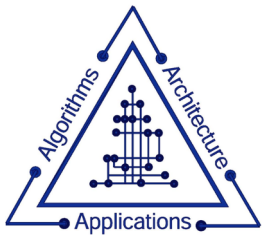
E4S: Toward an Ecosystem for HPC R&D Software

Michael A. Heroux

Senior Scientist, Sandia National Laboratories

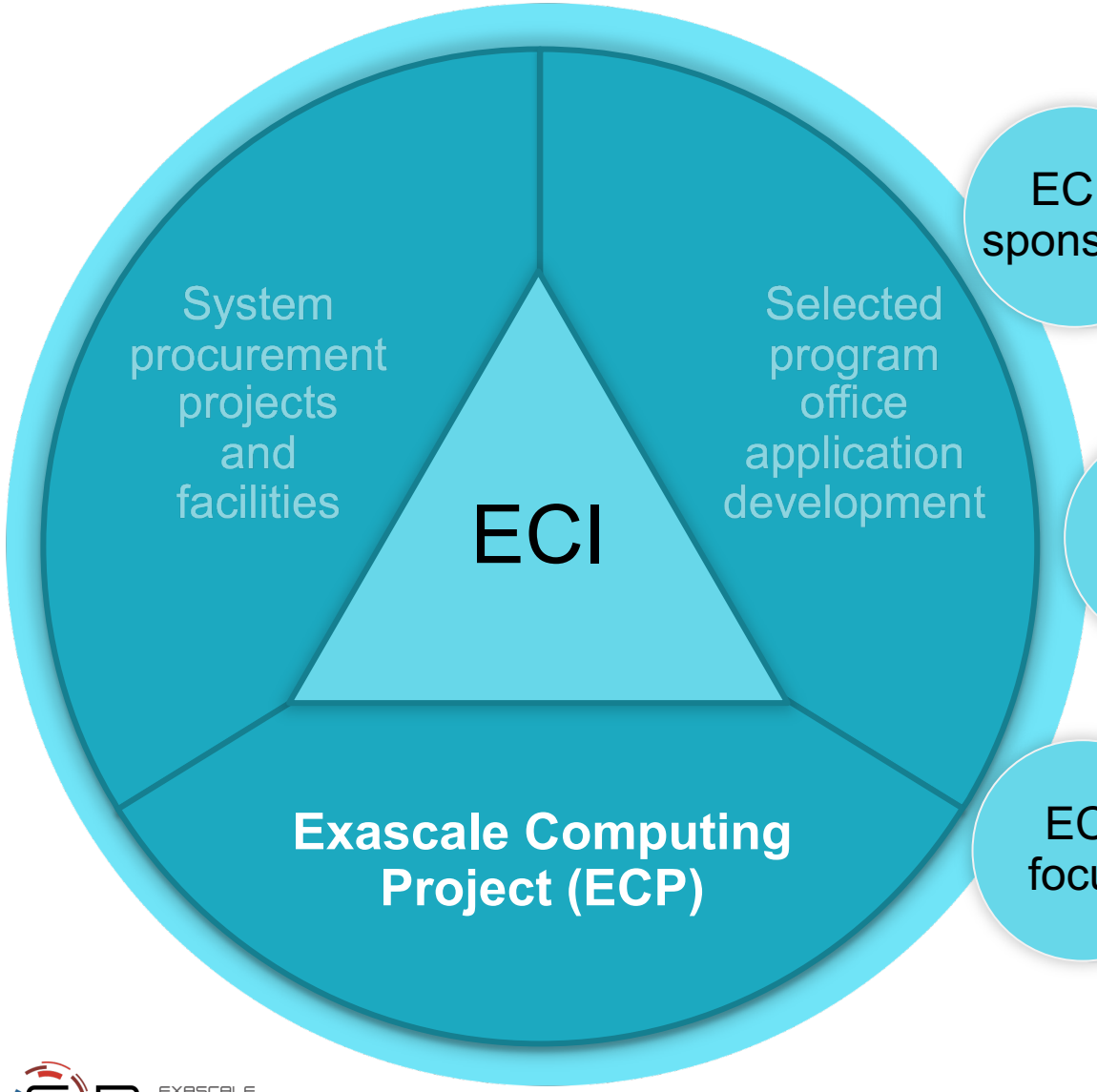
Director of Software Technology, US Exascale Computing Project



40TH ANNIVERSARY LECTURE SERIES

Brief Intro to the Exascale Computing Project (ECP)

DOE Exascale Computing Initiative (ECI)



ECI sponsors

DOE Office of Science (SC);
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 wid of applications of importance to DOE

ECP's holistic approach uses co-design and integration to achieve exascale computing

Performant mission and science applications at scale

Aggressive RD&D project	Mission apps; integrated S/W stack	Deployment to DOE HPC Facilities	Hardware technology advances

Application Development (AD)

Develop and enhance the predictive capability of applications critical to DOE

24 applications
National security, energy, Earth systems, economic security, materials, data

6 Co-Design Centers
Machine learning, graph analytics, mesh refinement, PDE discretization, particles, online data analytics

Software Technology (ST)

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

71 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 EXASCALE COMPUTING PROJECT

DOE HPC Roadmap to Exascale Systems

2012-2020

2021-2023

decommissioned

ORNL
Cray/AMD/NVIDIA

ANL
IBM BG/Q

ANL
Cray/Intel KNL

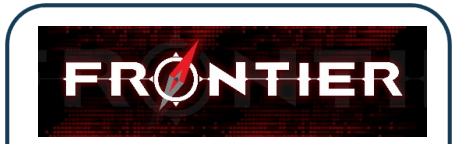
LBNL
Cray/Intel Xeon/KNL

LLNL
IBM BG/Q

LANL/SNL
Cray/Intel Xeon/KNL

ORNL
IBM/NVIDIA

LLNL
IBM/NVIDIA



ORNL
HPE/AMD

Exascale Systems

ANL
Intel/HPE

LBNL
HPE/AMD/NVIDIA

LANL/SNL
HPE/Intel

EXASCALE COMPUTING PROJECT

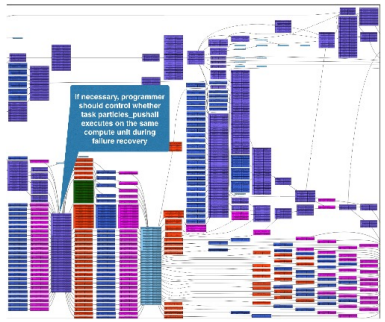
Brief Intro to ECP Software Technology (ST) Focus Area

ECP **Software Technology (ST)**

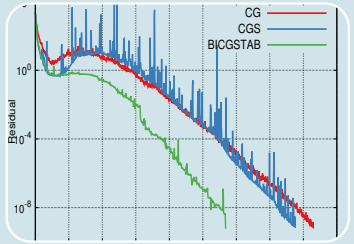
Goal

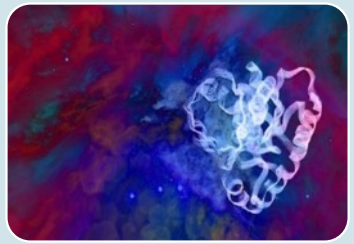
Build a comprehensive, coherent software stack that enables application developers to productively develop highly parallel applications that effectively target diverse exascale architectures

