ECP Update: Software Technologies

Mike Heroux
Todd Munson

ASCR Advisory Committee Meeting
July 22, 2022

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

ECP Software Technology Leadership Team

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

Lois Curfman McInnes, Software Technology Deputy Director
Lois is a senior computational scientist in the Mathematics and Computer Science Division of ANL. She has over 20 years of experience in HPC 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.

Kathryn Mohror, NNSA ST (2.3.6)
Kathryn is Group Leader for the CASC Data Analysis Group at LLNL. Her work focuses on I/O for extreme scale systems, scalable performance analysis and tuning, fault tolerance, and parallel programming paradigms. She is a 2019 recipient of the DOE Early Career Award.

ST L4 Leads

- WBS
- Name
- PIs
- PCs - Project Coordinators

ECP ST Stats

- 250 staff
- 70 products
- 35 L4 subprojects
- 30 universities
- 9 DOE labs
- 6 technical areas
- 1 of 3 ECP focus areas

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)	Guo, Yanfei	Guo, Yanfei
2.3.1.08	Legion	McCormick, Pat	McCormick, Pat
2.3.1.09	PaRSEC	Bosilca, 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	Graham, Jonathan	Turton, Terry
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++	Jagode, Heike	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	Hornick, Mike
2.3.2.11	SOLLVE: Scaling OpenMP with LLVm for Exascale	Chandrasekaran, Sunita	Chandrasekaran, Sunita
2.3.2.12	FLANG	McCormick, Pat	Perry-Holby, Alexis
2.3.3	**Mathematical Libraries**	**Li, Sherry**	
2.3.3.01	Extreme-scale Scientific xSDK for ECP	Yang, Ulrike	Yang, Ulrike
2.3.3.06	Preparing PETSc/TAO for Exascale	Munson, Todd	Munson, Todd
2.3.3.07	STRUMPACK/SuperLU/FFTX: sparse direct solvers, preconditioners, and FFT libraries	Li, Sherry	Li, Sherry
2.3.3.12	Enabling Time Integrators for Exascale Through SUNDIALS/ Hypre	Woodward, Carol	Woodward, Carol
2.3.3.13	CLOVER: Computational Libraries Optimized Via Exascale Research	Gates, Mark	Carr, Earl
2.3.3.14	ALExa: Accelerated Libraries for Exascale/ForTrilinos	Prokopenko, Andrey	Grundhoffer, Alicia
2.3.3.15	Sake: Solvers and Kernels for Exascale	Rajamanickam, Siva	Trujillo, Gabrielle
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	Hornick, Mike
2.3.4.10	DataLib: Data Libraries and Services Enabling Exascale Science	Ross, Rob	Ross, Rob
2.3.4.13	ECP/VTK-m	Moreland, Kenneth	Moreland, Kenneth
2.3.4.14	VeloC: Very Low Overhead Transparent Multilevel Checkpoint/Restart/Sz	Cappello, Franck	Ehling, Scott
2.3.4.15	ExaIO - Delivering Efficient Parallel I/O on Exascale Computing Systems with HDF5 and Unify	Byna, Suren	Bagha, Neelam
2.3.4.16	ALPINE: Algorithms and Infrastructure for In Situ Visualization and Analysis/ZFP	Ahrens, James	Turton, Terry
2.3.5	**Software Ecosystem and Delivery**	**Munson, Todd**	
2.3.5.01	Software Ecosystem and Delivery Software Development Kit	Willenbring, James M	Willenbring, James M
2.3.5.09	SW Packaging Technologies	Gamblin, Todd	Gamblin, Todd
2.3.5.10	ExaWorks	Laney, Dan	Laney, Dan
2.3.6	**NNSA ST**	**Mohror, Kathryn**	
2.3.6.01	LANL ATDM	Mike Lang	Vandenbusch, Tanya Marie
2.3.6.02	LLNL ATDM	Becky Springmeyer	Gamblin, Todd
2.3.6.03	SNL ATDM	Jim Stewart	Trujillo, Gabrielle

ECP Software Technology works on products that apps need now and in the future

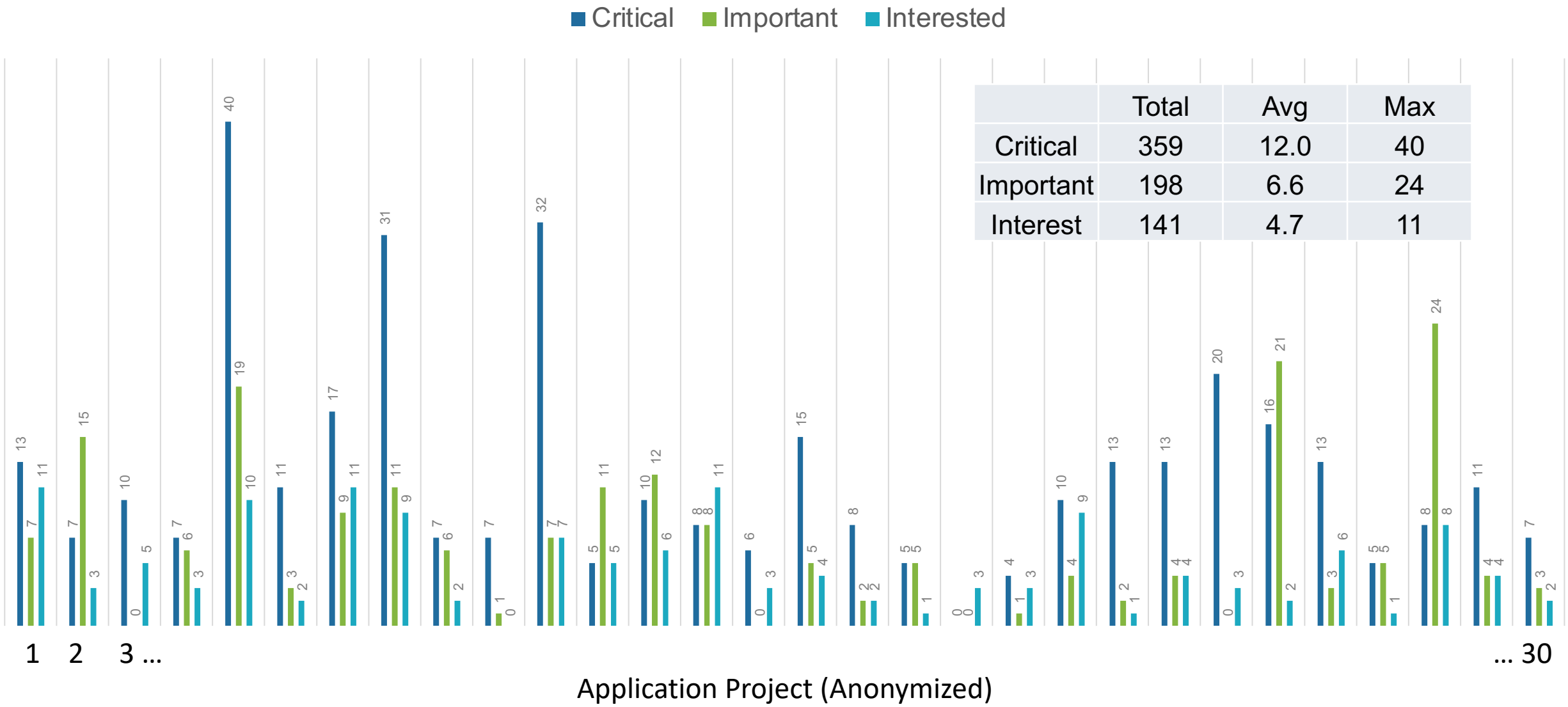
Key themes:
- Focus: GPU node architectures and advanced memory & storage technologies
- Create: New high-concurrency, latency tolerant algorithms
- Develop: New portable (Nvidia, Intel, AMD GPUs) software product
- Enable: Access and use via standard APIs

Legacy: A stack that enables performance portable application development on leadership platforms

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 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 NUMBER OF ECP SOFTWARE TECHNOLOGY PROJECT DEPENDENCIES FOR EACH ECP APPLICATION PROJECT (ANONYMIZED)



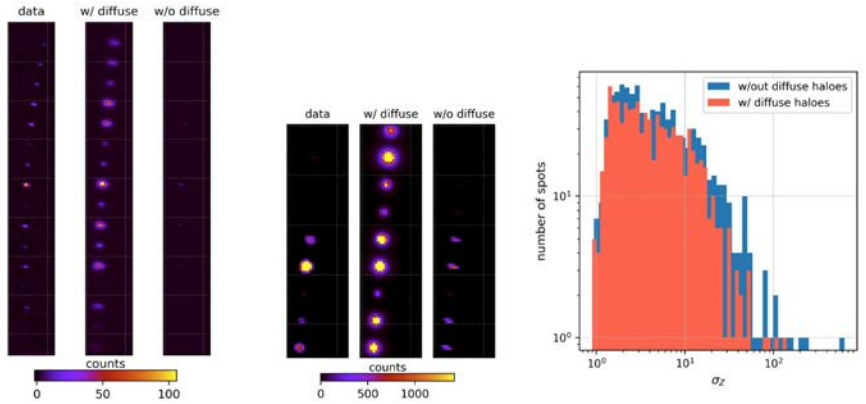
	Total	Avg	Max
Critical	359	12.0	40
Important	198	6.6	24
Interest	141	4.7	11

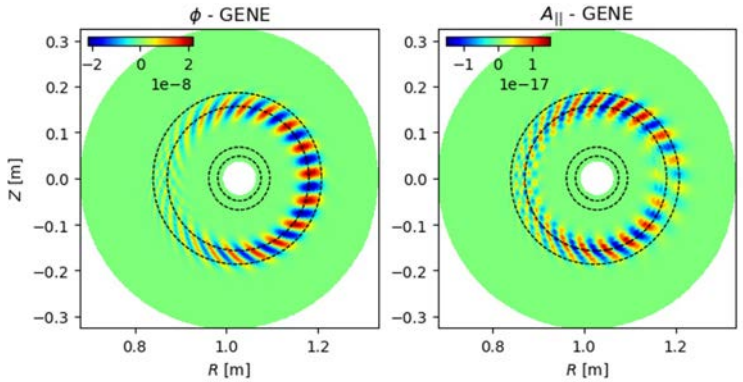
Integration: AD Teams Depend Heavily on ST Software to Meet KPPs

