S4PST: Sustainability for Programming Systems and Tools

https://ornl.github.io/events/s4pst2023/

May Workshop Report[*]

Organizers: Hartwig Anzt, University of Tennessee
Pedro Valero-Lara (co-PI), Oak Ridge National Laboratory
William F. Godoy (co-PI), Oak Ridge National Laboratory
Keita Teranishi (PI), Oak Ridge National Laboratory

Attendees: Sunita Chandrasekaran, University of Delaware
Damian Rouson, Lawrence Berkeley National Laboratory
Jeff Larkin, NVIDIA
Graham Lopez, NVIDIA
Joel Denny, Oak Ridge National Laboratory
Suzanne Parete-Koon, Oak Ridge National Laboratory
Patrick Diehl, Louisiana State University
Hartmut Kaiser, Louisiana State University
Valentin Churavy, Massachusetts Institute of Technology
Julian Samaroo, Massachusetts Institute of Technology
Johannes Blaschke, Lawrence Berkeley National Laboratory
Damien Lebrun-Grandie, Oak Ridge National Laboratory
Michel Schanen, Argonne National Laboratory
Francesco Rizzi, NexGen Analytics
Philippe P. Pébaÿ, NexGen Analytics
Christian Trott, Sandia National Laboratory
Swaroop Pophale, Oak Ridge National Laboratory
Terry R. Jones, Oak Ridge National Laboratory
Rafael Ferreira da Silva, Oak Ridge National Laboratory

August 24, 2023

[*]This manuscript has been authored by UT-Battelle, LLC under Contract No. DEAC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan)

1 Introduction

The US Department of Energy (DOE) Exascale Computing Project (ECP) [32, 21] has fostered and strengthened the use of modern software engineering practices for developing applications and libraries [12, 23], and this effort has resulted in the coordinated and interoperable E4S[1] and xSDK[2] ecosystems. Although this approach is cost-effective, it relies on robust programming systems and tools (PST) as the underlying foundation for our HPC software. At present, our primary PST stack consists of traditional high-performance computing (HPC) languages, namely Fortran, C, C++, and the popular Python language for data analysis and AI workflows [22]. These languages support various programming frameworks and run-time abstractions that enable parallelism and concurrency across multiple node architectures [17, 34, 35, 39] and thousands of nodes [14, 16, 24] through a variety of interconnect systems. However, to accommodate users' diverse needs, certain aspects of the HPC ecosystem are delegated to vendor-specific or third-party implementations that extend beyond a particular scientific domain. This broader scope results in a multitude of specifications and variations, which leads to a complex orchestration of *many-ecosystems*. Unfortunately, this complexity in the ecosystem imposes additional overhead costs on consumers during the latter stages of the development cycle [38].

In addition to the software ecosystem challenge, the upcoming conclusion of the ECP by December 2023 has raised significant concerns within the HPC programming systems community, from both the economic and social perspectives. The ECP has implemented a management structure for software development and funding decisions across all ECP participants by following a conventional hierarchical and centralized approach. However, this structure has prompted certain considerations within the community, particularly in anticipation of the Software Sustainability initiative by the DOE's Advanced Scientific Computing Research Program (ASCR). For the success of this new initiative, it is of utmost importance to secure consistent funding and foster close engagement with researchers and core developers of existing programming-system products. This collaboration is vital to maintaining the critical capabilities of the current software during the transition phase while proactively adapting to future technology and workforce trends. The community recognizes the significance of adapting to emerging trends and is aware of the inherent fragility of the HPC software ecosystem, particularly in relation to programming systems that cater to all users. The ability to adapt and evolve is essential to staying relevant and effectively addressing these technical, economic, and social challenges.

The S4PST team, which represents one of the six ASCR Software Sustainability seedling projects, is dedicated to tackling these challenges through community-based approaches that go beyond the scope of the DOE. This involves collaboration between national laboratories with academia, non-DOE institutions, hardware and system vendors, and international partners. By fostering these partnerships, we aim to create a robust and sustainable HPC software ecosystem that can effectively meet the needs of the community. This new community effort, driven by the eight DOE labs, will take on the responsibility of guiding funding decisions for programming-systems development and maintenance with transparency and consistency across all decisions. Additionally, the team will offer common technical services to the programming systems community, irrespective of their funding situations, and facilitate community-wide incubation to proactively nurture the software ecosystem. By actively engaging with stakeholders and employing a collaborative approach, we can collectively shape the future of programming systems and ensure a robust and thriving HPC software landscape.

On May 11–12, 2023, the S4PST team conducted its inaugural kick-off workshop at the Innovative Computing Laboratory (ICL) in the University of Tennessee, Knoxville, hosted by Hartwig Anzt. The workshop encompassed various sessions dedicated to presentations and discussions, with the aim of comprehending the team members' perspectives on the vision of software sustainability. Additionally, the workshop aimed to identify the technical, economic, and social requirements for sustaining the programming-systems community in the field of HPC.

[1]https://e4s-project.github.io
[2]https://xsdk.info

This report provides a summary of the S4PST effort by highlighting five major thrust areas discussed during the workshop: (i) community, (ii) technical support, (iii) training and diversity, (iv) verification, validation and correctness, and (v) emerging technologies. It also encompasses an overview of the presentations and discussions held throughout the event, our views and potential synergies with other seedling efforts, along with the outcomes and key takeaways from our initial discussions.

2 S4PST Thrusts Areas (Goals)

To make our S4PST vision possible, we propose five efforts to enhance the robustness of programming systems and their ecosystems, thereby ensuring their long-term sustainability. Some of these ideas may require several years before they fully benefit users and developers, regardless of their representation in the broader HPC community. Most of our discussion here is not overly technical, but these five efforts will serve as guiding principles for decision-making regarding the design, development, and promotion of high-quality programming systems.

2.1 Sustainable Community

To create a sustainable community for PST, we have identified several key efforts. The existing initiatives in the ECP, E4S and xSDK [3, 11], have made significant progress in establishing community policies and standards that enhance the software quality of large-scale production software suites. For example, xSDK began with six scientific library products in 2016. Owing to their persistent community engagement and implementation of the policies to encourage better scientific software engineering practice and interoperability, the xSDK 0.8.0 version today includes a total of 29 library packages. Notably, most of these packages do not receive direct funding from the xSDK project, thereby indicating how the community itself can improve the software quality and collaboration.

