The HDF Group

Submitted via regulations.gov to: RIN 86 FR 60021

December 13, 2021

Comments on Stewardship of Software for Scientific and High-Performance Computing

The HDF Group (HDFGroup) appreciates the opportunity to submit comments on the Department of Energy's (DOE's) *Stewardship for Scientific and High-Performance Computing*. HDFGroup has been creating HPC software since 1988, first as a university research group, and since 2006 as a not-for-profit company whose mission is to develop and support a software stack that provides rapid, easy and permanent access to complex data on every major platform and data scenario. The HDF Group supports technologies as old as 33 years, as new as yesterday, and everything in between.

HDF technologies include the HDF libraries and HDF cloud service software, almost all of it free and open source. They are used in almost every type of application that deals with data, and are part of global data standards in domains as wide-ranging as climate science, particle science, and brain science. Many HDF-based applications are mission-critical and must meet strict standards of performance and reliability. HDF is an ecosystem consisting of the core HDF technologies and thousands of third-party solutions including problem-specific software and more general-purpose tools as diverse as visualization, machine learning, and simulation.

HDFGroup commends DOE for this initiative as it speaks directly to existential concerns and it goes straight to the viability and sustainability of our organization, not to mention the critical needs of the many communities that rely on us. We want to contribute to the process with the following comments.

Topic (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?

The HDF Group uses GitHub for its distributed version control and source code management. Member accounts of The HDF Group organization use multiple-factor authentication (MFA) and commits to the HDF5 repository are digitally signed. Pull requests can be merged only after passing the CI test suite and the approval by two

410 E. University Avenue, Suite 200, Champaign, IL 61820
info@hdfgroup.org | www.hdfgroup.org

reviewers. The HDF Group regularly reviews results from static (e.g., Coverity) and dynamic (e.g., Valgrind) code analysis, and the CVE list, and takes appropriate action.

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?

To ensure HDF5 file integrity, HDF5 checksums all file metadata and (optionally) supports checksums on raw data.

HDF5 software comes with a *forward- and backward compatibility* guarantee, which means:

1. *Backward compatibility*: The most recent version of HDF5 software can read data from HDF5 files generated with any previous version of HDF5 software.
2. *Forward compatibility*: An older version of the HDF5 software can read HDF5 files generated with the most recent version of the software as long as no features that were introduced post-release of the older version are present in the file.

HDF5 files produced by a given version of HDF5 software are byte-identical for user data and are byte identical for internal metadata except for timestamps.

The HDF5 library is highly customizable through well-defined extension interfaces. Typical uses of these interfaces include the utilization of new data compression algorithms or the implementation of user data encryption. Through such extensions, HDF5 also supports version control and provenance management for data.

Because of its flexible user-defined metadata layer (HDF5 attributes), many scientific communities and industry consortia have chosen HDF5 to implement domain-specific conventions to record scientific provenance and to enable frictionless data sharing.

In the foreseeable future, we intend to provide the following:

1. Reference environments that guarantee the creation of byte identical executables, including plugins
2. HDF/A - a version of HDF5 specialized for archival and long-term data preservation. This includes a specification, tooling, and a set of best practices for preparing HDF5 files for accessibility in a future in which the original software may not be on hand.
3. Crash tolerance with configurable guarantees
4. Standard encryption plugins for both metadata and raw data
5. Support for common systems of physical units and user-defined units as part of a values type definition

Topic (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.

HDF5 software is tested daily by subjecting it to different test suites that are intended to discover regressions in functionality, performance, scalability, compatibility, portability, etc. The number of lines of test code is about equal to the number of lines of actual product code. We test about 200 different combinations of processor architectures, OSes, build systems, compilers, and runtimes. This is a very resource-intensive undertaking, especially when testing on HPC systems. To add to the complexity of testing, there are several community packages, such as `h5py` and netCDF-4, for which HDF5 is a critical dependency. Running the full versions of all test suites is very time-consuming, and, despite automation, the triaging and investigation of alerts still require human intervention.

