The government seeks individual input; attendees/participants may provide individual advice only.

Middleware and Grid Interagency Coordination (MAGIC) Meeting Minutes[1]
December 1, 2021, 12-2 pm ET

Virtual

Participants

Alan Sill (TTU)	Larry Kaplan (HPE)
Alison Derbenwick Miller (Oracle)	Lavanya Ramakrishnan (LBL)
Bill Miller (DOE)	Mallory Hinks (NCO)
Cheryl Martin (NVIDIA)	Mike Heroux (Sandia)
Christian Trott (SNL)	Misha Ahmadian (TTU)
David Bernholdt (ORNL)	Rich Carlson (DOE/SC)
David Martin (Northwestern)	Saswata Hier-Majumder (DOE)
Dhruv Chakravorty (Texas A&M)	Seung-Jong Park (NSF/OAC)
Glenn Lockwood (LBNL)	Sharon Broude Geva (U of Michigan)
Graham	Stefan Robila (NSF)
H Birali Runesha (U of Chicago)	Susan Gregurick (NIH)
Jack Wells (NVIDIA)	Terrill Frantz (Harrisburg University)
Jeff Larkin (NVIDIA)	Tevfik Kosar (NSF)
John Josephakis (NVIDIA)	Thomas Brown (TTU)
Joyce Lee (NSF)	Todd Gamblin (LLNL)
Katherine Austin (TTU)	Todd Shechter (Wisconsin)
Keith Beattie (LBNL)	Tom Gibbs (NVIDIA)
Kristin Davis (NTIA)	Tom Gulbransen (NSF)

Introductions: This meeting was chaired by Rich Carlson (DOE/SC)

Sustainable Software Speaker Series: Long Term Sustainable Software Issues – Academic and Industrial Software

Long Term Plans for Spack

Todd Gamblin, Advanced Technology Office, Livermore Computing, LLNL

- Gave an overview of Spack
  - Supercomputing PACKage manager
  - Manages scientific software ecosystem
    - With flexibility needed to build packages for diverse HPC machines

[1] Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Networking and Information Technology Research and Development Program.

    - Language-agnostic
        - Focused originally on build from source
        - Now focused on both source and binary
    - Has become a de-facto standard for packaging HPC software
- Todd said that the HPC software ecosystem and the scientific software ecosystem in general is getting more complex.
    - More packages interconnected
    - Spack helps get the whole thing set up and installed with one command on your supercomputer based on package information that people maintaining each of the different components has put into Spack.
- Spack environments enable users to build customized stack from an abstract description
- Spack's concretizer leverages ASP solvers to turn abstract constraints into a fully specified, buildable graph
- Spack's model lowers the maintenance burden of optimized software stacks
- Sustainability challenges
    - Community
    - Infrastructure
    - Deep Technical Challenges
    - Maintenance
- Many Contributors and maintainers
    - 6000+ software packages
    - 960+ contributors
    - 6 core developers
    - Extended core tea at Kitware, TechX
    - 30 trusted package maintainers on GitHub
    - 150 "package maintainers" (so far)
- Long-term strategy
    - Not sustainable without community
    - Continue to prioritize feature that get them external buy-in
    - Wide adoption in HPC gets industry attention
    - Portability and generality will become increasingly important as cloud environments diversify architectures
- Spack is critical for supporting ECP's E4S stack, which they hope will be sustained after ECP
    - Will be used to build software for 3 upcoming US exascale systems
- Project alongside Spack: BUILD – 3-year strategic initiative, aimed at reducing human maintenance burden
- Key Spack Priorities for Future Sustainability
    - Preserve Spack core team and feature development after ECP
    - Increase build automation to match rate of contribution
    - Generalize Spack's model to make the software stack as portable as possible
    - Continue to grow collaborator base with key features

Q&A

- Christian Trott: Do you have maintainers from outside Livermore right now?
    - Todd said that they have the core team at Livermore, which has the final say, but they do have 30 people who are able to actually merge PRs. If someone is interested in a small chunk of the software stack, or if they want to maintain some testing or aspect of the tool, they let them.
- Alan Sill: Have you considered deploying Spack-built applications via unit kernels to reduce the number of builds and operating system environments you might have to support?
    - Todd said no, but they have looked at the generic x86 targets, for example, to reduce the number of binaries that they would need to maintain in CI.
- Glenn Lockwood: Could you speak to how you avoid having individual key people in the project being "benevolent dictators"? Or if someone gets hit by a bus, how do you avoid the whole project grinding to a halt?
    - Todd said he didn't know because no one has been hit by a bus yet. If they were to lose some of their core developers, it would hurt, but there are probably 2 or 3 people that could run the project if something happened.
    - Glenn said that people leaving for industry is a big concern.
    - Todd said he thinks that's a reason you have to have room for growth on projects.
- Tevfik Kosar: You mentioned one of the priorities is to seek out sustainable funding. In addition to federal support and some vendors in the industry support, have you considered mechanisms like subscription or membership models or freemium type of services?
    - Todd said he has looked at ways that they could do that. Red hat is the closest one. They sell access to the optimized binaries or the optimized stack. Everything is open, but the actual binaries themselves are licensed and you have to buy a support contract to use them. He said he hasn't seen too many companies be successful at that other than red hat though.

