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
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 wide range of applications of importance to DOE

*Exascale Computing Project (ECP)*

System procurement projects and facilities

Selected program office application development
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's holistic approach uses co-design and integration to achieve exascale computing.
DOE HPC Roadmap to Exascale Systems

2012-2020

- **Titan**
  - ORNL
  - Cray/AMD/NVIDIA

- **Sequoia**
  - LLNL
  - IBM BG/Q

- **Cori**
  - LBNL
  - Cray/Intel Xeon/KNL

- **Theta**
  - ANL
  - Cray/Intel KNL

- **Trinity**
  - LANL/SNL
  - Cray/Intel Xeon/KNL

- **Summit**
  - ORNL
  - IBM/NVIDIA

- **Sierra**
  - LLNL
  - IBM/NVIDIA

2021-2023

- **Frontier**
  - ORNL
  - HPE/AMD

- **Aurora**
  - ANL
  - Intel/HPE

- **Perlmutter**
  - LBNL
  - HPE/AMD/NVIDIA

- **Sierra**
  - LANL/SNL
  - HPE/Intel

**Decommissioned**

- **Aurora**

**Exascale Systems**
Brief Intro to ECP
Software Technology (ST) Focus Area
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

**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
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)

<table>
<thead>
<tr>
<th>Example Products</th>
<th>Engagement</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI – Backbone of HPC apps</td>
<td>Explore/develop MPICH and OpenMPI new features &amp; standards</td>
</tr>
<tr>
<td>OpenMP/OpenACC –On-node parallelism</td>
<td>Explore/develop new features and standards</td>
</tr>
<tr>
<td>Performance Portability Libraries</td>
<td>Lightweight APIs for compile-time polymorphisms</td>
</tr>
<tr>
<td>LLVM/Vendor compilers</td>
<td>Injecting HPC features, testing/feedback to vendors</td>
</tr>
<tr>
<td>Perf Tools - PAPI, TAU, HPCToolkit</td>
<td>Explore/develop new features</td>
</tr>
<tr>
<td>Math Libraries: BLAS, sparse solvers, etc.</td>
<td>Scalable algorithms and software, critical enabling technologies</td>
</tr>
<tr>
<td>IO: HDF5, MPI-IO, ADIOS</td>
<td>Standard and next-gen IO, leveraging non-volatile storage</td>
</tr>
<tr>
<td>Viz/Data Analysis</td>
<td>ParaView-related product development, node concurrency</td>
</tr>
</tbody>
</table>
The Extreme-Scale Scientific Software Stack (E4S) and Software Development Kits (SDKs)
Core questions E4S is addressing

<table>
<thead>
<tr>
<th>How can new ECP software capabilities be effectively and efficiently integrated and sustained?</th>
</tr>
</thead>
<tbody>
<tr>
<td>• ECP success requires development, delivery and use of new GPU capabilities in 70 products</td>
</tr>
<tr>
<td>• Requires coordination of versioning, integration, testing, debugging, interaction with vendors and facilities</td>
</tr>
<tr>
<td>• Requires access to new documentation</td>
</tr>
<tr>
<td>• Requires focus on high quality</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How can E4S build upon, leverage and extend existing capabilities and activities?</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Using Spack for product installation, leveraging growing Spack capabilities</td>
</tr>
<tr>
<td>• Making E4S available via containers, cloud platforms</td>
</tr>
<tr>
<td>• Providing integration pathways to multiple destinations: from-source, LLVM, vendor stacks, facilities, etc</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How can E4s become a sustainable, open, collaborative software ecosystem for HPC?</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Hierarchical, open architecture to accept and manage community contributions</td>
</tr>
<tr>
<td>• Defined processes for community engagement within DOE, with other US agencies, industry, international partners</td>
</tr>
<tr>
<td>• Delivering the value proposition of the ecosystem vs each app managing its dependencies</td>
</tr>
</tbody>
</table>
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:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Harden wind plant design and layout against energy loss susceptibility; higher penetration of wind energy</td>
<td>Reliably guide safe long-term consequential decisions about storage, sequestration, and exploration</td>
<td>Prepare for ITER experiments and increase ROI of validation data and understanding; prepare for beyond-ITER devices</td>
</tr>
<tr>
<td>Lead: NREL DOE EERE</td>
<td>Lead: LBNL DOE BES, EERE, FE, NE</td>
<td>Lead: PPPL DOE FES</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ExaSky: Cosmological Probe of the Standard Model of Particle Physics</th>
<th>The MARBL Multi-physics Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unravel key unknowns in the dynamics of the Universe: dark energy, dark matter, and inflation</td>
<td>Multi-physics simulations of high energy-density physics and focused experiments driven by high-explosive, magnetic or laser based energy sources</td>
</tr>
<tr>
<td>Lead: ANL DOE HEP</td>
<td>Magneto-radiation-hydrodynamics at the exascale</td>
</tr>
<tr>
<td></td>
<td>Next-generation pulsed power / ICF modeling</td>
</tr>
<tr>
<td></td>
<td>High-order numerical methods</td>
</tr>
<tr>
<td></td>
<td>Lead: LLNL</td>
</tr>
</tbody>
</table>
ECP applications require consistency across the software stack