Access to, maintenance, and upkeep of test systems are costly necessities. For a small organization, it is neither feasible, affordable, nor economical to maintain a large fleet of systems on-prem dedicated to testing. We primarily rely on GitHub and GitLab continuous integration (CI). In some cases, we have systems on loan from a customer, we have access to specific customer systems, we receive test results from our partners, or we gather results from community volunteers, etc. What may appear like a balanced load is a precarious state-of-affairs: There are limits to CI testing. Customers tend to reclaim their property, and testing of that subset of configurations stops for the entire community. The gathering of results from users is not perfect either, because it sometimes creates missing information, false positives and false negatives, and the follow-up and establishment of what happened is costly and not feasible in a timely manner.

The flow of testing on HPC systems is not unlike testing on non-HPC systems:

1. Automated testing needs to be set up and configured on the target system
2. Automated testing needs to be maintained, e.g., updates and upgrades
3. Test reports need to be triaged and analyzed
4. Issues discovered during testing need to be diagnosed and addressed in a timely manner

To follow this flow we need the following:

- Continuous, uninterrupted (by project changes) access and dedicated allocations on DOE systems for testing are a *sine qua non* for HDF5 development.
- A uniform test environment across systems, e.g., GitLab CI
- Access to compilers and diagnostics tools
- Access to training

Allocations on smaller systems are sufficient for daily testing. For scalability testing we need access to large systems, since certain issues only appear “at scale .” We can’t emphasize enough the high costs associated with not having uniform test environments on HPC systems. Finally, the competent use of specialized tools goes a long way to timely issue resolution, and access to training for our developers is a requirement.

What are the key capabilities provided by this infrastructure that enables it to meet your needs?

1. Access to different data stores and a deep storage hierarchy such as parallel file systems, node-local NVMe drives, and Burst Buffers
2. Access to accelerators such as GPUs for testing the performance of accelerator specific features, which cannot be done by using emulators
3. High-speed interconnects
4. Scale

What key capabilities, which are not already present, do you anticipate requiring within the foreseeable future?

- Intel DAOS high-performance storage
- Persistent Memory (PMEM)

Over what timeframe can you anticipate these requirements with high confidence?

In anticipation that our users will need them in 1-2 years, and HDF5 being a stack in constant technology transfer mode, we need the two items listed now.

What are the most-significant foreseeable risks associated with this infrastructure and what are your preferred mitigation strategies?

Our most-significant foreseeable risks are delays in the delivery of actual systems and teething problems with hardware and buggy software or SDKs of poor usability, slowing down or jeopardizing HDF5 development and stalling technology transfer. Our preferred mitigation strategies include early access to scaled-down resources, where possible the use of Cloud-based resources (e.g., PMEM), and continuous professional development through training and skill-building.

When responding to these questions, please describe the scope of the relevant research and development activities motivating the response.

A large portion of the HDF5 development effort is about steady technology transfer. The maintenance and adaptation of HDF5 software to address new requirements, for example,

new use cases and workflows or access to data in cloud-based or on-prem object stores, is a big part of the sustainability and evolution efforts. Our user and customer base is diverse and presents many different needs. As a result, our maintenance effort encompasses making HDF5 software work on many systems, performance tuning, and integration of new features developed by us or by the HDF5 community. The work products are delivered via regular maintenance releases, tutorials, and workshops. We also conduct internal research or collaborate with research institutions (e.g., FNAL, ANL, IIT) to prototype and perform feasibility studies for new HDF5 features required to address emerging computational and storage environments (e.g., Exascale computing, DAOS storage), or to address specific enhancements to HDF5 (e.g., sparse data storage). Perhaps our biggest effort goes toward plotting an evolutionary path for HDF5 software into an uncertain future while preserving and safeguarding the HDF5 data assets of millions of users, which they have accumulated over 25 years.