2.2 Community-Wide Technical Support

These efforts accomplish the goals for a sustained PST ecosystem: (1) identifying, maintaining, and supporting current and future critical capabilities of the PST ecosystem; (2) tracking implementation, standard specification, and latest capabilities; (3) establishing vendor and open-source points of contact; (4) facilitating dependency tracking and deployment via package managers (e.g., Spack, Fortran/Julia Package Manager); and (5) tracking the interoperability of individual languages, programming systems, and tools. This information is often scattered without any coordination across different supercomputing facilities or small groups of experts. Having organized archives such as tables describing language feature requirements for certain programming systems (e.g., C++17, specific compiler versions) and interoperability for major libraries (e.g., MPI) facilitates quick referencing.

2.3 Training and Diversity

This thrust includes an aggressive community engagement and training effort that specifically targets traditionally underrepresented minorities to diversify the pipeline of expert personnel and to ensure that technical and social debts are minimized for future generations. This will be accomplished while sustaining and expanding the existing connections built between DOE and the broader scientific communities. Partnering with universities, we can push for training programs and internships and work with faculty members to include PST expertise in their curriculum.

Our goal is to define a training program that focuses on teaching modern programming methods. This program will expose students and application developers to techniques that make code safer (e.g., smart pointer vs. raw pointer), cross-language and cross-platform interoperability, and—more importantly—principals and best practices for software testing and packaging. These ideas have been implemented in the IDEAS-ECP project [4], and we will further explore opportunities in non-HPC venues, as seen in the new scientific computing track of the CppCon 2022 Conference [2] and other programming systems conferences such as RustCon [8] and JuliaCon [5].

2.4 Verification, Validation, and Correctness

The main goal of this effort is to create a strategy to reduce vendor dependence and identify gaps (e.g., bugs) in vendor and S4PST stacks. Software defects in widely used vendor software components (e.g., compilers and run-times) can have significant impacts on productivity. We will also continue building on our collaborations with vendors through early proactive verification and validation engagements that are independent but complementary to vendor efforts.

We will propose the application of proxy apps [33] to identify important workloads and a hierarchical approach to perform different levels of validation: functionality, accuracy, correctness, and scalability. The goal is to ensure that the DOE's interests for HPC programming systems and tools are represented in open-source and vendor-independent verification and validation test suites. Such suites will capture important use cases from DOE HPC applications and, for those use cases, check that programming systems and tools conform to relevant programming model standards, support interoperability across multiple programming systems and runtime systems, and support portability across HPC's increasingly heterogeneous hardware ecosystem.

The results from these suites will be useful to HPC users, programming systems and tools vendors, hardware vendors, and DOE program managers as they evaluate the suitability of programming systems and tools for DOE's needs. Although some relevant suites already exist [29, 30], we will propose efforts to identify gaps, create new suites, and extend existing suites as needed to ensure that the DOE's evolving requirements continue to be represented.

We also plan to engage with vendors in sustaining widely used correctness tools (e.g., memory checkers, data race checkers), debuggers, and the adoption of interfaces to facilitate debugging parallel code on emerging platforms (e.g., OMPD in OpenMP, DWARF extensions, and Python debugging).

2.5 Emerging Technologies

One of the major S4PST objectives is the definition of mechanisms to guarantee that DOE's priorities are part of the PST ecosystem. This will help develop a more robust, functional, and sustainable PST ecosystem. Important DOE Research Priorities [12, 23, 41], such as performance portability, extreme heterogeneity, and automatic verification, among many others, will help build a valuable long-term PST ecosystem that will exceed the current sustainability capacity and contribute significantly to key scientific milestones and transformational discoveries.

Although DOE and ASCR maintain a very important software stack, it is part of ASCR's identity and our mission to continue to propose new ideas and technologies that can be integrated into DOE's portfolio. The value and return on investment for software sustainability on emerging technologies, such as modern LLVM-based high-productivity languages and ecosystems (e.g. Julia, Rust, Python/Numba) and modern build systems (e.g. Meson), must be understood in the DOE context in conjunction with their success in the broader field of computing. This understanding will consider the lessons learned previous PST-related efforts, such as the Defense Advanced Research Projects Agency (DARPA) High Productivity Computing Systems (HPCS) program [20]. This is crucial in the convergence of AI + HPC because AI is driven by different community and business needs and does not necessarily focus on the scalability and science mission of DOE's HPC efforts.

3 S4PST Community

Institution Name	Point of Contact
—Universities—	
Louisiana State University	Hartmut Kaiser and Patrick Diehl
Massachusetts Institute of Technology	Alan Edelman and Valentin Churavy
University of Delaware	Sunita Chandrasekaran
University of Tennessee	Hartwig Anzt
University of Oregon	Sameer Shende
Carnegie Mellon University	Franz Franchetti and Tze-Meng Low
The Ohio State University	Dhabaleswar K. Panda and Hari Subramoni
—Vendors—	
AMD	Jakub Kurzak
HPE	Barbara Chapmann
Intel	Sarah Knupper
NVIDIA	Jeff Larkin
—DOE Laboratories—	
Argonne National Laboratory	Michel Schanen
Brookhaven National Laboratory	Sunita Chandrasekaran
Oak Ridge National Laboratory	Keita Teranishi, William F. Godoy, Pedro Valero-Lara
Lawrence Berkeley National Laboratory	Damian Rouson
Lawrence Livermore National Laboratory	Johannes Doerfert and Ignacio Laguna
Los Alamos National Laboratory	Patrick McCormick
Pacific Northwest National Laboratory	Roberto Gioiosa
Sandia National Laboratories	Christian Trott and Siva Rajamanickam
—non-DOE Laboratories—	
NASA	Piyush Mehrotra
—non-US Universities and Laboratories—	
BSC (Spain)	Antonio J. Peña
ETH Zürich (Switzerland)	Juan Gómez Luna
Inria (France)	Mathieu Faverge and Olivier Aumage
Riken CCS (Japan)	Miwako Tsuji

