LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-TR-816216

PSIP For HDF5Pilot Project Final Report

M. C. Miller, E. Pourmal, E. Gonsiorowski

October 30, 2020

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

PSIP For HDF5
Pilot Project Final Report

Mark C Miller (miller86@llnl.gov, LLNL),
Elena Pourmal (epourmal@hdfgroup.org, THG),
Elsa Gonsiorowski (gonsiorowski1@llnl.gov, LLNL)

Executive Summary

- A $150K contract was started with The HDF Group (THG) in late April 2020 to carry out a set of TBD software process improvements.
- The HDF5 project was selected as a test case for PSIP because it is a foundational library in the HPC/CSE software stack, is a mature project (20+ years) and it has been a recent goal of NNSA labs to foster quality improvements in HDF5.
- Three software process improvement goals (IGs) were selected: 1) Improve Documentation workflow and hosting, 2) Migrate the code base to GitHub, 3) Adopt and employ unified coding standards.
- Progress tracking cards (PTCs) were created for each IG and steps therein were executed late-May through end of contract, September 30, 2020.
- Progress on each IG exceeded our expectations. For example, the coding standards goal had been attempted multiple times previously, but success had eluded this team for many years.
- Some of the more positive observations THG staff had to say were…
    - "PSIP helped us come to consensus on coding standards by defining small steps within which we could then work as a team to find common ground. Maybe we would arrive to the same state with a different methodology, but the relaxing nature of PSIP process definitely made it easier."
    - "The completion of IG1 represents a true *culture shift* internally at THG to treat docs-like-code."
- Three sub-teams of between 7 and 14 THG staff members consumed a cumulative of only ½ FTE of effort. Subsequent productivity boosts from improved processes are likely to recapture this cost in the first year, if not sooner.
- THG staff and project leadership anticipate continued use of PSIP in other contexts, particularly as a means of managing GitHub community contributions to the HDF5 code base.

Introduction

Productivity and Sustainability Improvement Planning (PSIP) is a lightweight, incremental and iterative approach (much in the same spirit as Agile methodologies) for making routine software process improvements in software projects. It is designed to be easily applied in existing development workflows. Quoting from a November 2019 workshop paper describing PSIP…

> *PSIP breaks from classic software process improvement approaches such as CMM(I), SPICE, ISO 9000 or Six Sigma, in that it trades comprehensive standards and certification-driven assessment for self-defined, internally driven goals. It does, however, carry forward such ideas as having staged models of improvement (like CMM(I)) in the form of progress tracking cards. Additionally, PSIP is more aligned with lean and agile methods; it adopts their emphasis on iterative improvement and continuous learning. At its core, PSIP is an instantiation of the Plan-Do-Check-Act management cycle (PDCA, also known as Plan-Do-Study-Adjust) which provides the foundation for much of the modern software process improvement literature. This situates PSIP within a constellation of bottom-up, inductive software process improvement methods.*

PSIP is designed around the notion that already overburdened teams can define and carry out a series of small, incremental steps of progression towards improvement goals without significant (there will be some, but the goal is to avoid *significant*) disruption to ongoing development activities. A key enabling tool in PSIP is the use of Progress Tracking Cards (PTCs) which define the steps of progression towards a given improvement goal. This is the theory of PSIP.

The PSIP for HDF5 project was aimed at putting PSIP into practice with the purpose of evaluating its effectiveness in planning and facilitating quality and process improvements in a scientific software project as well as its associated artifacts. The HDF5 project was chosen as a use case to evaluate PSIP for several reasons. First, NNSA labs and LLNL in particular have a keen interest in how HDF5 quality impacts its uptake and sustainability as a community adopted and supported code. Next, HDF5 is a foundational library, a key substrate in the HPC/CSE software stack, and any improvements there realized through this contract will have benefits to many DOE applications depending on it. HDF5 also represents an older, legacy code with technical debts to pay down. These characteristics are similar to many NNSA and even some ECP code projects. But, because HDF5 is an I/O library, it represents a simpler use case within which to study PSIP than a full-fledged and significantly more complex PDE simulation code. We believe these attributes make HDF5 an ideal use case for evaluating PSIP.

