Preparing for the Sustainable Delivery of the DOE Exascale Software Stack

EXASCALE COMPUTING PROJECT

Michael A. Heroux, Sandia National Laboratories, Director of Software Technology
Rajeev Thakur, Argonne National Laboratory, Programming Models and Runtimes Area Lead
Jeffrey Vetter, Oak Ridge National Laboratory, Development Tools Area Lead

Advanced Scientific Computing Advisory Committee
September 25, 2020

National Nuclear Security Administration

U.S. DEPARTMENT OF ENERGY | Office of Science

ECP Software Technology Leadership Team

Mike Heroux, Software Technology Director
Mike has been involved in scientific software R&D for 30 years. His first 10 were at Cray in the LIBSCI and scalable apps groups. At Sandia he started the Trilinos and Mantevo projects, is author of the HPCG benchmark for TOP500, and leads productivity and sustainability efforts for DOE.

Lois Curfman McInnes, Software Technology Deputy Director
Lois is a senior computational scientist in the Mathematics and Computer Science Division of ANL. She has over 20 years of experience in high-performance numerical software, including development of PETSc and leadership of multi-institutional work toward sustainable scientific software ecosystems.

Rajeev Thakur, Programming Models and Runtimes (2.3.1)
Rajeev is a senior computer scientist at ANL and most recently led the ECP Software Technology focus area. His research interests are in parallel programming models, runtime systems, communication libraries, and scalable parallel I/O. He has been involved in the development of open source software for large-scale HPC systems for over 20 years.

Jeff Vetter, Development Tools (2.3.2)
Jeff is a computer scientist at ORNL, where he leads the Future Technologies Group. He has been involved in research and development of architectures and software for emerging technologies, such as heterogeneous computing and nonvolatile memory, for HPC for over 15 years.

Xaioye (Sherry) Li, Math Libraries (2.3.3)
Sherry is a senior scientist at Berkeley Lab. She has over 20 years of experien ... velopment of SuperLU and related linear algebra algorithms and software.

Jim Ahrens, Data and Visualization (2.3.4)
Jim is a senior research scientist at the Los Alamos National Laboratory ... ly contributes to many open-source data science packages including ParaVi

T...
... ce in high-performance
... e ECP CODAR project.

Many thanks to Rob Neely!
Rob was part of ECP leadership from the very beginning

Kathryn Mohror, NNSA ST (2.3.6)
Kathryn is Group Leader for the CAS ... scale systems, scalable performance analysis and tuning, fault tolerance ... DOE Early Career Award.

ST L4 Teams

- WBS
- Name
- PIs
- PCs - Project Coordinators

ECP ST Stats

- 34 L4 subprojects
- 11 PI/PC same
- 23 PI/PC different
- ~27% ECP budget

WBS	WBS Name	CAM/PI	PC
2.3	**Software Technology**	**Heroux, Mike, McInnes, Lois**	_
2.3.1	**Programming Models & Runtimes**	**Thakur, Rajeev**	_
2.3.1.01	PMR SDK	Shende, Sameer	Shende, Sameer
2.3.1.07	Exascale MPI (MPICH)	Balaji, Pavan	Guo, Yanfei
2.3.1.08	Legion	McCormick, Pat	McCormick, Pat
2.3.1.09	PaRSEC	Bosilica, George	Carr, Earl
2.3.1.14	Pagoda: UPC++/GASNet for Lightweight Communication and Global Address Space Support	Hargrove, Paul	Hargrove, Paul
2.3.1.16	SICM	Lang, Michael	Vigil, Brittney
2.3.1.17	OMPI-X	Bernholdt, David	Grundhoffer, Alicia
2.3.1.18	RAJA/Kokkos	Trott, Christian Robert	Trujillo, Gabrielle
2.3.1.19	Argo: Low-level resource management for the OS and runtime	Beckman, Pete	Gupta, Rinku
2.3.2	**Development Tools**	**Vetter, Jeff**	_
2.3.2.01	Development Tools Software Development Kit	Miller, Barton	Tim Haines
2.3.2.06	Exa-PAPI++: The Exascale Performance Application Programming Interface with Modern C++	Dongarra, Jack	Jagode, Heike
2.3.2.08	Extending HPCToolkit to Measure and Analyze Code Performance on Exascale Platforms	Mellor-Crummey, John	Meng, Xiaozhu
2.3.2.10	PROTEAS-TUNE	Vetter, Jeff	Glassbrook, Dick
2.3.2.11	SOLLVE: Scaling OpenMP with LLVm for Exascale	Chapman, Barbara	Kale, Vivek
2.3.2.12	FLANG	McCormick, Pat	Perry-Holby, Alexis
2.3.3	**Mathematical Libraries**	**Li, Sherry**	_
2.3.3.01	Extreme-scale Scientific xSDK for ECP	Yang, Ulrike	Yang, Ulrike
2.3.3.06	Preparing PETSc/TAO for Exascale	Munson, Todd	Munson, Todd
2.3.3.07	STRUMPACK/SuperLU/FFTX: sparse direct solvers, preconditioners, and FFT libraries	Li, Sherry	Li, Sherry
2.3.3.12	Enabling Time Integrators for Exascale Through SUNDIALS/ Hypre	Woodward, Carol	Woodward, Carol
2.3.3.13	CLOVER: Computational Libraries Optimized Via Exascale Research	Dongarra, Jack	Carr, Earl
2.3.3.14	ALExa: Accelerated Libraries for Exascale/ForTrilinos	Turner, John	Grundhoffer, Alicia
2.3.4	**Data and Visualization**	**Ahrens, James**	_
2.3.4.01	Data and Visualization Software Development Kit	Atkins, Chuck	Bagha, Neelam
2.3.4.09	ADIOS Framework for Scientific Data on Exascale Systems	Klasky, Scott	Grundhoffer, Alicia
2.3.4.10	DataLib: Data Libraries and Services Enabling Exascale Science	Ross, Rob	Ross, Rob
2.3.4.13	ECP/VTK-m	Moreland, Kenneth	Moreland, Kenneth
2.3.4.14	VeloC: Very Low Overhead Transparent Multilevel Checkpoint/Restart/Sz	Cappello, Franck	Ehling, Scott
2.3.4.15	ExaIO - Delivering Efficient Parallel I/O on Exascale Computing Systems with HDF5 and Unify	Byna, Suren	Bagha, Neelam
2.3.4.16	ALPINE: Algorithms and Infrastructure for In Situ Visualization and Analysis/ZFP	Ahrens, James	Turton, Terry
2.3.5	**Software Ecosystem and Delivery**	**Munson, Todd**	_
2.3.5.01	Software Ecosystem and Delivery Software Development Kit	Willenbring, James M	Willenbring, James M
2.3.5.09	SW Packaging Technologies	Gamblin, Todd	Gamblin, Todd
2.3.5.10	ExaWorks	Laney, Dan	Laney, Dan
2.3.6	**NNSA ST**	**Mohror, Kathryn**	_
2.3.6.01	LANL ATDM	Mike Lang	Vandenbusch, Tanya Marie
2.3.6.02	LLNL ATDM	Becky Springmeyer	Gamblin, Todd
2.3.6.03	SNL ATDM	Jim Stewart	Trujillo, Gabrielle

EXASCALE COMPUTING PROJECT

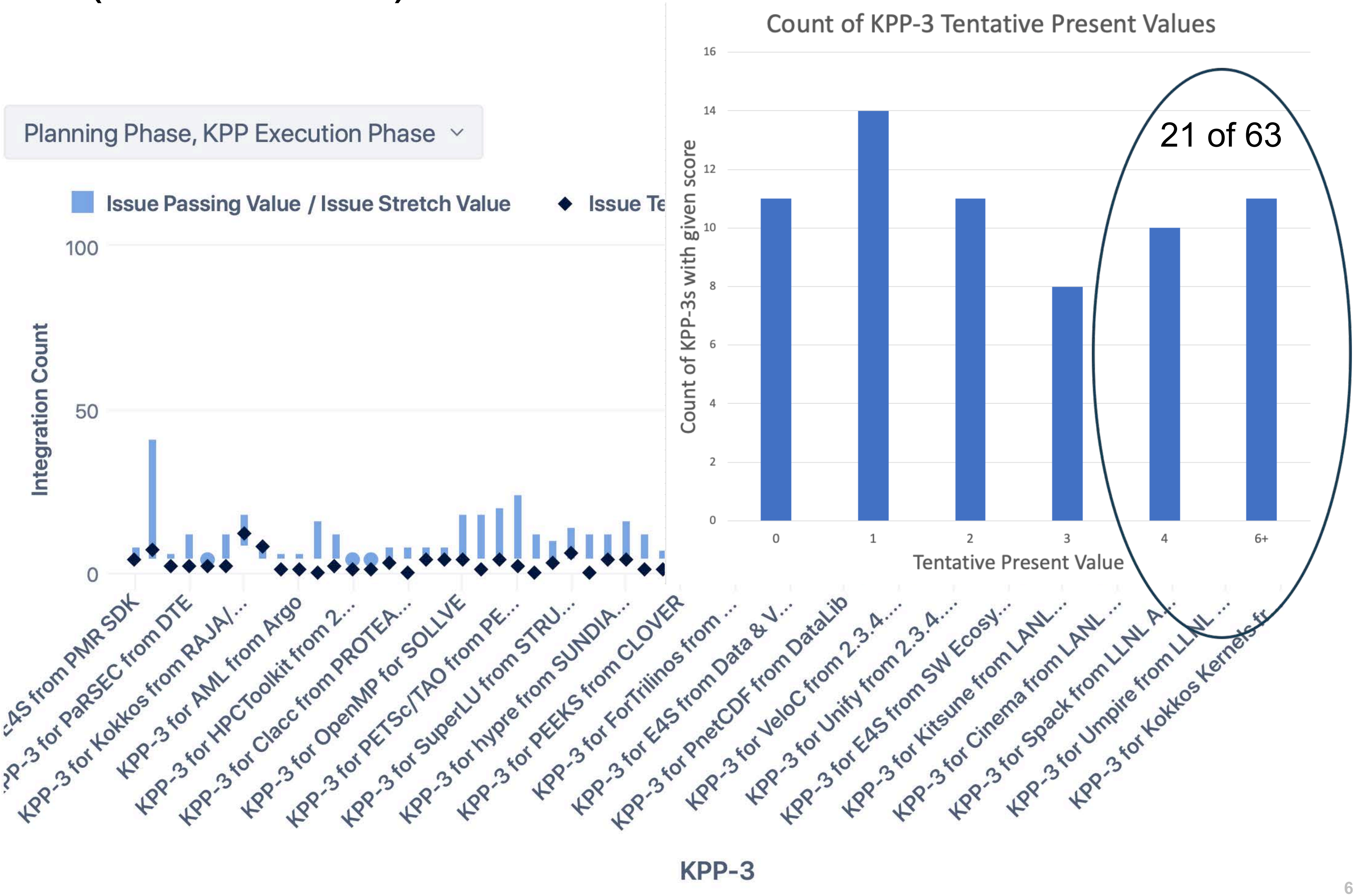
Software Technology Tracking: KPP-3

KPP-3: Focus on capability integration

- **Capability:** Any significant product functionality, including existing features adapted to the pre-exascale and exascale environments, that can be integrated into a client environment.
- **Capability Integration:** Complete, sustainable integration of a significant product capability into a client environment in a pre-exascale environment (tentative score) and in an exascale environment (confirmed score).

KPP-3 Dashboard (2020-09-24)

- Manage KPP-3 progress in Jira
- Special issue type
- Present values updated at least twice a year
- Updates include uploaded artifacts supporting scores
- Measuring tentative present values now
- Confirmed values possible when Aurora, Frontier arrive



Key Takeaways for ECP ST Progress Tracking

- KPP-3 measures the key value proposition of ECP ST activities:
  - The sustainable integration of capabilities demonstrated on the exascale platforms
- Final KPP-3 definition benefitted tremendously from review teams input
- Use of Jira enable real-time tracking of integration progress:
  - At any time, we can see status
  - We collect artifacts as we go
  - Currently 21 of 63 (33%) have achieved *tentative* passing value
  - Require 32 of 63 (>50%) achieving *confirmed* passing value to achieve threshold for KPP-3
- The KPP-3 approach of sustainable capability integration is applicable to other settings:
  - Other reusable software projects
  - Future DOE projects

The Extreme-Scale Scientific Software Stack (E4S):

A collaborative HPC Linux Ecosystem

Delivering an open, hierarchical software ecosystem

Levels of Integration | Product | Source and Delivery

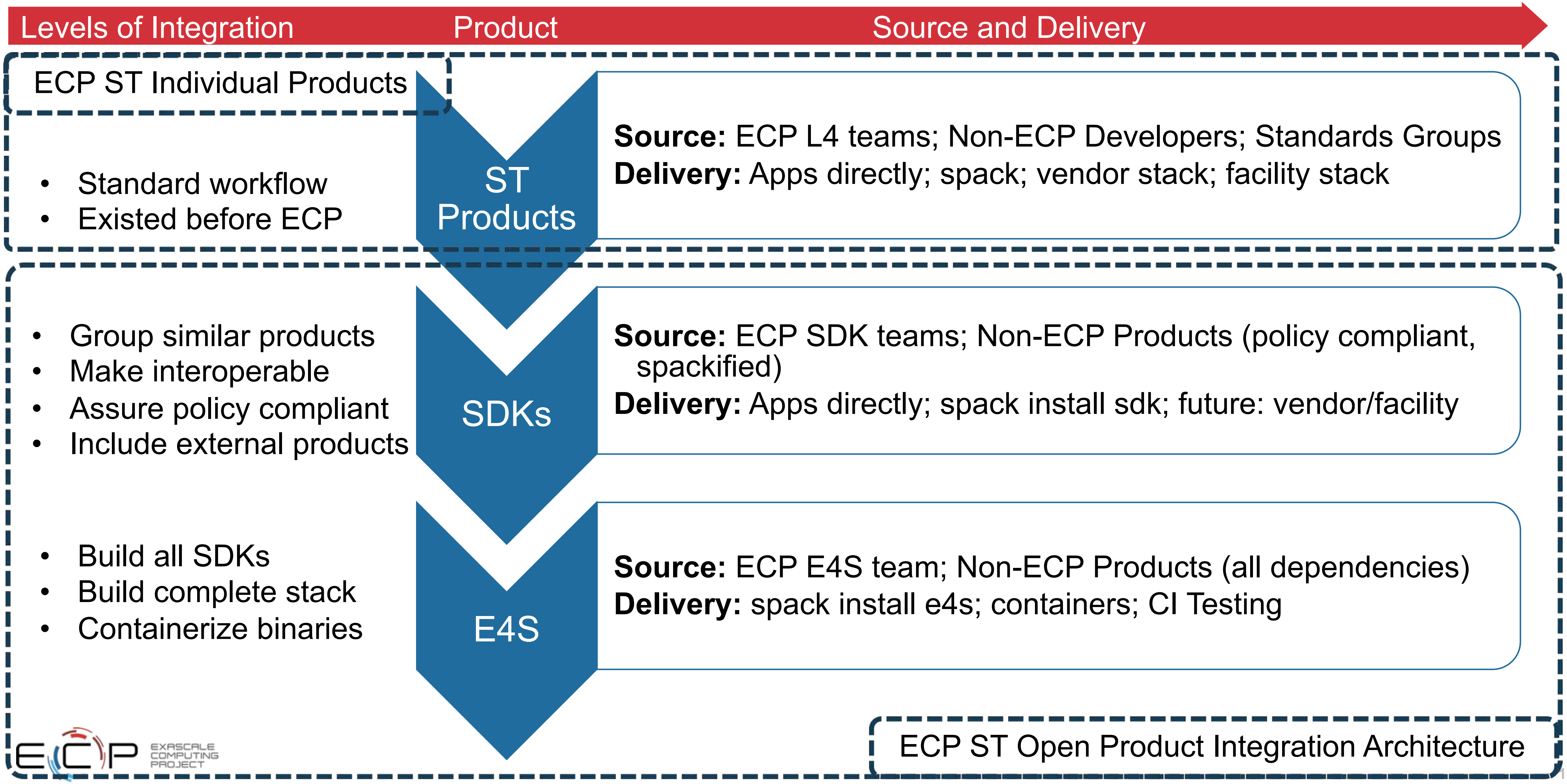
ECP ST Individual Products

- Standard workflow
- Existed before ECP

ST Products

Source: ECP L4 teams; Non-ECP Developers; Standards Groups
Delivery: Apps directly; spack; vendor stack; facility stack

Delivering an open, hierarchical software ecosystem



E4S Components

- E4S is a curated release of ECP ST products based on Spack [http://spack.io].
- E4S Spack cache to support bare-metal installs at facilities and custom container builds:
  - x86_64, ppc64le, and aarch64
- Container images on DockerHub and E4S website of pre-built binaries of ECP ST products.
- Base images and full featured containers (GPU support).
- GitHub recipes for creating custom images from base images.
- e4s-cl for container launch and for replacing MPI in application with system MPI libraries.
- Validation test suite on GitHub provides automated build and run tests.
- Automates build process via GitLab Continuous Integration to ensure packages can be built.
- E4S Doc Portal aggregates and summarizes documentation and metadata by raking product repos.
- E4S VirtualBox image with support for Docker, Shifter, Singularity, and Charliecloud runtimes.
- AWS image to deploy E4S on EC2.

https://e4s.io

E4S Community Policies

Community-driven Quality Commitments

xSDK community policies

https://xsdk.info/policies

xSDK compatible package: Must satisfy mandatory xSDK policies:

M1. Support xSDK community GNU Autoconf or CMake options.
M2. Provide a comprehensive test suite.
M3. Employ user-provided MPI communicator.
M4. Give best effort at portability to key architectures.
M5. Provide a documented, reliable way to contact the development team.
M6. Respect system resources and settings made by other previously called packages.
M7. Come with an open source license.
M8. Provide a runtime API to return the current version number of the software.
M9. Use a limited and well-defined symbol, macro, library, and include file name space.
M10. Provide an accessible repository (not necessarily publicly available).
M11. Have no hardwired print or IO statements that cannot be turned off.
M12. For external dependencies, allow installing, building, and linking against an outside copy of external software.
M13. Install headers and libraries under <prefix>/include/ and <prefix>/lib/.
M14. Be buildable using 64 bit pointers. 32 bit is optional.
M15. All xSDK compatibility changes should be sustainable.
M16. The package must support production-quality installation compatible with the xSDK install tool and xSDK metapackage.