4 S4PST Workshop

4.1 Agenda

	—May 11, 2023—	
	S4PST Intro	
10:00–10:15 AM	S4PST	Keita Teranishi
10:15–10:30 AM	Lessons from ECP	William F. Godoy
10:30–10:45 AM	S4PST Portfolio	Pedro Valero-Lara
	Thrusts Talks	
10:45–11:00 AM	S4PST Survey	Johannes Blaschke
11:00–11:15 AM	Sustainable Community	Keita Teranishi
11:15–11:30 AM	Community-Wide Technical Support	Keita Teranishi
11:30–11:45 AM	Training and Diversity	Damian Rouson
11:45–Noon	Verification & Validation	Sunita Chandrasekaran
12:00–12:15 PM	Emerging Technologies	Michel Schanen
	Lunch Break	
1:15–2:00 PM	**Breakout sessions (5 thrusts)**	
	Break	
	Talks	
2:15–2:45 PM	Fortran, UPC++, ...	Damian Rouson
2:45–3:15 PM	C++/Kokkos	Christian Trott
3:15–3:45 PM	HPC Workforce	Suzanne Parete-Koon
	Wrap-Up	
	—May 12, 2023—	
10:00–10:15 AM	**Opening Remarks**	Keita, William, Pedro
	Talks	
10:15–10:45 AM	Sustainable Software for Numerics Research on DOE Systems	Michel Schanen
10:45–11:15 AM	NVIDIA's programming language strategy	Jeff Larkin
11:15–11:40 AM	Julia	Valentin Churavy
11:40–Noon	SWAS	Rafael Ferreira da Silva
12:00–12:20 PM	STEP	Terry R. Jones
	Lunch Break	
1:20–3:20 PM	**Breakout Sessions (5 thrusts)**	
3:20–3:40 PM	**S4PST Work Plan**	Keita, William, Pedro
3:40–4:00 PM	**Final Remarks**	Keita

4.2 Talks

S4PST (Keita Teranishi): This presentation introduced the S4PST seedling project by outlining its initial vision. The presentation also provided an overview of the workshop and described the activities expected throughout the seedling projects. The speaker reemphasized the primary objective of the workshop: create an open forum for opinions and views on the sustainability of programming systems.

Lessons from ECP (William F. Godoy): This talk provided some perspectives of how ECP [31] was a complete game changer as one of the largest modernization projects undertaken to enable exascale computing in the United States. The speaker summarized a collection of experiences from contributing to the ADIOS2 [27] and PROTEAS-TUNE [28] software projects and pragmatic software engineering efforts in the QMCPACK application [26]. ECP enabled a large collaboration model in which collaborations among projects were largely encouraged. Any effort toward sustainability should preserve this legacy of collaboration and avoid uncoordinated and siloed efforts. Overall, this is a reflection of how ECP software enables science by providing very specialized equipment with the goal to strengthen scientific discovery and the community [21]. Care must be taken in trade-offs of quality and cost, and our goals might not necessarily align with more commercial or commodity software.

S4PST Portfolio (Pedro Valero-Lara): This presentation provided a comprehensive list of current programming systems and tools within the P4SPT ecosystem. Notably, the inclusion of these programming systems and tools in the list does not imply prioritization or essentiality for sustainability in the post-ECP era. Rather, the purpose of including a diverse range of systems and tools is to emphasize the array of options available for HPC.

The talk also highlighted the important role of the S4PST community in providing portfolio recommendations. S4PST envisions representatives from the S4PST center playing a crucial role in transparently and consistently evaluating the essential capabilities required for the development and deployment of sustainable applications. This evaluation process goes beyond the scope of the DOE and includes engagement with other stakeholders. These recommendations and actions may involve comprehensive evaluations, providing advice and suggestions, exploring opportunities to join the *emerging technology thrust*, and fostering collaborations with other initiatives and programs selected for sustainability or receiving funding.

—Programming Languages—
Fortran, C / C++, Julia, SyCL, Python
——Programming Models and Runtimes—
OpenMP, OpenACC, Kokkos, HPX, ParSEC, Caffeine, GASNet-EX UPC++, Coarray Fortran, Legion, IRIS, OpenMPI, MPICH, MVAPICH
—Compilers—
LLVM, Clang, Julia
—Math Libraries—
BLAS, LaPACK, SLATE, MAGMA, PLASMA, dPLASMA Ginkgo, FFTX, Enzyme, Kokkos Kernels, BLIS, libFLAME, MatRIS
—Packaging—
Spack, Fortran Package Manager, Julia
—Profilers/Performance Tools—
TAU, APEX, PAPI
—AI Platforms—
ChatGPT, Github-CoPilot, Llama-2
—Others—
Spiral

S4PST Survey (Johannes Blaschke): This talk introduced the preliminary version of the programming systems survey, which is aimed at DOE's broader HPC community. The draft includes over 40 questions regarding the practices and concerns of both users and developers. Throughout the discussion, many workshop participants proposed the idea of condensing the content to enable the majority of users to complete the survey without investing an excessive amount of time.

Sustainable Community (Keita Teranishi): This talk explored the concepts of community building, governance, and software quality assurance within the programming systems community. The community has experienced fragmentation, thereby necessitating a collaborative approach to accommodate the scientific computing needs of the ASCR missions. By focusing on the five key thrusts outlined in Section 2, this talk aimed to establish a new engagement with major programming systems communities. The goal is to serve a wide range of users while avoiding situations in which a single programming system dominates, thereby fostering a healthy research community.

Community-Wide Technical Support (Keita Teranishi): This talk provided an overview of the essential community services that S4PST should offer. Currently, information and documents related to programming systems are fragmented and lack coordination among developers and HPC facilities. Furthermore, gathering data on bugs, known issues, and interoperability with other programming systems, tools, and libraries requires significant effort from expert users. S4PST should perform a collective effort to coordinate the information gathering, continuous integration continuous deployment (CI/CD) services, and document archiving in collaboration with other seedling projects (e.g., PESO, COLABS, STEP, SWAS, and OSSF).

