ECP Continuous Integration and Software Deployment

EXASCALE COMPUTING PROJECT

WBS 2.4.4.01 Software Integration

CS workshop

National Nuclear Security Administration

LA-UR-19-26342

U.S. DEPARTMENT OF ENERGY | Office of Science

ECP Project: Software Deployment at Facilities Portfolio Context
2.4.4.01 – Software Integration

The Software Integration effort was established to bridge the ECP ST software development effort with the Exascale hardware and software environments deployed at the Facilities.

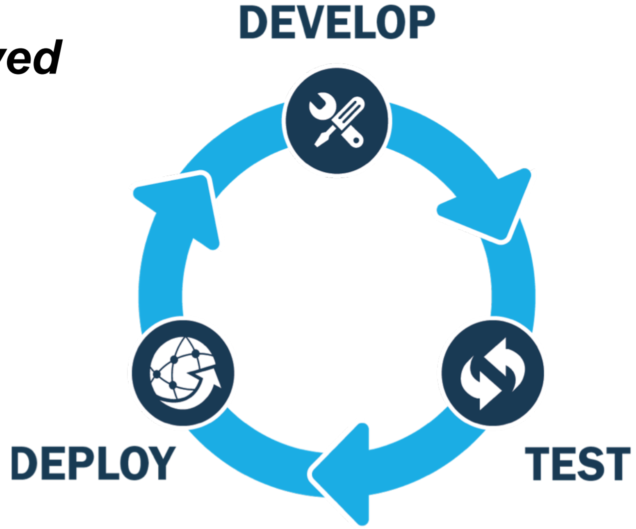
- **Continuous Integration (CI)** - Provide the ability to continuously test AD/ST software on facility hardware resources with software environments established at the Facility.

Key for software development teams targeting systems being deployed agile feedback loop is key for development

- **Software Deployment (SD)** - Establish integrated software packaging, testing, and deployment options that increase the compatibility and quality allowing ease of software deployment.

Integrated deployment of software packages

considering dependencies and capability packaging



LA-UR-19-26342

Security poses challenges for automation at large, multi-user HPC centers.

1. **Difficult to run persistent services (like CI systems)**
   - HPC workloads are mostly batch jobs; have a fixed time limit
   - Persistent services are difficult to deploy due to data security requirements
   - Batch jobs typically have a fixed time limit, but HPC centers built around batch.
2. **CI-like automation requires running arbitrary code**
   - Often in response to *external* repository check-ins
   - How do we know who ran the code?
   - How do we trust users, and who do we blame if it the code is malicious?
   - 2-factor authentication prevents automated ingress from outside
3. **All tasks at most HPC centers need to run *as* some user**
   - Can't allow different users' jobs to share data.
   - Need isolation between jobs run by user A and jobs run by user B
   - Can't have unauthenticated services listening on arbitrary ports

Continuous Integration (CI)

CI Infrastructure /Implementation

Collaboration with ECP ST software ecosystem project to define and implement

GitLab Enhancements:

