E4S: An Introduction to the Extreme-scale Scientific Software Stack

Michael A. Heroux
Director of Software Technology, US Exascale Computing Project
Senior Scientist, Center for Computing Research, Sandia National Laboratories
Scientist in Residence, St. John's University, MN

September 22, 2019

National Nuclear Security Administration
U.S. DEPARTMENT OF ENERGY | Office of Science

E4S Forum Agenda

09:00am-10:00am **E4S: An Introduction to the Extreme-scale Scientific Software Stack**
Michael Heroux, Sandia National Laboratories

10:00am-10:30am **Supercontainers for HPC**
Andrew J. Younge, Sandia National Laboratories

10:30am-11:00am **Break**

11:00am-11:30am **OpenHPC Update and Future Contributions**
Robert Wisniewski, Intel

11:30am-12:00pm **An Overview of the ECP Software Ecosystem**
Todd Munson, Argonne National Laboratory

12:00pm-12:30pm **Hartree Centre Experience in Extreme Scale Computing Addressing Industrial, Societal and Scientific Challenges**
Vassil Alexandrov, Hartree Centre-STFC

12:30pm-02:00pm **Break**

02:00pm-02:30pm **Experiences with Extreme-scale Scientific Software Stack (E4S) and Spack on SDSC systems**
Mahidhar Tatineni, SDSC

02:30pm-03:00pm **Kokkos: Performance Portability for C++ Applications**
Christian Trott, Sandia National Laboratories

03:00pm-03:30pm **xSDK: a Community of Diverse Numerical HPC Software Packages**
Ulrike Meier Yang, Lawrence Livermore National Laboratory

03:30pm-04:00pm **Break**

04:00pm-04:30pm **Toward interoperable and flexible scientific computing libraries: Lessons learned from xSDK**
Keita Teranishi, Sandia National Laboratories

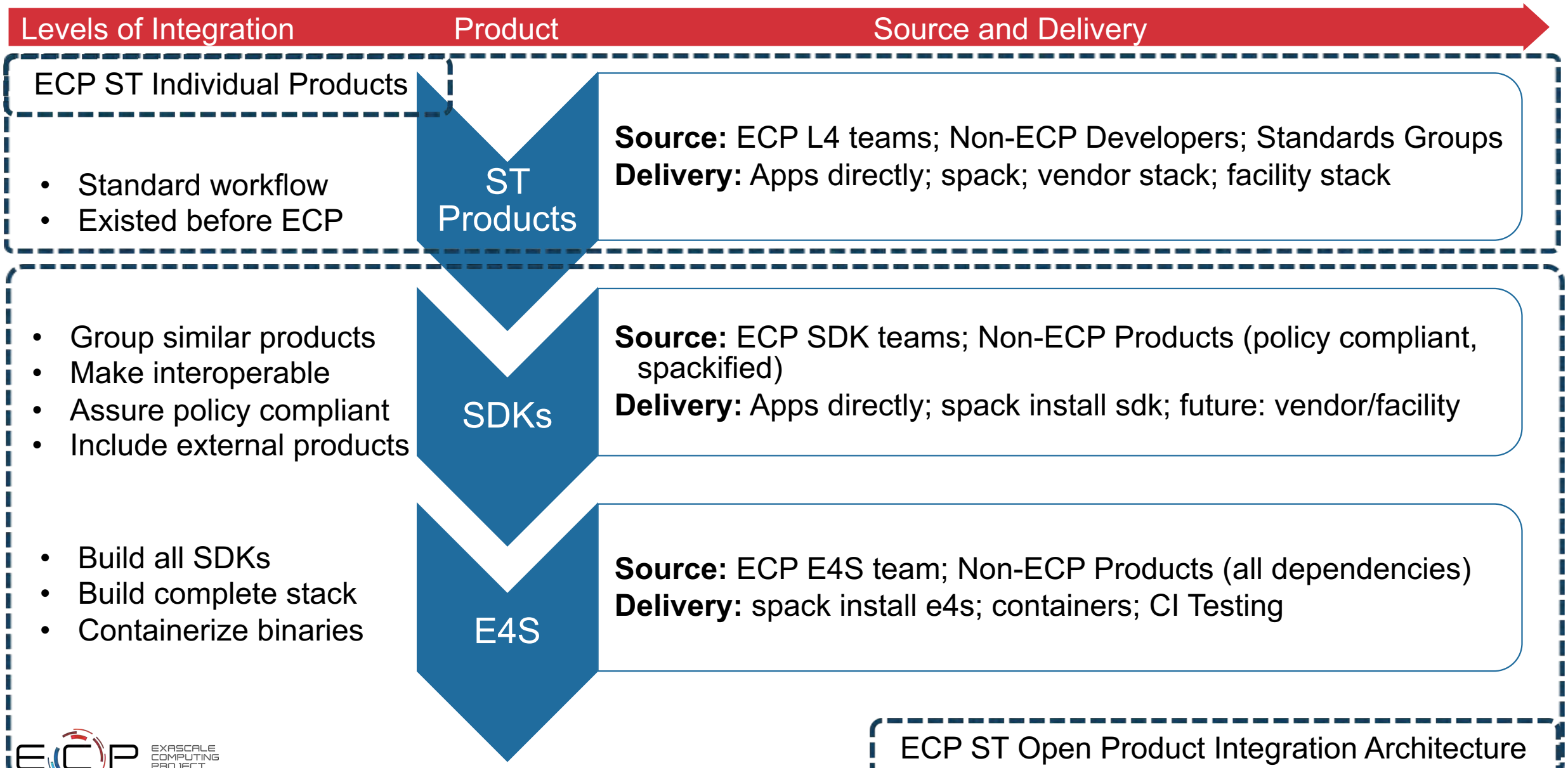
04:30pm-05:00pm **Data Management and Visualization Approaches for the U.S. Exascale Program**
James Ahrens, Los Alamos National Laboratory

05:00pm-05:30pm **Leveraging and Expanding the Capabilities of the LLVM Compiler Infrastructure for Exascale Computing**
Patrick McCormick, Los Alamos National Laboratory

05:30pm-06:00pm **Experiences in Designing, Developing, Packaging, and Deploying the MVAPICH2 Libraries**
Hari Subramoni, Ohio State University

E4S Overview

Software Technology Ecosystem



Extreme-scale Scientific Software Stack (E4S)

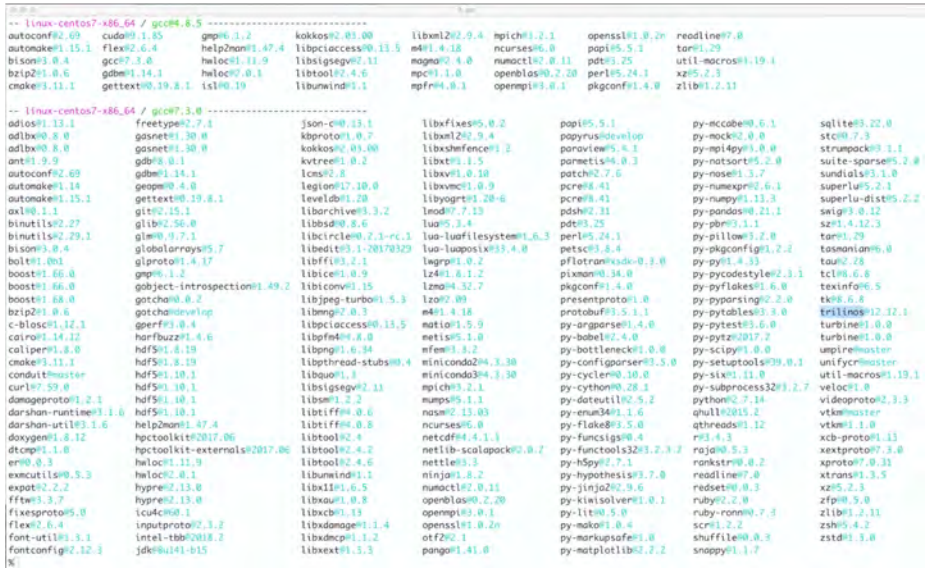
- E4S: A Spack-based distribution of ECP ST and related and dependent software tested for interoperability and portability to multiple architectures
- Provides distinction between SDK usability / general quality / community and deployment / testing goals
- Will leverage and enhance SDK interoperability thrust
- Oct: E4S 0.1 - 24 full, 24 partial release products
- Jan: E4S 0.2 - 37 full, 10 partial release products

e4s.io

Lead: Sameer Shende (U Oregon)

Sample of E4S Release and Installed Packages

- Adios
- Bolt
- Caliper
- Darshan
- Gasnet
- GEOPM
- GlobalArrays
- Gotcha
- HDF5
- HPCToolkit
- Hypre
- Jupyter
- Kokkos
- Legion
- Libquo
- Magma
- MFEM
- MPICH
- OpenMPI
- PAPI
- Papyrus
- Parallel netCDF
- ParaView
- PETSc/TAO
- Program Database Toolkit (PDT)
- Qthreads
- Raja
- SCR
- Spack
- Strumpack
- Sundials
- SuperLU
- Swift/T
- SZ
- Tasmanian
- TAU
- Trilinos
- VTKm
- Umpire



