7.5 ECTS Configuration Management – **git lab**

Bring also your CVS and Perforce lab instructions and notes with you as you are going to need them for this lab too. Pen and paper might be useful as you would probably like to make drawings to track what goes on.

Distributed version control systems (DVCS) have gained notice lately – and in this lab we will look at one such system, git. In this context, distributed does not mean that people are distributed (CVS and Perforce can support that without being DVCSs), but that there is no single centralized repository. Instead it is distributed out, so each developer has his own personal repository. An advantage of this approach is that the personal workspace turns into a personal repository with all the versioning functionality we could have wished for. A drawback (for some people) is that it is not so clear how we should/could share our contributions with others (your team).

The purpose of this lab is to look into the concepts and workings of a DVCS, to gain some confidence with the system and to give a short evaluation of the similarities and differences between centralized (CVS/Perforce) and distributed (git) systems. The lab is in three parts. In the first part, you try to figure out how to get started – in the second part you carry out the experiments you also did for CVS and Perforce – and in the third part you go off and explore new things.

PART ZERO:
First of all, you have to work on the same files that you used for the CVS- and Perforce-labs:
http://fileadmin.cs.lth.se/cs/Personal/Lars_Bendix/Teaching/1-ECTS-SCM/CVS/src1.zip

One person should "get those files" and create a git repository in a "convenient place" to contain them. Now he/she wants to "share the project" with the other people in the group. How do they get "copies" of his/her repository? How can you exchange changes between people in your group? Some kind of infrastructure/architecture should be set up so everyone can "see" everyone's repositories. How could you do that? Is there a place that everyone can access?

PART ONE:
Now that everyone in your group has a .git directory, which is their "private" repository – you are ready to start your exploration. I would like you to explore both "normal" work where you enjoy that you have full version functionality in your private workspace. So you should do the following:

1. work alone – status, diff, branches and merging
2. work in parallel – with and without merges and conflicts; try also more than two people
3. experiment a little with awareness possibilities
4. create releases and work with branches

PART TWO:
Explore some of the new and more exotic things in git. Like:

- what happens when you do a git rebase – and why on earth would people need that?
- what does squash (git rebase -squash) do – and in what situations would you like to do that?
- what is the difference between a git fetch and a git pull – and in what situations would it be better to use one or the other?

When you have to work more people, you will have to create more repositories.
If you need more help than the gittutorial(7) Manual Page, you can look here:
http://www.kernel.org/pub/software/scm/git/docs/user-manual.html
git --help command
git help -a

For the benefit of the lecturer, you should send him a short document (*3-4 pages*) that describes the results of your explorations from part one and part two. In addition, I would like you to answer and reflect on these questions:

- your opinion/evaluation of git/DVCS
- how well does git support merge tracking?
- does git support strict long transactions?
- does git support the composition model and the change set model?
- what does the git concept *fast-forward* mean – and why does it exist?
- what does git stash do – and when could it be useful?
- what happens if you *pull* something and you have local uncommitted changes?
- what could happen if you were allowed to *push* something to another person's repository?

In the report you should also state how experienced the members in the group were with using git before this lab.

The document should be in pdf and sent to bendix@cs.lth.se after you have completed the lab.