- Prepare SW stack for scalability with massive on-node parallelism
- Extend existing capabilities when possible, develop new when not
- Guide, and complement, and integrate with vendor efforts
- Develop and deliver high-quality and robust software products



ECP ST has six technical areas





E4S

NNSA
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

We work on products applications need now and into the future

Key themes:
- Focus: GPU node architectures and advanced memory & storage technologies
- Create: New high-concurrency, latency tolerant algorithms
- Develop: New portable (Nvidia, Intel, AMD GPUs) software product
- Enable: Access and use via standard APIs

Software categories:
- **Next generation established products:** Widely used HPC products (e.g., MPICH, OpenMPI, PETSc)
- **Robust emerging products:** Address key new requirements (e.g., Kokkos, RAJA, Spack)
- **New products:** 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 techn
IO: HDF5, MPI-IO, ADIOS	Standard and next-gen IO, leveraging non-volatile storag
Viz/Data Analysis	ParaView-related product development, node concurren

The Extreme-Scale Scientific Software Stack (E4S) and Software Development Kits (SDKs)

Core questions E4S is addressing

How can new ECP software capabilities be effectively and efficiently integrated and sustained?

- ECP success requires development, delivery and use of new GPU capabilities in 70 products
- Requires coordination of versioning, integration, testing, debugging, interaction with vendors and facilities
- Requires access to new documentation
- Requires focus on high quality

How can E4S build upon, leverage and extend existing capabilities and activities?

- Using Spack for product installation, leveraging growing Spack capabilities
- Making E4S available via containers, cloud platforms
- Providing integration pathways to multiple destinations: from-source, LLVM, vendor stacks, facilities, etc

How can E4s become a sustainable, open, collaborative software ecosystem for HPC?

- Hierarchical, open architecture to accept and manage community contributions
- Defined processes for community engagement within DOE, with other US agencies, industry, international partners
- Delivering the value proposition of the ecosystem vs each app r
  dependencies

ECP applications rely on ST products across all technical areas

24 ECP applications: National security, energy, Earth systems, economic security, materials, data
6 co-design centers: machine learning, graph analytics, mesh refinement, PDE discretization, particles, online data analytics

Consider ECP software technologies needed by 5 ECP applications:

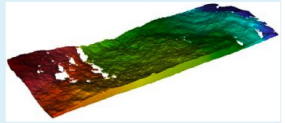
ExaWind: Turbine Wind Plant Efficiency

Harden wind plant design and layout against energy loss susceptibility; higher penetration of wind energy

Lead: NREL
DOE EERE

Subsurface: Carbon Capture, Fossil Fuel Extraction, Waste Disposal

Reliably guide safe long-term consequential decisions about storage, sequestration, and exploration



Lead: LBNL
DOE BES, EERE, FE, NE

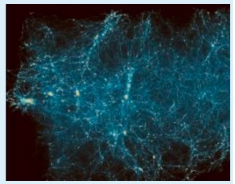
WDMApp: High-Fidelity Whole Device Modeling of Magnetically Confined Fusion Plasmas

Prepare for ITER experiments and increase ROI of validation data and understanding; prepare for beyond-ITER devices

Lead: PPPL
DOE FES

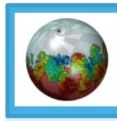
ExaSky: Cosmological Probe of the Standard Model of Particle Physics

Unravel key unknowns in the dynamics of the Universe: dark energy, dark matter, and inflation



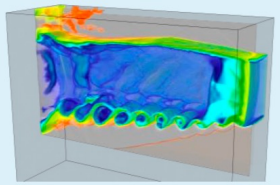
Lead: ANL
DOE HEP

The MARBL Multi-physics Code



Multi-physics simulations of high energy-density physics and focused experiments driven by high-explosive, magnetic or laser based energy sources

- Magneto-radiation-hydrodynamics at the exascale
- Next-generation pulsed power / ICF modeling
- High-order numerical methods



Lead: LLNL

ECP applications require consistency across the software stack

ECP Applications: C Cosmology (ExaSky) · F Fusion Energy (WDMApp) · N National Security (MAPP) · S Subsurface Flow · W Wind Farm (ExaWind) … and more

24 apps, 6 co-design centers

Shown are 36 ST products (used or being considered by the 5 apps above)

ST overall has 71 unique software products used by 24 apps and 6 co-design centers

Selected ECP Software Technologies

Programming Models and Runtimes
- MPI (C F N S W)
- Kokkos (F W)
- RAJA (N S)
- CHAI (N S)
- Umpire (N S)

… and more

Tools and Technology
- TAU (F W)
- HPCToolkit (C F N S)
- Flux (N)
- Caliper (N S)
- PAPI (C F S)

Compilers and Support
- LLVM (C F N S W)
- OpenMP (C F N W)

… and more

Math Libraries (xSDK)
- hypre (N S W)
- PETSc/TAO (F N S)
- STRUMPACK (F)
- SuperLU (F S W)
- Trilinos (F S W)
- SUNDIALS (N S)
- ArborX (C W)
- FFT (C W)
- BLAS, LAPACK (F W)
- MFEM (N)

… and more

Visualization Analysis and Reduction
- ALPINE (C N S W)
- Cinema (C N)
- VTK-m (C F N S W)
- SZ (C)
- zfp (F N)
- SPOT (N)

… and more

Data Mgmt, I/O, Checkpoint Restart
- SCR (N)
- MPI-IO (F)
- HDF5 (C F N S W)
- PnetCDF (W)
- ADIOS (F W)
- UnifyFS (N)
- VeloC (C)

… and more

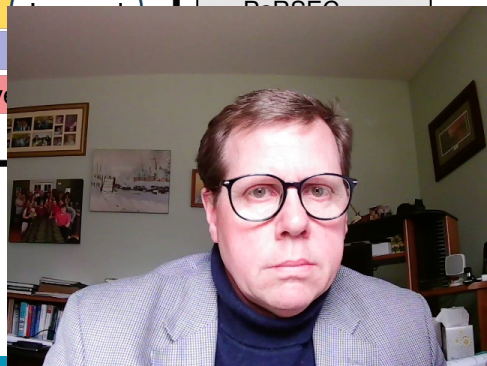
Ecosystem: E4S at large
- Spack (F N W)

… and more

Prog Models & Runtimes
Tools
Math Libraries
Data and Viz
Ecosystems and Deliv

See E4S.io for more ST products
AID
AML
BEE
Darshan
DTK
Dyninst
FleCSI
ForTriliinios
GASNet
Ginkgo
Kokkoskernels
Legion
libEnsemble
MarFS
NRM
OpenACC
Papyrus

ECP apps rely on multiple software technologies; some software products contribute to multiple distinctly developed components of a multiphysics app (such as fusion energy modeling) that must run within a single executable.



