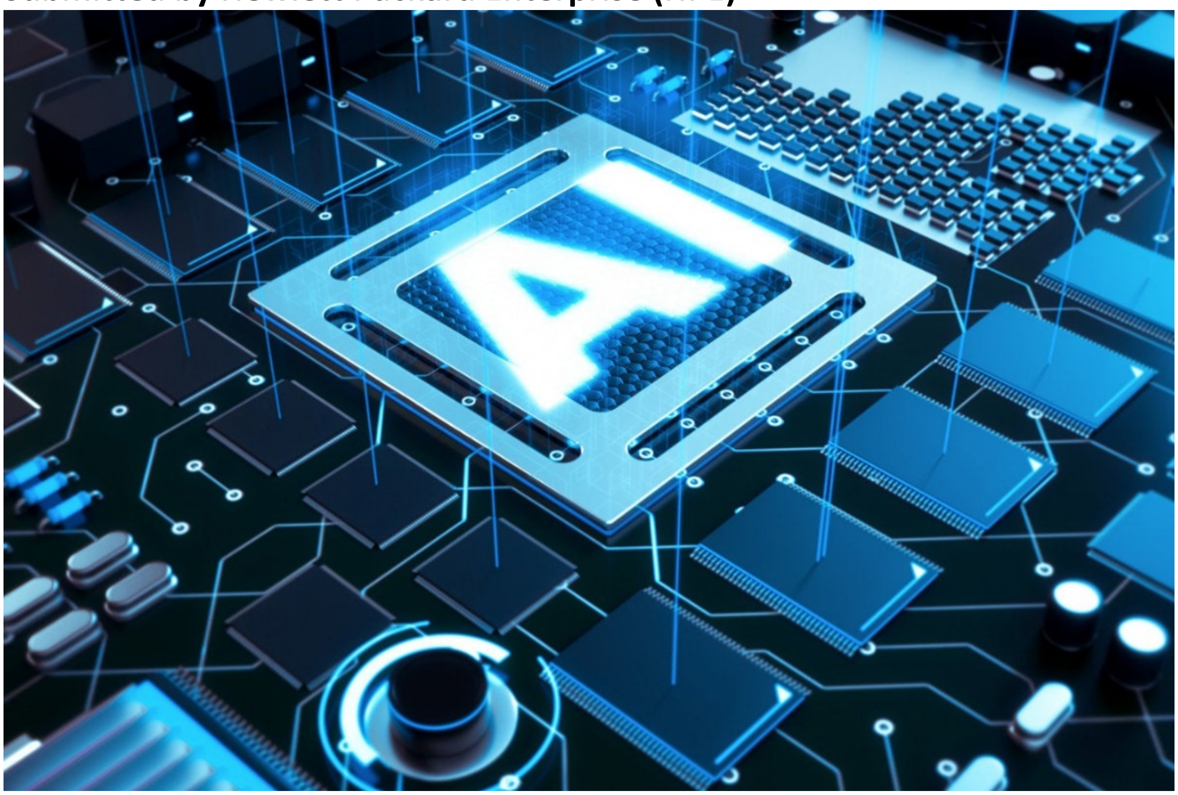
Hewlett Packard Enterprise

Response to the Department of Energy (DOE) for the Advanced Scientific Computing Research (ASCR) Program Request for Information (RFI) Response

Submitted by Hewlett Packard Enterprise (HPE)



December 13, 2021

This page intentionally left blank

Table of Contents

List of Figures

List of Tables

This page intentionally left blank

Introduction

Hewlett Packard Enterprise (HPE) believes that many of the initiatives being explored within the Office of Advanced Scientific Computing Research (ASCR), Office of Science, Department of Energy are very closely aligned to those of our own research divisions, and much of the research and development spawned by these efforts are also in our best national interest. That's why we appreciate the opportunity to respond to the Request for Information (RFI) to assist in the Stewardship of Software for Scientific and High-Performance Computing.

HPE, with the acquisition of Cray and SGI, along with our advanced research labs (Hewlett Packard Labs), is committed to investing in the High Performance Computing (HPC) and Artificial Intelligence (AI) segments of our industry. We are deploying some of the largest and most performant HPC systems in the world and are an active participant in the roll out of the next generation of supercomputing for the Exascale computing era. According to TOP500.org, HPE leads industry vendors for HPC performance share at 18.6%.

HPE offers an extensive software portfolio, so we are well qualified to respond and are active members of the large community of scientists, engineers, and software professionals served by the proposed stewardship. We also believe that meeting the future needs of both ASCR's research program and the computational-science performed in service of the nation's scientific enterprise depends on leveraging a sophisticated, highly interconnected, professionally developed software ecosystem resulting from substantial past investments.