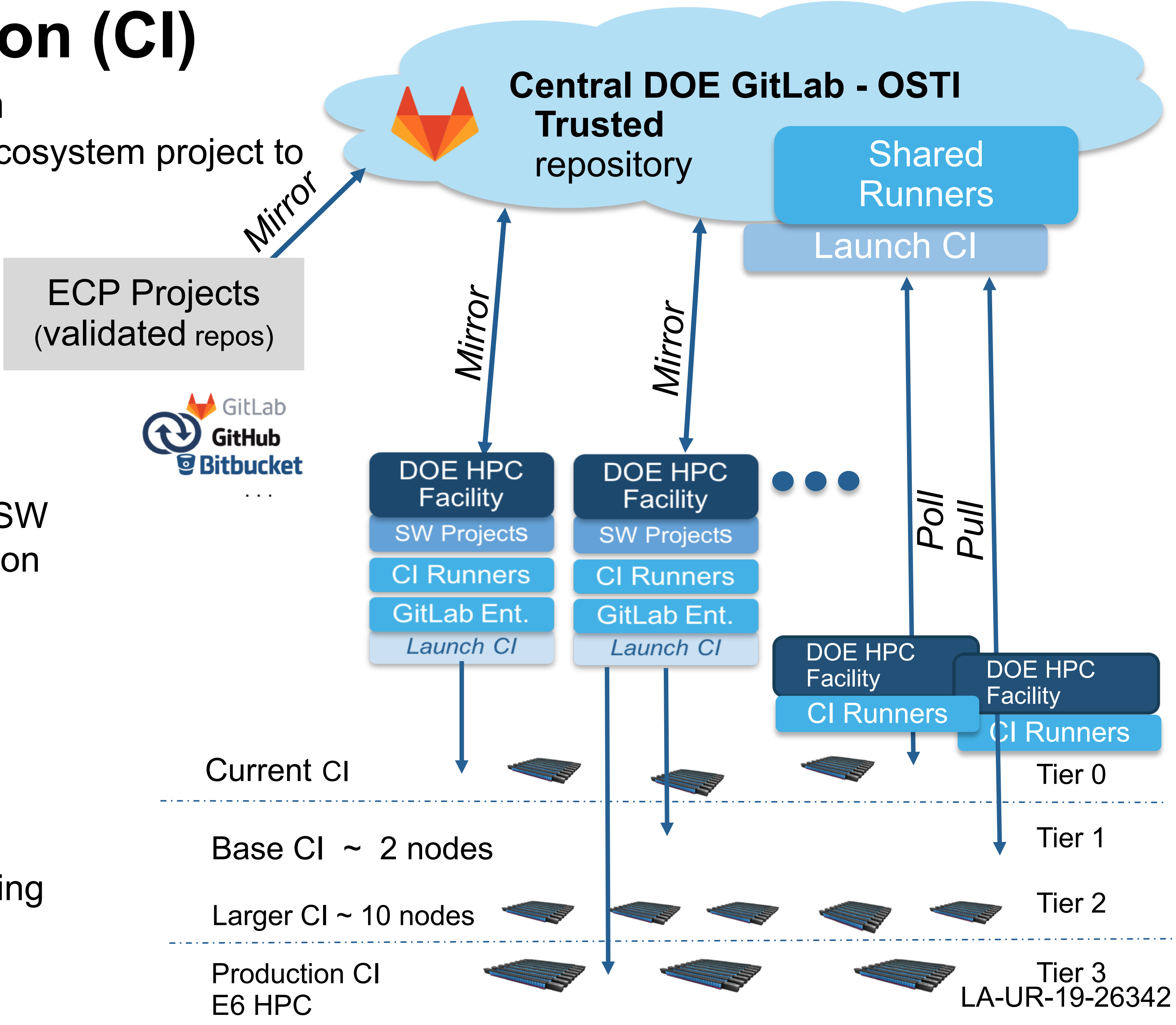
- Setuid and batch submission
- Internal and cross site account mapping

OSTI:

- Provide central GitLab for CI and SW deployment with cross site validation
- Support account authentication

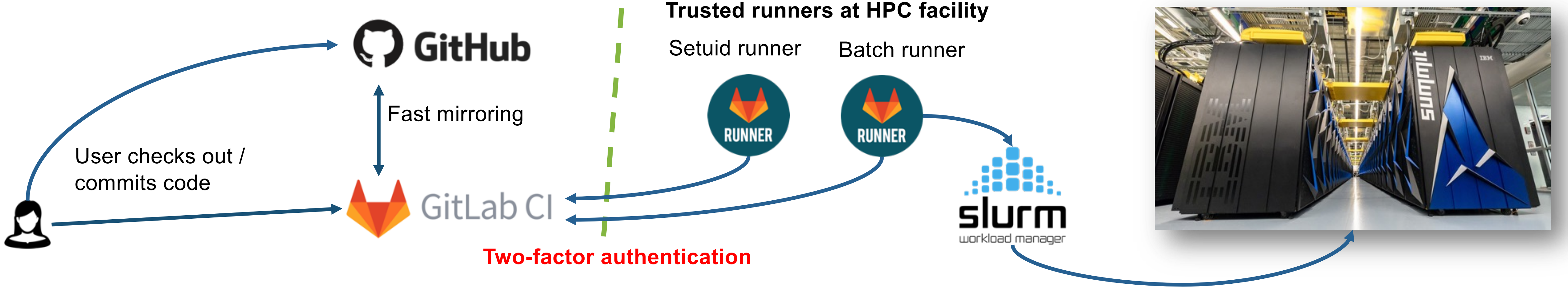
Facilities:

- Improved internal site GitLab capabilities
- Support account authentication
- Engaging with ECP AD and ST project teams to support on-boarding



LA-UR-19-26342

Through ECP, we are working with Onyx Point to deliver continuous integration for HPC centers



Trusted runners at HPC facility

Setuid runner

Batch runner

Fast mirroring

User checks out / commits code

Two-factor authentication

- CI at HPC centers is notoriously difficult
  - Security concerns prevent most CI tools from being run by staff or by users
  - HPC centers really need to deploy trusted CI services for this to work
- We are developing a secure CI system for HPC centers:
  - Setuid runners (run CI jobs as users); Batch integration (similar, but parallel jobs); multi-center runner support
- Onyx Point will upstream this support into GitLab CI
  - Initial rollout in FY19 at ECP labs: ANL, ORNL, NERSC, LLNL, LANL, SNL
  - Upstream GitLab features can be used by anyone!

CI FY19 Implementation Strategy

FY19 - capability development year..

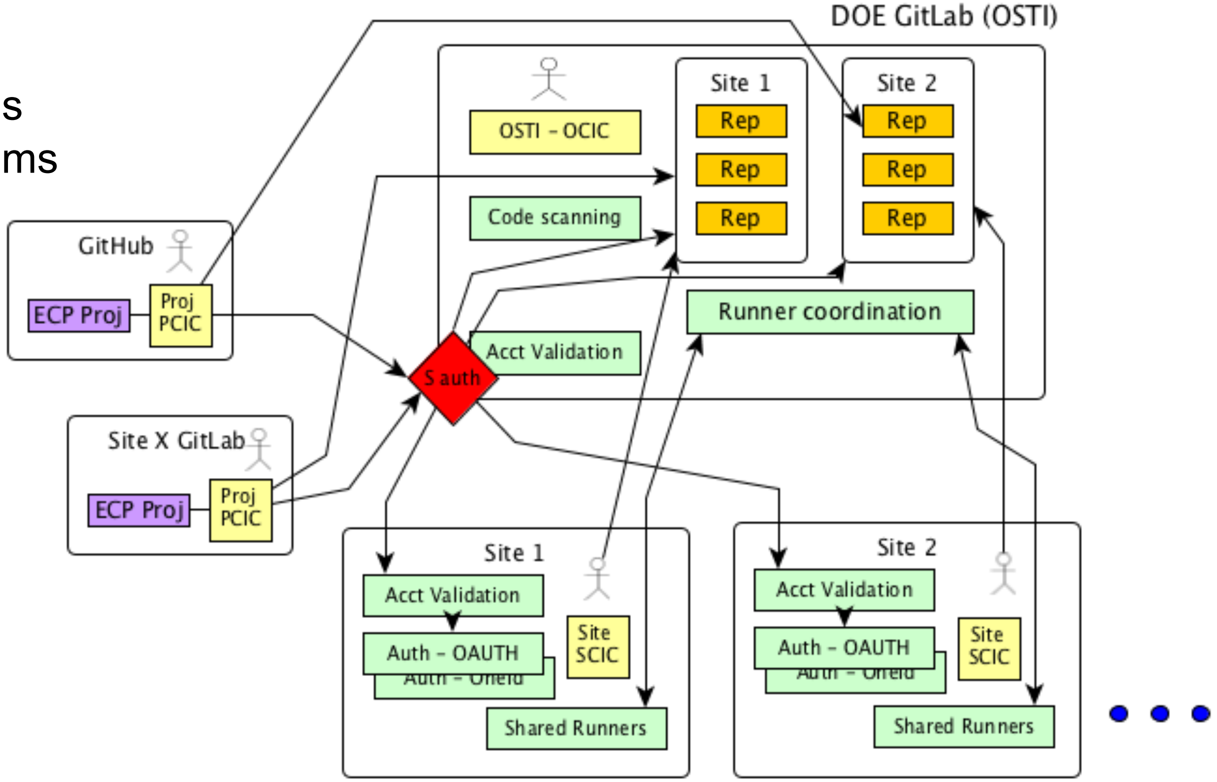
Central DOE GitLab – OSTI. In-Place

- Installed and support GitLab Instance (premium license)
- Process to establish ECP user accounts (through site authentication)
- Process to establish project repositories for mirroring (through site groups)
- Process to register site federated runners (for machines at sites)