Also **recommended policies**, which currently are encouraged but not required:

R1. Have a public repository.
R2. Possible to run test suite under valgrind in order to test for memory corruption issues.
R3. Adopt and document consistent system for error conditions/exceptions.
R4. Free all system resources it has acquired as soon as they are no longer needed.
R5. Provide a mechanism to export ordered list of library dependencies.
R6. Document versions of packages that it works with or depends on, preferably in machine-readable form
R7. Have README, SUPPORT, LICENSE, and CHANGELOG files in top directory.

xSDK member package: Must be an xSDK-compatible package, *and* it uses or can be used by another package in the xSDK, and the connecting interface is regularly tested for regressions.

We welcome feedback. What policies make sense for your software?

better scientific software

BSSw blog article:
P. Luszczek and U. Yang, Aug 2019,
https://bssw.io/blog_posts/building-community-through-software-policies

E4S Community Candidate Policies V 1.0 Beta

- **Spack-based Build and Installation**

  Each E4S member package supports a scriptable Spack build and production-quality installation in a way that is compatible with other E4S member packages in the same environment. When E4S build, test, or installation issues arise, there is an expectation that teams will collaboratively resolve those issues.

- **Minimal Validation Testing**

  Each E4S member package has at least one test that is executable through the E4S validation test suite (https://github.com/E4S-Project/testsuite). This will be a post-installation test that validates the usability of the package. The E4S validation test suite provides basic confidence that a user can compile, install and run every E4S member package. The E4S team can actively participate in the addition of new packages to the suite upon request.

- **Sustainability**

  All E4S compatibility changes will be sustainable in that the changes go into the regular development and release versions of the package and should not be in a private release/branch that is provided only for E4S releases.

- **Product Metadata**

  Each E4S member package team will provide key product information via metadata that is organized in the E4S DocPortal format. Depending on the filenames where the metadata is located, this may require minimal setup.

- **Public Repository**

  Each E4S member package will have a public repository, for example at GitHub or Bitbucket, where the development version of the package is available and pull requests can be submitted.

- **Imported Software**

  If an E4S member package imports software that is externally developed and maintained, then it must allow installing, building, and linking against a functionally equivalent outside copy of that software. Acceptable ways to accomplish this include (1) forsaking the internal copied version and using an externally-provided implementation or (2) changing the file names and namespaces of all global symbols to allow the internal copy and the external copy to coexist in the same downstream libraries and programs.

- **Error Handling**

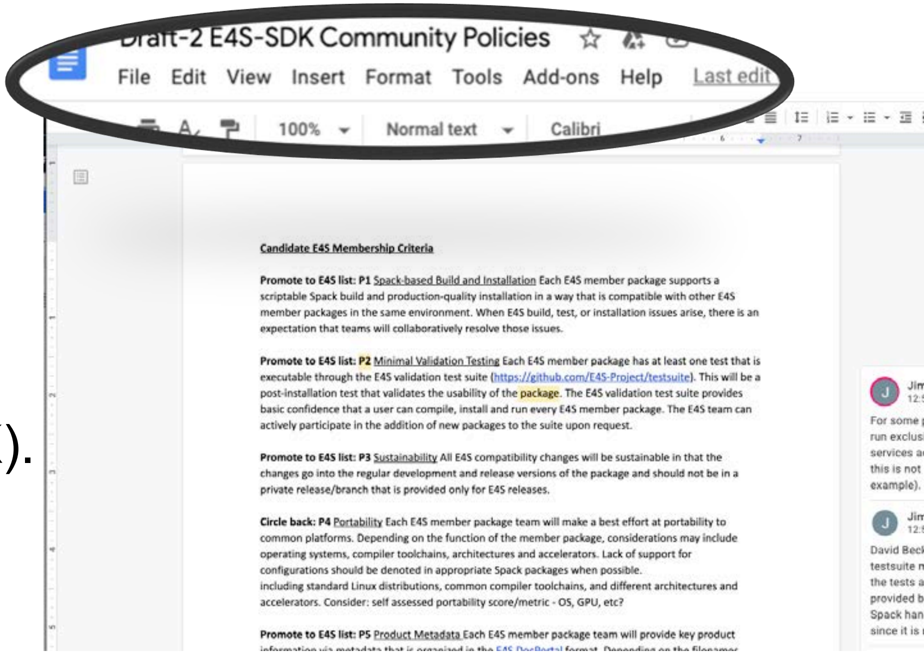
  Each E4S member package will adopt and document a consistent system for signifying error conditions as appropriate for the language and application. For e.g., returning an error condition or throwing an exception. In the case of a command line tool, it should return a sensible exit status on success/failure, so the package can be safely run from within a script.

- **Test Suite**

  Each E4S member package will provide a test suite that does not require special system privileges or the purchase of commercial software. This test suite should grow in its comprehensiveness over time. That is, new and modified features should be included in the suite.

E4S/SDK Policy Initiative Status

- Community policies are important for several reasons:
  - Commitment to quality
  - Membership criteria for the future
  - Community discussion
- Each SDK community developing policies like Math Libs (xSDK).
- Policies common to all SDKs will be promoted to E4S level
- Policies will determine:
  - Quality label
  - Membership in E4S and the SDKs
- Version 1.0 of policies due by end of 2020



Draft-2 E4S-SDK Community Policies

File Edit View Insert Format Tools Add-ons Help Last edit

Candidate E4S Membership Criteria

Promote to E4S list: P1 Spack-based Build and Installation Each E4S member package supports a scriptable Spack build and production-quality installation in a way that is compatible with other E4S member packages in the same environment. When E4S build, test, or installation issues arise, there is an expectation that teams will collaboratively resolve those issues.

Promote to E4S list: P2 Minimal Validation Testing Each E4S member package has at least one test that is executable through the E4S validation test suite (https://github.com/E4S-Project/testsuite). This will be a post-installation test that validates the usability of the package. The E4S validation test suite provides basic confidence that a user can compile, install and run every E4S member package. The E4S team can actively participate in the addition of new packages to the suite upon request.

Promote to E4S list: P3 Sustainability All E4S compatibility changes will be sustainable in that the changes go into the regular development and release versions of the package and should not be in a private release/branch that is provided only for E4S releases.

Circle back: P4 Portability Each E4S member package team will make a best effort at portability to common platforms. Depending on the function of the member package, considerations may include operating systems, compiler toolchains, architectures and accelerators. Lack of support for configurations should be denoted in appropriate Spack packages when possible.
including standard Linux distributions, common compiler toolchains, and different architectures and accelerators. Consider: self assessed portability score/metric - OS, GPU, etc?

Promote to E4S list: P5 Product Metadata Each E4S member package team will provide key product

E4S DocPortal

A Single Portal with Redirect to Product Documentation

Product Documentation Challenges: User Perspective

Finding info for specific product	What it does License Support Contact info More …
Finding new products	What can solve my problem
Trusting accuracy of information	Up to date Complete
Hierarchical	Summary to deep dive

Product Documentation Challenges: Developer Perspective

Efficient and Effective generation and maintenance

Getting noticed by new users

Conveying summary information *and* details

E4S Documentation Portal Strategy

All content resides in product repositories

Use open source community approach of specially-name files in software repositories.

Adopt commonly used file names when available.

ID new information items not already being requested.

Documentation portal provides single point of access

Web-based raking tool capture information from product repositories and present in summary form.

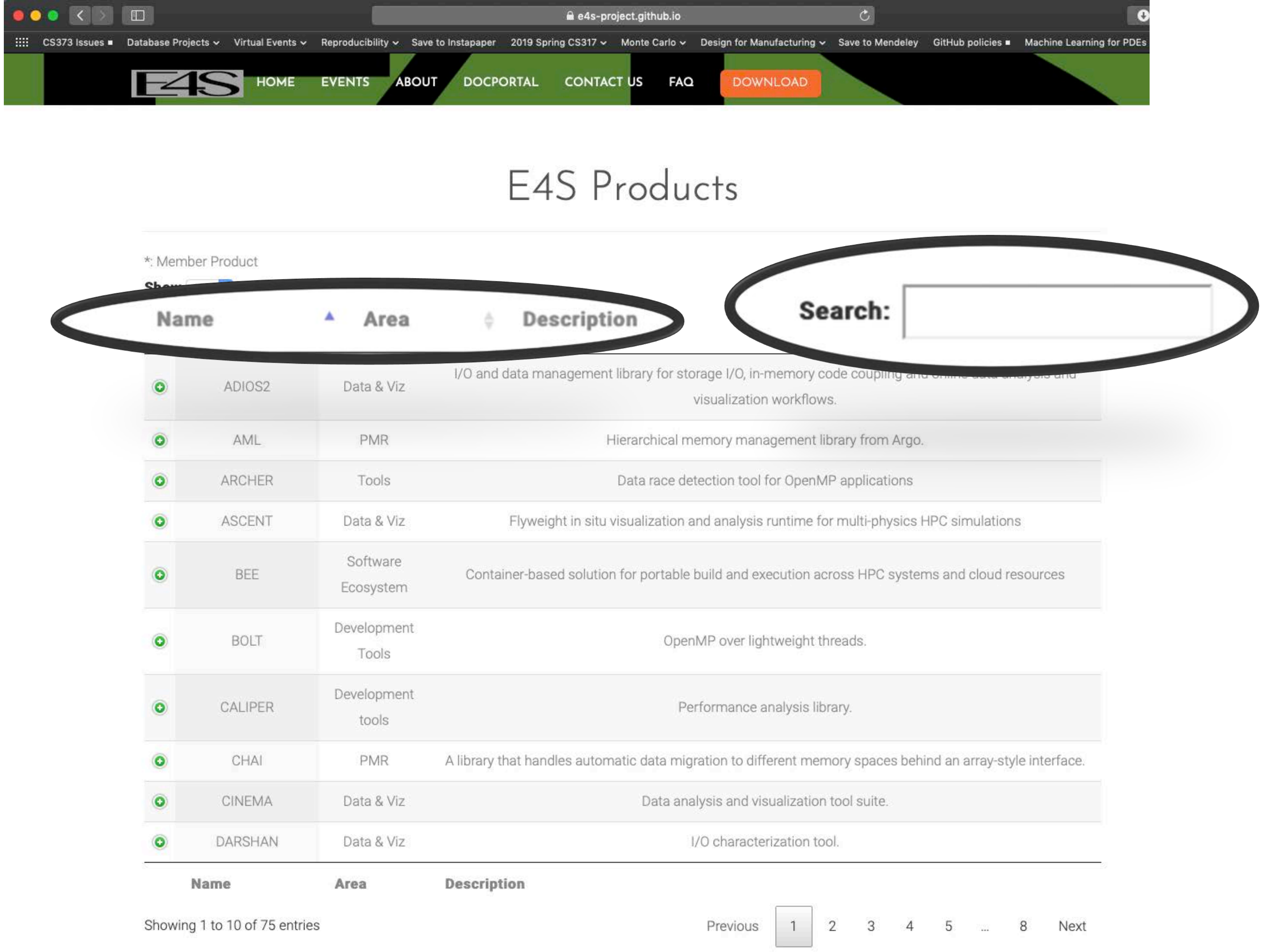
Aggregates and summarizes documentation and metadata for E4S products

Regularly updates information directly from product repositories

Location: https://e4s-project.github.io/DocPortal.html

E4S DocPortal

- The DocPortal is live!
- Summary Info
  - Name
  - Functional Area
  - Description
  - License
- Searchable
- Sortable



e4s-project.github.io

CS373 Issues · Database Projects · Virtual Events · Reproducibility · Save to Instapaper · 2019 Spring CS317 · Monte Carlo · Design for Manufacturing · Save to Mendeley · GitHub policies · Machine Learning for PDEs

E4S · HOME · EVENTS · ABOUT · DOCPORTAL · CONTACT US · FAQ · DOWNLOAD

E4S Products

*: Member Product

Search:

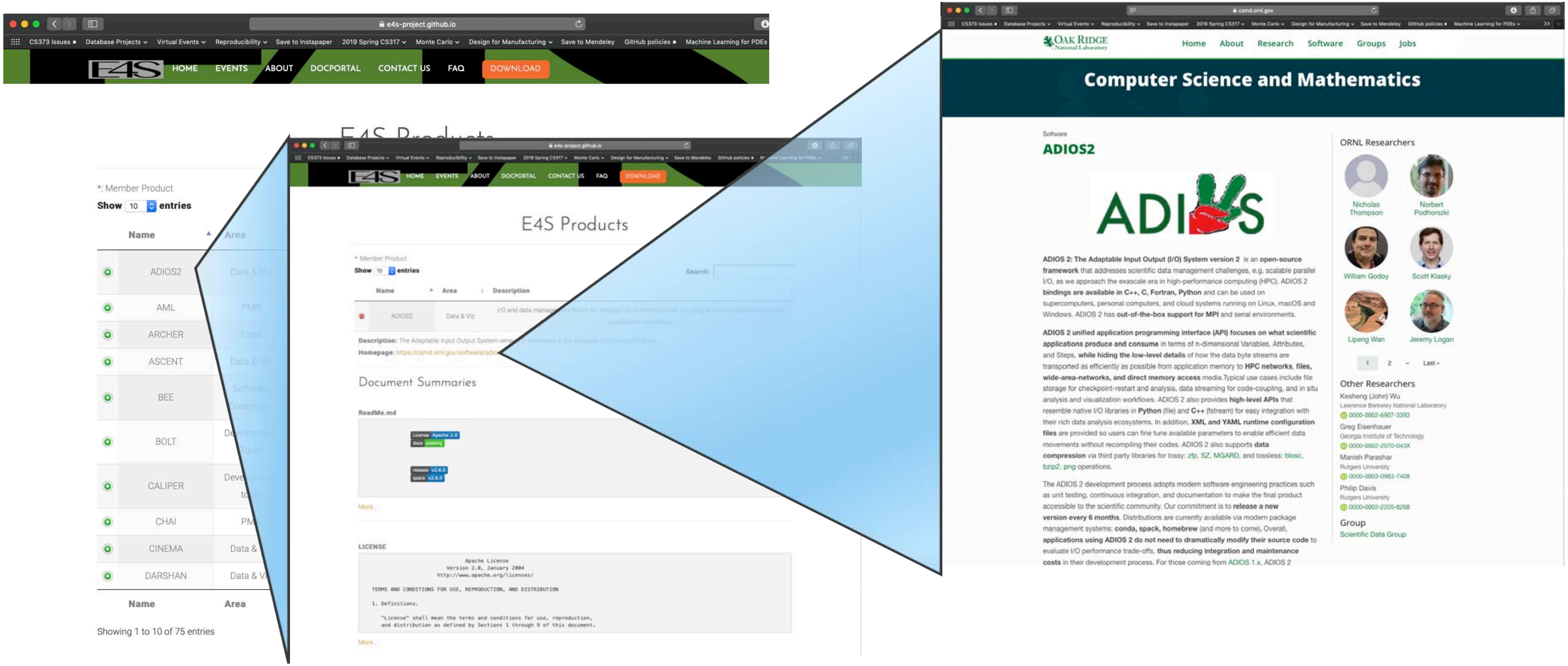
Name	Area	Description
ADIOS2	Data & Viz	I/O and data management library for storage I/O, in-memory code coupling and online data analysis and visualization workflows.
AML	PMR	Hierarchical memory management library from Argo.
ARCHER	Tools	Data race detection tool for OpenMP applications
ASCENT	Data & Viz	Flyweight in situ visualization and analysis runtime for multi-physics HPC simulations
BEE	Software Ecosystem	Container-based solution for portable build and execution across HPC systems and cloud resources
BOLT	Development Tools	OpenMP over lightweight threads.
CALIPER	Development tools	Performance analysis library.
CHAI	PMR	A library that handles automatic data migration to different memory spaces behind an array-style interface.
CINEMA	Data & Viz	Data analysis and visualization tool suite.
DARSHAN	Data & Viz	I/O characterization tool.
Name	Area	Description

