Part 1: The Extreme-scale Scientific Software Stack for Collaborative Open Source Software,

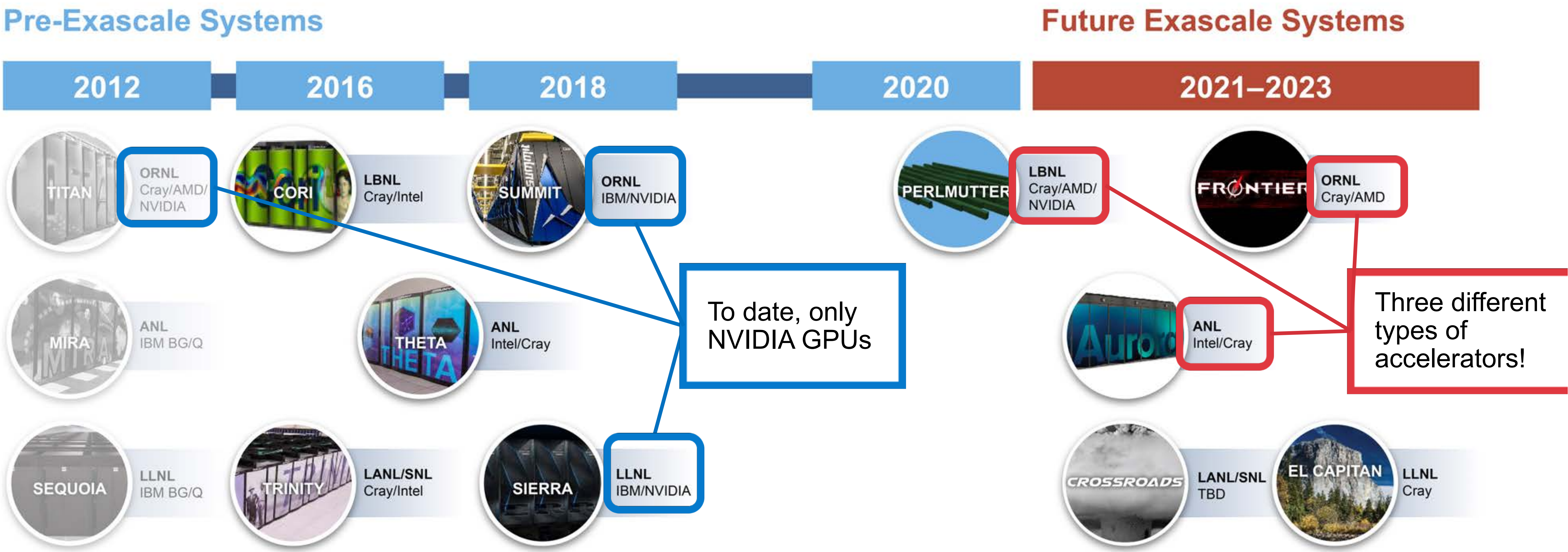
Part 2: HPCG Benchmark

Michael A. Heroux, Sandia National Laboratories
Director of Software Technology, US Exascale Computing Project

Advanced Computing Working Group
August 18, 2020

Department of Energy (DOE) Roadmap to Exascale Systems

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



Arm/SVE: Not an explicit ECP target but support is being developed concurrently in the same software stack via other funding.

EXASCALE COMPUTING PROJECT

ECP Software Technology (ST) is one of three focus areas

Performant mission and science applications @ scale

Aggressive RD&D Project	Mission apps & integrated S/W stack	Deployment to DOE HPC Facilities	Hardware tech advances

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

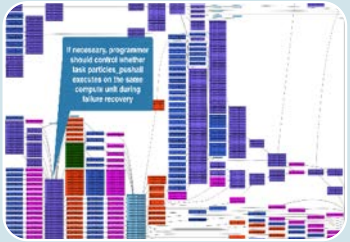
70 unique software products spanning programming models and run times, math libraries, data and visualization

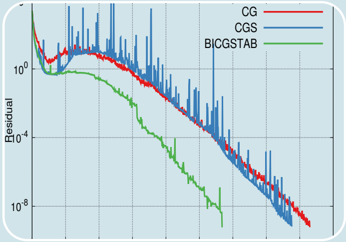
Hardware and Integration (HI)

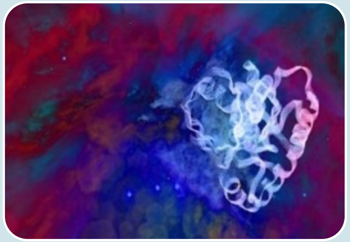
Integrated delivery of ECP products on targeted systems at leading DOE HPC facilities

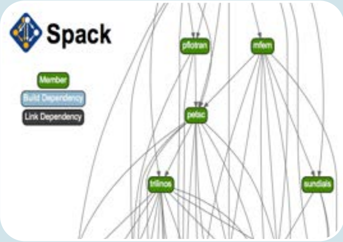
6 US HPC vendors focused on exascale node and system design; application integration and software deployment to facilities

ECP ST has six technical areas









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.

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 experience in high-performance numerical software, including development 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 (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.

ECP ST Subprojects

- WBS
- Name
- PIs
- Project Managers (PMs)

ECP ST Stats

- 33 L4 subprojects
- 10 PI/PC same
- 23 PI/PC different

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	Smith, Barry	Munson, Todd
2.3.3.07	STRUMPACK/SuperLU/FFTX: sparse direct solvers, preconditioners, and FFT libraries	Li, Sherry	Li, Xiaoye
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
2.3.6	**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

We work on products and applications needed 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 Libs: Kokkos, RAJA	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.