Site Integration – In-Process

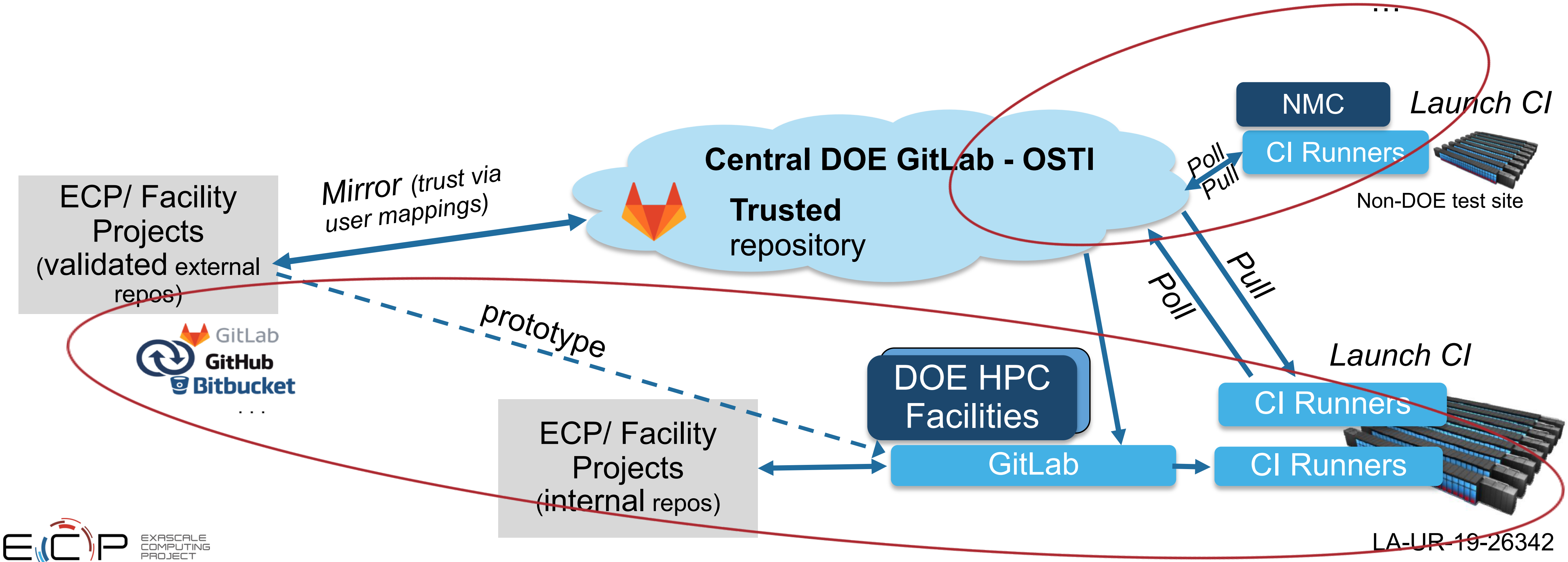
- Internal CI (based on internal GitLab instance)
  - Projects being integrated to test/production systems
  - Security review for runner integration to HPC systems
- Establish Authentication endpoints
  - NERSC (Shibboleth) – In-process
  - NNSA labs (OneID) – July
  - OLCF (Oauth) – August
  - ALCF (Oauth) – August
- Process for project integration and establishment of OSTI repository
  - User account validation with sites

ECP CI Workflow



Planned model for ECP CI

- Projects can keep their projects hosted elsewhere, and use mirroring to DOE GitLab repo (future)
- Facilities can also use the same model with an internal GitLab instance (current)
  - Can also mirror projects from DOE GitLab for internal CI testing capability
- Facilities have runners (HPC resources) polling for changes in a trusted location (GitLab)
- CI jobs launched via batch (exact policy TBD), or possibly on login nodes as-user