Showing 1 to 10 of 75 entries

Previous 1 2 3 4 5 ... 8 Next

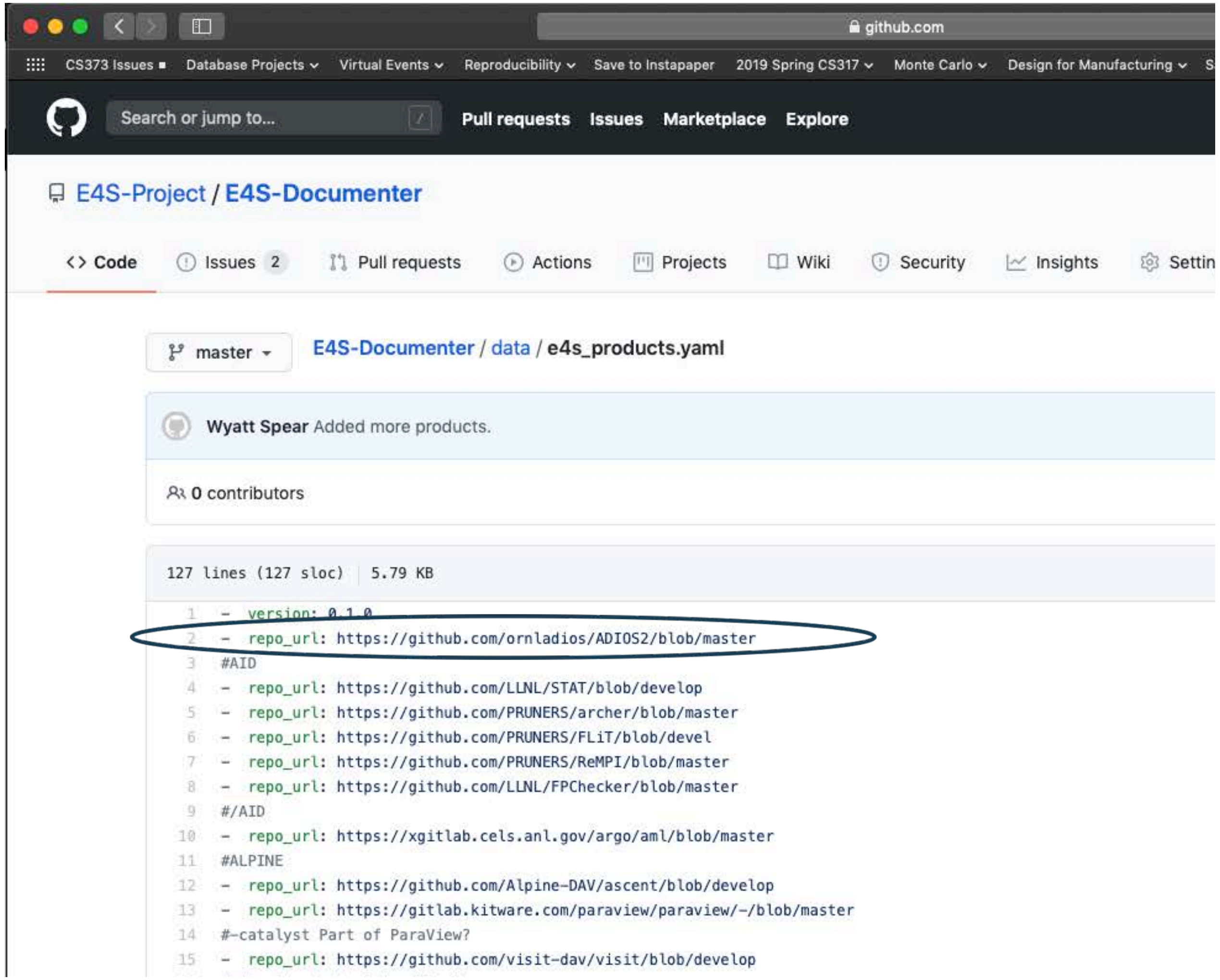
https://e4s-project.github.io/DocPortal.html

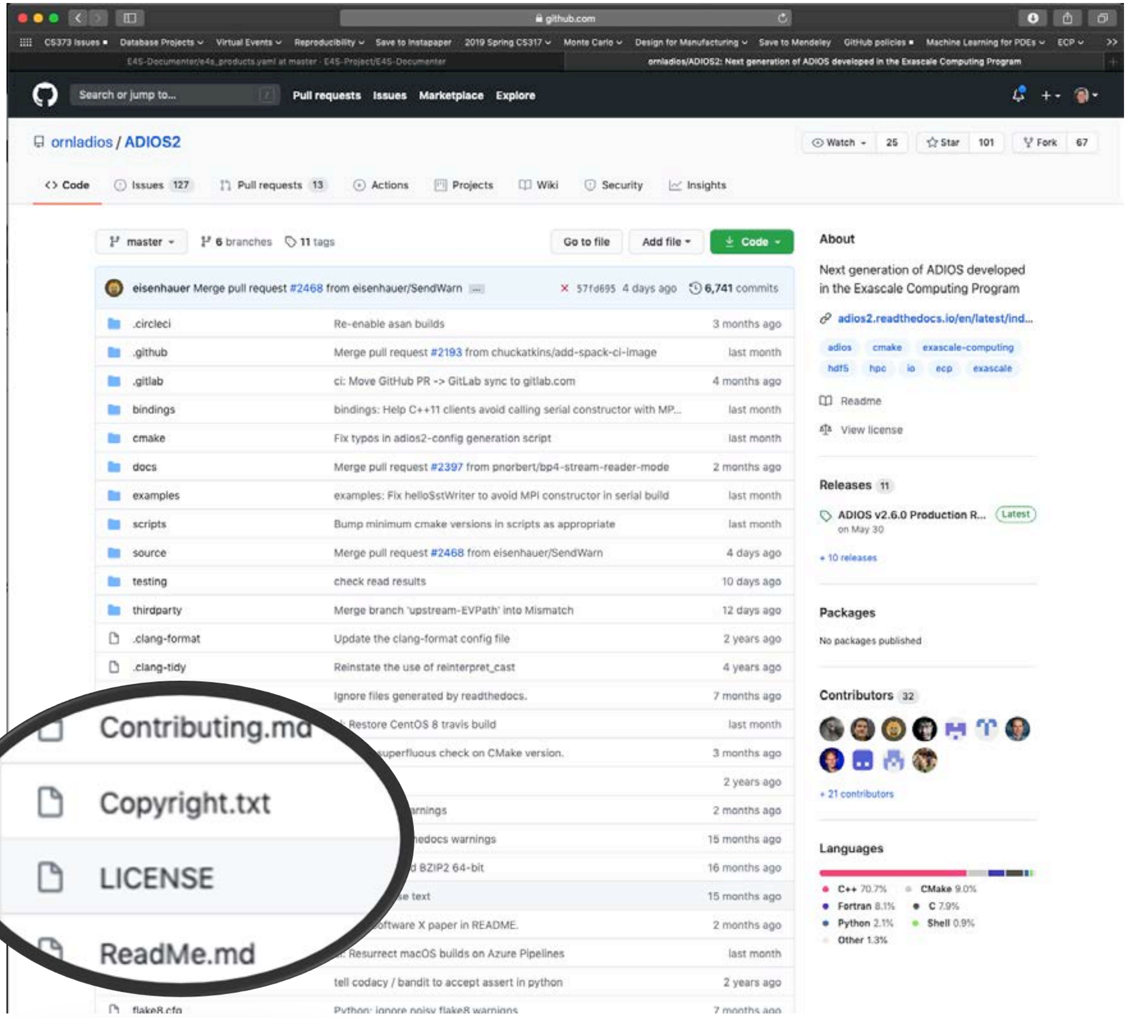
Goal: All E4S Product Documentation Accessible from single portal on E4S.io (Working Mock Webpage below)



https://e4s-project.github.io/DocPortal.html

Q: What do we need for adding a product to the DocPortal?
A: A repo URL + up-to-date meta-data files

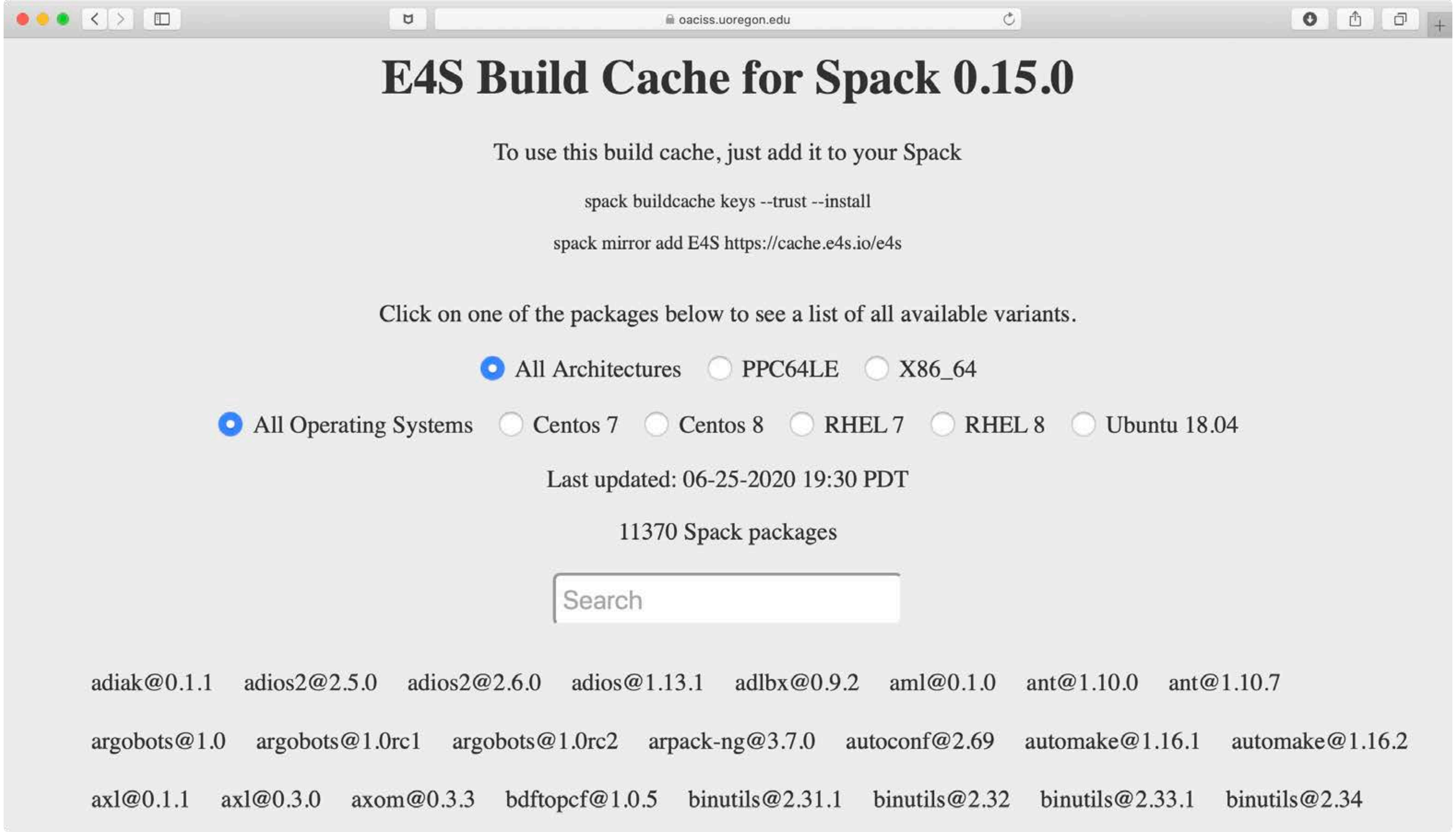




Takeaway: Adding new products is very straightforward!

E4S Spack Build Cache and Container Build Pipeline

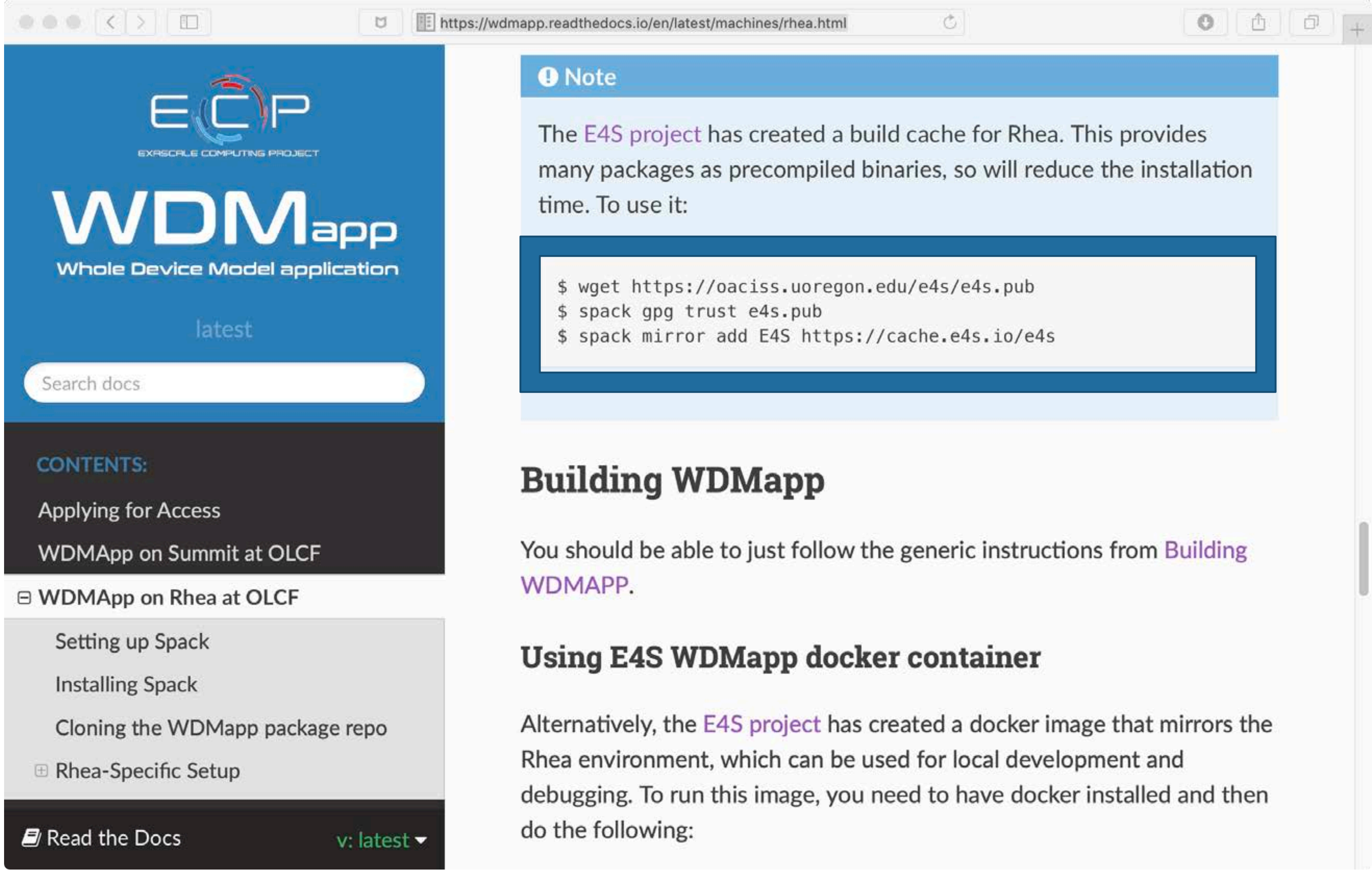
E4S: Spack Build Cache at U. Oregon



- 10,000+ binaries
- S3 mirror
- No need to build from source code!

