Software Development Kits

A Software Integration Strategy for CSE

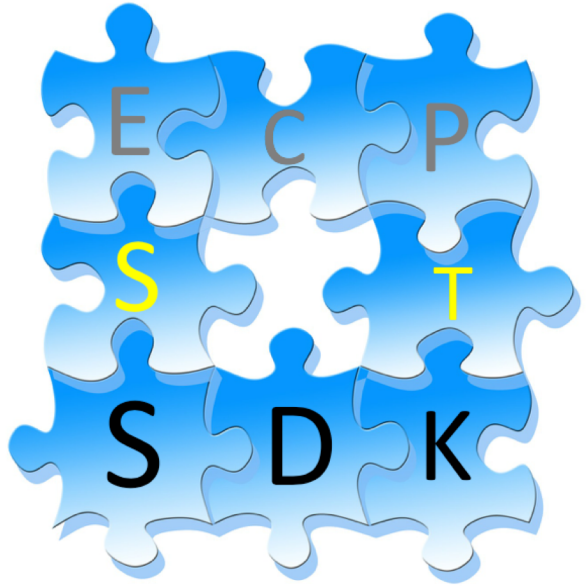
PI: James Willenbring - SNL

In collaboration with

ECP ST focus area SDK project leads:
- Chuck Atkins – Kitware
- Ulrike Meier Yang - LLNL
- Bart Miller – U Wisconsin
- Sameer Shende – U Oregon

ECP Software Deployment:
- Dave Montoya – LANL

February 26, 2019



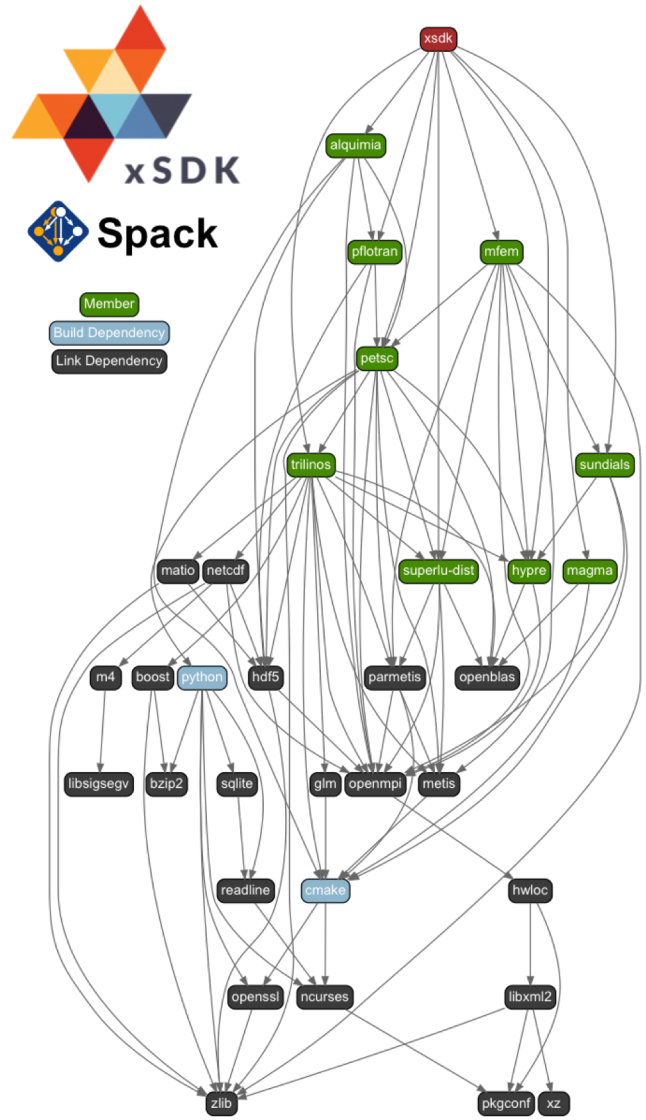
EXASCALE COMPUTING PROJECT

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

SAND2019-1978 C

SDK Motivation

- The exascale software ecosystem will be comprised of a wide array of software, all of which are expected to be used by DOE applications.
- The software must be:
  - interoperable
  - sustainable
  - maintainable
  - adaptable
  - portable
  - scalable
  - deployed at DOE computing facilities
- Without these qualities:
  - Value will be diminished
  - Scientific productivity will suffer