Industry Input

Software Dependencies

(1) Software dependencies and requirements for scientific application development and/or research in computer science and applied mathematics relevant to DOE's mission priorities: What software packages and standardized languages or Application Programming Interfaces (APIs) are current or likely future dependencies for your relevant research and development activities? What key capabilities are provided by these software packages? What key capabilities, which are not already present, do you anticipate requiring within the foreseeable future? Over what timeframe can you anticipate these requirements with high confidence? What are the most significant foreseeable risks associated with these dependencies and what are your preferred mitigation strategies? When responding to these questions, please describe the scope of the relevant research and development activities motivating the response.

HPE Response:

The HPE Cray Programming Environment (CPE) is a comprehensive set of tools and libraries that allow HPC developers and scientists to compile, debug, optimize, and execute applications at scale across a wide range of CPU and GPU architectures. CPE tools and libraries are built around industry standard languages and interfaces, and HPE is an active participant in these communities and standards bodies. In providing one of the industry's most capable programming environments, CPE incorporates a variety of software packages that are important to our scientific and government end users, particularly DOE scientists and the multi-disciplinary users that DOE centers support. The tables below summarize the open-source packages and open standards directly used by CPE that we believe will continue to be critical to many DOE scientists and DOE facility end users.

Table 1. Open-Source Software Package Capabilities

Software Packages	Key Capability
MPICH	High-performance reference implementation of the MPI standard.
LLVM	High-performance multi-platform extensible compiler infrastructure.
PAPI	Portable interface to hardware counters for performance tools.
GDB	Portable program debugger.
Valgrind	An instrumentation framework for building dynamic analysis tools.
HDF5	Library and portable storage format for high-performance I/O.
Parallel NetCDF	Library for high-performance I/O with files stored in CDF format.
FFTW	Library for fast Fourier transforms.
Kokkos Core	C++ library that provides portable high performance across processor and accelerator architectures.
RAJA/Umpire/CHAI/BLT	C++ libraries and tools that provide portable high performance across processor and accelerator architectures.
CMake	Tools for building, testing, and packaging software.

Software Packages	Key Capability
Spack	Tools for installing and managing versions and configurations of software packages.
PyTorch	A machine learning framework that accelerates the path from research prototyping to production deployment.
TensorFlow	An end-to-end platform for machine learning.

Table 2. Open Standardized Languages and API Capabilities

Standardized Languages and APIs	Key Capability
C/C++	High-performance compiled programming language.
Fortran	High-performance compiled programming language.
MPI	API for distributed-memory message passing.
OpenMP	Compiler directives and API for multithreading on processors and accelerators.
OpenSHMEM	API for distributed shared memory using a partitioned global address space (PGAS).
Libfabric	Middleware API for high-performance networking.
Python	General-purpose high-level interpreted programming language.
R	High-level interpreted programming language for statistical computing and graphics.
BLAS	API for high-performance vector and matrix operations.
LAPACK	API for dense-linear-algebra solvers.
BLACS	API for communication patterns needed for distributed-memory dense linear algebra.
ScaLAPACK	API for dense-linear-algebra solvers that support distributed-memory parallelism.

In addition to tools that are already broadly in use today, we expect the rapid increase in the use of AI techniques within HPC to continue, along with the related use of HPC resources for analyzing large datasets. It is critical to support these types of workloads as part of future HPC/AI programming environments. Future HPC/AI programming environments will need to include tools, libraries, and languages that support AI-specific accelerators, in addition to more traditional CPU and GPU architectures. To achieve this, we see a need for new and enhanced APIs and libraries that enable portability across architectures, developed through industry/DOE collaborations and partnerships. The pace of innovation in compute architectures is both an opportunity and risk to application developers and to the HPC software ecosystem. CPE offers developers and users an ecosystem focused on usability and performance with portability across architectures. Such an ecosystem lessens the risk to application developers and DOE mission priorities. In addition to evolving CPE, HPE intends to continue co-design and standardization efforts with the DOE to help mitigate future risks.

Software Integrity

(2) Practices related to the security and integrity of software and data: What strategies and technology do you employ, or intend to employ in the foreseeable future, to ensure the security and integrity of your software and its associated provenance metadata? What capabilities do you provide, or intend to provide in the foreseeable future, to assist users of your software with ensuring scientific reproducibility, recording the provenance of their work products, securing their information, protecting the privacy of others, and maintaining the integrity of their results?

HPE Response:

HPE has been attentive to software and firmware integrity for most of the last decade. We have contributed to NIST and Trusted Computing Group's standards for Trusted Protection Module (TPM), firmware, software, and runtime integrity. This intersects with our overall supply chain risk management strategy, as well as deep attention to vulnerability management, open-source provenance risk management, secure development lifecycle and security testing. The result is integrity assurance at runtime, based on HPE's Hardware Root of Trust (HWRoT) silicon security chip, and the HWRoT integrity is based in our formal Supply Chain Risk Management processes.