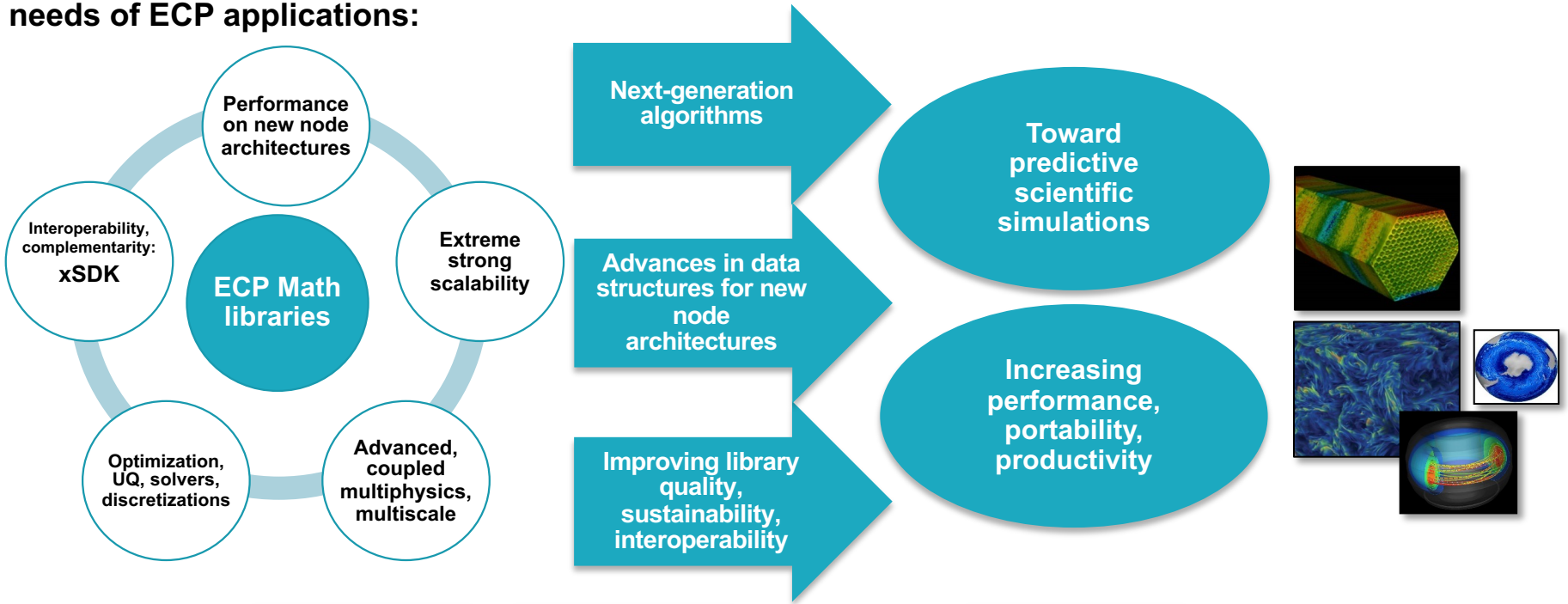
xSDK: Primary delivery mechanism for ECP math libraries' continual advancements toward predictive science

xSDK release 0.6.0 (Nov 2020)

- hypre
- PETSc/TAO
- SuperLU
- Trilinos
- AMReX
- ButterflyPACK
- DTK
- Ginkgo
- heFFTe
- libEnsemble
- MAGMA
- MFEM
- Omega_h
- PLASMA
- PUMI
- SLATE
- Tasmanian
- SUNDIALS
- Strumpack
- Alquimia
- PFLOTRAN
- deal.II
- preCICE
- PHIST
- SLEPc

} from the broader community

As motivated and validated by the needs of ECP applications:



Timeline: xSDK release 1 → xSDK release 2 → . . . . . → xSDK release n

Ref: xSDK: Building an Ecosystem of Highly Efficient Math Libraries for Exascale, **SIAM News**, Jan

Extreme-scale Scientific Software Stack (E4S)

- E4S: HPC Linux Ecosystem – a software portfolio
- A **Spack-based** distribution of software tested for interoperability and portability to multiple architectures
- Available from **source**, **containers**, **binary caches**
- Leverages and enhances SDK interoperability thrust
- Not a commercial product – an open resource for all
- 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 - 61 full release products
- Nov 2020: E4S 1.2 (aka, 20.10) - 67 full release products

E4S

e4s.io

Lead: Sameer Shende
(U Oregon)

E4S
produ
- Te
- Py
- Ho
IBM

EXASCALE COMPUTING PROJECT

Delivering an open, hierarchical software ecosystem

More than a collection of individual products

Levels of Integration	Product	Source and Delivery

ECP ST Open Product Integration Architecture

- Build all SDKs
- Build complete stack
- Assure core policies
- Build, integrate, test

E4S

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

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

SDKs

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

- Standard workflow
- Existed before ECP

ST Products

Source: ECP L4 teams; Non-ECP Developers; Sta
Delivery: Apps directly; spack; vendor stack; facilit

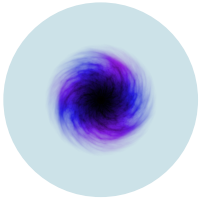
ECP ST Indi

ECP EXASCALE COMPUTING PROJECT

E4S: Better quality, documentation, testing, integration, delivery, building & use

Delivering HPC software to facilities, vendors, agencies, industry, international partners in a brand-new way

Community Policies
Commitment to software quality



DocPortal
Single portal to all E4S product info

Portfolio testing
Especially leadership platforms

Curated collection
The end of dependency hell

Quarterly releases
Release 1.2 – November

Build caches
10X build time improvement

Turnkey stack
A new user experience

https://e4s.io

E4S Str
US agencies,

E4S Community Policies

E4S Community Policies V1.0 Released

What is E4S?

The Extreme-scale Scientific Software Stack (E4S) is a community effort to provide open source software packages for developing, deploying and running scientific applications on high-performance computing (HPC) platforms. E4S provides from-source builds and containers of a broad collection of HPC software packages.

Purpose

E4S exists to accelerate the development, deployment and use of HPC software, lowering the barriers for HPC users. E4S provides containers and turn-key, from-source builds of more than 80 popular HPC products in programming models, such as MPI; development tools such as HPCToolkit, TAU and PAPI; math libraries such as PETSc and Trilinos; and Data and Viz tools such as HDF5 and Paraview.

Approach

By using Spack as the meta-build tool and providing containers of pre-built binaries for Docker, Singularity, Shifter and CharlieCloud, E4S enables the flexible use and testing of a large collection of reusable HPC software packages.

