2015-02-19 NSF PI Meeting Notes

People: Merce Crosas (scribe), Matthew Turk (scribe), Arthur Wetzel, Shava Smallen, Jeffrey Potoff, John Clyne (moderator), Eric Van Wyk, William Stein, Charles Torre, David Tarboton, Jan Verschelde, John McGregor, Hartmut Kaiser, Bruno Turcksin, Paul Navratil, Philip Wilsey, Anthony Aufdenkampe

Building communities breakout

Issues/Barriers

How do you engage a community to answer questions (and make other contributions), not just ask them?

There are different levels and/or types of engagement:

- users that may advocate (either actively or simply by recommending software to colleagues)
- users that actively help other users via forums, email lists, etc.
- users that test new releases
- users that contribute code, fix bugs

Recognizing contribution is important. tenure track faculty are, for example, reluctant to contribute to the code of others because it doesn't help them publish papers. Need to motivate people to contribute by finding ways to benefit them

The are a lot of technologies to facilitate engagement (e.g. googleplus, github, different mailing lists, stack overflow, stack exchange, researchgate) The choices can be overwhelming and it is difficult to decide which to use.

There are cultural barriers to communication (e.g. old school email vs new social networking)

Quality of contributions: how do you ensure that code contributions, for example, are up to snuff?

Choice of implementation language may help/hurt from contribution standpoint: Lots of scientists know python, few know C++,

Solutions

recognition: Having a story every week or every month about folks who have contributed is an incentive

Defining a roadmap is a good way to engage a community.

Elements of successful community building from http://yt-project.org/:

- A team with enthusiasm, not mean
- Transparency in your project; show how to contribute, show roadmap
- Make contributions, or be part of the community, fun and exciting
- Identify what your community is
- Prestige for the contributors

Lessons from http://www.sagemath.org/, which has over 500 user-developers:

- Use public code reviews for best quality control
- Put rigorous constraints in place for contributors (e.g. documentation standards, requirements for unit tests)

Recommended resources:

- Producing OSS: How to Run a Successful Free Software Project -- by Karl Fogel -- http://producingoss.com/
- The Art of Community by Jono Bacon -- http://www.artofcommunityonline.org/
- "The Success of Open Source" and various articles elsewhere - http://www.hup.harvard.edu/catalog.php?isbn=9780674018587
- WSSSPE -- http://wssspe.researchcomputing.org.uk/workshop-accepted/ and http://openresearchsoftware.metajnl.com/collections/special/working-towards-sustainablesoftware-for-science
- Software Sustainability Institute (see Community Building heading on each)
    - Guides: http://www.software.ac.uk/resources/guides-everything
    - Top tips: http://www.software.ac.uk/resources/top-tips
    - See following articles for example of how we fostered a new community around scientific software development professionals ("Research Software Engineers"):
        - Original concept arises in a workshop similar to SI2 workshop (see Session 4): http://software.ac.uk/cw12/cw12-five-important-things
        - Fostering debate in community: http://www.software.ac.uk/blog/2012-11-09-craftsperson-and-scholar
        - Pump-priming to get a sense of size of community: http://www.software.ac.uk/workshop-research-software-engineers
        - Handing over the reins: http://www.software.ac.uk/blog/2014-09-30-benign-dictatorship-democratic-association-rse-agm
        - The RSE community: http://www.rse.ac.uk/
    - Also: Richard Millington's work on online communities: http://www.feverbee.com/ (and book Buzzing Communities)

Actions

Learn lessons from existing successful relevant projects: e.g yt-project.org, http://www.sagemath.org/,

- What worked and what didnt

Publish best practices