# The ECP Software Stack

Michael Heroux, Director
Sandia National Laboratories

GoingArm Workshop
June 28, 2018

# Scope

**Deliver a software stack that enables sustainable exascale capabilities**

Mission need

Capabilities across the entire HPC software stack that complement and coordinate with facilities, vendors and other software providers to enable effective execution of ECP apps, and **deliver a capable, sustainable exascale ecosystem**.

Objective

Provide the next generation of DOE software capabilities targeted toward exascale applications and platforms. Provide these capabilities for the specific exascale systems as a high quality, sustainable product suite.

# ECP Software: Productive, sustainable ecosystem

## Goal

Build a comprehensive, coherent software stack that enables application developers to productively write highly parallel applications that effectively target diverse exascale architectures

- Extend current technologies to exascale where possible
- Perform R&D required for new approaches when necessary
- Coordinate with and complement vendor efforts
- Develop and deploy high-quality and robust software products

55 WBS L4 subprojects executing RD&D
185 L4 subproject (P6) milestones delivered in FY17
67 delivered so far in FY18, 77 in progress right now
564 L4 subproject (P6) milestones planned in FY18-19

# ECP software: Challenges

## Challenges

Qualitative changes: Massive concurrency; Multi-scale, multi-physics, data-driven science; Ecosystem integration

- Billion way concurrency: Several novel compute nodes.
- Coupled apps: Physics, scales, in situ data, more.
- Data-driven: New software HPC environments, containers.
- Ecosystem: Part of a large, complex, evolving SW environment.

# Exascale Computing Project 2.0

- **Project Management 2.1**
  - Project Planning and Management 2.1.1
  - Project Controls and Risk Management 2.1.2
  - Business Management 2.1.3
  - Procurement Management 2.1.4
  - Information Technology and Quality Management 2.1.5
  - Communications and Outreach 2.1.6
- **Application Development 2.2**
  - Chemistry and Materials Applications 2.2.1
  - Energy Applications 2.2.2
  - Earth and Space Science Applications 2.2.3
  - Data Analytics and Optimization Applications 2.2.4
  - National Security Applications 2.2.5
  - Co-Design 2.2.6
- **Software Technology 2.3**
  - Programming Models and Runtimes 2.3.1
  - Development Tools 2.3.2
  - Mathematical Libraries 2.3.3
  - Data and Visualization 2.3.4
  - Software Ecosystem and Delivery 2.3.5
- **Hardware and Integration 2.4**
  - PathForward 2.4.1
  - Hardware Evaluation 2.4.2
  - Application Integration at Facilities 2.4.3
  - Software Deployment at Facilities 2.4.4
  - Facility Resource Utilization 2.4.5
  - Training and Productivity 2.4.6

# 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.

**Jonathan Carter, Software Technology Deputy Director**
Jonathan has been involved in the support and development of HPC applications for chemistry, the procurement of HPC systems, and the evaluation of novel computing hardware for over 25 years. He currently a senior manager in Computing Sciences at Berkeley Lab.

**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.

**Lois Curfman McInnes, Math Libraries (2.3.3)**
Lois is a senior computational scientist in the Mathematics and Computer Science Division of ANL. She has over 20 years of experience in high-performance numerical software, including development of PETSc and leadership of multi-institutional work toward sustainable scientific software ecosystems.

