Software Deployment at Facilities

E4S at DOE Facilities with Deep Dive at NERSC
Oct 4, 2021

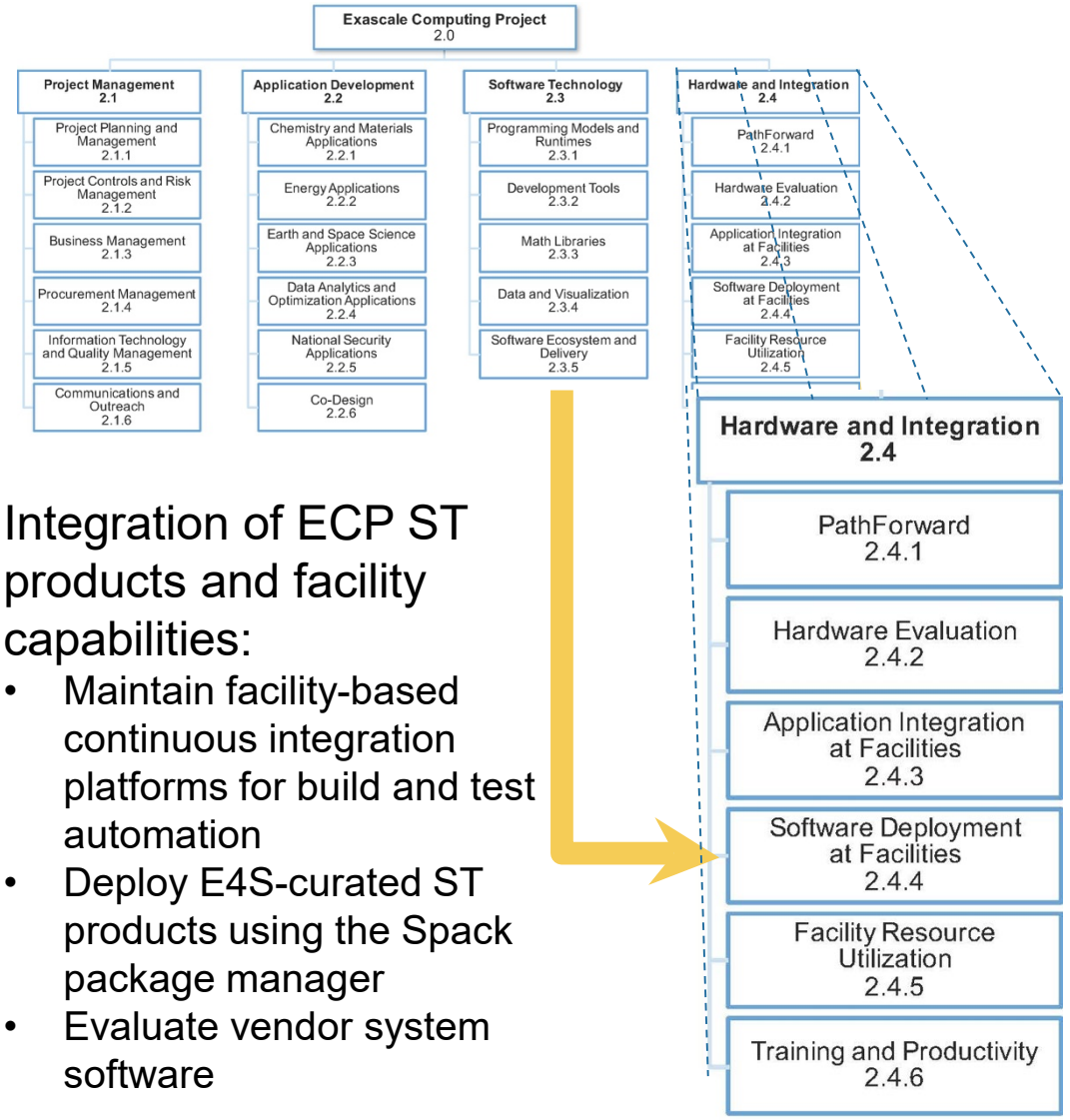
Ryan Adamson
2.4.4 L3 Lead

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

Overview:

ECP Software Deployment at Facilities (SD) Activity

2.4.4 Software Deployment at Facilities - Mission



Integration of ECP ST products and facility capabilities:

- Maintain facility-based continuous integration platforms for build and test automation
- Deploy E4S-curated ST products using the Spack package manager
- Evaluate vendor system software

Each software team and facility has individual (and often unique!) preferences, constraints, and strategies regarding scientific software development and deployment

