E4S at DOE Facilities with Deep Dive at NERSC

Sameer Shende, University of Oregon  
E4S Technical Lead

Monday, Oct. 4, 2021, 9am – noon PT  
Zoom:  https://exascaleproject.zoomgov.com/j/1612548862

Slides: https://e4s.io/talks/E4S_NERSC21.pdf
The Growing Complexity of Scientific Application Software Stacks
Challenges

• As our software gets more complex, it is getting harder to install tools and libraries correctly in an integrated and interoperable software stack.
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
Scientific software is becoming extremely complex
Even proprietary codes are based on many open source libraries.

- Half of this DAG is external (blue); more than half of it is open source.
- Nearly all of it needs to be built specially for HPC to get the best performance.
The Exascale Computing Project is building an entire ecosystem

- 15+ applications
- 80+ software packages
- 5+ target architectures/platforms
  - Xeon
  - Power
  - KNL
  - NVIDIA
  - ARM
  - Laptops?

- Up to 7 compilers
  - Intel
  - GCC
  - Clang
  - XL
  - PGI
  - Cray
  - NAG

- 10+ Programming Models
  - OpenMPI
  - MPICH
  - MVAPICH
  - OpenMP
  - CUDA
  - OpenACC
  - Dharma
  - Legion
  - RAJA
  - Kokkos

- 2-3 versions of each package + external dependencies

= up to 1,260,000 combinations!

- Every application has its own stack of dependencies.
- Developers, users, and facilities dedicate (many) FTEs to building & porting.
- Often trade reuse and usability for performance.

We must make it easier to rely on others’ software!
How to install software on a supercomputer

1. Download all 16 tarballs you need
2. Start building!
3. Run code
4. Segfault!?
5. Start over…
What about modules?

- Most supercomputers deploy some form of *environment modules*
  - TCL modules (dates back to 1995) and Lmod (from TACC) are the most popular

```bash
$ gcc
-bash: gcc: command not found

$ module load gcc/7.0.1
$ gcc --dumpversion
7.0.1
```

- Modules don’t handle installation!
  - They only modify your environment (things like PATH, LD_LIBRARY_PATH, etc.)

- Someone (likely a team of people) has already installed gcc for you!
  - Also, you can *only* `module load` the things they’ve installed
Spack Overview
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.
- https://spack.io
Spack is a flexible package manager for HPC

- How to install Spack (works out of the box):
  
  ```bash
  $ git clone https://github.com/spack/spack
  $ ./spack/share/spack/setup-environment.sh
  ```

- How to install a package:
  
  ```bash
  $ spack install tau
  ```

- TAU and its dependencies are installed within the Spack directory.

- Unlike typical package managers, Spack can also install many variants of the same build.
  - Different compilers
  - Different MPI implementations
  - Different build options

Visit spack.io
github.com/spack/spack
@spackpm
Spack provides the *spec* syntax to describe custom configurations

```bash
$ git clone https://github.com/spack/spack
$ . spack/share/spack/setup-env.sh
$ spack compiler find
$ spack external find
```

```bash
$ spack install tau
$ spack install tau@2.30.1
$ spack install tau@2.30.1 %gcc@9.3.0
$ spack install tau@2.30.1 %gcc@9.3.0 +level_zero +openmpi
$ spack install tau@2.30.1 %gcc@9.3.0 +mpi ^mvapich2@2.3~wrapperrpath
```

• Each expression is a *spec* for a particular configuration
  - Each clause adds a constraint to the spec
  - Constraints are optional – specify only what you need.
  - Customize install on the command line!

• Spec syntax is recursive
  - Full control over the combinatorial build space
`spack find` shows what is installed

- All the versions coexist!
  - Multiple versions of same package are ok.
- Packages are installed to automatically find correct dependencies.
- Binaries work regardless of user’s environment.
- Spack also generates module files.
  - Don’t have to use them.
The Spack community is growing rapidly

- **Spack simplifies HPC software for:**
  - Users
  - Developers
  - Cluster installations
  - The largest HPC facilities

- **Spack is central to ECP’s software strategy**
  - Enable software reuse for developers and users
  - Allow the facilities to consume the entire ECP stack

- **The roadmap is packed with new features:**
  - Building the ECP software distribution
  - Better workflows for building containers
  - Stacks for facilities
  - Chains for rapid dev workflow
  - Optimized binaries
  - Better dependency resolution

