The Software Paper

Edzer Pebesma[*]

Version 0.1, August 13, 2014

1 Introduction

Today, many scientific papers are based upon scientific computing, which in turn is based on using scientific software. Increasingly, developing original scientific software is an integral part of the process that leads to obtaining new scientific insights The *research software impact manifesto*[1] sais that *software has become the third pillar of research, supporting theory and experiment.*

The *software paper* is a specific type of scientific paper that describes an original scientific software contribution to (i) helps other scientists reproduce previously published and carry out new research, (ii) increase the transparency of (previous and future) scientific research, and (iii) allow proper citing of the software contribution when used in consecutive papers. This paper describes the software paper, its rationale, its requirements, and recommendations. It may yield a blueprint for software papers published in *Computers & Geosciences*.

2 Rationale

(How does the software paper differ from standard documentation?)

manual - tutorial - design paper

software paper describes the originality or scientific contribution, describes the context

3 Requirements

originality the paper should point out why the software contribution is original, and how it improves over existing solutions: why was there a need to develop this software.

context the paper should point how the software developed relates or compares to existing solutions, and how existing solutions were integrated or re-used.

[*]`edzer.pebesma@uni-muenster.de`

[1]`http://software.ac.uk/blog/2011-05-02-publish-or-be-damned-alternative-impact-manifesto-research-software`

source code a pointer to the source code, either a file with a tar ball, or a repository (git, subversion).

availability the software should be made available to the reviewers, meaning that it is part of the paper at time of first submission.

readability the software should should be readable, have comments in place where relevant, and be written having in mind that others would want to understand it.

version version number of source code, revision ID of SCM system

examples should show input, interaction (if any), and output; the examples should be easily reproducible with the source code available.

citing the paper should cite all relevant software mentioned, or used to build the software described.

portability the paper should describe on which operating systems and/or computing platforms the software runs

copyright, license the paper and software should both clearly state where the copyrights lies, and under which license it is distributed

3.1 Requirements to reviewers

Reviewers should primarily review the paper, but also address the software described. Minimum checks include:

- can the software be downloaded?
- can the software be installed, and run?
- can the examples be run?

The reviewers should address the question whether the contribution of the software paper is original, substantial, and sufficiently embedded in the context of existing solutions and/or embedding environments.

4 Recommendations

Author(s) should collect user feed-back before publication; report on user feed-back. Discuss usability.

((See Hyndman blog))

5 Discussion

Persistence of web sites, and of links

6 Links

- http://software.ac.uk/so-exactly-what-software-did-you-use
- http://robjhyndman.com/hyndsight/jss-rpackages/