Packages installed using Spack

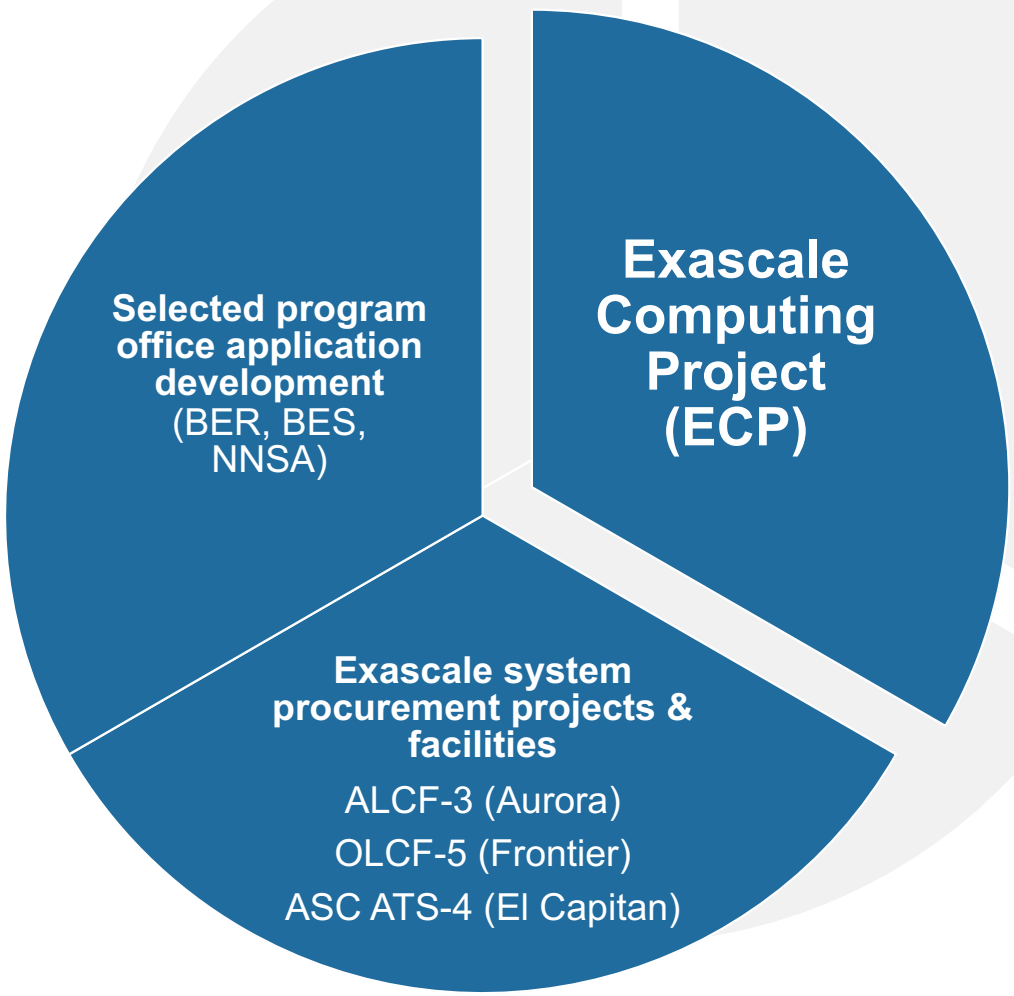
- UnifyCR
- Veloc
- xSDK
- Zfp

All ST products will be released through E4S

Exascale Computing Project (ECP) Research, Development & Delivery (RD&D) Overview

DOE Exascale Program: The Exascale Computing Initiative (ECI)

Three Major Components of the ECI



ECI partners

US DOE Office of Science (SC) and National Nuclear Security Administration (NNSA)

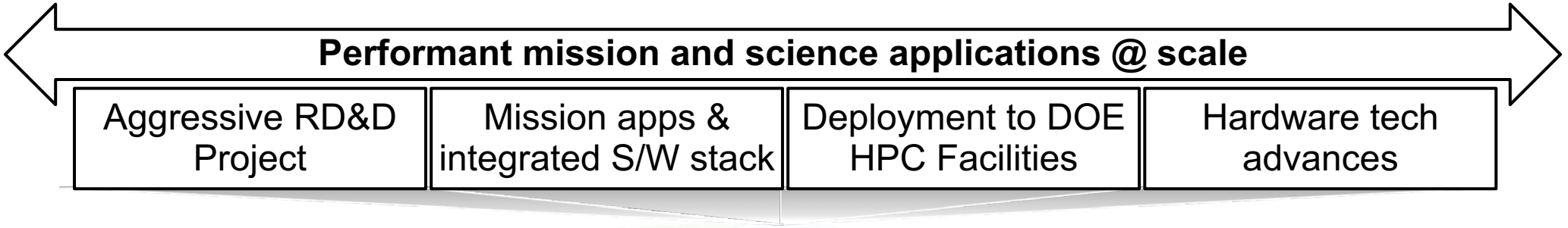
ECI mission

Accelerate R&D, acquisition, and deployment to deliver exascale computing capability to DOE national labs by the early- to mid-2020s

ECI focus

Delivery of an *enduring and capable exascale computing capability for use by a wide range of applications* of importance to DOE and the US

ECP's three technical areas have the necessary components to meet national goals



Application Development (AD)

Develop and enhance the predictive capability of applications critical to the DOE

24 applications including national security, to energy, earth systems, economic security, materials, and data

Software Technology (ST)

Deliver expanded and vertically integrated software stack to achieve full potential of exascale computing

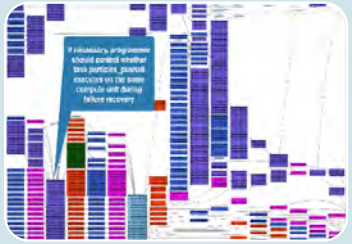
67 unique software products spanning programming models and run times, math libraries, data and visualization

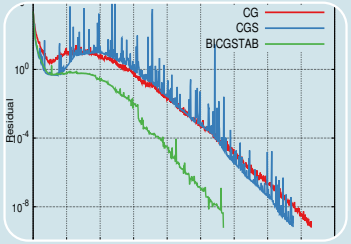
Hardware and Integration (HI)

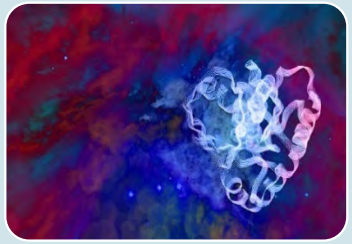
Integrated delivery of ECP products on targeted systems at leading DOE HPC facilities

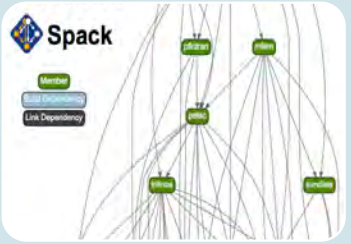
6 US HPC vendors focused on exascale node and system design; application integration and software deployment to facilities

ECP software technologies overview









National Nuclear Security Administration

Programming Models & Runtimes

- Enhance and get ready for exascale the widely used MPI and OpenMP programming models (hybrid programming models, deep memory copies)
- Development of performance portability tools (e.g. Kokkos and Raja)
- Support alternate models for potential benefits and risk mitigation: PGAS (UPC++/GASNet) ,task-based models (Legion, PaRSEC)
- Libraries for deep memory hierarchy and power management

Development Tools

- Continued, multifaceted capabilities in portable, open-source LLVM compiler ecosystem to support expected ECP architectures, including support for F18
- Performance analysis tools that accommodate new architectures, programming models, e.g., PAPI, Tau

Math Libraries

- Linear algebra, iterative linear solvers, direct linear solvers, integrators and nonlinear solvers, optimization, FFTs, etc
- Performance on new node architectures; extreme strong scalability
- Advanced algorithms for multi-physics, multiscale simulation and outer-loop analysis
- Increasing quality, interoperability, complementarity of math libraries

Data and Visualization

- I/O via the HDF5 API
- Insightful, memory-efficient in-situ visualization and analysis – Data reduction via scientific data compression
- Checkpoint restart

Software Ecosystem

- Develop features in Spack necessary to support all ST products in E4S, and the AD projects that adopt it
- Development of Spack stacks for reproducible turnkey deployment of large collections of software
- Optimization and interoperability of containers on HPC systems
- Regular E4S releases of the ST software stack and SDKs with regular integration of new ST products

NNSA ST

- Open source NNSA Software projects
- Projects that have both mission role and open science role
- Major technical areas: New programming abstractions, math libraries, data and viz libraries
- Cover most ST technology areas
- Subject to the same planning, reporting and review processes

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.

Jonathan Carter, Software Technology Deputy Director
Jonathan has been involved in the support and development of HPC applications for chemistry, the procurement of HPC systems, and the evaluation of novel computing hardware for over 25 years. He currently a senior manager in Computing Sciences at Berkeley Lab.

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.

Lois Curfman McInnes, Math Libraries (2.3.3)
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.

Jim Ahrens, Data and Visualization (2.3.4)
Jim is a senior research scientist at the Los Alamos National Laboratory (LANL) and an expert in data science at scale. He started and actively contributes to many open-source data science packages including ParaView and Cinema.

Todd Munson, Software Ecosystem and Delivery (2.3.5)
Todd is a computational scientist in the Math and Computer Science Division of ANL. He has nearly 20 years of experience in high-performance numerical software, including development of PETSc/TAO and project management leadership in the ECP CODAR project.

Rob Neely, NNSA ST (2.3.6)
Rob is an Associate Division Leader in the Center for Applied Scientific Computing (CASC) at LLNL, chair of the Weapons Simulation & Computing Research Council, and lead for the Sierra Center of Excellence. His efforts span applications, CS research, platforms, and vendor interactions.

We work on products applications need now and into the future

Key themes:
- Exploration/development of new algorithms/software for emerging HPC capabilities:
- High-concurrency node architectures and advanced memory & storage technologies.
- Enabling access and use via standard APIs.

Software categories:
- The next generation of well-known and widely used HPC products (e.g., MPICH, OpenMPI, PETSc)
- Some lesser used but known products that address key new requirements (e.g., Kokkos, RAJA, Spack)
- New products that enable exploration of emerging HPC requirements (e.g., SICM, zfp, UnifyCR)

Example Products	Engagement
MPI – Backbone of HPC apps	Explore/develop MPICH and OpenMPI new features & standards.
OpenMP/OpenACC –On-node parallelism	Explore/develop new features and standards.
Performance Portability Libraries	Lightweight APIs for compile-time polymorphisms.
LLVM/Vendor compilers	Injecting HPC features, testing/feedback to vendors.
Perf Tools - PAPI, TAU, HPCToolkit	Explore/develop new features.
Math Libraries: BLAS, sparse solvers, etc.	Scalable algorithms and software, critical enabling technologies.
IO: HDF5, MPI-IO, ADIOS	Standard and next-gen IO, leveraging non-volatile storage.
Viz/Data Analysis	ParaView-related product development, node concurrency.