Training and Diversity (Damian Rouson): This talk started with a question posed in a recent *New York Times* column: "Is a gay Republican Latino more capable of conducting a physics experiment than a white progressive heterosexual woman?" The talk reframed the question as "*Might* a gay Republican Latino have different capabilities that *could* influence how they would conduct a physics experiment or which physics experiments they *might* choose to conduct as compared to a white progressive heterosexual woman?" wherein the italicized words clarify that the answer is probabilistic because the traits given are insufficient to determine the outcomes, and the underlined phrase allows for a much richer and more complex understanding of human capabilities than a single measure for which one can write mathematical inequalities. In this rephrased format, the talk suggested that the answer is a resounding "yes."

Science answers questions that scientists ask. Asking different questions might yield very different answers—even sometimes seemingly contradictory ones—especially in the social sciences. The talk demonstrated that research software engineering is not so far from the social sciences: we write programming languages to communicate with humans, whether other developers or our future selves. Were the only goal to instruct the computer, we could all write hexadecimal machine code. Moreover, many concepts that linguists apply to natural languages also apply to programming languages, including potential language biases.

The talk also suggested that research software engineering for science is also not so far from the humanities: the talk cites the book *The Jazz of Physics*, in which black Brown University cosmologist and jazz musician Stephon Alexander describes the ways in which his music, which has African-American roots, inspires his physics theories and how music has inspired great accomplishments in physics history, including Einstein's relativity. The talk cites a study that demonstrated a diversity-innovation paradox in which smaller minority groups innovate more but their work is taken up less often by others. The talk discussed ways in which a diverse workforce does work differently and does different work. The talk also advocated for supporting alternative parallel programming models to avoid a monoculture of MPI+x. The talk concluded with the need for on-the-job training in HPC programming languages that are unlikely to be taught in academia.

Verification & Validation (V&V) (Sunita Chandrasekaran): This talk focused on the ongoing efforts to build a V&V suite for the directive-based programming models (e.g., OpenMP and OpenACC) with an emphasis on exploring the gaps in these suites. The talk also described the need for a regular CI for regression testing and the need for integration of tests for existing harnesses. Designing and writing test cases that can check for correctness or specification compliance was also described. Finally a need for stress tests for the runtime, memory space analysis, and data transfer times, among other features, was described.

Emerging Technologies (Michel Schanen): The presenter discussed the issues of programming mathematical methods and algorithms for emerging heterogeneous hardware architectures driven by custom vendor APIs (e.g., for GPUs, wafer architectures, and quantum computing). The talk briefly presented the incurred HPC development cost and the LLVM-based programming language Julia as a possible high-level abstraction layer.

Sustainable Software for Numerics Research on DOE Systems (Michel Schanen): This talk focused on the development pipeline of novel numerical methods required to leverage hardware accelerators. Instead of incremental changes, fundamental method developments are needed for algorithms such as fundamental linear solver libraries and preconditioners. This research is often hindered by a siloed research process of numerical methods in Matlab and HPC implementations in C/C++. As a possible solution, high-level language support priority should be significantly increased on DOE systems.

Fortran and UPC++ (Damian Rouson): This talk presented some of the risks associated with hybrid MPI+x programming model. The risks include the establishment of a monoculture and the risks of costly code rewrites as MPI and x evolve. The talk also presented several alternative, unified parallel programming models and focused primarily on two: Fortran and UPC++ [14, 40]. The talk argued that sustaining alternatives reduces the aforementioned risks and improves programmability and performance. The talk highlighted communities organized around these alternatives – notably the PAW-ATM workshop series at the SC conference. The talk then described the support for parallel programming in Fortran and UPC++, including overviews of the feature set and successful applications of both programming models. The talk also covered related work at Berkeley Lab on the LLVM Flang compiler, include the Flang Testing Project [36], a Coarray Fortran Runtime API design document, and the Caffeine parallel runtime library [37, 19]. These programming models and others are supported by the GASNet-EX communication library [25, 18], also developed at Berkeley Lab.

Kokkos and C++ ecosystem (Christian Trott): This talk provided an introduction into the level of effort dedicated to the Kokkos EcoSystem and related ISO C++ standardization. Most activities fall into the sustainability efforts category, and even the research-oriented parts (e.g., RemoteSpaces) may meet the criteria for incubation efforts discussed in S4PST. The talk pointed out that due to the large user base of the Kokkos EcoSystem and the diverse set of platforms and tool chains that must be supported to meet user requirements, several FTEs are required to maintain the current level of usability and reliability of Kokkos across deployed HPC systems. Furthermore, the need for staffing consistency for the ISO C++ standardization efforts was pointed out. Proposals take 2–5 years to make it from conception to actual acceptance, and larger efforts can take longer. Thus, successfully standardizing HPC capabilities requires long-term engagement—especially in terms of building consensus in the committee, which requires maintaining interpersonal connections.

Workforce development (Suzanne Parete-Koon): The Exascale Computing Project's (ECP) Broadening Participation Initiative, led by researchers and engineers from the DOE national laboratories, aims to establish a diverse and inclusive workforce in the high-performance computing (HPC) community by creating a supportive culture within the computing sciences at DOE labs and severs as an example of how large multi-institutional pojects can foster workforce development. The initiative encompasses three key thrusts: establishing a Workforce Development and Retention Action Group which provides a community scaffolding to welcome and encourage DOE researchers to join the workforce development effort, producing accessible 'Intro to HPC' training materials, and launching the Sustainable Research Pathways for High-Performance Computing (SRP-HPC) program. SRP-HPC, which originated from Lawrence Berkeley National Laboratory and is scaled up across the ECP community, engages underrepresented students and faculty in collaborative projects across application development, software technologies, and advanced computing facilities. By leveraging ECP's multilab partnership, the initiative strives to foster sustainable collaboration, aiming to transform the culture and demographics of DOE computing sciences for a more inclusive and diverse future. Efforts like this will need programmatic support to be sustained beyond ECP.