Kokkos

ExaFEL

nanoBragg code ported from Nvidia to AMD GPUs with minimal effort





ADIOS

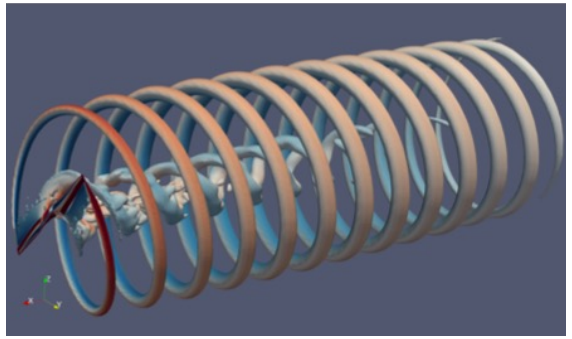
WDMApp

ADIOS enables in-memory coupling between GENE and XGC

hypre

ExaWind

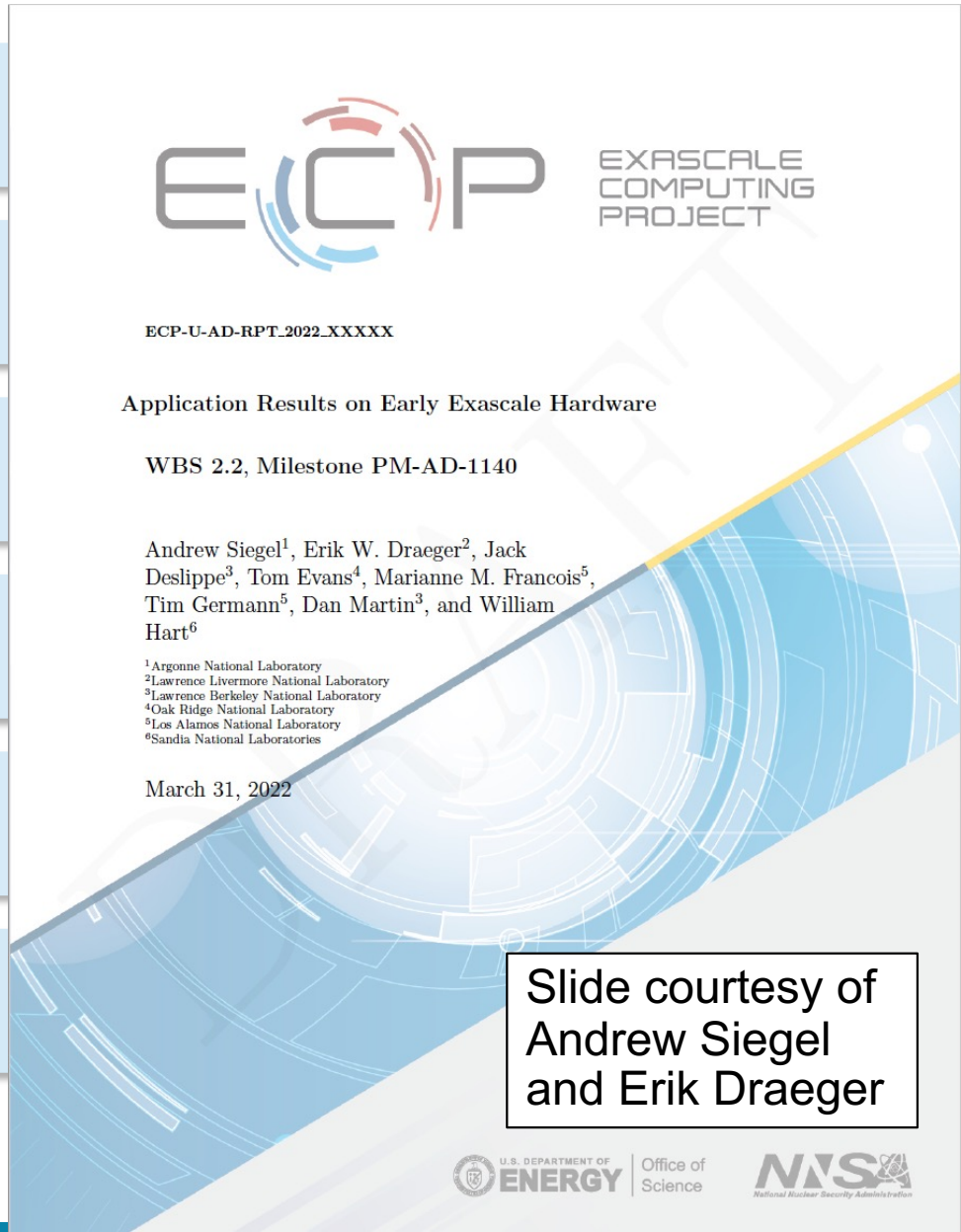
hypre solve performance on AMD GPUs 30-40% faster than Summit



Slide courtesy of Andrew Siegel and Erik Draeger

Interesting Themes Arising from 2021 AD Assessment

- **Sparse solver progress/research challenges**
- **Evolving OpenMP offload performance**
- **Co-maturation of vendor compilers, software stack**
- ✓ **ST and CD integration success stories**
- **Maturity of performance analysis tools**
- **Network performance**



Slide courtesy of Andrew Siegel and Erik Draeger

ST and Co-Design projects use KPP-3 to measure integration and drive creation of a productive and sustainable ecosystem

KPP-3 Basics

- **Integration Goal:**
  A statement of impact on the ECP ecosystem, consequential and sustainable use by client.
- **Metric: Capability integration**
  - **ST:** Use of the product for the first time or a significant feature set recently developed representing an FTE or more worth of effort.
  - **CD**: Number of applications using the co-design center's technologies in a sustained way.
- **Threshold/Objective:**
  50%/100% of the weighted (stretch) impact goals are met.

KPP-3 Details

- Weights correlate with scope of impact. Examples:
  - OpenMP, MPICH, AMReX – Weight of 2.
  - Most – Weight of 1.
  - Legion, ParSEC, ExaGraph – Weight of 0.5.
- Integration must represent sustainable progress, not just "tried it" or "considering it".
- Not looking for hero-level integration score counts. Integration is hard work.
  - Typical threshold goals: 4 integrations. A few are higher.

KPP-3 Integration Clients & Artifacts Overview

AD or ST Client

- ST product in use by an AD or ST client, demonstrated on exascale platform
  - May include multiple linked products
- Example:
  - MFIX-Exa + AMReX + ALPINE Catalyst + ALPINE statistical feature detection algorithm + VTK-m + Cinema
- Artifacts:
  - Merge requests/Change logs
  - Run and output logs
  - Journal papers, technical report, milestone report
  - Client Letter
  - Demos or visualizations

Tool Usage

- Utility/Library used in client workflow; pre-exascale or exascale
- Examples:
  - HPCToolKit
  - Darshan
- Artifacts:
  - Merge requests/Change logs
  - Client Letter
  - Performance studies (plots) demonstrating impact on client
  - Technical report, journal paper, milestone report

Facilities Deployment

- Utility/library deployed on exascale machine for general use
- Examples:
  - Performance toolkits
  - ParaView & VisIt visualization applications
- Artifacts:
  - Merge requests/Change logs
  - Module load screenshots
  - Log files from unit tests
  - Tutorial slides, documentation or other user-support activities
  - Milestone report

Community Ecosystem & Vendor Deployment

- Integration into sustainable community software environment or adopted by vendor
- Examples:
  - LLVM
  - OpenMP, OpenACC
- Artifacts:
  - Merge requests/Change logs
  - Meeting notes
  - Proposal to standards or vendor
  - Code review summary
  - Documentation
  - Milestone report

ECP ST Advisory & Review Team (START) for KPP-3 & Project Reviews

2.3.1 Programing Models & Runtimes L3: Rajeev Thakur	Role
Sanjay Kale (UIUC)	SME
Aparna Chandramowlishwaran (UC-Irvine)	SME
Bill Carlson (IDA)	SME
Heidi Poxon (Amazon)	SME
Hartmut Kaiser (LSU)	SME
Simon McIntosh-Smith (Bristol)	SME
Nicholas Wright (NERSC)	Fac & App
Kalyan (Kumar) Kumaran (ALCF)	Fac & App

2.3.2 Development Tools L3: Jeff Vetter	Role
Bernd Mohr (Jülich) - Facility (Jülich)	SME/Fac
Rudolf Eigenmann (Purdue)	SME
Jesus Labarta (Barcelona) - Facility (Barcelona)	SME/Fac
Dorian Arnold (Emory)	SME
Judy Hill (LLNL)	Fac & App
Reuben Budiardja (OLCF)	Fac & App

2.3.3. Math Library L3: Sherry Li	Role
Zhaojun Bai (UC-Davis)	SME
Wolfgang Bangerth (Colorado St)	SME
Edmond Chow (GA-Tech)	SME
Patrick Amestoy (Toulouse)	SME
Olga Pearce (LLNL)	Fac & App
Tom Beck (ORNL)	Fac & App

2.3.4 Data & Visualization L3: James Ahrens	Role
Paul Navratil (TACC)	SME
Jian Huang (Tennessee)	SME
Kate Isaacs (Arizona)	SME
Jay Lofstead (Sandia)	SME
Arjun Shankar (OLCF)	Fac & App
Ray Loy (ALCF)	Fac & App

2.3.6 NNSA ST L3: Kathryn Mohror	Role
John Levesque (HPE)	SME
Bob Lucas (ANSYS)	SME
Nick Malaya (AMD)	SME

- Independent domain experts to assess KPP-3
- Double duty as annual project review SMEs
- Annual (December) reviews: All L4 subprojects
- Ongoing (quarterly) review of KPP-3 capability integration evidence

2.3.5 Software Ecosystem L3: Todd Munson	Role
Karl Schulz (Texas)	SME
Joost VandeVondele (ETH-Zurich)	SME
Katherine Riley (ALCF)	Fac & App

E4S/SDK Review	Role
Martin Schulz (Munich)	SME
Sadaf Alam (CSCS)	Fac
Richard Gerber (NERSC)	Fac

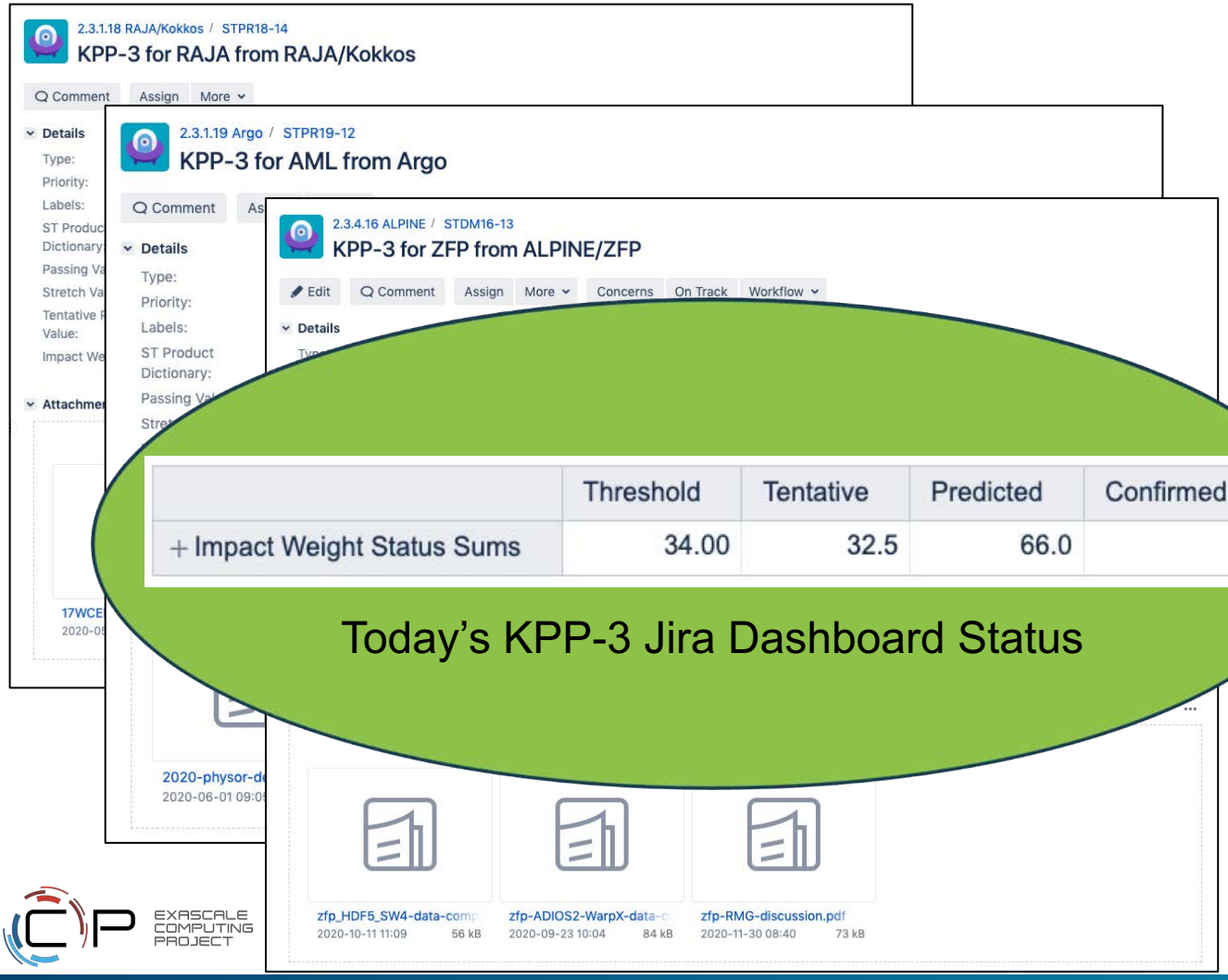
ECP ST Reviews Dec 6 – 10, 2021

- Three topics:
  - Status of integration (KPP-3)
  - Status of Early-Access system efforts
  - Gap analysis against E4S policies
- Refinement on several subprojects
- Emerging themes
  - Staffing pressure – industry recruitment
  - Concerns about post-ECP transition
  - Product usability for broad user base