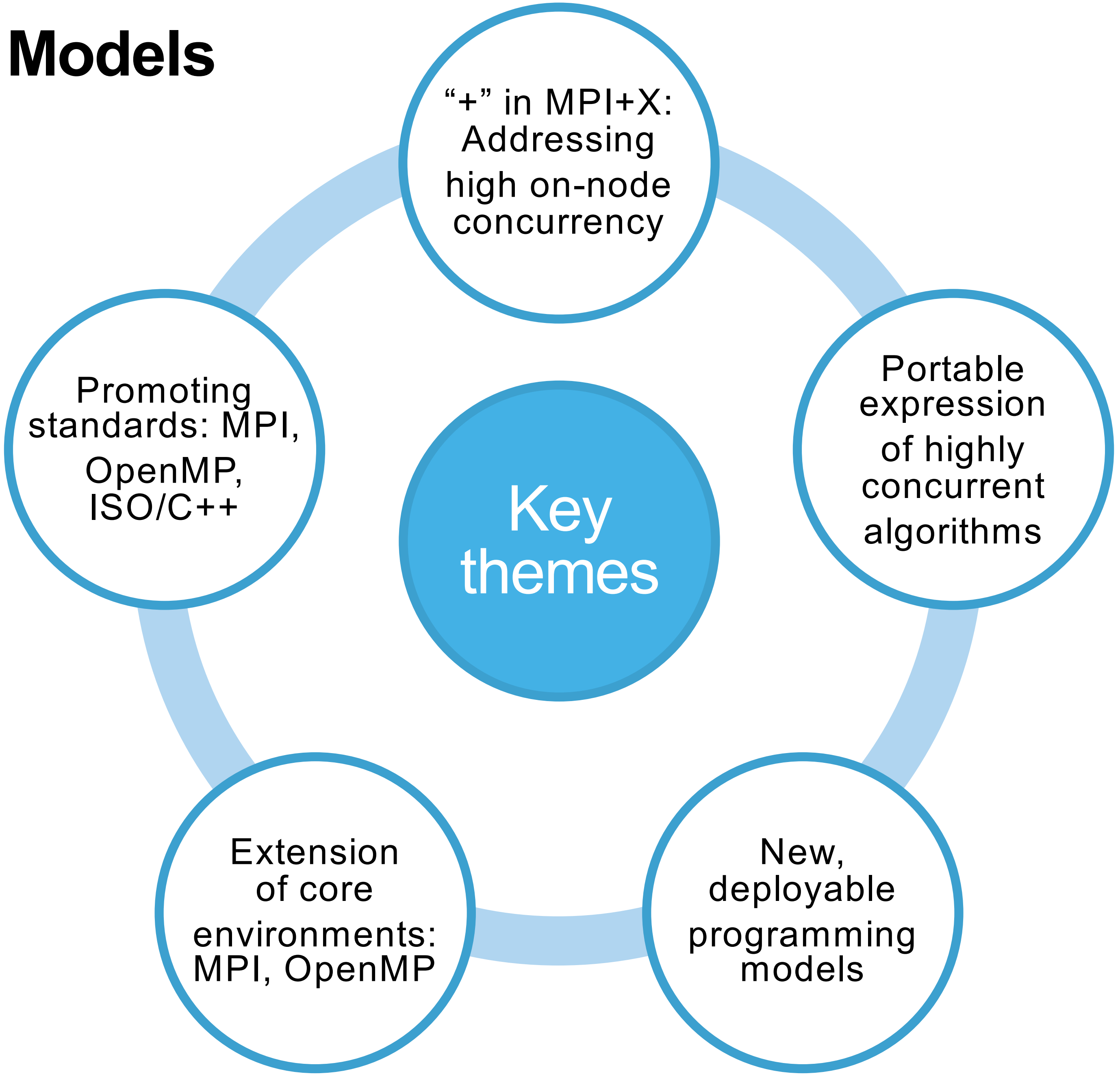
**Jim Ahrens, Data and Visualization (2.3.4)**
Jim is a senior research scientist at the Los Alamos National Laboratory (LANL) and an expert in data science at scale. He started and actively contributes to many open-source data science packages including ParaView and Cinema.

**Rob Neely, Software Ecosystem and Delivery (2.3.5)**
Rob has several leadership roles at LLNL spanning applications, CS research, platforms, and vendor interactions. He is an Associate Division Leader in the Center for Applied Scientific Computing (CASC), chair of the Weapons Simulation and Computing Research Council, and the lead for the Sierra Center of Excellence.