Visit spack.io

github.com/spack/spack

@spackpm
The Extreme-Scale Scientific Software Stack (E4S) Components
E4S: Extreme-scale Scientific Software Stack Components

- Curated, Spack based software distribution
- Spack binary build caches for bare-metal installs
  - x86_64, ppc64le (IBM Power 9), and aarch64 (ARM64)
- Container images on DockerHub and E4S website of pre-built binaries of ECP ST products
- Base images and full featured containers (with GPU support)
- GitHub recipes for creating custom images from base images
- GitLab integration for building E4S images
- E4S validation test suite on GitHub
- E4S-cl container launcher tool for MPI substitution in applications using MPICH ABI
- E4S VirtualBox image with support for container runtimes
  - Docker
  - Singularity
  - Shifter
  - Charliecloud
- AWS and GCP images to deploy E4S
  
  [https://e4s.io](https://e4s.io)
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
- 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
- Feb 2021: E4S 21.02 - 67 full release, 4 partial release
- May 2021: E4S 21.05 - 76 full release products
- August 2021: E4S 21.08 - 88 full release products

https://e4s.io

Lead: Sameer Shende (U Oregon)

Also include other products .e.g., AI: PyTorch, TensorFlow, Horovod
Co-Design: AMReX, Cabana, MFEM
Using E4S with containers
What are containers

A lightweight collection of executable software that encapsulates everything needed to run a single specific task
  Minus the OS kernel
  Based on Linux only
Processes and all user-level software is isolated
Creates a portable* software ecosystem
Think chroot on steroids
Docker most common tool today
  Available on all major platforms
  Widely used in industry
  Integrated container registry via Dockerhub
Hypervisors and Containers

Type 1 hypervisors insert layer below host OS
Type 2 hypervisors work as or within the host OS
Containers do not abstract hardware, instead provide “enhanced chroot” to create isolated environment
Location of abstraction can have impact on performance
All enable custom software stacks on existing hardware
E4S Docker and Singularity Containers

Using E4S Containers

The current E4S container offerings include Docker images for Red Hat Enterprise Linux 7, Red Hat Enterprise Linux 8, Ubuntu 18.04 (Bionic), and Ubuntu 20.04 (Focal Fossa). Our images can run on x86_64, PPA64, and AARCH64 depending on the particular image. In addition to offering a full E4S image containing a comprehensive selection of E4S software released on a quarterly cycle, we also offer a set of minimal base images suitable for use in Continuous Integration (CI) pipelines.

Docker images are available on the E4S Docker Hub.

Recipes for building images from scratch are available on the E4S GitLab repository.

Our recipes make use of Spack packages available as pre-built binaries in the E4S Build Cache.

Container Releases

- Docker Downloads
- Singularity x86_64 Download
- Singularity ppc64le Download
- DVA Download

From source with Spack

Visit the Spack Project

Spack contains packages for all of the products listed in the E4S 2021.08 Full Release category (see above 2021.02 Release Notes). General instructions for building software with Spack can be found at the Spack website. For more information, see /usr/local/packages/ep in the container referenced here. Questions concerning building those packages are deferred to the associated package development team.

AWS EC2 Image

The E4S 21.05 release is also available on AWS as an EC2 AMI with ID ami-05f7d60e83b63d67d in the US-West-2 (Oregon) region.

Note on Container Images

Container images contain binary versions of the Full Release packages listed above. A clone of Spack is also available in the container which can be used to compile the Full Release and Partial Release packages. Example Spack "recipes" (lists of configuration commands) are available in the container. See the README.txt file for more details. This release also includes an OVA file that has Docker, Charlecloud, Shiftir, and Singularity preinstalled in it. The Docker container image is also available from Dockerhub:

```
# docker pull ecps4c/ubuntu8.04-e4s-gpu
```
E4S base images for custom container deployment and CI images

E4S GPU Images
Multi-Arch Image (X86_64 and PPC64LE)
This is a multi-arch image, meaning that the same image name can be used to pull the appropriate image for your architecture.
- ecpe4s/ubuntu:18.04-e4s-gpu