Timeline of work

Towards this end, in mid-April of 2020, a $150K contract was established with The HDF Group, the organization responsible for developing and maintaining HDF5, to employ PSIP for a small set of software process improvement goals. We adopted the following timeline of work…

Who	What	When
PSIP Team	Kickoff training/meeting	April 22, 2020
THG	List of candidate improvement goals	May 4, 2020
LLNL (Elsa/Mark) + THG	Downselect goals list	May 6, 2020
THG	Develop PTCs	May 15, 2020
THG	Schedule/Resource Plan for PTCs	May 22, 2020
THG + LLNL	Execute Plan/PTCs + Updates	Bi-weekly through Sep 15th
THG	Draft Report	Sept 30, 2020

Kickoff Meeting/Training (late April 2020)

IDEAS-ECP team members leading the PSIP effort, Elsa Gonsiorowski and Reed Milewicz, hosted a two-hour training in which the PSIP process and PTC tools were introduced. HDF5 project manager, Elena Pourmal, and eight THG staff attended. The goals were to introduce THG staff to what PSIP is and how to think about applying it to effect software process improvements.

In these initial stages, getting the right "thinking cap" on with respect to what PSIP is and how to go about applying it is critical in helping the whole process be successful. Elsa, Reed and other members of the PSIP team do a phenomenal job in this introductory training. A key and important theme that comes through loud and clear is that the process is intended to be non-judgmental, objective and consensus-driven with input from all stakeholders. This whole-team engagement in understanding the project's overall productivity "pain points" is helpful in enabling the team to establish a candidate set of improvement goals.

Candidate Improvement Goals and Down-selection (early-May 2020)

After the introductory PSIP training, THG spent some time brainstorming about possible improvement goals. In addition, to help prime the thought processes here, Mark and Elsa offered some suggestions for possible improvement goals for the team to consider as examples. This was primarily to help bootstrap initial thinking.

The THG team came up with an initial list of 13 improvement goals, some more fully and clearly defined than others. The initial list included such items as improving new developer on-boarding processes, unifying the use of test-driven development across the THG organization and adopting consistent coding standards.

With the initial set of candidate goals, PSIP experts (Elsa) met with THG to help refine and down-select a set of improvement goals based upon how well defined the goal was as well as the likelihood of achievement (at least in substantial portions – see below) within the scope of the contract. Three improvement goal categories were selected

1. Improvement Goal #1 (IG1): Update Processes for maintaining the API Reference Manual
2. Improvement Goal #2 (IG2): Migrate HDF5 source code to GitHub
3. Improvement Goal #3 (IG3): Adopt a set of Consistent Coding Standards

Once these improvement goals were identified, THG then worked with PSIP experts (again Elsa) to draft initial Project Tracking Cards (PTC) for each goal. However, the PTCs initially developed were significantly refined by THG staff in follow-on meetings. The PTCs THG developed (as well as their progress status as of the contract end date of September 30, 2020) are provided in Appendix A. THG was encouraged to be free in defining PTCs that included steps beyond those to be completed within the terms of the contract.

Developing quality PTCs is essential but also non-trivial and often requires facilitation from an objective third party or PSIP expert. The challenge is ensuring that each step in the progression has a good definition of done and also represents a value-added end state in and of itself. If activity on successive steps winds up being postponed, the steps completed to that point still yield value to the project and are easily adopted as part routine development workflow. The work to that point is not orphaned on some parallel branch of activity making the effort invested in the earlier steps wasted. This is a critically important aspect of PSIP and PTC formulation.

PTC Development and Refinement (late-May 2020)

For each of the improvement goals identified, THG developed and refined a Progress Tracking Card (PTC). THG began with the example PTCs from the existing PSIP catalog but quickly evolved these to fit their own needs. Relevant THG staff members participated in a few PTC development meetings to help refine the steps involved in progressing towards each improvement goal. However, there was still significant dialog and consensus building among team members even to reach agreement on what each improvement goal actually meant. For example, improving the reference manual (IG1) for one stakeholder can easily mean something entirely different for another stakeholder. Several of the initial staff meetings involved helping the team to come to consensus on what the improvement goals specifically meant.

Next, eliciting input from team members on PTCs went poorly early on primarily because this activity represented a substantial change in thinking-caps team members were accustomed to (and that work pressures permitted) wearing. Team members were more accustomed to working under the extreme time pressures of delivering on promised features which don't ordinarily permit thinking holistically about development processes or thinking about and/or negotiating changes in those processes with other team members for overall productivity improvements. It took a while for THG staff members to put on this new thinking cap.