The Extreme-Scale Scientific Software Stack (E4S):

A collaborative HPC Linux Ecosystem

EXASCALE COMPUTING PROJECT

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

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

- Group similar products
- Make interoperable
- Assure policy compliant
- Include external products

SDKs

Source: ECP SDK teams; Non-ECP Products (policy compliant, spackified)
Delivery: Apps directly; spack install sdk; future: vendor/facility

- Build all SDKs
- Build complete stack
- Containerize binaries

E4S

Source: ECP E4S team; Non-ECP Products (all dependencies)
Delivery: spack install e4s; containers; CI Testing

ECP ST Open Product Integration Architecture

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

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

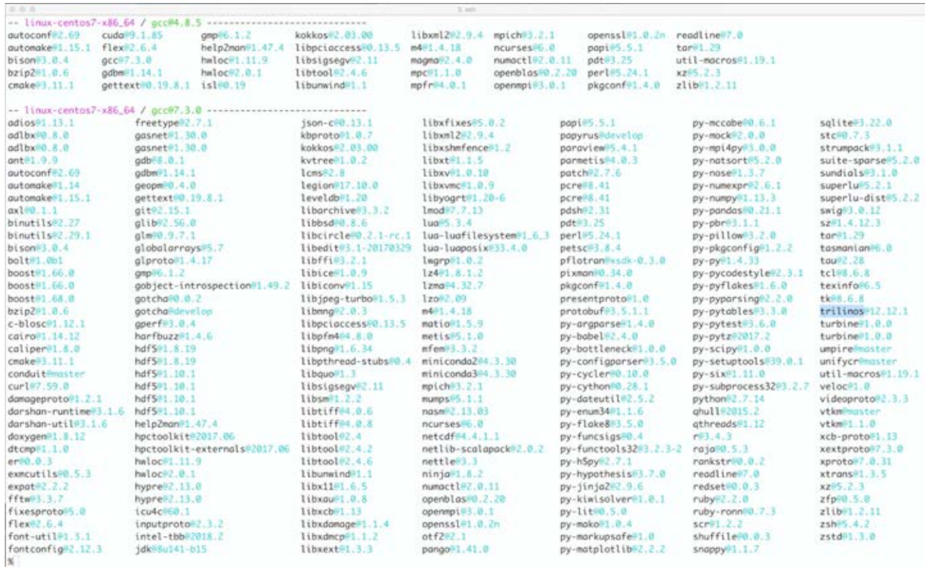
- Oct 2018: E4S 0.1 - 24 full, 24 partial release products
- Jan 2019: E4S 0.2 - 37 full, 10 partial release products
- Nov 2019: E4S 1.0 - 50 full, 5 partial release products
- Feb 2020: E4S 1.1 - 50 full,10 partial release products

e4s.io

Lead: Sameer Shende (U Oregon)

E4S 1.1 Full Release: 50 ECP Packages and all dependencies

- Adios
- AML
- Argobots
- Bolt
- Caliper
- Darshan
- Dyninst
- Faodel
- Flecsi
- Gasnet
- GEOPM
- Gotcha
- HDF5
- HPCToolkit
- Hypre
- Kokkos
- Legion
- Libnrm
- Libquo
- Magma
- Mercury
- MFEM
- MPICH
- MPIFileUtils
- Ninja
- OpenMPI
- PAPI
- Papyrus
- Parallel netCDF
- PDT
- PETSc
- Qthreads
- Raja
- Rempi
- SCR
- Spack
- Strumpack
- Sundials
- SuperLU
- SZ
- Tasmanian
- TAU
- Trilinos
- Turbine



Packages installed using Spack

- Umpire
- UnifyFS
- UPC++ Veloc
- Zfp

All ST products will be released through E4S

Partial (in-progress) Release: ASCENT, Catalyst, Flang, libEnsemble, LLVM, Visit, others

E4S Software Development Kits (SDKs):

Collaborative community development of complementary software capabilities

Software Development Kits (SDKs): Key delivery vehicle for ECP

A collection of related software products (packages) where coordination across package teams improves usability and practices, and foster community growth among teams that develop similar and complementary capabilities

- **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

ECP ST SDKs: Grouping similar products for collaboration & usability

Programming Models & Runtimes Core

Tools & Technologies

Compilers & Support

Math Libraries (xSDK)

Viz Analysis and Reduction

Data mgmt., I/O Services & Checkpoint/ Restart

"Unity in essentials, otherwise diversity"

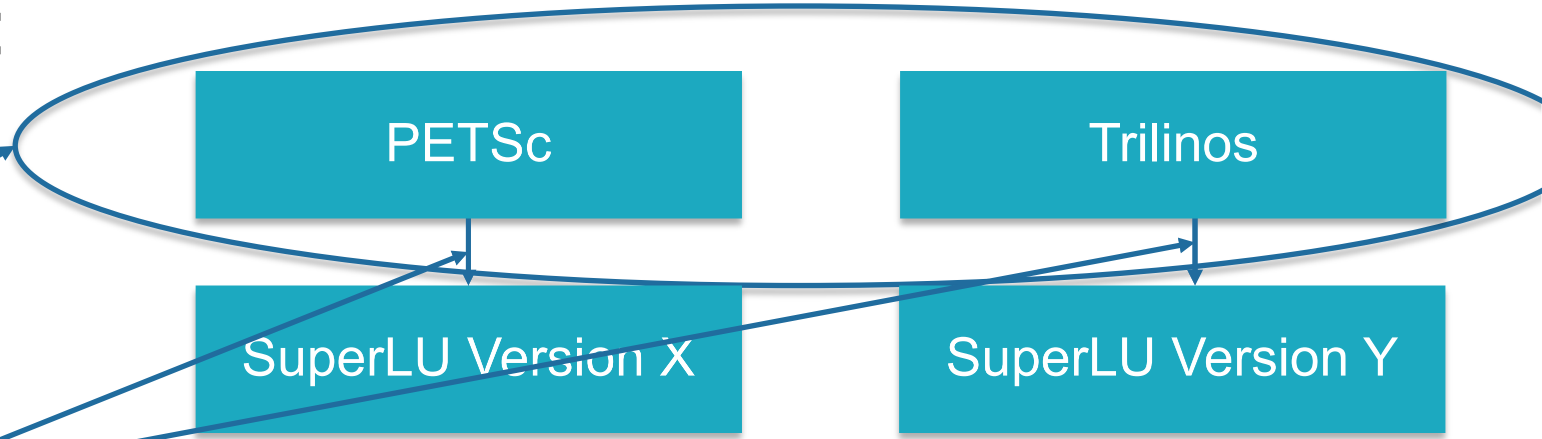
xSDK version 0.5: November 2019
(21 math libs, 2 domain-specific packages)

- AMReX
- ButterflyPACK
- DTK
- deal.ii
- Ginkgo
- hypre
- libEnsemble
- MAGMA
- MFEM
- Omega_h
- PETSc/TAO
- PHIST
- PLASMA
- preCICE
- PUMI
- SLEPc
- STRUMPACK
- SUNDIALS
- SuperLU
- Tasmanian
- Trilinos
- Pflotran
- Alquimia

Notes:

- Growth:
  - 5 in release 0.1.
  - 7 in 0.2
  - 9 in 0.3
  - 19 in 0.4
  - 23 in 0.5
- You do not need to build all packages.
- We build and test all packages.
- Any subset is guaranteed to build if using the same build parameters, platforms.
- Similar builds should work or require less effort for success.