E4S Extreme Scale Scientific Software Stack

Spack

GitLab / CI

Vendor HW / SW

ALCF

OLCF

NERSC

WBS 2.4.4: Software Deployment at Facilities

Project Short Name	PI Name, Inst	Short Description/Objective	Program Office(s)
2.4.4.01 Software Integration	Shahzeb Siddiqui (LBL)	Build/Test/Deploy ST products at facilities	ASCR
2.4.4.04 Continuous Integration	Paul Bryant (ORNL)	Develop and Deploy ECP CI infrastructure	ASCR
2.4.4.03 Shasta Testing	Jay Srinivasan (LBL)	Evaluate and expedite Shasta releases	ASCR
2.4.4.05 HPCM / Slingshot Testing	Scott Atchley (ORNL)	Explore HPCM and Slingshot as alternatives to Shasta	ASCR


Shahzeb Siddiqui


Paul Bryant

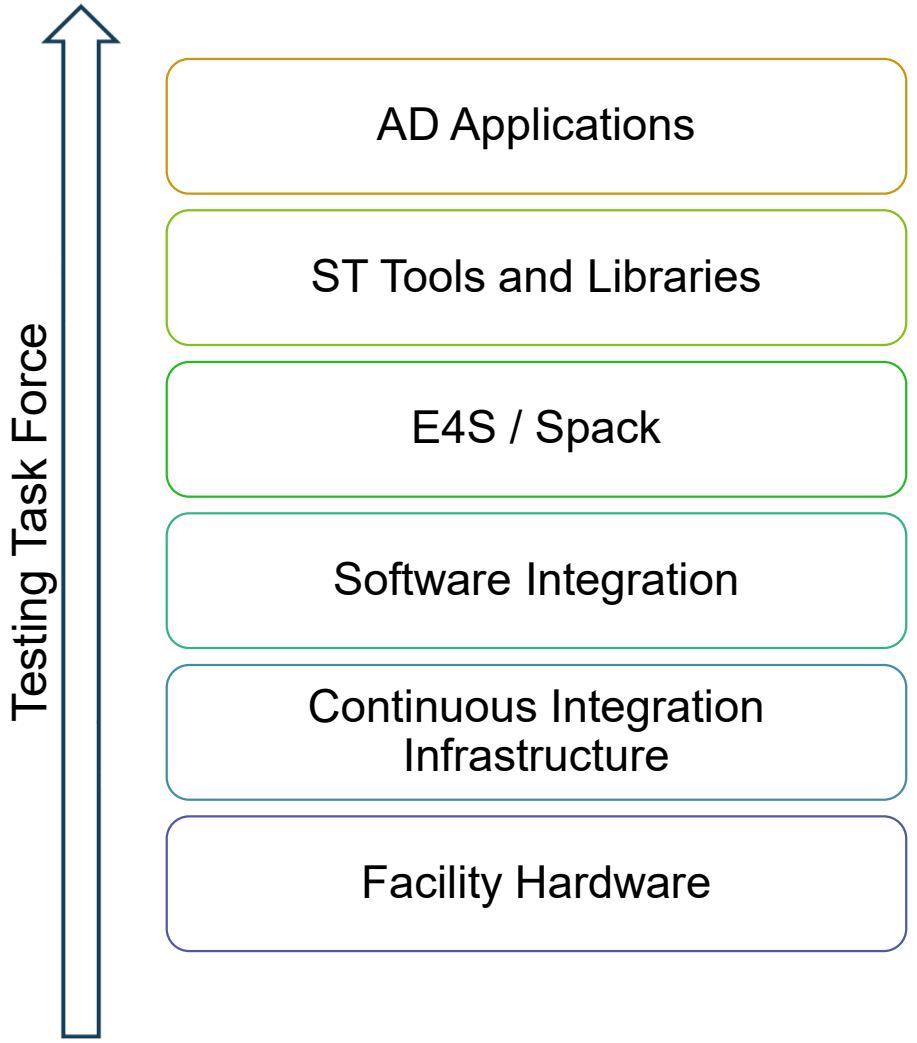

Jay Srinivasan


Scott Atchley

Integrated testing strengthens the ECP software ecosystem

Software Deployment Efforts

- Software Integration interfaces with the Testing Task Force, E4S, and ST to adapt builds, tests, and deployments to facility systems
- Continuous Integration Infrastructure supports the automation of activities on facility systems
- Slingshot and System Software testing provides insight into facility hardware from a system functionality perspective
- The Testing Task Force is the vertical integration of all layers of the ECP software ecosystem



Software Integration:

Integrating the ECP Software Ecosystem with Facility Systems

Software Integration Team

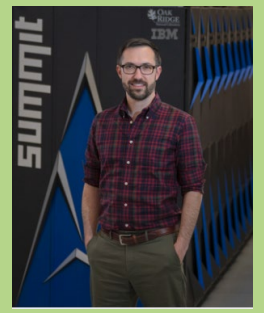
Frank Willmore

Aditya Kavalur

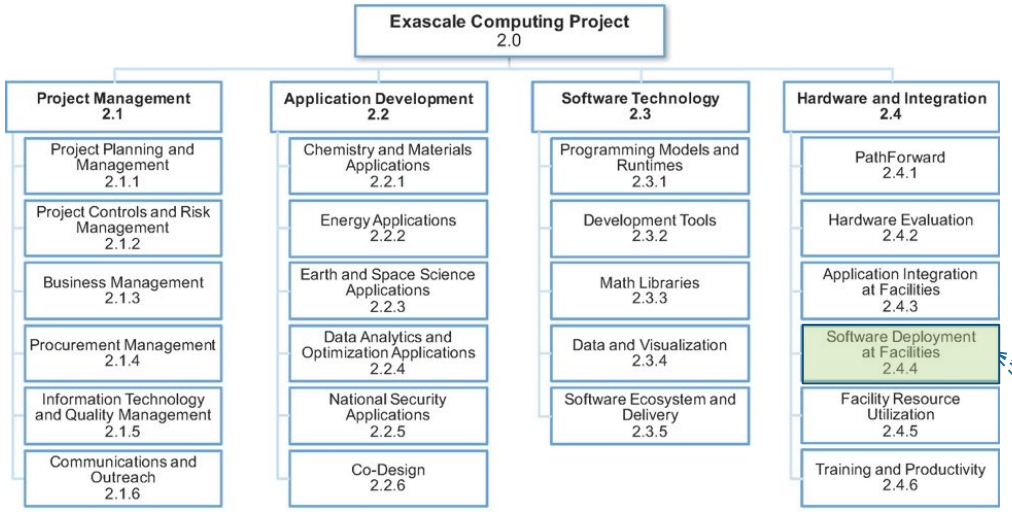
Shahzeb Siddiqui

Matt Belhorn

Jamie Finney



Ryan Adamson



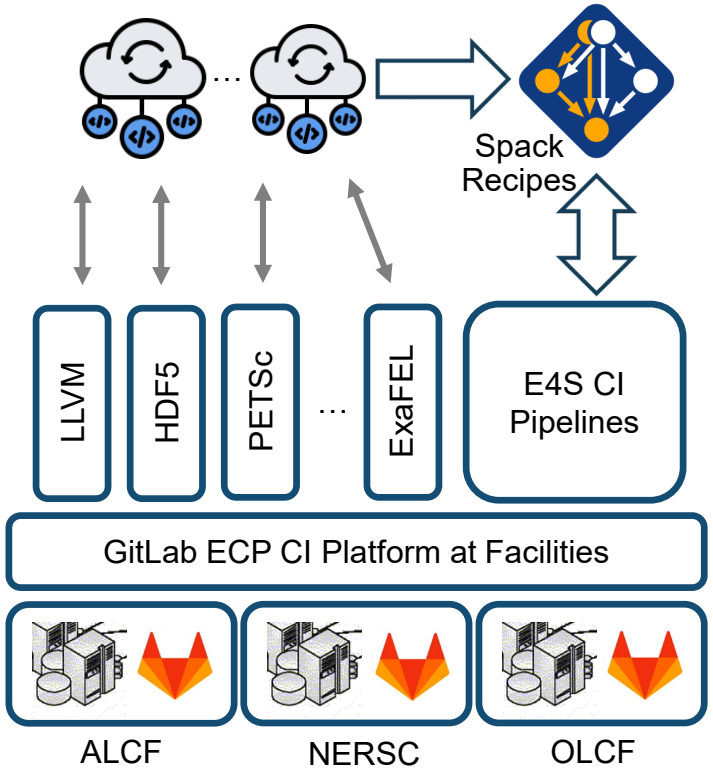
- 2.4.4.01 Software Integration
- 2.4.4.04 Continuous Integration
- 2.4.4.03 Shasta Testing
- 2.4.4.05 HPCM/Slingshot Testing

ECP Software Ecosystem Overview

AD and ST teams implement ECP CI regression testing as appropriate into existing CI frameworks and merge new build/test recipes into spack/develop

E4S Team Prepares Quarterly Releases for Facility Installation, freezing on a point-in-time commit of spack/develop