SW Development Kit (SDK) Overview

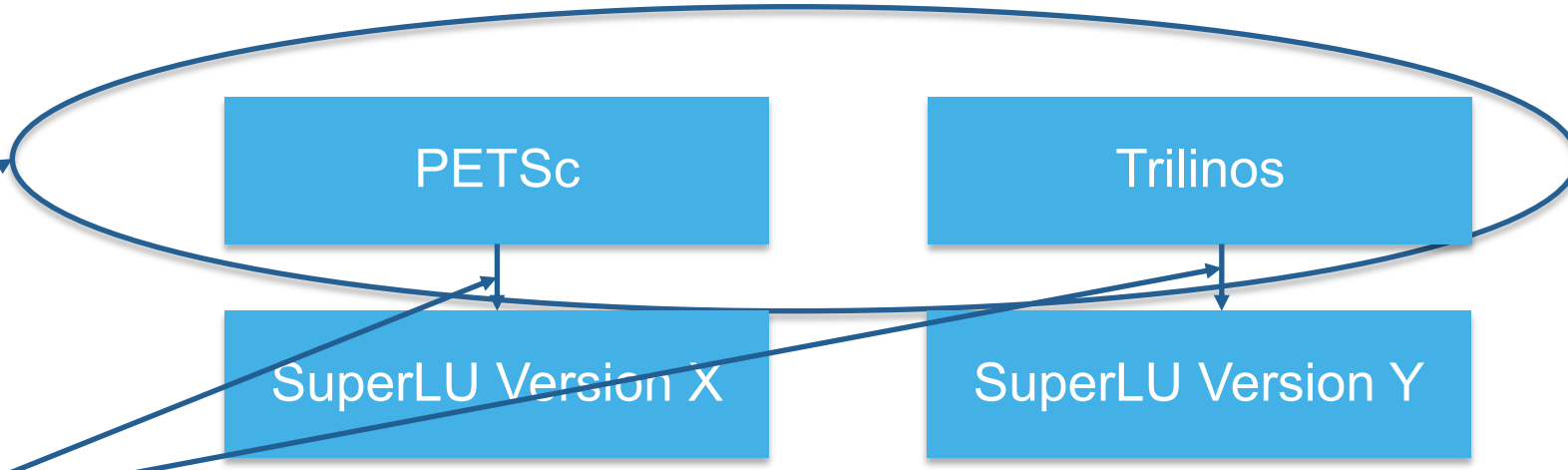
- SDK: A collection of related software products (called packages) where coordination across package teams will improve usability and practices and foster community growth among teams that develop similar and complementary capabilities. SDKs have the following attributes:
  - Domain scope: Collection makes functional sense.
  - Interaction model: How packages interact; compatible, complementary, interoperable.
    - Interfaces and common versions of 3rd party software.
  - Community policies: Value statements; serve as criteria for membership.
  - Community interaction: Communication between teams. Bridge culture. Common vocabulary.
  - Meta-infrastructure: Encapsulates, invokes build of all packages (Spack), shared test suites.
  - Coordinated plans: Inter-package planning. Does not replace autonomous package planning.
  - Community outreach: Coordinated, combined tutorials, documentation, best practices.
- Unity in essentials, otherwise diversity.

SDK Objectives

- SDKs will enable greater collaboration across ECP ST projects, and with the external software community. SDKs are essential to the ECP ST product delivery strategy, providing intermediate build and coordination points to better manage complexity.

- Through common community policies, deployment via the Extreme-Scale Scientific Software Stack (E4S) and testing that leverages the ECP Continuous Integration infrastructure (under development), SDKs will enhance interoperability and sustainability of ECP ST software.

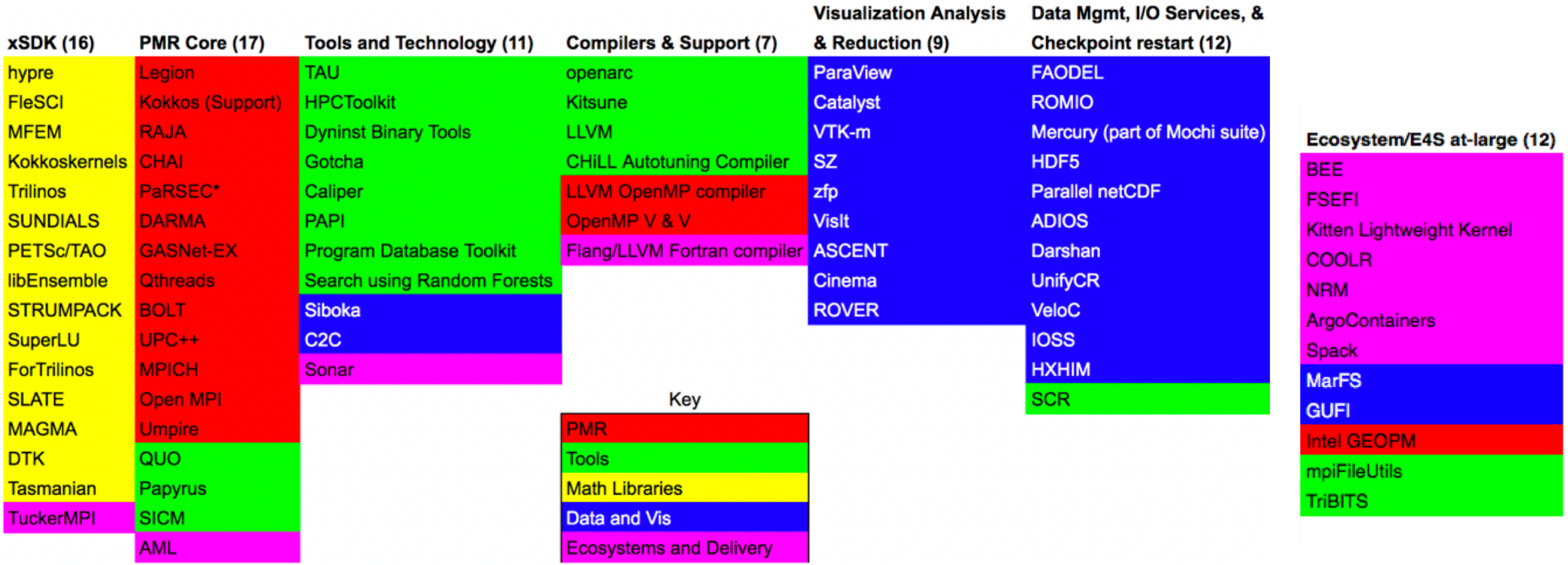
SDK "Horizontal" Grouping: Key Quality Improvement Driver