Kokkos and Software Sustainability
Christian Trott, Center for Computing Research, Sandia National Lab

- How does Kokkos help applications with sustainability concerns?
    - Kokkos sits in the middle. You write your stuff to the Kokkos ecosystem and then the Kokkos ecosystem is mapping it to all of the other platforms underneath.
- What is Kokkos?
    - C++ programming model for performance portability
    - Expanding solution for common needs of modern science/engineering codes
    - Open Source
    - Many users at a wide range of institutions
- CG Solve: Performance 2016
    - Comparison with other programming models
    - Straightforward implementation of kernels
    - OpenMP 4.5 was immature at that point

    - 2 problem sizes
        - 100x100x100 elements
        - 200x200x200 elements
- CG Solve Performance Today
    - As described above
    - Also try replacing SPMV with TPL
    - Running 100x100x100 heat conduction problem
    - Measure effective bandwidth
    - Why is this beating vendor libs?
        - Complicated, but real
- Stepping stones to success
    - Address a critical need: More platforms were coming, and apps simply can't rewrite for every single one
    - Demonstrate viability: The Kokkos team ported chunks of LAMMPS and Trilinos to demonstrate that it works and performs. They also demonstrated performance parity to native models in miniApps
    - User outreach
    - Long-term sustainability: Having support from Sandia's HPC management ensure stable long-term funding
- Christian went over the Kokkos Timeline at Sandia
- Testing and Verification
    - GitHub based developments with Pull Requests
    - Pull request testing
        - 30 configurations
    - "Rule of three": 2 additional developers need to approve a pull request (meaning "I could maintain this if the author left")
    - Strong backwards compatibility policy
    - Releases require integration testing with major customers
- Kokkos Uptake
    - Community Project
        - 25 Developers – 15 for Core
        - Regular contributions from HPC vendors
    - Kokkos Slack Channel
        - >700 registered users at 90 institutions
    - Applications and Libraries
        - ~150-250 HPC projects using Kokkos
        - ~2 dozen apps run science and engineering production runs with Kokkos
- Training and Documentation
    - Kokkos Lectures
    - Extensive Wiki
    - Slack – direct support

Q&A

- Todd Gamblin: When you said backwards compatibility, do you mean API or API compatible?
    - Christian said API compatible
- Jack Wells: What are your thoughts about transitioning from Kokkos to the standard language parallelism? How should this happen?
    - Christian - My gut feeling is that part of it will never happen because they won't be HPC specific enough in C++. There is certain programming model semantics which we won't be able to get into the C++ that we still need to express. And there will be new paradigms we need to develop, and it takes a long time to get something into the C++ standard. We need places where we can develop these things faster, deploy them faster, and get the deployment experience you need in order to get something into the C++ standard. The C++ standard is not in the business of standardizing new ideas. We need a place to do that, but we are transitioning things into a standard and essentially replacing pieces of internals and Kokkos with the standard facility.
    - Jack: So, you'll be including these features in your testing and integration?
    - Christian: Yes
- Stefan Robila: Have you received significant feedback in terms of the value brought to the users in terms of how you plan to adapt them? I'm looking at the YouTube tracking and there is significant engagement in the first couple of sessions, and then the numbers keep decreasing. How do you make sure that the right information is sent to the user?
    - Christian: Part of what you're seeing is, I think, a very common thing. The main is that the feedback is really running largely about over the slack channel. That's where we encourage people to go and ask questions. Part of the reason is that the initial presentations are the ones you really need initially and then it takes a while before you need the more advanced things. We are really encouraging asking questions on the Slack channel. Most of the time users get answers to their questions within minutes.
- Todd Gamblin: Do you have a paid Slack instance yet? Or is your information still captive to Slack?
    - Christian: It's not paid. I've talked with people about it. My gut feeling right now is, if I were to pay for this, I would need about the equivalent of almost a postdoc in money per year.
    - Todd: We have the same problem. If we started a foundation or something, we could get like 85% off. But right at the moment, we can't search the history past a month because it goes by that fast.
    - Christian: We are at 18 or 20 days or so. The big problem here is obviously we can't just fund that out of a single institution who contributes to this, so how do we fund collaborative things?
- Todd asked if any of the program managers had thoughts about foundations for software within DOE.
    - Rich: It is not an area that we have discussed internally, probably should.

