Adoption

Main Themes/Concepts:

We began with an informal “round the table” discussion of our individual backgrounds and interests. Along the way, several short related conversations and topics for further discussion were noted. After following up on a few of these and a quick review of what we had covered so far (at approximately the 30 minute mark), we continued the discussion via a few follow-up questions to each other and how the responses to these related to our earlier notes. We concluded by reviewing all of the notes and identifying a few themes and concepts that we feel are key to understanding and addressing the adoption of software engineering practices in CSE projects.

1. Back to Basics
2. Overhead to switch (mental and resources): Personal, organizational, community
3. OSS: Adoption challenges apply to success of scientific "ecosystems"
4. Reproducibility - as a driver for adoption
5. Self-image and what does it mean to be a "programmer"?
    a. Being a programmer means being "responsible for the code"
    b. Adding another label and another set of responsibilities could imply diminish the importance of the primary label: "scientist"

All Notes:

1. Asking “What are the pain points?” is important -- Do they (scientists/CSE developers) actually know?
2. What are the drivers of adoption?
    a. Incentives are key. Even in OSS development where reputation is a key incentive, reputation in CSE domains comes from the science and publication, not the software.
    b. Software is used to expand research -- this results in different goals and sources of difficulty.
    c. There is a large scope and diversity of scientific software. Understanding the drivers and basic dimensions helps.
    d. Scientists don’t know where the research/software is going. (“But I won’t need it again”...)
3. From the CSE background. Given a specific problem to solve
    a. Generally a “just do it and keep trying” approach: If successful, it grows and becomes a “beast”. Then make it nice and make documentation. It would be nice if it could “be nice” in the first place, but this often isn’t possible.
    b. Looking for something where “10%-20%” more effort has significant impact. Maybe not to result in “perfection”, but better.
    c. Don’t always need cutting edge SE research, a “back to basics” approach is often very beneficial
        i. Source control
        ii. software-carpentry.org
        iii. Heroux’s “barely sufficient practices”
4. Overhead
    a. With respect to the 10-20% mentioned above, who pays vs. who benefits are often different people. Funding models are not designed for lasting software (although the 2010 NSG strategic plan and some journal reproducibility requirements are starting to influence this.
    b. Reproducibility is about more than just the underlying data and the software source code. Meta-data about system configuration, compiler, etc. is often required. (e.g. including a link to an Amazon server image)
    c. software-carpentry.org is finding that practices taught aren’t neccasrily “sticking” long term. Scientists learn about what is possible and want to do it, but the overhead to switch is too high - both personally and at the institute, community, and domain levels.
5. Where do faults come from vs. how to we find them and how expertise impacts each?
    a. This related to Jette’s work on expertise and faults in CSE software. The distinction here was in the source of faults vs. detection of them in the first place.

6. What does it mean to be a programmer?
7. Relationships with open source software (OSS)
    a. Jim H.: There's a desire for "ecosystems" of scientific software, but it is challenging because of difference in languages, formats, goals, etc.
    b. Serban/Jim: Discussion of the use of OSS in commercial CSE environments and the hesitation/inability to contribute back. Jim referenced the concept of good v. bad OSS citizens.
    c. What do scientific software developers learn by using OSS/third party libraries?
        i. Jim: There is a "normalization" effect in OSS communities in general where some SE practices spread.
        ii. Scientists may pick up the use of source control, commenting, or other practices via the use of other people's source code/projects.

Group Members:
- Serban Georgescu
- Tom Becker
- Vitor Carvacho Neves
- Leonardo Murta
- Jette Henderson
- Cait Pickens
- Jim Herbsleb
- Erika Mesh