E4S Community Policies Version 1

A Commitment to Quality Improvement

- Will serve as membership criteria for E4S
  - Membership is not required for *inclusion* in E4S
  - Also includes forward-looking draft policies
- Purpose: enhance sustainability and interoperability
- Topics cover building, testing, documentation, accessibility, error handling and more
- Multi-year effort led by SDK team
  - Included representation from across ST
  - Multiple rounds of feedback incorporated from ST leadership and membership
- Modeled after xSDK Community Policies
- https://e4s-project.github.io/policies.html

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

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

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

P4 *Documentation* Each E4S member package should have sufficient documentation to support installation and use.

P5 *Product Metadata* Each E4S member package team will provide key product information via metadata that is organized in the *E4S DocPortal* format. Depending on the filenames where the metadata is located, this may require *minimal setup*.

P6 *Public Repository* Each E4S member package will have a public repository, for example at GitHub or Bitbucket, where the development version of the package is available and pull requests can be submitted.

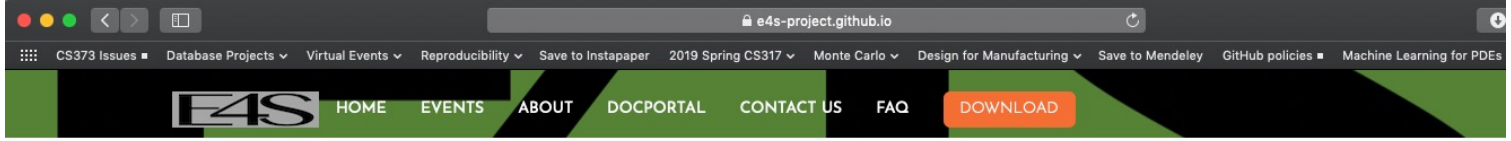
P7 *Imported Software* If an E4S member package imports software that is externally developed and maintained, then it must allow installing, building, and linking against a functionally equivalent outside copy of that software. Acceptable ways to accomplish this include (1) forsaking the internal copied version and using an externally-provided implementation or (2) changing the file names and namespaces of all global symbols to allow the internal copy and the external copy to coexist in the same downstream libraries and programs. This pertains primarily to third party support libraries and does not apply to key components of the package that may be independent packages but are also integral components to the package itself.

P8 *Error Handling* Each E4S member package will adopt and document a consistent system for signifying error conditions as appropriate for the language and application. For e.g., returning an error condition c
command line tool, it should return a sensible exit status on success/failure, so the
script.

P9 *Test Suite* Each E4S member package will provide a test suite that does not req
of commercial software. This test suite should grow in its comprehensiveness ove
should be included in the suite.

E4S DocPortal

E4S DocPortal



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

E4S Products

*: Member Product

Search:

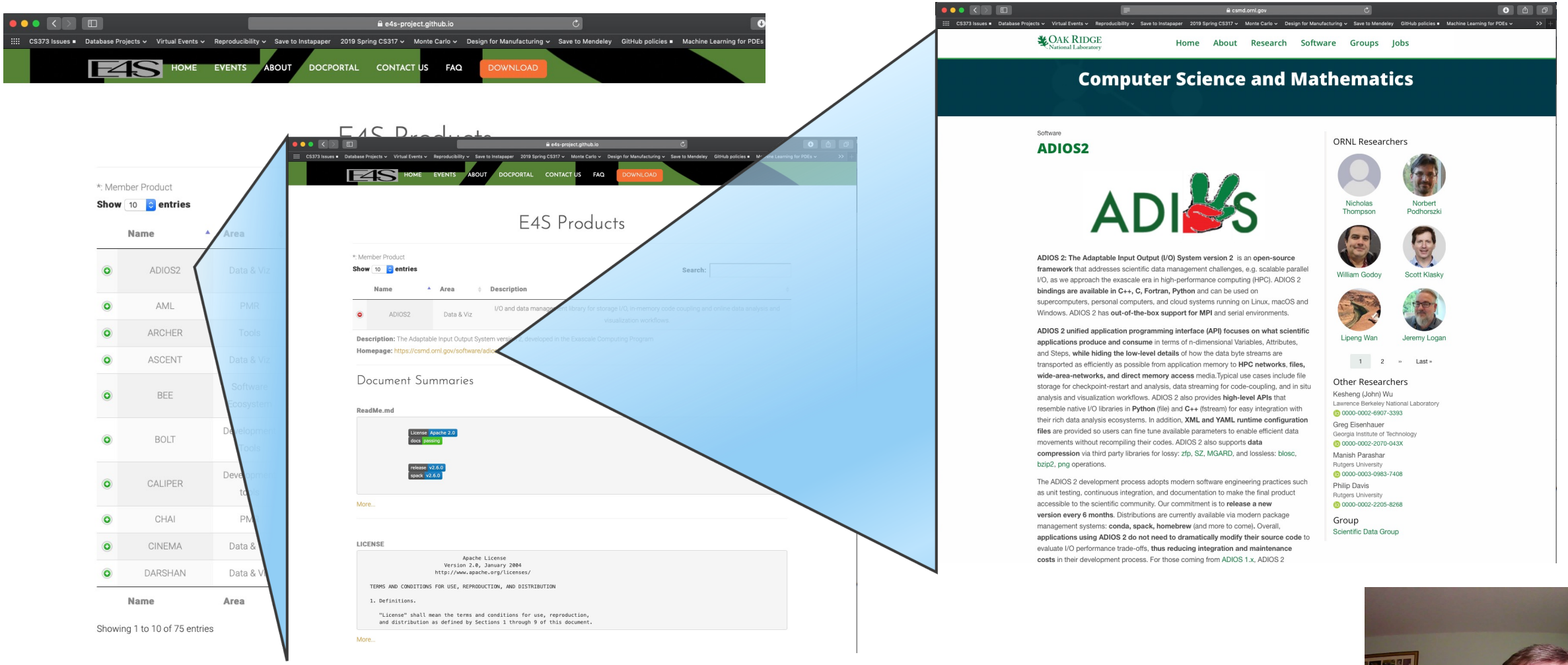
Name	Area	Description
ADIOS2	Data & Viz	I/O and data management library for storage I/O, in-memory code coupling and online data analysis and visualization workflows.
AML	PMR	Hierarchical memory management library from Argo.
ARCHER	Tools	Data race detection tool for OpenMP applications
ASCENT	Data &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

Showing 1 to 10 of 75 entries

Previous 1 2 3 4 5 … 8 Next

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

Goal: All E4S product documentation accessible from single portal on E4S.io (working mock webpage below)



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

The E4S Software Lifecycle

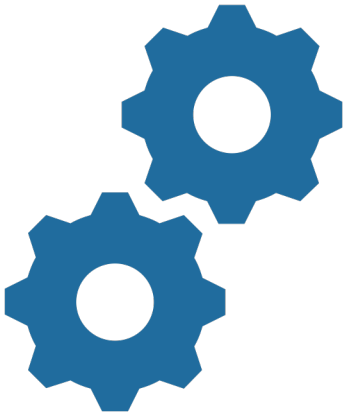
ECP ST Planning Process: Hierarchical, three-phase, cyclical