- Horizonal (vs Vertical) Coupling
  - Common substrate
  - Similar function and purpose
    - e.g. compiler frameworks, math libraries
  - Potential benefit from common Community Policies
    - Best practices in software design and development and customer support
  - Used together, but not in a long vertical dependency chain
  - Support for (and design of) common interfaces
    - Commonly an aspiration, not yet reality

Horizontal grouping:
- Assures X=Y.
- Protects against regressions.
- Transforms code coupling from heroic effort to turnkey.

ECP ST SDK Breakdown



ECP ST SDK community policies: Important team building, quality improvement, membership criteria.

SDK Community Policy Strategy

- Review and revise xSDK community policies and categorize
  - Generally applicable
  - In what context the policy is applicable
- Allow each SDK latitude in customizing appropriate community policies
- Establish baseline policies, continually refine

xSDK compatible package: Must satisfy mandatory xSDK policies:

M1. Support xSDK community GNU Autoconf or CMake options.

M2. Provide a comprehensive test suite.

M3. Employ user-provided MPI communicator.

M4. Give best effort at portability to key architectures.

M5. Provide a documented, reliable way to contact the development team.

...

Recommended policies: encouraged, not required:

R1. Have a public repository.

R2. Possible to run test suite under valgrind in order to test for memory corruption issues.

R3. Adopt and document consistent system for error conditions/exceptions.

R4. Free all system resources it has acquired as soon as they are no longer needed.

R5. Provide a mechanism to export ordered list of library dependencies.

xSDK member package: An xSDK-compatible package, *that* uses or can be used by another package in the xSDK, and the connecting interface is regularly tested for regressions.

https://xsdk.info/policies

Prior to defining and complying with these policies, a user could not correctly, much less easily, build hypre, PETSc, SuperLU and Trilinos in a single executable: a basic requirement for some ECP app multi-scale/multi-physics efforts.

Initially the xSDK team did not have sufficient common understanding to jointly define community policies.

Extreme-Scale Scientific Software Stack – E4S

- E4S: A Spack-based distribution of ECP ST and related and dependent software tested for interoperability and portability to multiple architectures
- Provides distinction between SDK usability / general quality / community and deployment / testing goals
- Will leverage and enhance SDK interoperability thrust
- Oct: E4S 0.1 - 24 full, 24 partial release products
- Jan: E4S 0.2 - 37 full, 10 partial release products
- Current primary focus: Facilities deployment

e4s.io

Lead: Sameer Shende (U Oregon)

E4S Full Release and Installed Packages

- Adios
- Bolt
- Caliper
- Darshan
- Gasnet
- GEOPM
- GlobalArrays
- Gotcha
- HDF5
- HPCToolkit
- Hypre
- Jupyter
- Kokkos
- Legion
- Libquo
- Magma
- MFEM
- MPICH
- OpenMPI
- PAPI
- Papyrus
- Parallel netCDF
- ParaView
- PETSc/TAO
- Program Database Toolkit (PDT)
- Qthreads
- Raja
- SCR
- Spack
- Strumpack
- Sundials
- SuperLU
- Swift/T
- SZ
- Tasmanian
- TAU
- Trilinos
- VTKm
- Umpire