ECP ST Projects

- WBS
- Name
- PIs
- PCs

ECP ST Stats

- 33 L4 projects
- 11 PI/PC same
- 22 PI/PC different

WBS	WBS Name	CAM/PI	PC
2.3	**Software Technology**	**Heroux, Mike, Carter, J.**	
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	Bayyapu, Neelima
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	Baden, Scott	Hargrove, Paul (and PI)
2.3.1.16	SICM	Lang, Michael	Vigil, Brittney
2.3.1.17	OMPI-X	Bernholdt, David	Alicia Grundhoffer
2.3.1.18	RAJA/Kokkos	Trott, Christian Robert	Trott, Christian
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	Mellor-Crummey, John
2.3.2.10	PROTEAS-TUNE	Vetter, Jeff	Glassbrook, Dick
2.3.2.11	SOLLVE: Scaling OpenMP with LLVm for Exascale	Chapman, Barbara	Kong, Martin
2.3.2.12	FLANG	McCormick, Pat	Perry-Holby, Alexis
2.3.3	**Mathematical Libraries**	**McInnes, Lois**	
2.3.3.01	Extreme-scale Scientific xSDK for ECP	Yang, Ulrike	Yang, Ulrike
2.3.3.06	Preparing PETSc/TAO for Exascale	Smith, Barry	Munson, Todd
2.3.3.07	STRUMPACK/SuperLU/FFTX: sparse direct solvers, preconditioners, and FFT libraries	Li, Xiaoye	Li, Xiaoye
2.3.3.12	Enabling Exascale Simulations with SUNDIALS and 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	Alicia Grundhoffer
2.3.4	**Data and Visualization**	**Ahrens, James**	
2.3.4.01	Data and Visualization Software Development Kit	Atkins, Chuck	Atkins, Chuck
2.3.4.09	ADIOS Framework for Scientific Data on Exascale Systems	Klasky, Scott	Alicia Grundhoffer
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
2.3.6	**NNSA ST**	**Neely, Rob**	
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

ECP ST Product Dictionary List – Actively managed

Widely-recognized product names. Enables mapping between AD & Facilities dependencies and ST development efforts.

- MPI – MPICH, OpenMP
- C++/C/Fortran - LLVM
- Fortran – Flang
- hypre – hypre

- Actively managed at L2 level.

ADIOS
AML
Ascent
BLAS
C
C++
Caliper
Catalyst
CHAI
Cinema
CUDA
Darshan
DTK
Dyninst
E4S
FFT
FleCSI

Flux
Fortran
GASNet
Ginkgo
HDF5
HPCToolkit
hypre
Kokkos
KokkosKernels
LAPACK
Legion
libEnsemble
MarFS
MFEM
MPI
OpenACC
OpenCL

OpenMP
PAPI
Papyrus
Paraview
PaRSEC
PETSc/TAO
PnetCDF
PowerStack
RAJA
MPI-IO
ScaLAPACK
SCR
SICM
Spack
SPOT
STRUMPACK

SUNDIALS
SuperLU
SYCL
SZ
TASMANIAN
TAU
Trilinos
UMap
Umpire
Unify
UPC++
VeloC
Visit
VTK-m
xSDK
ZFP

Product	URL	Description	Deployment Scope	Technical Area	POC
1. ADIOS	https://github.com/ornladios/ADIOS2	I/O and data management library for storage I/O, in-memory code coupling and online data analysis and visualization workflows.	Broad	Data & Viz	Scott Klasky
2. AID		Advanced Infrastructure for Debugging (AID) - a toolkit of debugging and correctness tools for extreme scale and GPU architectures	Experimental	Tools	Dong Ahn
AID: STAT	https://github.com/LLNL/stat	Stack Trace Analysis Tool - lightweight debugging at extreme scale	Broad	Tools	
AID: Archer	https://github.com/PRUNERS/archer	Data race detection tool for OpenMP applications	Broad	Tools	
AID: FLiT	https://github.com/PRUNERS/FLiT	Quickly detect discrepancies in floating point computation across hardware, compilers, libraries and software	Moderate	Tools	
AID: ReMPI	https://github.com/PRUNERS/ReMPI	A lightweight record-and-replay tool for MPI+OpenMP applications	Moderate	Tools	
AID: FPChecker	https://github.com/LLNL/FPChecker	Floating point exception trapping for NVIDIA GPUs	Experimental	Tools	

Partial List of Jira Dependency Issues

Search | Save as | Share | Export | Tools

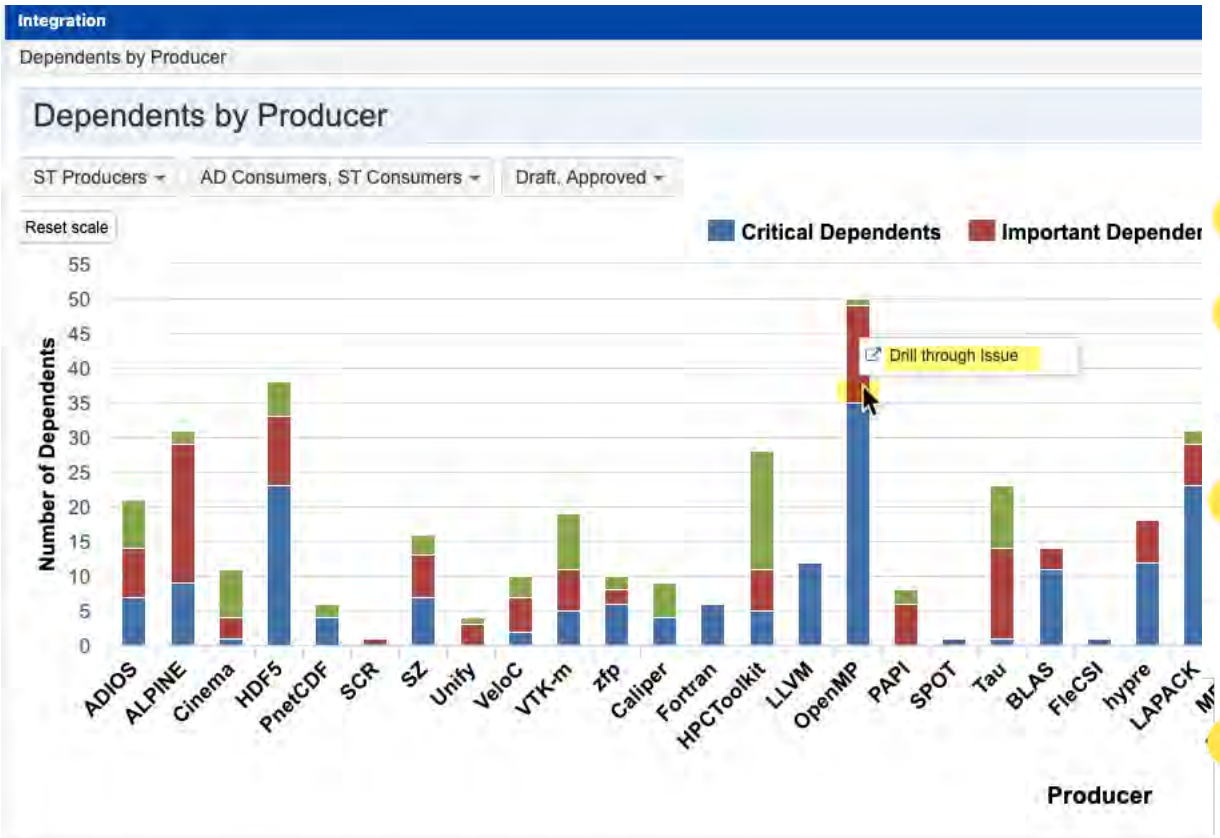
Project: All | Dependency | Status: All | Assignee: All | Contains text | More | Search | Advanced

1–50 of 814 | Columns

T	Key	Summary	Assignee	Reporter	Status	Resolution	Created	Due	Baseline end date	Links	Development
	INT-691	PETSc/TAO ↔ Spack	*Unassigned*	Todd Munson	APPROVED	Approved	2019-09-10				
	INT-686	HDF5 ↔ PETSc/TAO	*Unassigned*	Todd Munson	APPROVED	Approved	2019-09-10				
	INT-690	PETSc/TAO ↔ xSDK	*Unassigned*	Todd Munson	APPROVED	Approved	2019-09-10				
	INT-687	MPI-IO ↔ PETSc/TAO	*Unassigned*	Todd Munson	APPROVED	Approved	2019-09-10				
	INT-689	OpenMP ↔ PETSc/TAO	*Unassigned*	Todd Munson	APPROVED	Approved	2019-09-10				
	INT-677	libEnsemble ↔ xSDK	*Unassigned*	Todd Munson	APPROVED	Approved	2019-09-10				
	INT-681	BLAS ↔ PETSc/TAO	*Unassigned*	Todd Munson	APPROVED	Approved	2019-09-10				

Jira Dependency Issue Type

- Official ST and AD Product lists enable rigorous dependency management.
- New Jira Dependency issue type.
- Explicit dependency connections between AD/CD/ST products



Edit Issue : INT-1065

Configure Fields

Reporter* ECP Support

Start typing to get a list of possible matches.

6 Producer* Programming Models and Runtimes | Legion

Select the application code or product name that the Consumer *depends on*.

7 Consumer* Data Analytics and Optimization | CANDLE

Select the application code or ST product that *depends on* the Producer.

Dependency Level*

8

- Critical
- Important
- Interested

Critical: The team is entirely dependent on the producer for this functionality, and there are no alternatives available.

Important: The team believes this producer is the best source for this functionality, but alternative sources exist.

Interested: The team is interested enough in the functionality that they are likely to try to adopt it into their work.

9 Functionality Description: ENTER BRIEF DESCRIPTION HERE

Enter a brief description of the functionality that this Producer provides this Consumer.

10 Trigger Event* Unknown

Provide the event/quarter you expect this dependency will be needed by.

Optional Items

11 Linked Issues: blocks

Issue

Begin typing to search for issues to link. If you leave it blank, no link will be made.

Attachment: Drop files to attach, or browse.

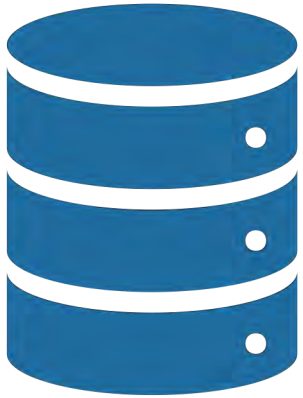
12 Labels: auto-generated ×

Begin typing to find and create labels or press down to select a suggested label.