ST Dashboard (JIRA)

JIRA dashboard for KPP-3: https://jira.exascaleproject.org/secure/Dashboard.jspa?selectPageId=12200

- **Current JIRA KPP-3 Issue tracks KPP-3 (tentative) points**
- **Rolls up to ST JIRA Dashboard: *KPP-3 Goal Status***



KPP-3 Goal	Status (Tentative)	Status (Confirmed)	KPP-3 Success Risk
KPP-3 for CODAR			
KPP-3 for CoPA suite			
KPP-3 for AMReX suite			
KPP-3 for CEED suite			
KPP-3 for ExaGraph suite			
KPP-3 for ExaLearn suite			
KPP-3 for VTK-m from ECP-VTK-m			
KPP3 for ADIOS from ADIOS			
KPP-3 for Darshan from DataLib			
KPP-3 for PnetCDF from DataLib			
KPP-3 for HDF5 from DataLib			
KPP-3 for VeloC from VeloC-SZ			
KPP-3 for SZ from VeloC-SZ			
KPP-3 for HDF5 from ExaIO			
UnifyFS from			
LPINE from			
FP from			
or E4S from Tools			
KPP-3 for LLVM from PROTEAS-TUNE			

KPP-3 Goal	Status (Tentative)	Status (Confirmed)	KPP-3 Success Risk
KPP-3 for TAU from PROTEAS-TUNE	Passing	Below Passing	Normal
KPP-3 for PDT from PROTEAS-TUNE			
KPP-3 for E4S from Data & Viz SDK			
KPP-3 for E4S from SW Ecosystem SDK			
KPP-3 for Container Images from Packaging Technologies			
KPP-3 for Spack from Packaging Technologies			
KPP-3 For ExaWorks SDK from ExaWorks			
KPP-3 for SUNDIALS from SUNDIALS-hypre			
KPP-3 for hypre from SUNDIALS-hypre			
KPP-3 for SLATE from CLOVER			
KPP-3 for FFT-ECP from CLOVER			
KPP-3 for PEEKS/Ginkgo from CLOVER			
KPP-3 for Tasmanian from ALExa			
KPP-3 for ForTrilinos from ALExa			
KPP-3 for ArborX/DTK from ALExa			
KPP-3 for Trilinos from Sake			
KPP-3 for KokkosKernels from Sake			
KPP-3 for E4S from xSDK			
KPP-3 for PETSc/TAO from PETSc/TAO			
KPP-3 for libEnsemble from PETSc/TAO			

KPP-3 Goal	Status (Tentative)	Status (Confirmed)	KPP-3 Success Risk
KPP-3 for STRUMPACK from STRUMPACK/SuperLU/FFTX	Below Passing	Below Passing	Normal
KPP-3 for SuperLU from STRUMPACK/SuperLU/FFTX	Passing	Below Passing	Normal
KPP-3 for FFTX from STRUMPACK/SuperLU			
KPP-3 for Kitsune from ATDM ST			
KPP-3 for Legion/Flec LANL ATDM ST			
KPP-3 for BEE Workfl from LANL ATDM ST			
KPP-3 for Cinema fro ATDM ST			
KPP-3 for Flux from LL ATDM ST			
KPP-3 for AID from LL ATDM ST			
KPP-3 for Spack from ATDM ST			
KPP-3 for MFEM from ATDM ST			
KPP-3 for RAJA from L ATDM ST			
KPP-3 for Umpire from ATDM ST			
KPP-3 for CHAI from L ATDM ST			
KPP-3 for Kokkos fro ATDM ST			
KPP-3 for Kokkos Ker from SNL ATDM ST			
KPP-3 for VTK-m from ATDM ST			
KPP-3 for MPICH from Exascale MPI			
KPP-3 for Legion from			
KPP-3 for PaRSEC fro			

KPP-3 Goal	Status (Tentative)	Status (Confirmed)	KPP-3 Success Risk
KPP-3 for LLVM from SOLLVE	Below Passing	Below Passing	Normal
KPP-3 for OpenMP from SOLLVE	Passing	Below Passing	Normal
KPP-3 for UPC++ from Pagoda	Below Passing	Below Passing	Normal
KPP-3 for GASNet from Pagoda	Below Passing	Below Passing	Normal
KPP-3 for E4S from PMR SDK	Passing	Below Passing	Normal
KPP-3 for MPI from OMPI-X	Below Passing	Below Passing	Normal
KPP-3 for Kokkos from RAJA/Kokkos	Passing	Below Passing	Normal
KPP-3 for RAJA from RAJA/Kokkos	Met Stretch	Below Passing	Normal
KPP-3 for NRM from Argo	Below Passing	Below Passing	Normal
KPP-3 for AML from Argo	Below Passing	Below Passing	Normal
KPP-3 for PowerStack from Argo	Passing	Below Passing	Normal
KPP-3 for UMap from Argo	Below Passing	Below Passing	Normal
KPP-3 for SICM from SICM	Below Passing	Below Passing	Normal
KPP-3 for PAPI from Exa-PAPI++	Below Passing	Below Passing	Normal
KPP-3 for HPCToolkit from HPCToolkit	Below Passing	Below Passing	Normal
KPP-3 for Dyninst from HPCToolkit	Below Passing	Below Passing	Normal
KPP-3 for Fortran from FLANG	Below Passing	Below Passing	Normal

Software Platforms: “Working in Public” Nadia Eghbal

- Platforms in the software world are digital environments that intend to improve the value, reduce the cost, and accelerate the progress of the people and teams who use them
- Platforms can provide tools, workflows, frameworks, and cultures that provide a (net) gain for those who engage
- Eghbal Platforms:

	HIGH USER GROWTH	LOW USER GROWTH
HIGH CONTRIBUTOR GROWTH	**Federations** (e.g., Rust)	**Clubs** (e.g., Astropy)
LOW CONTRIBUTOR GROWTH	**Stadiums** (e.g., Babel)	**Toys** (e.g., ssh-chat)

Eghbal, Nadia. Working in Public: The Making and Maintenance of Open Source Software (p. 60). Stripe Press. Kindle Edition.

About Platforms and ECP

- ECP is commissioned to
  - Provide new scientific software capabilities
  - On the frontiers of apps, algorithms, software and hardware
- ECP provides two new platforms to foster collaboration and cooperation as we head into the frontier:
  - **E4S:** a comprehensive portfolio of HPC products and dependencies
  - **SDKs:** Domain-specific collaborative and aggregate product suites

Software Ecosystem and Delivery

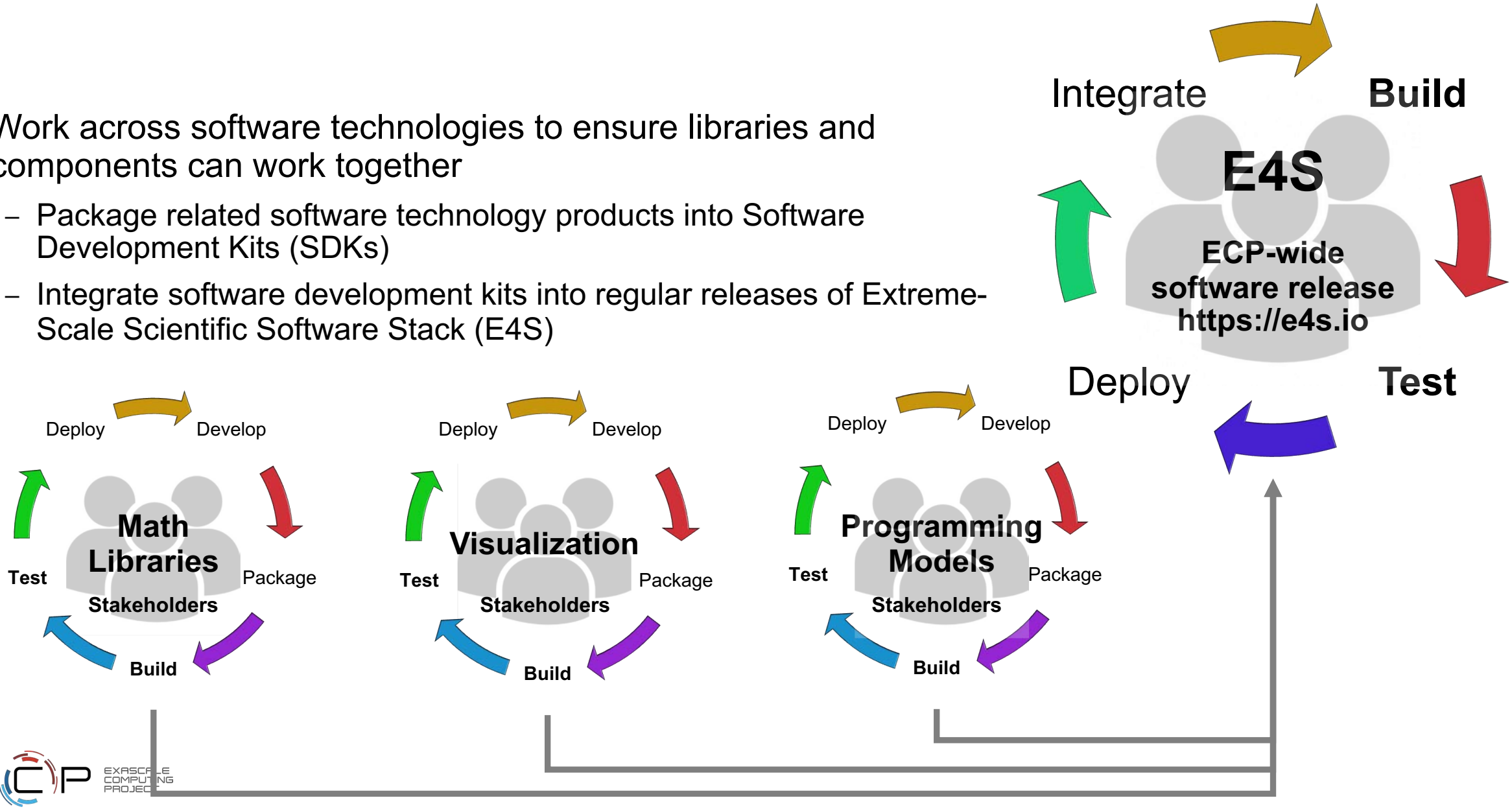
Todd Munson

Software Ecosystem and Delivery Portfolio

Project Short Name	PI Name, Inst	Short Description/Objective
Software Ecosystem SDK	Willenbring, SNL	Coordinate the software development kit activities across the software technologies and regularly release the Extreme-Scale Scientific Software Stack
Packaging Technologies	Gamblin, LLNL	Develop the Spack technologies needed to deploy large software collections at the facilities and optimize and improve interoperability of container technologies for exascale computing environments
ExaWorks	Laney, LLNL	Produce a community curated, portable, scalable, interoperable, sustainable, and trusted workflows software development kit that addresses critical application needs and is deployed on advanced computing platforms