EXASCALE COMPUTING PROJECT

LA-UR-19-26342

CI Testing Tiers – HPC Resources

CI Cross-Site Targeting

CI Cross-Site Facilitating

Testing Tier	Description	Notes
Tier 0	• What AD/ST projects do now • Existing CI • Regression tests (no CI)	• May include GitHub/ Travis - internet • cron job based regression on misc. hardware
Tier 1	• Base ECP CI - Build and Run resources • Possible 2 build and 2 run nodes • Build and Smoke tests • Run multiple builds on resource • unit / Integration tests • Cross-site CI target	• What is ratio of build to test resources? • Work with AD and ST teams to support their needs • Possible to allocate from other HPC resources with separate scheduling policy
Tier 2	• Facility test resource (~10 + nodes) • In security enclave – site dependent • Larger scale tests • Facility approval for projects	• Facility managed and may want to approve projects • Possible production security constraints
Tier 3	• Production machines • Need allocation • Production job rules • Scale tests	• Facility managed and may want to approve projects • Production security constraints

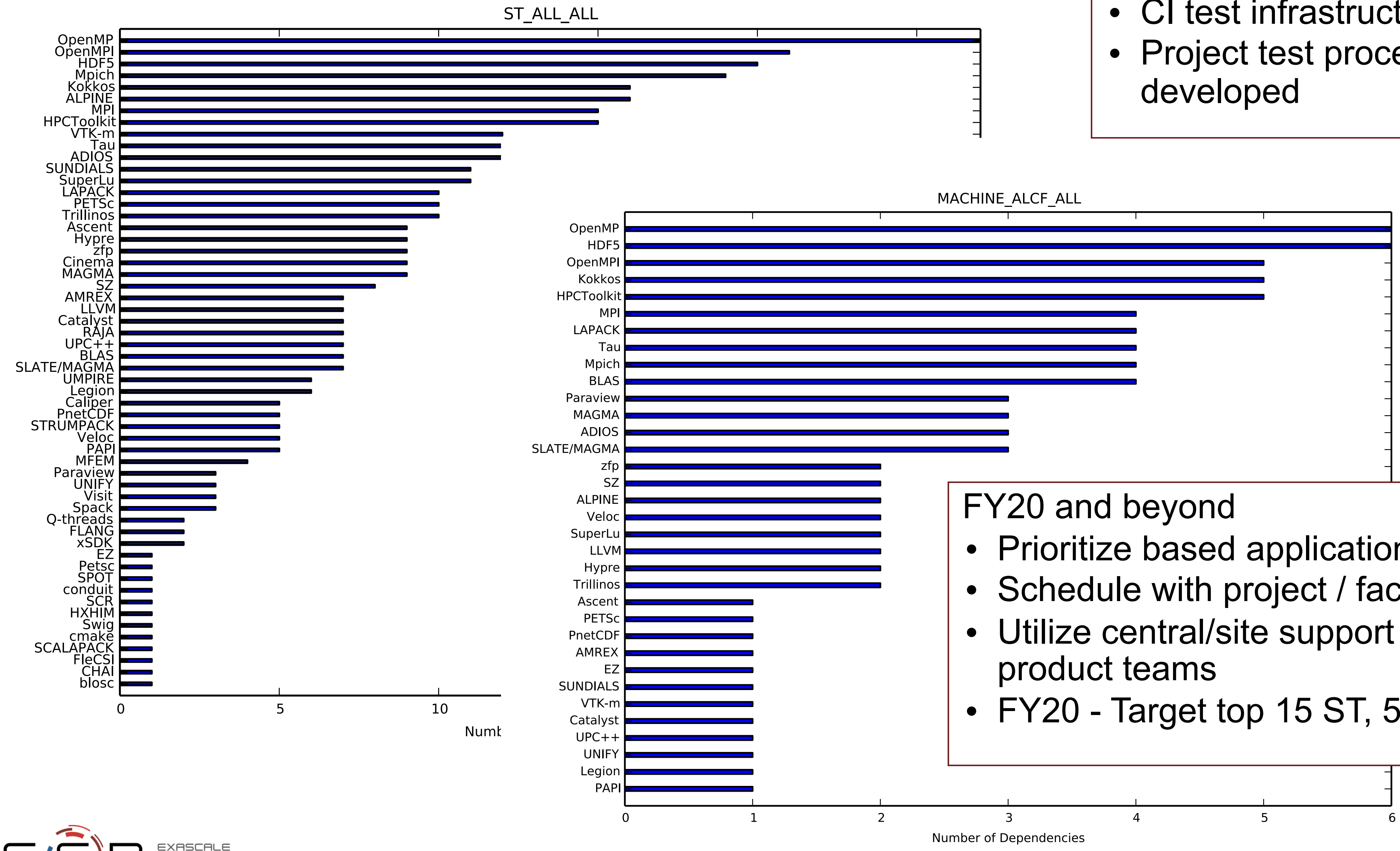
State of Site Internal CI Deployments and Integration at DOE HPC Facilities