Delivering a modular, interoperable, and deployable software stack

ECP ST Planning Process: Hierarchical, three-phase, cyclical

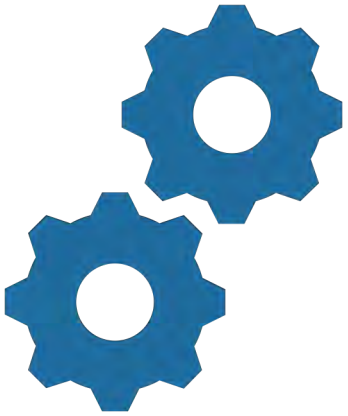
Baseline



FY20–23 Baseline Plan
High level Definitions

- Q2 FY19 start
- FY20 Base plan
- FY21–23 planning packages

Annual Refinement



FY Refine Baseline Plan
As Needed
Basic activity definitions

- 6 months prior to FY
- 4–6 P6 Activities/year
- Each activity:
  - % annual budget
  - Baseline start/end
  - High level description

Per Activity

Detailed Plan
Complete activity definitions

- 8 weeks prior to start
- High-fidelity description
- Execution strategy
- Completion criteria
- Personnel details

Two-level Change Control

Changes to Cost, Scope, and Schedule

Minor	Major
Lightweight Review in Jira, L3 and L2 leads	Change Control Board Review, ECP leadership

Variance Recorded in Jira

Proceed with Execution

KPP-3 Definition

KPP ID	Description of Scope	Threshold KPP	Objective KPP	Verification Action/Evidence
KPP-1	Performance of scientific and national security applications relative to today's performance	50% of selected applications achieve Figure of merit* improvement ≥50	100% of selected applications achieve Figure of merit improvement stretch goal	Independent assessment of measured results and report that threshold goal is met
KPP-2	Broaden the reach of exascale science and mission capability	50% of selected applications can execute their challenge problem*	100% of selected applications can execute their challenge problem stretch goal	Independent assessment of mission application readiness
KPP-3	**Productive and Sustainable Software Ecosystem**	**Software teams meet 50% of their weighted impact goals***	**Software teams meet 100% of their weighted impact stretch goals**	**Independent assessment verifying threshold goal is met**
KPP-4	Enrich the HPC Hardware Ecosystem	Vendors meet 80% of all the PathForward milestones	Vendors meet 100% of all the PathForward milestones	Independent assessment of the impact and timeliness of PathForward milestones

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).
- **Product:** Capabilities are integrated into primary products chosen from a product dictionary.
- **Primary products examples:**
  - MPI is commonly known by users. MPICH and OpenMPI both provide implementations of that product.
  - Fortran is a product. Flang is a particular Fortran product. LLVM is a backend for some Fortran compilers.
  - FFT is a product. FFTX, FFT-ECP provide FFT capabilities through interchangeable interfaces.
  - C++ is a product. Clacc provides capabilities for Clang, as does LLVM.
- **Details:** See Confluence page export.

Delivering a Modular, Interoperable, and Deployable Software Stack

Objectives

- Lower barrier to use ST products
- Lower barrier to enable Facilities to install all or parts
- Enable interoperability between ST products
- Enable uniform APIs where possible
- Improve KPP-3 success

Challenges

- Large diverse group of ST products
- Different project management styles
- Lack of initial drivers for integration

Spack

https://spack.io github.com/spack @spackpm

A flexible package manager for HPC

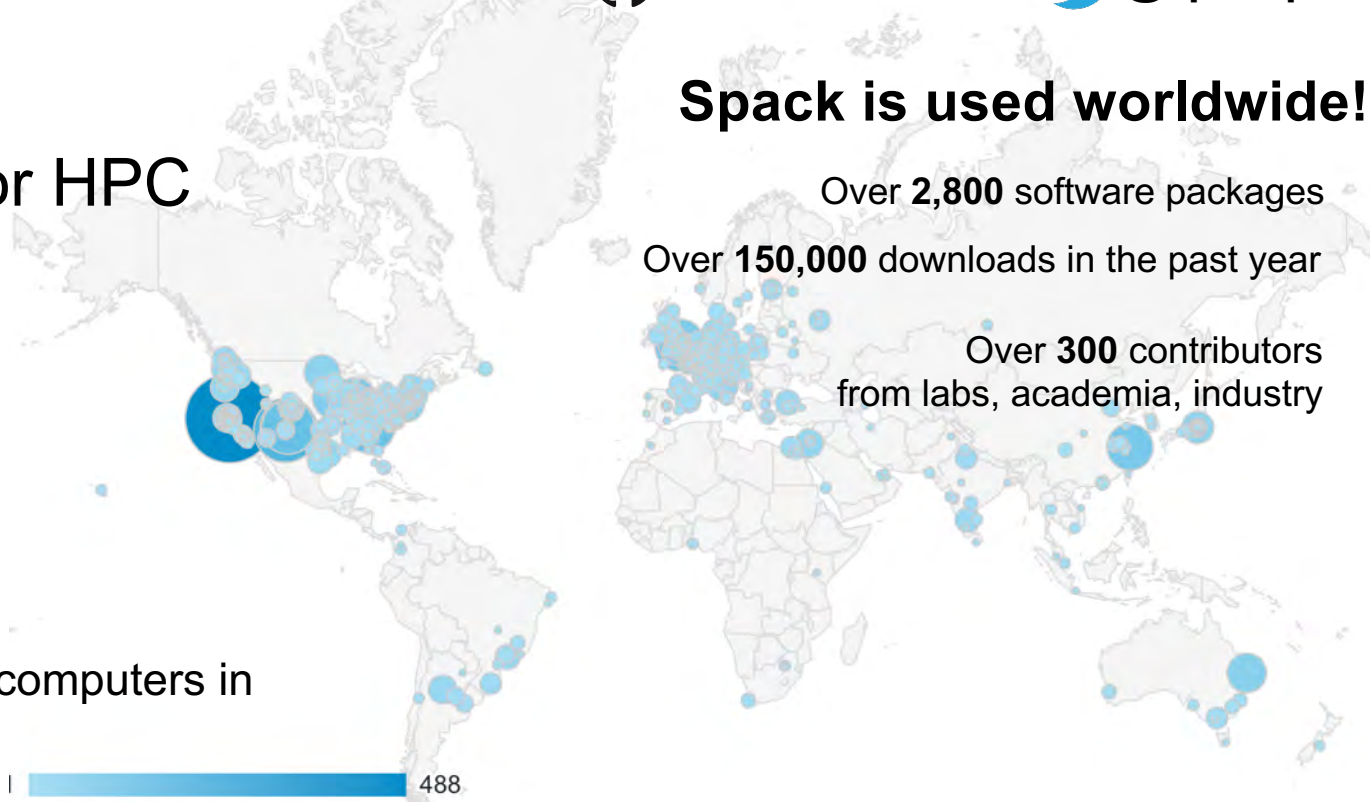
- Inspired by Homebrew, Nix, some others
- Support scientific stacks with multiple languages
- Flexibility:
  - Build packages many different ways
  - Change compilers and flags in builds
  - Swap implementations of libraries (MPI, BLAS, etc.)
- Run on laptops, Linux clusters, and the largest supercomputers in the world

Spack is used worldwide!

Over **2,800** software packages

Over **150,000** downloads in the past year

Over **300** contributors
from labs, academia, industry

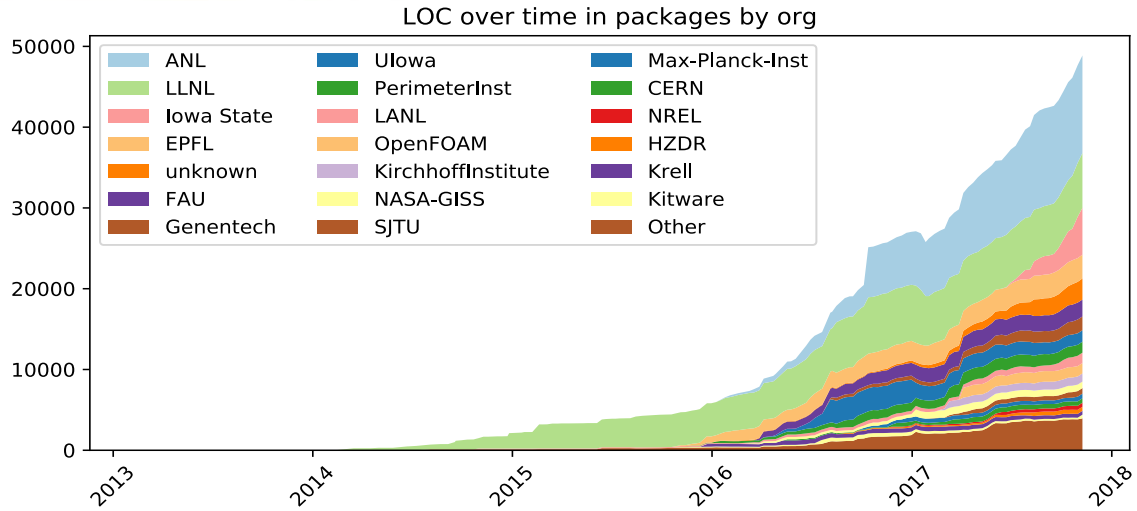


Easy installation

```
$ git clone https://github.com/spack/spack
$ . spack/share/spack/setup-env.sh
$ spack install hdf5
```

Easy customization

```
$ spack install mpileaks@3.3
$ spack install mpileaks@3.3 %gcc@4.7.3 +threads
$ spack install mpileaks@3.3 cppflags="-O3 -g3"
$ spack install mpileaks@3.3 target=haswell
$ spack install mpileaks@3.3 ^mpich@3.2
```