Topic (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?

For HDF5, we would estimate the effort as about 8 full-time equivalent (FTE) hours. The breakdown would be as follows:

Category	**FTE**
Maintenance, QA, support	4.0
Staff competency growth	1.2
Outreach and marketing	2
Ecosystem & standards	1

In HDF5's case, the additional effort needed is comparable to, if not larger than, that for development and maintenance in specific research projects. At a high level, the reasons can perhaps be summarized as follows:

1. HDF5 is a software stack in constant technology transfer mode.
2. HDF5 is not a research project and the software has been under development for over 25 years.
3. We feel responsible for disruption-free access to data stored in HDF5 to a global audience.

What tasks are the largest contributors to that additional effort?

After the adoption of HDF5 by the original sponsors, DOE and NASA, for specific projects and missions, adoption quickly spread in academia, government, and enterprise, throughout the world. (For example, between July and November of 2021, we counted downloads from 153,000 unique IP addresses from 191 countries in 9,273 cities.) There's considerable overlap between the needs of this global community and the needs of DOE projects, otherwise we wouldn't have seen this level of adoption. The additional effort begins where these needs diverge, and where methods suitable for project-level interaction and communication are no longer effective. Specifically:

1. There are many platforms and compilers that are or were not relevant to DOE projects, but that are important to the community at large. For example, the use of Windows is still pervasive in corporate America and around the world. Another example are less mainstream processor architectures. Before entering the HPC market, the ARM processor architecture was already common in the embedded and IoT spaces, and HDF5 was used on ARM-based systems almost 10 years ago. Ensuring portability, e.g., rather esoteric build systems, and doing the necessary Q/A is a costly task, apart from finding hardware for testing.
2. There are many features in HDF5 that are not particularly relevant to DOE projects, but important to users outside the DOE. For example, the support for Unicode in HDF5 is incomplete. Not in all places in HDF5 files where string values can occur do we support the full set of Unicode code points, and where full Unicode is available, only UTF-8 based encoding is supported. This is clearly a concern for an international audience, but also American multinationals.
3. Obviously, we cannot effectively communicate with the community at large as if it were a project involving a few dozen people. And we try to help users to help themselves as much as we can via our online documentation, the HDF forum, our website, and outreach activities such as tutorials, workshops, clinics, etc. The creation and maintenance of the documentation for a complex software is not a trivial task and requires technical knowledge and above average presentation skills. Ideally, HDF5 software could rely on at least one full-time person to steer the documentation effort and 5%-10% of HDF5 developers' time to assist with documentation tasks.
4. All development work on HDF5 takes place in the open, and under public scrutiny. We encourage and enjoy working with external community contributors, but this, at times, can be very disruptive and time consuming. External contributions may not satisfy requirements of software portability and compatibility, and that may have a knock-on effect on data stored in HDF5 at large; when removing defects, contributions may address a symptom but not the root cause thereby creating new technical debt, or the contribution may break existing features. Tools like GitHub are extremely useful for managing community contributions, but they cannot replace a competent person who oversees and manages the contribution process itself. HDF5 software maintenance needs one-full time person to manage community contributions and at least 5% from each HDF5 developer to contribute to code reviews and communications with the contributors.

In outlining the largest contributors to the additional effort, it would be short-sighted and one-sided to present them as beyond the scope and in no way beneficial to the DOE. Quite the opposite is the case! Community contributions make HDF5 more robust (e.g., reporting and removal of CVEs, reductions in compiler warnings, enhancements to the CMake and Autotools build environments, porting and testing on platforms not available to HDF5 developers), and they enable new features (e.g., dynamically loaded compression filters, optimized I/O access, computational datasets) and they stimulate new ideas, which, in the end, benefit also DOE users of HDF5. It's not just a one-way deal.

What are the largest non-monetary impediments to performing this additional work?