Impact of Extra Funding Specifically for Process Improvements

The fact that team members were, by virtue of this contract, obligated to deliver on whatever process improvements team consensus determined as well as the free energy to think about, plan and execute such process improvements the extra funding enabled was certainly an important motivator and factor in getting staff members to look at their work activities with new, overall productivity-oriented, eyes. Nonetheless, it would be a mistake to conclude from this that the additional funding alone is the sole reason or even majority reason for any progress and successes attributable to PSIP. All stakeholders in this work agree that while additional funding certainly did help, it helped primarily from the perspective of creating the expectation/requirement to *deliver on something*, in this case process improvements. For this team, being asked to improve their processes as a deliverable for a customer was not something

with which they had much experience, especially when the particular improvements and how to go about enacting them were to be determined primarily by the team and not the customer.

As THG staff got more comfortable developing PTCs, a sort of two-tiered approach to each PTC emerged where each major progress step was itself decomposed into its own PTC with a few to several sub-steps.

PSIP Execution and Progress Updates (bi-weekly through mid-September)

By the end of May, improvement goals were selected, PTCs for each were developed and resource allocation and plans for execution on PTC steps was created. The remainder of the contract involved execution of this plan and progressing through steps on each of the PTCs with periodic evaluation of progress.

Three sub-teams of between 7 and 14 THG staff members met regularly and ultimately consumed a cumulative ½ person year of effort (which included key decision-making and preparatory activities) making substantial progress on all three improvement goals (IGs).

PSIP experts met with Elena Pourmal, HDF5 software project leader and Neelam Bagha, Program Manager, every other week to check in on progress and more importantly to inquire about how the PSIP process itself was helping to bring about the process improvement goals. Below are highlights from these discussions.

- Getting a development team that is frequently overburdened with development work to take a step back out of their work on their deliverables to consider process improvements is difficult. This is a challenging trap teams can easily fall into and then winds up obscuring any thought/effort on how processes could be changed to improve productivity. Any tips/suggested activities that could be included in PSIP training resources to help teams routinely put on this thinking cap would be a good idea.
- Because PSIP is a bottom-up process, engaging team members to help identify, refine and drive improvement goals, it can be difficult to get team members who are either shy, used to working remotely/without much collaboration with others to contribute their opinions to the process. It would probably be beneficial to PSIP documentation resources to identify and/or suggest ice-breaker-like, team-based activities to help reduce these sorts of communication barriers among team members.
- Because PSIP is bottom up, enlisting input from the team members has the impact of creating shared ownership of goals. Team members are naturally more invested in the process and the outcomes.
- In the use case studied here, the $150K contract created ample “free energy” for the project to make room for the activities performed here. This “extra money” definitely had a positive impact on the ability for the team to engage in the improvement goals that were identified.
- Because the team was free to decide whatever improvement goals it chose, this also helped to create broader engagement and ownership. Instead of working towards a goal identified and funded by a particular customer, the team was able to work on items internally identified as important but nonetheless still on the hook to actually deliver on progress towards those goals.

- The fact that the team was on the hook to deliver something (whatever improvement goals it identified), it was not an option to not work on the activities identified here in favor of other customer's needs.
- Because PSIP is lightweight, there is no extra "baggage" hurdles in the form of learning new terminology or preparing planning documentation (e.g. CMM) to get over before work begins. This makes it conceptually easier to get into and start making progress. There isn't a lot of overhead to learn and start applying PSIP. A couple of team members emphasized this point. In other examples either with HDF5 or other projects, there was always much more overhead to get in place (estimates, WBS, planning documentation, other up-front tasks)
- It was important for each improvement goal to have a champion that helped to keep progress moving forward. For the RM, one team member was particularly passionate about this and naturally took ownership. For the other two improvement goals, the actions involved impacted more team members in more obvious and immediate ways and so there was more consensus building needed to help clarify the work and PTC steps. It took a while for champion(s) to emerge and in one case, those who started with the most interest in championing the effort were eventually overwhelmed by other priorities and unable to really serve as champions. Nonetheless, other team members picked up those reigns.
- Two broad motivations seemed to reveal themselves. One was to start doing a great, new thing. The other was to adjust process to help avoid having to do a really painful thing.
- The RM work is also resulting a key culture shift in the way THG developers manage reference manual and related (detailed) documentation and that is to treat documentation like code (which is part of a broader, modern approach known as "docs like code"). One of the biggest and most obvious advantages of this is that it's easier to keep documentation and code in sync.
- There are a surprising number of detail issues w.r.t to handling documentation for a large project like HDF5.
- The set of example PTCs in the PSIP resources was very important in helping THG to craft its PTCs here. It helped to THG to get the right "thinking cap" on with regard to developing these.
- The PSIP is more practical and hands-on than top-down approaches. This makes it more approachable for a team and easier to try to integrate into existing processes.
- Not being overly constrained by customer requirements - in terms of customer requirements and/or having to address pre-defined scope of work - and only focusing on the process to get work done was a huge plus. It eliminated pre-existing bias when defining work, so scoping of work was mostly a collaborative effort while keeping greater good in mind - eliminate obstacles to efficiency, biggest pain points etc.
- One of the nice qualities about the PSIP approach is that it tends to emphasize steps in progression to completion. The completion of each step offers a sense of accomplishment even if the ultimate goal remains unrealized. This is a more satisfying way to operate because it provides opportunities for positive feedback to a development team showing their steps in progress.