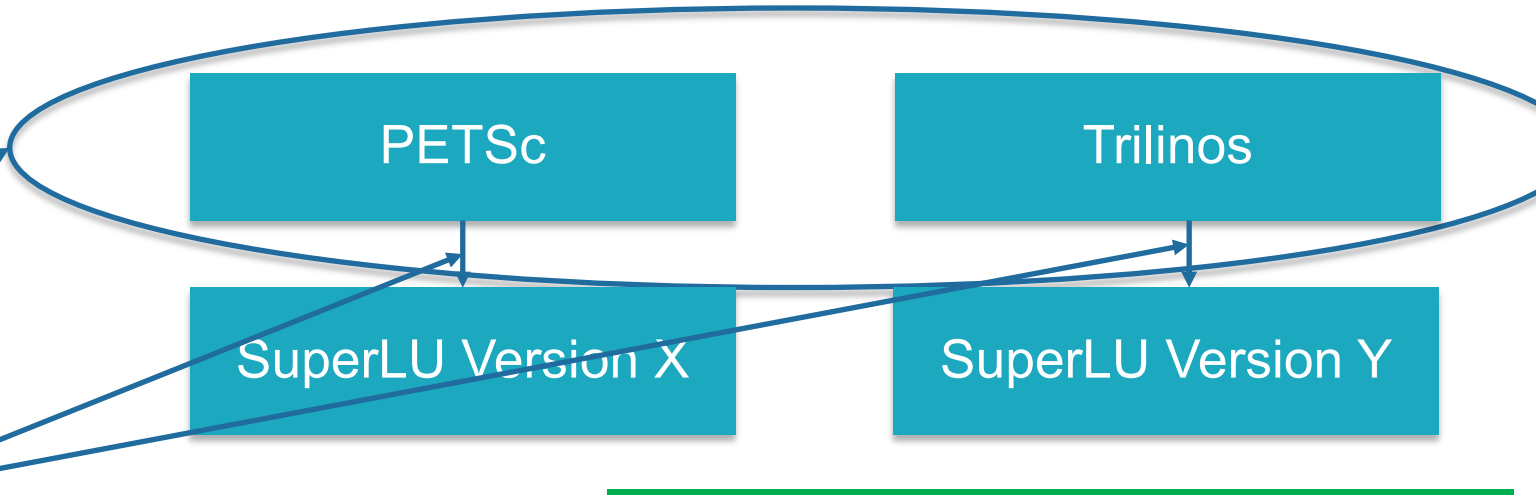


Software Development Kits are a key delivery vehicle for ECP

- A collection of related software products (called packages) where coordination across package teams will improve usability and practices and foster community growth among teams that develop similar and complementary capabilities
- Attributes
  - **Domain scope:** Collection makes functional sense
  - **Interaction model:** How packages interact; compatible, complementary, interoperable
  - **Community policies:** Value statements; serve as criteria for membership
  - **Meta-infrastructure:** Invokes build of all packages (Spack), shared test suites
  - **Coordinated plans:** Inter-package planning. Augments autonomous package planning
  - **Community outreach:** Coordinated, combined tutorials, documentation, best practices
- *Overarching goal: Unity in essentials, otherwise diversity*

ECP ST SDKs: Grouping Similar Products for collaboration & usability	
Prog Models & Runtimes Core	Math Libraries (xSDK)
Tools & Technologies	Viz Analysis and Reduction
Compilers & Support	Data mgmt., I/O services & Checkpoint/Restart

SDK “Horizontal” Grouping: Key Quality Improvement Driver



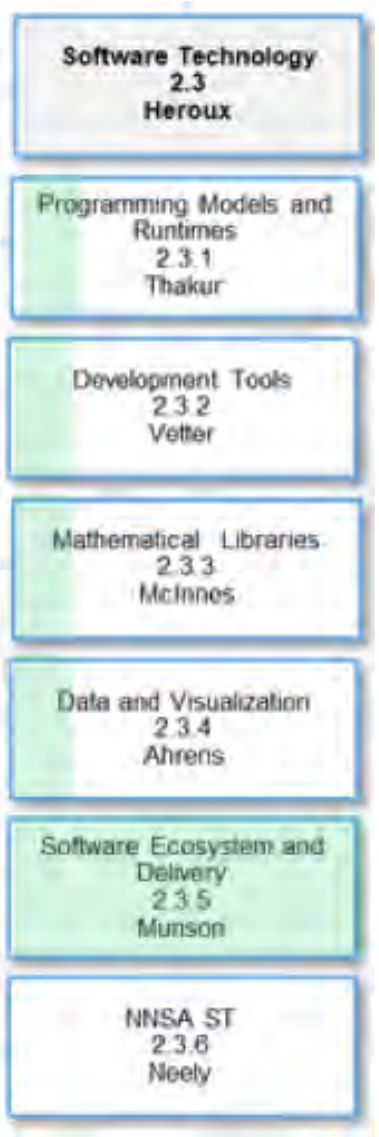
Horizonal (vs Vertical) Coupling

- Common substrate
- Similar function and purpose
  - e.g., compiler frameworks, math libraries
- Potential benefit from common Community Policies
  - Best practices in software design and development and customer support
- Used together, but not in the long vertical dependency chain sense
- Support for (and design of) common interfaces
  - Commonly an aspiration, not yet reality

Horizontal grouping:
- Assures X=Y.
- Protects against regressions.
- Transforms code coupling from heroic effort to turnkey.

There are 5 L4 projects to define and/or enhance SDKs

- Each L3 area has an L4 project devoted to SDK definition and coordination across the portfolio
- Software ecosystem L4 project focuses on packaging
  - Spack
  - Containers
- Strong coordination with HI Software Deployment projects
- Drives milestone-based planning

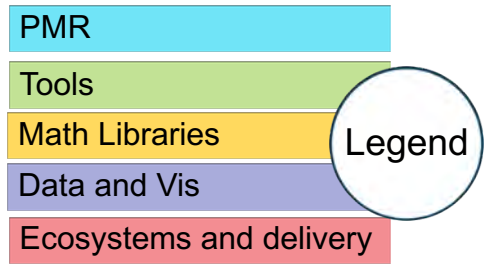


ECP ST SDKs will span all technology areas

Motivation: Properly chosen cross-team interactions will build relationships that support interoperability, usability, sustainability, quality, and productivity within ECP ST.

Action Plan: Identify product groupings where coordination across development teams will improve usability and practices, and foster community growth among teams that develop similar and complementary capabilities.

PMR Core (17)	Compilers and Support (7)	Tools and Technology (11)	xSDK (16)	Visualization Analysis and Reduction (9)	Data mgmt, I/O Services, Checkpoint restart (12)	Ecosystem/E4S at-large (12)
QUO	openarc	TAU	hypre	ParaView	SCR	mpiFileUtils
Papyrus	Kitsune	HPCToolkit	FleSCI	Catalyst	FAODEL	TriBITS
SICM	LLVM	Dyninst Binary Tools	MFEM	VTK-m	ROMIO	MarFS
Legion	CHiLL autotuning comp	Gotcha	Kokkoskernels	SZ	Mercury (Mochi suite)	GUFI
Kokkos (support)	LLVM openMP comp	Caliper	Trilinos	zfp	HDF5	Intel GEOPM
RAJA	OpenMP V & V	PAPI	SUNDIALS	VisIt	Parallel netCDF	BEE
CHAI	Flang/LLVM Fortran comp	Program Database Toolkit	PETSc/TAO	ASCENT	ADIOS	FSEFI
PaRSEC*		Search (random forests)	libEnsemble	Cinema	Darshan	Kitten Lightweight Kernel
DARMA		Siboka	STRUMPACK	ROVER	UnifyCR	COOLR
GASNet-EX		C2C	SuperLU		VeloC	NRM
Qthreads		Sonar	ForTrilinos		IOSS	ArgoContainers
BOLT			SLATE		HXHIM	Spack
UPC++			MAGMA			
MPICH			DTK			
Open MPI			Tasmanian			
Umpire			TuckerMPI			
AML						



SDK Summary

- SDKs will help reduce complexity of delivery:
  - Hierarchical build targets.
  - Distribution of software integration responsibilities.
- New Effort: Started in April 2018, fully established in August 2018.
- Extending the SDK approach to all ECP ST domains.
  - SDKs create a horizontal coupling of software products, teams.
  - Create opportunities for better, faster, cheaper – pick all three.
- First concrete effort: Spack target to build all packages in an SDK.
  - Decide on good groupings.
  - Not necessarily trivial: Version compatibility issues, Coordination of common dependencies.
- Longer term:
  - Establish community policies, enhance best practices sharing.
  - Provide a mechanism for shared infrastructure, testing, training, etc.
  - Enable community expansion beyond ECP.

ST works with HI and Facilities to produce a stack, build, and container infrastructure

E4S/Software

- Regular additions of software technologies, integration testing, and releases of E4S
- Certified Spack recipes and binaries for software technologies
- Currently includes 37 software technologies
- Broad adoption of reliable, performant, exascale-ready software

Packaging Tech/Facilities

- Turnkey deployment of large collections of software technologies at facilities
- Coordinate with and leverage facility personnel and resources, such a continuous integration capabilities
- Spack recipes for over 2,800 software packages
- Broad deployment of exascale-ready software technologies

Packaging Tech/Applications

- Optimize and improve interoperability of container technologies
- Enable entire application to be packaged into reproducible container images
- Accelerate application development and deployment workflows

Continuous Integration (CI) is critical for delivery of high-quality SW

Fundamental value: Minimize time between fault injection & detection

ST Delivers

- Focus on production-quality software processes
- Provide SDKs as a natural unit of software groupings
- Allow for custom "turnkey" builds via Spack
- Optionally - provide containers of products and SDKs
- Regular software releases made available to Facilities and open source community

HI Deploys

- Work with ST/Facilities on a consistent deployment model for ST products
- Coordinate a support model for ST products
- Provide linkages to Facility "site stacks" and vendor offerings
- **Build common infrastructure for testing and deployment**
- Training, knowledge transfer, promotion of ST products

Continuous Integration (CI) is critical for delivery of high-quality SW
Fundamental value: Minimize time between fault injection & detection

ECP Goal: To establish a cross-site Continuous Integration testing infrastructure that:

- Provides for account authentication and access to CI test resources across multiple sites
- Provides unique and targeted HPC test resources to support software development teams
- Establishes a standard process across the DOE sites for software development testing

- Allows for the verification of development efforts through automated building and testing across sites to better identify errors and improve code efficiency
- Continuous Integration/Continuous Delivery pipelines are enabled through a combination of the web application and project configuration files and are executed by selected runners

Preparing for Exascale Platforms

Department of Energy (DOE) Roadmap to Exascale Systems

An impressive, productive lineup of *accelerated node* systems supporting DOE's mission

Pre-Exascale Systems [Aggregate Linpack (Rmax) = 323 PF!]