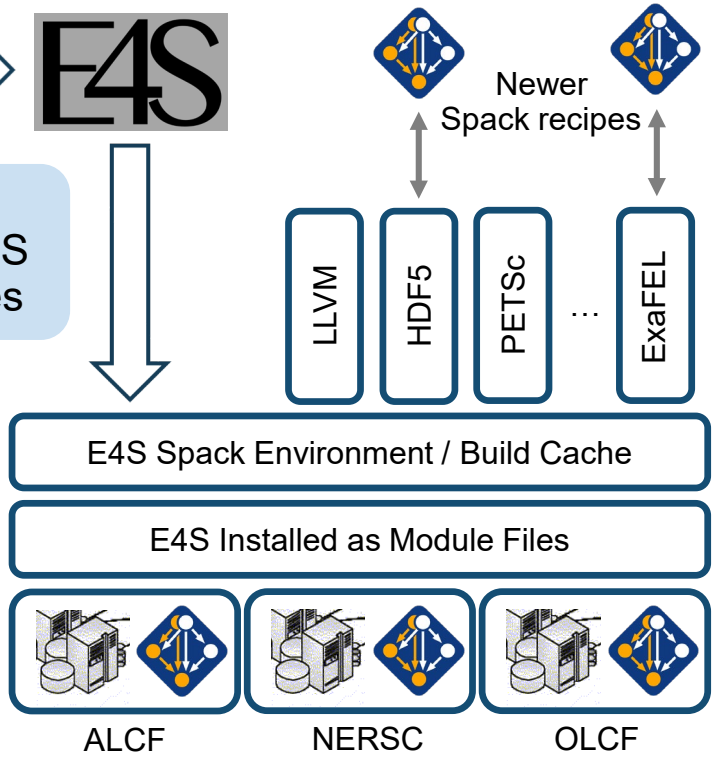
AD teams access stable ST software through Facility-maintained modules. Build cache and config available to AD/ST teams for integrating hotfixes



Continuous Integration (2.4.4.04)

SI Team Deploys tested, versioned E4S on Facility Resources

E4S Team implements CI build and smoke testing for spack/develop



Software Integration (2.4.4.01)

Roles and Responsibilities within the ECP Ecosystem

ST Developers

- AD and ST teams implement ECP CI regression testing as appropriate into existing CI frameworks and merge new build/test recipes into spack/develop
  - https://e4s-project.github.io/policies.html
- These software development teams typically have other automated CI pipelines, developer-driven unit testing, and other software assurance best practices as determined by their internal policies
  - https://ideas-productivity.org/ideas-ecp/

E4S Software Curators

- Spack Team implements CI build and smoke testing for spack/develop
  - https://github.com/spack/spack
  - https://cdash.spack.io
- E4S Team Prepares Quarterly Releases for Facility Installation, freezing on a point-in-time commit of spack/develop
  - https://github.com/E4S-Project/e4s/tree/master/environments/21.05

SI / Facilities

- SI Team Deploys packages selected from a tested and versioned E4S release onto Facility Resources.
  - https://docs.nersc.gov/applications/e4s/cori/21.05/
- Facility software and operations teams have additional verification and validation tests that are performed according to facility software and operations policies
  - https://www.exascaleproject.org/event/buildtest-21-09/

Five types and six layers of testing for the ECP Software Ecosystem

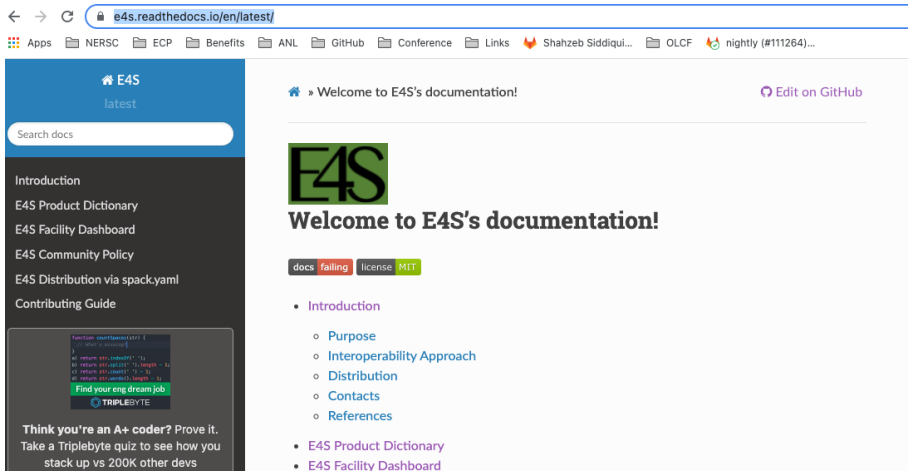
Test type	Complexity	Notes
Build	Low	Build tests check that all software components have been built successfully by using the appropriate compilers and tunables.
Validation	Low	Validation tests are run to demonstrate that a piece of software is installed correctly and to clearly demonstrate usage of the appropriate unique hardware features of the system, such as accelerators or high-speed interconnects.
Integration	Medium	Integration tests are implemented to ensure that versions of libraries along with compile time options do not conflict adversely with other related libraries as part of a larger software ecosystem.
Regression	High	Regression tests are maintained and run as frequently as developers create new software changes to determine whether the changes impact any already working features of the software.
Performance	High	Performance tests are implemented to detect regressions in runtime or degradation in time to solution or network bandwidth.

