VERSION CONTROL WITH GIT II

Anton Serjogin
Centre for Information and Speech Processing, LMU
anton.serjogin@gmail.com

26.06.2017

A typical workflow with GIT is the following: getting the current project state from remote, making changes to it, adding and merging the changes locally, fetching and merging the changes from remote and finally pushing the changes to remote. Tagging specific points in history is really important, however, such tags need to be pushed seperately. One of the biggest advantages of GIT is that branching (diverging from the main line of development) is cheap in comparison to ”traditional” version control systems. The default branch name in GIT is *master*, every time you commit, the master points to the last commit made. When creating a branch and switching to it, one must do this explicitely as it is not automatic. In order to merge branches, one has to change to the branch that is needed to be merged into and then merge the changes from the other branch. If merging fails, GIT will inform about it. There are a number of hosting services, for instance, GitLab (open source), GitHub, Bitbucket (third party).