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.

xSDK community policies

https://xsdk.info/policies

<u>xSDK compatible package</u>: 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.

<u>xSDK member package</u>: 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 <u>your</u> software?

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

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						

Legend

- PMR
- Tools
- Math Libraries
- Data and Vis
- Ecosystems and delivery

E4S/SDK Policy Initiative Status: Summer 2020

- Each SDK community has developed its own set of policies similar to Math Libs (xSDK).
- Policies common to all SDKs will be promoted to E4S level.
- Policies will be used to determine quality label and membership in E4S and the SDKs.
- Version 1.0 of policies due by end of 2020.

E4S DocPortal

A Single Portal with Redirect to Product Documentation

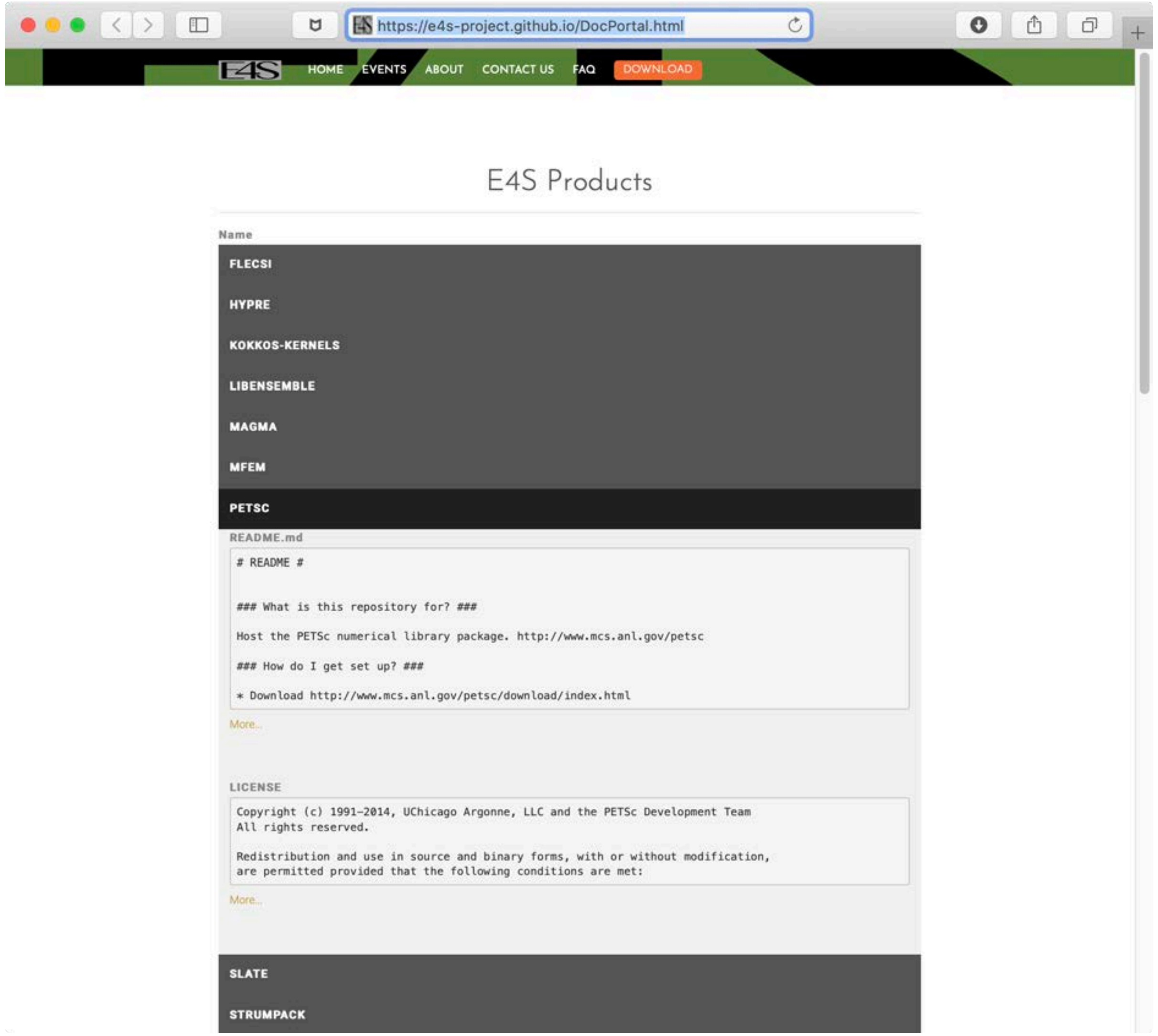
ECP EXASCALE COMPUTING PROJECT

E4S DocPortal

- Provide a single online location for *accurate* product descriptions for ECP software products.
- Derived requirements:
  - Sustainable: Must be integrated into software team workflows.
  - Incremental: Must build on community approaches to providing this kind of information.
  - Extensible: Must be usable by any open source software team.
- Strategy:
  - Use the open source community approach of specially-name files in software repositories.
  - Adopt commonly used file names when available.
  - Identify new information items not already being requested.
  - Develop new special file names for information beyond what is already captured.
  - Create web-based raking tool to capture information from product repositories and present in summary form on a webpage.
  - Aggregates and summarizes documentation and metadata for E4S products
  - Regularly updates information directly from product repositories
  - Prototype: https://e4s-project.github.io/DocPortal.html