Packages installed using Spack

- UnifyCR
- Veloc
- xSDK
- Zfp

E4S: Providing a Common Environment Using Containers

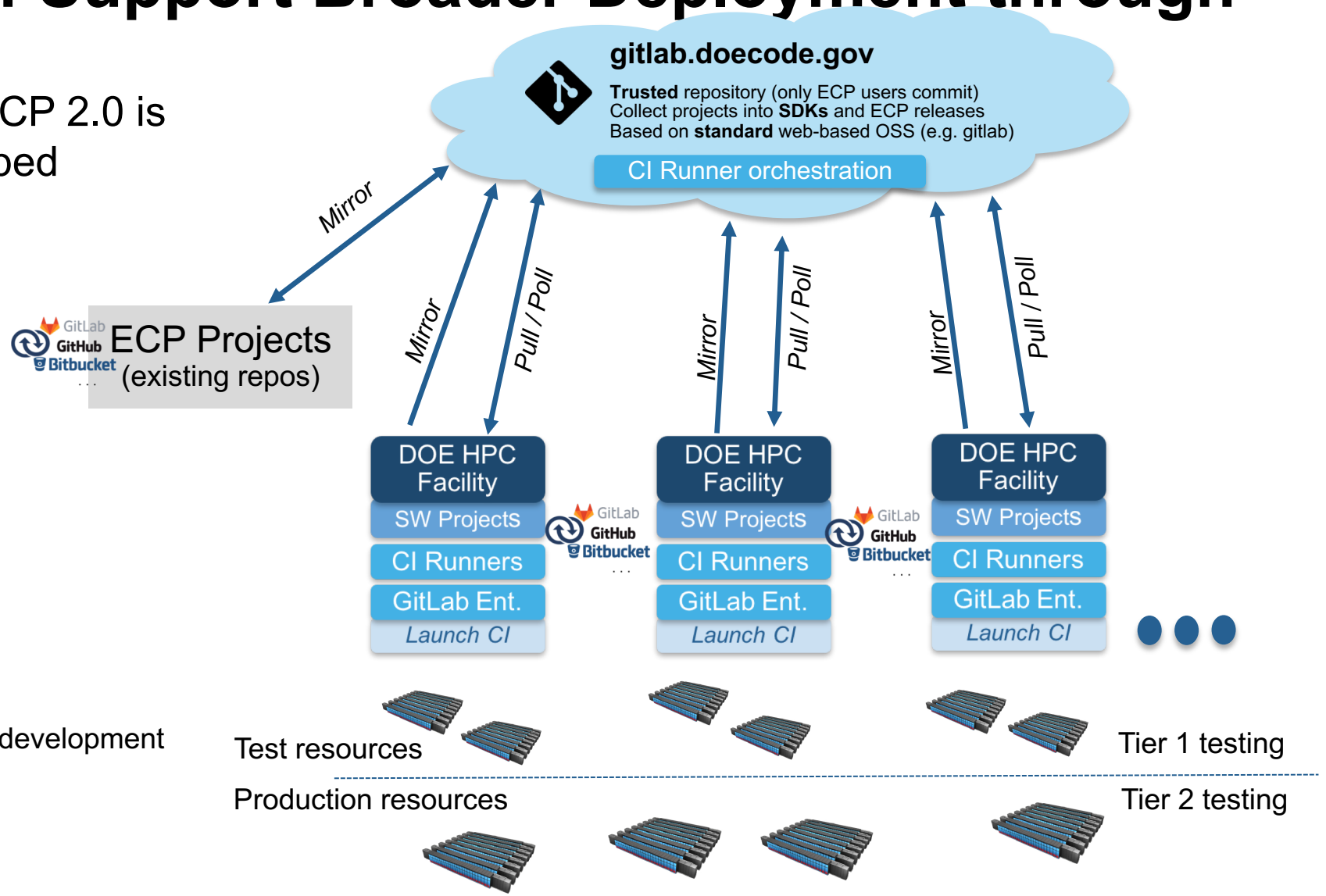
- Useful for:
  - Testing
    - Target platforms are well-defined and accessible
  - Development
  - Demonstration
    - Already used for different tutorials, including CANDLE
  - Deployment
  - Achieving interoperability
  - Creating Spack “recipes” and Spack Stacks
- Not a replacement for Spack-based build-from-source installations
  - Near-term deployment primarily bare metal
- Docker, Shifter, Singularity, and Charliecloud are supported
  - Different facilities support and are exploring different technologies

Long-term Impact: Support Broader Deployment through DOE Facilities

Hardware and Integration in ECP 2.0 is a key enabler to move developed infrastructure into deployment

Key Issues:

- Facility security
- Site identification – other resource sites
- Authentication
- Account approvals
- Gatekeeping
- User Id mapping
- Testbed availability
- Application CI testing
- Tests beyond build tests – Tier 2
- Integration of local CI environments
- Integration into local schedulers
- Testing and feedback during CI vendor development
- Process/Policy development
- Working groups for CI related efforts
- Etc..