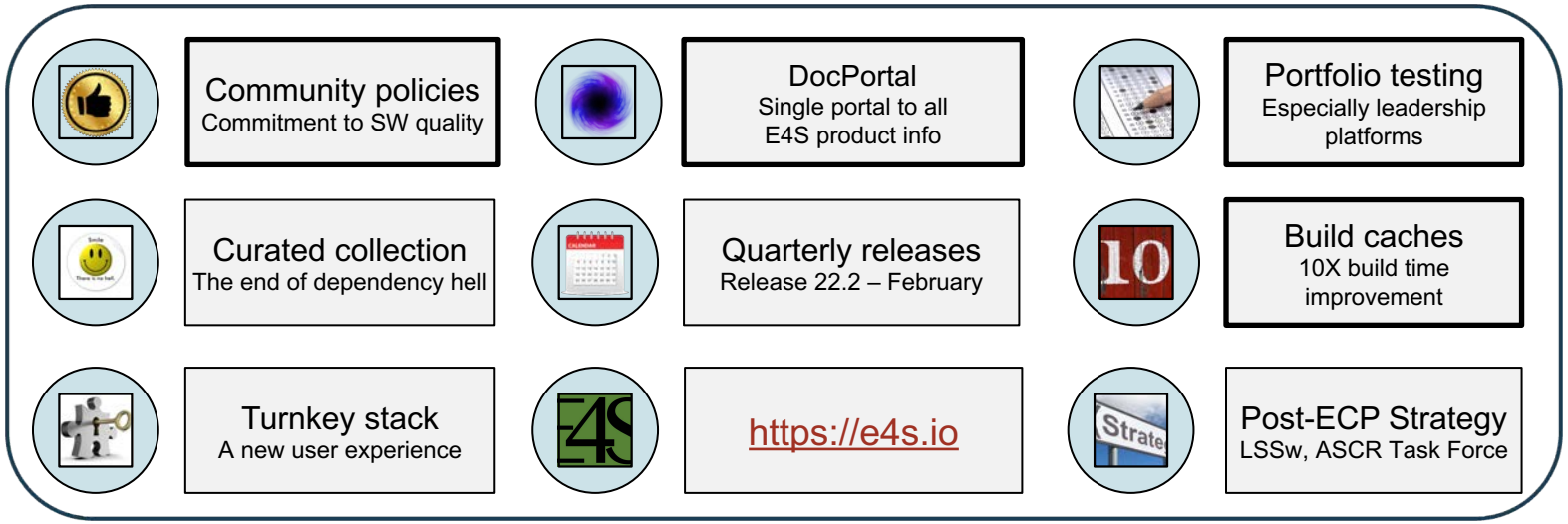
Software Development Kits: Philosophy

- Work across software technologies to ensure libraries and components can work together
  - Package related software technology products into Software Development Kits (SDKs)
  - Integrate software development kits into regular releases of Extreme-Scale Scientific Software Stack (E4S)



Extreme-scale Scientific Software Stack (E4S)

- E4S: HPC software ecosystem – a curated software portfolio
- A **Spack-based** distribution of software tested for interoperability and portability to multiple architectures
- Available from **source**, **containers**, **cloud, binary caches**
- Leverages and enhances SDK interoperability thrust
- Not a commercial product – an open resource for all
- Growing functionality: May 2022: E4S 22.05 – 100+ full release products



Spack

https://spack.io

Spack lead: Todd Gamblin (LLNL)

E4S

https://e4s.io

E4S lead: Sameer Shende (U Oregon)

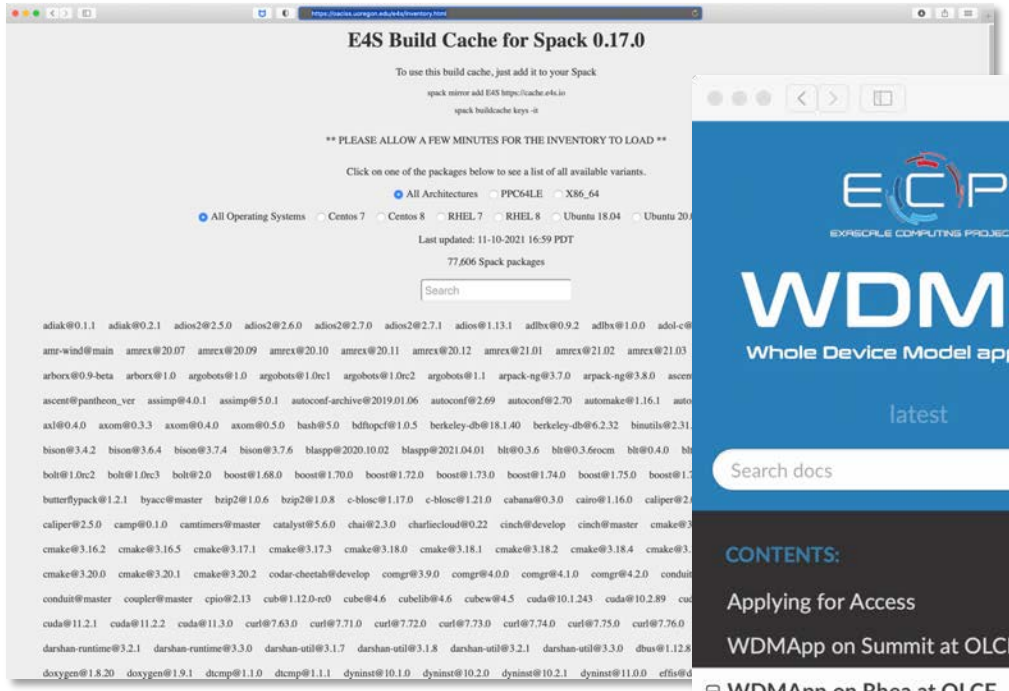
Also includes other products, e.g.,
AI: PyTorch, TensorFlow, Horovod
Co-Design: AMReX, Cabana, MFEM

Reproducible, Customizable Container Builds, and Spack Mirrors

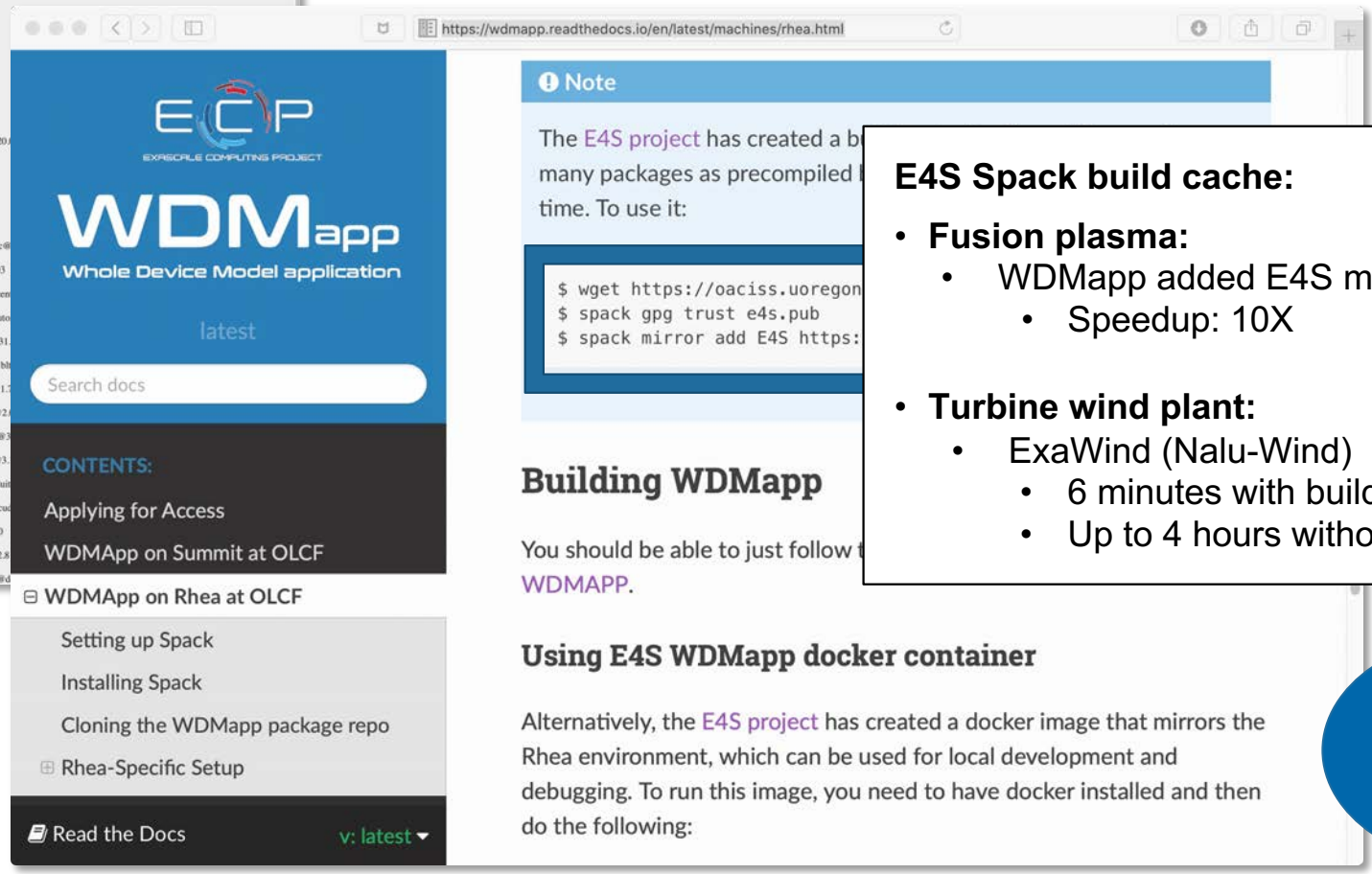
- E4S provides base images and recipes for building Docker containers based on SDKs
  - Git: https://github.com/UO-OACISS/e4s
  - Base images released (February 2022):
    - UBI 7.6, 8.0, 8.2 (RHEL Universal Binary Image for container builds) for x86_64
    - Centos 7.6 for x86_64
    - Ubuntu 18.04, 20.4 for x86_64, ppc64le
    - UBI 7.6, 8.0, 8.2 (RHEL) for ppc64le
- E4S provides build caches for Spack for bare-metal and container-based installation of ST products
  - Build caches: https://oaciss.uoregon.edu/e4s/inventory.html
    - The build cache model can be extended to target platforms, and can be managed by facilities staff when appropriate

Speeding up bare-metal installs using the E4S build cache

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

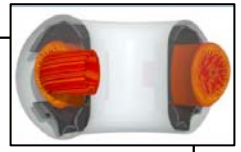


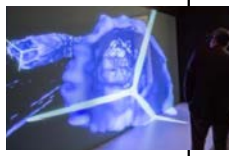
- 88,000+ binaries
- S3 mirror
- No need to build from source code!