- https://oaciss.uoregon.edu/e4s/inventory.html

WDMApp: Speeding up bare-metal installs using E4S build cache



Special Thanks to Sameer Shende, WDMapp Team

- E4S Spack build cache
- Adding E4S mirror
- WDMApp install speeds up!

- https://wdmapp.readthedocs.io/en/latest/machines/rhea.html

EXASCALE COMPUTING PROJECT

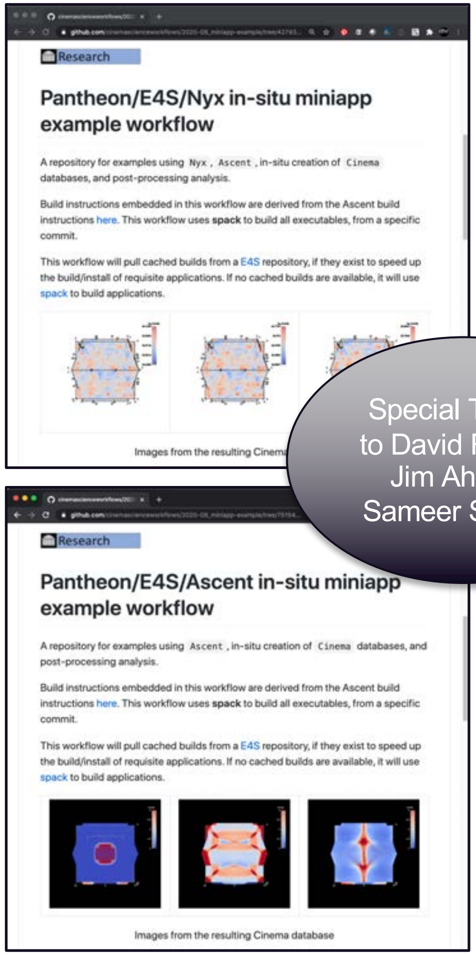
Pantheon and E4S support end-to-end ECP examples

Overview: The Exascale Computing Project (ECP) is a complex undertaking, involving a myriad of technologies working together. An outstanding need is a way to capture, curate, communicate and validate workflows that cross all of these boundaries.

The **Pantheon** and **E4S** projects are collaborating to advance the integration and testing of capabilities, and to promote understanding of the complex workflows required by the ECP project. Utilizing a host of ECP technologies (spack, Ascent, Cinema, among others), this collaboration brings curated workflows to the fingertips of ECP researchers.

Contributions

- Curated end-to-end application/in-situ analysis examples can be run quickly by anyone on Summit. (https://github.com/pantheonscience/ECP-E4S-Examples)
- Pantheon/E4S integration speeds up build/setup times over source builds due to cached binaries (approx.10x speed up).



Special Thanks to David Rogers, Jim Ahrens, Sameer Shende

Instructions page for (top) Nyx, Ascent and Cinema workflow repository, and (bottom) Cloverleaf3d, Ascent, Cinema workflow. These curated workflows use Pantheon, E4S and spack to provide curated workflows for ECP.

E4S Summary

What E4S is not	What E4S is
• A closed system taking contributions only from DOE software development teams.	• Extensible, open architecture software ecosystem accepting contributions from US and international teams. • Framework for collaborative open-source product integration.
• A monolithic, take-it-or-leave-it software behemoth.	• A full collection if compatible software capabilities **and** • A manifest of a la carte selectable software capabilities.
• A commercial product.	• Vehicle for delivering high-quality reusable software products in collaboration with others.
• A simple packaging of existing software.	• The conduit for future leading edge HPC software targeting scalable next-generation computing platforms. • A hierarchical software framework to enhance (via SDKs) software interoperability and quality expectations.

ECP EXASCALE COMPUTING PROJECT

The Second Extreme-scale Scientific Software Stack Forum (E4S Forum) September 24th, 2020, Workshop at EuroMPI/USA'20

- Presenters from 11 institutions, 6 non-DOE
- 70 participants
  - DOE Labs, NASA
  - AMD
  - HLRS, CSCS

- E4S: The Extreme-scale Scientific Software Stack for Collaborative Open Source Software, Michael Heroux, Sandia National Laboratories
- Title: Practical Performance Portability at CSCS, **Ben Cumming, CSCS**
- Title: An Overview of High Performance Computing and Computational Fluid Dynamics at NASA, **Eric Nielsen, NASA Langley**
- Towards An Integrated and Resource-Aware Software Stack for the EU Exascale Systems, **Martin Schulz, Technische Universität München**
- Spack and E4S, Todd Gamblin, LLNL
- Rocks and Hard Places – Deploying E4S at Supercomputing Facilities, Ryan Adamson, Oak Ridge Leadership Computing Facility
- Advances in, and Opportunities for, LLVM for Exascale, Hal Finkel, Argonne National Laboratory
- Kokkos: Building an Open Source Community, Christian Trott, SNL
- Experiences in Designing, Developing, Packaging, and Deploying the MVAPICH2 Libraries in Spack, **Hari Subramoni, Ohio State University**
- Software Needs for Frontera and the NSF Leadership Class Computing Facility – the Extreme Software Stack at the Texas Advanced Computing Center, **Dan Stanzione, TACC**
- Building an effective ecosystem of math libraries for exascale, Ulrike Yang
- Towards Containerized HPC Applications at Exascale, Andrew Younge, Sandia
- E4S Overview and Demo, Sameer Shende, University of Oregon
- The Supercomputer “Fugaku” and Software, programming models and tools, **Mitsuhisa Sato, RIKEN Center for Computational Science (R-CCS), Japan**

Vision for E4S Now and in the Future

- E4S has emerged as a new top-level component in the DOE HPC community, enabling fundamentally new relationships
- E4S has similar potential for new interactions with other US agencies, US industry and international collaborators. NSF and UK are examples
- The E4S portfolio can expand to include new domains (ML/AI), lower—level components (OS), and more.
- E4S can provide better (increased quality), faster (timely delivery of leading edge capabilities) and cheaper (assisting product teams)

E4S/SDK Summary

- E4S/SDK Software: Curated release of complete production-quality HPC Linux software stack:
  - **Latest ECP-developed features** for 50+ products.
  - **Ported and validated** regularly on all common and emerging HPC platforms.
  - **Single DocPortal access** to all product documentation.
  - **Collaborative development communities** around SDKs to build culture of quality.
  - **Policies** for SW and user experience quality.
  - **Containers, build caches** for (dramatic) reduction in build time and complexity.
- E4S: A new member of the HPC ecosystem:
  - **A managed portfolio** of HPC software teams and products.
  - **Enabling first-of-a-kind collaboration:** vendors, facilities, US agencies, industry and internationally.
  - Extensible to new domains: **AI/ML.**
  - **A new way of delivering reusable HPC software** with ever-improving quality and functionality.

https://e4s.io

Next

- Rajeev Thakur will highlight progress in programming models, highlighting efforts in MPICH, Kokkos, RAJA
- Jeff Vetter will describe efforts to integrate ECP ST work into the LLVM ecosystem

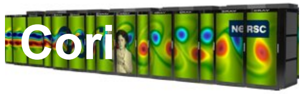
Programming Models and Performance Portability

Rajeev Thakur
Argonne National Laboratory

September 25, 2020

EXASCALE COMPUTING PROJECT
National Nuclear Security Administration
U.S. DEPARTMENT OF ENERGY
Office of Science

DOE HPC Roadmap to Exascale Systems

FY 2012	FY 2016	FY 2018	FY 2021	FY 2022	FY 2023
 **ORNL** Cray/AMD/NVIDIA		 **ORNL** IBM/NVIDIA	 **ORNL** HPE/AMD		
 **ANL** IBM BG/Q	 **ANL** Cray/Intel KNL		 **ANL** Intel/HPC		
	 **LBNL** Cray/Intel Xeon/KNL		 **LBNL** HPE/AMD/NVIDIA		
 **LLNL** IBM BG/Q	 **LANL/SNL** Cray/Intel Xeon/KNL	 **LLNL** IBM/NVIDIA		CROSSROADS **LANL/SNL** HPE/TBD	 **LLNL** HPE/AMD

Exascale Systems

To date, only NVIDIA GPUs

GPUs from three different vendors

EXASCALE COMPUTING PROJECT

Trends in Internode Programming

- Individual compute nodes are becoming very powerful because of accelerators
- As a result, fewer total number of nodes are needed
- MPI will continue to serve as the main internode programming model
- Nonetheless, improvements are needed in the MPI Standard and in MPI implementations
  - Hybrid programming (integration with GPUs & GPU memory and with the node programming model)
  - Scalability, low-latency communication, optimized collective algorithms
  - Overall resilience and robustness
  - Optimized support for exascale interconnects and lower-level communication paradigms (OFI, UCX)
  - Scalable process startup and management
- PGAS models (e.g., UPC++) are also available to be used by applications that rely on them, and they face similar challenges as MPI on exascale systems

Trends in Intranode Programming

- Main challenge for exascale is in achieving performance and portability for intranode programming
- Vendor-supported options for GPUs
  - NVIDIA: CUDA, OpenACC
  - Intel: SYCL/DPC++ (C++ abstractions on top of OpenCL)
  - AMD: HIP (similar to CUDA)
- OpenMP (portable standard)
  - Supports accelerators via the `target` directive (since OpenMP version 4.0, July 2013)
  - Subsequent releases of OpenMP (4.5 and 5.0) have further improved support for accelerators
  - Supported by vendors on all platforms
- Kokkos and RAJA (developed at SNL and LLNL)
  - Portable, heterogenous-node programming via C++ abstractions
  - Support complex node architectures with multiple types of execution resources and multilevel memories
  - Many ECP applications use Kokkos and RAJA to write portable code for a variety of CPUs and GPUs

Intranode Programming Models being used in ECP Application Codes (1)

Application project	Code	Main language	GPU programming model
ExaStar	FLASH	Fortran	OpenMP
ExaStar	CASTRO	Fortran, C++	OpenMP, OpenACC
EQSIM	SW4	C++	RAJA
ExaSky	HACC	C++	CUDA, OpenCL
ExaSky	CRK-HACC	C++	CUDA, OpenCL
ExaSky	Nyx	C++	AMReX
Subsurface	Chombo-Crunch	C++	PROTO, UPC++
Subsurface	GEOSX	C++	RAJA
E3SM-MMF	E3SM	Fortran	OpenACC, moving to OpenMP
Combustion-PELE	PeleC	Fortran	CUDA, OpenACC
Combustion-PELE	PeleLM	Fortran	CUDA, OpenACC
WarpX	WarpX + PICSAR	C++	AMReX abstractions
ExaSMR	Nek5000	Fortran	OpenACC
ExaSMR	NekRS	Fortran	libParanumal (OCCA)
ExaSMR	OpenMC	C++	OpenMP, OpenCL or SYCL
ExaSMR	Shift	C++	CUDA
WDMApp	GENE	Fortran	OpenMP
WDMApp	GEM	Fortran	OpenACC
WDMApp	XGC	Fortran	OpenMP, OpenACC
MFIX-Exa	MFIX-Exa	C++	AMReX abstractions
ExaWind	Nalu-Wind	C++	Kokkos
ExaWind	OpenFAST	Fortran 90	N/A

ECP Application Development Milestone Report, Appendix A, March 2020

Intranode Programming Models being used in ECP Application Codes (2)

ExaBiome	MetaHipMer	C++	UPC++
ExaBiome	GOTTCHA	C++	OpenMP, HIP, SYCL
ExaBiome	HipMCL	C++	OpenMP, HIP, SYCL
ExaFEL	M-TIP	C++	CUDA, HIP, OpenCL
ExaFEL	PSANA	C++	Legion
CANDLE	CANDLE	Python	TensorFlow, PyTorch
ExaSGD	GridPACK	C++	
ExaSGD	PIPS	C++	RAJA or Kokkos
ExaSGD	StructJuMP	Julia	
QMCPACK	QMCPACK	C++	OpenMP
ExaAM	MEUMAPPS-SS	Fortran	OpenMP, OpenACC
ExaAM	ExaConstit	C++	MFEM
ExaAM	TruchasPBF	Fortran	AMReX
ExaAM	Diablo	Fortran	OpenMP
ExaAM	ExaCA	C++	Kokkos
NWChemEx	NWChemEx	C++	CUDA, Kokkos
LatticeQCD	Chroma	C++	Kokkos
LatticeQCD	CPS	C++	GRID library
LatticeQCD	MILC	C	GRID library
GAMESS	GAMESS	Fortran	libcchem, libaccint
GAMESS	libcchem	C++	libaccint
EXAALT	ParSplice	C++	N/A
EXAALT	LAMMPS	C++	Kokkos
EXAALT	SNAP	C++	Kokkos

ECP Application Development Milestone Report, Appendix A, March 2020

ECP Programming Models and Runtimes Portfolio

Project Short Name	PI Name, Inst	Short Description/Objective
PMR SDK	Shende, UOregon	Support the deployment, testing and usage of PMR products
Exascale MPI	Balaji, ANL	Enhancement of the MPI Standard and the MPICH implementation of MPI for exascale
Legion	McCormick, LANL	Task-based programming model
PaRSEC	Dongarra, UTK	Task-based programming model
UPC++/GASNet	Hargrove, LBNL	Partitioned Global Address Space (PGAS) programming model
SICM	Lang, LANL	Interface and library for accessing complex memory hierarchy
OMPI-X	Bernholdt, ORNL	Enhancement of the MPI Standard and the Open MPI implementation for exascale
Kokkos / RAJA	Trott, SNL	C++ abstractions for node-level performance portability
Argo	Beckman, ANL	Low-level resource management for the operating system and runtime

*OpenMP (SOLLVE) project moved to Development Tools area to keep all LLVM-related efforts in one area.

