LSSw Meeting 2

- Topic: Progress, impediments, priorities and gaps in leadership scientific software: A panel discussion
- Description: This month we have five panelists representing DOE software projects with a variety of relationships to leadership scientific software:

  - Ann Almgren, Berkeley Lab, PI of the AMReX project
  - Todd Gamblin, Lawrence Livermore National Lab, PI of the Spack project
  - Paul Kent, Oak Ridge National Lab, PI of the QMCPACK project
  - J. David Moulton, Los Alamos National Lab, PI of the IDEAS Watersheds project
  - Todd Munson, Argonne National Lab, PI of the PETSc/TAO project

Preview for LSSw Meeting 3: Nov 18, 2021

- Topic: Expanding the Leadership Scientific Software Developer and User Communities: A panel discussion
- Description: This month we have panelists representing the broader scientific software developer and user communities:
  - Panelists, TBD
- Prompts:
  - How has the traditional definition of leadership (HPC) scientific software developers and users limited who can be involved?
  - What is required to make the traditional definition more inclusive?
  - What do you see as the most urgent priority activities in planning for a holistic leadership software ecosystem over the next few years?
  - What is missing from the conversation about sustainable leadership scientific software?

Panelist Prompts

- In the past few years, how have we improved the development and sustainable delivery of leadership scientific software?
- What are the most important impediments to further improvement?
- What do you see as the most urgent priority activities in planning for a sustainable leadership software ecosystem over the next few years?
- What is missing from the conversation about sustainable leadership scientific software?

- Goal for the day: Identify and record the progress, impediments, priorities and gaps in leadership scientific software.

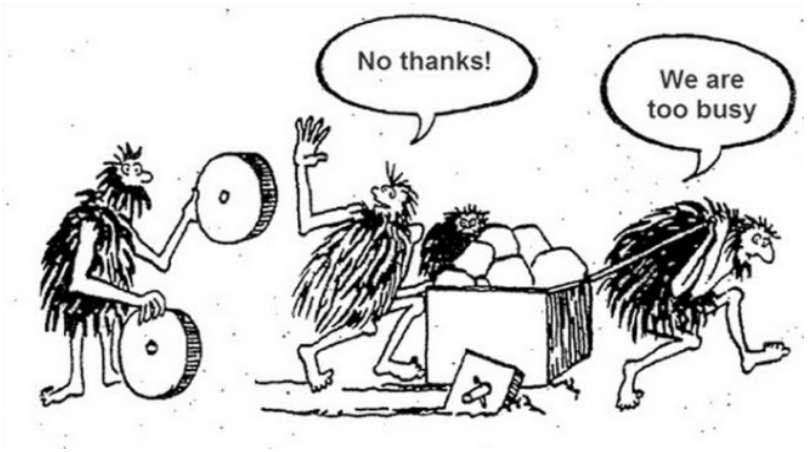
Ann Almgren

- Senior Scientist and Group Lead, LBNL
- Deputy Director, AMReX Co-Design Center (PI: Bell)
- ECP: member of AMReX, ExaSky, ExaStar, MFIX-Exa and WarpX teams, collaborating with ExaWind, Pele projects
- Member of SciDAC projects (FASTMath Institute and partnerships)
- Presenter at Math Libraries day at ATPESC
- Code development for years in
  - Fluid dynamics generally (compressible and incompressible / low Mach)
  - Structured grid / Adaptive mesh refinement
  - Multiphysics applications
  - One of the many Fortran → C++ converts

Philosophy

High-quality sustainable scientific software serves two fundamental purposes:

- Provides the building blocks for new capabilities and science applications so that forward progress does not require re-inventing the wheel



- Embodies expertise of specialists – not every science team can have expertise in every area

Philosophy (p2)

For sustainable core scientific software to be broadly successful, we need

not

In the past few years, how have we improved the development and sustainable delivery of leadership scientific software?

- The most impactful change has been the increased availability and adoption of continuous integration (CI) testing. Codes have gone from lacking any form of regular unit or regression testing to multi-platform testing before commits are merged, including the ability to trigger tests to run at the leadership class facilities. This

  - shrinks the “down time” (time when the code may be newly buggy) substantially
  - increases user trust
  - improves developer productivity

What are the most important impediments to further improvement?

- A large fraction of the impediments are sociological not technical. We need to
  - Maintain the workforce who can create and maintain quality software (this requires both $$ and status)
  - Work with the user community to make software as broadly available, usable and responsive as possible: code must be used to survive

What do you see as the most urgent priority activities in planning for a sustainable leadership software ecosystem over the next few years?

- Finding the balance between from-the-top mandates and community-driven “grass-roots” support
  - For example, ECP has “mandated” that AD projects use ST products. The SciDAC program is “mandating” that partnership projects make use of the Institutes. This has brought about some good changes but runs the risk of being too top-heavy, and is not viable across multiple funding sources.
- Finding the balance between software maintenance (especially on new architectures) and “research-y” tasks to add new capabilities and improve performance

What is missing from the conversation about sustainable leadership scientific software?