E4S Spack build cache:

- **Fusion plasma:**
  - WDMapp added E4S mirror
    - Speedup: 10X
- **Turbine wind plant:**
  - ExaWind (Nalu-Wind)
    - 6 minutes with build cache
    - Up to 4 hours without





Special thanks to Sameer Shende, WDMapp and ExaWind teams

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

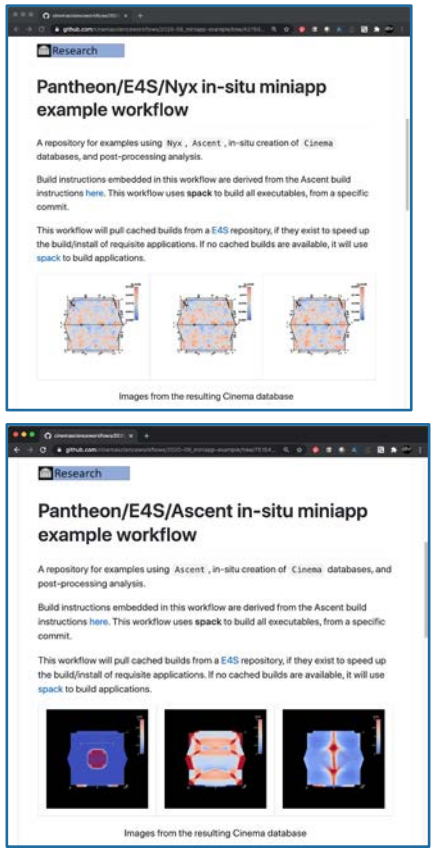
Pantheon and E4S build cache support end-to-end ECP examples

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

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

Contributions

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

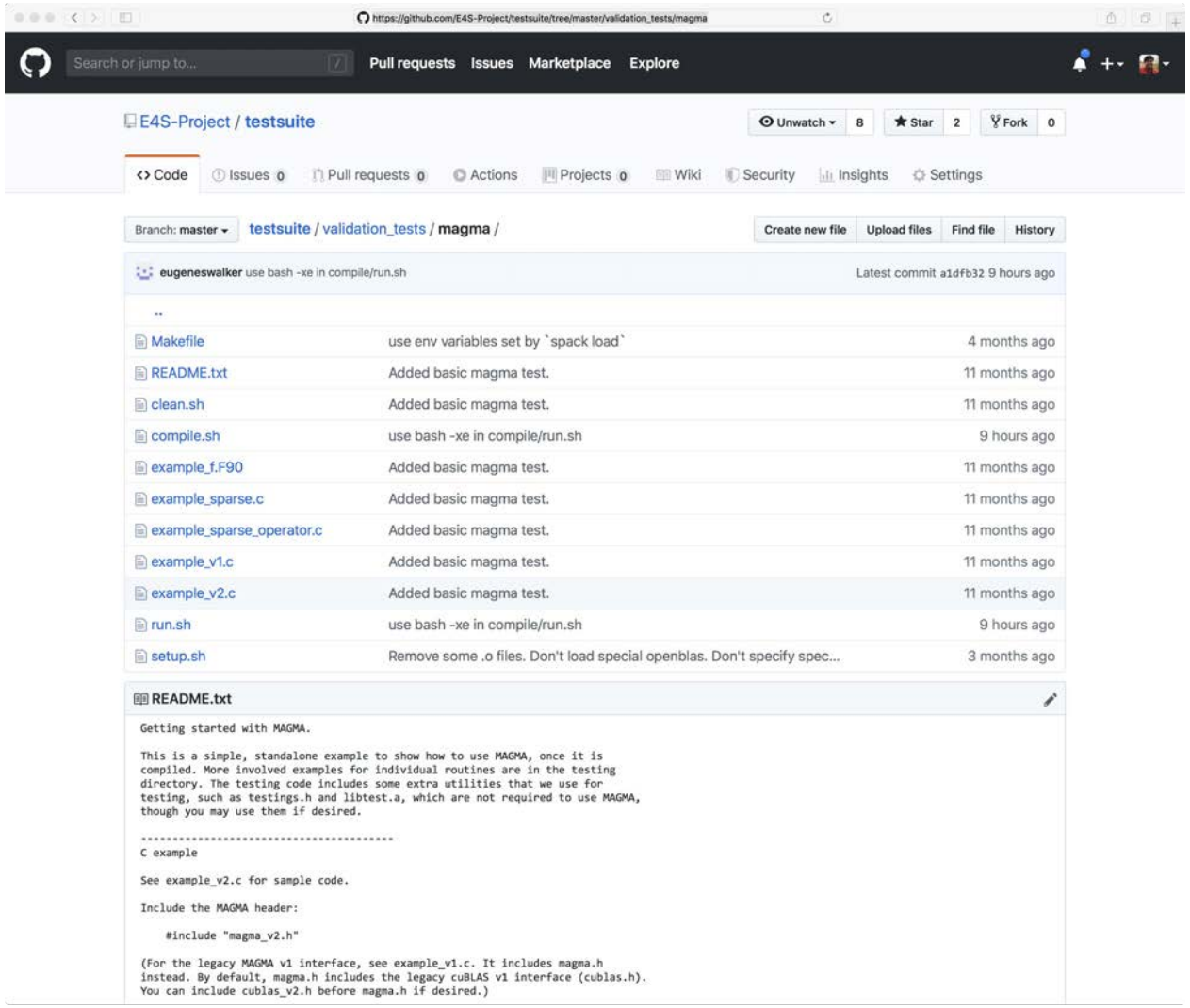


Instructions page for (top) Nyx, Ascent and Cinema workflow repository, and (bottom) Cloverleaf3d, Ascent, Cinema workflow. These curated workflows use Pantheon, E4S and spack to provide curated workflows for ECP.

LA-UR-20-27327 12/6/20

E4S Validation Test Suite

- Automate build and run tests using architecture-specific spack recipes
- Provide test suite to validate container environments and products
- Populate dashboard



- git clone https://github.com/E4S-Project/testsuite.git

Frank – Designed for Libs & Tools Developers

- Prep system for ECP libs & tools
- Access to latest non-NDA HW/SW
- Shared file system – 1 copy of SW
- Port to many device types at once
- Porting support from E4S team
- CI testing workhorse (500K builds)
- Next: Bare metal, BIOS-changing support for low-level software work

This is a list of all OACISS serve

The NetworkInfrastructure page
short hostname is needed for ss
The Service:storage describes a
Click on the server links to acce
OACISS has a large amount of s

Name
Compute: Orthus
Compute: Jupiter
Compute: Saturn
Compute: Reptar
Compute: Illyad
Compute: Gilgamesh
Compute: Instinct
Compute: Voltar
Compute: Cyclops
Compute: Gorgon
Compute: Medusa
Compute: Typhon
Compute: Delphi
Compute: Aurora
Compute: Godzilla
Compute: Centaur
Compute: Minotaur
Compute: Eagle
Compute: Pegasus
Compute: Vina
Compute: Pike
Compute: Cirrus-AIX.stor
Compute: Cumulus-AIX.stor
Compute: Nimbus-AIX.stor
Compute: KNL Grover
Compute: Axis cluster (axis1-8)

Name
Visualization: Chymera
Visualization: Cerberus
NUC cluster
Jetson ARM64 cluster
Jetson ARM64 cluster
Compute: Xavier
Compute: OD1K
Compute: Omicron
Compute: Sever
Compute: Silicon

Name
Infrastructure:orion
Infrastructure:mecha
Infrastructure:newstorage
Infrastructure:mnemosyne
Infrastructure:lighthouse

Description
Primary login gateway
Quad Cooper lake + Intel DG1
Quad Cooper lake + A100 (80GB)
Cascade lake 6248 node
AMD + 2 A100 (40GB)
AMD + 2 MI50 + A100 (40GB)
Intel + 2 AMD MI100 + MI50
A100 (80GB) + P100 + V100 GPU node
IBM Power9 + 4 V100
IBM Power9 + 4 V100
IBM Power9
IBM Power9
Intel + GV100
NEC SX-Aurora demo machine
Intel DG1 + 2 x K80 node
IBM Power8 + 2 K80
IBM Power8 + 2 K80
IBM Power9 + 3 x T4
Compute node
Raptor Talos II
Raptor Talos II + MI25
AIX machine
AIX machine
AIX machine
Intel Phi system
DL580 G7 nodes

for systems of that type.

within the OACISS racks in the machine room. All OACISS systems automatically search .nic.uoregon.edu for DNS, so only the

ays (orthus, cerberus) are accessible by machines outside of nic.uoregon.edu.

latacenter

	Processors	Local Network	Physical location
	2 x 8c Xeon E5-2667 v2 @ 3.3GHz	10GbE	
	4 x 24c Xeon Gold 6438 @ 2.3GHz	100GbE + EDR	R86.U10
	4 x 26c Xeon Platinum 8367HC @ 3.2GHz	100GbE + EDR	R86.U10
	2 x 24c Xeon Gold 6248R @ 2.9GHz	10GbE + 100GbE	R84.U37
	2 x 24c Epyc Rome 7402 @ 2.8GHz	100GbE + 2xEDR	R85.U22
	2 x 24c Epyc Milan 7413 @ 2.6GHz	100GbE + 2xEDR	R85.U26
	2 x 14c Xeon E5-2660 v4 2.0GHz	100GbE	R85.U6
	2 x 16c Xeon Gold 6226R @ 2.9GHz	10GbE + EDR	R86.U26
	2 x 20c Power9 @ 3.66GHz	10GbE + 2xHDR (200 Gbps)	R86.18
	2 x 20c Power9 @ 3.66GHz	10GbE + 2xHDR (200 Gbps)	R86.U16
	2 x 20c Power9 @ 3.66GHz	10GbE	R86.U14
	2 x 20c Power9 @ 3.66GHz	10GbE	R86.U12
	2 x 18c Xeon E5-2697 v4	100GbE	R86.U35
ector Engine	8c Xeon 4108 Silver @ 1.8GHz	10GbE + EDR	R85.U31
	2 x 14c Xeon E5-2680v4 @ 2.3GHz	40GbE + EDR	R85.U6
	2 x 20c Power8 @ 3.5GHz	10GbE	R85.U18
	2 x 20c Power8 @ 3.5GHz	10GbE	R85.U20
	2 x 16c Power9 @ 2.1GHz	10GbE + 2xEDR	R86.U24
	2 x 18c Xeon Gold 6140 @ 2.3GHz	100GbE + EDR	R86.U22
	2 x 22c Power9 @ 2.2GHz	10GbE	R84.U44
	2 x 22c Power9 @ 2.2GHz	10GbE	R84.U29

rm

inger

Description
Drives 8K display in 472
Secondary login gateway; Jetson/Nucs + NFS
Intel NUCs (16)
Tegra TX-1
Tegra TX-2
NVidia Tegra 3
ARM64 v8
M1 Mac
Intel Xe
VLSI simulation node

E4S Dashboard at https://dashboard.e4s.io/

Facility Deployments - ROCm-enabled Specs

Spec	Crusher, PrgEnv-gnu (ROCm 5.1)	Crusher, PrgEnv-cray (ROCm 5.1)	Crusher, PrgEnv-amd (ROCm 5.1)
amrex@22.05	OK	OK	OK
arborx@1.2	OK	OK	OK
cabana@0.4.0	Not Installed Yet	Not Installed Yet	Not Installed Yet
chai@2.4.0	OK	OK	OK
gasnet@2022.3.0	OK	OK	OK
ginkgo@1.4.0	OK	OK	OK
heffte@2.2.0	OK	OK	OK
hpctoolkit@2022.04.15	Not Installed Yet	Not Installed Yet	Not Installed Yet
hpx@1.7.1	OK	OK	OK
kokkos@3.6.00	OK	OK	OK
magma@2.6.2	OK	OK	OK
mfem@4.4.0	OK	OK	OK
papi@6.0.0.1	Not Installed Yet	Not Installed Yet	Not Installed Yet
petsc@3.17.1	OK	OK	OK
raja@0.14.0	OK	OK	OK
slate@2021.05.02	Not Installed Yet	Not Installed Yet	Not Installed Yet
slepc@3.17.1	OK	Not Installed Yet	Not Installed Yet
strumpack@6.3.1	OK	Not Installed Yet	Not Installed Yet
sundials@6.2.0	OK	OK	OK
superlu-dist@7.2.0	OK	OK	OK
tasmanian@7.7	OK	OK	OK
tau@2.31.1	OK	OK	OK
umpire@6.0.0	OK	OK	OK
upcxx@6.0.0	OK	OK	OK
vtk-m@1.7.1	Not Installed Yet	Not Installed Yet	Not Installed Yet