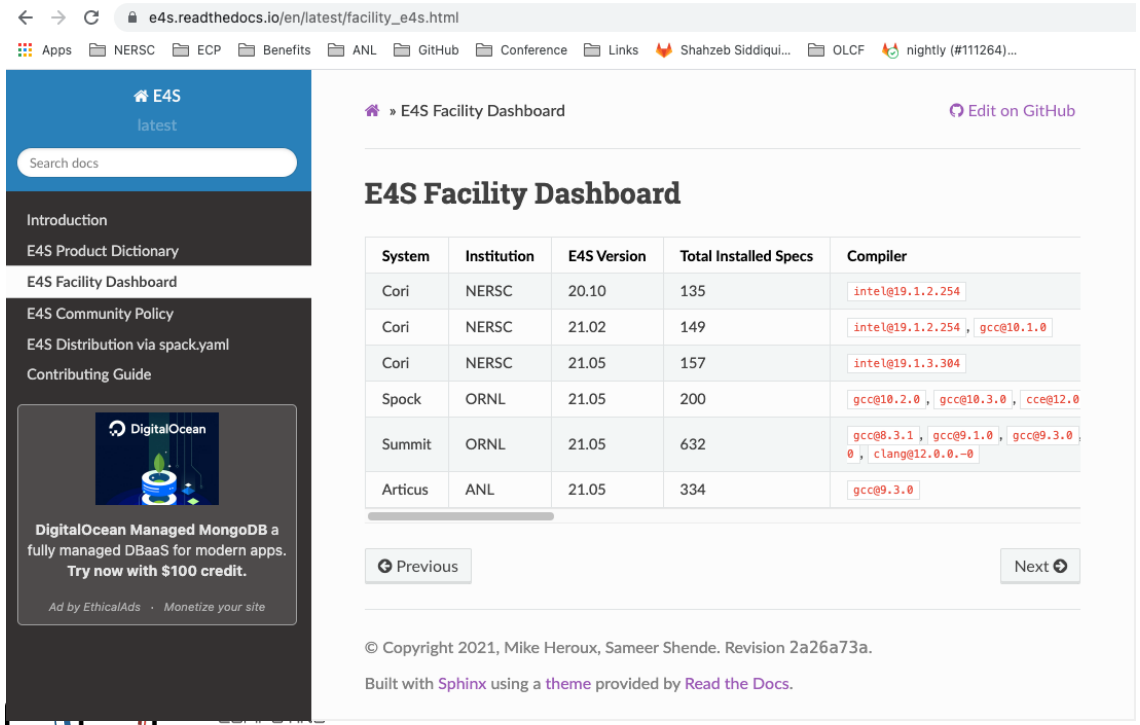
Teams and Testing Assurance

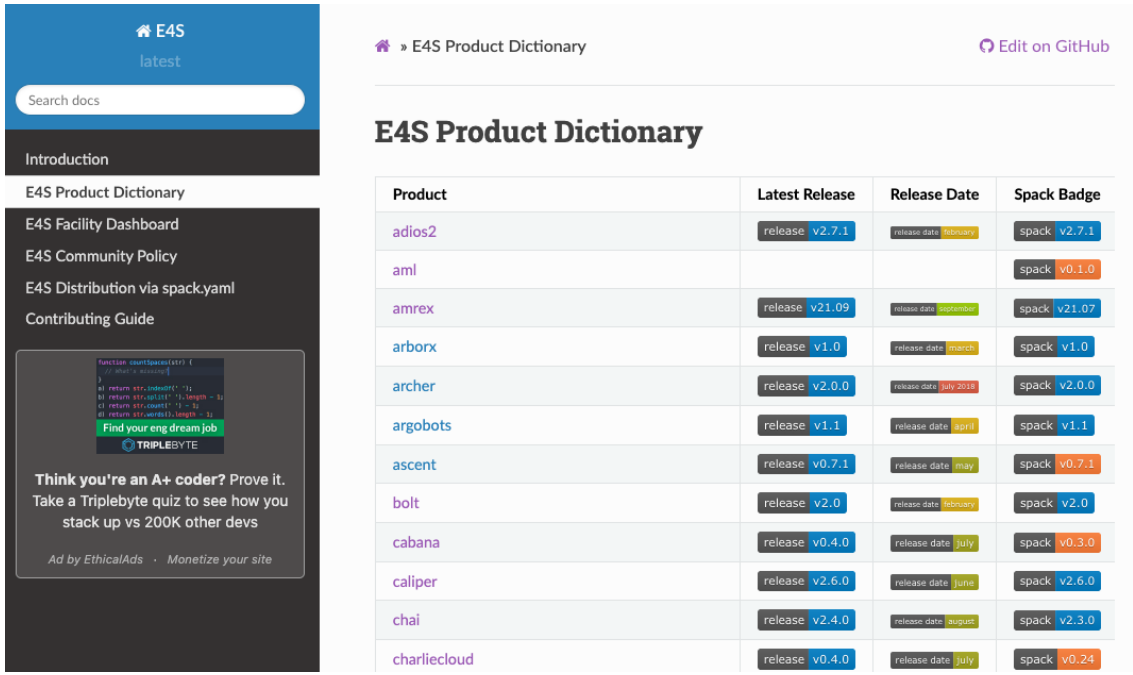
- ST teams implement **all test types** on developer, cloud, or product specific build and test CI pipelines
- ST teams run selected, high-value tests of **all test types** on facility CI platforms
- Spack CI pipelines test **builds** of packages in the cloud and at some facilities
- E4S team performs **build, validation, and integration** tests on facility systems and elsewhere
- SI teams run **build, validation, and integration** tests on facility systems
- Facility users run **validation** tests such as ‘spack test’ and the E4S test suite