- We need to be sure to talk not just about how to fund the software development and maintenance, but also how to fund the engagement with (potential) users. A one-size-fits-all throw-it-over-the-fence model is not the answer.

- What do we do when currently free options (such as github itself) are no longer free? Or when sustainable software testing overwhelms the resources allocated for it?

Todd Gamblin

Senior Principal MTS
Livermore Computing Advanced Technology Office

- Leads:
  - Spack Project (ASC/ATDM)
  - Packaging Technologies Project (ECP)
    - Works closely w/E4S team, EPC facilities
  - BUILD Strategic Initiative (LLNL LDRD)
    - Binary analysis, modeling, and ML for automating software integration
- Past Involvement:
  - ECP Continuous Integration
  - IDEAS, IDEAS ECP
  - Lots of performance analysis work

In the past few years, how have we improved the development and sustainable delivery of leadership scientific software?

- DOE-wide collaboration has increased
  - Projects work together to ensure compatibility, synergy of components
- Mindset is oriented towards broad usage
  - Production model of ECP pushes projects to support more users and machines
- CI is used much more frequently
  - Largely as a result of Travis, GitHub, other free CI services
  - starting to appear at HPC sites
- Packaging. Massive increase in number of packages in Spack:
  - Oct 2018: 2,948
  - Oct 2021: 5,947 (2x in only 3 years!)

What are the most important impediments to further improvement?

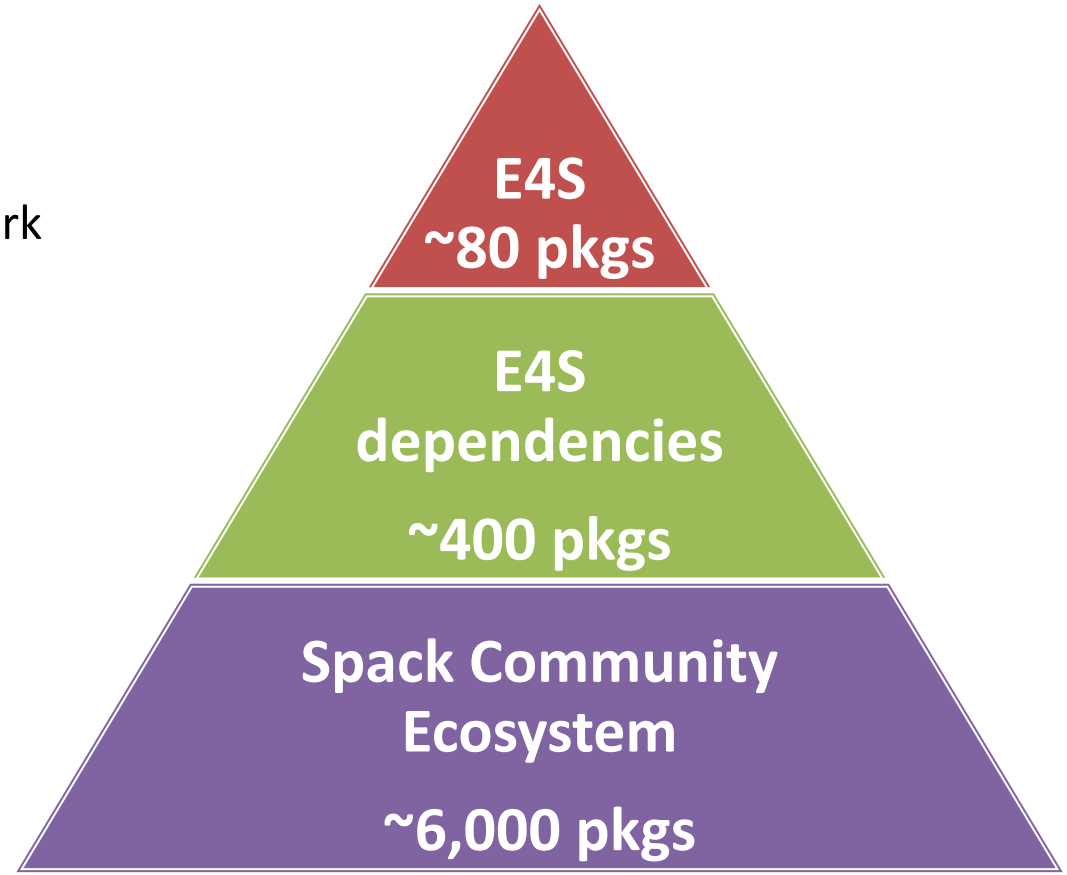
- Budget Uncertainty
  - Project teams are not as productive when there is a looming shortfall
- Funding to keep up with the rate of growth
  - Many successful projects started during ECP are growing rapidly
  - Unlike industry, growth != increased profits
  - Can't easily scale support effort w/popularity
- Facility security models
  - Automation is essential to scale projects
  - Facility security model does *not* allow testing of untrusted PRs (most OSS projects)
  - Can't test adequately on leadership SW/HW environment

What do you see as the most urgent priority activities in planning for a sustainable leadership software ecosystem over the next few years?