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 Review Process

Changes to Cost, Scope, and Schedule

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

Variar

Proc

KPP-3: Focus on Capability Integration

- **Capability:** Any significant product functionality, including existing features adapted to early access 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 an early access environment (tentative score) and in an exascale environment (confirmed score).
- **Brief History:**
  - Started with ”Impact Metric”, loosely defined
  - Analyzed L4 subproject team definitions to see emerging themes
  - Evolved to capability integration
  - Helped immensely by review committees, esp. Bill Carlson

ECP ST Lifecycle Summary

Managed by P6 Activity Process

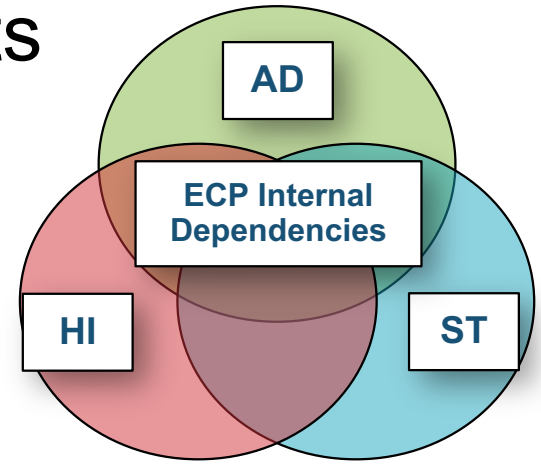
- **Create Annual Planning Package**
  - Each product has its own planning packages
  - Defined for all FYs
- **Refine upcoming FY plan**
  - Complete 6 months prior to FY
  - 4 or more P6 activities per product
- **Refine upcoming P6 activity**
  - Complete 8 weeks prior to activity start
  - Include all details
- **Develop capabilities and track progress via tailored EVM**

Measured by KPP-3 Process

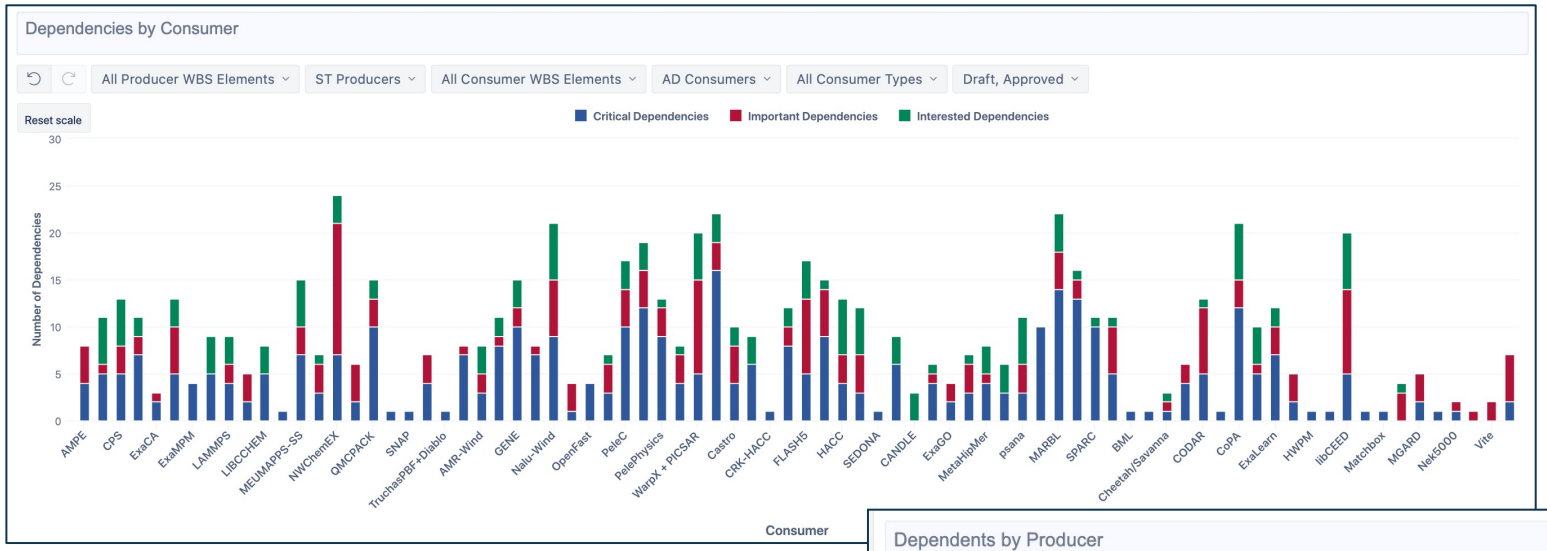
- **Integrate into product**
  - Full testing, documentation, etc.
  - Direct access for some users
- **Integrate into SDK**
  - Satisfy SDK community policies
  - Direct access for some users
- **Integrate into E4S**
  - Satisfy E4S community policies
  - Full ecosystem with high value
- **Deliver to users**
  - From source (spac
  - Containers, cloud,

