Software Engineering Challenges and Best Practices for Multi-Institutional Scientific Software Development
(the slides are available under “Presentation Materials” in the above URL)
Date: August 4, 2021
Presented by: Keith Beattie (Lawrence Berkeley National Laboratory)

Q. At the start of projects, some non-profit research institutes and universities hire contractors to architect separate modules for asynchronous development. For organizations that lack large project expertise, how common are non-traditional contractors (i.e. not postdoc, staff scientist, PI) participating in initial setup of multi-institutional scientific software projects? Or maybe there are volunteer scientific working groups that review software architecture and governance proposals and help with mentoring?

A. This is very common. It becomes challenging to integrate all these pieces together. I have often been brought on at the beginning of the project to see how the different components are being developed and how they are being integrated together. That is a real challenge. When you bring in contractors and temporary postdocs there is a real risk that when they move on all of the institutional knowledge disappears with them, so make sure they’ve done the documentation or tests have been written.

Q. What are the challenges you faced regarding the storage of data when you were working on IceCube? How did you solve them? Why is writing data to tape an issue?

A. Not sure if I’m the best person to ask, because I worked on data acquisition. Asking about “challenges” is a very general question. Challenge was at the South Pole they would write data to tape. Also there was a satellite collection. Was a processing and filtering stage.

Q. What types of funding sources or funding institutions are favorable to renewing grants for large multi-institutional projects, especially for software maintenance and community support? Some institutions like the NSF seem resistant to renewals and seem to prefer finite project-driven proposals.

A. The ones that do will be the preferred ones :) At the end I was trying to get at the general scientific community has a problem taking scientific software development seriously enough. Need to identify pieces of software that should be maintained separately from scientific projects. Also as important is identifying software that “dies”. Funding organizations don’t look across multiple projects. This is starting to change. This is why I brought up the RSE organization. If we can identify there are people that can do this, then we can have software that persists across projects. But it’s a tough sell.

Q. How do you balance meetings ending on time with being willing to dive into the weeds?

A. It’s a tough call. I don’t usually let the meetings go over [time]. People usually have another Zoom call on the hour. Could tell them you would “schedule another meeting to go into those

details". Constant balance you have to do your best to maintain and hard to find a fully satisfying solution. Feeds into idea of scheduled releases and whatever you get done in that time slice is what you get done in that time slice. This bug fix or feature. Then do more later.

Q. Can you paint more of a picture of code reviews? It sounds like it's asynchronous and through issues tracker rather than video conferencing, etc.

A. Yes, usually back and forth and is asynchronous. That works well for some situations, but maybe other situations for having a shared conference. Zoom and screen sharing is powerful. Also paired programming though I haven't talked about it. There is also an old school formal structured review process, but that depends on the culture of your team. Have seen a PR closed and recreated as 2-3 other ones to make it more manageable.

Q. What format do you recommend for developer onboarding (e.g., tutorial, documentation, 1-1 sessions, etc.)? What about existing developers that are 15% time and may not remember all of the project processes (do we need refresher onboarding).

A. That's good. When we transitioned with IDAES it was transitioning intellectual property that motivated onboarding. Had them take a quiz and get a perfect 15/15 score. Even existing people on the project had to go through it. This may not be best - the person asking the question has a good point - "read these slides and when you pass this quiz I can add you into the team for elevated permissions in GitHub." Good for the team to have workshops or collaborative meetings for re-onboarding. The scheduled meeting itself is very educational for having discussions.

Q. (Slide 13) Do you have recommendations for security testing, or recommendations for certifications to build competence in security testing relevant to scientific software?

A. These are hard skills to build. Also very project dependent. Some automated tools that do static security analysis.

Q. At IDAES, did you analyze data generated from simulations? How did you store all of the needed data?

A. We do have a challenge with the data. It's often proprietary so it does have to be separated. Usually a specific run of the data. IDAES is a framework for writing custom models, so it's very instance specific how you would store data for a particular situation.

Q. How do you build culture in your project?

A. Slowly! It's a challenge. That's why having the meetings is good and trying to allow people to have communication and give training. Have been on projects where people were very much against the idea of even version control or having releases. You need to convince them that you are a nice guy that has good ideas. Try to be open and try to be inclusive. Listen to people. Take them seriously. It takes time! I don't know if there is any single answer to that other than taking your time, doing your best, listening to people, demonstrating that these tools can help.

Q. How should one who is at the smallest scale (individual) and domain expertise (i.e., not yet an RSE, some familiarity with tools) get started on a path toward larger efforts?

A. General career question. Reach out to people. Try to demonstrate that you have some abilities. Try to get yourself added onto larger projects. That depends on your institution and larger environment. Professional organizations are a good way to do this. Talk to your coworkers. Bring up your technical chops as best you can and get involved with projects.

Q. How do you achieve *deep work* on multiple scientific projects?

A. Haven't read this book, but skimming the abstract I can at least give two general answers. First, a lot of what a release manager does is socio-technical: talk to people and lead meetings, which have topics that, as I said in the talk, are mostly determined in real-time by the participants. So, the "distraction" here *is* the task, so to speak. For more focused technical work that could fall under the "deep work" category, the basic strategy is to create enough common technical ground across projects that slices of time can be combined to solve common problems. This doesn't always work, but does help a lot and I am still improving this process.

C. This SC21 workshop might be of interest to this audience: Research Software Engineers in HPC: Creating Community, Building Careers, Addressing Challenges.