Software and Sustainability

Larry Kaplan, Senior Distinguished Technologist, Chief Software Architect for HPC

- Where are we heading? -> The Internet of Workflows
    - Exascale Capability
    - Workload Diversification
    - Compute Heterogeneity
    - Data Sovereignty
- Internet of Workflows
    - From Edge to Exascale
    - Democratization of Parallel Runtime Environments
    - Performance Portability
    - Secure Networking Foundation
    - Enabling Silicon Innovation
    - Worldwide Data Web
    - Hybrid Execution Pipelines
    - Open -You're in Control
- Software
- Future of HPC: Enabling Heterogeneity
    - New workloads that can be mapped to custom silicon show great potential
    - HPC delivery models will embrace federation and multi-tenancy
- Two Sustainability Challenges for Software
    - Standardization – many conflicting priorities from API creators
    - Productization – for software to be sustainable, it must be hardened and supported.

Q&A

- Christian Trott: Todd mentioned the problem with things like security and stuff like that and the question of where do we run CI. Is there a place for companies with pretty big HPC footprints like HP to provide CI resources for some of these more central community projects?
    - Larry: That is an area where we are starting to have discussions with customers and other vendors about how to do better CI, to get better previews of how integrations are going to pan out as we go to do new releases
- Rich Carlson: It was mentioned earlier today that one of the big issues was funding to maintain this sustainable platform. Do you believe that industry has the same kind of issues or are there different fundamental issues in sustaining the software that you have to write as a company?
    - Larry: They are a bit different, thought I think there is some relationship because of the tendrils that we're discussing in terms of whether we have dependence, whether HPE has dependencies on community software vs whether those dependencies are more at the customer or application level. From the point of view of sustainability within our vendor software, I think our big challenges are usually just the tail: How long is the tail? We know we need to support our customers. Supercomputers have about a 5-year lifetime. Servers may have a

similar lifetime. Ensuring that software stacks are supported for that lifetime is really the big challenge for us. The other thing that we tend to run into is that we would like to, and we actually need to keep these software stacks used for these systems up to date. Especially with respect to security vulnerability. But we also run into the tension of customers that don't want to have to update very often because of the challenges doing updates. And so, we're trying to address some of those challenges.

- Rich Carlson: Is staffing an issue?
    - Larry: Staffing can be an issue. It's a very competitive market out there. I think it's a little bit less of an issue for us than it is for the labs and academia, but I can't say that it's not an issue.
- Todd Gamblin: If the software price is included in the cost of the system, what's the point of not just making the system cost a little more and make the software open?
    - Larry: The way we've been looking at this is for software that isn't providing differentiation, and even in some places where it is, we are willing to open source. But in other areas, there are certainly places that we do want to maintain differentiation. That's where we look at the standardization of the APIs and whether that's done via public standards or some other kind of standards. Here we want to allow for some differentiation because there are places in our software where we believe we do provide that. So, the programming environment is a good example of that.
- Rich Carlson: There's a big difference between open source and open community. What are the implications for the sustainability of these software packages, runtime environments, applications, and an open community model vs an open-source model? Noted that in open source is where you can freely get the code, but you might say Globus is open source, but not open community.
    - Christian Trott: I see that difference quite clearly. We had that discussion reasonably early on in Kokkos. That discussion had to go through management quite a bit. Making open source was a reasonable no-brainer. But making an open community means that you in some sense have to give up part of that control or you must be willing to give up part of the control. Getting approval for that, that took quite a bit of time internally.
    - Todd Gamblin: We've tried to be open community on the Spack project. That wasn't as hard to get approved for an open-source project at Livermore, so that probably differs by lab. If we see people doing really big contributions, we try to work hard to take them in because we actually want people to invest in the tool and continue adding more features. I think it's very hard to sustain a community unless people feel at lease a little ownership over the tool. Donating IP from the lab is an interesting process. I've talked to our IP organization about that, and they haven't seen anything like that before.
    - Rich: Is this one of the keys to having long-term sustainability?
    - Todd: It seems to be for industry because I think there's a lot of skepticism about a single company owning projects and not having anti-competitive interests in it. To

some extent as a government lab, we're seen as more neutral than a typical industry player would be, so we're in a better initial position to have an open community. I think to get industry players to work together, it might be good to pursue something like that.
    - Larry: Getting vendors to all cooperate is sometimes challenging.
- Rich Carlson: Besides funding, what is your biggest challenge in sustaining your software for the long-term?
    - Todd Gamblin: I would say the HPC environment. And I think the facilities. I think if we're going to sustain an open-source software ecosystem in DOE, either the facilities or someone needs to provide cycles for build and test and to prioritize that because right now we kind of test in production and I think that's more costly.
    - Larry Kaplan: We're going to focus on the stuff that we depend upon from the community, but when we start looking at all the other software that's just dependent upon from the application or user level, it's very hard for us to understand how to pick and choose and prioritize.
    - Christian Trott: The long-term retention of staff. A lot of folks are just getting hired away out of the core HPC environment with salaries we just can't compete with.

Next Meeting

February 2 (12 pm ET)