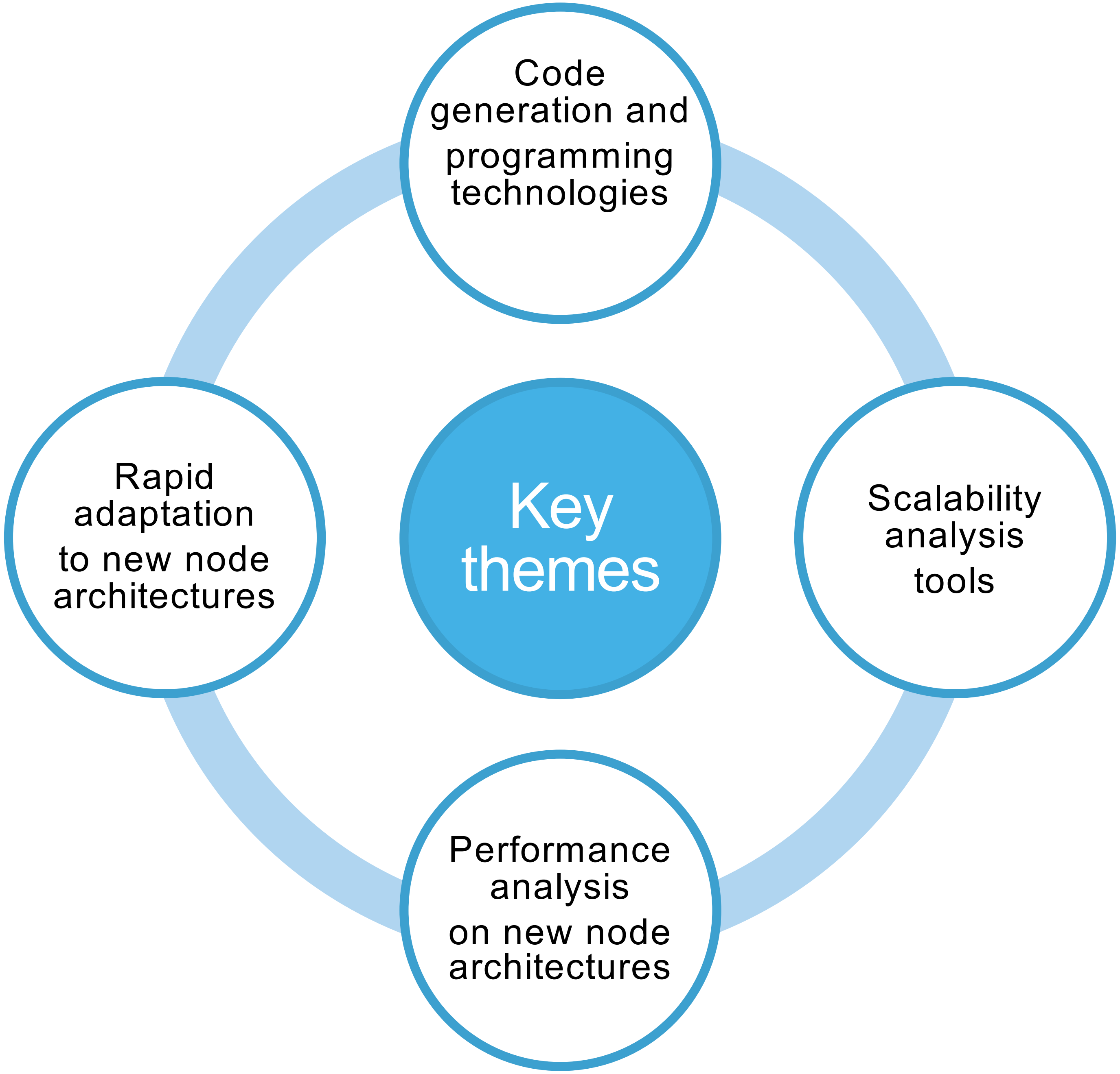
ECP EXASCALE COMPUTING PROJECT

## *Overview of ECP ST Technical Areas*

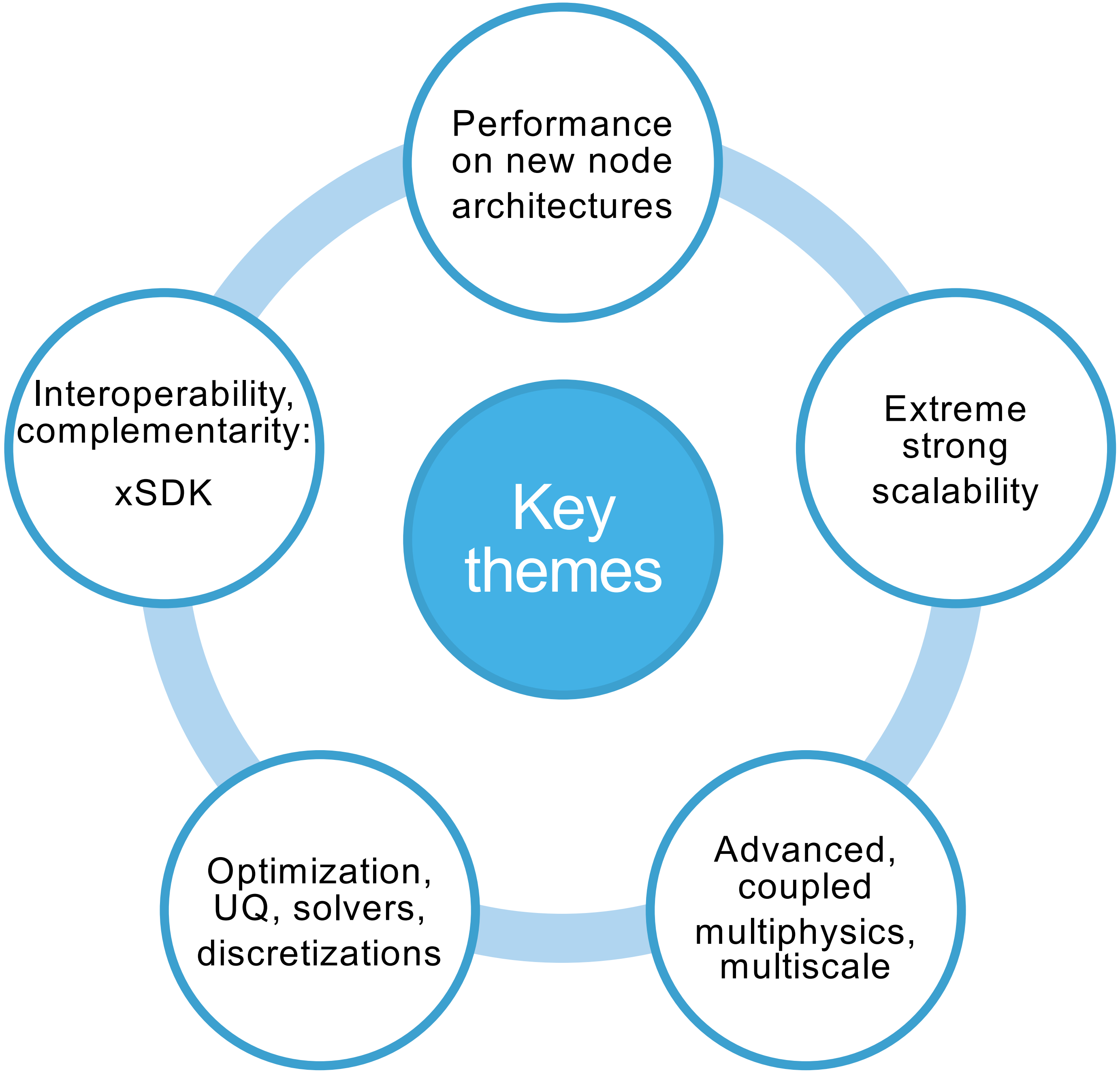
# Programming Models and