Details of Key Activity on IG1: Reference Manual

THG began IG1 with a lot of pre-existing, high quality documentation. This included an API reference manual for the HDF5 C Library interface, Fortran, C++ and Java APIs manuals, a User's Guide, a Developer's Guide, Release Notes, an HDF5 Formal File Format Specification, Advanced and Special Technical Topics papers and others. This collection of documentation served the needs of a wide variety of stakeholders both internal and external to THG.

In addition, the workflow(s) involved in composing, updating, cross-referencing, maintaining and deploying this collection of documentation (and doing so for multiple released versions of the HDF5 library) involved disparate technologies, processes and various productivity burdens. The PTC THG worked to develop was aimed at reducing productivity burdens for maintaining these and other documentation artifacts while also improving various functionality such as reducing effort to ensure API documentation stays in sync with source code changes and improving on-line search capabilities.

This required a review of their basic documentation requirements as well as available technologies and any necessary re-formatting of pre-existing documentation to match new technology requirements. It also included THG developer receptivity to and training needed for new technologies to be used. Early steps in the PTC were aimed at this preparatory work and included documenting stakeholder requirements for documentation, engaging their user community on their forum (forum.hdfgroup.org) and developing some throw-away prototypes.

One developer at THG had a particular interest in forwarding this IG and so became its *Champion* often helping to keep stakeholders engaged, completing various preparatory tasks and ensuring progress of additional PTC steps stayed on track. A key outcome of this PSIP work has been a culture shift within THG towards a docs-like-code approach to handling documentation.

Details of Key Activity on IG2: GitHub Migration

THG has been thinking about migrating to GitHub for a while mainly to achieve three things: 1) make HDF5 project more visible, 2) make community contributions easier, and 3) to take advantage of free CI resources to reduce burden on THG testing and DevOps staff. A PTC card was created with those goals in mind.

The migration to GitHub represents the fourth transition in revision control systems HDF software has undergone over the past 25+ years (including earlier history from HDF4). A move from Subversion to Git using our own, internal Bitbucket instance, occurred about 4 years ago. Because of this recent experience, the team had good understanding and agreement in what was required for the migration. In fact, THG reused some of our checklists created from that previous experience to cleanup branches in preparation for the migration to GitHub.

There are GitHub experts among the THG development team who answered questions and helped with the list of steps for the move. The PSIP team mostly focused on identifying the steps that we didn't need when we moved from Subversion to Bitbucket, for example, handling outstanding pull requests and syncing individual forks.

Because of prior experience and in-house expertise, there was no any difficulty in creating a PTC and achieving score of 3 on that PTC very early in the project. The majority of our activities

on IG3 was on identifying specific tasks to get from score of 3 to score of 4 and to coordinate with the C coding standard team. The tasks to get from score of 3 to 4 were also documented in PTC. Under the PSIP project the move to GitHub was completed (PTC score 5) in first week of October 2020 where we announce the move on FORUM and to the HDF User's Group Meeting participants.

Additional Training needed in Support of IG2

We will need to hold several coaching or best practices sessions on using Git and GitHub. While all the development team members know the Git basics, not everyone feels comfortable in syncing forks and branches. The team will also need to develop and adopt new process of tracking and discussing issues in GitHub, working with community members on code contributions and leveraging GitHub Wiki. The testing team will need to extend CI and optimize it for GitHub free testing resources.

PSIP methodology provided us with a quick and easy way to create a high-level plan and focus the PSIP team members on defining and executing critical steps for switching between Bitbucket and GitHub. Since the PSIP team members were very much in charge of identifying the tasks, and their roles and responsibilities, the whole process of switching between Bitbucket and GitHub was well coordinated.