First U.S. Exascale Systems

2012	2016	2018	2020	2021-2023

ORNL
Cray/AMD/NVIDIA

ORNL
IBM/NVIDIA

ORNL
TBD

ANL
IBM BG/Q

ANL
Cray/Intel KNL

ANL
Intel/Cray

LBNL
Cray/Intel Xeon/KNL

LBNL
Cray/AMD/NVIDIA

LLNL
IBM BG/Q

LANL/SNL
Cray/Intel Xeon/KNL

LLNL
IBM/NVIDIA

LANL/SNL
TBD

LLNL
TBD

EXASCALE COMPUTING PROJECT

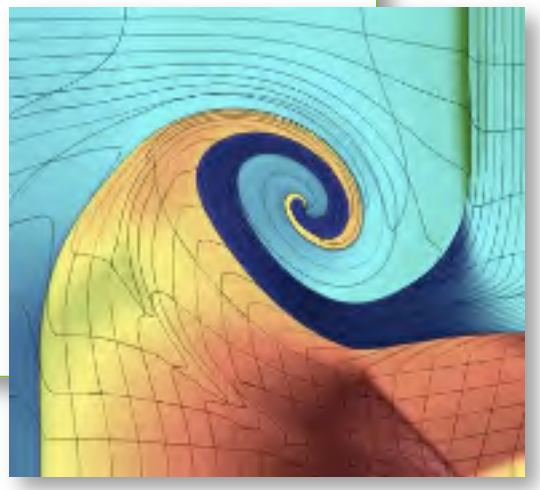
Hardware realities are forcing new thinking of algorithmic implementations and the move to new algorithms

Algorithmic Implementations

- Reduced communication/data movement
  - Sparse linear algebra, Linpack, etc.
- Much greater locality awareness
  - Likely must be exposed by programming model
- Much higher cost of global synchronization
  - Favor maxim asynchrony where physics allows
- Value to mixed precision where possible
  - Huge role in AI, harder to pin down for PDEs
- Fault resilience?
  - Likely handled outside of applications

New Algorithms

- Adopting Monte Carlo vs. Deterministic approaches
- Exchanging on-the-fly recomputation vs. data table lookup (e.g. neutron cross sections)
- Moving to higher-order methods (e.g. CFD)
- Particle algorithms that favor collecting similar events together rather than parallelism though individual histories



Exascale Application Development Challenges Overall

1) **Porting to accelerator-based architectures**
2) **Exposing additional parallelism**
3) **Coupling codes to create new multiphysics capability**
4) **Adopting new mathematical approaches**
5) **Algorithmic or model improvements**
6) **Leveraging optimized libraries**

Topology-Aware Collectives

ECP WBS 2.3.1.07 STPM09

PI Pavan Balaji, ANL

Scope and objectives

- Exascale applications rely heavily on MPI collectives to efficiently implement group communication.
- MPI collectives can be more efficiently performed with knowledge of both the application and machine topology.
- Design and implement topology-aware support for applications and machine topologies for use at exascale.

Impact

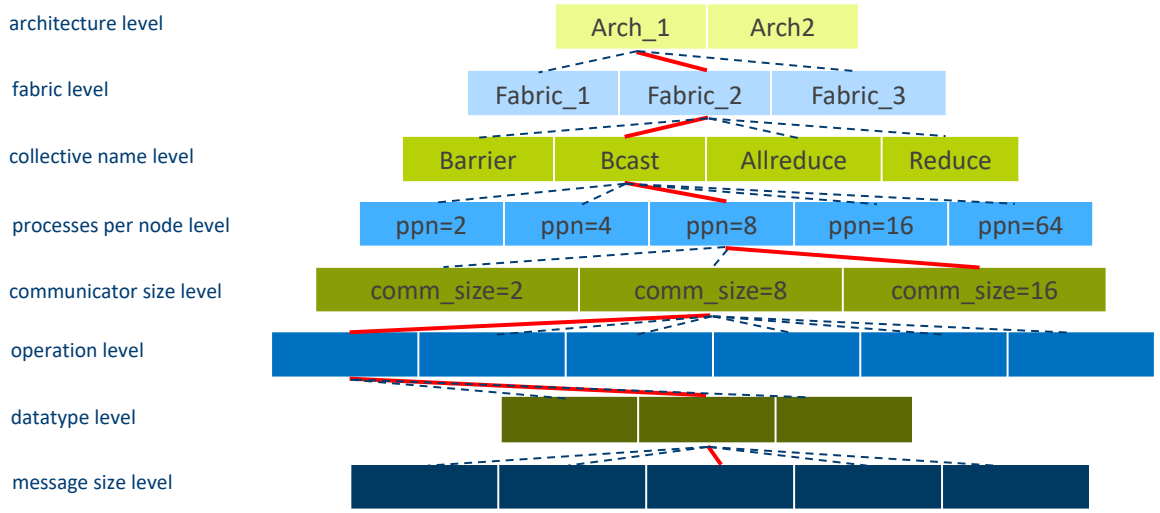
- It is imperative for communication libraries such as Exascale MPI to efficiently perform collective operations on both physical and virtual topologies. Many ECP applications rely on MPI collectives for performance and efficiency.

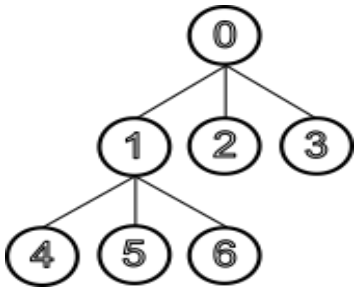
Project accomplishment

- We have prototyped a suite of new collective algorithms intended to perform well on large multicore machines, as well as network topologies such as a dragonfly.
- We have prototyped neighborhood collectives using sub-communicators to exploit the virtual topology of many ECP applications, i.e. nearest neighbor exchange.
- We have designed and prototyped a flexible framework for selecting the best collective algorithm for a given set of parameters.

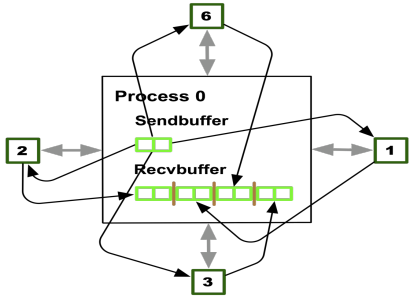
Deliverables

Topology-Aware Collective Algorithms and Selection





k-ary tree (k=3)



MPI_Neighbor_ allgather

Software

K-ary, K-nomial, Recursive Exchange collective algorithms: http://www.mpich.org/static/downloads/3.3.1/mpich-3.3.1.tar.gz
Intelligent Collective Selection: https://github.com/pmodels/mpich/pull/3897
Message-combining Optimizations for Neighborhood Collectives: https://github.com/pmodels/mpich/pull/3892
Sub-communicator Optimizations for Neighborhood Collectives: https://github.com/pmodels/mpich/pull/3883

Enhanced LLVM OpenMP compiler

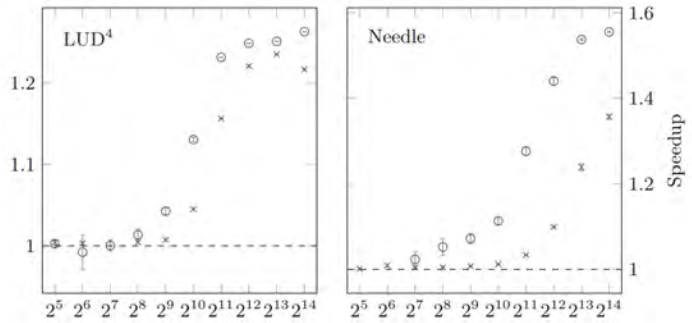
ECP WBS	SOLLVE
PI	Barbara Chapman, BNL
Members	BNL,ANL, Rice, ORNL, LLNL

Scope and objectives

- OpenMP, the primary programming model for on-node currency in ECP applications
- Improve the performance and functionality of LLVM's OpenMP implementation, especially on GPUs
- Improve the interaction between OpenMP and loop-nest optimizations, and the ability to express loop-nest optimizations in OpenMP
- Implement new OpenMP features in LLVM, specifically features for mapping complex data structures in accelerators

Cool image

OpenMP for GPU "target region" optimizations. Evaluation of four Rodinia v3.1 on a NVIDIA K40 and V100. Speedup vs. Input size.



Impact

- OpenMP is the de facto standard multithreading even outside ECP. Any impact within ECP translates into impact outside of it
- Most ECP applications rely on OpenMP for intranode and MPI for internode programming. Improvements in OpenMP performance on GPUs will translate directly to performance on exascale architectures
- OpenMP features implemented in LLVM are often incorporated into LLVM-based vendor toolchains

Project accomplishment

- Implemented improved general infrastructure allowing for the optimization of OpenMP target regions, with specific optimization enablement for GPUs.
- Designed and prototyped new OpenMP loop-optimization pragmas. Demonstrated that these pragmas can replace complex source-level transformations yielding performance and productivity improvements.
- Implemented initial support for OpenMP user-defined mapping directives.

Deliverables

Publications

- *Johannes Doerfert, Jose Manuel Monsalve Diaz and Hal Finkel*. “The TRegion Interface and Compiler Optimizations For OpenMP Target Regions”, IWOMP 2019.
- Michael Kruse and Hal Finkel. “Design and Use of Loop-Transformation Pragmas”, IWOMP 2019.

Project Report Uploaded on Jira (see report for code under review and repository links)

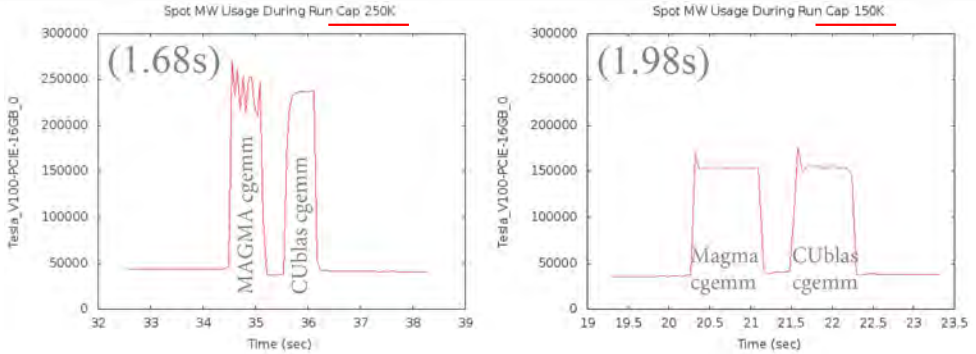
Fulfillment of NVIDIA GPU Performance Counter and Power Management Support

ECP WBS	Exa-PAPI
PI	Dongarra, UTK
Members	Heike Jagode (Co-PI) Anthony Danalis (Co-PI) Tony Castaldo

Scope and Objectives

- PAPI support for NVML, CUPTI, NVLINK performance events, and power consumption management on GPUs.
- Added PAPI support for **TESLA V100 GPUs** and **NVLINK**.
- Increase options and info for multi-objective optimization; e.g. reducing energy usage when GPUs are not the speed bottleneck, or a speedy result is not paramount.