Runtimes



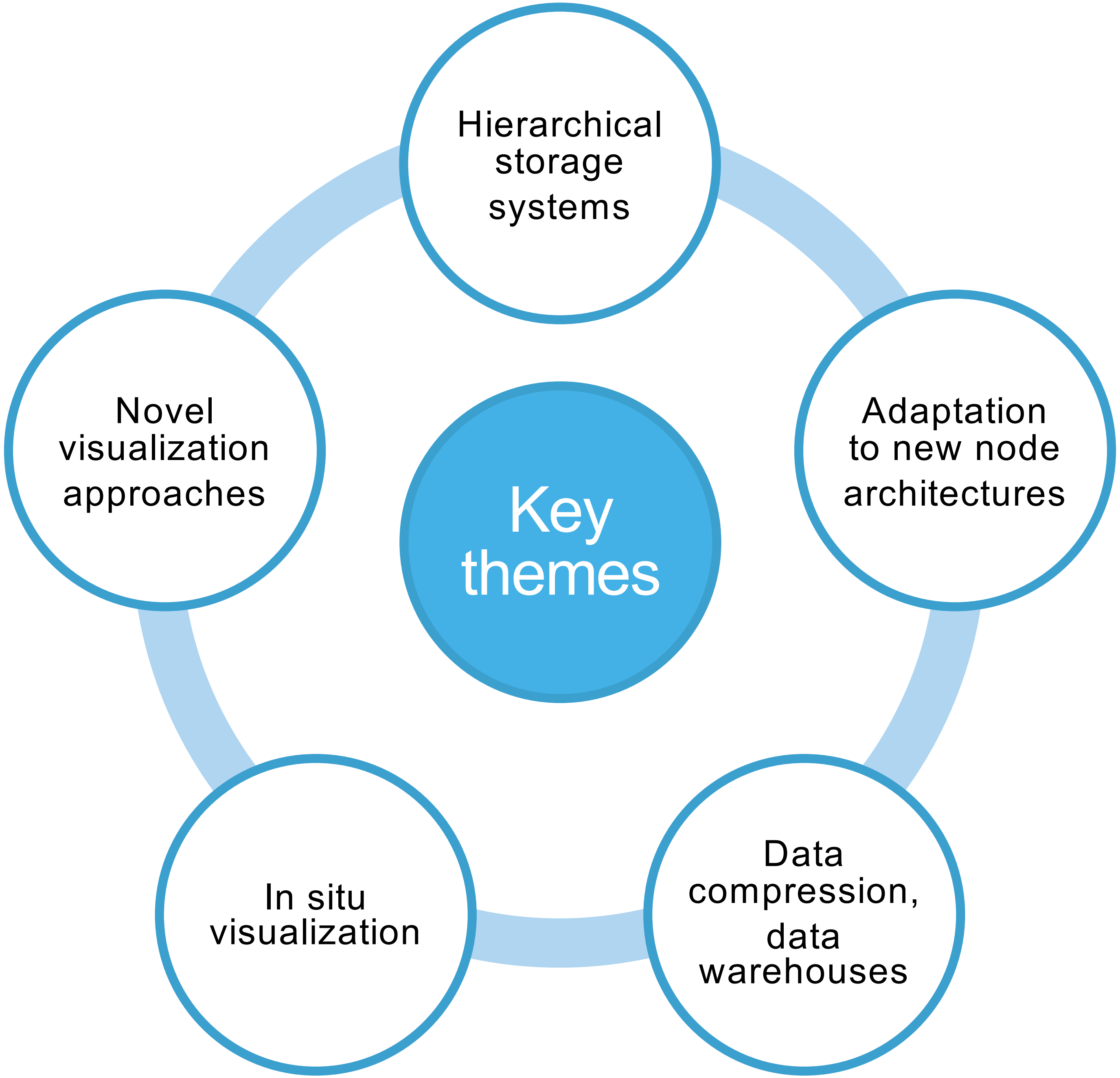
# Development Tools



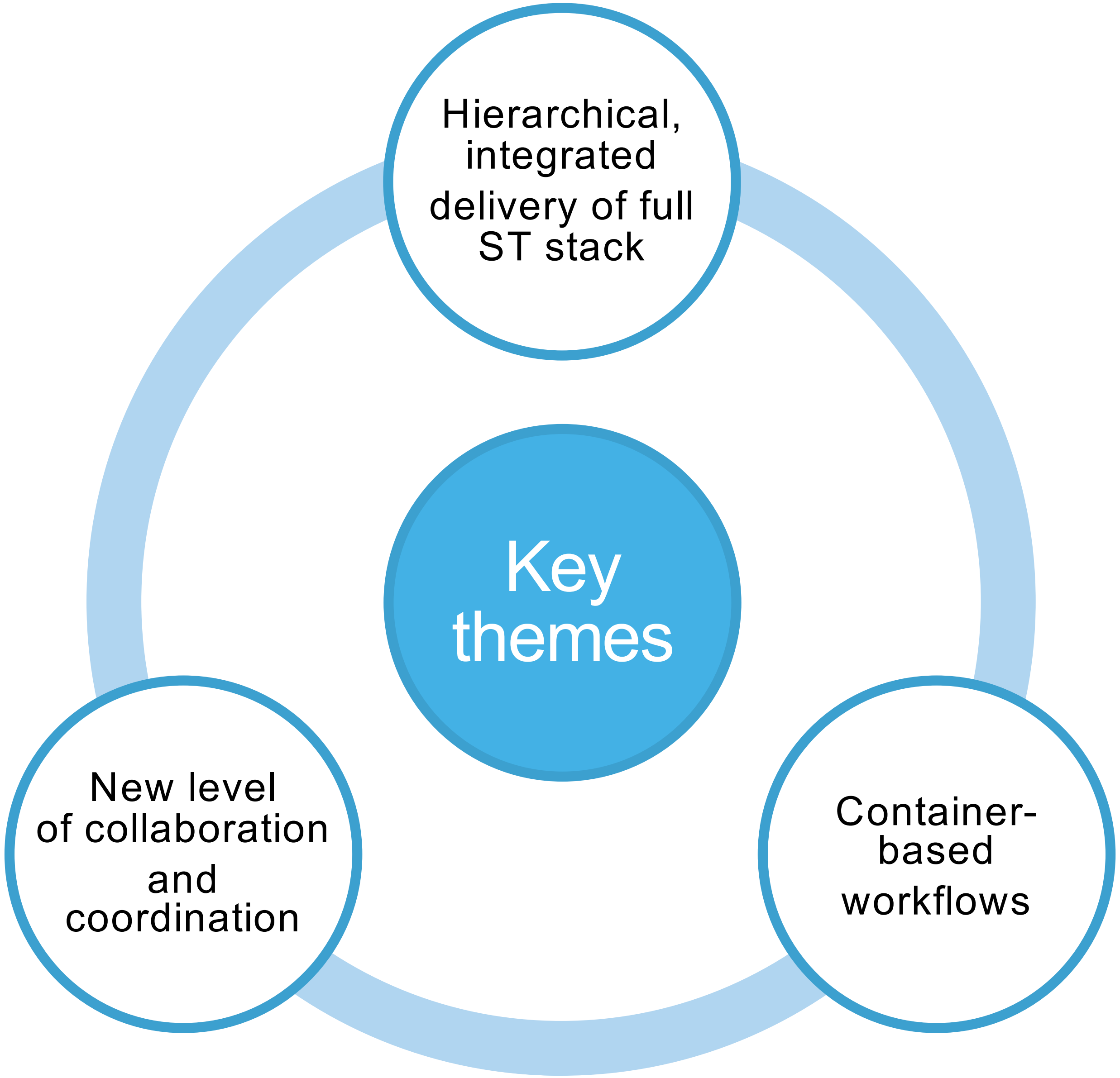
# Math Libraries



# Data and Visualization



# Software Ecosystem and Delivery



# ECP ST Products:
# 55 Projects contribute to 89 Unique Products

| Spack Package Support | |
|---|---|
| Have Spack Package | 43 |
| Spack Package in progress | 21 |