Using E4S

ECP apps (AD) are primary consumers of ST products

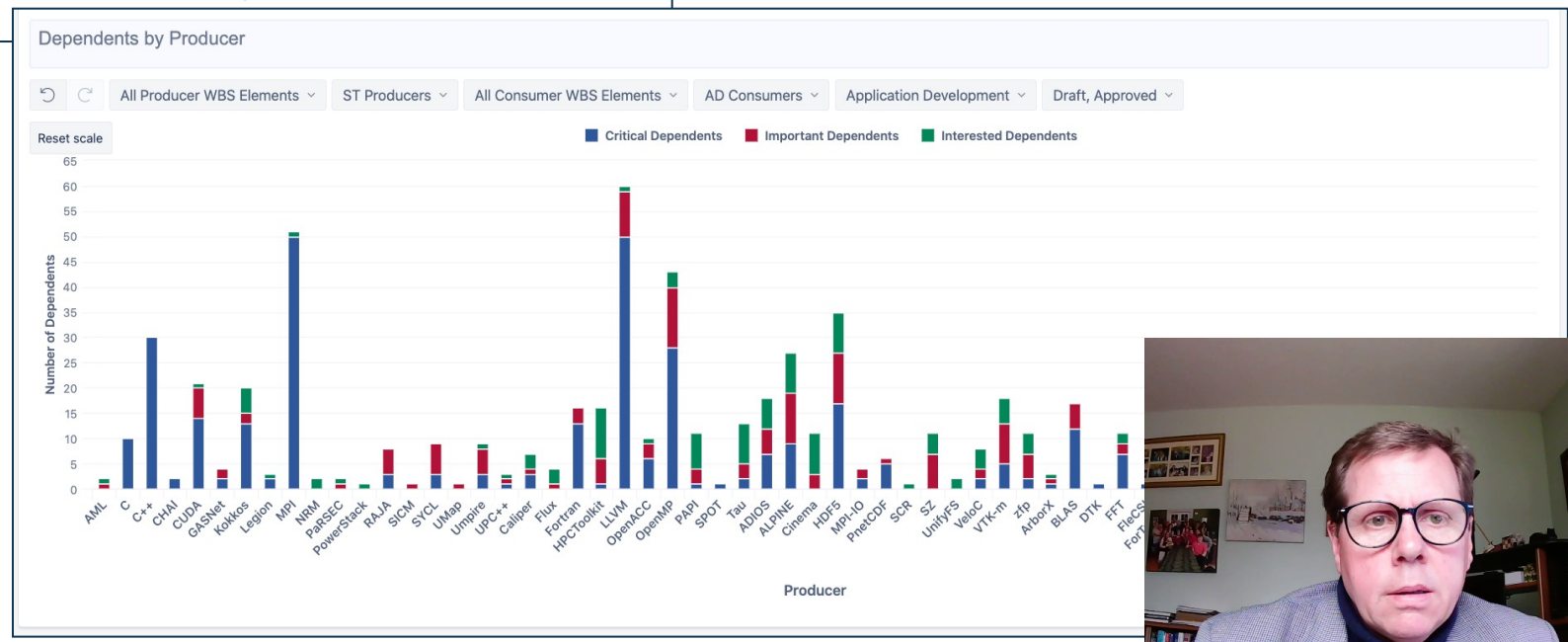


Dependency Database



View by ST producers

View by AD consumers



https://dx.doi.org/10.1038/s43588-021-00033-y

nature computational science

Comment | Published: 22 February 2021

How community software ecosystems can unlock the potential of exascale computing

Lois Curfman McInnes ✉, Michael A. Heroux, Erik W. Draeger, Andrew Siegel, Susan Coghlan & Katie Antypas

Nature Computational Science 1, 92–94(2021) | Cite this article

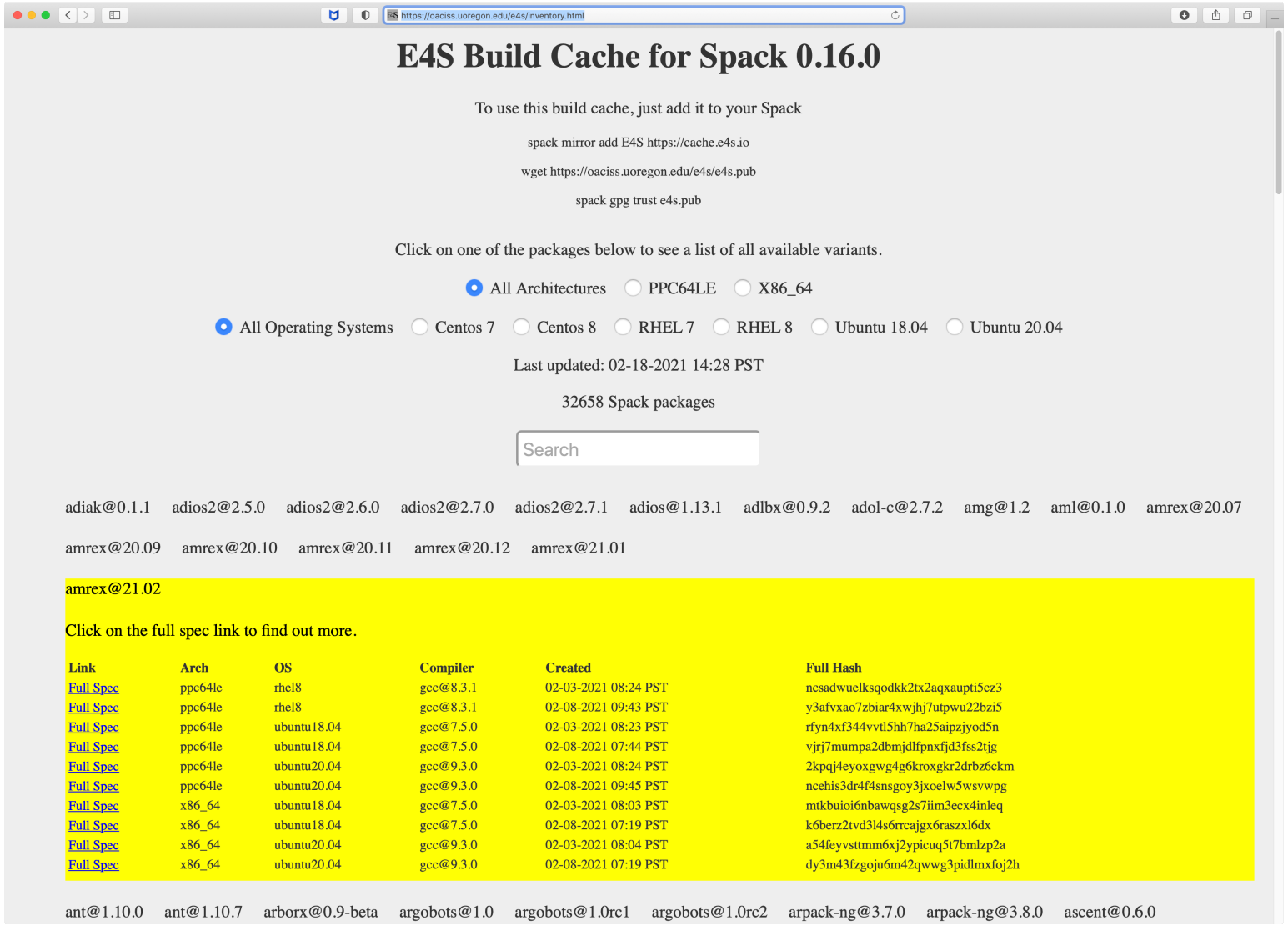
Metrics

Emerging exascale architectures and systems will provide a sizable increase in raw computing power for science. To ensure the full potential of these new and diverse architectures, as well as the longevity and sustainability of science applications, we need to embrace software ecosystems as first-class citizens.

Spack

- E4S uses the Spack package manager for software delivery
- Spack provides the ability to specify versions of software packages that are and are not interoperable.
- Spack is a build layer for not only E4S software, but also a large collection of software tools and libraries outside of ECP ST.
- Spack supports achieving and maintaining interoperability between ST software packages.