HPE is deliberately becoming even more active in the Open-Source ecosystem related to security and software supply chain integrity. We are a founding member in the Open-Source Security Foundation (OpenSSF), and well as a participant in the bootstrapping funding for the Linux Foundation Sigstore project[1]. Additionally, we participate in the Cloud Native Computing Foundation's (CNCF) Security Technical Advisory Group and interact regularly with industry partners including but not limited to the major OS vendors, etc regarding software supply chain integrity.

Our broad use of CNCF software both in the mainline enterprise with Greenlake and in the Cray Shasta Exascale systems with Cray Systems Management (which has been formally open sourced), has been an evolutionary driver for our internal software supply chain including Continuous Integration Continuous Development (CICD) approaches to automation with respect to software integrity. We have been actively working with the Sigstore project since its launch to refine approaches to enterprise deployment of the tooling, including the addition of open-source SPIFFE / SPIRE (cloud native secure identify management of containers at run time) based workload attestation in the CICD platform operations. We have also begun the process of aligning our future CICD architecture and ecosystems to bring in the SLSA[2] framework for the formal assessment of build system integrity and capabilities. DOE entities that produce software or consume open-source software, must necessarily follow a similar path. This is particularly important considering the accelerating convergence of HPC/scientific computing with the CNCF software ecosystem, and as policy level requirements begin to dictate certain actions and capabilities. We expect that a large amount of our experiential capital and some level of tooling that we produce would be available to the DOE as they undertake related efforts. In fact, some of this has already started to occur as the result of our adoption of the Sigstore tooling and its use for the Shasta CSM software. We expect this to continue and expand as we mature the Cray System Management platform along with the community of users and through efforts related to AI for Science.

Data provenance presents the same challenges as the software side in most cases. Data provenance approaches and tooling lag that of software at this point in the context of a universally accepted set of resources or standards. Only very recently has a tooling like the Sigstore project shown up on the software side of the house. DOE will need

[1] Sigstore - Linux Foundation project for a new standard for signing, verifying, and protecting software supply chains.

[2] SLSA - A security framework from source to service, giving anyone working with software a common language for increasing levels of software security and supply chain integrity. https://slsa.dev

to aggressively engage with the open source community and industry to work toward a set of standards and tooling that assure cryptographic verification of data sets occurs comprehensively. This situation is acute in AI and presents a core challenge for the AI for Science effort. Storage systems will have to evolve, along with the way in which the provenance build-up and verification takes place. HPE is planning to present proposals in the context of the AI for Science program that begin to address this topic.

The Solar Winds incident has resulted in a focused effort by the Federal Government to address the current state of software supply chain integrity through both RFI's and Executive Order. HPE has been active in responding to RFI's from NTIA on the topic of SBOM's in general and has been successful in advocating for increased Federal participation in the CNCF ecosystem specifically. HPE is actively implementing tooling and processes necessary to comply with the executive order for delivery of SBOM's with software products. We continue to interact regularly with the DHS/CISA staff on the topic of software supply chain integrity.

Scientific and HPC Infrastructure

(3) Infrastructure requirements for software development for scientific and high-performance computing: What infrastructure requirements do you have in order to productively develop state-of-the-art software for scientific and high-performance computing? These requirements might include access to testbed hardware, testing allocations on larger-scale resources, hosting for source-code repositories, documentation, and other collaboration tools. What are the key capabilities provided by this infrastructure that enables it to meet your needs? What key capabilities, which are not already present, do you anticipate requiring within the foreseeable future? Over what timeframe can you anticipate these requirements with high confidence? What are the most-significant foreseeable risks associated with this infrastructure and what are your preferred mitigation strategies? When responding to these questions, please describe the scope of the relevant research and development activities motivating the response.

HPE Response:

The architectures of processors and accelerators are diverse. Scientific and HPC software developers need access to a wide range of architectures for development, testing and support. Extending the useful life of software and extending the software beyond the systems on which it was developed is an important contribution to the wider open science community. DOE should provide access to testbed systems with diverse technologies and software environments, as well as allocations on leadership class systems, for teams who develop, extend or maintain scientific software and middleware. The testbeds should prioritize developer productivity with documentation, training materials, and hackathons. The testbeds should enable a range of usage style including interactive use. These systems should include programming environments for both compiled and interpreted languages, debuggers, and performance analysis tools. DOE should drive efforts within the scientific and high-performance communities to define a "control point" for standardizing the software stack. The expansion of the diversity of processor architectures has caused an expansion in the diversity of the software stacks needed to program the new architectures. Each vendor needs to optimize the kernel to maximize the performance of the architecture. A standardized point within the software stack to which processor vendors and application developers can design will increase overall community productivity. Projects such as Apache TVM can provide a solution to this siloed diversity.