| Source Build System | |
|---|---|
| Cmake | 44 |
| Configure/Make (autotools) | 32 |
| Custom | 4 |

| Delivery | |
|---|---|
| Direct to users from source | 81 |
| Vendor stack | 11 |
| ALCF | 19 |
| OLCF | 20 |
| NERSC | 20 |
| LLNL | 18 |
| LANL | 17 |
| OpenHPC | 9 |
| Containers (Docker) | 3+ |

| User Support | |
|---|---|
| Documentation | 81 |
| Tutorials | 50 |
| Support staff training | 21 |
| Email/phone contact | 70 |
| User-access issue tracking | 65 |

- 48% support Spack.
- 24% Spack in progress.

➢ **Requirement for Q1FY19 participation.**

- Most users directly manage ST software from source.

➢ **Spack packages, SDKs will improve access and management.**

- ST projects have diverse delivery experience with:
  - vendors,
  - leadership facilities,
  - binary release,
  - Containers

➢ **Can leverage across other projects.**

Stats collected April 2018

# Programming Models and Runtimes Products (18)

| | |
|---|---|
| Legion | http://legion.stanford.edu |
| ROSE | https://github.com/rose-compiler |
| Kokkos | https://github.com/kokkos |
| DARMA | https://github.com/darma-tasking |
| Global Arrays | http://hpc.pnl.gov/globalarrays/ |
| RAJA | https://github.com/LLNL/RAJA |
| CHAI | https://github.com/LLNL/CHAI |
| Umpire | |
| MPICH | http://www.mpich.org |
| PaRSEC | http://icl.utk.edu/parsec/ |
| Open MPI | https://www.open-mpi.org/ |
| Intel GEOPM | https://geopm.github.io/ |
| LLVM OpenMP compiler | https://github.com/SOLLVE |
| OpenMP V&V Suite | https://bitbucket.org/crpl_cisc/sollve_vv/src |
| BOLT | https://github.com/pmodels/argobots |
| UPC++ | http://upcxx.lbl.gov |
| GASNet-EX | http://gasnet.lbl.gov |
| Qthreads | https://github.com/Qthreads |

# Development Tools (19)

| | |
|---|---|
| SICM | ~~https://confluence.exascaleproject.org/display/STSS07~~ |
| QUO | ~~https://github.com/lanl/libquo~~ |
| Kitsune | ~~https://github.com/lanl/kitsune~~ |
| SCR | ~~https://github.com/llnl/scr~~ |
| Caliper | ~~https://github.com/llnl/caliper~~ |
| mpiFileUtils | ~~https://github.com/hpc/mpifileutils~~ |
| Gotcha | ~~http://github.com/llnl/gotcha~~ |
| TriBITS | ~~https://tribits.org~~ |
| Exascale Code Geneneration Toolkit | |
| PAPI | ~~http://icl.utk.edu/exa-papi/~~ |
| CHiLL Autotuning Compiler | |
| Search using Random Forests (SuRF) | |
| HPCToolkit | ~~http://hpctoolkit.org~~ |
| The Dyninst Binary Tools Suite | ~~http://www.paradyn.org~~ |
| Tau | ~~http://www.cs.uoregon.edu/research/tau~~ |
| Papyrus | ~~https://ft.ornl.gov/research/papyrus~~ |
| openarc | ~~https://ft.ornl.gov/research/openarc~~ |
| LLVM | ~~http://llvm.org/~~ |
| Program Database Toolkit (PDT) | ~~https://www.cs.uoregon.edu/research/pdt/home.php~~ |