Facility	CI Resources	Teams/Products Integrating into CI
OLCF	Internal Open science Gitlab Runners - HPC cluster Ascent (summit like)	ECP-Copa/cabana, ECP-Proxy, FleCSI
ALCF	Internal GitLab server VM Runners – HPC systems Theta, Iota	Argo-AML, Datalib/Darshan, ECP-Proxy, PETSc, NWChem, LLVM
NERSC	Internal GitLab server VM Runners - HPC systems Edison/Cori	HDF5, ECP-Proxy
LLNL	Internal Gitlab server (accessible to LLNL HPC users) Runners – Quartz, butte (P9 cluster)	Spack, RAJA, CHAI, Umpire, SAMRAI, RADIUSS, UnifyCR, VeloC, Ascent
LANL	Internal GitLab CCS Runners - HPC test system Darwin	FleCSI/Legion, EAP, ECP-Proxy, Adios

State of Cross-Site CI Integration and testing across DOE HPC Facilities
- Prototype mode – working through workflow

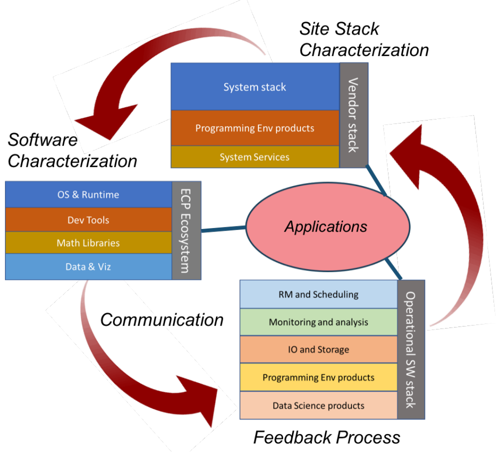
Facility	CI Resources	Teams/Products Integrating into CI
NMC	DOE GitLab - OSTI Runners – HPC CI test system – P9	FleCSI HDF5, ECP-Proxy, Dyninst, Adios
NERSC	DOE GitLab - OSTI Runners – HPC systems Edison/Cori	HDF5, ECP-Proxy, Dyninst, Adios

Implementation Plan for ST – 2020+



FY19

- CI test infrastructure in place
- Project test process and support developed



FY20 and beyond

- Prioritize based application/facility targeted
- Schedule with project / facility
- Utilize central/site support to work with product teams
- FY20 - Target top 15 ST, 5 AT