SDK Project Risks and Issues

- Achieving compatibly and interoperability among and between dozens of software products is hard. Maintaining it is even harder
  - Continuous Integration (CI) testing capability through ECP Hardware Integration (HI) is crucial
  - Sustaining the interoperability beyond ECP requires proper workflows and sufficient funding
- Community building opportunities are more limited than for xSDK
  - Mutual understanding important for defining community policies
- xSDK project members were naturally (or forcefully) inclined to work together
  - Many other ECP objectives for everyone involved
  - IDEAS productivity project included xSDK creation as a primary focus area
- Opportunities for value added vary for each SDK
  - Effort will be required to define these opportunities

SDK and E4S Project Next Steps

- Organize individual SDKs
- ‘Kick the tires’ and leverage new ECP CI testing capability
  - Essential for maintaining interoperability
- Coordinate and negotiate E4S deployment with facilities
- Continue adding new products to E4S
- New software quality and assessment thrust
  - Focus on continuous improvement
    - Assessment required for baselining and measuring improvement
  - Quality goal for each ST project
  - Joint effort with IDEAS ECP

Summary

- Extending the SDK approach to all ECP ST domains
  - SDKs create a horizontal coupling of software products, teams
  - Enhance collaboration
  - Manage complexity
  - Improve interoperability, maintainability, sustainability and productivity
    - Shared testing
    - Community policies, best practices
- Deployment via E4S
  - Spack-based install, develop Spack recipes/stacks
  - Handle version compatibility issues and coordination of common dependencies
  - Support for container environments

Appendix:

- Packages installed through E4S ‘spack install’
- Location of E4S Containers
- Software Technology Release Vectors
- Project interactions

Packages Installed through E4S ‘spack install’