NVIDIA's programming language strategy (Jeff Larkin): This talk described NVIDIA's programming-systems products for their GPU and upcoming CPU platforms and how their proposed strategy should apply to other systems. NVIDIA is offering programming systems to support users of different skill levels and needs, including low-level GPU programming languages, performance portable abstraction (OpenMP and OpenACC), ISO standard languages, and productivity-focused library products. Notably, NVIDIA is spearheading the support of the latest ISO C++ and Fortran standards for data parallelism and continues to engage with the standards bodies on future standards.

Julia (Valentin Churavy): This talk described the Julia programming language in the context of HPC. Julia leverages the LLVM compiler infrastructure to target a variety of CPU and GPU architectures. Julia provides a flexible foreign function interface (FFI) to interoperate with C, Fortran, Python, and R. A remaining challenge is direct interoperability with high-level C++ constructs. In the context of S4PST, we believe Julia is an ideal language to extend existing DOE mission codes with additional capabilities. One can use Julia for easy user-extensibility by leveraging its JIT compilation and using its rich automatic-differentiation ecosystem for adding uncertainty quantification and machine learning. The Julia community has invested much in V&V through continuous integration (CI) and continuous benchmarking of the base language and compiler. Furthermore, we perform ecosystem-wide verification of the Julia compiler before new releases. A recent survey of the Julia ecosystem showed that at least 95% of Julia packages had some form of CI. The ecosystem-wide verification comes from Julia having one consistent way of installing and running tests on a package, and also through the business (JuliaHub) and research (MIT) endeavors of community stakeholders.

Sustaining Workflows and Application Services (Rafael Ferreira da Silva): This presentation introduced the Sustaining Workflows and Application Services (SWAS) project [10]. SWAS is a collaborative initiative that brings together academia, national labs, and industry to establish a sustainable software ecosystem to support a wide range of workflows, including the software and services used in workflow orchestration. With a focus on growth, support, and long-term sustainability, SWAS aims to cover the entire spectrum of analysis, simulation, experiment, and machine learning workflows. The objective is to identify critical software components and develop a customized plan for their sustained development by accounting for the specific needs of this software ecosystem. By leveraging a community-endorsed sustainability model, SWAS will enhance and maintain workflows and application services, thereby incorporating robust validation and verification capabilities. This approach ensures that the provided systems and application services deliver the essential functionality and reliability required by DOE science users. SWAS actively encourages engagement from software teams and communities involved in workflows and application services. Notably, in this context, *workflows* encompass a variety of software and services necessary for configuring, orchestrating, and executing modern analysis, modeling, and simulation campaigns. Specifically, SWAS seeks collaboration with the following communities: workflow systems, data management frameworks, visualization frameworks, AI/ML tools used in modern workflows, and application services.

Sustainable Tools Ecosystem Project (Terry R. Jones): This talk presented the Sustainable Tools Ecosystem Project (STEP) [13]. STEP is a software sustainability activity that seeks to understand the capabilities and opportunities of the HPC tools community and find ways to sustain and improve the tools ecosystem going forward. Tools are closely bound to architectures and system software in ways that other types of software, such as libraries and scientific applications, are not. For example, a tool that tracks how an application uses computing resources must be able to measure low-level architectural events and metrics and relate them to program progress and source code. As a result, tools must closely track changes to applications, architectures, and their software stacks. To complicate matters, the need for tools is most acute for understanding code performance on systems that push the boundaries of technology and scale, but these systems' novelty makes them extremely difficult for tool developers to support when first deployed. The advent of exascale systems, increases in architectural diversity, and additional complexity driven by heterogeneity, all make providing effective tools for scientists and engineers essential to avoid impeding scientific discovery.

5 Potential S4PST Synergies with Other Seedlings Projects

During the workshop and subsequent meetings, we discussed the engagement with the other proposed software sustainability projects. We recognize that while S4PST covers a relevant aspect of software sustainability, its success requires exploring synergies and collaboration across the entire DOE ecosystem.

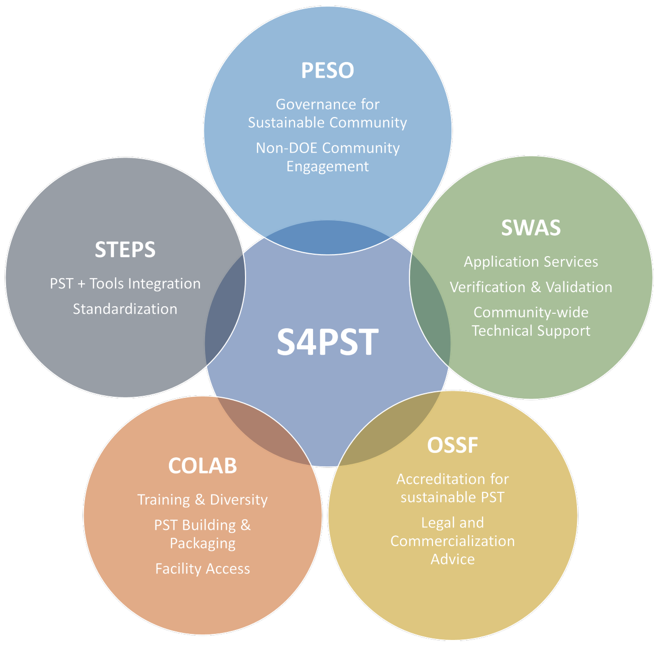
In this section, we introduce the other DOE ASCR seedlings projects and describe our vision for synergies and collaborations. Figure 5 summarizes some of the key synergies discussed during the last few months with the existing seedlings.

S4PST-PESO: The PESO (Toward a Post-ECP Software-Sustainability Organization) [7] seed initiative seeks to establish robust and sustainable scientific software capabilities aligned with the interests of DOE's ASCR. They propose partnerships with software teams and communities within DOE, other US agencies, commercial scientific software developers, and the broader open-source software community. The goal of PESO is to advance scientific pursuits by providing advanced scientific libraries and tools to the user community in ways that optimize the cost and benefit sharing of DOE's investments in scientific libraries and tools.