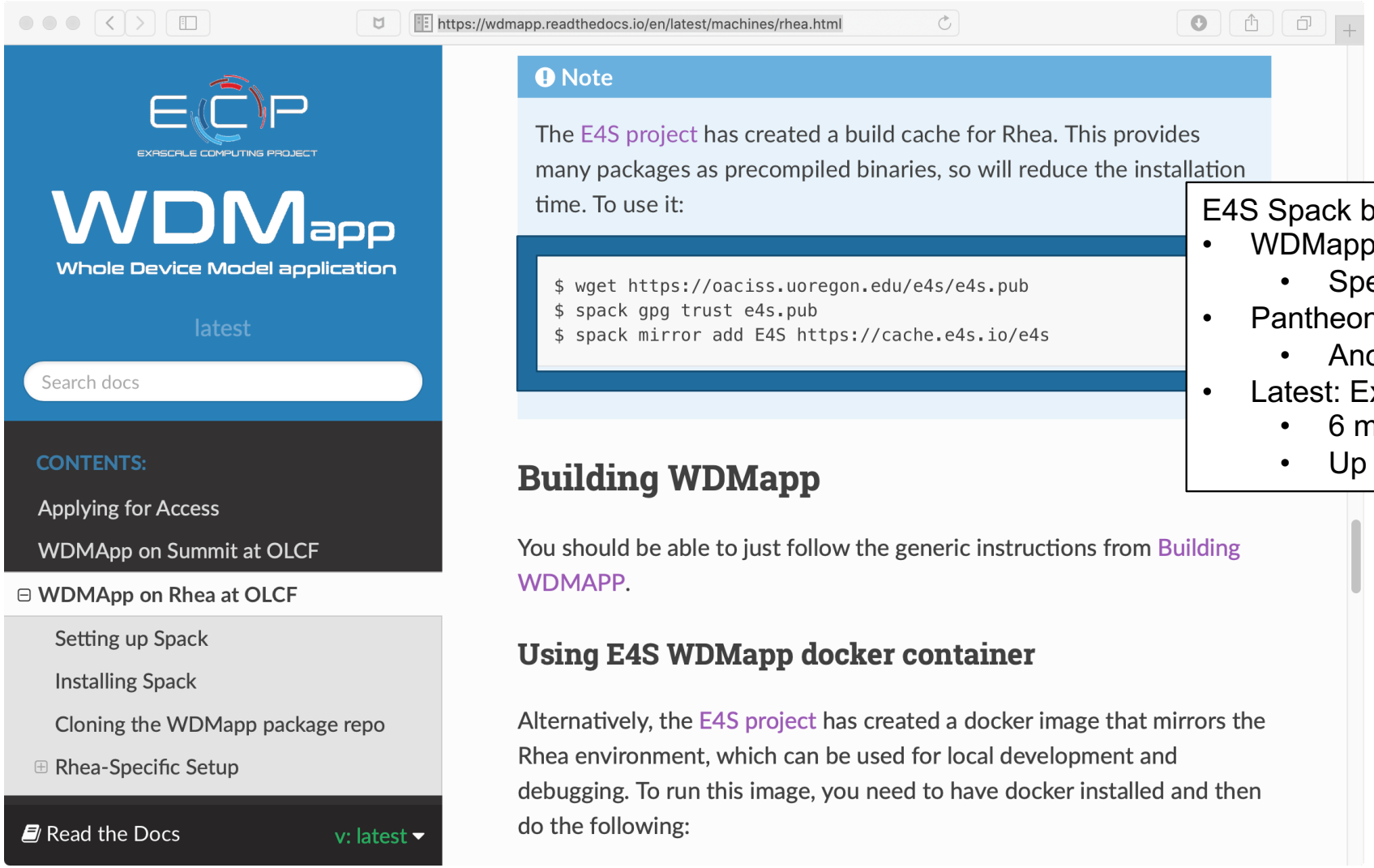
E4S: Spack Build Cache



- 32,000+ binaries
- S3 mirror
- No need to build from source code!
- Speeds up installations 10x

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

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



E4S Spack build cache:

- WDMapp added E4S mirror
  - Speedup: 10X
- Pantheon: 10X
  - Another 10X via “smoother” installs
- Latest: ExaWind (Nalu-Wind)
  - 6 minutes with build cache
  - Up to 4 hours without

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

E4S Community Engagement

Opportunities within DOE via E4S

- E4S enables portfolio strategy for ASCR R&D software delivery:
  - Facilities: Robust planning, delivery, integration and testing at Facilities
  - Community: MPI Forum, C++, OpenMP, LLVM
  - Vendor: Coordinated integration into vendor software stacks
  - Users: Turnkey delivery of capabilities to DOE program offices, US agencies, industry, international partners
- E4S provides incentives and support for high-quality research software products
  - Community policies: Drives quality by explicit expectations and clear view of gaps
  - SDKs for community interaction: Build awareness and collaboration across independent teams
  - Transparency: E4S DocPortal, build, test, integration shows quality (good or poor) of a product
- E4S provides direct path for software teams to reach users and other stakeholders
  - Example: ArborX is brand new geometric search library
    - Part of E4S, available at DocPortal, tested regularly on many platforms
    - Installed anywhere E4S is installed, users can count on it being there
    - Without E4S: ArborX would take years to become visible and available
  - Availability and adoption timeline reduced from years (or never) to months

Broader Community Engagement

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

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

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

E4S provides a natural collaboration vehicle for interacting within DOE, with other US agencies, industry and international partners

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 for ECP & beyond, including AI and Quantum.
• A monolithic, take-it-or-leave-it software behemoth.	• Full collection if compatible software capabilities **and** • Manifest of a la carte selectable software capabilities.
• A commercial product.	• Vehicle for delivering high-quality reusable software products in collaboration with others. • New entity in the HPC ecosystem enabling first-of-a-kind relationships with Facilities, vendors, other DOE program offices, other agencies, industry & international partners.
• A simple packaging of existing software.	• Hierarchical software framework to enhance (via SDKs) software interoperability and quality expectations. • Conduit for future leading edge HPC software targeting scalable computing platforms.

Looking Forward

Lessons learned from E4S/ECP ST to carry forward

- Deliver DOE reusable software as a portfolio
  - E4S value is already more than the sum of its parts
  - Community policies drive quality
  - DocPortal, testing, containerization, cloud, build caches, modules, etc., greatly improve access & usability
  - Poor performing products are ID'ed, then improved or removed
- E4S is ready to extend to next-generation software and hardware needs
  - AI/ML products already in portfolio, ready for any new products
  - Quantum, FPGA, neuromorphic devices likely to be accelerators
    - From a macro software architecture, similar to GPUs
    - Software for these devices can and should be part of the same HPC software stack for holistic environment
- DOE software as a portfolio is a first-class entity in the ecosystem
  - Planning, executing, tracking, assessing is peer collaboration with Facilities, program offic
  - E4S can become a perennial asset for DOE/ASCR as part of its mission impact within anc

Challenges for E4S sustainability

- ECP has established a robust project management infrastructure as a tailored 413.3b project
- Transitioning and adapting this infrastructure is challenging but essential for continued success
- Funding models, portfolio management, org structure are particularly critical
- Payoff if done right: better, faster and cheaper – get all three

Final Thoughts

Ecosystem: A group of independent but interrelated elements comprising a unified whole

Diversity **is essential for an ecosystem to thrive.**

- No element functions in isolation.
- Each element fulfills unique roles.
- The whole is
  than the sur
  parts.

We must explicitly consider **community software ecosystem perspectives** for next-generation computational science

Historically: Organic growth of software ecosystems around packages

What's new now? Bigger challenges, advances in technologies

Let's be intentional.

