WSSSPE Working Groups – Summaries

WSSSPE4 (Manchester, 2016): 12 Working Groups have formed to work on different topics during, and beyond, the workshop. If you want to join (or restart) a Working Group, see the WSSSPE4 report (`https://arxiv.org/abs/1705.02607`) for details.

1. Verifying best practices & metrics for sustainable research software
2. Software Sustainability Alliance
3. Scientific Software Prototyping Infrastructure (S2PI)
4. CodeMeta
5. White paper on developing sustainable software
6. Social science for scientific software
7. Software best practices for undergraduates
8. Meaningful metrics for sustainable software
9. Coordinating access to continuous integration (CI) for research software
10. Software engineering processes tailored for research software
11. Open research index
12. Letters of evaluation for computational scientists

(1) Verifying best practices & metrics for sustainable research software

The Working Group addressed the following questions:

- Are the suggested software engineering for science/academia best practices verified when evaluated against open source research software projects that currently use modern software engineering?
- Are the suggested metrics for science/academia verified to be relevant when evaluated against open source research software projects that currently use modern software engineering?
- Do research groups have role models to follow for successful best practices in research software?

(1) Verifying best practices & metrics for sustainable research software cont.

The team identified the following research projects to use in addressing the above, though they did not get further than this:

- FLASH (astrophysics, `http://flash.uchicago.edu/site/flashcode/`)
- GROMACS (chemistry, `http://www.gromacs.org/`)
- CIG (Earth science, `http://geodynamics.org/`)
- CUAHSI (Earth science, `https://www.cuahsi.org/`)
- CSDMS (Earth science, `https://csdms.colorado.edu/wiki/Main_Page`)
- OntoSoft (biology, `http://www.ontosoft.org/`)
- ASCL (astronomy, `http://ascl.net`), specifically ASCL entries sorted by citation count16
- yt-project (astronomy, `http://yt-project.org`)
- Einstein Toolkit (physics, `https://einsteintoolkit.org/`)
- SpEC (physics, `https://www.black-holes.org/SpEC.html`)
- pyCLOUDY (astrophysics, `http://ascl.net/1304.020`)

(2) Software Sustainability Alliance

The Software Sustainability Alliance working group aims to establish an alliance between organizations interested in improving the sustainability of research software. Such an alliance would ease the collaboration between member organizations to improve the sharing of expertise, resources and best practices. It is currently seeking feedback on potential member organizations, as well as the aims and scope of this alliance.

`http://softwaresustainability.org/`

(3) Scientific Software Prototyping Infrastructure (S2PI)

There is a productivity bottleneck in HPC pertaining to the development of simulation tools: few domain scientists develop simulation software, even fewer know how to prototype such tools. This group attempts to tackle this challenge through the development of novel software prototyping infrastructure with an emphasis at a higher degree of abstraction. The group deems the ability to rapidly construct software artifacts that can be trusted in terms of the methods from their design up, easily discarded when wrong and extended when right at low human and computational cost to be one aspect of scientific software sustainability. In essence, a motto for scientific software prototyping is fail hard, fail fast.

(4) CodeMeta

This group, which included people who had previously been working on the CodeMeta project, formed to address the proposed project, "Define a standard for metadata to be used in current repos including authors, dependencies, . . . , repo URL."
CodeMeta is an active project (`https://codemeta.github.io/`), and is creating a minimal metadata schema for science software and code, in JSON and XML. The goal of CodeMeta is to create a concept vocabulary that can be used to standardize the exchange of software metadata across repositories and organizations. CodeMeta started by comparing the software metadata used across multiple repositories, which resulted in the CodeMeta Metadata Crosswalk. That crosswalk was then used to generate a set of software metadata concepts, which were arranged into a JSON-LD context for serialization.
The CodeMeta schema has been released in version 2.0 in 2017 (`https://doi.org/10.5063/schema/codemeta-2.0`).

(5) White paper on developing sustainable software

Many diverse aspects and dimensions of developing sustainable software can be investigated, such as economic, technical, environmental, and social. This group aims to write white papers that will focus on scientific environments and their implications, targeted at developers and project managers of scientific software. Given the complexity of this field, it is important to select a subset of sustainability aspects for the white papers. The idea is to create a series of papers instead of trying to tackle all important topics in one paper. For the first white paper, the group aims to set the stage with successful use cases and an analysis of why they have been successful. Further topics will include community-related practices, government and management, funding, metrics, tools, and usability. The group will collect feedback from the WSSSPE community on this first white paper and extend it to a journal paper.

Update: The group is finalizing the first version.