Continuous Integration Images
- ecpe4s/thf7-runner-x86_64
- ecpe4s/thf8-runner-x86_64
- ecpe4s/ubuntu:18.04-runner-x86_64
- ecpe4s/ubuntu:20.04-runner-x86_64

Custom Images
- ecpe4s/ubuntu:18.04.aarch64.swagger
- ecpe4s/superlu.sce

E4S Facility Deployment
NERSC
OLCF
21.08 Release: 88 Official Products + dependencies (x86_64 gcc)

Support for GPUs

- nvhpc 21.7
- cuda 11.4
- oneAPI 2021.1.1
- ROCm 3.8
21.08 Release: 88 Official Products + dependencies (x86_64 gcc)

- GCC
- DOE fork of LLVM
- AI/ML frameworks
- PyTorch
- Tensorflow
21.08 Release: x86_64 clang (DOE fork of LLVM)
21.08 Release: x86_64 clang (DOE fork of LLVM)
21.08 Release: 88 Official Products + dependencies (ppc64le gcc)

1: adios2
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0-adios2-2.7.1-5ypq4cpy5saarjfhorbxs5neyfbf2ssn
2: aml
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0-aml-8.1.0-7561bdupon02ksfsx6hv6jhn2bgh0vc
3: amrex
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0-amrex-21.88-p165s0bebenymnxoaos44c4kheacd07a
4: arborx
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0-arborx-1.0-zhmhyfs5mcyqphq0d7ooteislnv4puzk
5: archer
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/archer-2.0.8-qwvexfnsqpc3y306off3cygjzr76rul
6: argobots
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/argobots-1.1-rrjx6dgcu3yqhtlnwcweqt46md1f
7: ascent
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/ascent-8.7-1.62vcmgnzr65wshbndj1q24vub7zqms
8: axom
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/axom-0.5.5-mzarurq3pmsdgsb7tu5vmumqncle2yv
9: bolt
   /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/bolt-2.8-eqv7z7lbrlwq87260blt
10: cabana
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/cabana-3.8-3-vsn86rktz8tadgtsfwr07cfobk6y5c4
11: caliper
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/caliper-2.6-0.72lcgyp6s6wnumw50uqexevgwlyk
12: charliecloud
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/charliecloud-0.246-nwmwa3uyg5x3qnt1vzhlwrcwr3k5aw
13: conduit
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/conduit-0.7-2-yprn76qbb56q1xda97ywr6cm6n6u7f5fh
14: darshan-runtime
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/darshan-runtime-3.1.3-5yuvzb26b2nralieje351iw2er1570ddd2
15: datatransferkit
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/datatransferkit-3.1.2-cdcxcm6d4z4agkw5wpq5wstbt73shew7p
16: dyninst
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/dyninst-11.0.1-j6w16s0drvsc671yrend1ldw
17: faodel
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/faodel-1.906.1-07l2z6bepllkbv4jldssx4dnavsnppdne
18: flecsi
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/flecsi-1.4-2-vdjklkqew433x3k2t6aj31jqwshw12m
19: flit
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/flit-2.1.6-744merry2prttg2sl30bttwumrdmn
20: flux-core
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/flux-core-0.28.11-4czniebm666cflkh5ppbph
21: fortilinos
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/fortilinos-2.0-0-ebybxxzsl214ca74wqknbfyfzq7aun
22: gasnet
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/gasnet-2021.3-3dcjcmoerqzsd6erumbompcs5l5x
23: ginkgo
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/ginkgo-1.3.0-ehwp4l7ih4jha7av3d6xwmgvrh
24: globalarrays
    /spack/opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/globalarrays-2.56.globalarrays��りょmey3faspq/spack/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/gotcha-0.8-lx2grawwunh2rcsrgg72qn6jgbjgvd
26: hdf5
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/hdf5-1.12.0-krqagbxub777tдркuxuc6kv6jojpiri
27: heffte
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/heffte-2.1-wnmohwqhyrgkc4pom3lk5wqgqap
28: heprofkit
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/heprofkit-2021.89.0.35-wv466xmgwyr6x3psih123xwcp7ecsp74nd
29: hpx
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/hpx-1.7-1.72-7272quvyqgk1lf4bxrj7mdumyvrxjcb
30: hypre
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/hypre-2.2.0-0-z4v2y4s4yak4vxhpi4kdfzdvkxhup4t3
31: kokkos
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/kokkos-3.4.0-0-dicdwr0rntkq0dbuwkwj2iy5hsvxk
32: kokkos-kernels
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/kokkos-kernels-3.2.0-0etrvi2or2cpuldterp5p63iik41zyt
34: legion
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/legion-21.03-0.1xxkkp6m3r4etdjlwyyiyqytrkmks5p
35: libnrm
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/libnrm-0.1-qfka8n5f6evp9eqxusaojcrwqgcdwo
36: llibco
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/llibco-0.8-cj03q6cilbqoo5
37: lmv-doe
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/lmv-doe-ue-npl07x0dsms6310egevv1ltdzggp04
38: loki
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/loki-0.7-1.75fyegurayey75if123q4yuxsx3nt1bn
39: magma
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/magma-2.6.1-uw4q5sywewvm3a0q4h3xbkayapeow
41: mettall
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/mettall-0.15-16ezxqixr44stap7grn1kynle
42: mfen
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/mfen-4.3-0b67y2s0mir3xwkhieyvktl12b5szs
43: mpich
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/mpich-3.4.2-zotkdluwet516x7f727orx4j7mnoq7m
44: markov-variant
    /spack.opt/spack/linux-ubuntu18.04-ppc64le/gcc-7.5.0/markov-variant-1.4.8-uglb2pr0q643gjrxevalgqf5syjyeh