Power Reading and Capping on TESLA V100 GPUs



Impact

- **PAPI "nvml":** Monitoring of GPU power consumption, fan speed, temperature, and power capping support: allows developers to change run profiles to reduce energy cost.
- **PAPI "cuda":** Monitoring of GPU and NVLINK performance counters aids developers in producing more efficient code by profiling the utilization of the latest GPU resources and diagnosing performance bottlenecks.

Project Accomplishment

- Production interface of PAPI to CUPTI and NVML, with well-documented example code, proving the availability and efficacy of the controls in a standardized way.
- We demonstrate PAPI NVIDIA GPU power reading and control, and the use of performance counters across multiple GPUs

Deliverables	git clone https://bitbucket.org/icl/papi.git. In papi/src/components/nvml/tests: nvmlcap_plot.cu, nvml_power_limiting_test.cu. In papi/src/components/cuda/tests: simpleMultiGPU.cu. Corresponding Makefile in each directory, commentary in code.

Hierarchical Off-Diagonal Low-Rank Matrix Compression in the STRUMPACK Preconditioner

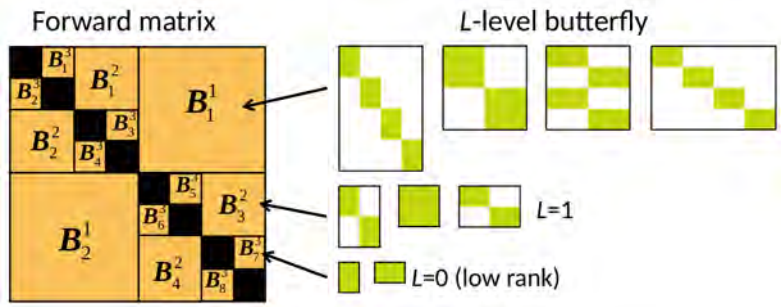
ECP WBS	2.3.3.07 Sparse solvers and preconditioners
PI	X. Sherry Li, LBNL
Members	Pieter Ghysels, Yang Liu: LBNL

Scope and objectives

- Sparse direct solvers and factorization based preconditioners. This milestone focuses on dense matrix compression for use in the preconditioner
- Need to improve efficiency and rate of compression for frontal matrices (fill-in in multifrontal sparse factorization)
- Accurate and fast compression improves preconditioning for numerically hard problems

HODLR and Butterfly data-sparse formats

Off diagonal blocks are low-rank/butterfly compressed. Butterfly multi-level low-rank compression is based on ideas from FFT.



Impact

- Asymptotically reduce memory and computation in sparse solvers. This allows ECP and other users, (also through MFEM or PETSc) to solve larger problems, faster
- Provide robust, purely algebraic, parallel solvers and preconditioners for numerically challenging ECP applications, for instance, high frequency Helmholtz

Project accomplishment

- HODLR + butterfly in ButterflyPACK (Fortran/C), and integrated in STRUMPACK (C++) preconditioner
- Evaluation on MFEM indefinite Maxwell test code, typically solved with direct solvers
- Observed much reduced memory usage compared to direct solver: 20x compression rate on largest fronts using butterfly

Deliverables	● https://github.com/liuyangzhuan/ButterflyPACK ● https://github.com/pghysels/STRUMPACK ● Summit (OLCF) was used for the evaluation

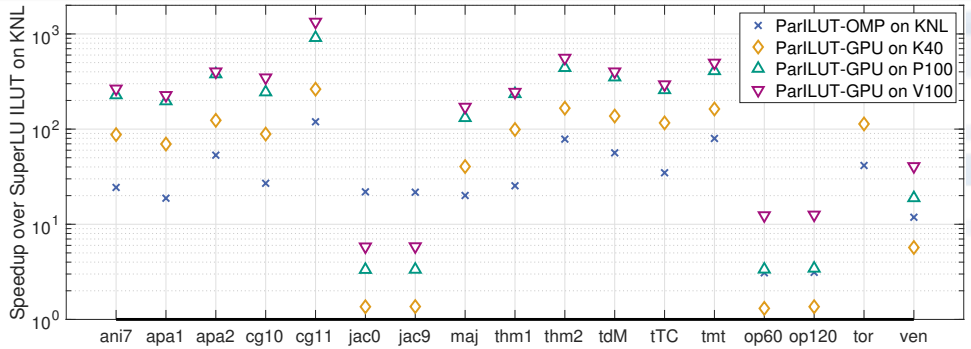
Cross-platform implementation for parallel incomplete factorizations

ECP WBS	STMS11-PEEKS
PI	Dongarra (UTK) Boman (SNL)
Members	Hoemmen (SNL) Anzt, Yamazaki (UTK)

Scope and objectives

- MAGMA-sparse release containing implementations for parallel incomplete factorizations (ParILU) for CPU & GPU architectures.
- MAGMA-sparse release containing implementations for parallel incomplete factorizations based on thresholding (ParILUT) for CPU & GPU architectures.
- MAGMA-sparse release containing implementations for parallel sparse triangular solves for CPU & GPU architectures.

Performance results for cross-platform ParILUT



Impact

- Cross-Platform implementations for parallel ILU preconditioning, parallel threshold ILU preconditioning, and sparse triangular solves available in MAGMA-sparse part of the xSDK software ecosystem.
- Presented the performance results of the first threshold ILU preconditioner (ParILUT) at the IPDPS 2019 conference and the JLESC 2019 meeting in Knoxville.

Current efforts

- Exploring the use of faster preconditioner pattern generation via Machine Learning.
- Looking for ECP application projects amenable to threshold-ILU preconditioning.
- Integrating ParILU / ParILUT into the Ginkgo software ecosystem featuring Continuous Benchmarking (CB).

Deliverables The ParILU and ParILUT preconditioners for architectures supporting OMP or CUDA are available in the MAGMA repository:
https://bitbucket.org/icl/magma/src
Performance results on the SUMMIT architecture were presented at the IPDPS conference in Rio de Janeiro, Brazil:
https://hartwiganzt.github.io/slides/IPDPS_ParILUT.pdf

Data Model Support in ADIOS for Visualization and Analysis

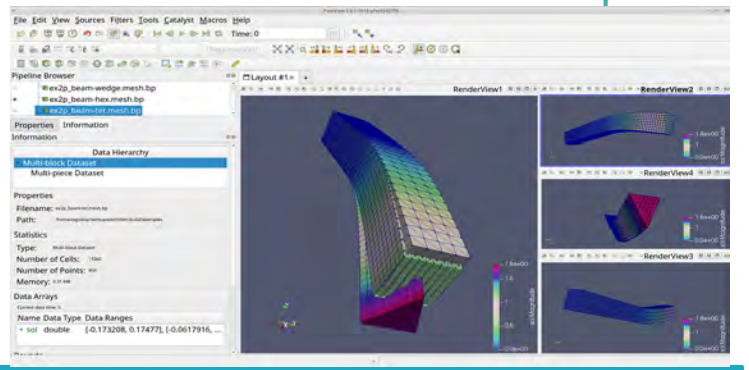
ECP WBS	2.3.4.09 STDM11-ADIOS2
PI	Scott Klasky, ORNL
Members	ORNL, Kitware, Georgia Tech, Rutgers, LBNL

Scope and objectives

- Describe variables in ADIOS using VTK XML format
- Develop the VTK::IOADIOS2 module to enable seamless visualization in Paraview
- Support ECP products:
  - MEUMAPPS (ExaAM) for ImageData
  - MFEM (CEED) and UnstructuredGrid

Visualization of ADIOS BP files

Seamless visualization in Paraview using the VTK::IOADIOS2 module and
VTK XML description in BP Files



Impact

- Applications using ADIOS can visualize their data in parallel using a simple string descriptor base on the well-documented VTK File data models
- Leverage current scalable I/O capabilities in ADIOS with VTK (and potentially VTK-m) for insitu visualization using a pluggable ecosystem delivered by the ECP data analysis and visualization software stack.

Project accomplishment

- MEUMAPPS (ExaAM) and MFEM (CEED) users can now visualize their data by adding simple XML descriptors either during the data generation or after
- The VTK::IOADIOS2 provides a reusable architecture that expands these capabilities and take advantage or the rich VTK ecosystem

Deliverables VTK::IOADIOS2 module merge request: https://gitlab.kitware.com/vtk/vtk/merge_requests/5613

ECP ST Review Focus

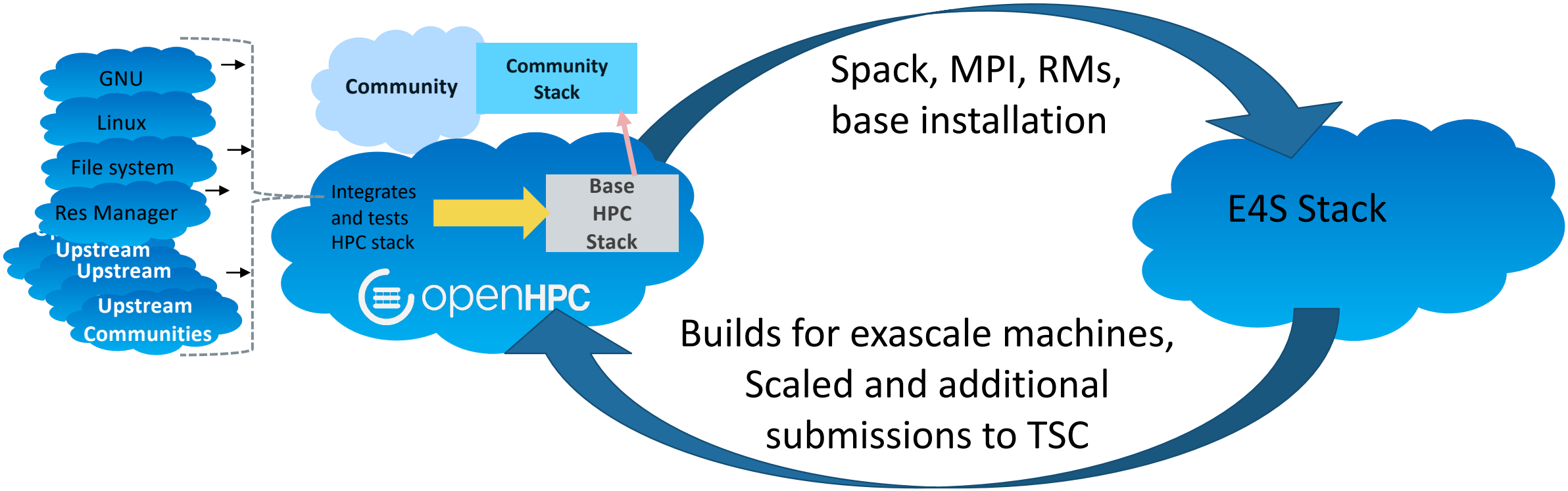
- Describe your overall approach followed by specific activities in preparation for exascale platforms.
  - Include a description of your overall strategy for your L4 project.
  - Include description of pre-exascale environments you are using.
  - Include status and any results from pre-exascale environments (GPU porting, use of Summit) that illustrate your strategy.
  - Include discussion of your major challenges.
- Describe your overall approach and describe the status of your client integration efforts.
  - Include a description of your overall strategy for your L4 project.
  - Include specific client interactions that illustrate your strategy.
  - Include recent integration progress.
  - Include discussion of your major challenges.