# Mathematical Libraries Products (16)

| | |
|---|---|
| xSDK | https://xsdk.info |
| hypre | http://www.llnl.gov/casc/hypre |
| FleCSI | http://www.flecsi.org |
| MFEM | http://mfem.org/ |
| Kokkoskernels | https://github.com/kokkos/kokkos-kernels/ |
| Trilinos | https://github.com/trilinos/Trilinos |
| SUNDIALS | https://computation.llnl.gov/projects/sundials |
| PETSc/TAO | http://www.mcs.anl.gov/petsc |
| libEnsemble | https://github.com/Libensemble/libensemble |
| STRUMPACK | http://portal.nersc.gov/project/sparse/strumpack/ |
| SuperLU | http://crd-legacy.lbl.gov/~xiaoye/SuperLU/ |
| ForTrilinos | https://trilinos.github.io/ForTrilinos/ |
| SLATE | http://icl.utk.edu/slate/ |
| MAGMA-sparse | https://bitbucket.org/icl/magma |
| DTK | https://github.com/ORNL-CEES/DataTransferKit |
| Tasmanian | http://tasmanian.ornl.gov/ |



EXASCALE COMPUTING PROJECT

# Data & Visualization Products (25)

| | |
|---|---|
| HXHIM | http://github.com/hpc/hxhim.git |
| Cinema | https://datascience.lanl.gov/Cinema.html |
| MarFS | https://github.com/mar-file-system/marfs |
| GUFI (The Grand Unified File Index) | https://github.com/mar-file-system/GUFI |
| Siboka | |
| ROVER | |
| C2C | |
| TuckerMPI | |
| ParaView | https://www.paraview.org/ |
| Catalyst | https://www.paraview.org/in-situ/ |
| VTK-m | http://m.vtk.org |
| FAODEL | https://github.com/faodel/faodel |
| IOSS | https://github.com/gsjaardema/seacas |
| VeloC | https://xgitlab.cels.anl.gov/ecp-veloc |
| SZ | https://github.com/disheng222/SZ |
| UnifyCR | https://github.com/LLNL/UnifyCR |
| HDF5 | https://www.hdfgroup.org/downloads/ |
| ADIOS | https://github.com/ornladios/ADIOS2 |
| Parallel netCDF | http://cucis.ece.northwestern.edu/projects/PnetCDF/ |
| Darshan | http://www.mcs.anl.gov/research/projects/darshan/ |
| ROMIO | http://www.mcs.anl.gov/projects/romio/ |
| Mercury (part of Mochi suite) | http://www.mcs.anl.gov/research/projects/mochi/ |
| zfp | https://github.com/LLNL/zfp |
| VisIt | https://wci.llnl.gov/simulation/computer-codes/visit |
| ASCENT | https://github.com/Alpine-DAV/ascent |

# SW Ecosystem & Delivery Products (11)

| | |
|---|---|
| BEE | |
| FSEFI | |
| Spack | https://github.com/spack/spack |
| Sonar | |
| Secure JupyterHub | |
| Kitten Lightweight Kernel | https://github.com/HobbesOSR/kitten |
| AML | https://xgitlab.cels.anl.gov/argo/aml |
| ArgoContainers | https://xgitlab.cels.anl.gov/argo/containers |
| COOLR | https://github.com/coolr-hpc |
| NRM | https://xgitlab.cels.anl.gov/argo/nrm |
| Flang/LLVM Fortran compiler | http://www.flang-compiler.org |

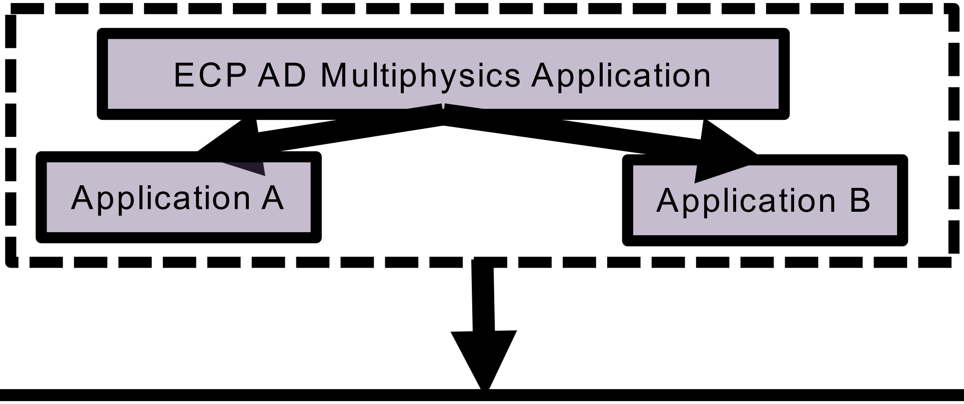
# *ECP ST SDKs*

# 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 package interact; compatible, complementary, interoperable.
  - Community policies: Value statements; serve as criteria for membership.
  - 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
- Overarching goal: Unity in essentials, otherwise diversity.