EXASCALE COMPUTING PROJECT

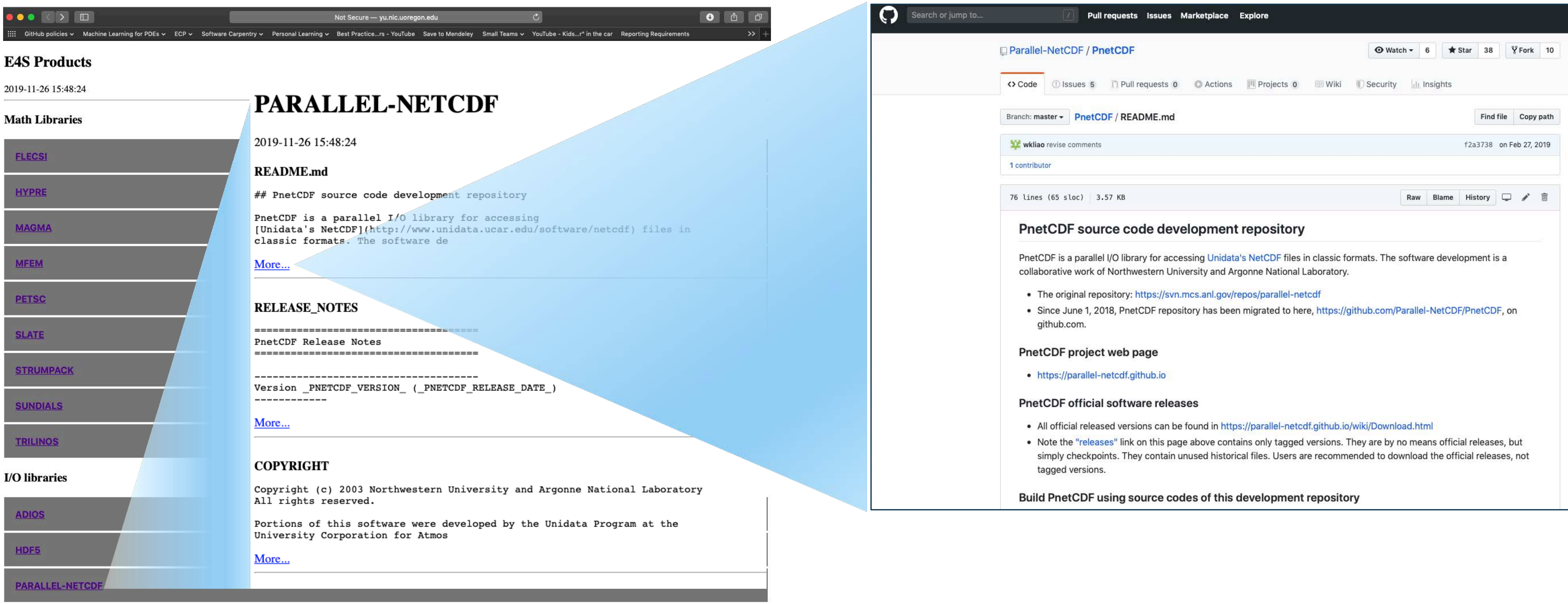
E4S DocPortal



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

EXASCALE COMPUTING PROJECT

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



Prototype: http://yu.nic.uoregon.edu/~wspear/E4S/E4S-Products.html
Will involve Facilities Staff in Requirements/Design as we further prototyping.

E4S DocPortal Initiative Status: Summer 2020

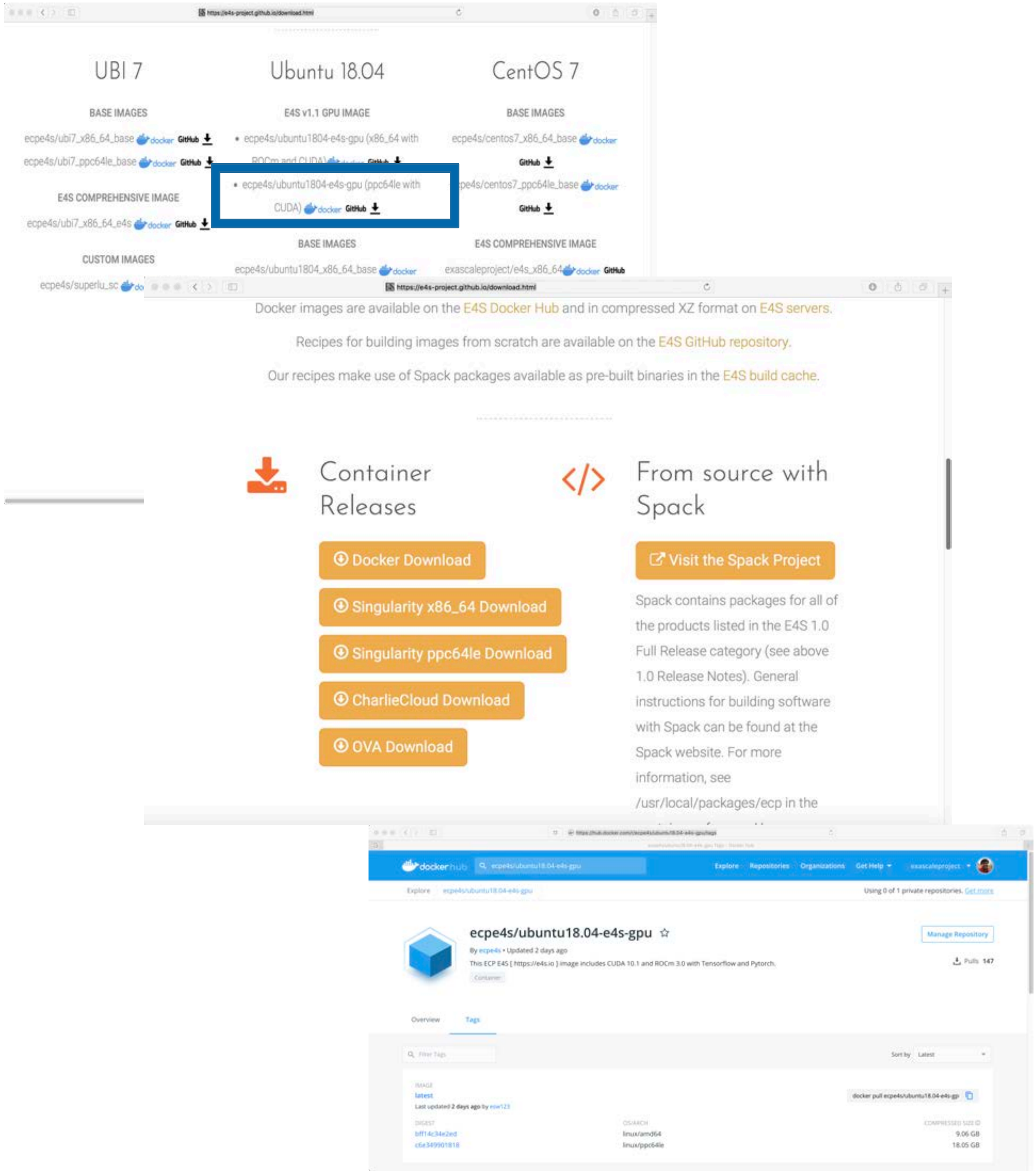
- Completed DocPortal Prototype and Design Document.
- Reviewed prototype/design with Facilities and ST developers.
- Deploying version 1.0 now.