Kokkos and RAJA

- C++ performance portability abstractions developed at Sandia and Livermore labs
- Primarily funded by NNSA
- ECP ST provides additional funding to develop optimized backends for Aurora and Frontier (OpenMP, SYCL/DPC++, HIP) and for outreach to ECP applications
- Organized as one project in ECP ST. The two teams collaborate on common features, C++ and backend support, and outreach activities
- The combined project involves six labs: SNL, LLNL, ANL, LANL, LBNL, and ORNL
  - PI: Christian Trott (SNL); Co-PIs: Rich Hornung (LLNL), Hal Finkel (ANL), Galen Shipman (LANL), Jack Deslippe (LBNL), Damien Lebrun-Grandie (ORNL)

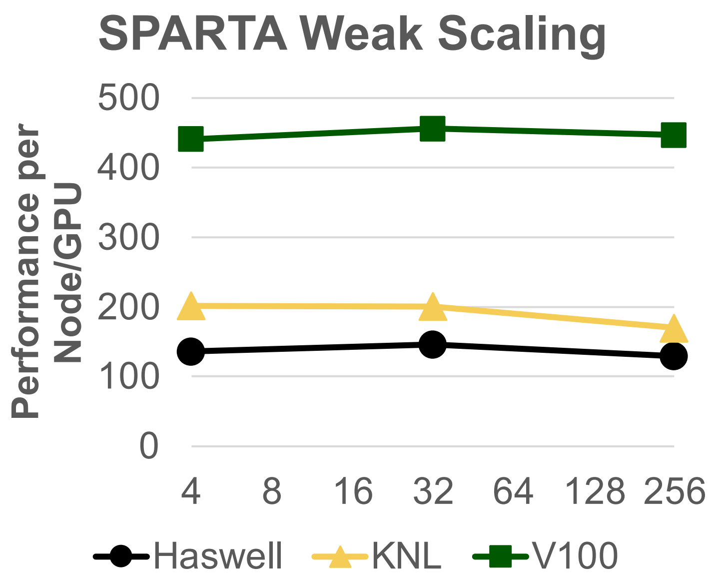
Kokkos: Preparation for Exascale Platforms

- Kokkos provides a production-quality solution for C++ performance portability
  - Kokkos Core: C++ template-based library for the programming model
  - Kokkos Tools: Profiling, debugging and tuning tools that connect into Kokkos Core
  - Kokkos Kernels: Math libraries based on Kokkos Core
- Distributed development team: SNL, ORNL, ANL, LBNL, and LANL
- Long-term goal is alignment and extension of the ISO C++ standard
  - Fundamental capabilities from Kokkos are proposed for the C++ standard
- Kokkos is currently used on most DOE and many European production HPC systems
  - Numerous applications use Kokkos to run on Sierra, Summit, Astra, Trinity, Theta, PizDaint, and others
- Extensively used across ECP AD and ST projects
  - Exawind, EXAALT, WDMApp, ExaAM, SNL and LANL ATDM apps, ExaGraph, CoPA, ALExa (ArborX and DTK), Kokkos Kernels, Trilinos, FleCSI, …
- Numerous workshops and training events
  - Now available “The Kokkos Lectures”: 15 hours of recorded lectures https://kokkos.link/the-lectures.

EXASCALE COMPUTING PROJECT

Kokkos: SPARTA – An Example of Production Runs

- **S**tochastic **PA**rallel **R**arefied-gas **T**ime-accurate **A**nalyzer
- A direct simulation Monte Carlo code
- Developers: *Steve Plimpton, Stan Moore, Michael Gallis*
- Only application to have run on all of Trinity
- Benchmarked on 16K GPUs on Sierra



Exascale Support Status

- HIP support in Kokkos 3.2 for AMD GPUs
  - Currently missing hierarchical parallelism and tasking due to compiler bug
  - Some apps already running (ArborX, some LAMMPS runs, ...)
- OpenMP Target support for Intel GPUs in Kokkos 3.2
  - Some MiniApps are already running
- Expect most functionality will be available for HIP and OpenMP in Kokkos 3.3 in Nov 2020
- SYCL/DPC++ port in progress. Encountered compiler/runtime bugs, which are getting fixed.

RAJA: Preparation for Exascale Platforms

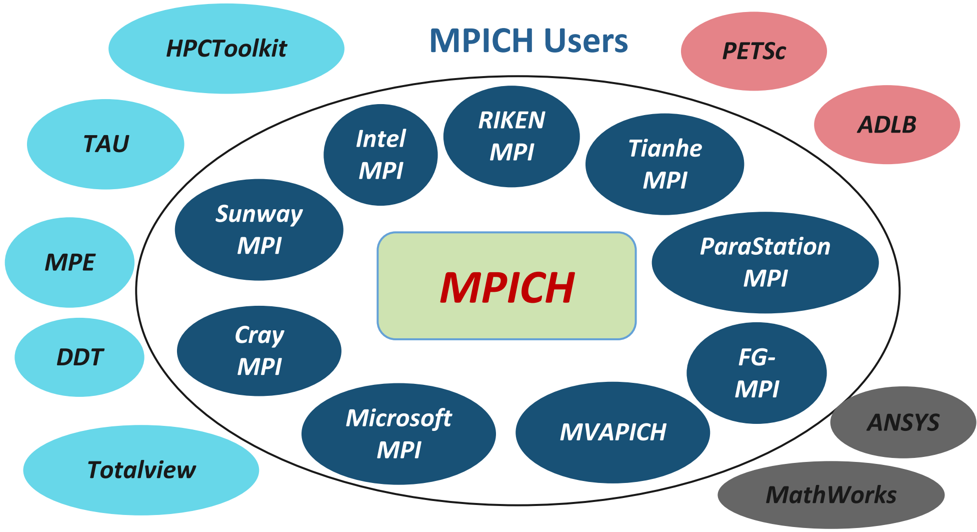
- A suite of performance-portability tools developed at LLNL
  - RAJA (C++ kernel execution abstractions); CHAI (C++ array abstractions); Umpire (memory management)
- Used in a diverse set of ECP and LLNL mission applications and support libraries
  - Supports majority of production LLNL weapons program apps and apps in other LLNL programs
  - ECP: SW4 (EQSIM), GEOSX (Subsurface), MFEM (CEED), ExaSGD, SUNDIALS, DevilRay (Alpine)
- Exascale platform support
  - Full support for HIP back-end in public releases since January 2020. Key part of El Capitan CoE activities.
  - SYCL back-end in progress, including SW4 & RAJA Performance Suite running on pre-Aurora systems now
- ECP applications have demonstrated substantial GPU node (Sierra) vs. CTS-1 node speedups
  - SW4: 28X speedup on Sierra node (4 NVIDIA Volta GPUs) vs. CTS-1 CPU-only node (36 core Intel Xeon).
    - Hayward fault earthquake simulation of unprecedented resolution (26B grid points) runs in 10.3 hours on 256 nodes of Sierra (6% of machine). Same problem (grid size, run time) requires 8196 nodes of Cori-II (85% of machine)
  - GEOSX: 14X speedup on Sierra node vs. CTS-1 CPU-only node for explicit time integration solid mechanics model needed for ECP challenge problem

Exascale MPI (MPICH)

- PI: Pavan Balaji (ANL)
- MPICH has been a key influencer in the adoption of MPI
  - First or most comprehensive implementation of each new version of the MPI standard
  - Allows supercomputing centers to not compromise on what features they demand from vendors
- R&D 100 Award in 2005
- MPICH and its derivatives are the world's most widely used MPI implementations on large-scale supercomputers
- <u>MPICH will be the primary MPI implementation on Aurora, Frontier, and El Capitan (via Intel MPI and Cray MPI)</u>

Ongoing activities (September 2020)

- MPI Forum and standardization efforts
- Addressing ECP application-specific issues
- Efficient and transparent support for GPUs from multiple vendors (Intel, AMD, NVIDIA)
- New library for efficient noncontiguous data communication (MPI derived datatypes)



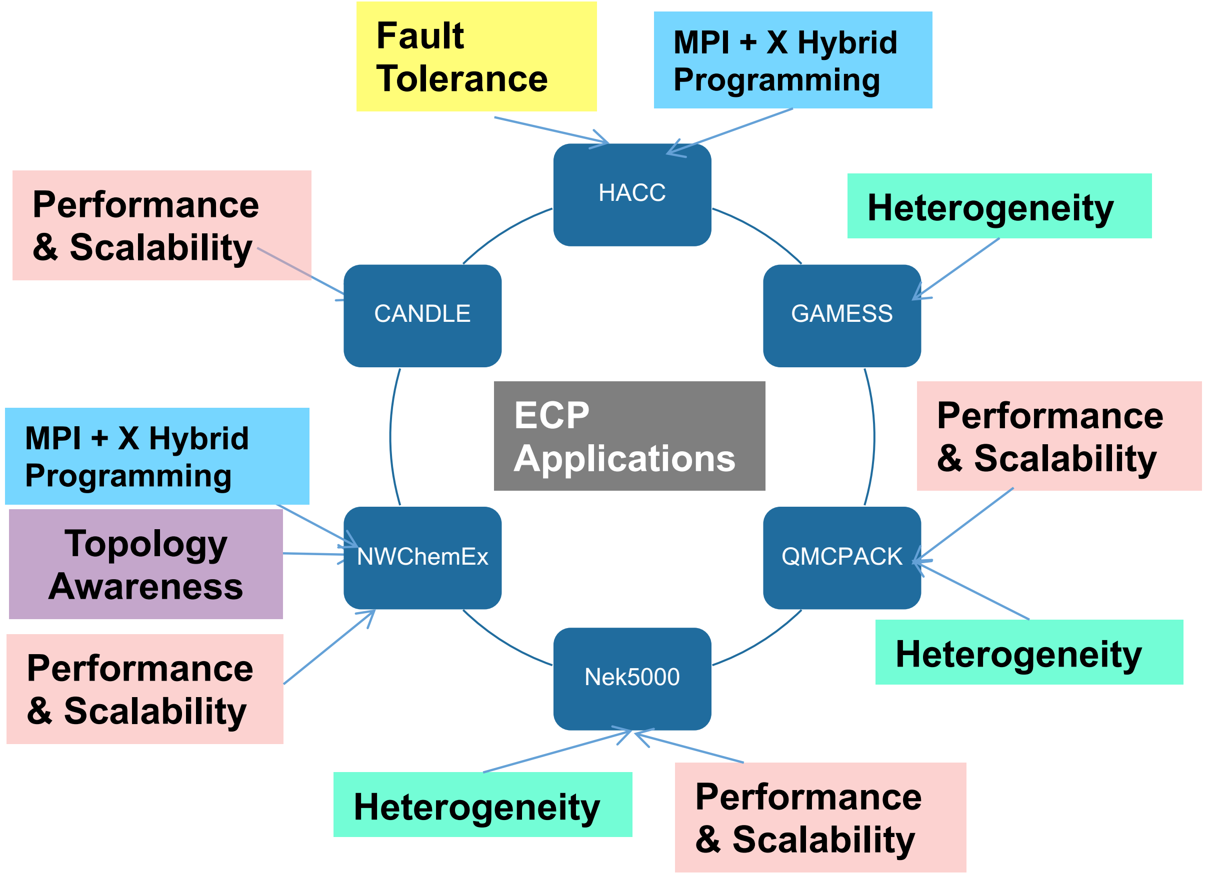
MPICH is not just a software

It's an Ecosystem

https://www.mpich.org/
https://github.com/pmodels/mpich

ECP Application Interactions, MPI Standardization Activities, and Vendor Collaboration

Interactions with ECP Applications
- Improvement on Key Technologies for ECP Applications
- Collaboration on Evaluation of Prototypes



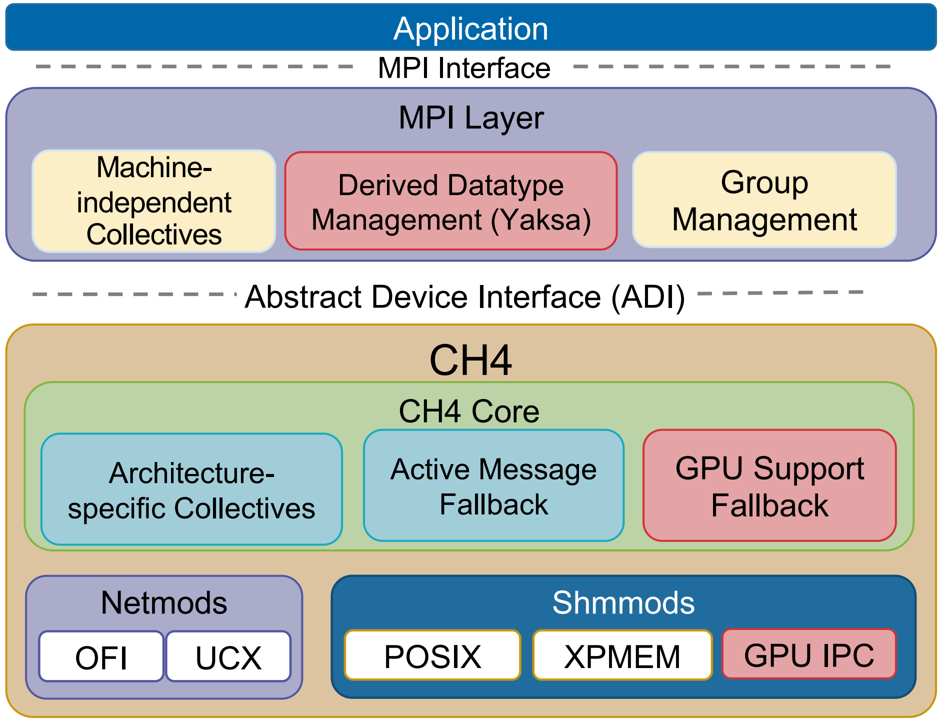
Vendor Collaborations
- Weekly telecon for project updates
- Deep dives with individual teams to discuss development priorities and plans
- Hackathons: Week-long accelerated development for priority topics

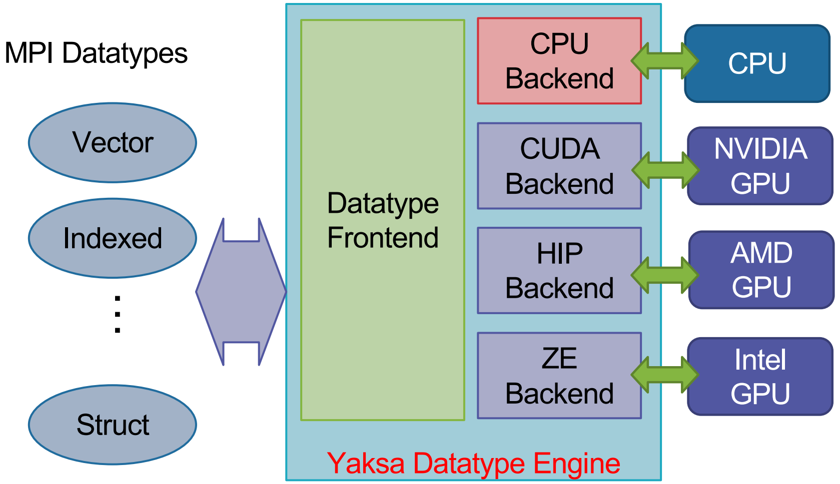
Standardization Efforts for MPI-4.0
- Leading the Hybrid Working Group
- Participating in other Working Groups
  - Point-to-point Communication
  - Fault Tolerance
  - Remote Memory Access
  - Hardware-Topologies
  - I/O

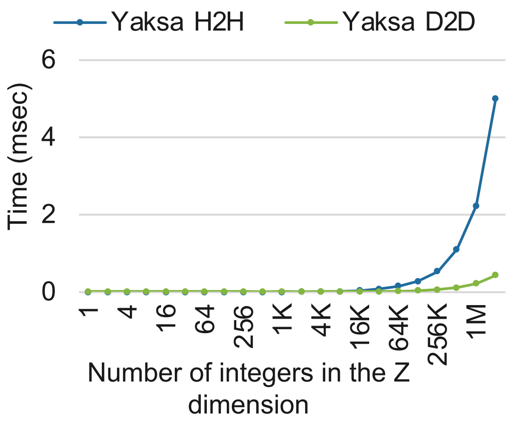
EXASCALE COMPUTING PROJECT

GPU Support in MPICH

- **Communication using GPU buffers**
  - Support native RDMA on GPU buffers through Libfabric (OFI) and UCX
  - Fallback for the case where RDMA is not available with GPU
  - Support GPU intranode communication through GPU IPC
  - Datatype handling with GPU buffers
- **Supporting noncontiguous datatypes with GPU**
  - Yaksa: A high performance datatype engine
  - Multiple backends provide datatype support for CPUs and GPUs from different vendors







Preliminary results for pack/unpack noncontiguous datatype GPU buffers to leverage high GPU memory bandwidth.