(6) Social science for scientific software

This working group is motivated by the goal of building better connections between academic researchers who are studying topics in or relating to software sustainability with practitioners, managers, and administrators who are working in the area of software sustainability. In any domain, bringing research and practice closer together is a mutually beneficial goal, but also has its challenges. This group met to discuss existing research projects and findings that might be relevant for RSEs and others in the software sustainability domain, then identified several gaps and challenges to tackle in bringing research and practice together.

(7) Software best practices for undergraduates (Raniere Silva)

Participants: Jonah Miller, Aleksandra Nenadic, Raniere Silva, Francisco Queiroz, Hans Fangohr, Prahbjyot Sing
Motivation: We were motivated by the perceived prevalence of "hidden code" in scientific communities: code written by researchers in an unsustainable way that is never shared with the larger community.
Original objective: To implement a course describing software best-practices aimed at domain science.
What we actually did: We were unable to muster the time or momentum to facilitate our original goal. However, thanks to the leadership and efforts of Francisco Queiroz, we were able to come together to write a paper discussing best practices for user interface design and to provide some recomendations.
Our recomendations help address hidden code, if they are ever read by authors of hidden code. However, a course curriculum would still be valuable.
Next steps: Return to the original SMART steps we mapped out.

(8) Meaningful metrics for sustainable software

Meaningful Metrics for Sustainable Scientific Software aims to increase the visibility of the quality of scientific software, facilitate the reusability of scientific software, and promote the best software practices by standardizing metrics via interviews with scientific software developers. This working group believes improving the current software metrics system will increase software sustainability. Currently, there are inefficiencies regarding software duplication, sustainability, and selection, as well as others, within the scientific software community. In order to address these inefficiencies, Meaningful Metrics for Sustainable Scientific Software aims to create a goal-oriented method to collecting productive metrics by focusing on the developer side of software.

Goals: To identify the population of scientific software developers who were willing to be interviewed $\rightarrow$ Form and categorize interview questions $\rightarrow$ Interview participants $\rightarrow$ Map the results to goals $\rightarrow$ Convert goals to metrics $\rightarrow$ Analyze collected metrics

(9) Coordinating access to continuous integration (CI) for research software

Each developer of software with uncommon needs (hardware, software, libraries, data sets), non-public code, or tests that exceed time limits for free plans must acquire, setup, and maintain their own continuous integration systems because their needs make them ineligible for popular free services such as Travis CI. For example, software groups that develop BIOUNO, CI4SI, or GROMACS have done this. Debian provides a testing infrastructure (https://ci.debian.net/doc/) that is mature and supports hardware and other requirement specifications.

Objectives: This group is interested in reducing the burden of different projects having to build and maintain their own continuous integration systems (when publicly available CI are not a fit), by coordinating and sharing this burden across multiple projects.

(10) Software engineering processes tailored for research software

This working group is concerned with identifying processes that are not adequately covered by general software engineering. Verification and testing is the first subtopic that it addresses. Computational science and engineering applications have many moving parts that need to interoperate with one another. The accuracy and reliability of results produced by scientific software depends not only on the individual components behaving correctly, but also on the validity of their interactions. As scientific understanding grows, the corresponding computational software models are refined, leading to more complex codes. Increasing complexity makes them more prone to defects, not only in individual code units, but also in interaction among units. Therefore, a strong verification process combined with a rigorous testing regime plays a critical role in the prevention of generating incorrect scientific results. However, most science teams struggle to find a good solution for themselves. Causes range from lack of exposure to the practices, to distrust of adopting practices because they do not meet the needs of the teams developing such software. The current focus of our effort is on testing because it is the first step towards building a software processes that can lead to provenance and reproducibility, the hallmarks of quality science.

(11) Open research index (Daniel S. Katz)

The aim of this group is to investigate the building of an index of research products in an open sustainable manner. Its goal is not to eliminate commercial products, but to build on what is there and provide data and services that are missing.
The Open Research Index should take in all research products (papers, software, datasets, workflows, etc.) from their publishers and recorders (journals, societies, domain repositories, government [open access] repositories, preprint servers, general repositories [e.g., figshare, zenodo]) and other services (CrossRef, ORCID). Each product should list authors and citations and allow people to search the resulting network. Users should also be able to interact with their own record and edit it, like Google Scholar allows.

(12) Letters of evaluation for computational scientists

Letters of reference used for hiring, tenure, and promotion purposes are typically first read by departmental committees that have expertise in the core areas represented by the department. On the other hand, computational scientists and in particular researchers working on scientific software – more or less by definition – are typically interdisciplinary, and the achievements that are assessed in such letters are consequently often meaningless or at least hard to evaluate for disciplinary committees. Both committees and letter writers need to be aware of this. Providing best practice examples, and training both writers and recipients of letters may be necessary to level the playing field for computational scientists.