Support for GPUs
- nvhpc 21.7
- cuda 11.4.0
21.08 Release: 88 Official Products + dependencies (ppc64le llvm)
Using E4S with Shifter on Cori

% shifterimg images | grep e4s
% cd; cp -r /global/homes/s/sameer/demo/testsuite/ ./; ln -s ~/testsuite/trilinos/Zoltan ~/n
% shifter -E --image=ecpe4s/ubuntu18.04-e4s-gpu:21.08 -- /bin/bash --rcfile /etc/bashrc
% spack find; cd ~/Zoltan; ./compile.sh
% which mpicc
% exit
% cat /etc/os-release
% cat run.job
#!/bin/bash
#SBATCH -N 4 -t 5
#SBATCH --image=ecpe4s/ubuntu18.04-e4s-gpu:21.08
#SBATCH -C haswell
srun -n 4 shifter -- /bin/bash -c 'unset CRAYPE_VERSION; unset MODULEPATH; .
/spack/share/spack/setup-env.sh; spack load --first trilinos%gcc ; spack unload mpich; .
/Zoltan'

% sbatch run.job
% cat *.out
Using E4S with Shifter on Perlmutter

% cd; cp -r ~sameer/demo/NPB3.1 .
% shifter -E --image=ecpe4s/ubuntu18.04-e4s-gpu:latest -- /bin/bash --rcfile /etc/bashrc
% spack find
% spack load --first trilinos % gcc
% which mpicc
% cd ~/NPB3.1; make clean; make ; cd bin; mpirun -np 4 ./lu.W.4
Extreme-Scale Scientific Software Stack (E4S) version 21.08

Exascale Computing Project (ECP) Software Technologies (ST) software, Extreme-Scale Scientific Software Stack (E4S) v21.08, includes a subset of ECP ST software products, and demonstrates the target approach for future delivery of the full ECP ST software stack. Also available are a number of ECP ST software products that support a Spack package, but are not yet fully interoperable. As the primary purpose of the v21.08 is demonstrating the ST software stack release approach, not all ECP ST software products were targeted for this release. Software products were targeted primarily based on existing Spack package maturity, location within the scientific software stack, and ECP SDK developer experience with the software. Each release will include additional software products, with the ultimate goal of including all ECP ST software products.

E4S v21.08 Notes.

E4S Container Installation Instructions.
E4S for bare-metal installation

E4S Release 21.08

August 2021 release of E4S

Files

- spack.yml -- Model Spack environment

Specs in the Model Spack Environment are commented out if (a) there are outstanding build issues or (b) if their Spack package does not offer a versioned installation option

Spack Version

E4S 21.08 uses Spack branch e4s-21.08

- https://github.com/spack/spack
- Branch e4s-21.08

Spack Build Cache

- https://cache.e4s.io
- https://cache.e4s.io/21.08

$> spack mirror add E4S https://cache.e4s.io/21.08
$> spack buildcache keys --all
E4S: Spack Build Cache at U. Oregon and AWS