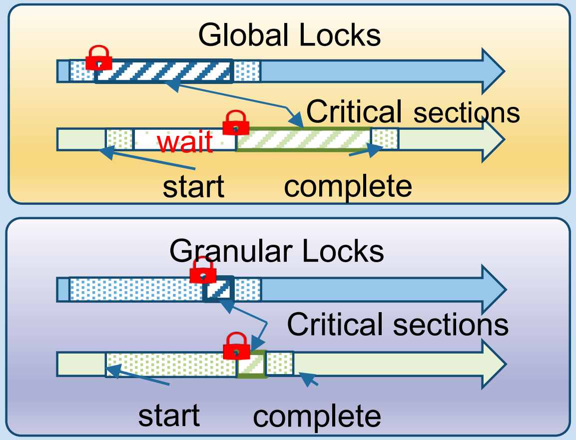
The GPU support in MPICH is developed in close collaboration with vendors (AMD, Cray, Intel, Mellanox, NVIDIA).

EXASCALE COMPUTING PROJECT

https://github.com/pmodels/yaksa

Improvements to MPI + Threads Support

Lock Optimization

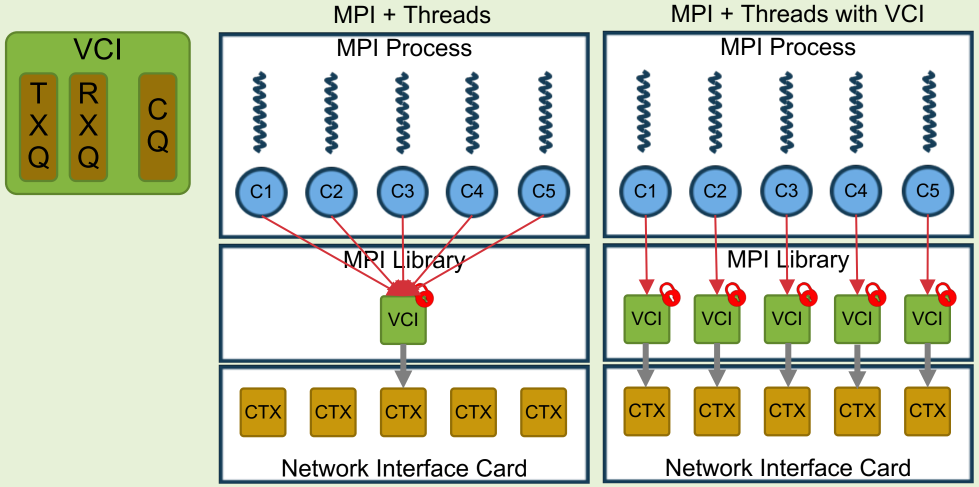


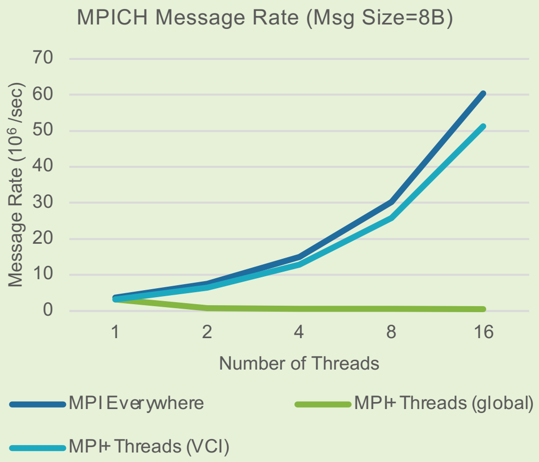
- Replace global lock with per-object locks
- Reduce lock scope

Argobots Integration

- MPICH works with Argobots, a lightweight high-performance user-level thread library.
- Increased opportunities for computation and communication overlapping
- Reduced thread synchronization

Virtual Communication Interface (VCI)





- VCIs are independent sets of communication resources in MPICH. They can be mapped to network hardware contexts.
- Having multiple VCIs allows the communication from different threads to be handled by different MPI resources, therefore reducing contention in the MPI library.
- It also allows multithreaded MPI applications to fully utilize network hardware contexts
- With reduce contention and full utilization of network hardware, MPI+Threads can achieve a performance close to MPI-only application which leads to significant improvement on strong scaling performance of MPI+Threads applications

Brief updates on other PMR Projects

PMR Software Development Kit (SDK)

- PI: Sameer Shende (UOregon)
- Developed draft set of community policies for PMR projects
- Integrated and released PMR products in a containerized distribution with support for GPUs (ROCm 3.x and CUDA 10.1). Available for download from https://e4s.io or DockerHub for Linux x86_64 and ppc64le.
- E4S Spack build cache for bare-metal installation has over 20,000 binaries [http://oaciss.uoregon.edu/e4s/inventory.html] with support for the latest release of Spack. E4S is used for Spack pull request (PR) merge validation testing [cdash.spack.io].
- E4S validation testing [https://github.com/E4S-Project/testsuite] planned for early access systems and includes LLVM support for Shasta Testing Project (STP)
- Examples of E4S build cache being used in ECP applications:
  - WDMapp: https://wdmapp.readthedocs.io/en/latest/machines/rhea.html
  - Pantheon: http://pantheonscience.org/projects/e4s
-

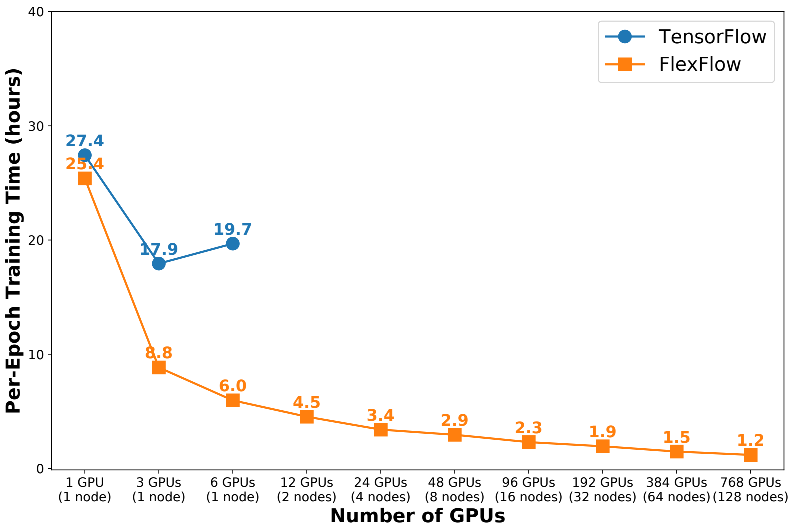
Open MPI (OMPI-X)

- PI: David Bernholdt (ORNL); Co-PIs: Howard Pritchard (LANL), Ignacio Laguna (LLNL), Ron Brightwell (SNL), George Bosilca (UTK)
- Enhance the MPI Standard and the Open MPI implementation for exascale
- Open MPI (via IBM Spectrum MPI) is the default implementation on Summit and Sierra
- Two new features championed by the group have been officially accepted for the upcoming MPI-4 Standard
  - Partitioned point-to-point communication
  - MPI sessions
- These features are already available in Open MPI for ECP teams to try out

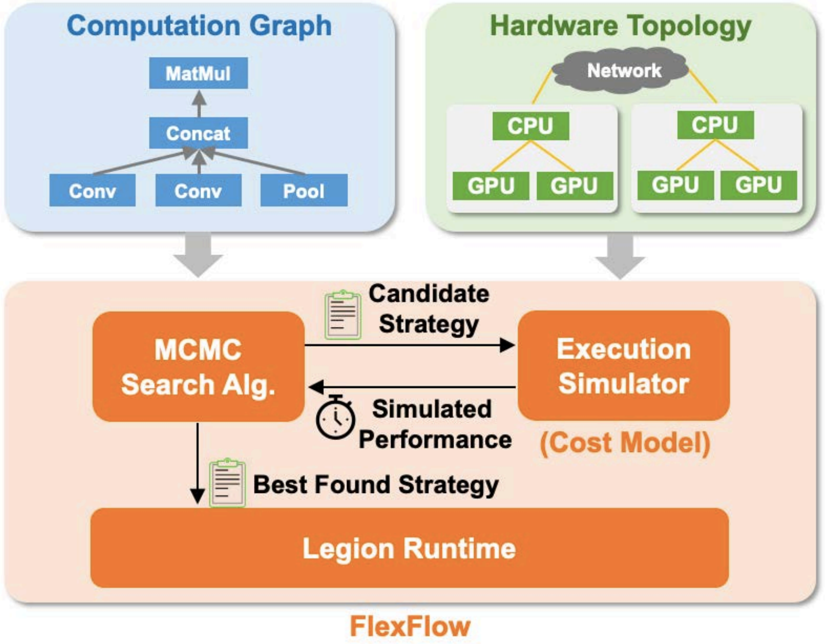
Legion & FlexFlow

- Task-based programming model
  - PI: Pat McCormick (LANL); Co-PIs: Alex Aiken (Stanford), Pavan Balaji (ANL)
  - Used in ExaFEL, S3D, CANDLE, NNSA PSAAP II & III, ...
  - Ports to exascale architectures in progress

- FlexFlow is a distributed deep learning framework
  - Built on Legion
  - Exploits Legion's first-class data partitioning & distributed execution
  - Secret sauce: Automatic search to find a high-performance partitioning
  - Dramatically improves locality and scalability
  - Reduces CANDLE run times from days to hours
  - Portability interfaces for Keras and PyTorch in progress



The FlexFlow system is based on Legion and, in comparison to Google's TensorFlow, can scale to 768 GPUs and reduces the per-epoch training time from 18 hours to 1.2 hours for CANDLE's Uno benchmark.



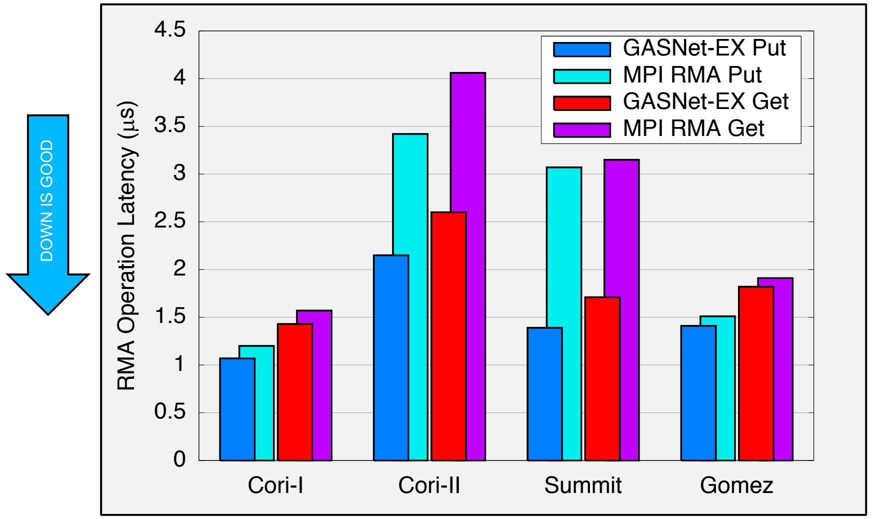
Argo

- PI: Pete Beckman (ANL); Co-PIs: Tapasya Patki and Maya Gokhale (LLNL)
- **Deep Memory:** Enable applications to improve usage of new memory types
  - AML: Application-aware management of deep memory systems
    - New: Enhanced memory topology API for complex multi-CPU, multi-GPU exascale nodes; ongoing integration effort with OpenMC
  - UMap: User-level mapping of NVRAM/SSD into the memory hierarchy
    - New: UMap handler can access memory from other nodes, e.g., a dedicated memory server or data producer nodes
- **Power Management:** Dynamically manage power to improve energy usage and performance
  - PowerStack: Global management through power-aware job scheduling
    - New: Variorum + GEOPM integration, initial power management support for Kokkos tools
  - NRM: Node-level power management infrastructure
    - New: ML-based power management controller
- **Co-Design:** Working with vendors, improve interfaces to exploit new hardware capabilities in HPC and ML
- **Platform Readiness**
  - NRM runs on Theta; UMap runs on Sierra, AMD systems; PowerStack is deployed in TOSS; ongoing work on using ECP-CI to test AML on Theta and Summit

Pagoda (UPC++, GASNet-EX)

- PI: Paul Hargrove (LBNL)
- GASNet-EX: Portable high-performance networking layer for PGAS runtimes
- UPC++: C++ template library providing asynchronous one-sided RMA and RPC for rapid development of PGAS applications, implemented using GASNet-EX
- ECP use cases: ExaBiome, ExaGraph, NWChemEx, Legion
- Recent Activities
  - Semi-annual software releases of UPC++, GASNet-EX
  - Optimizations for GPUs in progress (efficient GPU-GPU memory transfers)
  - ECP training event for UPC++ in May
  - SC20 tutorial accepted

8-Byte RMA Operation Latency (one-at-a-time)



EXASCALE COMPUTING PROJECT

2.3.2 Development Tools (and Compilers)

Jeffrey Vetter, Oak Ridge National Laboratory

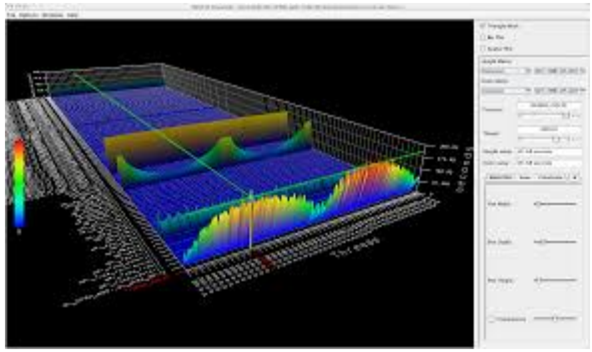
ASCAC Meeting
25 Sep 2020

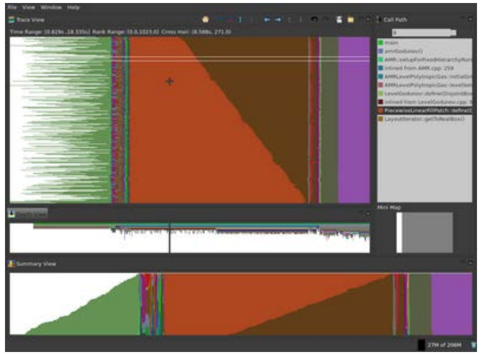
WBS 2.3.2 Development Tools: Context for the portfolio

Vision	Enable exascale computing by providing architecture-ready compilers and programming tools	
Challenges	Complex and heterogeneous node architectures with immature software systems; exact system architectures not known until early systems arrive; diverse application requirements	
Mission	Develop compilers and programming tools that enable applications teams to achieve high performance while providing basic portability across diverse exascale architectures	
Objective	Produce production-ready compilers and programming tools, tuned for application needs, deployed at facilities via vendors or as part of SDK	
Starting Point	Existing heterogeneous architecture programming models including CUDA, OpenACC, OpenMP, OpenCL, and various research efforts	
Portfolio Goals	**Compilers**	Provide OpenMP, OpenACC, Fortran, and other architectural specific capabilities via the LLVM ecosystem
	Prog Tools	Deliver development tools for performance and correctness: TAU, HPCToolkit, PAPI, Autotuning,
	Memory	Deliver libraries that provide portable abstractions for managing deep memory hierarchies including nonvolatile memory like Optane DIMMs.

