in core.autocrlf=true
- TortoiseGit/Settings/Git/Config: check "AutoCRLF"

8 Tricks when git is messing around

When having problems with pull, you can discard all local changes and get back to the current remote state by:

```
git fetch origin
git reset --hard origin/master
```

9 Tricks for the master

To push a locally created branch to the Sourceforge repository for the first time

```
git push origin branch-name
```

After that, you must edit your local configuration file .git/config. Copy the "master" section and change the name "master" -> "branch-name".

Cleanup:

```
git gc
```

To delete a local branch

```
git branch -D local-branch-name
```

To delete a remote branch

```
git push origin :remote-branch-name
```

To avoid comitting changes to configuration files like `conf.py`, `startup.m` etc.

```
git update-index --assume-unchanged path/to/file.txt
```

9.1 Rebasing

Don't rebase!

To update a development branch with the latest changes in master:

```
git rebase master
```

This will always create a conflict between the local branch and the remote branch. Use

```
git pull --rebase
```

to fix this conflict, and then push again.

10 Getting more help

More help on `git` can be found online, e.g. git book