AI-enhanced "Software 2.0" Development and Test

The pervasive use of AI and ML represents the beginning of a fundamental shift toward a new way of developing software, which is usually referred to as "Software 2.0"[3]. A classical software stack relies on traditional imperative languages such as C, C++, Fortran or Python, and a large set of tools is available for development of testing of traditional software for scientific applications. The new "Software 2.0" stack involves the creation of more abstract models, such as the weight of a large neural network, that no human programmer is expected to manually touch.

[3] [Karp17] A. Karpathy. "Software 2.0" https://karpathy.medium.com/software-2-0-a64152b37c35

Instead, the role of a “Software 2.0” programmer is to specify high-level goals, a model architecture, and a sample data set to use to train the model to achieve the target behavior. In this case, most of the software development flow (traditionally involving coding, compilation and testing) is replaced by curating, cleaning and labeling data sets.

While this transition has initially started in areas such as visual and speech recognition or machine translation, we see an accelerating trend towards uses of “Software 2.0” techniques in scientific computing, such as AI-enhanced simulation. We expect this transition to have a significant impact on software development for scientific applications in the next 3-5 years, but tools and methodologies are lagging[4]. An increased ASCR focus on AI software development and testing would be important to help close the gap. Two important areas are:

- *Infrastructure for ML development*. The Software 2.0 “compilation” process involves training several large networks over a very large training set. This is a compute-intensive and data-intensive development flow, which requires specialized accelerators and fast access to petabytes of data. Even with accelerators, it can take weeks to train a large model, and availability of a large ML development cluster would be important to improve the programmer productivity.
- *Testing of ML models*. Because of the complexity of large ML models and the reliance on training data, it can be difficult to determine the quality of a model. Testing practices of ML models are still in the research realm, and significant advances are required to introduce relevant metrics for scientific applications, such as UQ.

One advanced area we are researching at HPE (in Hewlett Packard Labs) is the use of “*AI for testing AI*”. We are developing “*smart test agents*” that can dynamically alter the static test data sets with typical distortions experienced at deployment, based on the responses of the ML models under test. This mechanism stresses the ML model to find vulnerabilities to measure its robustness and stability. Using Reinforcement Learning we can handle the complexity of altering the test data set with different distortions, while providing observability and explainability to identify the greatest points of vulnerabilities for the ML models under test. This approach also generates additional synthetic training data that can be fed into the software development and test process to fix the model weakness. Generative Adversarial Network (GAN) based surrogates can be used by test agents to modify test datasets with targeted attributes to evaluate bias, drift, robustness, or other aspects of trustworthiness.

These concepts can also be used to enhance development platforms (such as the HPE Cray AI Development Platform, built on Determined.AI) as an advanced “Auto ML” tool. In addition to traditional phases such as hyperparameter optimization or neural architecture search (NAS), the new testing techniques can incorporate various trustworthiness attributes that are specific to scientific applications, and the analytics necessary to track them.

Community Software

(4) Developing and maintaining community software: How much additional effort is needed to develop and maintain software packages for use by the wider community above the effort needed to develop and maintain software packages solely for use in specific research projects or for internal use? What tasks are the largest contributors to that additional effort? What are the largest non-monetary impediments to performing this additional work? How is any such additional effort currently funded? How does that funding compare to a level of funding needed to maximize impact?

HPE Response:

Open-source projects are a do-it-yourself adventure and often only succeed when there is a strong enterprise support system behind it. (e.g. include RedHat for Linux, Google for TensorFlow, Kubernetes etc.). Within the DOE community, successful software projects that have potential for reuse should be prioritized to be placed into a hardening process sustainability. This can be achieved via Linux/Apache foundation style project management and resource alignment.

[4] [Louk19] M. Loukides, B. Lorica. “The road to Software 2.0”. https://www.oreilly.com/radar/the-road-to-software-2-0/

Currently, the DOE is in a similar situation to most legacy enterprises, having a significant portfolio, and the associated management challenges, of internal and proprietary software. Making this more complicated is that within the DOE user community, the ratio of open source to proprietary software use is rapidly shifting to the open source side. To keep up with these shifts, DOE has two primary challenges:

1) Governance

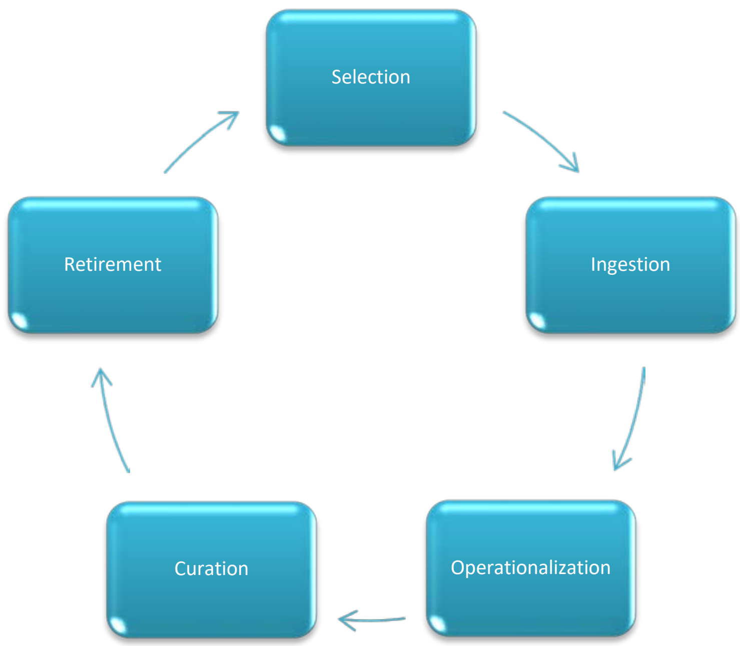
2) Organizational structure and resource allocation

Governance

DOE will need a governance model that contemplates two distinct scenarios, the first being the consumption of OSS, the second being the portfolio management of the proprietary software that is necessary for use cases not covered through available open source. However, when considered from afar the internal case ends up being a microcosm of the OSS governance process.

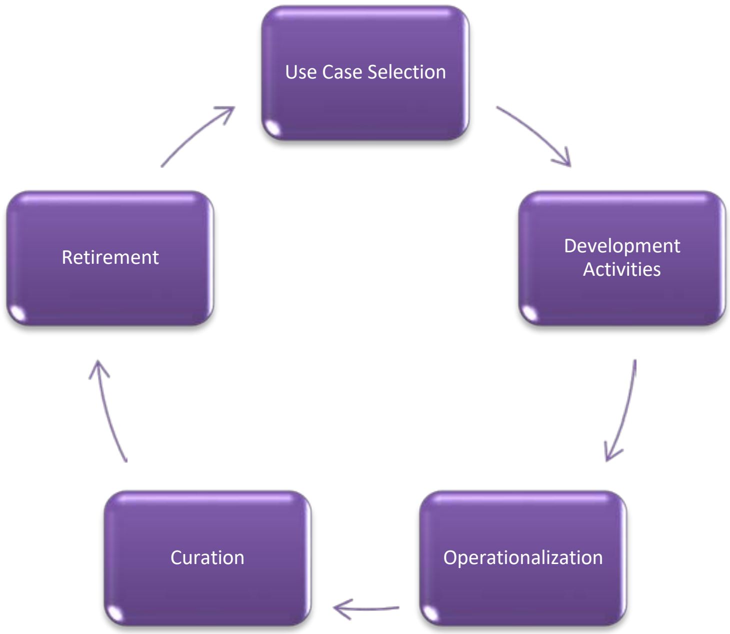
Governance of OSS consumption / participation is best considered as a lifecycle.

Figure 1 . OSS Consumption Governance



Internal projects follow the same pattern with the only difference being the initial activities.

Figure 2 . Internal Projects



Most large OSS projects have a hierarchical/tiered governance model.

Figure 3. Kubernetes Example

This governance structure allows for a decentralized approach to managing the software lifecycle with a large, diverse group of participants across organizational and geographic boundaries, while maintaining a reasonable level of consistency and control of a complex ecosystem. DOE's challenges are quite similar in nature. From a governance perspective, it's optimal to embrace existing structures that many people are already familiar with and that align organically to the software ecosystems that are driving modern enterprise operations.

There is little to no question that over time OSS will come to comprise the majority, if not the entirety platform level software within the labs and DOE itself. In that dimension “Selection” of the core OSS components to be consumed must become a DOE level governance process, notionally, the “DOE Infrastructure Software Steering Committee" to avoid duplicative effort at various DOE sites and the introduction of unnecessary integration complexity which translates to wasted resource investment and poor financial stewardship.

Further, precisely because DOE desires a decentralized landscape that fosters innovation and a certain level of competition between labs, the hierarchical nature of the above governance structure is important. Within the domain specific SIG’s for example new ideas can be presented and moved through the governance process for ultimate adoption (or investment).

In any governance model there is hazard in the creation/perpetuation of an overly bureaucratic disconnected process that ultimately impedes progress. The one word that best describes the challenges of living in the current world of software is ***velocity.*** DOE’s trickiest challenge in designing a governance model is how to put a structure together that embraces the rate at which modern OSS software evolves and does so in a way that can accommodate the realities of the software supply chain fidelity requirements and ecosystem that are coming into play.