E4S Software Access

From Source and Many Other Ways

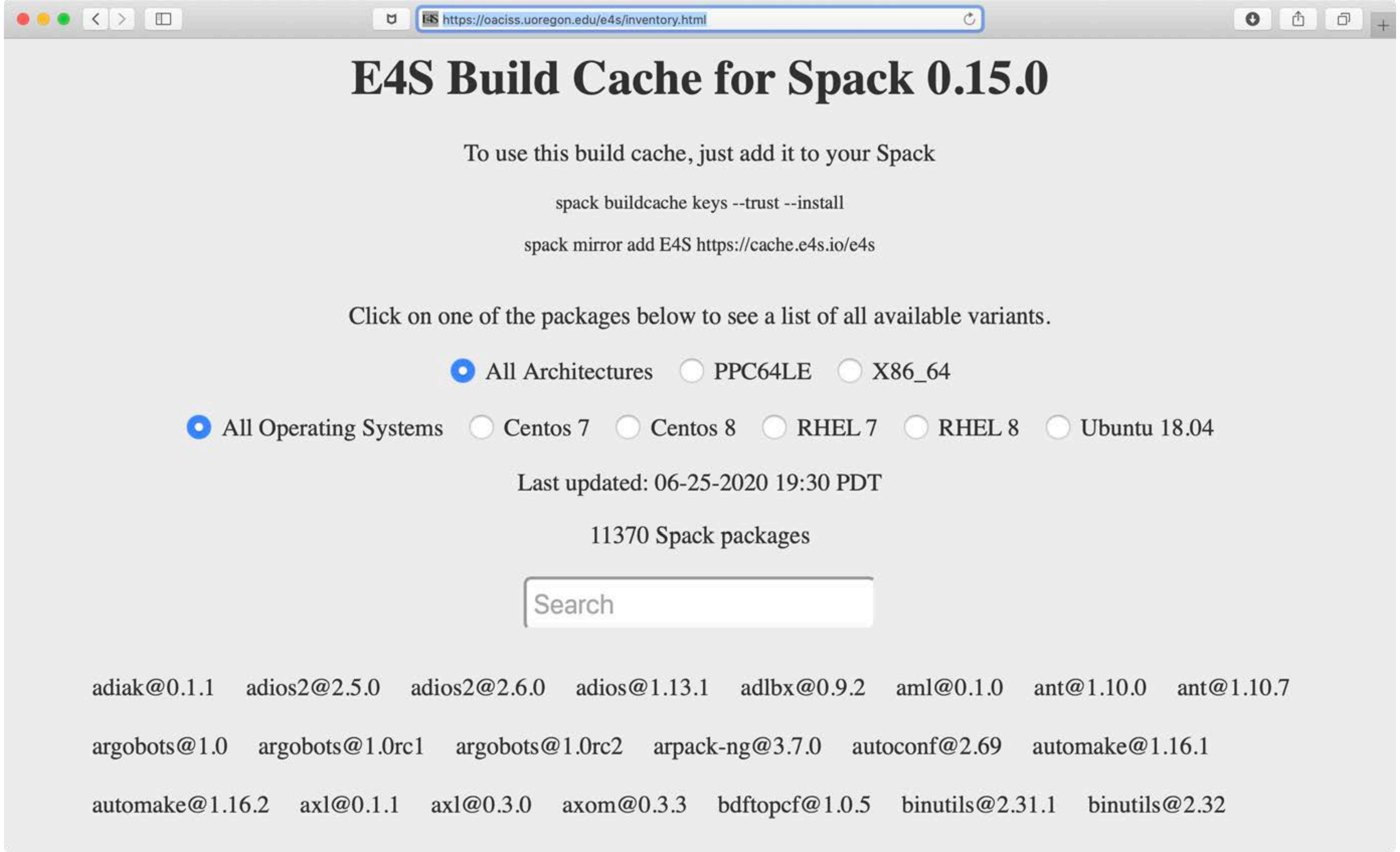
E4S v1.1 Access

- From e4s-project.github.io
  - Source via Spack
  - E4S v1.1 GPU image
  - Docker, Singularity (ppc64le, x86_64) …
- E4S v1.1 Release Available at DockerHub
  - 40+ ECP ST Products
  - Support for GPUs
    - NVIDIA (CUDA 10.1.243)
    - ppc64le and x86_64
- Visit https://e4s.io for more details



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

EXASCALE COMPUTING PROJECT

New E4S Spack Cache Website (under development)

E4S Spack Cache

Search Packages

(●) All ( ) Ubuntu 18.04 ( ) RHEL 7 ( ) RHEL 8 ( ) CentOS 7 ( ) CentOS 8

(●) All ( ) X86_64 ( ) PPC64LE

Showing 306 packages representing 8607 binaries

adiak
Adiak collects metadata about HPC application runs and provides it to tools.