3. ssh

```
-- linux-centos7-x86_64 / gcc@4.8.5 ------------------------------
autoconf@2.69    cuda@9.1.85       gmp@6.1.2         kokkos@2.03.00       libxml2@2.9.4  mpich@3.2.1      openssl@1.0.2n  readline@7.0
automake@1.15.1  flex@2.6.4        help2man@1.47.4   libpciaccess@0.13.5  m4@1.4.18      ncurses@6.0      papi@5.5.1      tar@1.29
bison@3.0.4      gcc@7.3.0         hwloc@1.11.9      libsigsegv@2.11      magma@2.4.0    numactl@2.0.11   pdt@3.25        util-macros@1.19.1
bzip2@1.0.6      gdbm@1.14.1       hwloc@2.0.1       libtool@2.4.6        mpc@1.1.0      openblas@0.2.20  perl@5.24.1     xz@5.2.3
cmake@3.11.1     gettext@0.19.8.1  isl@0.19          libunwind@1.1        mpfr@4.0.1     openmpi@3.0.1    pkgconf@1.4.0   zlib@1.2.11

-- linux-centos7-x86_64 / gcc@7.3.0 ------------------------------
adios@1.13.1           freetype@2.7.1              json-c@0.13.1        libxfixes@5.0.2      papi@5.5.1          py-mccabe@0.6.1       sqlite@3.22.0
adlbx@0.8.0            gasnet@1.30.0               kbproto@1.0.7        libxml2@2.9.4        papyrus@develop     py-mock@2.0.0         stc@0.7.3
adlbx@0.8.0            gasnet@1.30.0               kokkos@2.03.00       libxshmfence@1.2     paraview@5.4.1      py-mpi4py@3.0.0       strumpack@3.1.1
ant@1.9.9              gdb@8.0.1                   kvtree@1.0.2         libxt@1.1.5          parmetis@4.0.3      py-natsort@5.2.0      suite-sparse@5.2.0
autoconf@2.69          gdbm@1.14.1                 lcms@2.8             libxv@1.0.10         patch@2.7.6         py-nose@1.3.7         sundials@3.1.0
automake@1.14          geopm@0.4.0                 legion@17.10.0       libxvmc@1.0.9        pcre@8.41           py-numexpr@2.6.1      superlu@5.2.1
automake@1.15.1        gettext@0.19.8.1            leveldb@1.20         libyogrt@1.20-6      pcre@8.41           py-numpy@1.13.3       superlu-dist@5.2.2
axl@0.1.1              git@2.15.1                  libarchive@3.3.2     lmod@7.7.13          pdsh@2.31           py-pandas@0.21.1      swig@3.0.12
binutils@2.27          glib@2.56.0                 libbsd@0.8.6         lua@5.3.4            pdt@3.25            py-pbr@3.1.1          sz@1.4.12.3
binutils@2.29.1        glm@0.9.7.1                 libcircle@0.2.1-rc.1 lua-luafilesystem@1_6_3 perl@5.24.1         py-pillow@3.2.0       tar@1.29
bison@3.0.4            globalarrays@5.7            libedit@3.1-20170329 lua-luaposix@33.4.0  petsc@3.8.4         py-pkgconfig@1.2.2    tasmanian@6.0
bolt@1.0b1             glproto@1.4.17              libffi@3.2.1         lwgrp@1.0.2          pflotran@xsdk-0.3.0 py-py@1.4.33          tau@2.28
boost@1.66.0           gmp@6.1.2                   libice@1.0.9         lz4@1.8.1.2          pixman@0.34.0       py-pycodestyle@2.3.1  tcl@8.6.8
boost@1.66.0           gobject-introspection@1.49.2 libiconv@1.15       lzma@4.32.7          pkgconf@1.4.0       py-pyflakes@1.6.0     texinfo@6.5
boost@1.68.0           gotcha@0.0.2                libjpeg-turbo@1.5.3  lzo@2.09             presentproto@1.0    py-pyparsing@2.2.0    tk@8.6.8
bzip2@1.0.6            gotcha@develop              libmng@2.0.3         m4@1.4.18            protobuf@3.5.1.1    py-pytables@3.3.0     trilinos@12.12.1
c-blosc@1.12.1         gperf@3.0.4                 libpciaccess@0.13.5  matio@1.5.9          py-argparse@1.4.0   py-pytest@3.6.0       turbine@1.0.0
cairo@1.14.12          harfbuzz@1.4.6              libpfm4@4.8.0        metis@5.1.0          py-babel@2.4.0      py-pytz@2017.2        turbine@1.0.0
caliper@1.8.0          hdf5@1.8.19                 libpng@1.6.34        mfem@3.3.2           py-bottleneck@1.0.0 py-scipy@1.0.0        umpire@master
cmake@3.11.1           hdf5@1.8.19                 libpthread-stubs@0.4 miniconda2@4.3.30    py-configparser@3.5.0 py-setuptools@39.0.1 unifycr@master
conduit@master         hdf5@1.10.1                 libquo@1.3           miniconda3@4.3.30    py-cycler@0.10.0    py-six@1.11.0         util-macros@1.19.1
curl@7.59.0            hdf5@1.10.1                 libsigsegv@2.11      mpich@3.2.1          py-cython@0.28.1    py-subprocess32@3.2.7 veloc@1.0
damageproto@1.2.1      hdf5@1.10.1                 libsm@1.2.2          mumps@5.1.1          py-dateutil@2.5.2   python@2.7.14         videoproto@2.3.3
darshan-runtime@3.1.6  hdf5@1.10.1                 libtiff@4.0.6        nasm@2.13.03         py-enum34@1.1.6     qhull@2015.2          vtkm@master
darshan-util@3.1.6     help2man@1.47.4             libtiff@4.0.8        ncurses@6.0          py-flake8@3.5.0     qthreads@1.12         vtkm@1.1.0
doxygen@1.8.12         hpctoolkit@2017.06          libtool@2.4          netcdf@4.4.1.1       py-funcsigs@0.4     r@3.4.3               xcb-proto@1.13
dtcmp@1.1.0            hpctoolkit-externals@2017.06 libtool@2.4.2       netlib-scalapack@2.0.2 py-functools32@3.2.3-2 raja@0.5.3       xextproto@7.3.0
er@0.0.3               hwloc@1.11.9                libtool@2.4.6        nettle@3.3           py-h5py@2.7.1       rankstr@0.0.2         xproto@7.0.31
exmcutils@0.5.3        hwloc@2.0.1                 libunwind@1.1        ninja@1.8.2          py-hypothesis@3.7.0 readline@7.0          xtrans@1.3.5
expat@2.2.2            hypre@2.13.0                libx11@1.6.5         numactl@2.0.11       py-jinja2@2.9.6     redset@0.0.3          xz@5.2.3
fftw@3.3.7             hypre@2.13.0                libxau@1.0.8         openblas@0.2.20      py-kiwisolver@1.0.1 ruby@2.2.0            zfp@0.5.0
fixesproto@5.0         icu4c@60.1                  libxcb@1.13          openmpi@3.0.1        py-lit@0.5.0        ruby-ronn@0.7.3       zlib@1.2.11
flex@2.6.4             inputproto@2.3.2            libxdamage@1.1.4     openssl@1.0.2n       py-mako@1.0.4       scr@1.2.2             zsh@5.4.2
font-util@1.3.1        intel-tbb@2018.2            libxdmcp@1.1.2       otf2@2.1             py-markupsafe@1.0   shuffile@0.0.3        zstd@1.3.0
fontconfig@2.12.3      jdk@8u141-b15               libxext@1.3.3        pango@1.41.0         py-matplotlib@2.2.2 snappy@1.1.7
%
```