Detailed instructions for each L3 area: PMR

- Perlmutter, Aurora and Frontier represent very different node programming environments. As appropriate, how is your project addressing this challenge, including node performance, performance portability, performance of MPI+X and preparation for further heterogeneity?
- What is the path for your capabilities to realize sustainability in the software ecosystem?
- How is your work impacting vendor capabilities?

E4S Collaborations

How E4S and OpenHPC Work Together



- OpenHPC
  - Provides installation and base components for installing and running cluster to supercomputer
  - Provides generic binaries for all systems (working on targeted builds)

- E4S
  - Built from scratch for your system
  - Focus on upper layers of system software stack (libraries, runtimes, etc.)
  - Targeted for capability-class machines

44

NSF Collaborations

1st Workshop on NSF and DOE High Performance Computing Tools

Dates: July 10-11, 2019
Location: 312 Lillis, University of Oregon, Eugene, OR 97403

Workshop Agenda

Wednesday, July 10, 2019: 312 Lillis, University of Oregon

8:00am - 8:45am:	Registration
8:45am - 9:15am:	Welcome, introductions, David Conover (VPRI, UO), Vipin Chaudhary (Program Director, CISE/OAC, NSF), Jonathan Carter (Deputy Director, Software Technology, ECP, and Deputy for Science, LBL)[slides] [video]
9:15am - 10:30am:	Spack tutorial - Todd Gamblin and Greg Becker (LLNL) [slides] [video]
10:30am - 11:00am:	Break
11:00am - 12:30pm:	Spack tutorial - Todd Gamblin (LLNL) [video]
12:30pm - 2:00pm:	Lunch break: 440 Lillis
2:00pm - 3:30pm:	Spack tutorial - Todd Gamblin and Greg Becker (LLNL) [video]
3:30 pm - 4:00pm:	Break
4:00pm - 5:30pm:	Spack tutorial - Todd Gamblin and Greg Becker (LLNL) [video 1] [video 2] [video 3] [video 4] [video 5]
6:30pm - 9:30pm:	Working dinner at the Jordan Schnitzer Museum of Art, UO

Thursday, July 11, 2019: 312 Lillis, University of Oregon

9:00am - 9:30am:	Unifying Software Distribution in ECP - Todd Gamblin (LLNL) [slides] [video]
9:00am - 9:30am:	Containers for HPC - Andrew Younge (Sandia National Laboratories, NM) [slides] [video]
9:30am - 10:00am:	Software deployment at DOE facilities - David Montoya (Los Alamos National Laboratory, NM) [slides] [video]
10:00am - 10:30am:	E4S - Sameer Shende (University of Oregon) [slides] [video]
10:30am - 11:00am:	Break
11:00am - 11:30am:	Overview of software architecture - Todd Gamblin, LLNL
11:30am - 12:30pm:	Hands-on, applying Spack to applications.
12:30pm - 2:00pm:	Lunch: 440 Lillis
2:00pm - 3:30pm:	Hands-on, applying Spack to applications.
3:30pm - 4:00pm:	Break
4:00pm - 5:00pm:	Hands-on, Spack and E4S.
5:00pm - 6:15pm:	Closing remarks, planning for the next workshop - Jonathan Carter (Lawrence Berkeley National Laboratory)
6:30pm:	Dinner at Excelsior Inn

https://oaciss.uoregon.edu/NSFDOE19/agenda.html

Extending Collaborations

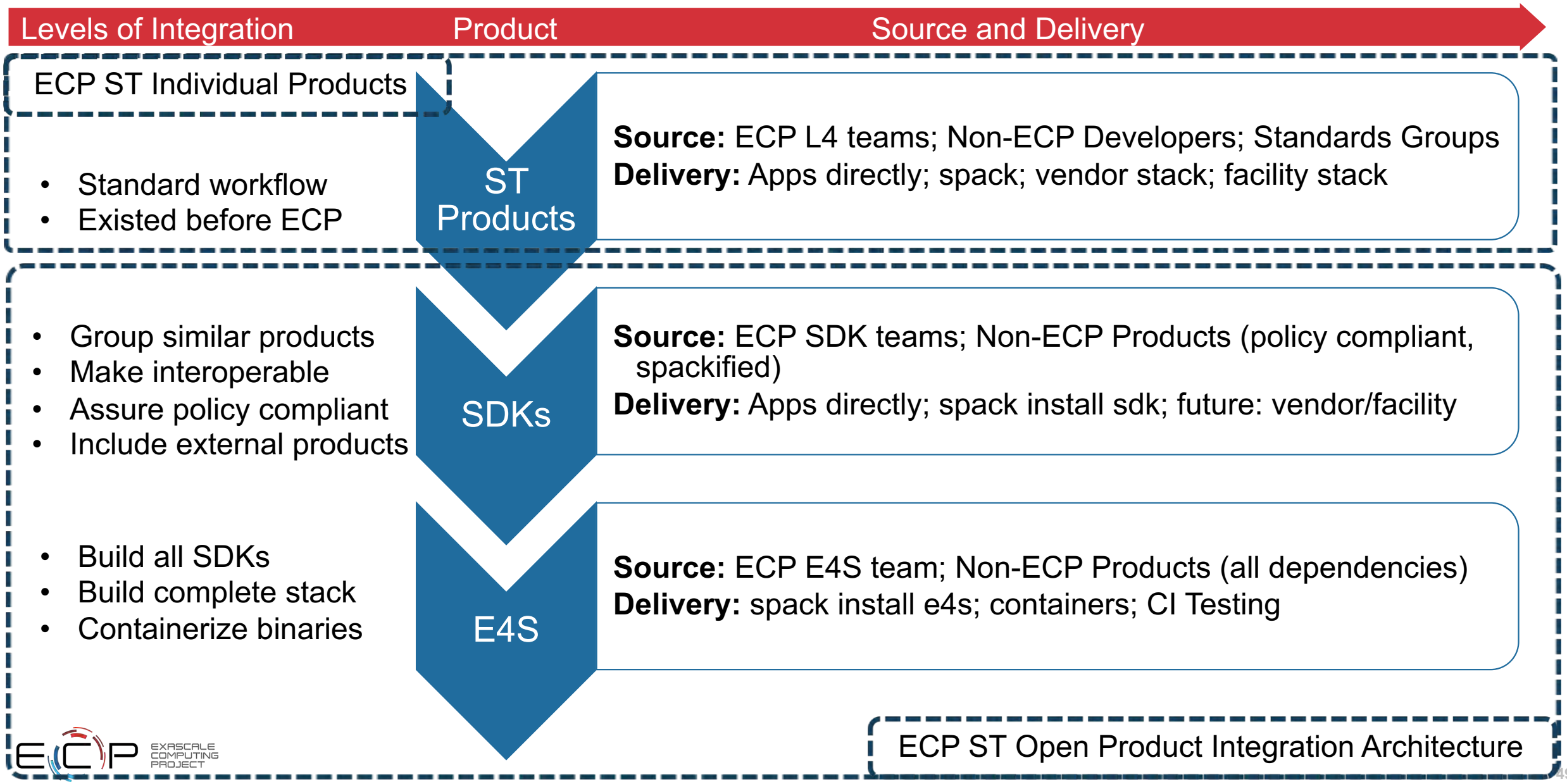
- E4S/SDK collaborations make sense with:
  - HPC open source development projects:
    - deal.II (NSF math library),
    - PHIST (DLR Germany library).
  - Application teams in search of open source software foundation:
    - NSF science teams.
    - NOAA, others.
  - Commercial open source packagers/distributors
    - OpenHPC.
    - HPC systems vendors.
  - HPC systems facilities:
    - SDSC, TACC, others.

E4S: Building on top of previous efforts

- E4S did not emerge from nothing.
- Leveraging the work of many others.
- HPC Linux: Work done at U of Oregon, and at ParaTools.
- IDEAS-Classic: xSDK – the original SDK continuing under ECP.
- Spack – Pre-dates E4S.

E4S Wrap Up

Software Technology Ecosystem



E4S Basics

What E4S is not

- A closed system taking contributions only from DOE software development teams.
- A monolithic, take-it-or-leave-it software behemoth.
- A commercial product.
- A simple packaging of existing software.

What E4S is

- Extensible, open architecture software ecosystem accepting contributions from US and international teams.
- Framework for collaborative open-source product integration.
- A full collection if compatible software capabilities **and**
- A manifest of a la carte selectable software capabilities.
- Vehicle for delivering high-quality reusable software products in collaboration with others.
- 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.

Final Remarks

- Software Development Kits (SDKs) provide a new software aggregation mechanism:
  - Do not significantly increase the number of software products in the HPC ecosystem.
  - Provide intermediate build, install and test targets for E4S (reducing complexity).
  - Enable development teams to improve and standardize look-and-feel, best practices, policies.
  - Improve interoperability, interchangeability of similar products.
  - Fosters communities of developers in a cooperative/competitive environment.
  - **Provides integration point for SDK-compatible, non-ECP products.**
- The Extreme-scale Scientific Software Stack (E4S) provides a complete HPC software stack:
  - Does not significantly increase the number of software products in the HPC ecosystem.
  - Provides a coordinated approach to building, installing and testing HPC software.
  - Tests some products with a subset of configurations on a subset of platforms.
  - Improves stability of the ST stack so that any subset is more stable.
  - **Provides a complete software stack, including non-ECP products.**
- **The SDK/E4S architecture is open:**
  - Enables light-weight coordinated collaboration among open source software teams.
  - ECP seeks collaboration: libraries/tools and SDKs, facilities and E4S.