E4S Community Policies: *A commitment to quality improvement*

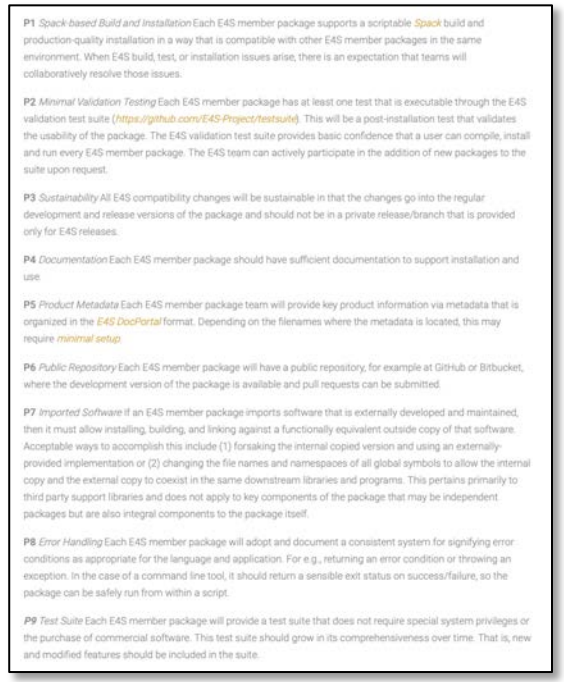
- Enhance sustainability and interoperability
- Serve as membership criteria for E4S
  - Membership is not required for *inclusion* in E4S
  - Also includes forward-looking draft policies
- Modeled after xSDK community policies
- Multi-year effort led by SDK team
  - Included representation from across ST
  - Multiple rounds of feedback incorporated from ST leadership and membership

SDK lead: Jim Willenbring (SNL)

Policies: Version 1

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

- **P1: *Spack-based Build and Installation***
- **P2: *Minimal Validation Testing***
- **P3: *Sustainability***
- **P4: *Documentation***
- **P5: *Product Metadata***
- **P6: *Public Repository***
- **P7: *Imported Software***
- **P8: *Error Handling***
- **P9: *Test Suite***



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.

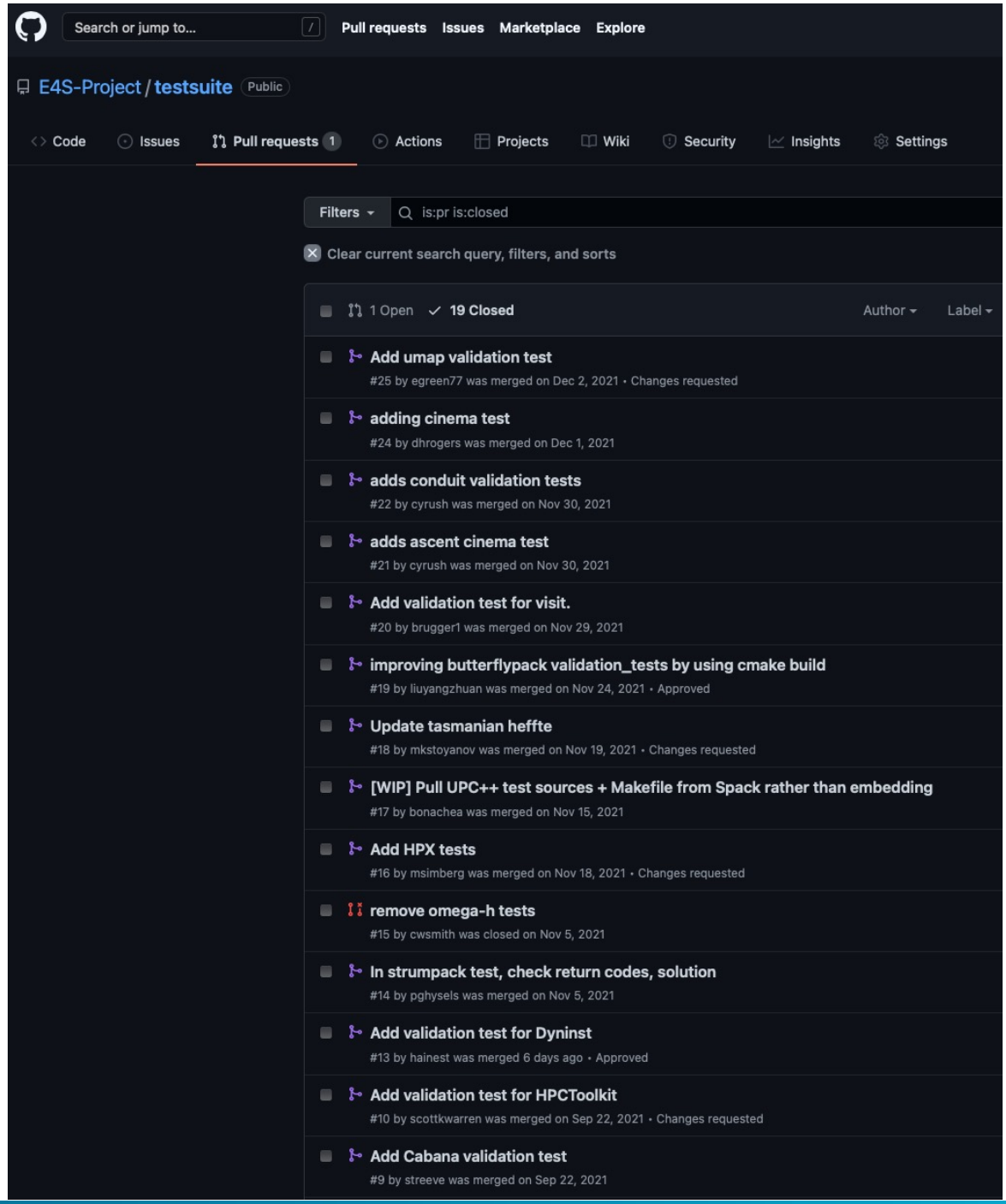
P7 *Imported Software* If an E4S member package imports software that is externally developed and maintained, then it must allow installing, building, and linking against a functionally equivalent outside copy of that software. Acceptable ways to accomplish this include (1) forsaking the internal copied 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 or throwing an exception. In the case of a command line tool, it should return a sensible exit status on success/failure, so the package can be safely run from within a script.

P9 *Test Suite* Each E4S member package will provide a test suite that does not require special system privileges or the purchase of commercial software. This test suite should grow in its comprehensiveness over time. That is, new and modified features should be included in the suite.

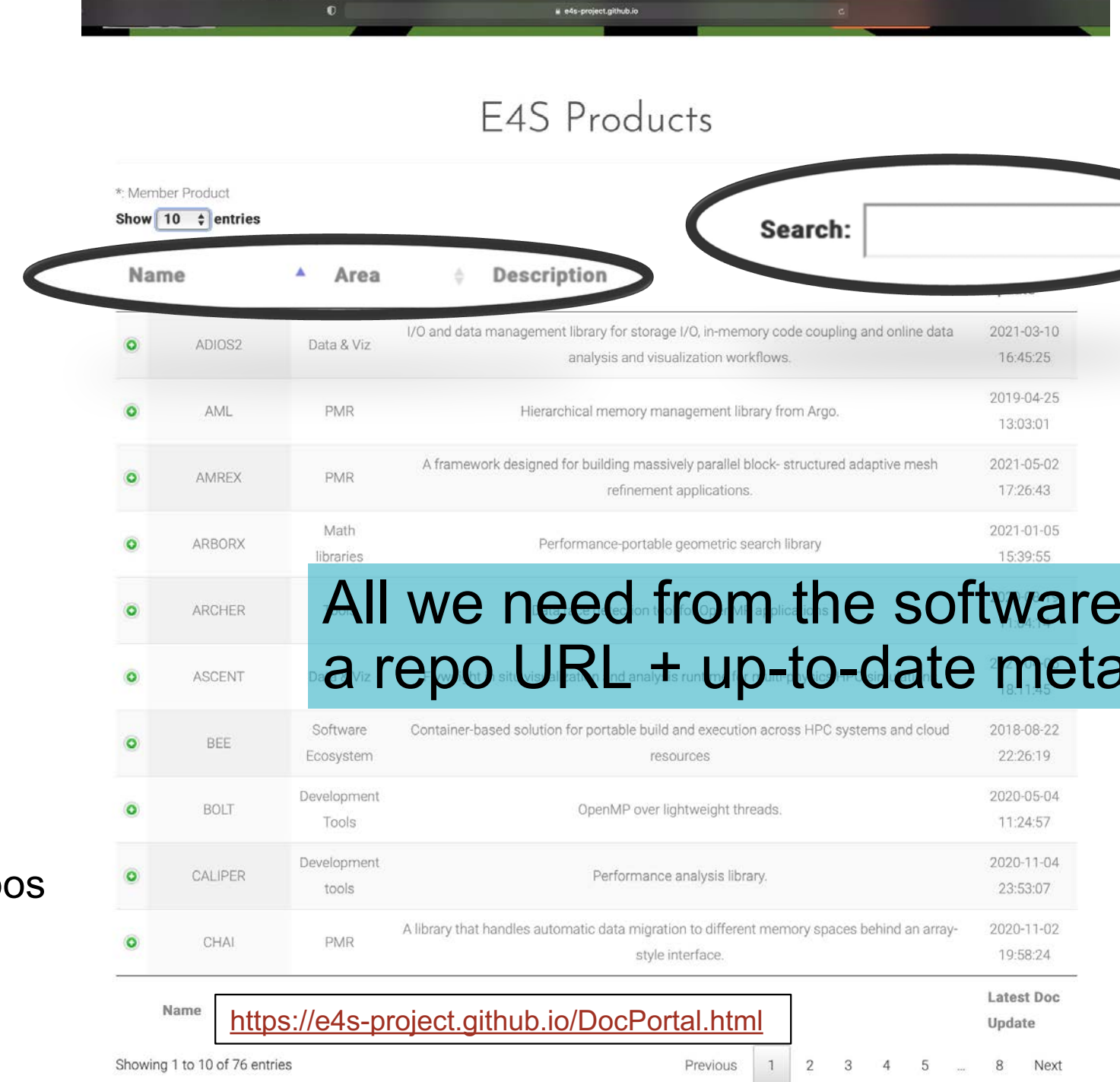
Request for E4S Policy Status Drove Software Improvements

- L4 Project reviews required gap assessment against E4S Policies
- But no requirement to increase compatibility
- However, teams responded by reducing gaps
- On the right:
  - Flurry of E4S Validation Test Suite PRs prior to reviews
  - Other low hanging fruit changes made too



E4S DocPortal

- Single point of access
- All E4S products
- Summary Info
  - Name
  - Functional Area
  - Description
  - License
- Searchable
- Sortable
- Rendered daily from repos



