Geophysical Research Abstracts
Vol. 18, EGU2016-13376, 2016
EGU General Assembly 2016

Software Writing Skills for Your Research – Lessons Learned from Workshops in the Geosciences

Martin Hammitzsch

Helmholtz Centre Potsdam - GFZ German Research Centre for Geosciences, Potsdam, Germany
(martin.hammitzsch@gfz-potsdam.de)

Findings presented in scientific papers are based on data and software. Once in a while they come along with data - but not commonly with software. However, the software used to gain findings plays a crucial role in the scientific work. Nevertheless, software is rarely seen publishable. Thus researchers may not reproduce the findings without the software which is in conflict with the principle of reproducibility in sciences. For both, the writing of publishable software and the reproducibility issue, the quality of software is of utmost importance.

For many programming scientists the treatment of source code, e.g. with code design, version control, documentation, and testing is associated with additional work that is not covered in the primary research task. This includes the adoption of processes following the software development life cycle. However, the adoption of software engineering rules and best practices has to be recognized and accepted as part of the scientific performance.

Most scientists have little incentive to improve code and do not publish code because software engineering habits are rarely practised by researchers or students. Software engineering skills are not passed on to followers as for paper writing skill. Thus it is often felt that the software or code produced is not publishable. The quality of software and its source code has a decisive influence on the quality of research results obtained and their traceability. So establishing best practices from software engineering to serve scientific needs is crucial for the success of scientific software.

Even though scientists use existing software and code, i.e. from open source software repositories, only few contribute their code back into the repositories. So writing and opening code for Open Science means that subsequent users are able to run the code, e.g. by the provision of sufficient documentation, sample data sets, tests and comments which in turn can be proven by adequate and qualified reviews. This assumes that scientist learn to write and release code and software as they learn to write and publish papers.

Having this in mind, software could be valued and assessed as a contribution to science. But this requires the relevant skills that can be passed to colleagues and followers. Therefore, the GFZ German Research Centre for Geosciences performed three workshops in 2015 to address the passing of software writing skills to young scientists, the next generation of researchers in the Earth, planetary and space sciences. Experiences in running these workshops and the lessons learned will be summarized in this presentation.

The workshops have received support and funding by Software Carpentry, a volunteer organization whose goal is to make scientists more productive, and their work more reliable, by teaching them basic computing skills, and by FOSTER (Facilitate Open Science Training for European Research), a two-year, EU-Funded (FP7) project, whose goal to produce a European-wide training programme that will help to incorporate Open Access approaches into existing research methodologies and to integrate Open Science principles and practice in the current research workflow by targeting the young researchers and other stakeholders.