Details of Key Activity on IG3: Coding Standards

Adopting coding standards was anticipated to be the most difficult improvement among the three areas. On a few occasions in the last 20 years, the adoption of uniform coding standards had been attempted without success primarily due to an inability to come to consensus on a variety of coding standards details. Past efforts had always started with a review of the multiple standards currently in use throughout the code and then an attempt to unify these and then document and agree to require all future code contributions to follow the unified standard. But, in these earlier attempts, no agreement was ever reached. Key developers had strong and differing opinions. In addition, many were skeptical about this new effort given past failures and on-going disagreements. In fact, key developers believed that this new effort would fail, and developers would never come to consensus. New approach to creating a standard was needed.

The first three things THG staff accomplished during the first few months of work on IG3 were:

- **Adoption of Generic PTC from PSIP Catalog**

  Initial disagreement was substantial enough that the team opted to adopt a generic PTC found in the PSIP catalog and then focus on what would be involved in moving from score 0 to score 1 on that PTC. The PTC was generically enough written that the development team couldn't find much to disagree about.

- **Agreement on why we need to use a standard**

  Next, key developers identified and reviewed portions of the HDF5 source code believed to be among the most difficult to write and maintain. From this exercise the next big step forward was that the team agreed on a key purpose of a coding standard; to help

make the code clear and easier to understand. The team agreed that whatever standard and formatting is chosen it cannot cause a loss in clarity and, in most cases, should improve code clarity. This became a major factor in defining the standard.

The team also acknowledged and agreed that different developers have different habits and preferences (for example, working in multiple windows on one or several monitors, different font sizes, preferences of different line lengths and spacing between lines, etc.) Any change in the standard may affect developers' productivity and the adopted coding standard should minimize any negative impacts on the developers' productivity.

- **Agreement on what the coding style meant**

  During discussions and team interviews the coding standard refined to include to the three very different notions

  i. C coding style

     Everyone agreed that C coding style is generic and not HDF5 specific, it can be used for any other software. The development team also agreed to consider already documented coding styles and consider automation of code formatting. Automation became another guiding factor in helping the team reach consensus on the chosen standard.

  ii. HDF5 coding standard

     An HDF5 specific coding standard includes source code organization, functions organization, etc., i.e., it is HDF5 specific and doesn't depend on C coding standard. The team didn't have any disagreement about importance of this part. The team also agreed that the main goal of documenting is to help bring new developers and community contributors up to speed.

  iii. HDF5 best practices

     The practices include the issues the HDF5 developers should keep in mind when developing and maintaining HDF5 software, for example, portability between OS (e.g., Linux vs. Windows), different compilers, performance, coding and debugging.

After THG agreed on the purpose of a coding standard and its guiding principles, outlining the set of steps to get to score 1 became a trivial task that everyone on the development team agreed on. They were also able to divide the team based on the members' interest to focus on the three subtasks: C coding style and its automation, HDF5 coding standard, and HDF5 best practices.

THG finished the HDF5 coding standard and outlined the best practices. Both documents will be checked into and become part of the source code and can be developed further as needed by THG developers and community contributors.

Automation Enables Flexible Approach to IG3

THG evaluated clang as an automated formatting tool to experiment with different styles (expressed via yml files) and to find the style that is the closest to that already in use in much of

the source code in the HDF5 library. Differences in standards used within the HDF5 library were presented to the whole team and everyone agreed that the majority of the code was very close to the clang style. Key differences could be addressed by modifying the corresponding yml file. The differences that couldn't be solved with the yml file, were resolved by applying the agreed principles – clarity and automation. We decide not to format some parts of the code to preserve clarity and made compromise on the issues (for example, line length) that can be automatically fixed and provide an author of the code with a tool to control the line length.

Several iterations and code reviews with the whole team were performed to make sure team's concerns were addressed. The GitHub migration (IG2) sub-team worked closely with the C Coding style sub-team (IG3) to automate conformance to the coding standard upon commits using GitHub Actions. A code formatter tool is run, and errors are reported when PR is created. The code formatter tool is now part of the source code and can be run manually to check for errors before creating PR. Taking this approach, the team was able to nimbly side-step many stylistic fiefdoms that often crop up during such efforts and that would have otherwise prevented progress. However, deciding to adopt this approach required significant effort in the evaluation of existing automated formatting tools as well as the integration of those tools into existing development workflows.