2.3.2 Development Tools Portfolio

Project Short Name	PI Name, Inst	Short Description/Objective
SDK	Miller, UW Madison	SDK: System-facing delivery of software products, APIs, and vendor integration
EXAPAPI	Dongarra, UTK	EXAPAPI: The Exascale Performance Application Programming Interface
HPCToolkit	Mellor-Crummy, Rice	Extending HPCToolkit to Measure and Analyze Code Performance on Exascale Platforms
PROTEAS-TUNE	Vetter, ORNL	PROTEAS-TUNE: Programming, Autotuning, and Optimization Toolchain for Emerging Architectures and Systems
SOLLVE	Chapman, Stony Brook/BNL	SOLLVE: OpenMP for LLVM
Flang	McCormick, LANL	Flang: Fortran for LLVM


Tau


HPCToolkit

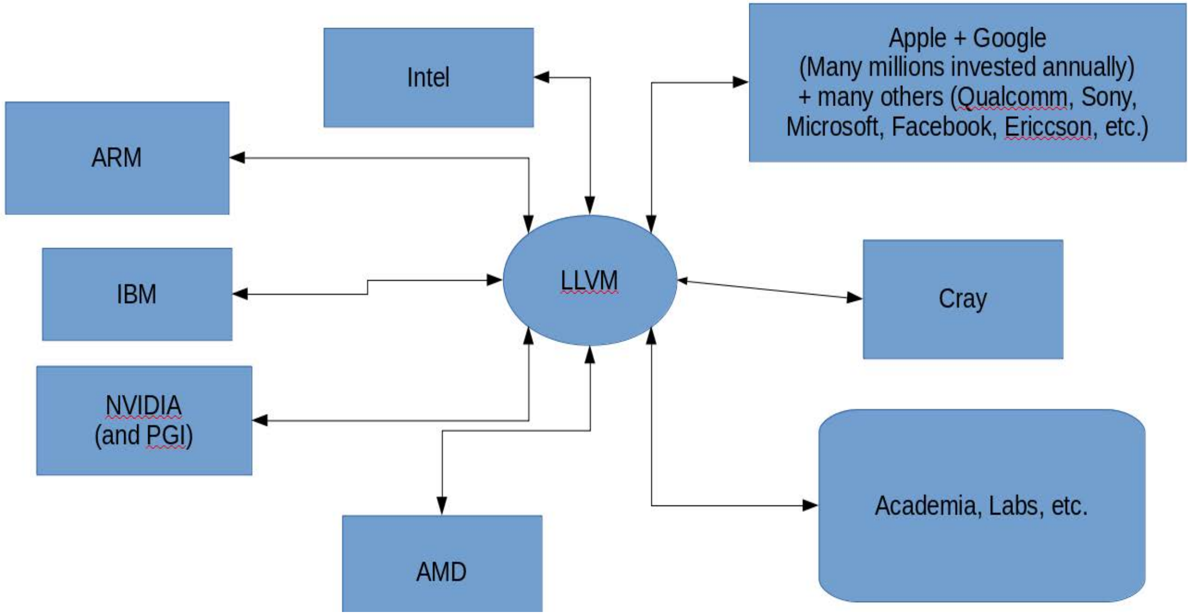
Development Tools: LLVM Ecosystem

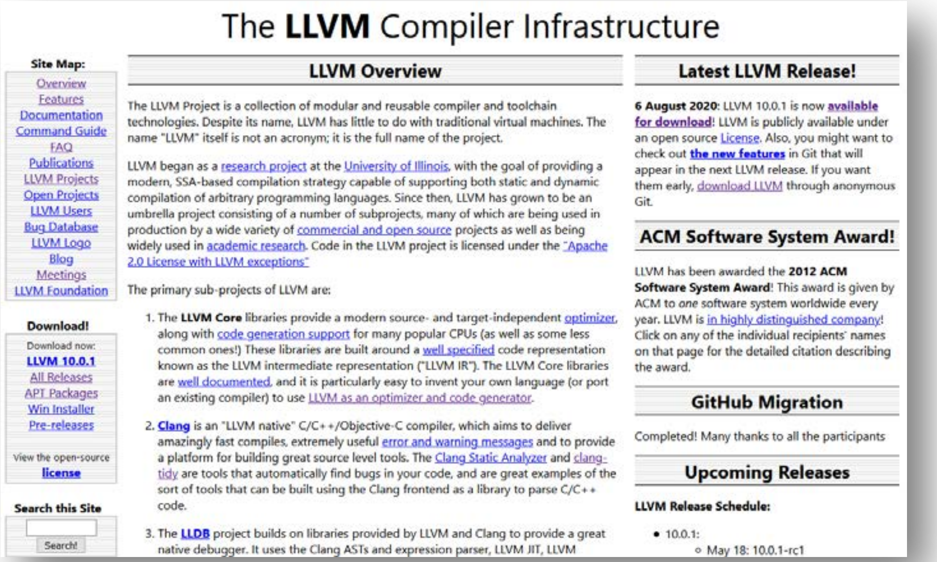
Deep Dive

LLVM is an infrastructure for creating compilers

https://llvm.org/

- Features: LLVM has become well known for an important set of features:
  - LLVM is a liberally-licensed(*) infrastructure for creating compilers, other toolchain components, and JIT compilation engines.
  - A modular, well-defined IR allows use by a lot of different languages (C, C++, Fortran, Julia, Rust, Python (e.g., via Numba), Swift, ML frameworks (e.g., TensorFlow/XLA, PyTorch/Glow), and many others.
  - A backend infrastructure allowing the efficient creation of backends for new (heterogeneous) hardware.
  - A state-of-the-art C++ frontend, CUDA support, scalable LTO, sanitizers and other debugging capabilities, and more.
  - High code-quality community standards and review process





The **LLVM** Compiler Infrastructure

Site Map:
Overview
Features
Documentation
Command Guide
FAQ
Publications
LLVM Projects
Open Projects
LLVM Users
Bug Database
LLVM Logo
Blog
Meetings
LLVM Foundation

Download!
Download now:
LLVM 10.0.1
All Releases
APT Packages
Win Installer
Pre-releases

View the open-source
license

Search this Site
Search!

LLVM Overview

The LLVM Project is a collection of modular and reusable compiler and toolchain technologies. Despite its name, LLVM has little to do with traditional virtual machines. The name "LLVM" itself is not an acronym; it is the full name of the project.

LLVM began as a research project at the University of Illinois, with the goal of providing a modern, SSA-based compilation strategy capable of supporting both static and dynamic compilation of arbitrary programming languages. Since then, LLVM has grown to be an umbrella project consisting of a number of subprojects, many of which are being used in production by a wide variety of commercial and open source projects as well as being widely used in academic research. Code in the LLVM project is licensed under the "Apache 2.0 License with LLVM exceptions"

The primary sub-projects of LLVM are:

1. The **LLVM Core** libraries provide a modern source- and target-independent optimizer, along with code generation support for many popular CPUs (as well as some less common ones!) These libraries are built around a well specified code representation known as the LLVM intermediate representation ("LLVM IR"). The LLVM Core libraries are well documented, and it is particularly easy to invent your own language (or port an existing compiler) to use LLVM as an optimizer and code generator.
2. **Clang** is an "LLVM native" C/C++/Objective-C compiler, which aims to deliver amazingly fast compiles, extremely useful error and warning messages and to provide a platform for building great source level tools. The Clang Static Analyzer and clang-tidy are tools that automatically find bugs in your code, and are great examples of the sort of tools that can be built using the Clang frontend as a library to parse C/C++ code.
3. The **LLDB** project builds on libraries provided by LLVM and Clang to provide a great native debugger. It uses the Clang ASTs and expression parser, LLVM JIT, LLVM

Latest LLVM Release!

6 August 2020: LLVM 10.0.1 is now **available for download**! LLVM is publicly available under an open source License. Also, you might want to check out **the new features** in Git that will appear in the next LLVM release. If you want them early, download LLVM through anonymous Git.

ACM Software System Award!

LLVM has been awarded the **2012 ACM Software System Award**! This award is given by ACM to *one* software system worldwide every year. LLVM is in highly distinguished company! Click on any of the individual recipients' names on that page for the detailed citation describing the award.

GitHub Migration

Completed! Many thanks to all the participants

Upcoming Releases

LLVM Release Schedule:

- 10.0.1:
  - May 18: 10.0.1-rc1

Slide courtesy of Hal Finkel

Next week’s LLVM Dev Meeting Brings Together (virtually) over 500+ Developers from around the World including ECP Staff

https://llvm.org/devmtg/2020-09/

- The Present and Future of Interprocedural Optimization in LLVM - J. Doerfert; B. Homerding; ...
- Proposal for A Framework for More Effective Loop Optimizations - M. Kruse; H. Finkel
- Changing Everything With Clang Plugins: A Story About Syntax Extensions, Clang’s AST, and Quantum Computing = H. Finkel; A. Mccaskey
- (OpenMP) Parallelism-Aware Optimizations - J. Doerfert; S. Stipanovic; H. Mosquera; J. Chesterfield; G. Georgakoudis; J. Huber
- A Deep Dive into the Interprocedural Optimization Infrastructure - J. Doerfert; B. Homerding; S. Baziotis; S. Stipanovic; H. Ueno; K. Dinel; S. Okumura; L. Chen
- From Implicit Pass Dependencies to Effectiveness Prediction - H. Ueno; J. Doerfert; E. Park; G. Georgakoudis; T. Jayatilaka; S. Badruswamy
- OpenACC support in Flang with a MLIR dialect - V. Clement; J. Vetter
- Flang Update - S. Scalpone
- Code Feature Analysis, Tracking, and Future Usage - T. Jayatilaka; J. Doerfert; G. Georgakoudis; E. Park; H. Ueno; S. Badruswamy
- Loop Optimization BoF - M. Kruse; K. Barton

ECP is Improving the LLVM Compiler Ecosystem

LLVM

- Very popular open source compiler infrastructure
- Easily extensible
- Widely used and contributed to in industry
- Permissive license
- Used for heterogeneous computing

+SOLLVE

- Enhancing the implementation of OpenMP in LLVM
- Unified memory
- OMP Optimizations
- Prototype OMP features for LLVM
- OMP test suite
- Tracking OMP implementation quality

+PROTEAS-TUNE

- Core optimization improvements to LLVM
- OpenACC capability for LLVM
  - Clacc
  - Flacc
- Autotuning for OpenACC and OpenMP in LLVM
- Integration with Tau performance tools

+FLANG

- Developing an open-source, production Fortran frontend
- Upstream to LLVM public release
- Support for OpenMP and OpenACC
- Recently approved by LLVM

+HPCToolkit

- Improvements to OpenMP profiling interface OMPT
- OMPT specification improvements
- Refine HPCT for OMPT improvements

+ATDM

- Enhancing LLVM to optimize template expansion for FlexCSI, Kokkos, RAJA, etc.
- Flang testing and evaluation

Vendors

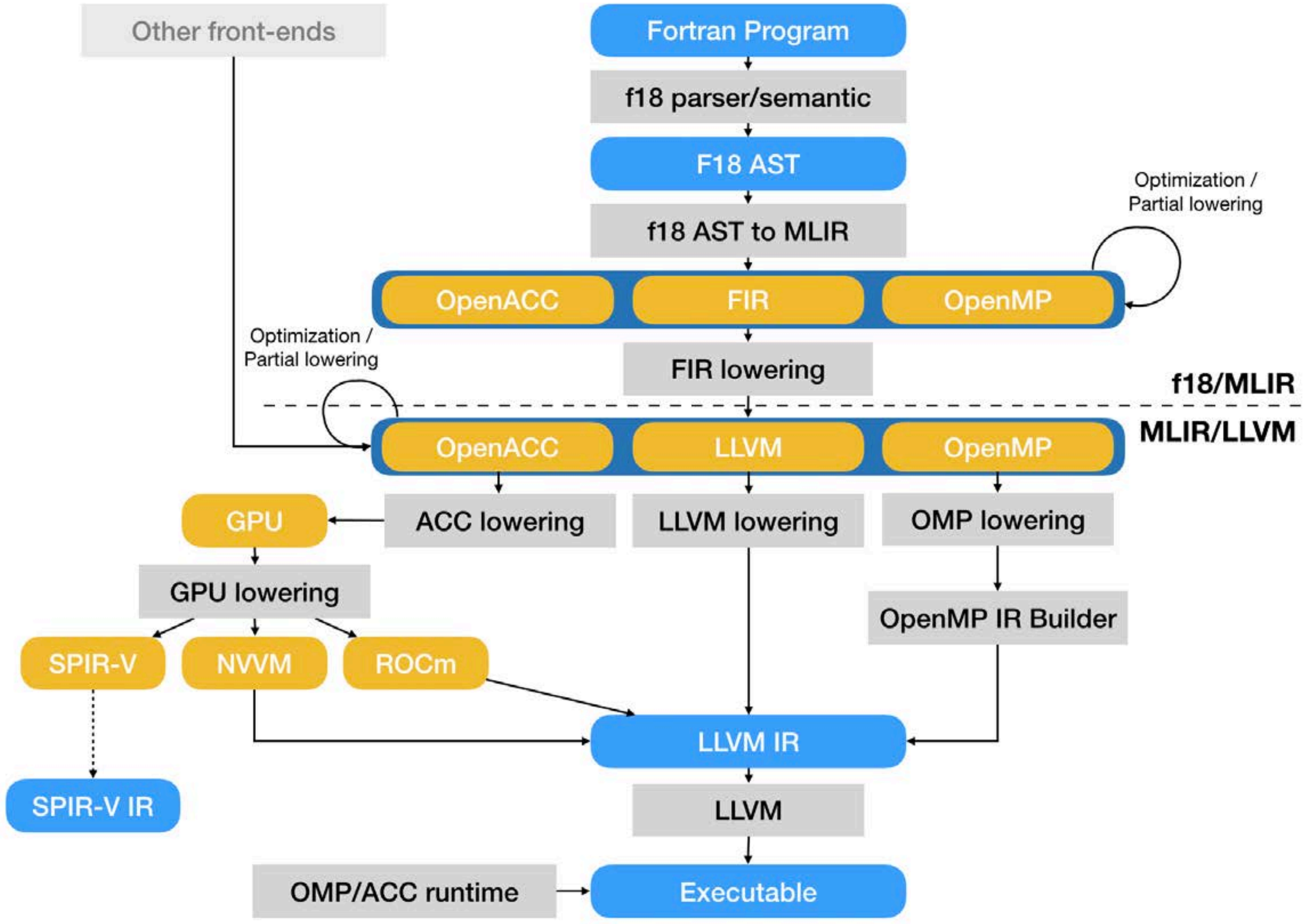
- Increasing dependence on LLVM
- Collaborations with many vendors using LLVM
  - AMD
  - ARM
  - Cray
  - HPE
  - IBM
  - Intel
  - NVIDIA

Active involvement with broad LLVM community: LLVM Dev, EuroLLVM

ECP Projects: Flang, SOLLVE, PROTEAS-TUNE
Many other contributors: NNSA, NVIDIA, ARM, Google, …

Leveraging LLVM Ecosystem to Meet a Critical ECP (community) need : FORTRAN

- Fortran support continues to be an ongoing requirement
- Flang project started in NNSA funding NVIDIA/PGI to open source compiler front-end into LLVM ecosystem
- SOLLVE is improving OpenMP dialect, implementation, and core optimizations
- PROTEAS-TUNE is creating OpenACC dialect and improving MLIR
- ECP projects are contributing many changes upstream to LLVM core, MLIR, etc
- Many others are contributing: backends for processors, optimizations in toolchain, …
  - Google contributed MLIR