We envision two main areas of collaboration with PESO. (1) Governance for sustainable community: S4PST could be part of this effort and participate actively in these activities, thereby integrating the S4PST community and portfolio. (2) Non-DOE community engagement: the S4PST community and portfolio are beyond DOE laboratories and tools; our community involves universities, vendors, non-DOE laboratories, and non-US institutions. The non-DOE S4PST community provides critical capacities to DOE. This is an opportunity for DOE to have more influence on the PST community, thereby leading the sustainability efforts at the community-wide level even outside of DOE.

S4PST-SWAS: The Center for Sustaining Workflows and Application Services [10] targets the software and services used in workflows as well as the workflow orchestration software itself. This includes a full range of analysis, simulation, experiment, and machine learning workflows. SWAS can leverage and enhance two key initiatives of S4PST: community-wide technical support and V&V and correctness. By extending these services to the workflow community, SWAS can benefit from the expertise and resources already established by S4PST. Additionally, SWAS can incorporate specific workflow requirements into the existing efforts of S4PST, thereby further strengthening the support provided to the workflow community. Moreover, SWAS will play a crucial role in offering essential services

Figure 1: S4PST synergies with other seedlings.


PESO
Governance for Sustainable Community
Non-DOE Community Engagement
STEPS
PST + Tools Integration
Standardization
SWAS
Application Services
Verification & Validation
Community-wide Technical Support
S4PST
COLAB
Training & Diversity
PST Building & Packaging
Facility Access
OSSF
Accreditation for sustainable PST
Legal and Commercialization Advice

to various applications. This collaboration can prove highly advantageous for S4PST as well because it provides an opportunity for S4PST to engage with other application communities to expand its reach and impact. The mutual exchange of knowledge and collaboration between SWAS and S4PST will create a synergistic relationship that benefits both initiatives and facilitates the advancement of the broader scientific community.

S4PST-STEP: The Sustainable Tools Ecosystem Project (STEP) [9] targets the HPC Tools community to find ways to sustain and improve the tools ecosystem going forward as required due to the evolution in the instrumentation of hardware architectures, applications and libraries. The STEP concept advocates for an open and broad collaborative approach to identifying and implementing the best way to build a sustainable HPC tools ecosystem. We envision two important engagements between S4PST and STEP. (1) The integration of tools with programming systems: this will help our communities have a more sustainable and robust software ecosystem by providing much simpler-to-use and integrated software solutions. (2) Standardization of tools: although standardization is a common practice in programming systems and an important effort in S4PST, it is more difficult to find examples of that in the community represented by STEP. This is related to the previous engagement and could be very beneficial for software sustainability.

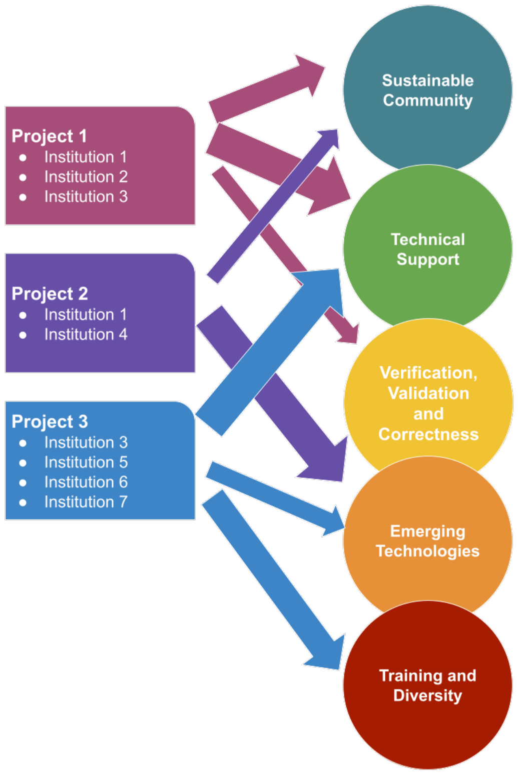
S4PST-COLABS: The Collaboration for Better Software (COLABS) [1] software sustainability organization (SSO) will provide a range of services to client software projects and the broader community in partnership with ASCR's user facilities: training and outreach, essential and advanced software engineering services, and a modest R&D effort. The R&D component will ensure continued enrichment of the SSO's offerings. The SSO will place a strong emphasis on workforce development and retention and will provide long-term stability, training, and support to enable and encourage research software engineers (RSEs) and other staff to build their careers and excel in this role. S4PST could collaborate with COLABS complementing training aspects in the software sustainability effort. S4PST and COLABS expertise is crucial to have a deep understanding of the HPC landscape and balance the application of software engineering for a greater return in the scientific mission [15, 26]. Also, aspects of the software life cycle, e.g. building, testing, continuous integration and deployment (CI/CD) and packaging PST solutions can be beneficial for both efforts. We also realize the importance of the RSE role, but not at the expense of highly-specialized core development teams in ASCR-funded software. Therefore, we encourage collaboration and inclusion of the spectrum of the skills and personal in the multidisciplinary endeavor that is sustaining software in a way that maximizes the benefit for DOE.

S4PST-OSSF: The primary goal of OSSF (Open Scientific Software Foundation) [6] is to create an organizational structure with strong stakeholder and community support along with a detailed funding strategy and implementation plan, so that it is possible to transition to an operational mode as soon as possible when the appropriate conditions are met. To achieve this, OSSF plans to undertake six core activities: (1) realize a foundation structure, (2) determine the sustainability objectives and create a strategy for meeting them, (3) engage with stakeholders, (4) engage with sponsors, (5) create a transition plan for key projects, and (6) establish a plan for standing up the organization. We understand that much of the ASCR-funded scientific software value is in the research produced, rather than for commercial purposes. Hence, we are still learning how the OSSF's proposed model will operate in the post-ECP landscape. As national laboratories and universities we are subject to our own rules and regulations for legal and commercialization aspects. In general, we are expecting to see how the concept proposed by OSSF evolves and how it fits in the existing landscape in which organizations such as Kitware Inc., the HDF5 Group, and NumFocus provide value to very specific scientific software products. Additionally, we would like to explore the synergies between such a foundation with more traditional non-profit organizations, including national laboratory sub-contractors and universities. We also realize the importance and historical role of national laboratories in scientific software and HPC as key government investments with existing mechanisms for external engagement (e.g. CRADA, SBIRs). Incorporating an external foundation would need to add value (rather than reduce it with additional overhead or top-heavy structures) to the mission of national laboratories and universities as unique platforms for science.