The code in the development branch was reformatted before the migration to GitHub. A detailed transition plan was created for GitHub migration to coordinate the effort in IG2 and IG3. By choosing an automation approach, the team eliminated the need for steps to reach scores 2,3, and 4 in the PTC. Reaching score 1 for IG3 will be easier after feedback is collected on automated formatting via GitHub Actions from the developers and members of the community and address deficiencies found during the testing period (between moving to GitHub and announcing to FORUM about GitHub move). No future special training on the standard is expected to be needed due to the use of automation.

PSIP was very efficient for this improvement since the team was forced to choose a different path that allowed us to identify and focus on small steps everyone agreed on and think about principles that can be used to reach a compromise.

Outcomes and Overall Assessment

This PSIP pilot has been a phenomenal success! Substantial progress was made on all three improvement goals identified at the beginning of this work. The progress achieved exceeded our expectations. In one example, the coding standards goal (IG3) had been attempted multiple times previously but success had eluded this team for many years.

Three sub-teams of between 7 and 14 THG staff members consumed a cumulative of only ½ FTE of effort to effect critical process improvements. Subsequent productivity boosts due to improved processes are likely to recapture this cost in the first year, if not sooner. Project leadership and frontline developers at THG anticipate continuing to use PSIP for future process improvements.

Some of the more positive things THG staff had to say were…

- "It was an enjoyable project and we did good things. There was a lot of enthusiasm and good energy."
- "PSIP helped us come to consensus on coding standards by defining small steps within which we could then work as a team to find common ground. Maybe we would arrive to the same state with a different methodology, but the relaxing nature of PSIP process definitely made it easier."
- "The PSIP kick-off meeting was very useful. It helped explain the process very well. The PSIP website was informative and example PTC cards helped us quite a bit."
- "The completion of IG1 represents a true *culture shift* internally in THG regarding documentation."

Some of the challenges THG staff identified were…

- Avoid having team members work in isolation for extended periods without opportunities for others to provide feedback on directions being taken.
- Eliciting feedback from developers outside of routine PTC progress meetings was difficult due primarily to availability of developer's attention.
- As new processes are adopted by key stakeholders, eventually the whole staff needs to be re-trained on new processes. Finding time in everyone's schedule and suitable training resources isn't necessarily easy nor are these costs immediately apparent when embarking on PSIP.

Comparison with Other Methodologies

An important bit of experience THG brings to the question of evaluating PSIP effectiveness is its 20+ years history in developing HDF5. Over the years, this team has used a variety of software process improvement methodologies and has the ability to offer some comparative assessment of PSIP relative to these other approaches.

PSIP was found to be significantly more light weight than other approaches such as CMMI or PMBOK. But that doesn't necessarily imply that PSIP is only for lightweight tasks. PSIP avoids a lot of front loaded, preparatory work such as cost estimating, WBS, project planning. In those other methodologies, a lot of that preparatory work is done by PMs or PIs and the team is just

given their assignments. In PSIP, all stakeholders are engaged from the beginning and the whole process of planning an execution feels more democratized.

PSIP also relieves team members of the baggage of having to learn a lot of special verbiage or terminology to start applying PSIP. Top-down process improvement models such as CMM(I) can often begin with too much abstraction making them disconnected from the sometimes unique, concrete software development tasks any specific project demands. This gives people opportunities to hide behind (or become overly enamored with) these abstractions. The simplicity of PSIP is that it gives people an opportunity to propose and advocate for specific, concrete process improvements. PSIP is more hands-on and practical and less abstract than these other methodologies. There are more opportunities to demonstrate progress and engender a feeling of having completed various steps.

In PSIP, the vision for improvement goals and implementation strategy comes from the team. The team members are not assigned by management. They are involved because they are passionate and interested in the improvement. Unfortunately, for THG it is rare when someone on the team has the opportunity to work on a task s/he is passionate about (except that THG staff is passionate about its software in general). Team members are also motivated to learn new technologies and/or share their knowledge. Better team chemistry and members engagement in process improvement goals was observed during the use of PSIP.

If THG wants to get contributions from the community, the PSIP approach may be very useful. PTCs should be created with community participation with each step divided into smaller achievable steps. Different members of community can join THG at different times contributing self-contained improvements toward a given process improvement goal.