- Reduce uncertainty for critical projects
  - Risk of staff looking elsewhere if their projects are left out in the cold
  - Short-term funding is hard to hire on (e.g., PCRs)
- Ensure reliability of ECP products
  - Critical investment of ECP is resulting in adoption of tools
  - Users need confidence that the tools will be there after ECP
- Increase adoption by removing barriers for users and developers
  - Continue to make it easier to use ECP software
  - Solve security and automation problems at facilities
  - Make testing in a relevant facility environment as easy as testing in the cloud

What is missing from the conversation about sustainable leadership scientific software?

- Leadership software:
  - doesn't exist in a vacuum
  - is developed on non-leadership machines
  - relies on *lots* of "mundane" dependencies that also require work
- Alan Edelman on Julia (paraphrasing):
  - Scalability and performance are important, but
  - Adoption is also a *hugely* important measure of impact
- Without followers are we really leaders?
  Need outreach to potential contributors:
  - Cloud vendors –leadership stack on every HPC VM
  - ISVs – get them using and contributing to libraries
  - Regular academia / industry users



More to be gained from increasing the pool of contributors than from making every existing contributor marginally more efficient.

Paul Kent

Distinguished R&D Staff, Oak Ridge National Laboratory.

PI for QMCPACK Application Development project in ECP.

PI for Center for Predictive Simulation of Functional Materials.

Gordon Bell Prize winner. Fellow of the American Physical Society.

Selected Observations & Challenges

Based on redesigning and updating open source QMCPACK for performance portability while increasing performance (C++, ~300K SLOC, CPU+GPU).

1. Vital role of technical professionals and software engineering support

Essential to establish and maintain a solid foundation for long lived science codes, adjust to changing complex requirements, nurture best practices. Most important change in last few years.

2. Standards and specifications are insufficient

We need real world performance as well. Biggest impediment to our progress. Crucial to have real-world integrations and feedback mechanisms to the “software technology” developers. Implementations should closely track specifications cf. C++.

Selected Observations & Challenges

3. Integration and testing are key

e.g. Without continuous integration we would not have been able to reliably refactor large parts of QMCPACK. Testing is still a challenge, e.g. for multiple vendor GPUs. More discussion on development efficiency needed, best practices need to spread further.

4. Complexity is increasing, not decreasing

How could we more effectively drive and reward greater simplicity for the entire community of developers and users? E.g. Why are we dealing with C interfaces to libraries in a C++ world? How and when could we express our algorithms with 10x fewer lines of code?

David Moulton

- Scientist in Applied Mathematics at Los Alamos National Laboratory
  - PI of IDEAS-Watersheds (a derivative of the original IDEAS Productivity) focusing on developing a software ecosystem through use cases and partnerships.
- Experience related to leadership scientific software
  - (Long ago) led the refactoring of a collection of Dendy's logically structured robust variational multigrid codes (very legacy) into the MPI-based BoxMG library.
    - Passed onto Andrew Reisner, now known as Cedar.
  - Led the design and development of Amanzi, a multi-process flow and reactive transport simulator for Environmental Management
  - Initially co-led (now Ethan Coon leads) development of the Advanced Terrestrial Simulator, which is built on the low-level infrastructure provided by Amanzi
    - Designed Arcos multiphysics framework within Amanzi for the ATS
    - Amanzi-ATS (https://amanzi.github.io/) won R&D 100 Award in 2020.

Prompts

- We are better able to support best practices with cloud services
  - Open Development, automated testing, and CI on GitHub (or similar)
  - Docker/Docker Hub to help with CI and deployment
- Investment from sponsors in continuing on this path
  - Direct support for code releases, documentation, testing, tutorials
- Demonstrate benefits of the software ecosystem for scientific productivity
  - Well documented interfaces and interoperability are needed to realize a software ecosystem that is more than a collection of things that happen to be built together.
- Address the fragmented funding that supports most codes
  - Need a well supported code maintainer for each code (a champion of best practices, standards, releases, feature integration, etc., and hence, sustainability)
  - This role is not dependent on scientific productivity of the developers or users

Todd Munson

- Senior Computational Scientist at Argonne National Laboratory
- Experience related to leadership scientific software
  - Developed linear and nonlinear solvers in the Portable Extensible Toolkit for Scientific computing (PETSc)
  - Developed methods in the Toolkit for Advanced Optimization (TAO)
  - Software architect for TAO
  - Lead the PETSc/TAO ECP

Software Sustainability

Software sustainability is the capacity to persist and flourish as the active community grows and evolves.

Maintenance is the capacity to keep the software working and is a subset of sustainability.

Sustainable Delivery

+ Improved communication with facilities
+ Standardization on spack for delivery
+ E4S containers and build pipelines
+ Availability of resources for testing
+ Community standards

– Scalability of processes
– Complexity of software
– Single point of “failure”
– Inadequate resources

Urgent Priorities

- Clarity on resources
  - For maintenance and “core” activities
  - For research and development
  - For users and applications
- Support models
- Career pathways

Missing Items

- Agreed upon definition of sustainability
- Scalable communities – not just tools
- Growing next generation of users
  - Meeting potential users where they are, not where we want them to be