Selected ECP Software Technologies

<table>
<thead>
<tr>
<th>Programming Models and Runtimes</th>
<th>Tools and Technology</th>
<th>Math Libraries (xSDK)</th>
<th>Visualization Analysis and Reduction</th>
<th>Data Mgmt, I/O, Checkpoint Restart</th>
<th>Ecosystem: E4S at large</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI</td>
<td>TAU</td>
<td>hYPRE</td>
<td>ALPINE</td>
<td>SCR</td>
<td>Spack</td>
</tr>
<tr>
<td>Kokkos</td>
<td>HPCToolkit</td>
<td>PETSc/TAO</td>
<td>Cinema</td>
<td>MPI-IO</td>
<td></td>
</tr>
<tr>
<td>RAJA</td>
<td>Flux</td>
<td>VTK-m</td>
<td>SUNDIALS</td>
<td>HDF5</td>
<td></td>
</tr>
<tr>
<td>CHAI</td>
<td>Caliper</td>
<td>SuperLU</td>
<td>Sz</td>
<td>PnetCDF</td>
<td></td>
</tr>
<tr>
<td>Umpire</td>
<td>PAPI</td>
<td>Trilinos</td>
<td>zip</td>
<td>ADIOS</td>
<td></td>
</tr>
</tbody>
</table>

... and more

<table>
<thead>
<tr>
<th>Compilers and Support</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LLVM</td>
<td>BLAS, LAPACK</td>
<td>MFEM</td>
</tr>
<tr>
<td>OpenMP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

... and more

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.

See E4S.io for more ST products:
- AID
- AML
- BEE
- Darshan
- DTK
- Dyninst
- FleCSI
- ForTrilinios
- GASNet
- Ginkgo
- Kokkoskernels
- Legion
- libEnsemble
- MarFS
- NRM
- OpenACC
- Papyrus
- PaRSEC
- PDT
- PowerStack
- ScaLAPACK
- SCR
- SCR
- SICM
- SLATE
- SWIG
- Tasmanian
- Umap
- UPC++
As motivated and validated by the needs of ECP applications:

**ECP Math libraries**
- Performance on new node architectures
- Interoperability, complementarity: xSDK
- Extreme strong scalability
- Optimization, UQ, solvers, discretizations
- Advanced, coupled multiphysics, multiscale

**Next-generation algorithms**
- Advances in data structures for new node architectures
- Improving library quality, sustainability, interoperability

**Toward predictive scientific simulations**
- Increasing performance, portability, productivity

**Timeline:**
- xSDK release 0.6.0 (Nov 2020)
- 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 also includes non-ST products such as TensorFlow, PyTorch, Horovod

IBM uses E4S for ML/AI

**e4s.io**

Lead: Sameer Shende (U Oregon)
Delivering an open, hierarchical software ecosystem
More than a collection of individual products

Levels of Integration | Product | Source and Delivery
--- | --- | ---
• 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; Standards Groups
Delivery: Apps directly; spack; vendor stack; facility

ECP ST Open Product Integration Architecture
ECP ST Individual Products
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 Strategy Group
  US agencies, industry, international partners

- "Turnkey stack"
  A new user experience
E4S Community Policies V1.0 Released

The Extreme-scale Scientific Software Stack

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
E4S DocPortal
E4S DocPortal

• The DocPortal is live!