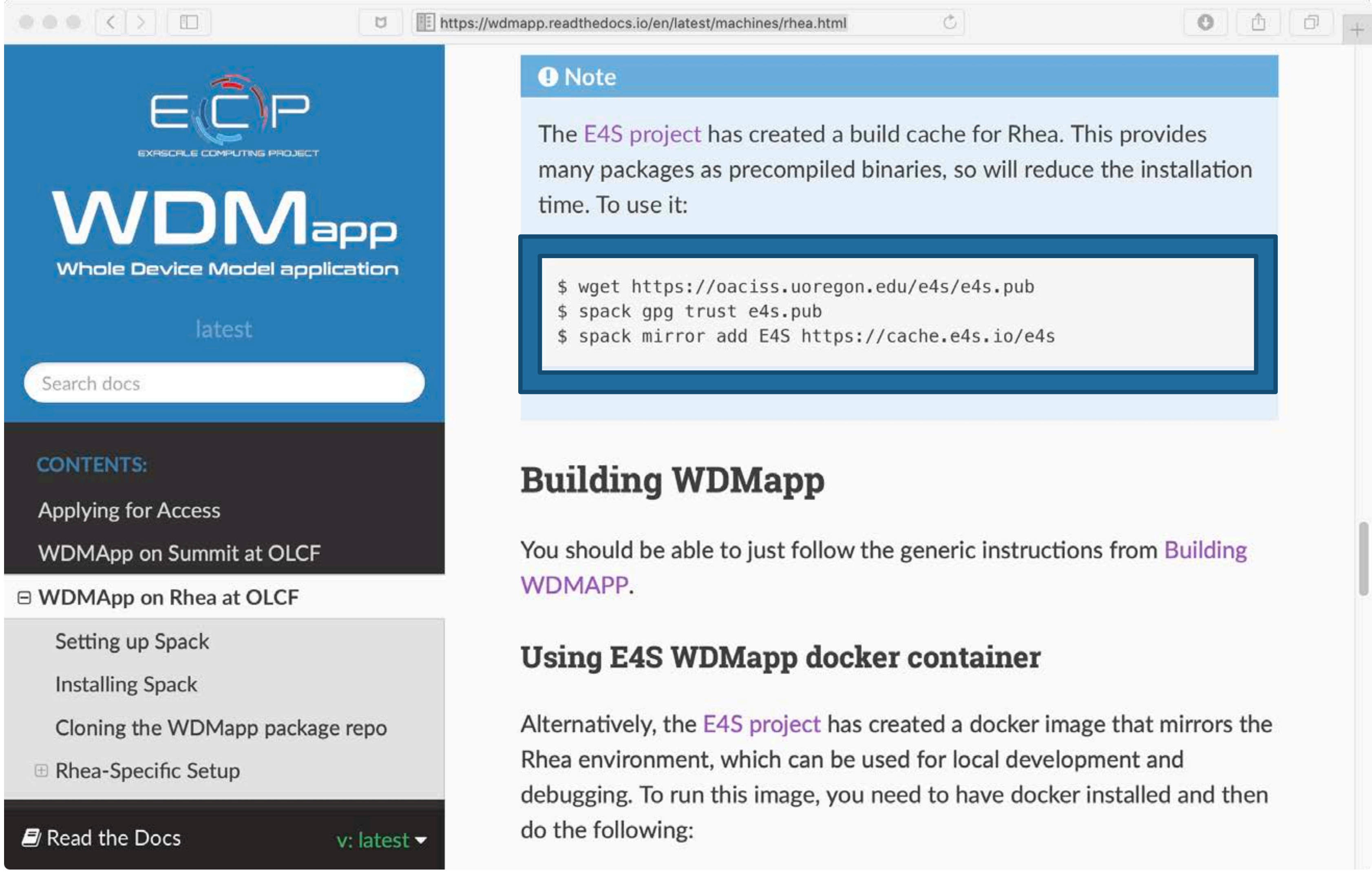
adios
The Adaptable IO System (ADIOS) provides a simple, flexible way for scientists to describe the data in their code that may need to be written, read, or processed outside of the running simulation.

adios2
The Adaptable Input Output System version 2, developed in the Exascale Computing Program

adlbx
ADLB/X: Master-worker library + work stealing and data dependencies

aml

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


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

EXASCALE COMPUTING PROJECT

WDMapp
Whole Device Model application

latest

Search docs

CONTENTS:

Applying for Access
WDMApp on Summit at OLCF
WDMApp on Rhea at OLCF
Setting up Spack
Installing Spack
Cloning the WDMapp package repo
Rhea-Specific Setup

Read the Docs v: latest

Note

The E4S project has created a build cache for Rhea. This provides many packages as precompiled binaries, so will reduce the installation time. To use it:

```
$ wget https://oaciss.uoregon.edu/e4s/e4s.pub
$ spack gpg trust e4s.pub
$ spack mirror add E4S https://cache.e4s.io/e4s
```

Building WDMapp

You should be able to just follow the generic instructions from Building WDMAPP.

Using E4S WDMapp docker container

Alternatively, the E4S project has created a docker image that mirrors the Rhea environment, which can be used for local development and debugging. To run this image, you need to have docker installed and then do the following:

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

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

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, Inc.
- IDEAS-Classic: xSDK – the original SDK continuing under ECP.
- Spack – Pre-dates E4S.

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.

E4S Collaborations

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).
  - Commercial open source packagers/distributors:
    - OpenHPC.
    - HPC systems vendors.
  - HPC systems facilities:
    - SDSC, TACC, others.
  - Other organizations in search of collaborative open source software foundation:
    - NSF science teams.
    - NOAA, others.
    - International collaborators.

Some UK-ECP Collaboration Models

Approach	Comments/Potential
Participate in ECP-related tutorials and webinars	Many ST technologies offer tutorial/webex forums to learn more; range from introductory to advanced
Develop *de facto* and ISO standards	MPI, OpenMP, C++, Fortran, PAPI, BLAS: Happening, more is better.
Evaluate/prototype new capabilities using ECP software products	**Accelerator-enabled software stack** (compilers, programming environments, tools, math libraries, in situ), **next-generation IO** (HDF5, ADIOS, PNetCDF)
Adopt and rely upon ECP software (as an option)	A goal for us: Want to explore how to make this possible. Collaboration can help us improve our product development and delivery.
Investment in E4S, Spack, SDKs	E4S, Spack and SDKs are open architectures enabling light-weight product coupling, improved user experience.
Overall	Two way interactions benefit everyone.