E4S Documentation is Live!

- E4S User documentation is now available at https://e4s.readthedocs.io/en/latest/
- The E4S Product Dictionary is now available with list of E4S products with CI badges, reference to Spack package and assigned maintainers
- E4S Facility Dashboard outlines E4S deployments at the facility and anyone can contribute to this page provided they have a deployment of E4S
- E4S Product Documentation: https://e4s-project.github.io/DocPortal.html

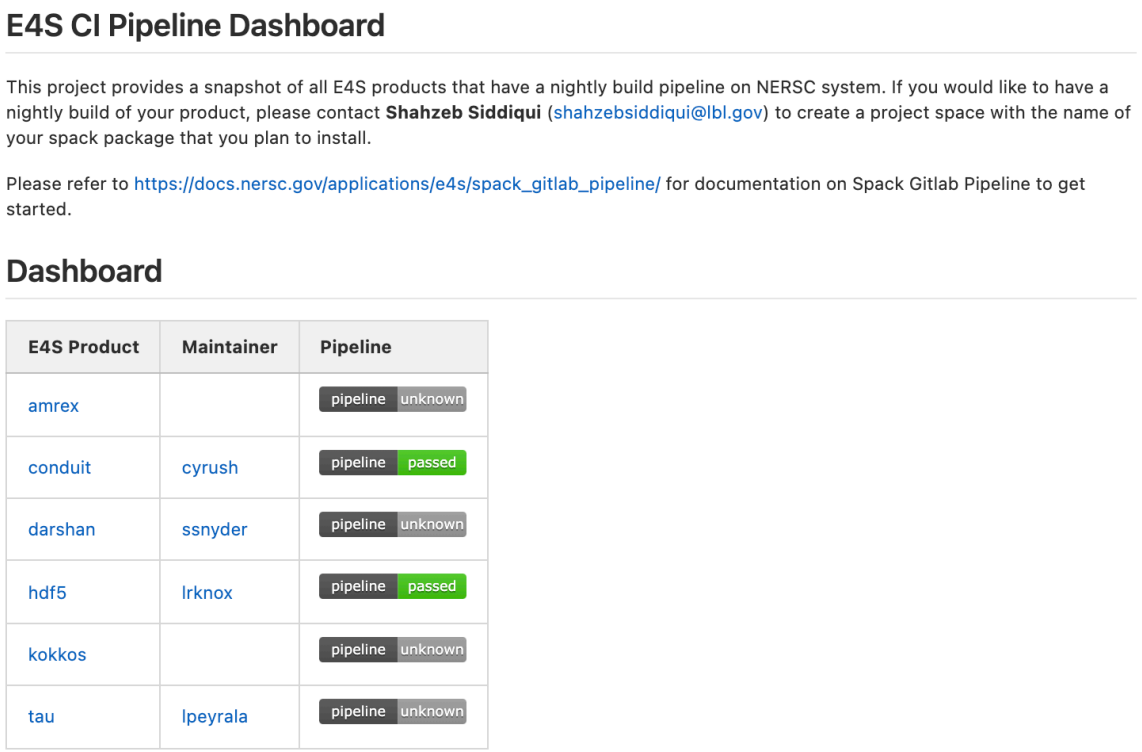






E4S Nightly Build Pipeline

- SI maintains a CI dashboard for E4S nightly builds which reports the project, maintainers, and CI badges. This can be found at https://software.nersc.gov/ecp/e4s-ci-pipelines/dashboard.
- All projects are located at https://software.nersc.gov/ecp/e4s-ci-pipelines/ with a project name per spack package



E4S CI Pipeline Dashboard

This project provides a snapshot of all E4S products that have a nightly build pipeline on NERSC system. If you would like to have a nightly build of your product, please contact **Shahzeb Siddiqui** (shahzebsiddiqui@lbl.gov) to create a project space with the name of your spack package that you plan to install.