6 S4PST Organization Strategy

We envision a workforce that will be very different depending on the singular characteristics of each project. Not all projects have the same scope, size, and requirements for sustainability. Also, some projects may have other sources for funding, while others may not. Some of them may need more effort in some of the S4PST objectives than other projects or may not need to participate in some efforts at all, as illustrated in Figure 6. The projects can be composed of different institutions and involve DOE or non-DOE laboratories, universities, and/or vendors.

Figure 2: S4PST organization strategy.



7 Outcome and Future Directions

The initial S4PST workshop provided a unique opportunity to present a wide of views of the community invested in the variety of programming systems in the landscape of HPC. Some key takeaways can be summarized as follows:

- Identify workforce requirements for the software sustainability of critical components of different S4PST projects in the current landscape. The unprecedented investment made in the ECP software stack needs to be maintained and preserved for the upcoming exascale computing era and beyond. In the S4PST ecosystem, sustainability means maintaining the unique capabilities provided by the highly specialized core development teams at national laboratories and universities, essentially the few people who understand and can make critical software evolve for future needs. This set of individuals forms the foundation of DOE software and for any attempt to provide additional value to its ecosystem.

- Acknowledge the importance of effective training practices to attract a diverse future workforce to be stewards of DOE software. In the current landscape, DOE needs are unique when compared to mainstream educational curricula driven by AI, high-level languages, etc. We need to reinforce outreach programs and activities (e.g. internships, tutorials, hackathons) and strengthen the national labs and university synergies and collaborations. Our PIER plans need to address and tackle the niche nature of our software and required skills.

- Extend and maintain modern programming practices judiciously. These include, but are not limited to: V&V testing, correctness, CI/CD to the S4PST ecosystem as new architectures, compilers, and programming models evolve to align with the needs of DOE science. Hence investing in a more proactive and predictive, than a reactive, software ecosystem.
- We need to implement a governance model, including a steering committee, to ensure the sustainability of the software stack. This is still a pending item as we understand that a sustainability effort is different from the uniqueness of ECP.
- We understand that software evolves rapidly and we need to be on the lookout for how new emerging technologies (e.g. generative AI, modern languages, and ecosystems) can impact the overall sustainability effort as we continue moving towards an AI-dominated computational landscape.

Acknowledgement

This work is supported by the S4PST Software Sustainability Seed awarded by the US Department of Energy Office of Science (SC) Advanced Scientific Computing Research under SC's Funding Opportunity Announcement DE-FOA-0002844. We would like to thanks Hartwig Anzt, Director of the Innovative Computing Laboratory at the College of Engineering of the University of Tennessee for hosting the workshop event.

References

[1] COLABS: Collaboration for Better Software (for Science). https://colabs-science.github.io/.

[2] CppCon web page. cppcon.org. Accessed: 2022-12-11.

[3] E4S Project web page. https://e4s-project.github.io/. Accessed: 2022-12-11.

[4] ECP-IDEAS web page. https://ideas-productivity.org/ideas-ecp/. Accessed: 2022-12-11.

[5] JuliaCon web page. https://juliacon.org/. Accessed: 2022-12-11.

[6] OSSF: Open Scientific Software Foundation. https://software4science.org/.

[7] PESO: Toward a Post-ECP Software-Sustainability Organization. https://leadershipscientificsoftware.github.io/PESO/PESO.html.

[8] RustCon web page. https://rustconf.com/. Accessed: 2022-12-11.

[9] STEPS: Sustainable Tools Ecosystem Project. https://ascr-step.org/leadership/.

[10] SWAS: Center for Sustaining Workflows and Application Services. https://swas.center/.

[11] xSDK web page. https://xsdk.info. Accessed: 2022-12-11.

[12] James Ang, Andrew A. Chien, Simon David Hammond, Adolfy Hoisie, Ian Karlin, Scott Pakin, John Shalf, and Jeffrey S. Vetter. Reimagining codesign for advanced scientific computing: Report for the ascr workshop on reimagining codesign. https://www.osti.gov/biblio/1822199, 4 2022.

[13] ASCR-STEP.org. The sustainable tools ecosystem project. https://www.ascr-step.org/, July 2023.

[14] John Bachan, Scott B. Baden, Steven Hofmeyr, Mathias Jacquelin, Amir Kamil, Dan Bonachea, Paul H. Hargrove, and Hadia Ahmed. UPC++: A high-performance communication framework for asynchronous computation. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 963–973, 2019. https://doi.org/10.25344/S4V88H.

[15] Victor R. Basili, Jeffrey C. Carver, Daniela Cruzes, Lorin M. Hochstein, Jeffrey K. Hollingsworth, Forrest Shull, and Marvin V. Zelkowitz. Understanding the high-performance-computing community: A software engineer's perspective. *IEEE Software*, 25(4):29–36, 2008.

[16] Michael Bauer, Sean Treichler, Elliott Slaughter, and Alex Aiken. Legion: Expressing locality and independence with logical regions. In *SC '12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2012.

[17] D. A. Beckingsale, J. Burmark, R. Hornung, H. Jones, W. Killian, A. J. Kunen, O. Pearce, P. Robinson, B. S. Ryujin, and T. R. W. Scogland. RAJA: Portable performance for large-scale scientific applications. In *2019 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC)*, 2019.

[18] Dan Bonachea and Paul H. Hargrove. GASNet-EX: A High-Performance, Portable Communication Library for Exascale. In *Proceedings of Languages and Compilers for Parallel Computing (LCPC'18)*, volume 11882 of *LNCS*. Springer, October 2018. https://doi.org/10.25344/S4QP4W.

[19] *Caffeine: CoArray Fortran Framework of Efficient Interfaces to Network Environments*. https://go.lbl.gov/caffeine.