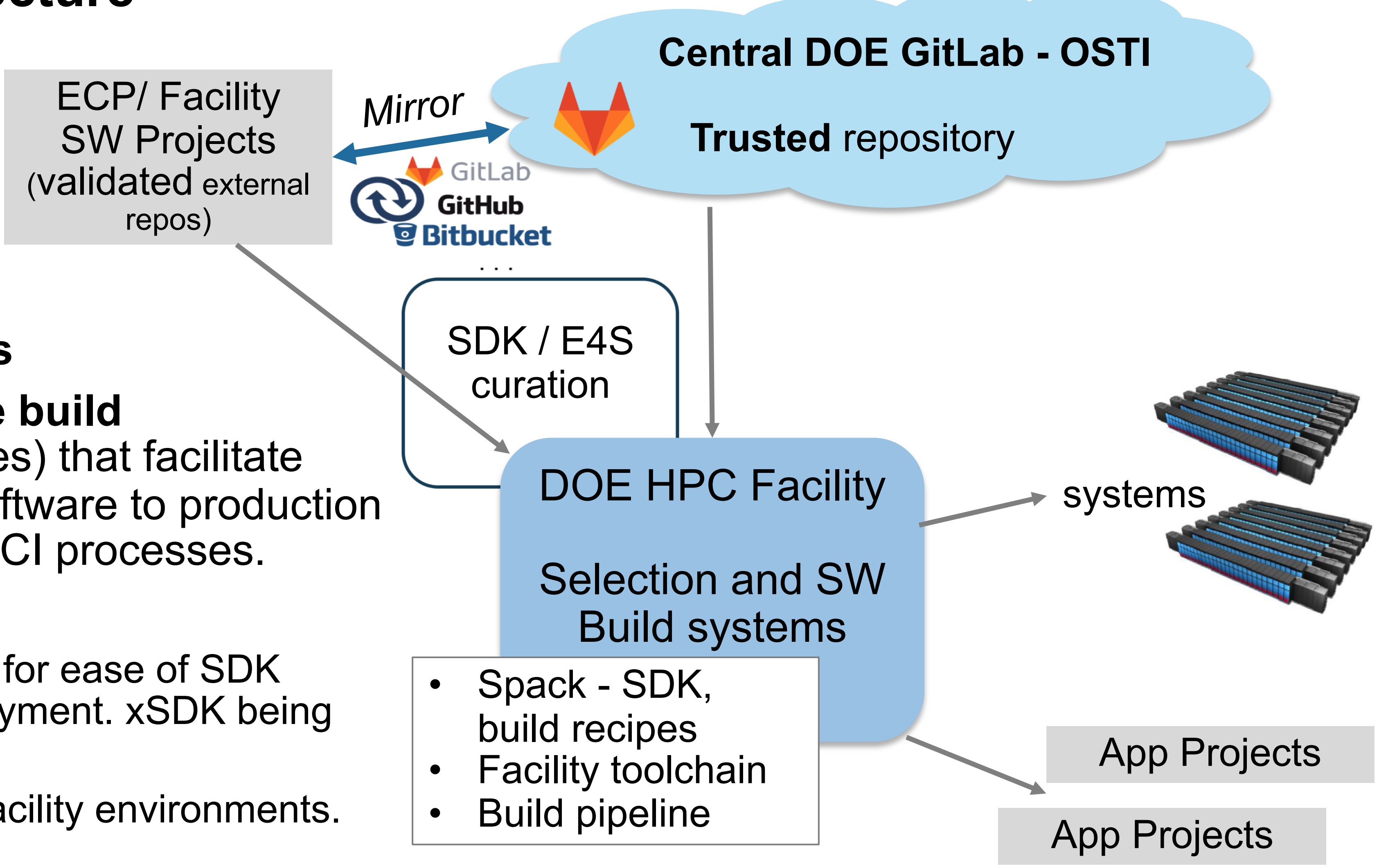
ECP Software Integration and Deployment

SW Deployment Architecture

In **partnership** with the ST software ecosystem project to **further develop Spack/ SDKs**, leverage software build pipelines, and assess container deployment.

In-Process

- **Establish efficient software build environments** (build pipelines) that facilitate testing and deployment of software to production and user environments. Use CI processes.
- **Spack software packaging**
  - Broader Spack relationships for ease of SDK packaging and Facility deployment. xSDK being implemented
  - Spack integration into ST/ Facility environments.



ECP/ Facility SW Projects (validated external repos)

Mirror

GitLab
GitHub
Bitbucket
. . .