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

ECP Software Technology Capability Assessment Report (CAR) Version 2.0

- Comprehensive document about ECP ST structure, progress and planning.
- Version 2.0 – extensive revision:
  - FY20 – 23 Organization.
  - Planning, Execution, Tracking & Assessment processes.
  - E4S/SDK details.
  - 2-page writeups for each product development effort.
  - Released February 1, 2020.

https://exascaleproject.org/wp-content/uploads/2020/02/ECP-ST-CAR-V20-1.pdf

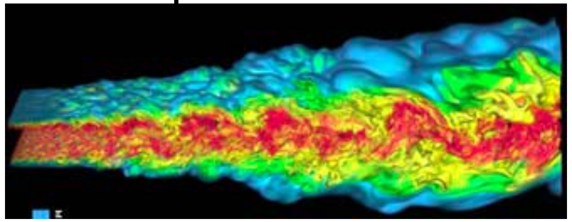
THE HPCG BENCHMARK

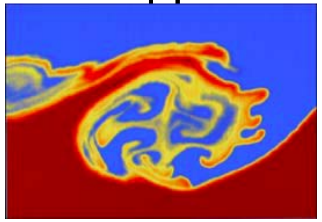
Michael Heroux

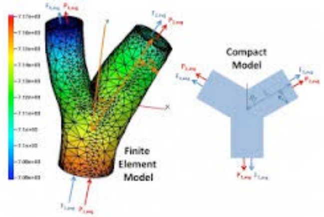
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

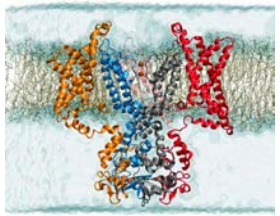
HPCG Benchmark Goals

- Augment the TOP500 listing with a benchmark that correlates with important scientific and technical apps not well represented by HPL









- Encourage vendors to focus on architecture features needed for high performance on those important scientific and technical apps.
  - Stress a balance of floating point and communication bandwidth and latency
  - Reward investment in high performance collective ops
  - Reward investment in high performance point-to-point messages of various sizes
  - Reward investment in local memory system performance
  - Reward investment in parallel runtimes that facilitate intra-node parallelism
- Provide an outreach/communication tool
  - Easy to understand
  - Easy to optimize
  - Easy to implement, run, and check results

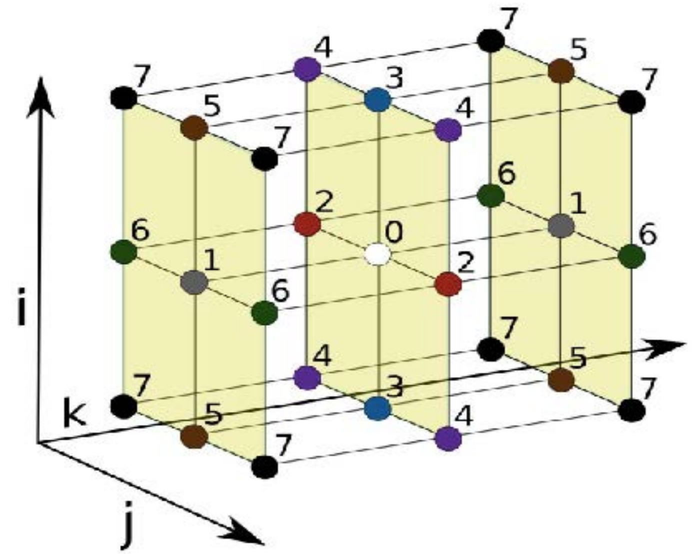
HPCG Snapshot

- High Performance Conjugate Gradients (HPCG).
- Solves $Ax=b$, A large, sparse, b known, x computed.
- An optimized implementation of PCG contains essential computational and communication patterns that are prevalent in a variety of methods for discretization and numerical solution of PDEs

- Patterns:
  - Dense and sparse computations.
  - Dense and sparse collectives.
  - Multi-scale execution of kernels via MG (truncated) V cycle.
  - Data-driven parallelism (unstructured sparse triangular solves).
- Strong verification (via spectral & symmetry properties of PCG).

Model Problem Description

- Synthetic discretized 3D PDE (FEM, FVM, FDM).
- Zero Dirichlet BCs, Synthetic RHS s.t. solution = 1.
- Local domain: $(n_x \times n_y \times n_z)$
- Process layout: $(np_x \times np_y \times np_z)$
- Global domain: $(n_x * np_x) \times (n_y * np_y) \times (n_z * np_z)$
- Sparse matrix:
  - 27 nonzeros/row interior.
  - 8 – 18 on boundary.
  - Symmetric positive definite.



27-point stencil operator

HPCG Design Philosophy

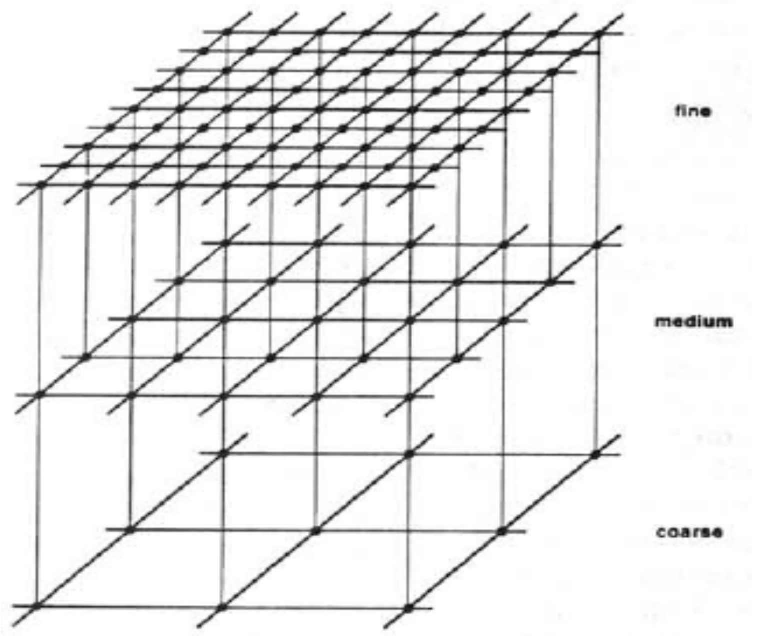
- Relevance to broad collection of important apps.
- Simple, single number.
- Few user-tunable parameters and algorithms:
  - The system, not benchmarker skill, should be primary factor in result.
  - Algorithmic tricks don’t give us relevant information.
- Algorithm (PCG) is vehicle for organizing:
  - Known set of kernels.
  - Core compute and data patterns.
  - Tunable over time (as was HPL).
- Easy-to-modify:
  - _ref kernels called by benchmark kernels.
  - User can easily replace with custom versions.
  - Clear policy: Only kernels with _ref versions can be modified.