• Summary Info
  – Name
  – Functional Area
  – Description
  – License

• Searchable
• Sortable

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

<table>
<thead>
<tr>
<th>Minor</th>
<th>Major</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lightweight Review in Jira, L3 and L2 leads</td>
<td>Change Control Board Review, ECP leadership</td>
</tr>
</tbody>
</table>

Variance Recorded in Jira
Proceed with Execution

Plan
Assess
Execute
Track
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

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

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 (spack)
- Containers, cloud, vendor stacks

Managed by P6 Activity Process

Measured by KPP-3 Process
Using E4S
ECP apps (AD) are primary consumers of ST products

Dependency Database

View by AD consumers

View by ST producers

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

Nature Computational Science

How community software ecosystems can unlock the potential of exascale computing

Lous Gurjani, Minnovesto El, Michael A. Henne, Dik D. Diener, Andrew Sigg, Susan Gepner & Khile Anoos

Nature Computational Science 1, 84-94 (2021) | Cit this article

Emerging exascale architectures and systems will provide a sizable increase in in-computing power for sciences. 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

---

**E4S Build Cache for Spack 0.16.0**

To use this build cache, just add it to your Spack

```bash
gist add E4S https://oaciss.uoregon.edu/e4s
```

Last updated: 02-18-2021 14:28 PST

32658 Spack packages

---

<table>
<thead>
<tr>
<th>Link</th>
<th>Arch</th>
<th>OS</th>
<th>Compiler</th>
<th>Created</th>
<th>Full Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Static</td>
<td>ppc64le</td>
<td>rel=8</td>
<td>gcc=4.3.1</td>
<td>02-05-2021 08:24</td>
<td>ncmn0szdrk23t2gacsg8g8c33</td>
</tr>
<tr>
<td>Full Static</td>
<td>ppc64le</td>
<td>rel=8</td>
<td>gcc=4.3.1</td>
<td>02-05-2021 08:24</td>
<td>ncmn0szdrk23t2gacsg8g8c33</td>
</tr>
<tr>
<td>Full Static</td>
<td>ppc64le</td>
<td>ubuntu-18.04</td>
<td>gcc=4.3.1</td>
<td>02-05-2021 08:24</td>
<td>ncmn0szdrk23t2gacsg8g8c33</td>
</tr>
</tbody>
</table>

---

- https://oaciss.uoregon.edu/e4s/inventory.html
WDMApp: Speeding up bare-metal installs using E4S build cache

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:

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

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

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

- 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
- 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

- Presenters from 11 institutions, 6 non-DOE
- 70 participants
  - DOE Labs, NASA
  - AMD
  - HLRS, CSCS
## E4S summary

<table>
<thead>
<tr>
<th>What E4S is not</th>
<th>What E4S is</th>
</tr>
</thead>
<tbody>
<tr>
<td>• A closed system taking contributions only from DOE software development teams.</td>
<td>• Extensible, open architecture software ecosystem accepting contributions from US and international teams.</td>
</tr>
<tr>
<td>• A monolithic, take-it-or-leave-it software behemoth.</td>
<td>• Framework for collaborative open-source product integration for ECP &amp; beyond, including AI and Quantum.</td>
</tr>
<tr>
<td>• A commercial product.</td>
<td>• Full collection if compatible software capabilities and</td>
</tr>
<tr>
<td>• A simple packaging of existing software.</td>
<td>• Manifest of a la carte selectable software capabilities.</td>
</tr>
<tr>
<td></td>
<td>• Vehicle for delivering high-quality reusable software products in collaboration with others.</td>
</tr>
<tr>
<td></td>
<td>• New entity in the HPC ecosystem enabling first-of-a-kind relationships with Facilities, vendors, other DOE program offices, other agencies, industry &amp; international partners.</td>
</tr>
<tr>
<td></td>
<td>• Hierarchical software framework to enhance (via SDKs) software interoperability and quality expectations.</td>
</tr>
<tr>
<td></td>
<td>• Conduit for future leading edge HPC software targeting scalable computing platforms.</td>
</tr>
</tbody>
</table>
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 offices, etc.
  – E4S can become a perennial asset for DOE/ASCR as part of its mission impact within and beyond DOE
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 greater than the sum of its 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, we must consider the longevity and sustainability of software applications. To this end, we argue for software ecosystems and software management practices as first-class citizens of computational science. This broad and inclusive approach is essential to the future of computational science.

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

Thank you

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.

Thank you to all collaborators in the ECP and broader computational science community. The work discussed in this presentation represents creative contributions of many people passionately working toward next-generation computational science.
40TH ANNIVERSARY LECTURE SERIES