Software Management Plans in Research Projects
(the slides are available under “Presentation Materials” in the above URL)
Date: August 14, 2019
Presented by: Shoaib Ahmed Sufi (UK Software Sustainability Institute)

Q. In a research institute, what other people should control or modify the SMP document? Your supervisor? Other parties? Is there any reason not to include it with the source code (discussion on Slide 10 suggests Zenodo)?

A. Will talk about lifecycle shortly. SMP is a plan about what you will do, whereas software is more about what you have done; a manifestation of the plan. No harm adding to the source code. Other people might want more flexibility of managing that document separately.

Q. The contribution to research of a new piece of software would require the comparison with existing software. Should a state-of-the-art report be part of an SMP? And how would you find out about existing research software for your use case? How do you **find** existing software? (<- for research software, typically published literature?<<so you require an article for every piece of important software? << Unfortunately, that’s academic tradition at least in my field :/ They also measure importance by citations for continued funding << is there a citable publication on exactly this kind of funding justification?)

A. SMP should refer to state-of-the-art report being part of the proposal.

Q. Typically who should do the software evaluation other than the project lead? Some software developers treat their products like children and can be resistant to criticism <3

A. If you don’t have anyone to do it, the project should evaluate itself; not the best thing. In the UK there is the growing notion of the Research Software Engineer (https://society-rse.org/): people with a research background who help researchers improve their software. They can help run software evaluations. Reciprocal help of reviewing each other’s software can help people averse to criticism and help not take feedback personally.

Q. You mention on slide 7 about the SMP measuring how helpful the research software is. What are typical metrics for doing this? There is some discussion in question 2 above about citation count, but what are other good measures of helpfulness and impact? The question was more about how do you decide the importance and impact of the software you’re making or evaluating.

A. Slide 9-10: the point I was making about assessing how you’re doing is to encourage software evaluation. Primarily the Importance should be assessed for your needs: does writing the software allow you to do things you could not have done before? Model a larger system

than before? Your own perspective. From an external perspective, unless you're particularly skilled, it's about how well the piece of software your are evaluating solves your specific problems.

Notes:

- Software Management Plan can mean different things to different people. By SMP we mean:
  - "Statement of intent"
- (Slide 7) SMP contains
  - What problem your software solves
  - How your software fits into the broader area
  - Audience
  - Level of support
  - How you measure how helpful it is
- (Slide 13) Even for a running project, it can serve as a review of software assets. Helps one take a more consistent approach over personal repositories: whether to use GitHub's wiki, RTFD, licenses, etc. More than source code: how do you give different people credit?
- (Slide 14) Meant to be a living document. Visit every ~3-6 months. Version them! Project lead is responsible for making sure SMPs are implemented and used.
- (Slide 15) Software Evaluation - about what you are doing, not what you are planning to do. What you "think" you're doing.
- (Slide 18) CHAOSS metrics are one way to assess, but more suited to larger software which have started to have a community.
- (Slide 20) Laurent Gatto suggests an output management plan that includes SMP with data management plan (DMP).
- (Slide 26) Since 2003 NIH Research Sharing Plan (RSP) includes DMP.
- (Slide 31) Primary benefit is for you!.