PCG ALGORITHM

- $p_0 := x_0$, $r_0 := b\text{-}Ap_0$
- Loop $i = 1, 2, \ldots$
  - $z_i := M^{-1} r_{i-1}$
  - if $i = 1$
    - $p_i := z_i$
    - $\alpha_i :=$ `dot_product`(r_{i-1}, z)
  - else
    - $\alpha_i :=$ `dot_product`(r_{i-1}, z)
    - $\beta_i := \alpha_i / \alpha_{i\text{-}1}$
    - $p_i := \beta_i * p_{i-1} + z_i$
  - end if
  - $\alpha_i :=$ `dot_product`(r_{i-1}, z_i) /`dot_product`$(p_i, A*p_i)$
  - $x_{i+1} := x_i + \alpha_i * p_i$
  - $r_i := r_{i-1} - \alpha_i * A * p_i$
  - if $||r_i||_2 <$ tolerance then Stop
- end Loop

Preconditioner

- Hybrid geometric/algebraic multigrid:
  - Grid operators generated synthetically:
    - Coarsen by 2 in each x, y, z dimension (total of 8 reduction each level).
    - Use same GenerateProblem() function for all levels.
  - Grid transfer operators:
    - Simple injection. Crude but…
    - Requires no new functions, no repeat use of other functions.
    - Cheap.
  - Smoother:
    - Symmetric Gauss-Seidel [ComputeSymGS()].
    - Except, perform halo exchange prior to sweeps.
    - Number of pre/post sweeps is tuning parameter.
  - Bottom solve:
    - Right now just a single call to ComputeSymGS().
    - If no coarse grids, has identical behavior as HPCG 1.X.



- Symmetric Gauss-Seidel preconditioner
  - In Matlab that might look like:

```
LA = tril(A); UA = triu(A); DA = diag(diag(A));

x = LA\y;
x1 = y - LA*x + DA*x; % Subtract off extra
      diagonal contribution
x = UA\x1;
```

Meta-computations: HPCG Benchmark

- Exploit two properties:
  - Spectral properties of CG:
    - Eigenvalue clustering.
    - CG convergence related to number of *distinct* eigenvalues.
  - Operator symmetry:
    - Compact Finite Difference operator is symmetric.
    - Multigrid is symmetric.

Symmetry Test: HPCG Benchmark

- Symmetry:
  - For any linear operator A, $x^T A y = y^T A^T x$.
  - If A symmetric $A = A^T$, so $x^T A y = y^T A x$.
  - And $\mathbf{x^T A y - y^T A x = 0}$.
- HPCG computes the above expression for:
  - User matrix and the preconditioner.
  - Numerical detail: Need to scale by vector & matrix norms.

Spectral properties test: HPCG Benchmark

- Eigenvalue clustering:
  - HPCG matrix is 27-point finite difference stencil.
    - -1 off diagonals, diagonally dominant, zero Dirichlet BCs.
    - Max diagonal value – 27.
  - Idea: Temporarily replace diagonal values.
    - For i=1:9 A(i,i) = (i+1)*1.0E6
    - For i>9 A(i,i) = 1.0E6
- Questions:
  - How many distinct diagonal values?
  - How many unpreconditioned CG iterations?
  - How many preconditioned CG iterations?

Merits of HPCG

- Includes major communication/computational patterns.
  - Represents a minimal collection of the major patterns.
- Rewards investment in:
  - High-performance collective ops.
  - Local memory system performance.
  - Low latency cooperative threading.
- Detects/measures variances from bitwise reproducibility.
- Executes kernels at several (tunable) granularities:
  - nx = ny = nz = 104 gives
  - nlocal = 1,124,864; 140,608; 17,576; 2,197
  - ComputeSymGS with multicoloring adds one more level:
    - 8 colors.
    - Average size of color = 275.
    - Size ratio (largest:smallest): 4096
  - Provide a "natural" incentive to run a big problem.
- Full performance discussion:
  - http://www.hpcg-benchmark.org -> "Performance Overview" tab.

HPL vs. HPCG: Bookends

- Some see HPL and HPCG as “bookends” of a spectrum.
  - Applications teams know where their codes lie on the spectrum.
  - Can gauge performance on a system using both HPL and HPCG numbers.
- Problem of HPL execution time still an issue:
  - Need a lower cost option. End-to-end HPL runs are too expensive.
  - Work in progress.

ISC20 HPCG Rankings 1 – 5

HPCG Rank	HPL Rank	System	Rmax (TFlop/s)	HPCG (TFlop/s)
1	1	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	415,530.0	13366.40
2	2	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	148,600.0	2925.75
3	3	**Sierra** - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	94,640.0	1795.67
4	6	**HPC5** - PowerEdge C4140, Xeon Gold 6252 24C 2.1GHz, NVIDIA Tesla V100, Mellanox HDR Infiniband, Dell EMC Eni S.p.A. Italy	35,450.0	860.32
5	11	**Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect, Cray/HPE DOE/NNSA/LANL/SNL United States	20,158.7	546.12

ISC20 HPCG Rankings 6 – 10

HPCG Rank	HPL Rank	System	Rmax (TFlop/s)	HPCG (TFlop/s)
6	7	**Selene** - DGX A100 SuperPOD, AMD EPYC 7742 64C 2.25GHz, NVIDIA A100, Mellanox HDR Infiniband, Nvidia NVIDIA Corporation United States	27,580.0	509.39
7	12	**AI Bridging Cloud Infrastructure (ABCI)** - PRIMERGY CX2570 M4, Xeon Gold 6148 20C 2.4GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR, Fujitsu National Institute of Advanced Industrial Science and Technology (AIST) Japan	19,880.0	508.85
8	10	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100, Cray/HPE Swiss National Supercomputing Centre (CSCS) Switzerland	21,230.0	496.98
9	4	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, NRCPC National Supercomputing Center in Wuxi China	93,014.6	480.85
10	17	**Nurion** - Cray CS500, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path, Cray/HPE Korea Institute of Science and Technology Information South Korea	13,929.3	391.45

EXASCALE COMPUTING PROJECT