All we need from the software team is a repo URL + up-to-date meta-data files

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

Expanding the Value and Impact of Software Ecosystems Going Forward

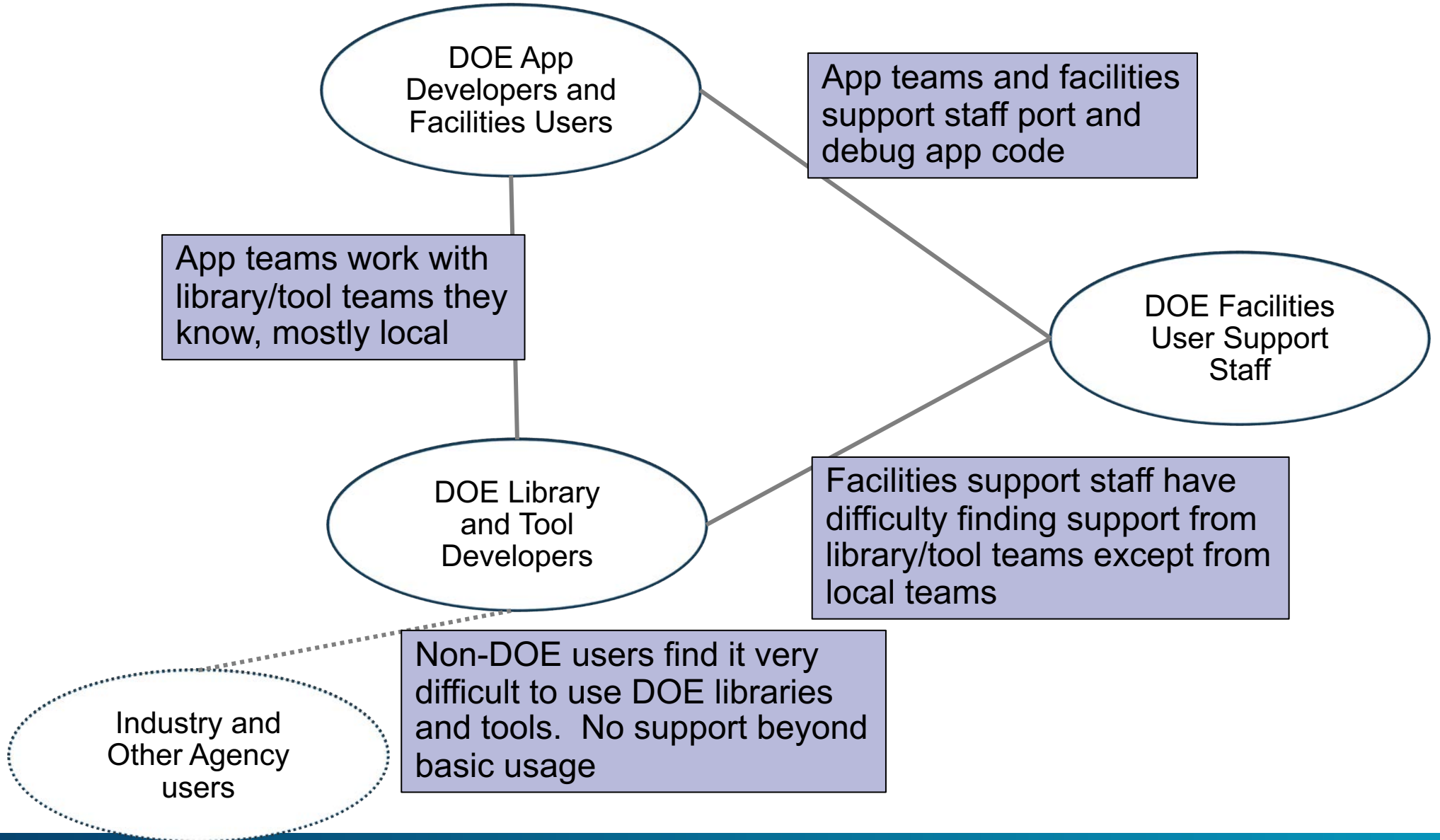
Mike Heroux

E4S and SDKs as platforms are providing tremendous value

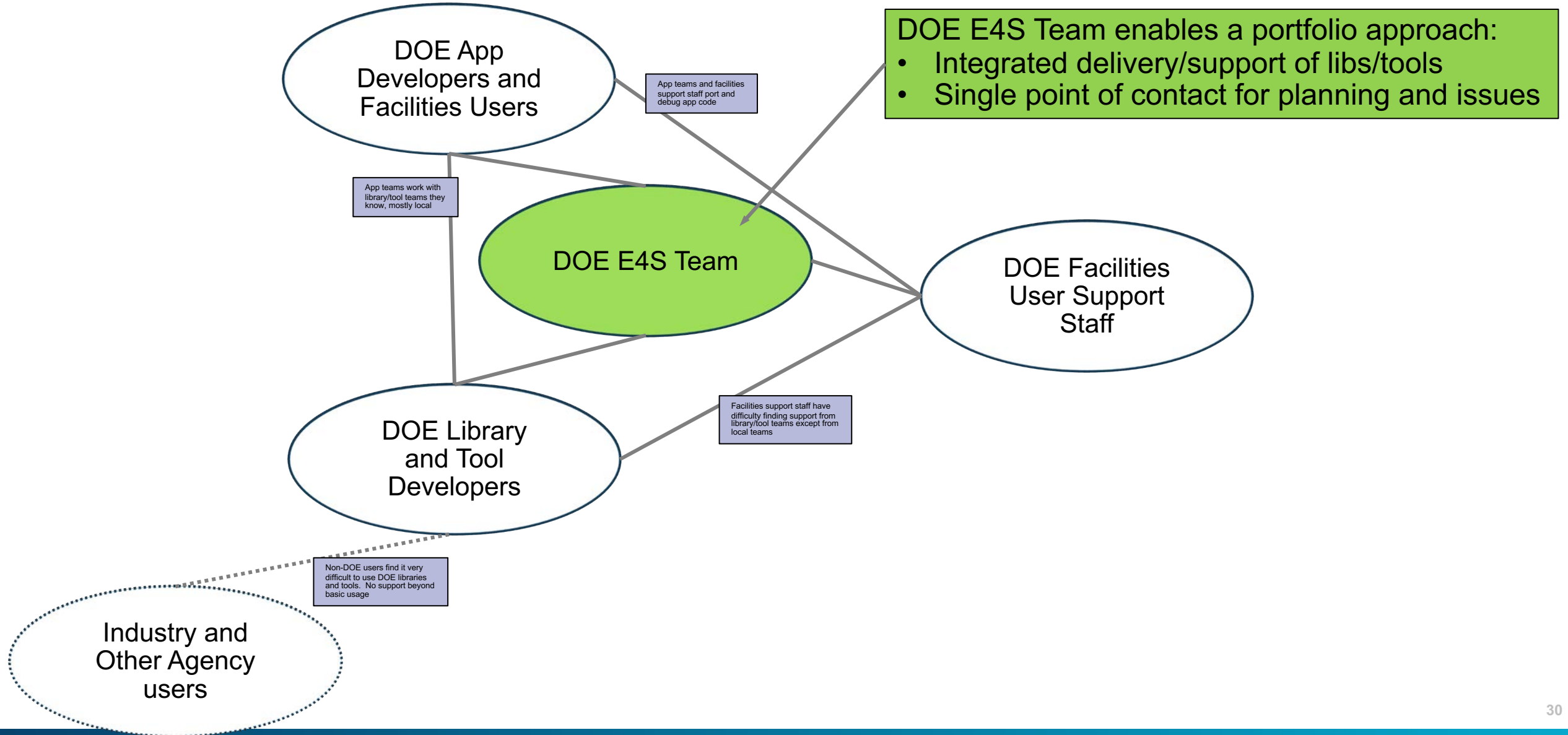
Activity	SDKs	E4S
Planning	Transparent and collaborative requirements, analysis and design, delivery – better plans, less effort, improved complementarity	Campaign-based portfolio planning coordinated with Facilities, vendors, community ecosystem, non-DOE partners
Implementation	Leverage shared knowledge, infrastructure, best practices	ID and assist product teams with crosscutting issues
Cultivating Community	Within a specific technical domain: Portability layers, LLVM coordination, sparse solvers, etc.	Across delivery and deployment, with software teams, facilities' staff, with non-DOE users in industry, US agencies
Resolving issues, sharing solutions	Performance bottlenecks and tricks, coordinated packaging and use of substrate, e.g., Desul for RAJA and Kokkos	Build system bugs and enhancements, protocols for triage, tracking & resolution, leverage across & beyond DOE
Improving quality	Shared practice improvement, domain-specific quality policies, reduced incidental differences and redundancies, per-commit CI testing of portfolio	Portfolio-wide quality policies with assessment process and quality improvement efforts, documentation portal, portfolio testing on many platforms not available to developers. Address supply chain needs
Path-finding	Collaborative exploration and development of leading-edge tools and processes	Exploration and development of leading-edge packaging and distribution tools and workflows that provide capabilities and guidance for others
Training	Collaborative content creation and curation, coordinated training events for domain users, deep, problem-focused solutions using multiple products	Portfolio installation and use, set up of build caches, turnkey and portable installations, container and cloud instances
Developer experience	Increased community interaction, increased overhead (some devs question value), improved R&D exploration, e.g., variable precision	Low-cost product visibility via doc portal, wide distribution via E4S as from-source/pre-installed/container environment
User experience	Improve multi-product use, better APIs through improved design, easier understanding of what to use when	Rapid access to latest stable feature sets, installation on almost any HPC system, leadership to laptop
Scientific Software R&D	Shared knowledge of new algorithmic advances, licensing, build tools, and more	Programmatic cultivation of scientific software R&D not possible at smaller scales
Community development	Attractive and collaborative community that attracts junior members to join, establishes multi-institutional friendships & careers	Programmatic cultivation of community through outreach and funded opportunities that expand the sustainable membership possibilities

The SDK and E4S platforms provide compelling value for modest cost in ways that become more important going forward

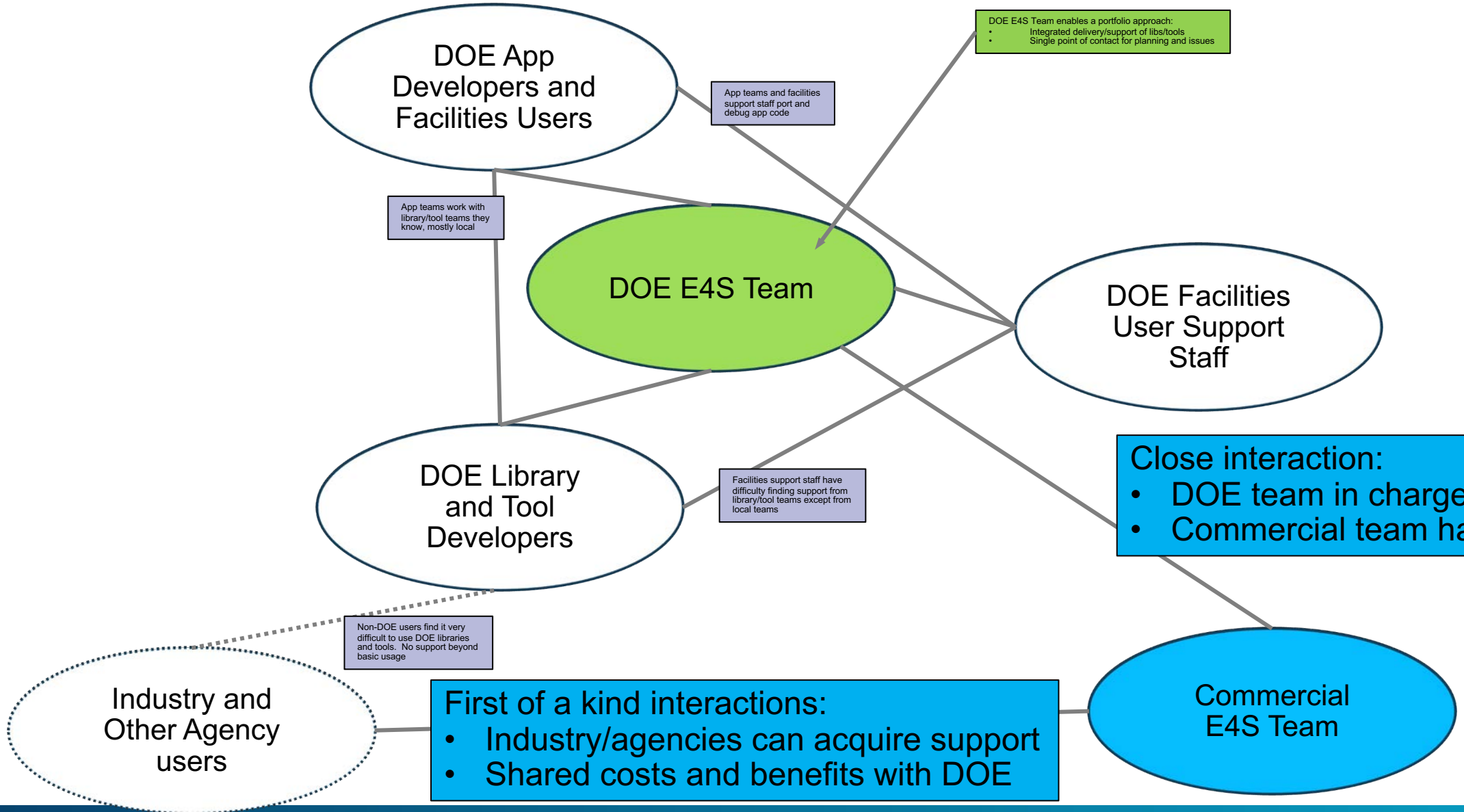
Pre-E4S User Support Model



E4S Phase 1 Support Model – Old relationships plus DOE E4S



E4S Phase 2 Support Model – Previous plus commercial E4S



Expanding the Scope of Cost and Benefit Sharing for DOE Software Libraries and Tools

