Research Software Engineering Team Management and Scalability?

Test Subtitle

Hartwig Anzt[1,2], Terry Cojean[1], Fritz Göbel[1], Thomas Grützmacher[1], Pratik Nayak[1], Tobias Ribizel[1], and Yu-Hsiang Tsai[1]

[1]Karlsruhe Institute of Technology
[2]Innovative Computing Laboratory, University of Tennessee, Knoxville

Abstract

This is a temporary title, please change it.

As computer science research teams grow, gain success, and see their software adopted, a significant amount of Research Software Engineering (RSE) is required to maintain and expand the research software. In a way, a research team becomes similar to a small scale company with multiple people dealing with multiple roles including governance, financing, human resources as well as research and engineering roles such as code review, software quality assessment, … At the same time, research teams often only have an informal organization and lack proper management skill, both personal but more importantly collective. We believe that this usual scientific organization level is not well adapted to such mix of RSE and science and this can lead to scalability issues when trying to expand the team, either in terms of software or personel. Some solutions may come from organization strategies in companies or startups, while acknowledging that research requires more freedom to be effective.

Organization in a Scientific Team

A scientific research team which has a reasonable amount of success looks a lot like a company. There are three to four main types of activities: research, administrative tasks, software engineering tasks and depending on the case education or training, usually to find and form the next generation of scientists/engineers.

Some tasks are very costly and critical yet generally done by the PI. For example in terms of administration, you need to do advertisement (websites, some conference talks…), raise money (proposals), manage increasingly complex budgeting and hiring processes (with administrative barriers as well), etc.

Software engineering tasks are even more costly in terms of resources and feature a great diversity which is a challenge for the team to process all of them efficiently. Some examples of these tasks are technical administration (CI stuff), maintainer work (releases etc.), QA (tests, static analysis, reviews, …), support (issues, feature requests, …), architecture (library design), documentation (technical writing, examples), implementation (add new features in particular coming from research ideas), and management (who works on what).

The Scalibility issues

There are maybe two central ideas: - RSE team scalability issues in terms of engineering tasks and role attribution and sharing per person. Also, issues with doing all tasks: currently, QA in RSE is gaining traction but many important roles, in particular the management is left aside. Some tasks like QA, technical administration or support can be big time sinks which mean either some scientists will transition full time on those tasks or the whole team will try to deal with them in an unorganized fashion. In both cases, this can lead to starvation issues (threading term) with other tasks like science gradually taking a back-seat. - RSE team scalability issues in terms of governance, financing, PR and organization (single PI model doing most PR/financial aspects etc maybe doesn't scale above a certain threshold). For now, the PI usually tries to do (most) of these, which works up to small enterprise level, like <10 people and more brings big scalability issues. Making people who supervise subprojects deal with all of this can also not scale as it makes more people (collectively) lose their time, so there would be a point where it doesn't payoff anymore.

Solutions

In terms of better organization/management and scalability improvements, either or both: - Creating a position similar like a scientific manager person, able to manage role attribution of people and help with proposals, etc. Issues: should be **very** flexible and keep research freedom nonetheless (it's not 100% a company). - Flexibility in team organization (multiple persons are both proposal pushers, code reviewers, developers, …), but requires strict and coherent personnel scheduling.

Another aspect in terms of governance when trying to grow past the point where the PI is not able to cope with all the tasks: - Another idea: you could even do equivalents of some company direction roles, like at some point a research team if it keeps growing could eventually become more like a complete lab, in which case it might makes sense to split the main leading position into equivalents of CFO, CTO and CEO for example, if the PI cannot cope with all of this alone. - Down the line, is a hybrid model where a research team is factually associated to a company tasked with engineering on the software as well as advertising it? Probably not for all cases.

Conclusion

References