Organizational Structure and Resource Allocation

Of course, the governance model is only half the battle. What most enterprises fail to internalize and act on are the required organizational and resource allocation changes that are critical to success in the current software ecosystem. Both DOE and its National Lab-based computing facilities are going to have to look at organizational structure changes that start to align them to the generalized governance models above. In aligning the organizational structures at DOE and the labs to the governance model presented above, activities including portfolio management and active curation become readily achievable as does a harmony with the way in which many OSS sources are governed outside of DOE and the labs. Changes to vendor contracting mechanisms must be factored in as the rate of change in the broader software ecosystem is fundamentally incompatible with legacy contracting models.

Of additional note, as DOE and the labs pursue the AI for Science effort and the associated desire for more closely connected and broadly available systems resources, DOE and the labs will need to adopt many of the functional/operational/security elements of modern (i.e., “Cloud”) platform engineering. Over the course of the Shasta development and rollout, we’ve heard many times that there is a desire to pursue a high availability/zero-downtime approach to the Shasta systems. Operating in a HA/Zero-downtime model requires a level of resource commitment and organizational structure that is not currently in place in the computing facilities.

Ultimately, the software governance model needs to facilitate the strategic objectives of the DOE within the context of the software ecosystem itself and provide a valuable feedback loop to evolve the strategic plan. The key takeaways for this are: 1) Don’t fight the ecosystem, work in a common governance model with the outside world and adapt internally. 2) Embrace velocity, the fastest way to fail is to ignore this driver. 3) Keep broad mandates to the minimally essential a bare minimum so as not to stifle innovation.

Diverse and Inclusive Environment

(5) Challenges in building a diverse workforce and maintaining an inclusive professional environment: What challenges do you face in recruiting and retaining talented professionals to develop software for scientific and high-performance computing? What additional challenges exist in recruiting and retaining talented professionals from groups historically underrepresented in STEM and/or individuals from underserved communities? What challenges exist in maintaining inclusivity and equity 8 in the development community for scientific and high-performance computing software? What successful strategies have you employed to help overcome these challenges? What opportunities for professional recognition and career advancement exist for those engaged in developing scientific and high-performance computing software?

HPE Response:

The challenge of recruiting and retaining talented professionals to develop software for scientific and high-performance computing is essentially overcoming the supply and demand dilemma. Too often, there simply more open jobs for HPC software engineers than there are candidates. One mechanism to overcome this may be for employers to provide new hires having closely-adjacent skills with on-the-job training and skills-development opportunities that will allow them to succeed within HPC and scientific computing. An over-reliance on demonstrated experience instead of aptitude can limit the talent pool and unnecessarily eliminate high-potential candidates. Furthermore, the growth of the industry continues to outpace the ability of our universities to deliver diverse graduates in the engineering and scientific field.

We've also determined that traditional recruiting practices can prove challenging as a method for identifying and bringing candidates who identify with groups that are under-represented in HPC through the door. To overcome this, HPE has learned that we must cast our talent acquisition net in places where diverse candidates are. Target the high potential candidates wherever they are, provide them the resources and support they need to succeed, and establish the appropriate policies and values that support an inclusive workplace for all employees.

In today's competitive labor environment, the ability to attract and retain talent will signal success in this endeavor. If your current recruitment policy doesn't include diversity and inclusion as a key piece of the talent acquisition puzzle, it should.

HPE recommends that an advisory board with membership in a diverse community of stakeholders, including membership from individuals that identify with groups that are under-represented in scientific computing, be established to work with the office responsible for implementation, deployment, and administration. To ensure the ASCR effort is broadly available throughout the United States, in urban, suburban and rural communities, and especially within communities that are traditionally underrepresented in the development and use of computing technology, the federal government will need the participation of nearly every agency.

There are also a number of diversity organizations with a focus on the promotion and development of a technical workforce that we've successfully engaged at HPE for our diversity recruiting initiatives. These have included the BDPA, formally known as Black Data Processing Associates (BDPA), ITSMF and HITEC among others. These groups may be leveraged to provide opportunities for professional recognition and career advancement for your diverse workforce engaged in developing scientific and high-performance computing software. The following table will provide additional information about the referenced diversity groups:

Table 3. Diversity Workforce Groups

Organization	Web site for more information
BDPA	https://www.bdpadc.org/about-us
ITSMF	https://legacy.itsmfonline.org/about-itsmf/
HITEC	https://hitecglobal.org/page/About

Technology Transfer and Funding

(6) Requirements, barriers, and challenges to technology transfer, and building communities around software projects, including forming consortia and other non-profit organizations: ASCR recognizes that successful software for scientific and high performance computing often has many stakeholders, including academic research activities, research laboratories, and industry. Moreover, while DOE has provided funding for the development of a significant number of foundational software packages within the modern software ecosystem for scientific and high-performance computing, as the complexity of the software ecosystem continues to increase, and number of stakeholders has grown, ASCR seeks to understand how it might encourage sustainable, resilient, and diversified funding and development models for the

already-successful software within the ecosystem. Such models include, depending on circumstances that ASCR seeks to better understand, both the private sector and non-profit organizations. Non-profit organizations include both charitable organizations e.g., those with 501(c)(3) status) and R&D consortia (e.g., those with 501(c)(6) status). What are the important characteristics and components of sustainable models for software for scientific and high-performance computing? What are key obstacles, impediments, or bottlenecks to the establishment and success of these models? What development practices and other factors tend to facilitate successful establishment of these models?