Location of E4S Containers

- Docker:
  - http://tau.uoregon.edu/ecp.tgz
  - % gunzip ecp.tgz | docker load
  - % docker run –v <localdir>:<mountdir> -i –t sameer/ecp-pmr /bin/bash
  - # spack find
- Shifter
  - % shifterimg images
  - % shifter --image=registry/sameer/ecp-pmr -- /bin/bash --rcfile /etc/bashrc
- Singularity
  - http://tau.uoregon.edu/ecp.simg
  - % singularity exec ecp.simg /bin/bash --rcfile /etc/bashrc
- Charliecloud
  - http://tau.uoregon.edu/ecp-cc.tgz
  - % tar xf ecp-cc.tgz;
  - % ch-run --bind=<localdir>:<mountdir> ./ecp-cc -- /bin/bash --rcfile /etc/bashrc
- Archives:
  - http://tau.uoregon.edu/ecp

Software Technologies Release Vectors

ECP software projects

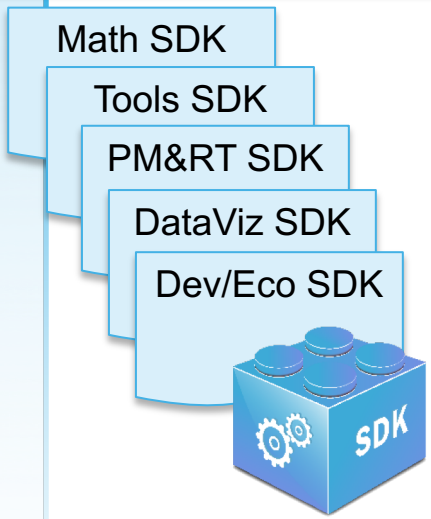
Each project to define (potentially ≥2) release vectors

More projects ← → Fewer projects

SDKs

Reusable software libraries embedded in applications; cohesive/interdependent libraries released as sets modeled on xSDK

- Regular coordinated releases via E4S
- Hierarchical collection built on Spack
- Establish community policies for library development
- Apply Continuous Integration and other robust testing practices



OpenHPC

Potential exit strategy for binary distributions

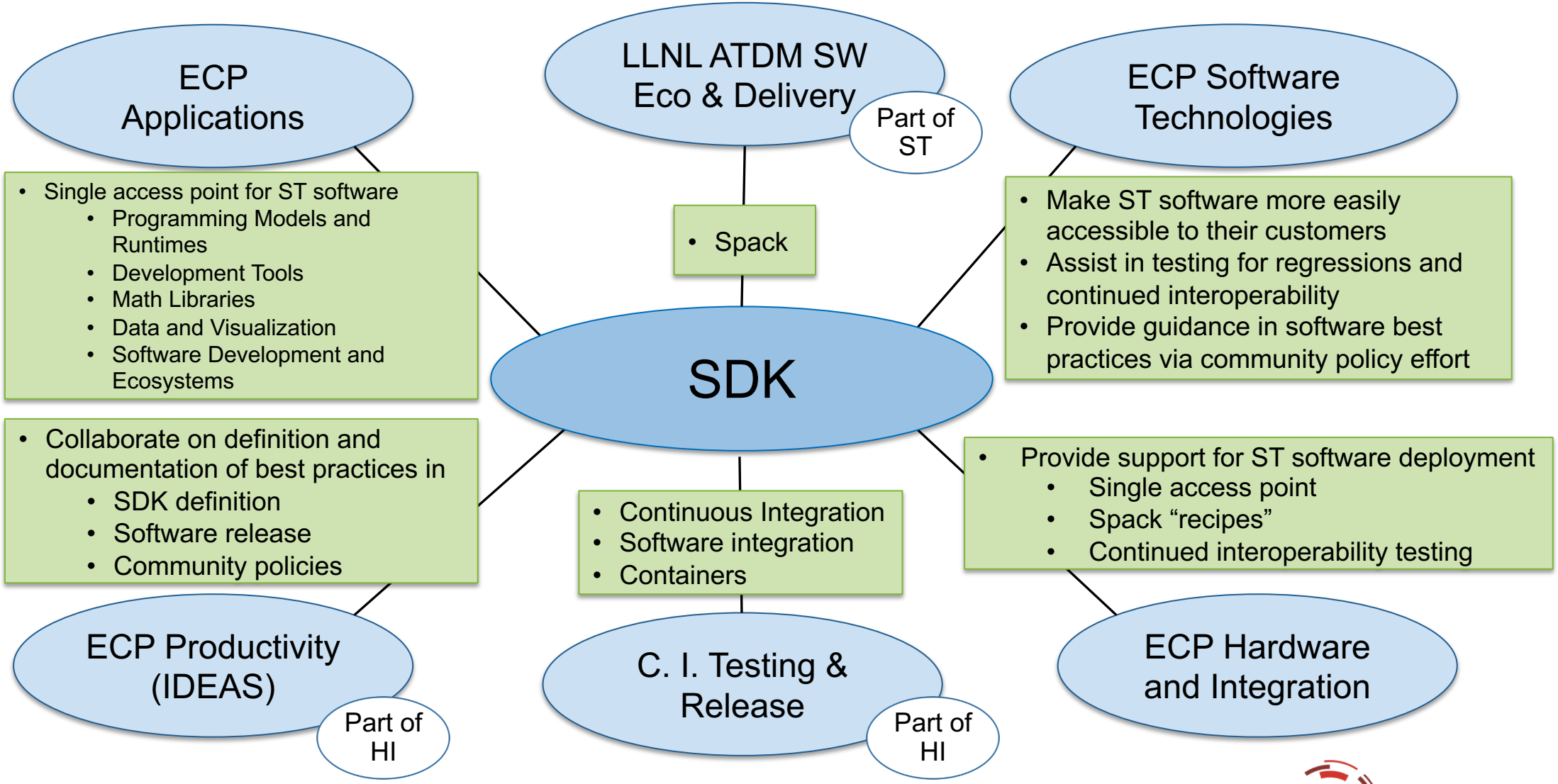
- Target similar software to existing OpenHPC stack
- Develop super-scalable release targeting higher end systems