Not-for-profit companies come in all sizes, and with about 30 employees (Not all developers!) The HDF Group is perhaps on the small end of the spectrum. At that size, it is almost inevitable that we eventually lack the knowledge, skill, and agility to respond to a lot of community needs. Likewise, the pressure to align our capabilities with our biggest sponsors is enormous, and, in some cases, drives us away from serving the broader community better. Our biggest sponsors want us to implement software, so that's where our skills are. Serving the broader community requires specific communications and management skills, not to mention time, that our current focus doesn't allow the time or resources to foster.

There is also a demographic mismatch between the community at large and our talent pool. The broader community consists of researchers, engineers, domain experts, students, data scientists, and others who are not skilled software engineers and rightly just want to work with their data. While many HDF Group staffers are trained scientists and engineers, we rarely get a chance to take advantage of (let alone develop!) their non-software development-related skills.

How is any such additional effort currently funded?

Any additional efforts are currently funded through one or more of the following:

- Non-ASCR DOE, such as ECP or NNSA
- Specific projects funded by non-DOE government sources, such as NASA
- Profits, especially from projects in the commercial sector

In some cases, with the agreement of DOE program managers, we were able to perform specific maintenance activities by "shoehorning" them into existing projects.

How does that funding compare to a level of funding needed to maximize impact?

Using the breakdown provided earlier under this topic as a guide, the level of funding is less than 10% of what would have a sustained impact on the community at large.

Topic (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?

Compensation is always a factor, but we would be quick to point out that for most of our staff this is perhaps not an overriding concern. We don't offer stock options either.
To develop software for scientific and high-performance computing has an esoteric whiff, especially with middleware such as HDF5.
Champaign-Urbana, IL should be a fertile ground for outreach activities at all levels, but we have no funding or staff availability for outreach activities at local high schools, community colleges, or UIUC. Motivated graduates are looking to the "local competition" (Chicago's ANL & FNAL), and beyond (other DOE labs).
HDFGroup has a low profile and is not known outside a small professional community.
We have very limited resources for professional development.
We have no resources to hire top talent for outreach and marketing.

What successful strategies have you employed to help overcome these challenges?

Discounting serendipity as a strategy, we have had some success in articulating and projecting our values. People inside and outside the organization have told us that they share our values, that working for HDFGroup is to do something that has meaning beyond personal gain and something that contributes to the common good. At HDFGroup, there is very little that resembles an organizational hierarchy, and, because we are a small organization, perhaps a climate of greater collegiality.

We count ourselves fortunate to on occasion be grandfathered into professional development programs that some of our customers offer to their employees. While highly unpredictable, as a tactic, we are always looking for crumbs off bigger tables until the day when we might have the resources to adequately fund professional development and hire professional consulting services, where that makes sense.

What opportunities for professional recognition and career advancement exist for those engaged in developing scientific and high-performance computing software?

- R&D 100 Awards
- Best paper awards it workshops and conferences
- Participation in professional societies (e.g., IEEE, ACM)
- Grants from funding agencies, corporate sponsors, foundations and charitable trusts
- If we had the funding, we could establish a company "fellow" program.

Topic (6) Requirements, barriers, and challenges to technology transfer, and building communities around software projects, including forming consortia and other non-profit organizations

What are the important characteristics and components of sustainable models for software for scientific and high-performance computing?

To avoid inevitable pressure to put profit over mission, the HDF Group was established as a not-for-profit company. This decision has paid enormous dividends in terms of the ability to provide the best products and services for research communities, but it has been accompanied by a corresponding challenge of making ends meet. No investors lined up to infuse funds or expertise into the company. And as one whose brand emphasized free and open source software, the company found itself unable to sell products that produce surplus revenues that could support R&D and sustaining engineering.

We are aware of very few comparable non-profits that have had even our success. Nor are we aware of case studies where similar topics have been examined rigorously, and It would be valuable if the DOE could produce such studies.