HPE Response:

In our response to question 4, we outlined ways in which DOE could leverage established practices in the open source software community to manage DOE community-developed projects and expand their reach beyond the limited set of users that have adopted them.

In addition, ASCR should take a developer-friendly encouraging stance to CRADA's, copyrights and software IP – encouraging developers to think of impact beyond research and academic interest.

Stewardship Scope

(7) Overall scope of the stewardship effort: The section labeled Potential Scope, mentioned earlier in the RFI, outlines activities that ASCR currently anticipates potentially including in future programs stewarding the software ecosystem for scientific and high-performance computing. Are there activities that should be added to, or removed from, this list? Are there specific requirements that should be associated with any of these activities to ensure their success and maximize their impact?

HPE Response:

Scientific software stewardship has been and will remain a multi-faceted effort. The focus on support of practitioners at the intersection of computer engineering and the range of scientific disciplines that encounter computational challenges in doing their research must continue to be a primary consideration. But it is also important to note that there is a confluence of factors undergoing broad change, some for the first time in decades, which will drive change not only in computer science, but more broadly in the pursuit of all the sciences and the behaviors and aspirations of the scientists who pursue them.

The second era of Moore's Law scaling--the post-Dennard equivalent scaling era--sought to continue predictable performance increase via greater and greater monolithic integration, which in turn lead to homogeneity and monoculture, offering a stable architectural basis for software development. Stability has its own cost, as the security model has also remained static even as the threat landscape continues to evolve. As this era draws to a close, we are seeing a proliferation of novel architectural approaches that are now being explored, reintroducing a diverse, heterogeneous computational ecology. What we compute on is changing.

Simultaneously, we observe that the simulation and modeling which have been the backbone of scientific computation are now complimented by massive data analytics driven by increasingly capable sensors, facilities, and data capture technologies. Whether a massive telescope array, a particle collider, or a PCR instrument capable of sequencing an evolving virus population…scientific instrumentation is enabling the outputs of HPC simulation and modelling to be paired with their physical analogues. To this is added the rise of AI/ML techniques used pervasively to discover correlations, solve equations, manage experiments, and optimize operational efficiencies. What we need to compute is changing.

Looking deeper into the scale and proliferation of data, we see that it is already exceeding the limits of even next generation networks to centralize. By the middle of the decade as much as 75%, three bytes out of four, of enterprise data may never be centralized in a data center, the term itself may become an anachronism. To even begin to consider analyzing a majority of the data generated, we will have to process it at the edge with the space, weight, power, and security challenges unique to each remote environment. Where we need to be able to compute is changing.

As the lifecycle of scientific software evolves from maintain monolithic codebases running in secured data centers of homogeneous hardware to composition of AI augmented data-driven workflows hosted from edge to cloud to exascale core and beyond, we need to solve for more than just physics and economics, we also need to solve for law: security, privacy, and ethics over the lifetime not just of an application, but over the duration of the impact of the societal decisions of the application. What we need to do to have measurable confidence in computing is changing.

Finally, we circle back to the practitioner. Development is now an agile, community-driven experience. For many professionals, their github commit logs and their status as maintainers on important packages and subsystems are their most prized achievements and the most valuable records of their impact an ingenuity. Jupyter notebooks, AI-augmented IDEs with auto-generators, and low-code/no-code frameworks will enable a broader set of scientists to be more directly connected to the highest performing compute systems. Who is computing is changing.

Looking at the scope of stewardship efforts through the lens of this confluence of changes and continuing to focus on the practitioners, we can look at the stewardship activities anticipated by ASCR and add to them new elements to reflect the changing face of computing:

- *Training* – prepare for a cloud-native, AI/ML augmented, workforce capable of composing workflows from edge to cloud to core. Add security, privacy and ethics training to the core curriculum.
- *Workforce Support* – Enabled and reward engagement with the global distributed open source development community. Fund and foster participation in standards, professional societies and communities of practice.
- *Infrastructure* – Providing infrastructure spanning edge to cloud to exascale core and beyond, incorporating the full range of heterogeneous accelerators. Adopt provable provenance in hardware and software supply chains including attestation down to the component level.
- *Curation* – Drive innovation and adoption of data management frameworks so that data sources can be maintained with confidence as first-class elements of advanced scientific workflows. Drive innovation in identity management and low power cryptography to allow attestation built up from the silicon and software.
- *Maintaining Situational Awareness* – Create communities of practice around workflows bringing together cross-disciplinary teams and encouraging application of heterogeneous accelerated edge-to-core computing. Ensure that the most challenging problems and the most innovative solutions are brought together.
- *Shared Engineering Resources and Project Support* – Research into broad application of AI/ML into the software generation cycle, adaptation of existing workflows to novel acceleration, and the operational efficiency of the edge to exascale infrastructure.

