git(hub) vs. SVN

Introduction

git is a distributed version control system. github is an online service that provides repository hosting and a suite of source code management tools for projects that use git.

Why not SVN?

SVN is a good version control system (keeps track of who changed what, when), but lacks source code management (SCM) features. It works well if everyone is working independently and committing code directly to the main repository with no review.

Since SVN lacks features for moving code between the "working copies" on developer's PCs, users wishing to share code must generate a diff then email it to their collaborator, who in turn will patch their working copy. This practice loses all history of who changed what.

Code also cannot be moved between repositories, making difficult the establishment of an intermediate "sandbox" repository for code review. Such a system would require the reviewer to "diff" the sandbox and "patch" the main repository for every commit.

Why git?

git is one of many distributed version control systems, all of which have a very similar user interface. Other popular DVCSes include Bazaar, Mercurial, and Darcs.

git stands out for a few reasons. First, it has an excellent track record: it was developed by Linus Torvards (author of Linux) and has been used for several years for Linux kernel development. For this reason, it was designed for extremely high performance and fault-tolerance. Corruption is virtually impossible as commits are checksummed against the entire history of the project.

Another major advantage to git is github, an online git repository hosting service. github has developed an impressive source code management suite that makes simple the code sharing and review process that is so painful in SVN.

What does it mean for me?

For most users, moving to git and github will involve changing the commands you run to check out and commit code. A "map" from SVN commands to git commands is included in this document.

For users wishing to share code, life will be much easier. The patches and diffs of SVN are replaced with a few button clicks. Plus, history is totally preserved when sharing code.

What does it mean for SNO+?

The establishment of the SNO+ Code Integrity Committee shows that SNO+ collaborators want more code oversight. With the current SVN system, the only feasilble approach to code review is to have people commit as usual, and have reviewers look at commits after the fact. This means that possibly incorrect or undocumented code is available to the collaboration for some time. Reverting commits is very difficult in SVN, especially since they can "pile up," and there is no mechanism for the reviewer to discuss the problems with the developer.

With github, developers make changes and essentially send a request for review. The reviewer is then presented with a summary of changes and an easy way to merge them in. In case of problems, the request becomes a conversation between the developer and reviewer, a process where the developer can get feedback and change their code to meet review criteria. Since pre-commit review is so simple, it is easy to guarantee a high-quality main repository while promoting good coding practices.