Please refer to https://docs.nersc.gov/applications/e4s/spack_gitlab_pipeline/ for documentation on Spack Gitlab Pipeline to get started.

Dashboard

E4S Product	Maintainer	Pipeline
amrex		pipeline unknown
conduit	cyrush	pipeline passed
darshan	ssnyder	pipeline unknown
hdf5	lrknox	pipeline passed
kokkos		pipeline unknown
tau	lpeyrala	pipeline unknown

E4S/21.05 at Cori

- There is a **e4s/21.05** modulefile on Cori with up to 157 installed specs built with intel@19.1.3.304 compiler.
- Spack configuration is published upstream at https://github.com/spack/spack-configs/tree/main/NERSC/cori/e4s-21.05
- Deployment issues are tracked in an ECP software deployment specific repository, for example: https://gitlab.com/ecp-swd/issuetracking/-/issues/35

1.2. **E4S Facility Deployment Dashboard**

Cori

E4S Version	Gitlab Project	Installed Specs	Compiler	Spack commit	Spack.yaml	Installed Specs
20.10	https://software.nersc.gov/NERSC/e4s-2010	135	intel@19.1.2.254	e1e0bbb4cbe11a3f0d7e50466ffa86071ee653b7	https://github.com/spack/spack-configs/blob/master/NERSC/cori/e4s-20.10/spack.yaml	https://github.com/spack/spack-configs/blob/master/NERSC/cori/e4s-20.10/e4s-20.10.txt
21.02	https://software.nersc.gov/NERSC/e4s-2102	149	intel/19.1.2.254 and gcc@10.1.0	b56d65fce5f4743a23399f0cde006bed1b52d53d	https://github.com/spack/spack-configs/blob/main/NERSC/cori/e4s-21.02/spack.yaml	https://github.com/spack/spack-configs/blob/main/NERSC/cori/e4s-21.02/e4s-21.02.txt
21.05	N/A	157	intel@19.1.3.304	https://github.com/spack/spack/tree/e4s-21.05	https://github.com/spack/spack-configs/blob/main/NERSC/cori/e4s-21.05/spack.yaml	https://github.com/spack/spack-configs/blob/main/NERSC/cori/e4s-21.05/e4s-21.05.txt

E4S/21.05 at Cori

```
siddiq90@cori09> spack find -x
==> In environment e4s
==> Root specs
adios@1.13.1                  faodel@1.1906.1              libquo@1.3.1            precice@2.2.1                    superlu@5.2.1
adios2@2.7.1                  flecsi@1.4 +cinch            libunwind@1.5.0         pumi@2.2.5                       superlu-dist@6.4.0
aml@0.1.0                     flit@2.1.0                   loki@0.1.7              py-libensemble@0.7.2             swig@4.0.2
amrex@21.05                   gasnet@2021.3.0              mercury@2.0.1           py-petsc4py@3.15.0               swig@4.0.2-fortran
arborx@1.0                    ginkgo@1.3.0                 metall@0.13             py-warpx@21.05                   sz@2.1.11.1
argobots@1.1                  globalarrays@5.8             mfem@4.2.0              py-warpx@21.05                   tasmanian@7.5
ascent@0.7.1 ~fortran         gmp@6.2.1                    mpark-variant@1.4.0     py-warpx@21.05                   tau@2.30.1
bolt@2.0                      gotcha@1.0.3                 ninja@1.10.2            qthreads@1.16  scheduler=distrib turbine@1.3.0
cabana@0.3.0                  hdf5@1.10.7                  openpmd-api@0.13.4      raja@0.13.0                      umap@2.1.0
caliper@2.5.0                 hypre@2.20.0                 papi@6.0.0.1            scr@3.0rc1                       umpire@4.1.2
chai@2.3.0 ~benchmarks~tests  kokkos@3.4.00 +openmp        papyrus@1.0.1           slepc@3.15.0                     upcxx@2021.3.0
conduit@0.7.2                 kokkos-kernels@3.2.00 +openmp parallel-netcdf@1.12.2 stc@0.9.0                       zfp@0.5.5
darshan-runtime@3.3.0         legion@21.03.0               pdt@3.25.1              strumpack@5.1.1 ~slate
darshan-util@3.3.0            libnrm@0.1.0                 petsc@3.15.0            sundials@5.7.0

==> 58 installed packages
-- cray-cnl7-haswell / intel@19.1.3.304 --------------------------
adios@1.13.1  caliper@2.5.0          globalarrays@5.8      mercury@2.0.1          petsc@3.15.0          raja@0.13.0          tasmanian@7.5
adios2@2.7.1  chai@2.3.0             gmp@6.2.1             metall@0.13            precice@2.2.1         scr@3.0rc1           tau@2.30.1
aml@0.1.0     darshan-runtime@3.3.0  gotcha@1.0.3          mfem@4.2.0             pumi@2.2.5            slepc@3.15.0         umap@2.1.0
amrex@21.05   darshan-util@3.3.0     hypre@2.20.0          mpark-variant@1.4.0    py-libensemble@0.7.2  stc@0.9.0            upcxx@2021.3.0
arborx@1.0    faodel@1.1906.1        kokkos-kernels@3.2.00 ninja@1.10.2           py-petsc4py@3.15.0    strumpack@5.1.1
argobots@1.1  flecsi@1.4             legion@21.03.0        openpmd-api@0.13.4     py-warpx@21.05        sundials@5.7.0
ascent@0.7.1  flit@2.1.0             libnrm@0.1.0          papyrus@1.0.1          py-warpx@21.05        superlu@5.2.1
bolt@2.0      gasnet@2021.3.0        libquo@1.3.1          parallel-netcdf@1.12.2 py-warpx@21.05        swig@4.0.2
cabana@0.3.0  ginkgo@1.3.0           loki@0.1.7            pdt@3.25.1             qthreads@1.16         swig@4.0.2-fortran
```