Central DOE GitLab - OSTI

Trusted repository

SDK / E4S curation

DOE HPC Facility

Selection and SW Build systems

- Spack - SDK, build recipes
- Facility toolchain
- Build pipeline

systems

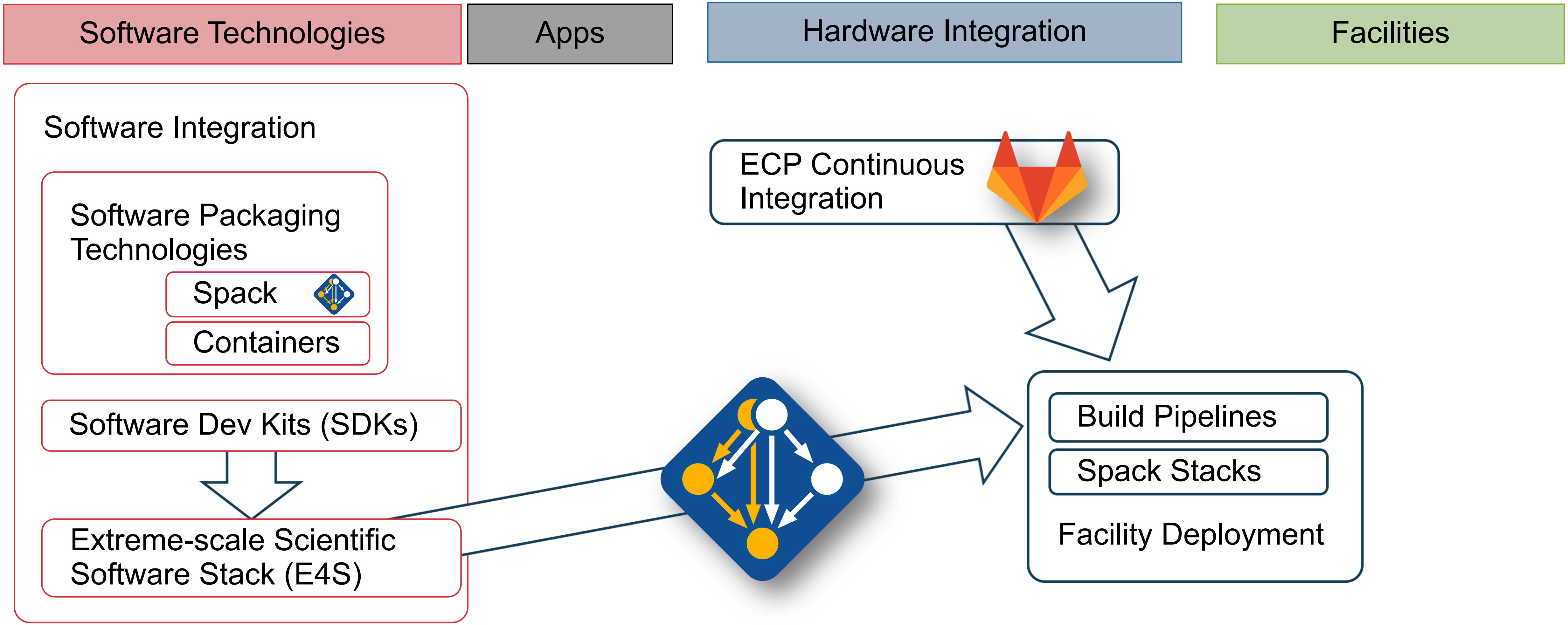
App Projects

App Projects

Spack is the delivery platform for the ECP software stack

- U.S. Exascale Computing Project (ECP) will release software through Spack
- Software in ECP stack needs to run on ECP platforms, testbeds, clusters, laptops
  - Each new environment requires effort.
- ECP asks us to build a robust, reliable, and easy-to-use software stack
- We will provide the infrastructure necessary to make this tractable:
  1. A dependency model that can handle HPC software
  2. A hub for coordinated software releases (like xSDK)
  3. Build and test automation for large packages across facility
  4. Hosted binary and source software distributions for *all* ECP HPC platforms

There are many activities around Spack within ECP



Software Technologies

Apps

Hardware Integration

Facilities

Software Integration

Software Packaging Technologies

Spack

Containers

Software Dev Kits (SDKs)

Extreme-scale Scientific Software Stack (E4S)

ECP Continuous Integration

Build Pipelines

Spack Stacks

Facility Deployment

Software Deployment in ECP Includes Many Efforts

Contributions to OSS projects

- ECP is building key infrastructure
- Working to bring more cloud-like services and automation to HPC
- Continuous Integration

OnyxPoint

Automating build and deployment

- Standardizing on a common package manager (Spack)
- Implementing build automation across HPC sites
- Trying to balance simple deployment with the complexity of the ecosystem

Kitware

Spack contributors

Facility Integration

Towards regular releases

- Socializing a release process with researchers and scientists
- Bringing teams together to do better integration testing
- Regular ECP-wide releases

ECP

EXASCALE COMPUTING PROJECT

By the end of the ECP, the Software Deployment project will..

Have established a cross-site Continuous Integration testing infrastructure that:

- Provides for account authentication and access to CI test resources across multiple sites
- Provides unique and targeted HPC test resources to support software development teams
- Established a standard process across the DOE sites for software development testing

Have an established and updated process to understand software needs between applications and software technology projects and established a feedback process to facility software support teams

- Software characterization / mapping and feedback processes

Have established a deployment process of ECP (and other) software via SDKs, Spack and an optimized build infrastructure

- Leveraging and building on software packaging tool infrastructure
- Establishing sharing and building on best practices across facilities
- Embracing new approaches to software deployment such as containers

Questions?