- 50,000+ binaries
- S3 mirror
- No need to build from source code!

https://oaciss.uoregon.edu/e4s/inventory.html
Using E4S: From source using Spack and build caches
E4S Spack environment spack.yaml

- Bare-metal install
  `% cat spack.yaml
  % spack -e . install`

- Docker build:

  ```
  #!/bin/bash -x
  docker build --no-cache -t ecpe4s/ubuntu18.04-e4s-x86_64:1.2 .
  ```
E4S: ppc64le Base Container Images

- Hub.docker.com
- ecpe4s
- Ubuntu 18.04
- RHEL/UBI 7.6
- Centos 7.6
Multi-platform E4S Docker Recipes

FROM ecpe4s/ubuntu18.04-spack-x86_64:0.14.1

WORKDIR /e4s-env

COPY /spack.yaml .

RUN spack install --cache-only \ 
  && spack clean -a && rm -rf /tmp/root/spack-stage

WORKDIR /
E4S: Multi-platform Reproducible Docker Recipes

- x86_64
- ppc64le
- aarch64

https://e4s.io
E4S VirtualBox Image

Container Runtimes
- Docker
- Shifter
- Singularity
- Charliecloud

https://e4s.io
e4s-cl: A tool to simplify the launch of MPI jobs in E4S containers

- E4S containers support replacement of MPI libraries using MPICH ABI compatibility layer.
- Applications binaries built using E4S can be launched with Singularity using MPI library substitution for efficient inter-node communications.
- e4s-cl is a new tool that simplifies the launch and MPI replacement.
- Usage:
  1. e4s-cl init --backend singularity --image ~/ecp.simg --soure ~/source.sh
  2. e4s-cl mpirun -np <N> -hosts <> <command>

https://github.com/E4S-Project/e4s-cl
e4s-cl Container Launcher

Login host

Work hosts

Containers

- e4s-cl launch
- e4s-cl execute
- library resolution
- container launch
- Host libraries
- MPI program
- Host libraries
- MPI program

https://e4s.io
E4S Continuous Integration Testing
E4S Validation Test Suite

- Provides automated build and run tests
- Validate container environments and products
- New LLVM validation test suite for DOE LLVM

• git clone https://github.com/E4S-Project/testsuite.git
Reproducible Container Builds using E4S Base Images

- PMR SDK base image has Spack build cache mirror and GPG key installed.
- Base image has GCC and MPICH configured for MPICH ABI level replacement (with system MPI).
- Customized container build using binaries from E4S Spack build cache for fast deployment.
- No need to rebuild packages from the source code.
- Same recipe for container and native bare-metal builds with Spack!
E4S: GitLab Runner Images

- Dockerhub
- Bare-bones
- Multi-platform
- Build E4S
E4S Builds:
- Ubuntu 18.04
- Ubuntu 20.04
- RHEL 7.6
- RHEL 8
- CentOS 7
- CentOS 8

Architectures: ppc64le and x86_
GitLab GPU Runners on Frank, U. Oregon

- A100 NVIDIA GPU
- DG1 Intel GPU
- MI50 AMD GPU

Frank @ UO: https://oaciss.uoregon.edu/frank
Multi-stage E4S CI Build Pipeline on Cori, NERSC
ORNL GitLab Build Pipeline for E4S Spack Build Cache

- ppc64le (Ascent @ ORNL)
- Reproducible container builds
E4S CI Badges

This repository displays CI badges for E4S products.