- broader perspectives
- productivity, sustainability

Better science,
Broader impact

nature computational science

Comment | Published: 22 February 2021

How community software ecosystems can unlock the potential of exascale computing

Lois Curfman McInnes ✉, Michael A. Heroux, Erik W. Draeger, Andrew Siegel, Susan Coghlan & Katie Antypas

Nature Computational Science **1**, 92–94(2021) | Cite this article

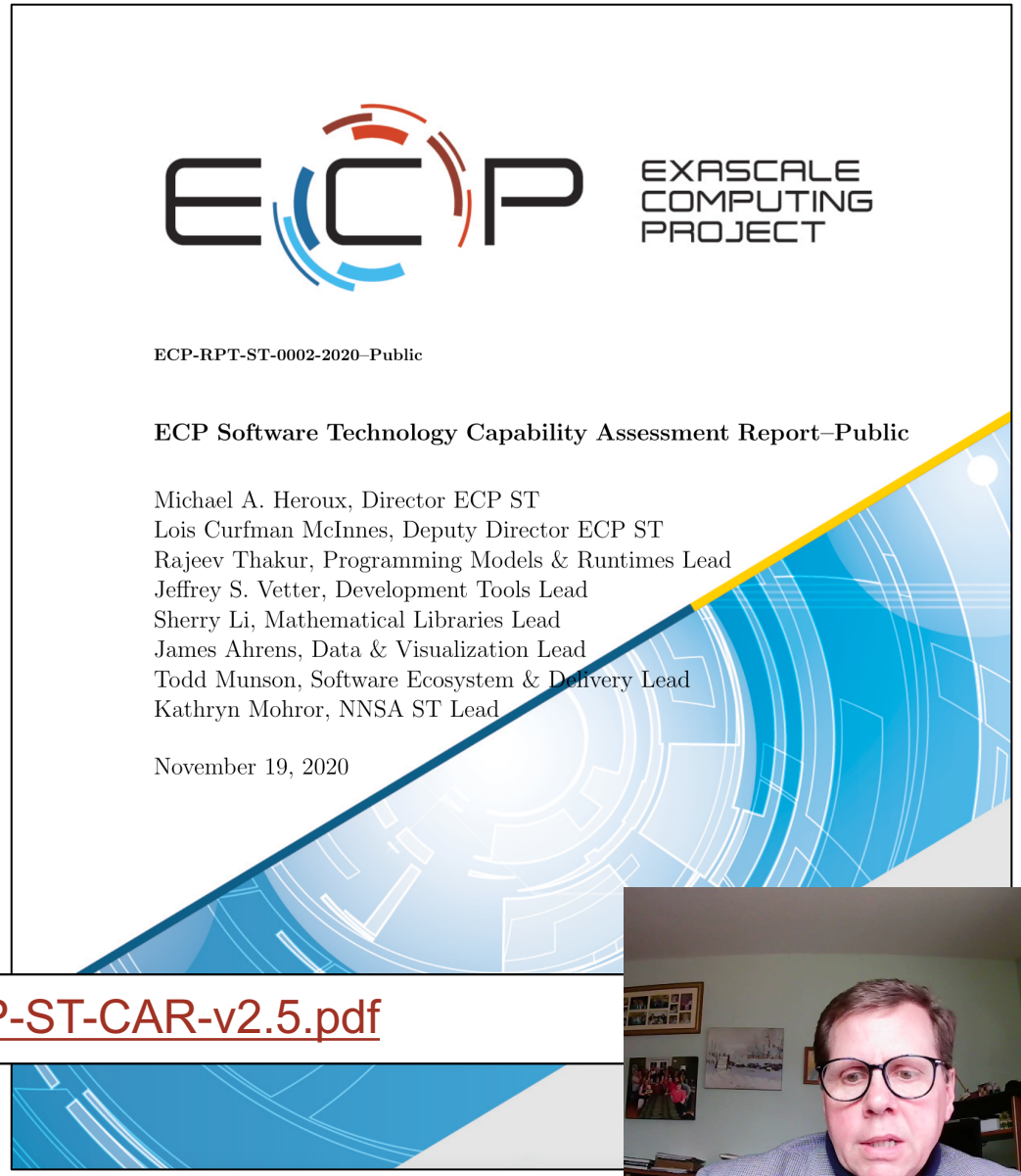
Metrics

Emerging exascale architectures and systems will provide a sizable increase in raw computing power for science. To ensure the full potential of these new and diverse architectures, longevity and sustainability of science applicati embrace software ecosystems as first-class citize

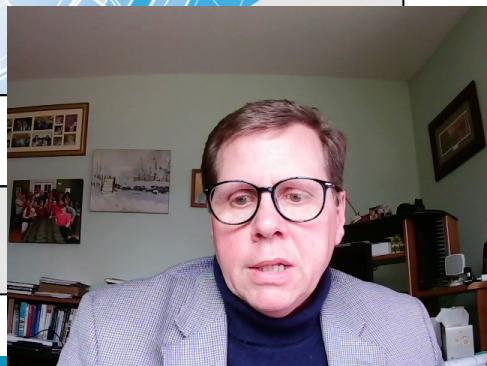
https://dx.doi.org/10.1038/s4358

ST Capability Assessment Report (CAR)

- Tiered discussion of ECP Software Technology structure, strategy, status and plans
- From high-level overview to details about each team's activities and next steps
- Produced about twice a year
- Includes gap analyses
- E4S scope updated for emerging needs



https://www.exascaleproject.org/wp-content/uploads/2021/01/ECP-ST-CAR-v2.5.pdf



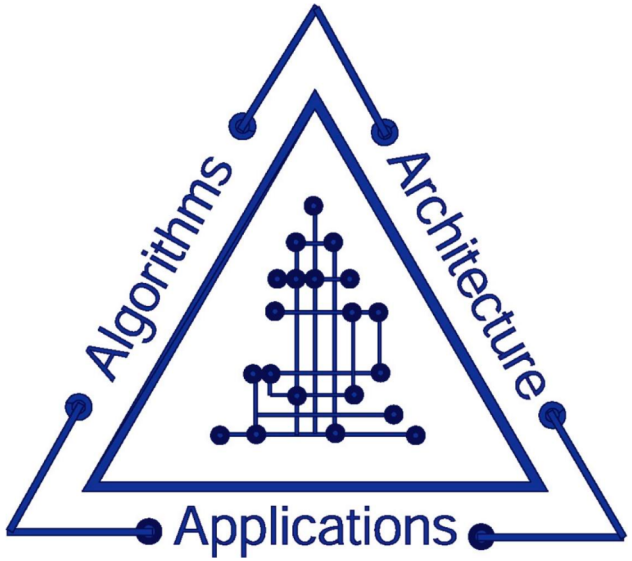
Thank you

https://www.exascaleproject.org

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.

EXASCALE COMPUTING PROJECT

Thank you to all collaborators in the ECP and broader computational science comm
work discussed in this presentation represents creative contributions of many people
passionately working toward next-generation computational science.



40TH ANNIVERSARY LECTURE SERIES