[20] Jack Dongarra, Robert Graybill, William Harrod, Robert Lucas, Ewing Lusk, Piotr Luszczek, Janice McMahon, Allan Snavely, Jeffrey Vetter, Katherine Yelick, et al. DARPA's HPCS program: History, models, tools, languages. In *Advances in Computers*, volume 72, pages 1–100. Elsevier, 2008.

[21] Jack Dongarra et al. The International Exascale Software Project roadmap. *The International Journal of High Performance Computing Applications*, 25(1):3–60, 2011.

[22] Thomas M Evans, Andrew Siegel, Erik W Draeger, Jack Deslippe, Marianne M Francois, Timothy C Germann, William E Hart, and Daniel F Martin. A survey of software implementations used by application codes in the exascale computing project. *The International Journal of High Performance Computing Applications*, 36(1):5–12, 2022.

[23] Hal Finkel and Ignacio Laguna. Report of the workshop on program synthesis for scientific computing. https://arxiv.org/abs/2102.01687, 2021.

[24] MPI Forum. MPI: A message-passing interface standard version 4.0. https://www.mpi-forum.org/docs/mpi-4.0/mpi40-report.pdf, 2021.

[25] GASNet home page. https://gasnet.lbl.gov.

[26] William F. Godoy, Steven E. Hahn, Michael M. Walsh, Philip W. Fackler, Jaron T. Krogel, Peter W. Doak, Paul R. C. Kent, Alfredo A. Correa, Ye Luo, and Mark Dewing. Software engineering to sustain a high-performance computing scientific application: QMCPACK, 2023.

[27] William F. Godoy, Norbert Podhorszki, Ruonan Wang, Chuck Atkins, Greg Eisenhauer, Junmin Gu, Philip Davis, Jong Choi, Kai Germaschewski, Kevin Huck, Axel Huebl, Mark Kim, James Kress, Tahsin Kurc, Qing Liu, Jeremy Logan, Kshitij Mehta, George Ostrouchov, Manish Parashar, Franz Poeschel, David Pugmire,

Eric Suchyta, Keichi Takahashi, Nick Thompson, Seiji Tsutsumi, Lipeng Wan, Matthew Wolf, Kesheng Wu, and Scott Klasky. Adios 2: The adaptable input output system. a framework for high-performance data management. *SoftwareX*, 12:100561, 2020.

[28] William F. Godoy, Pedro Valero-Lara, T. Elise Dettling, Christian Trefftz, Ian Jorquera, Thomas Sheehy, Ross G. Miller, Marc Gonzalez-Tallada, Jeffrey S. Vetter, and Valentin Churavy. Evaluating performance and portability of high-level programming models: Julia, Python/Numba, and Kokkos on exascale nodes, 2023.

[29] Thomas Huber, Swaroop Pophale, Nolan Baker, Michael Carr, Nikhil Rao, Jaydon Reap, Kristina Holsapple, Joshua Hoke Davis, Tobias Burnus, Seyong Lee, David E. Bernholdt, and Sunita Chandrasekaran. Ecp sollve: Validation and verification testsuite status update and compiler insight for openmp, 2022.

[30] A. M. Jarmusch, A. Liu, C. Munley, D. Horta, V. Ravichandran, J. Denny, and S. Chandrasekaran. Analysis of validating and verifying openacc compilers 3.0 and above, 2022.

[31] Douglas Kothe, Stephen Lee, and Irene Qualters. Exascale computing in the united states. *Computing in Science & Engineering*, 21(1):17–29, 2018.

[32] Douglas Kothe, Stephen Lee, and Irene Qualters. Exascale computing in the united states. *Computing in Science & Engineering*, 21(1):17–29, 2019.

[33] Satoshi Matsuoka, Jens Domke, Mohamed Wahib, Aleksandr Drozd, Andrew A. Chien, Raymond Bair, Jeffrey S. Vetter, and John Shalf. Preparing for the future—rethinking proxy applications. *Computing in Science & Engineering*, 24(2):85–90, 2022.

[34] OpenACC-Standard.org. The OpenACC application program interface version 3.3. https://www.openacc.org/sites/default/files/inline-images/Specification/OpenACC-3.3-final.pdf, November 2022.

[35] OpenMP Architecture Review Board. OpenMP application program interface version 5.2. http://www.openmp.org/mp-documents/spec30.pdf, November 2021.

[36] Katherine Rasmussen, Damian Rouson, Naje George, Dan Bonachea, Hussain Kadhem, and Brian Friesen. Agile Acceleration of LLVM Flang Support for Fortran 2018 Parallel Programming. In *Research Poster at the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC22)*, Nov 2022. https://doi.org/10.25344/S4CP4S.

[37] Damian Rouson and Dan Bonachea. Caffeine: CoArray Fortran Framework of Efficient Interfaces to Network Environments. In *Proceedings of the Eighth Annual Workshop on the LLVM Compiler Infrastructure in HPC (LLVM-HPC2022)*, November 2022. https://doi.org/10.25344/S4459B.

[38] Dominik Straßel, Philipp Reusch, and Janis Keuper. Python workflows on hpc systems. In *2020 IEEE/ACM 9th Workshop on Python for High-Performance and Scientific Computing (PyHPC)*, pages 32–40, 2020.

[39] Christian R. Trott, Damien Lebrun-Grandié, Daniel Arndt, Jan Ciesko, Vinh Dang, Nathan Ellingwood, Rahulkumar Gayatri, Evan Harvey, Daisy S. Hollman, Dan Ibanez, Nevin Liber, Jonathan Madsen, Jeff Miles, David Poliakoff, Amy Powell, Sivasankaran Rajamanickam, Mikael Simberg, Dan Sunderland, Bruno Turcksin, and Jeremiah Wilke. Kokkos 3: Programming model extensions for the exascale era. *IEEE Transactions on Parallel and Distributed Systems*, 33(4):805–817, 2022.

[40] UPC++ website. https://upcxx.lbl.gov.

[41] Vetter, Jeffrey S. et al. . Extreme Heterogeneity 2018 - Productive Computational Science in the Era of Extreme Heterogeneity: Report for DOE ASCR Workshop on Extreme Heterogeneity. https://www.osti.gov/biblio/1473756, 12 2018.