Recommendations and Feedback to PSIP Team

1. Overburdened software developers tend to focus on project milestones and have little time to stop, take a breath, step back and consider the processes followed to reach those milestones. Any suggested activities and tools to help teams routinely put on this new and different thinking cap would be extremely useful to include in PSIP training resources.
2. With limited resources available for enacting process improvements, it is important for teams to understand productivity hurdles sufficiently to prioritize them relative to overall project goals and milestones. Teams should naturally focus process improvement effort on those productivity hurdles expected to have the biggest impact. On the other hand, while this is a nice way to *optimize* the use of resources, when using PSIP for the first time, it may be more important that a team just reach agreement on *any* useful process improvements to tackle regardless of priority.
3. Initial communication flow in PSIP is bottom up, from those in the trenches, doing the development work and most closely aware of where process improvements would be most beneficial laterally to each other and upwardly to project leadership. However, even in the most mature teams, that communication doesn't necessarily happen easily, naturally or automatically. Important information can be missed. It is important to create the safe space for all relevant stakeholders to contribute to selecting and defining improvement goals. In particular, a top-down approach where project leadership alone selects and defines process

improvement goals should be avoided. Aside from possibly failing to address key productivity hurdles, it will likely stifle team enthusiasm and ownership for outcomes.

4. It would be beneficial for PSIP training resources to include suggested ice-breaker-like, activities and tools to help reduce of communication barriers among stakeholders, both laterally among team peers and upwardly to leadership.
5. In the process improvements enacted here in this pilot project, it was important for each improvement goal to have a *champion*, someone with a passion for the activity and that will help keep progress moving forward. This can be a great opportunity for team members who have not previously had any leadership role.
6. Although artifacts from some process improvements may become visible to external stakeholders (e.g. changes in how on-line documentation is hosted), for the most part process improvements are visible only internally to the team. Unlike feature enhancements or bug fixes, there isn't necessarily any external stakeholder waiting on the completion of some process improvement as a deliverable. Treating process improvements as deliverables can be an important incentive for teams that are a little too comfortable doing things the way they've always done them. In addition, finding ways to make progress improvements visible to, even expected deliverables by, external stakeholders can also be a useful way to help galvanize a team.
7. In the use case studied here, a $150K contract created ample "free energy" for the team to make room for the activities performed here. This "extra money" definitely had a positive impact on the ability for the team to engage in the improvement goals that were identified. Nonetheless, it would be a mistake to conclude from this that the additional funding is required for PSIP to be successful. While additional funding certainly did help, it helped primarily from the perspective of creating the expectation to deliver on process improvements.
8. In projects with long term plans and goals, it can be difficult for teams to feel a sense of accomplishment and progress. A great selling point of PSIP is that it provides opportunities for immediate gratification. A PTC has a well-defined beginning, middle and end. The completion of each PTC step offers a sense of accomplishment even if the ultimate goal remains unrealized. This is a more satisfying way to operate because it provides opportunities for positive and relatively more immediate feedback.

Acknowledgements

We would like to thank Elena Pourmal and The HDF Group (THG) staff for including us in internal software development activities not ordinarily observed by "outsiders." We appreciate the trust THG has put in us and part of that trust involves ensuring that this report has been fully reviewed and approved by THG. We would also like to thank Elsa Gonsiorowski and the PSIP team for supporting this effort both with initial training and with ongoing guidance throughout the project. Finally, we would like to thank the LLNL, WSC Research Coordination Council (WRCC) for providing the funding for this study.

Appendix A: Project Tracking Cards (PTCs)

A1: Reference Manual PTC

Status of Reference Manual (RM) PTC as of Sep 30, 2020 is shown below. Strike out items have been completed. Each of the completed steps has involved a significant amount of development work. A large portion (~70%) of the old RM has already been reformatted for the new workflow. THG anticipates the new workflow to be fully in place by January 2021 and releases of code thereafter to be using it.

End User Documentation - HDF5 Library Reference Manual (RM)

Created by Gerd Heber, last modified by Elena Pourmal on Sep 28, 2020

Target

Create end-user documentation for the HDF5 library

User Story

As an HDF5 stakeholder, I want an easy-to-use reference manual, so that I can have access to accurate information about the HDF5 library.

Card

Items in blue are in works now.

Score	Description
0	~~Multiple RM versions based on different technologies exist. Stakeholders and requirements are not documented and partially unknown.~~
1	~~Stakeholder requirements are identified~~ and documented. The requirements list is prioritized.
2	~~Potential technical solutions have been researched~~ and the pros & cons of each have been evaluated.
3	~~A technical solution has been selected and an outline of the end product vetted by stakeholders.~~
4	Existing artifacts have been converted to the new system.
5	The process for updating the API and the accompanying RM entries is documented and followed.
6	An RM maintenance plan is created.
7	The process for reviewing and improving RM entries is documented and incorporated into the product life cycle.

