SAND2021-14850C

Can Scientific Software Development Use the Outsourcing Model Successfully?

Sivasankaran Rajamanickam[1]

[1] *Center for Computing Research, Sandia National Laboratories*[a], *Albuquerque, NM 87123, USA*

I. CHALLENGE

The primary challenge that we focus on is to arrive at a scalable model for scientific software development **and** maintenance within the Department of Energy (DOE). Scientific software development has stayed within the traditional model of in-house development within the DOE with a few notable exceptions of using libraries developed by the industry. While this model has several advantages, this position paper explores if it is possible to use an alternate approach of outsourcing certain aspects of software development for better efficiencies while also meeting the stringent requirements placed on the Department of Energy software. We use the word outsourcing in the broadest sense possible, essentially to partner with software-centric businesses for software development/maintenance as opposed to offshoring which may not be desirable for the DOE.

Based on past experience, it is not a stretch to say almost every line of code we commit today to a DOE software product requires maintaining for a decade or even two. However, the budgets for scientific software is part of new research which leads to new software development. Software maintenance has not been high priority for funding within the Department of Energy. Even with new funding, building a software focused organization that can maintain millions of lines of code will require a monumental effort. It will require competition with industry for software development engineers, a model that prioritizes both software development and software maintenance, and a reward structure that prioritizes both aspects. While it is possible to build this software-centric organization, we owe it to ourselves to explore alternate models as well. One such approach could be to outsource some aspects of software maintenance and/or development while allowing DOE scientists to focus on the science, algorithm design and software design and core development aspects.

II. OPPORTUNITY

Scientific software development and maintenance are both critical for the DOE mission in science and engineering. Development of new scientific software requires new algorithms, new interfaces, and design of new software components. Maintenance, on the other hand, is more modularized, based on well-defined, stable requirements, built on top of clean existing interfaces, and easily verified. Both efforts are key to the long term success of the DOE software ecosystem. The former requires co-design with hardware vendors, deep knowledge of domain use cases, back and forth with domain scientists, and inter-disciplinary work. The later requires attention to both past, current and upcoming hardware, deep knowledge of software testing and maintenance, experience in software engineering and a spirit to dive into a decade old code that was written for a hardware two generations earlier with little to no documentation. Both require unique and somewhat mutually exclusive skill set.

The challenging question to be addressed is whether DOE laboratories can use the outsourcing model for such software development/maintenance? If outsourcing is possible then what are the key first steps to try a model like that. In a different context, it has been argued that *process standards* are one of the key factors to decide what functions can be effectively outsourced [1]. The closest process standard to scientific software development is the process management standard developed by the Carnegie Mellon's Software Engineering Institute (SEI) called Capability Maturity Model (CMM). As a *process management standard* CMM only addresses that processes are in place for software quality as opposed to requiring a performance metric or unifying software processes for all organizations. Such a process management standard has allowed businesses trust external organizations with their non-core functionality and successfully outsource several software components that the US economy depends on (e.g. financial transactions). However, such a standard has been considered too onerous for the purposes of science software within the DOE. As someone who started his career as a software engineer in a CMM level-5 organization this author sympathises with this view.

While a process standard used by the software services industry is onerous for scientific software components, there is a new process that has emerged within the past few years. This process has come out of the Exascale Computing Program (ECP) where all software in the projects (within software technology, co-design, and application areas) follow a similar process management standard. As this is developed by the ECP, all the labs that are part of the ECP

[a] Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

use this process standard for their ECP projects. The ECP process allows clear steps for requirements gathering, approval by a management chain within the program, results gathering and reporting, review of the results, change management etc. This process has also been accepted by the science community within DOE and several large DOE software use it as part of their ECP efforts. This process is not perfect and could be improved further. This process is also not a standard currently. We argue that standardizing this process from ECP, with community input to make it better, will allow developing a process management standard based on a widely adopted process within DOE.

Formalizing a process standard for DOE scientific software could allow for an external organization to develop software practices that adhere to the standard. This could lead to confidence in the processes of the external organization and a path to verification and acceptance of their product and thereby a successful model for outsourcing.

Such a standard would also allow the DOE laboratories to assess what are their core capabilities that have to developed in-house and what are the support capabilities that can be developed in partnership with external organizations while meeting DOE needs. For example, one can ask the question to see if it will be possible to outsource activities such as maintenance of legacy code, or testing and release of software especially considering use cases from industry and other partners into account. Both these activities can be well defined using a process management standard, hence could be well supported by an external organization. This model already works at a very small scale with few companies without a defined standard for delivery. There are some Small Business Innovation Research work that are current examples for this model as well but without a defined standard.

The process standards could also allow more than one organization to develop the abilities to serve DOE needs while meeting the standards, leading to a healthy environment where DOE could choose from several organizations to partner with based on the value they provide. This approach should not be considered in any way as an alternate approach to having a software engineering team in house. Having a healthy software engineering team will still be required for a successful software delivery as needed by DOE applications. This model assumes that the in house software engineering team will both focus on the core development and will be the primary interface for external organizations that help with software maintenance or other capabilities.

There are several open questions that need to be addressed from security issues in outsourcing maintenance, training of the external organization, and funding models for such partnerships. However, if successful, this model could lead to a scalable approach for software maintenance while reducing that burden of DOE scientists and research engineers.

III. TIMELINESS

It is important to address the issue of software maintenance now as several DOE teams have invested considerable amount of time in rewriting their software to new GPU architectures. With the exascale computers coming online in the next couple of years, we expect several small changes to improve performance on additional architectures. Some of our codes still maintain different code paths for different hardware and they have to be tested and maintained on all new architectures. New efforts, such as the effort use mixed-precision or multiple precision within our software stack, lead to a combinatorial explosion of data types that need to be tested and maintained. The variety of architectures and different precision available combined lead to a large set of combinations that need to be tested and maintained that is typically beyond the capabilities of small in house software engineering teams. In addition, several open challenges in software require a rethink of current designs. For example, developing new software for emerging data flow hardware and finding the best path for integrating traditional software libraries with machine learning frameworks for scientific machine learning use cases require new software designs. We believe the in house software engineering teams will be overwhelmed adopting and maintaining software to all these changes and serve the new needs that rise due to science use cases and hardware changes. Partnerships with external organizations has the potential reduce this burden while still successfully meet the needs of science and engineering applications.

IV. REFERENCES

[1] Thomas H Davenport. The coming commoditization of processes. *Harvard business review*, 83(6):100–108, 2005.