Installed packages of E4S 21.05 are discoverable with Spack on Cori

Continuous Integration:

Automating builds and tests of
the ECP Software Ecosystem

There are three main use cases for CI on facility systems

1) Software Development Testing

- Regression Testing
  - Software failures are detected when new code is introduced. This prevents latent bugs from existing well after a feature has been implemented.
  - Correctness of results is assessed by examining changes in the output of a well-understood and tested problem.
- Performance Testing
  - Performance regressions are caught in the exact same environment where performance is important.

2) Ecosystem Integration

- Extreme Scale Scientific Software Stack (E4S) integration tests
  - Regular builds of spack versions and release candidates of E4S ensure that facility software integrators will not experience issues during installation.
  - Recurring tests of E4S team installed software stacks detect underlying issues that change over time.
- Individual software 'build' recipes
  - Individual products can be built periodically on facility systems and failures addressed by developers or integrators as appropriate.

3) Facility Operations Assurance

- Regular tests of installed facility software stack
  - Environments drift over time. CI can catch issues related to hardware, vendor PE, or other Facility service updates.
- Regular tests of user managed container images
  - Security assessments can be automated with CI pipelines to inspect container images, run static analysis tools, and pass security unit tests.

There is a Value Per Cycle tradeoff

Runner/Server Capability – Deployment Overview

Resource	CI Status	GitLab Ver.	Jacamar CI Ver.	License
OLCF - Ascent	Functional and available to ECP	14.1+	0.8.0	Starter
OLCF - Spock	CI deployed and planning ECP access	14.1+	0.8.0	Premium
ALCF - JLSE	Functional and available to ECP	14.1+	0.8.2	Premium
ALCF – Theta	Functional and available to ECP	14.1+	0.8.2	Starter
NERSC - Cori	Functional and available to ECP	14.1+	N/A	Core

A typical month of CI at facilities (Apr 2021 – May 2021)

Apr 2021 – May 2021	ALCF	NERSC	OLCF
Active Projects	23	86	36
Average Job Times	1-120 minutes	1-120 minutes	1-120 minutes
Total Jobs Executed	2773	3584	3837
Total Hours Spent	1553	46000	~1500 (estimated)
Tickets Per Week	~3	~2	~2
Special Queue Policies	No	Yes	No
Resource Adequacy	Yes	Yes	Yes

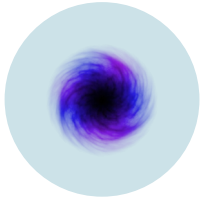
- Jobs are generally short, and a couple of projects have longer running CI pipelines
- Queue policies are sufficient for starting out, but tuning can make CI 'easier' for certain workloads

- Ticket load is reasonable, and typically about 10 or so a month
- Resources are sufficient for the workloads being performed at the moment
- Software Development CI users typically self regulate and find cycles elsewhere where they can

E4S: Better Quality, Documentation, Test, Integration, Delivery, Build & 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 nightmares

Quarterly releases
Release 21.08 – August

Build caches
10X build time improvement

Turnkey stack
A new user experience

https://e4s.io

E4S Strategy Group
US agencies, industry, international

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.

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?