Management and Oversight of Software Ecosystem

(8) Management and oversight structure of the stewardship effort: What do you anticipate will be effective models for management and oversight of the scientific and high performance computing software ecosystem, and how would that management structure most-effectively interact with DOE and other stakeholders? In addition to DOE, who are the key stakeholders? How can the management structure coordinate with DOE user facilities and others to provide access to relevant testbed systems and other necessary infrastructure?

HPE Response:

There are several projects, involving many stakeholders, within the existing HPC software ecosystem that can serve as models of successful stewardship. MPI, for example, is a backbone for much of the DOE's simulation application portfolio. The MPI standard committee includes a diverse group of industry and DOE representatives with expert knowledge in the needs of application developers and capabilities of HPC interconnects. Developing a well-designed standard was critical to its success. Support from the DOE for open-sourced reference implementations, such as OpenMPI and MPICH, provided a stable basis for vendor-optimized implementations in addition to lowering the risk of having a single library with no alternatives.

Graphblas is another example demonstrating how collaboration on standard interfaces and reference implementations resulted in a successful addition to the HPC software ecosystem. Stakeholders in the high performance computing software ecosystem include other US government agencies, academia, the private sector as well as hardware and software vendors and cloud service providers.

The management and oversight processes associated with open source software should be adopted as part of an overall strategy for the HPC software ecosystem. Discipline to standardize key languages and APIs and provide or support open source reference implementations is needed to avoid fragmentation across hardware vendors. Standard bodies and reference implementations should include end users (e.g., scientific domain experts), application developers, HPC application support engineers, and vendors. Any stakeholders, such as vendors and other national agencies can further invest in implementations for specific hardware and/or performance targets. DOE should provide funding to standards bodies for developing standards and reference implementations. DOE should also provide development and test infrastructure directly to the teams developing reference implementations through allocations on DOE-managed systems specifically set aside for this purpose.

Assessment and Criteria

(9) Assessment and criteria for success for the stewardship effort: What kinds of metrics or criteria would be useful in measuring the success of software stewardship efforts in scientific and high-performance computing and its impact on your scientific fields or industries?

HPE Response:

The software stewardship effort has the potential to add value along the following dimensions for academic, industry and government stakeholders:

- Scientific – e.g., make discoveries possible
- Technological – e.g., create an ecosystem of innovation that includes for-profit and non-profit uses
- Scalability – e.g., ability to grow to more users, more partners, more contributors
- Collaborative – e.g., attract and develop talent

True success on the scientific, technological, access, scalability and collaboration dimensions can be measured using one or more of the following measures of success:

- Increased productivity and reduced time-to-insight
- Number of individual, small-business, non-profit, and corporate users
- Number of papers, press-releases, publications that acknowledge the resource
- Increasing activity among networks of vibrant communities of creative domain scientists and data scientists that can collaborate effectively
- Increase use of HPC leveraged AI in addressing high-priority, cross-cutting urgent national challenges such as climate change

Other

(10) Other: What are key obstacles, impediments, or bottlenecks to progress by, and success of, future development of software for scientific and high performance computing? Are there other factors, issues, or opportunities, not addressed by the questions above, which should be considered in the context of stewardship of the ecosystem of software for scientific and high performance computing?

HPE Response:

Additional obstacles that DOE may encounter in this effort include:

- The economics and consumption model of software engineering and performance optimization is less understood. A better understanding of ROI on re-usable software when documented, tracked, and rewarded will encourage broader community engagement. For example, horovod is an open-source AI codebase that draws from previous HPC optimizations and is of tremendous value to the commercial and scientific AI community.

- Commercial ISV HPC-codes become legacy too fast, making them unable to exploit newer architectures and programming models. DOE should encourage mechanisms for software vendors and system integrators to work together and continue to offer optimized performance.

Additional ideas DOE may consider:

- Enabling a “scientific software-as-a-service” model for some broadly used applications
- Adopting an API-culture for ease-of-use
- Establishing mechanisms by which software contributions can be treated as a research artifact in academia. In other words, offer “publication credit” or similar to document and reward authorship of and contributions to scientific and HPC software.
- Social engineering (broad-based recruiting, developing cross-domain abstractions) for software projects