Comments

- The conversion of existing artifacts will take place incrementally, e.g., 30% / 60% / 100%.
- Opportunities for conversion automation will be explored.
- Opportunities for automation of RM updates in response to API changes will be explored.

A2: GitHub Migration PTC

Prior to the GitHub migration, which was completed the first week of October 2020, HDF5 source code was hosted on a THG-managed Bitbucket instance. Although Bitbucket is a git-oriented code hosting system, because THG was running their own internally managed Bitbucket instance, wide-spread community contributions were effectively stifled.

Status of GitHub Migration PTC as of September 30, 2020 is shown below. There is a slew of details in attaining step 4 in this PTC that are not shown here. For historical development reasons, the team determined that resolution of portions of the Coding Standards goal should be decided/completed before the GitHub migration because of the potential opportunities it offers in reducing future feature branch merging complexity.

The HDF5 library has been available on GitHub now for just a few weeks and there have already been several PRs submitted (and accepted) from community contributors.

Adopt A Git Workflow that Supports CI and Encourages Community Contributions

Created by Gerd Heber, last modified by Elena Pourmal on Sep 28, 2020

Target

Transition from a collaborative Git workflow that supports only non-CI testing to one that supports low-cost CI testing and is transparent to community contributors.

User Stories

As a developer or reviewer, I want to be certain that all code changes pass all merge-request tests.

As a community contributor, I want my contributions to be held to the same standards as the HDF Group's internal development.

Card

Score	Description
0	~~HDF5 source code is managed in Bitbucket.~~ ~~No CI - testing is decoupled from merge requests.~~ ~~Development and contribution processes are not documented.~~
1	~~Evaluation of GitHub as a replacement is complete.~~
2	~~The migration plan is complete and approved.~~
3	GitHub account setup and staff training are complete. (Tasks in blue are in progress) › Tasks to Score 4
4	Repository migration from Bitbucket to GitHub is complete.
5	The move has been announced to the community and the HDF5 repository is open for contributions.
6	Library developer documentation is in place; the GitHub Wiki and issues database are active.
7	Improvements to the development and contribution processes are identified and implemented.
8	The maintenance process for the GitHub repository is in place.

Comments

- CI - Continuous Integration
- In addition to its local Bitbucket server, The HDF Group already has an organizational GitHub account.

A3: Adopt Consistent Coding Standards PTC

The status of the Coding Standards PTC as of September 30, 2020 is shown below. All of the progress is wrapped up in the initial step, step 0. Just attaining group consensus on the coding standard was an important step, not previously realizable in the 20+ year long history of this project. Instead of requiring all developers to code to a strict, specific standard, the approach the team has adopted is to agree upon the coding standard for *committed* code and have automated formatting tools perform modest conversions between minor variations in styles employed by different developers. This enables developers to code in ways they are accustomed while ensuring the code base also remains in a consistently good state following the adopted standards.

Adopting a Consistent Coding Standard for the HDF5 Library

Created by Gerd Heber, last modified by Elena Pourmal on Sep 28, 2020

Target

Steadily convert the codebase over to an agreed-upon standard.

User Stories

As a person responsible for software quality and correctness for the HDF5 library, I want guidance on selecting and implementing coding standards so that we can make our code easy for everyone to read and understand.

As an HDF5 library developer or community contributor, I want tool support so that I am complying with the standards with minimal additional effort or ambiguity.

Card

Score	Description
0	No coding standard adopted. Tasks to Score 1 ☑ Select representative source files ☑ Create outlines for **Best practices** and **HDF5 things** ☑ Review different standards and use clang-format tool the reformat selected files ☑ Present finding to HDF developers and select HDF5 C coding standard 24 Jul 2020 ☐ Review feedback and modify yml files used by formatter. ☐ Document deviations from the standard and create HDF5 style sheet ☐ Publish HDF5 C coding style ☑ Enable automatic checks (vis Actions) on PRs and check-ins in Github ☐ Complete Best Practices document ☐ Complete HDF5 things document ☐ Publish Best Practices document ☐ Publish HDF5 things document
1	The team has selected and documented an agreed-upon standard.
2	New code that is written is required to comply with the standard, and the team has conducted a feedback session to assess and revise the standard.
3	The team has developed and put into place a refactoring plan to bring pre-existing code into compliance with the standard.
4	Tool support has been put in place to help ensure compliance, and running the tool is made part of contribu Screenshot