As to what characteristics and components we did find to be important, here is a list:

1. Establishing and maintaining a strong brand with products, services, customers
2. A continuously growing user-, application-, and support base
3. Staff capability
4. Consistent staffing levels and staff continuity
5. Institutional support for R&D and sustaining engineering.

Perhaps The HDF Group has had the most success under 2. and 3., but clearly the five characteristics are not independent, at least in the long run. Item 5 is something we believe the DOE can address directly. For organizations that provide mission critical open source software, there should be a line item budget for R&D, for addressing technical debt, and for sustaining engineering. This budget should cover a minimum of five years, and should be renewable as long as the software remains important to the Labs.

What are key obstacles, impediments, or bottlenecks to the establishment and success of these models?

Being in constant technology transfer mode, perhaps the most critical element to the “survival” of the HDF5 ecosystem is, to borrow a term from systems theory, graceful extensibility. HDF5 is not perfect, and where it doesn’t perform at its best, that degradation in performance should be graceful rather than total failure. HDF5 continues to evolve through software extensibility of a slowly changing architecture, without the need to constantly return to first principles.

Perhaps the two biggest threats to graceful extensibility are premature extensions to the HDF5 software and funding streams that eventually deprive HDFGroup of the resources to maintain a posture of graceful degradation in organizational performance.

Frequently, a contribution of a “newly developed” HDF5 feature is a by-product of a research project or a feasibility study, and is of proof-of-concept rather than prototype quality that can be productized. The transition of an external contribution to a maintainable feature in HDF5, which can be released to the broader community, often requires substantial rework of the prototype. It is very rare that adequate funding is available for this work, the net effect being that a contribution is not integrated at all or “integrated” in a way that adds technical debt.

Under Topic (4) Developing and maintaining community software, we commented on the “black hole” in our books, the unfunded maintenance of free open-source software (FOSS) for the community at large. What appears to exacerbate our inability to fill this gap in a sustainable manner is how the awards of programs for which not-for-profit companies can or cannot apply are structured. The programs of which we are aware do not allow us to recover costs for outreach activities, or to fund community software maintenance. On the other hand, not-for-profits are barred from bidding directly for SBIR/STTR-type contracts, which allow for a profit margin. Perhaps there are programs to which not-for-profit companies can apply and that allow cost-reimbursement plus type contracts, but we have not yet found them.

That said, we would like to use this comment to convey the sentiment of our contracts office that among different funding agencies the DOE counterparts stand out through their competence, responsiveness, openness, flexibility, and general assistance.

What development practices and other factors tend to facilitate the successful establishment of these models?

In the case of HDF5 and based on what we have heard from our users, the following factors are key facilitators:

1. Values, clearly articulated and communicated
2. Availability of pre-packaged binaries (library and tools) for all major processor architectures and OSes, and integration with popular Linux distributions
3. Language bindings for most programming languages and relevant third-party tools (MATLAB, IDL, Mathematica, etc.)
4. A painless build process (from source) and support for cross-compilation
5. A helpdesk
6. Documentation, documentation, documentation
7. Outreach activities via the HDF forum, tutorials, annual workshops, conference presentations, webinars, etc.

The relative importance of these factors, of course, varies with the audience. If anything stands out, it's perhaps a responsive and competent helpdesk, whose importance cannot be overstated, at least for a product of the reach and complexity such as HDF5.

Topic (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?

For individual projects, we believe the three most relevant dimensions are project performance, software adoption, and community or industry penetration. Assuming that the software is hosted in a publicly accessible source control management system, there are several metrics a combination of which might form the basis for an overall assessment. For example:

- The number of contributors outside the team/organization
- The number of dependent projects
- The number of citations in the literature
- The average number of pull requests
- The average time a PR is open
- The average number of issues of a certain severity and the average time it takes on to address them
- The number of downloads of releases
- Sentiment analysis on community forums
- Staff retention and revenue (where there is a public record) over time
- Integration into (commercial) third-party tools
- Industry penetration, for example, measured via standards integration