Support Phase	Primary Scope	Primary Cost and Benefit Sharing Opportunities
Pre-E4S	Local facility	**Local costs and benefits:** Prior to ECP and E4S, libraries and tools were typically strongly connected to the local facility: ANL libs and tools at ALCF, LBL at NERSC, LLNL at Livermore Computing, etc.
+ ECP E4S	All DOE facilities	**DOE complex-shared costs and benefits:** ECP requires, and E4S enables, interfacility availability and use of libs across all facilities: First-class support of ANL libs and tools at other facilities, etc.
+ Commercial E4S	DOE facilities, other US agencies, industry, and more	**Universal shared costs and benefits:** Commercial support of E4S expands cost and benefit sharing to non-DOE entities: DOE costs are lower, software hardening more rapid. US agencies, industry and others can contract for support, gaining sustainable use of E4S software and contributing to its overall support.

Software Sustainability Activities

Responses to the 2021 IPR Recommendation to work with ASCR on Software Sustainability

LSSw Meeting 10: Thursday, July 28, 2022, 3 - 4:30 pm ET

Topic: Expanding Laboratory, University, and Industry Collaborations: An Industry Panel Discussion

- **Description:** The open-source scientific software community benefits from complementary and leveraged contributions from universities, laboratories, and industry. Numerous partnerships are already in place, but more opportunities exist. The cost of making high-quality scientific software libraries and tools has decreased due to widely used tools and platforms such as GitHub, and the need for high-quality software ecosystems has increased due to growing scientific demands and increased interconnection between scientific disciplines. The importance of collaboration in sustaining and leveraging laboratory, university, and industry investments is even more important as we go forward. In this panel discussion, we bring community members with strong industry experience together to explore how we can further improve leverage and complementarity so that the whole scientific community can realize the benefits of new software capabilities as they emerge.
- This month our panelists are:
  - John Cary, Tech-X Corp
  - Sarah Knepper, Intel Corp
  - Pete Mendrygal, HPE, Inc
  - Jeff Larkin, NVIDIA Corp
  - Bob Lucas, ANSYS, Inc
- In opening remarks, panelists briefly address the following questions from their perspectives:
  - What are some existing examples of scientific software collaboration between federal agency-sponsored programs (at labs and universities) and software vendor product development?
  - What has worked and not worked well with past leverage and complementarity efforts?
  - What are some near-term opportunities to improve leverage and complementarity?
  - What are some long-term opportunities and constraints on leverage and complementarity?
- Why attend: To discuss the opportunities and strategies for improved leverage and complementarity of laboratory, university, and industry scientific software efforts.

https://lssw.io

First-of-a-kind US DOE Workshop

- The Science of Scientific-Software Development and Use
  - Dec 13 – 16, 2021
  - https://www.orau.gov/SSSDU2021
- Workshop Brochure available:
  - https://doi.org/10.2172/1846008
- Workshop Report in progress:
  - 3 Priority Research Directions
  - 3 Crosscutting Themes

SSSDU Priority Research Directions

- **PRD1: Develop methodologies and tools to comprehensively improve team-based scientific software development and use** Focus: Team Impact
  - **Key question:** *What practices, processes, and tools can help improve the development, sustainment, evolution, and use of scientific software by teams?*
- **PRD2: Develop next-generation tools to enhance developer productivity and software sustainability** Focus: Developer Impact
  - **Key questions:** *How can we create and adapt tools to improve developer effectiveness and efficiency, software sustainability, and support for the continuous evolution of software? How can we support and encourage the adoption of such tools by developers?*
- **PRD3: Develop methodologies, tools, and infrastructure for trustworthy software-intensive science** Focus: Societal Impact
  - **Key questions:** *How can we facilitate and encourage effective and efficient reuse of data and software from third parties while ensuring the integrity of our software and the resulting science? How can we provide flexible environments that "bake in" the tracking of software, provenance, and experiment management required to support peer review and reproducibility?*

SSSDU Cross-cutting Themes

- **Theme 1:** We need to consider **both human and technical elements** to better understand how to improve the development and use of scientific software.
- **Theme 2:** We need to address urgent challenges in **workforce recruitment and retention** in the computing sciences with growth through **expanded diversity, stable career paths, and the creation of a community and culture** that attract and retain **new generations** of scientists.
- **Theme 3:** Scientific software has become essential to all areas of science and technology, creating opportunities for **expanded partnerships, collaboration, and impact**.

Takeaways from Expanding Impact in the Future

- Introduction of commercial support for E4S users makes broad benefit & cost sharing possible
- Other agencies & industry can use E4S with confidence because they can acquire support
- The pursuit of effective and efficient scientific software can itself be informed by science

Software Sustainability Organization (SSO) Scenarios

Software portfolio assumptions: Motivation for creating SSO

- DOE Libraries and Tools (called Products in remaining discussion) will be managed as a portfolio
- Portfolio aggregation occurs at two levels
  - SDK – Aggregation of similar products for coordination, interoperability, and improvement
  - E4S – Comprehensive collection of all products and dependencies for deployment
- DOE wants:
  - These products to be high quality, available across facilities, available as a stack
  - New products, and evolution of important existing products
  - Transitioning to community or vendor stacks of stable, mature products
  - Sunsetting of products that don't reach critical usability levels
- There will be some form of a cross-lab software organization, even if loosely coordinated
  - Calling it a Software Sustainability Organization (SSO) in these slides

Software portfolio management activities

- Based on assumptions, on a regular basis, SSO stakeholders need to:
  - **Identify:** The emerging needs in scientific libraries and tools
  - **Select**: New products, and new functionality within existing products
  - **Retain**: Existing products, measure ongoing need and impact of recent efforts
  - **Transition**: Trim existing product portfolio to make room for new efforts
    - Identify and plan transition of products and functionality that can move to non-DOE support environment
    - Identify and plan transition of products that are no longer, or did not become, sufficiently useful
- Roles & responsibilities of stakeholders (DOE, lab management, apps, SSO, etc.)
  - Under discussion
  - Basics should be easy; details will take time
  - Will likely evolve with experience

SSO organizational strategies

- Hub and spoke model: Hybrid centralized-distributed approach
- Hub:
  - Hub works with product teams on integration, quality improvement, usability, availability
  - Hub can have relationships with industry, other agencies
  - Hub would provide assessment of spoke product quality and impact within the hub:
    - Good member of E4S? SDK?
    - Good integration into client environments?
    - Assessment used by product sponsors to review and select future product sponsorship
  - Hub reviewed independently from spokes
- Spokes:
  - A spoke could be a collection of products organized by sponsor, technical area, other…
  - New spokes can be added, removed as needed
  - Products and functionality selected to address mission needs, derived requirements, anticipated future needs
- Strategy supports multiple stakeholders with distinct and overlapping roles and responsibilities

A tiered approach to development, delivery, deployment & support

Split ECP libraries & tools efforts into four “layers”:

Layer	Scope
Ecosystem	SSO Leadership, SQA, SW R&D, E4S/SDKs, support product integration/deployment
Harden & Delivery	Make product ready for broad use, integrate into SDK/E4S, deliver to clients
Port & Optimize	Adapt software to new platform and optimize it
Develop capabilities	Develop new functionality, new algorithms

Budget Model for SSO: Derived from ECP ST portfolio

Activity	Budget/Product/Year	Notes
Harden & Deliver	$125K - $250K/Product	Varies with novelty of new system
Port & Optimize	$125K - $250K/Product	
Develop Capabilities	$250K - $500K/Product	

	Scenarios – 1 year budget							
Tier	50 Products Low system novelty		50 Products High system novelty		75 Products Low system novelty		75 Products High system novelty	
	Activity	Total	Activity	Total	Activity	Total	Activity	Total
Base Ecosystem	$10.0M	$10.0M	$10.0M	$10.0M	$10.0M	$10.0M	$10.0M	$10.0M
+ Harden & Deliver	$6.3M	$16.3M	$12.5M	$22.5M	$9.4M	$19.4M	$18.8M	$28.8M
+ Port & Optimize	$6.2M	$22.5M	$12.5M	$35.0M	$9.4M	$28.8M	$19.7M	$47.5M
+ Develop Capabilities	$12.5M	$35.0M	$25.0M	$60.0M	$19.7M	$47.5M	$37.5M	$85.0M

ASCR Programs and Software Sustainability Organization (SSO)

Element/Project Type	ASCR Research	SciDAC	SSO	Facilities
App Engagement Model	Indirect, future	Direct, selected apps	Direct, broad collection	ESP, CAAR, NESAP, direct user support
FOA approach	Most open, others invited	Open	Hybrid* based on gap analysis	RFP based on strategic analysis
Scope selection	ASCR PMs	ASCR PMs, Institute lab leads, partnerships driven by other offices	Stakeholders*, gap-driven	RFP selection
Scope management	Project PI	Institute, Lab leads	SSO leads, lab leadership	Formal EVM project
Scope review	ASCR PMs, independent review	ASCR PMs, independent review	Stakeholders*, independent review	Formal EVM project
Scope assessment cycle	3 years	5 years	Annual, ongoing	Annual, ongoing
Software Delivery Model	Not emphasized	One-to-one: lib/tool to app	Portfolio: SDK, E4S	Vendor stack
Deliverables	Papers (on math/CS, algorithms)	Papers (on algorithms), targeted client integrations	SDK, E4S integrations, papers (on software)	Leadership system, user support
Software TRL**	**1 – 3**	**3 – 5**	**5 – 9**	**7 – 9**

Key point: SSO fills a long-term technology readiness level (TRL) gap

*Stakeholders: ASCR, Apps, Facilities, Vendors, …

**TRL DOE (1-9): https://www.gao.gov/assets/gao-20-48g.pdf, p 118

Summary

- Using a portfolio-based approach for HPC software is about **going together vs going alone**
- While products vary greatly, we all face the **same frontiers**: **Evolving demands and systems**
- Success on the frontier is important for all HPC configurations: **leadership to laptop**
- Progress is made by **collaborating and sharing** information
- The new and evolving E4S and SDK platforms enable **better, faster and cheaper**, in net
- A collective approach, E4S, enables **new relationships with facilities, vendors, apps, industry, other US agencies, international partners**
- Improving how we **develop & use software is an important scientific effort itself**
- Software ecosystem futures
  - ECP has given us a chance to bootstrap a portfolio-based scientific software ecosystem
  - We are encouraged by the potential for post-ECP funding opportunities to continue and expand this work
  - Expansion can include partnerships with other US agencies, industry, and internationally

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.

ECP Director: Doug Kothe
ECP Deputy Director: Lori Diachin

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

Questions?