<table>
<thead>
<tr>
<th>E4S Product</th>
<th>Latest Release</th>
<th>Release Date</th>
<th>CI Badges</th>
</tr>
</thead>
<tbody>
<tr>
<td>adios</td>
<td><img src="https://github.com/E4S-ECP/adios/actions" alt="GitHub Workflow" /></td>
<td><img src="https://github.com/E4S-ECP/adios/actions" alt="GitHub Workflow" /></td>
<td><img src="https://choosealicense.com/licenses/apache-2.0/" alt="License" /> <img src="https://github.com/E4S-ECP/adios/actions" alt="docs" /> <img src="https://github.com/E4S-ECP/adios/actions" alt="Passing" /> <img src="https://github.com/E4S-ECP/adios/actions" alt="m2e-2.7.0" /> <img src="https://github.com/E4S-ECP/adios/actions" alt="create" /> <img src="https://github.com/E4S-ECP/adios/actions" alt="passing" /> <img src="https://github.com/E4S-ECP/adios/actions" alt="build" /> <img src="https://github.com/E4S-ECP/adios/actions" alt="passing" /></td>
</tr>
<tr>
<td>adios2</td>
<td><img src="https://github.com/E4S-ECP/adios2/actions" alt="GitHub Workflow" /></td>
<td><img src="https://github.com/E4S-ECP/adios2/actions" alt="GitHub Workflow" /></td>
<td><img src="https://github.com/E4S-ECP/adios2/actions" alt="license" /> <img src="https://github.com/E4S-ECP/adios2/actions" alt="docs" /> <img src="https://github.com/E4S-ECP/adios2/actions" alt="passing" /> <img src="https://github.com/E4S-ECP/adios2/actions" alt="m2e-2.7.0" /> <img src="https://github.com/E4S-ECP/adios2/actions" alt="create" /> <img src="https://github.com/E4S-ECP/adios2/actions" alt="passing" /> <img src="https://github.com/E4S-ECP/adios2/actions" alt="build" /> <img src="https://github.com/E4S-ECP/adios2/actions" alt="passing" /></td>
</tr>
<tr>
<td>ani</td>
<td><img src="https://github.com/E4S-ECP/ani/actions" alt="GitHub Workflow" /></td>
<td><img src="https://github.com/E4S-ECP/ani/actions" alt="GitHub Workflow" /></td>
<td><img src="https://github.com/E4S-ECP/ani/actions" alt="license" /> <img src="https://github.com/E4S-ECP/ani/actions" alt="docs" /> <img src="https://github.com/E4S-ECP/ani/actions" alt="passing" /> <img src="https://github.com/E4S-ECP/ani/actions" alt="m2e-2.7.0" /> <img src="https://github.com/E4S-ECP/ani/actions" alt="create" /> <img src="https://github.com/E4S-ECP/ani/actions" alt="passing" /> <img src="https://github.com/E4S-ECP/ani/actions" alt="build" /> <img src="https://github.com/E4S-ECP/ani/actions" alt="passing" /></td>
</tr>
<tr>
<td>amrex</td>
<td><img src="https://github.com/E4S-ECP/amrex/actions" alt="GitHub Workflow" /></td>
<td><img src="https://github.com/E4S-ECP/amrex/actions" alt="GitHub Workflow" /></td>
<td><img src="https://github.com/E4S-ECP/amrex/actions" alt="license" /> <img src="https://github.com/E4S-ECP/amrex/actions" alt="docs" /> <img src="https://github.com/E4S-ECP/amrex/actions" alt="passing" /> <img src="https://github.com/E4S-ECP/amrex/actions" alt="m2e-2.7.0" /> <img src="https://github.com/E4S-ECP/amrex/actions" alt="create" /> <img src="https://github.com/E4S-ECP/amrex/actions" alt="passing" /> <img src="https://github.com/E4S-ECP/amrex/actions" alt="build" /> <img src="https://github.com/E4S-ECP/amrex/actions" alt="passing" /></td>
</tr>
</tbody>
</table>
E4S Community Engagement
Opportunities 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
## 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 <strong>and</strong> Manifest of a la carte selectable software capabilities.</td>
</tr>
<tr>
<td>• A simple packaging of existing software.</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, membership
  – 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 stack for holistic HPC environment

• DOE software as a portfolio is a first-class entity in the ecosystem
  – E4S planning, executing, tracking, assessing is peer collaboration with Facilities, program offices, vendors, etc
  – E4S can become a perennial asset for DOE/ASCR as part of its mission impact within and beyond DOE
Final points

• E4S is a curated software stack with quality improvement incentives, moving toward turnkey use

• With DOE program managers ECP is starting
  – Software ecosystem sustainability planning
  – E4S strategic plan (will include monthly townhalls)

• We believe
  – E4S has reduced important gaps that limit usefulness of DOE software for industry
  – But some gaps remain

• Next steps:
  – Better characterize these gaps
  – Explore models to further reduce and close gaps
  – Plan and execute toward sustainability
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 communities. The work discussed in this presentation represents creative contributions of many people who are passionately working toward next-generation computational science.