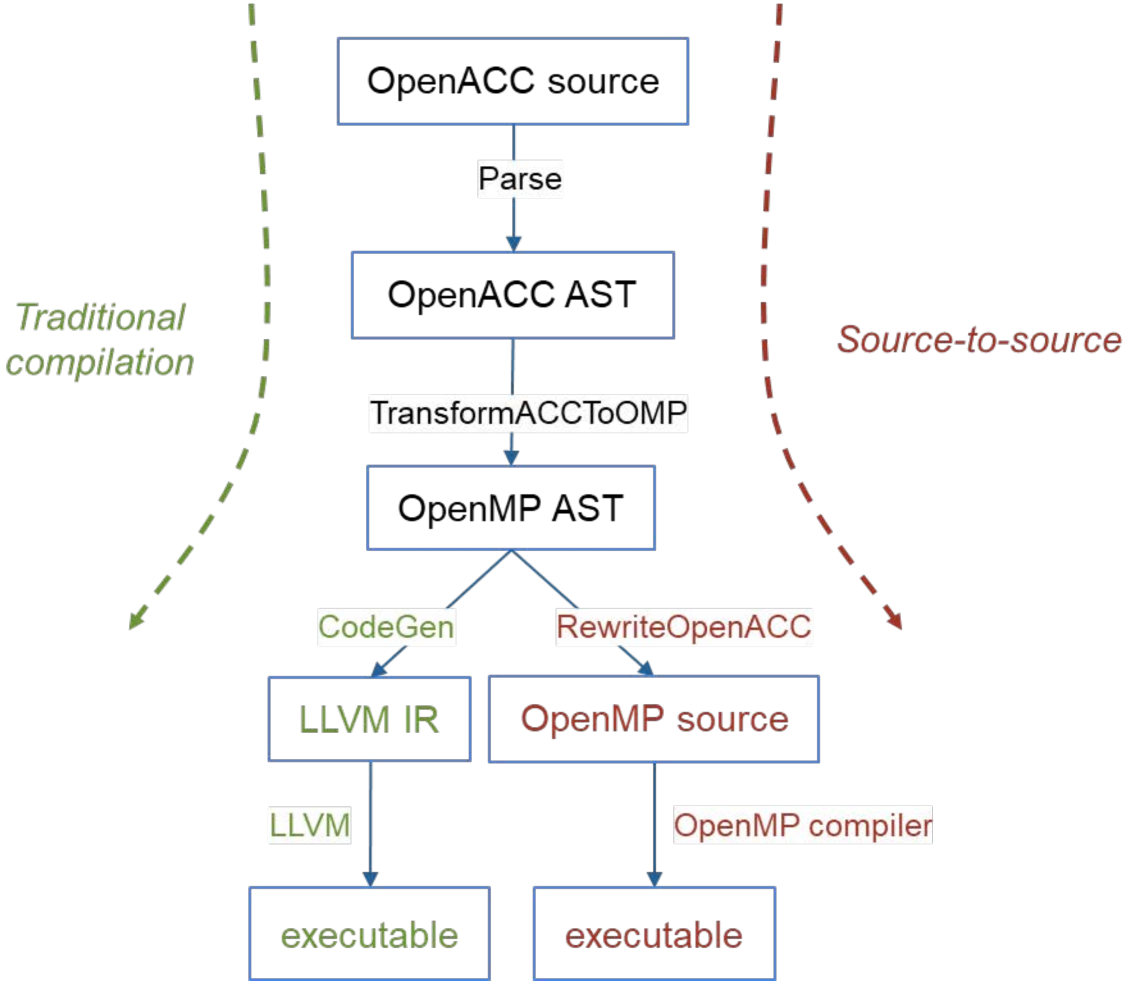


EXASCALE COMPUTING PROJECT

PROTEAS-TUNE: Clacc – OpenACC in Clang/LLVM

2.3.2.10	PROTEAS-TUNE
POC	Joel Denny, ORNL

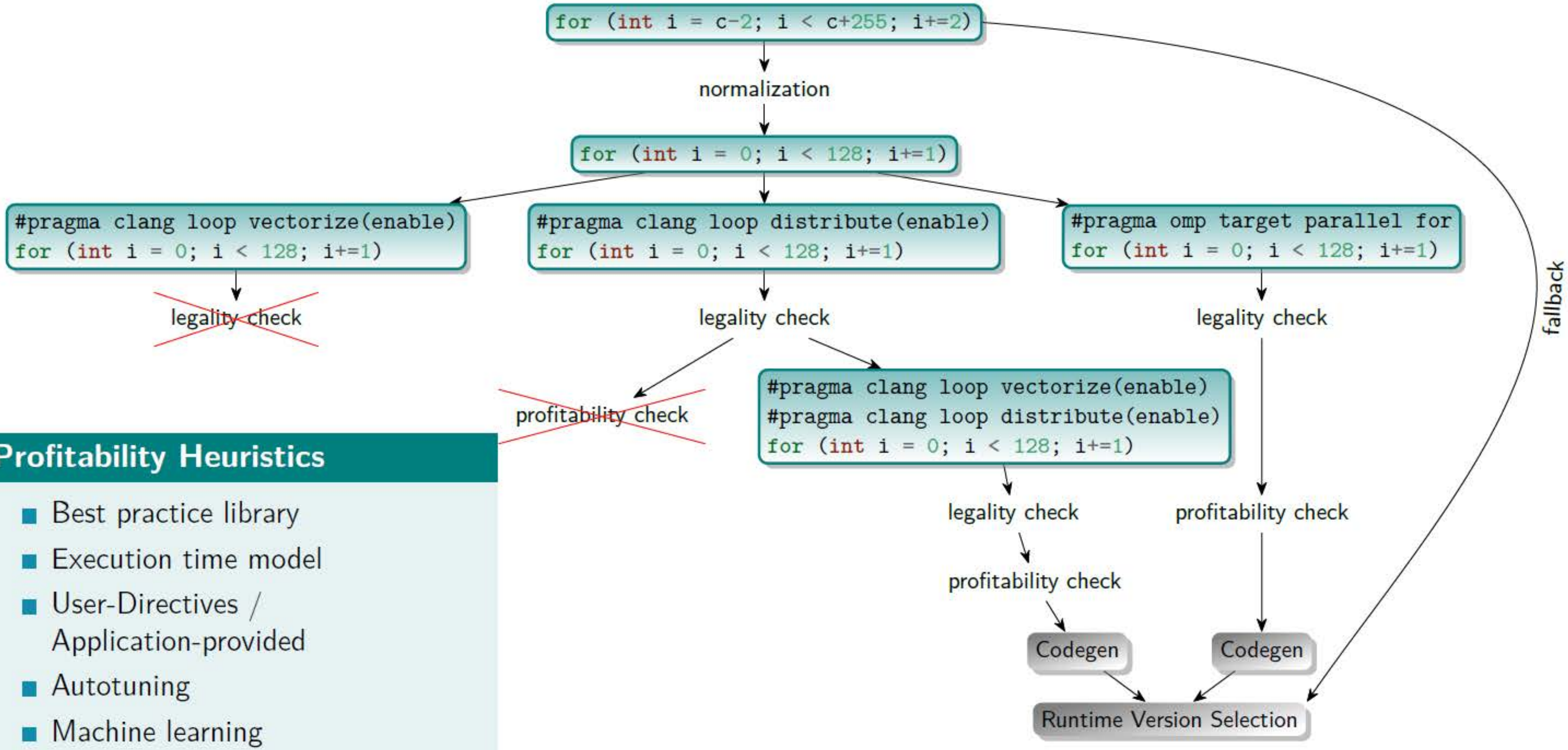
- Develop production-quality, standard-conforming traditional OpenACC compiler and runtime support by extending Clang and LLVM
  - Build on existing OpenMP infrastructure
- Enable research and development of source-level OpenACC tools
  - Design compiler to leverage Clang/LLVM ecosystem extensibility
  - E.g., Pretty printers, analyzers, lint tools, and debugger and editor extensions
- Actively contribute improvements to the OpenACC specification
- Actively contribute upstream all Clang and LLVM improvements that are mutually beneficial
  - Many contributions are already in LLVM
- Open-source with multiple collaborators (vendors, universities)



Clacc: Translating OpenACC to OpenMP in Clang, Joel E. Denny, Seyong Lee, and Jeffrey S. Vetter, 2018 IEEE/ACM 5th Workshop on the LLVM Compiler Infrastructure in HPC (LLVM-HPC), Dallas, TX, USA, (2018).

2.3.2.11	SOLLVE
POC	Michael Kruse, ANL

Speculative Loop Transformation Representation

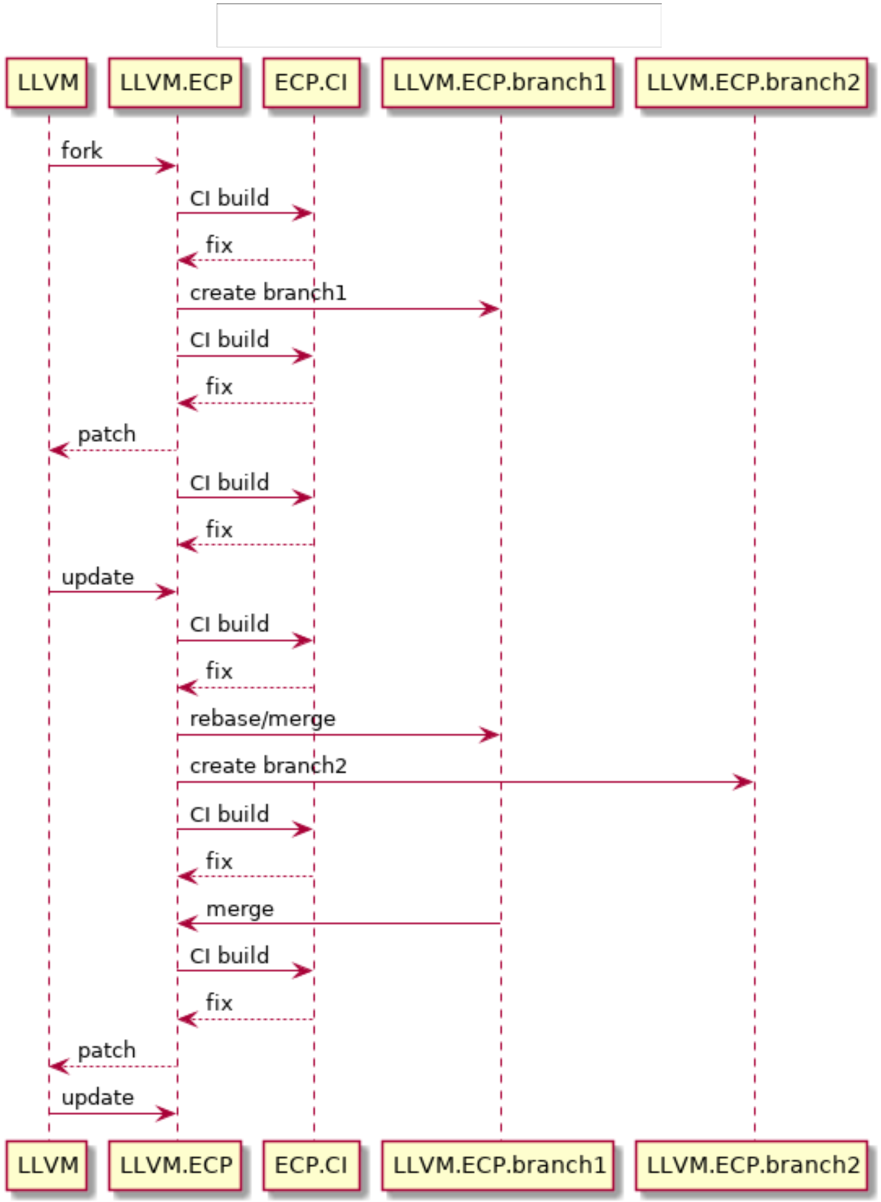


Profitability Heuristics

- Best practice library
- Execution time model
- User-Directives / Application-provided
- Autotuning
- Machine learning

ECP LLVM Integration and Deployment

https://github.com/llvm-doe-org/llvm-project



- Develop an integrated ECP LLVM distribution
  - Integrating different ECP projects using LLVM
  - CI on target architectures
  - Shared vehicle for improvements in LLVM
  - Increased collaboration within ECP
- Operations
  - ECP LLVM distro will be closely maintained fork of LLVM mono repo
  - Multiple branches of individual ECP projects will exist at git branches
  - Branches will be integrated into ECP LLVM
- Periodic upstreaming and patching of LLVM monorepo

EXASCALE COMPUTING PROJECT

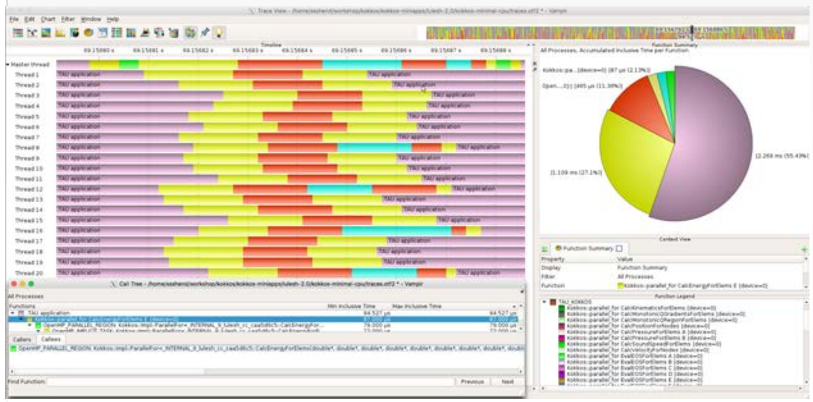
Questions?

Bonus Slides

Example Integration Activities

PROTEAS/Kokkos

- Performance profiling of Kokkos applications
- Tau and Kokkos team collaborating on initial interface to allow transparent performance profiling
- Impact on application performance and user productivity



ExaPAPI/NWChemEx

- Co-design a prototype implementation of a PAPI component for NWChemEx-related software-defined events (SDE).
- Development of a proof-of-concept PAPI component that supports the performance metrics the NWChemEx team requires

PAPI-NWChemEx Performance Counter	Counter Description
sde:::NWCHEM::t28_chain_cnt	Total Number of chains with sequential DGEMMs in CCSD kernel t2_8()
sde:::NWCHEM::t28_max_chain_length	Maximum number of sequential DGEMMs per chain in CCSD kernel t2_8()
sde:::NWCHEM::t28_dgemms_cnt	Total number of DGEMMs in CCSD kernel t2_8()
sde:::NWCHEM::t28_flops_cnt	Total number of floating-point operations in CCSD kernel t2_8()

Flang//LLVM Community

- Developing an open source, production quality Fortran front-end for LLVM
- Work in progress for several years including rewriting major portions of front-end in order to merge successfully with LLVM design and standards
- Received approval from LLVM leadership for upstreaming of final Flang implementation

LLVM COMPILER INFRASTRUCTURE

HPCToolkit: Analysis of Data Movement

2.3.2.08	Enhancing HPCToolkit
PI	J. Mellor-Crummey, Rice.
Members	Rice., U. of Wisconsin-Madison

Scope and objectives

- Monitor the costs of data movement.
- Identify latency and bandwidth bound code regions and quantify data movement associated with individual variables.
- Support accurate, low-overhead measurement of a broad range of parallel applications.

Pinpoint and quantify data movement

HPCToolkit utilizes the additional information from Intel PEBS and IBM Marked events to directly measure memory access latency to both variables and instructions.

Type to enter a caption.

Impact

- HPCToolkit is installed on many DOE systems.
- HPCToolkit supports low overhead measurement and analysis of performance for a wide range of applications.
- Application, library and tool developers can use HPCToolkit to analyze the performance of their software.

Project accomplishment

- Added a feature in HPCToolkit to monitor both dynamic allocation of memory and static variables in a load module or executable.
- Added a feature in the HPCToolkit user interface to present the cost of data movement for important variables.

HPCToolkit's Github at https://github.com/HPCToolkit/hpctoolkit/tree/datacentric

EXASCALE COMPUTING PROJECT

PaRSEC

- PIs: Jack Dongarra and George Bosilca (UTK)
- Data-centric programming environment based on asynchronous tasks executing on a heterogeneous distributed environment
- ECP use cases: SLATE, NWChemEx, GAMESS
- Recent Activities
  - Redesigning the communication infrastructure to support one-sided communication
  - New Domain Specific Languages targeting developer productivity and facilitating the expression of parallel/distributed algorithmic constructs (including collective behaviors)
  - Initial support for ARM processors (atomic operations, gather/scatter, SVE)
  - Improved interoperability with DPLASMA, ScaLAPACK, and SLATE
    - Automatic support for heterogeneous environments
  - Heterogeneous-memory-aware algorithms and integration with MPQC (NWChemEx)
  - Large scale performance evaluation on all pre Exascale platforms

Simplified Interface to Complex Memory (SICM)

- PI: Mike Lang (LANL); Co-PIs: Terry Jones (ORNL), Maya Gokhale (LANL), Michael Jantz (UTK), Frank Mueller (NC State)
- Portable low-level interface targeting Intel KNL, IBM Power9+Volta, Intel Optane DC PMM, Frontier, and Aurora
- High-level interface that makes reasonable decisions for applications
- Meta allocator for persistent memory focusing on graph applications
- ECP use cases: SOLLVE (OpenMP), OMPI-X, Umpire, Kokkos, NaluWind, Trilinos
- Recent Activities
  - Integration as an allocator in CLANG/OpenMP
  - Integration into Kokkos
  - Integration into NaluWind

Concluding Remarks

- ECP Programming Models and Runtimes projects are actively working on performance tuning for exascale platforms and working closely with applications, vendors, and facilities
- There isn't one single intranode programming model for meeting the performance portability needs of applications
- Instead, a variety of models are available, and applications are using them
- At the end of ECP, we will have a lot of useful data on what works well (and what doesn't), which will help achieve convergence