Direct2Facility

Platform-specific software in support of a specified 2021–2023 exascale system

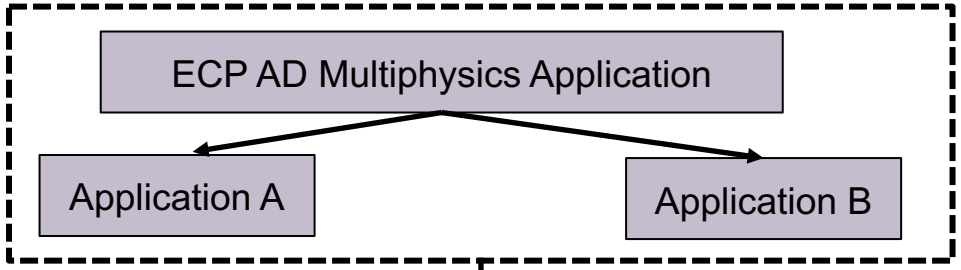
- Software **exclusively** supporting a specific platform
- System software, some tools and runtimes

SDK relationships throughout ECP



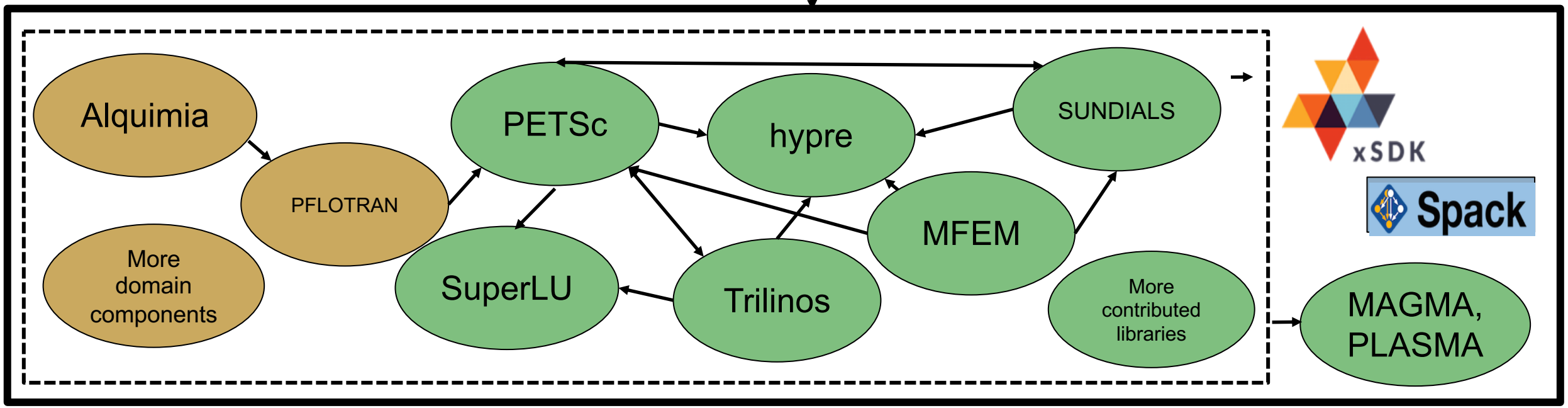
xSDK: The First SDK

Notation: A → B:
A can use **B** to provide functionality on behalf of **A**



xSDK functionality, Nov 2017

Tested on key machines at ALCF, NERSC, OLCF, also Linux, Mac OS X



EXASCALE COMPUTING PROJECT