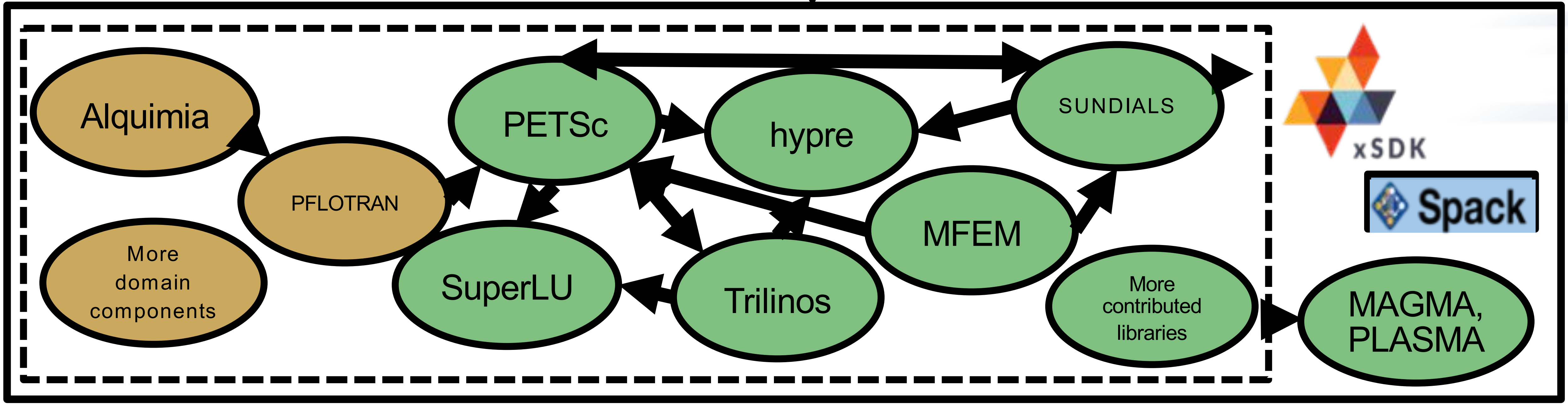
# xSDK interactions: Apps, packages, Spack

**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



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

**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.
**M6.** Respect system resources and settings made by other previously called packages.
**M7.** Come with an open source license.
**M8.** Provide a runtime API to return the current version number of the software.
**M9.** Use a limited and well-defined symbol, macro, library, and include file name space.
**M10.** Provide an accessible repository (not necessarily publicly available).
**M11.** Have no hardwired print or IO statements.
**M12.** Allow installing, building, and linking against an outside copy of external software.
**M13.** Install headers and libraries under <prefix>/include/ and <prefix>/lib/.
**M14.** Be buildable using 64 bit pointers. 32 bit is optional.
**M15.** All xSDK compatibility changes should be sustainable.
**M16.** The package must support production-quality installation compatible with the xSDK install tool and xSDK metapackage.

Also specify **recommended policies**, which currently are encouraged but 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**: Must be an xSDK-compatible package, *and* it uses or can be used by another package in the xSDK, and the connecting interface is regularly tested for regressions.

*Prior to defining and complying to 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.*

https://xsdk.info/policies

Version 0.3.0, Nov 2017

EXASCALE COMPUTING PROJECT

# Software Development Kits Progress: Leadership in place, Spack packaging making rapid progress

## 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
- Hierarchical collection built on Spack
- Products may belong to >1 SDK based on dependences
- Establish community policies for library development
- Apply Continuous Integration and other robust testing practices

Math SDK

Tools SDK

PM&RT SDK

DataViz SDK


### 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 Leadership Team: Decades of Software Experience**

- **Jim Willenbring – SDK Coordinator and Release Manager**
- **Sameer Shende – Programming Models & Runtimes**
- **Bart Miller – Development Tools**
- **Lois McInnes – Math Libraries**
- **Chuck Atkins - Data & Viz**

# SDK Summary

- Extending the SDK approach to all ECP ST domains.
- First concrete effort: Spack target to build all packages in an SDK.
  - Decide on good groupings.
  - Not necessarily trivial: Version compatibility issues.  Coordination of common dependencies.
- SDKs will help reduce complexity of delivery:
  - Hierarchical build targets.
  - Distribution of software integration responsibilities.
- Longer term:
  - Establish community policies, enhance best practices sharing.
  - Provide a mechanism for shared infrastructure, testing, training, etc.
  - Enable community expansion beyond ECP.

# HI and Facilities Interaction and Integration

- ECP ST has an entire technical area dedicated to HI interactions and integration:
  - ECP ST technical area 5: SW Ecosystem and Delivery.
  - Primary focus is on coordinated software delivery and integration.
- Delivery vs deployment:
  - ST *delivers.*
  - HI, Facilities, vendors, OpenHPC *deploy.*

# ST-HI Interplay

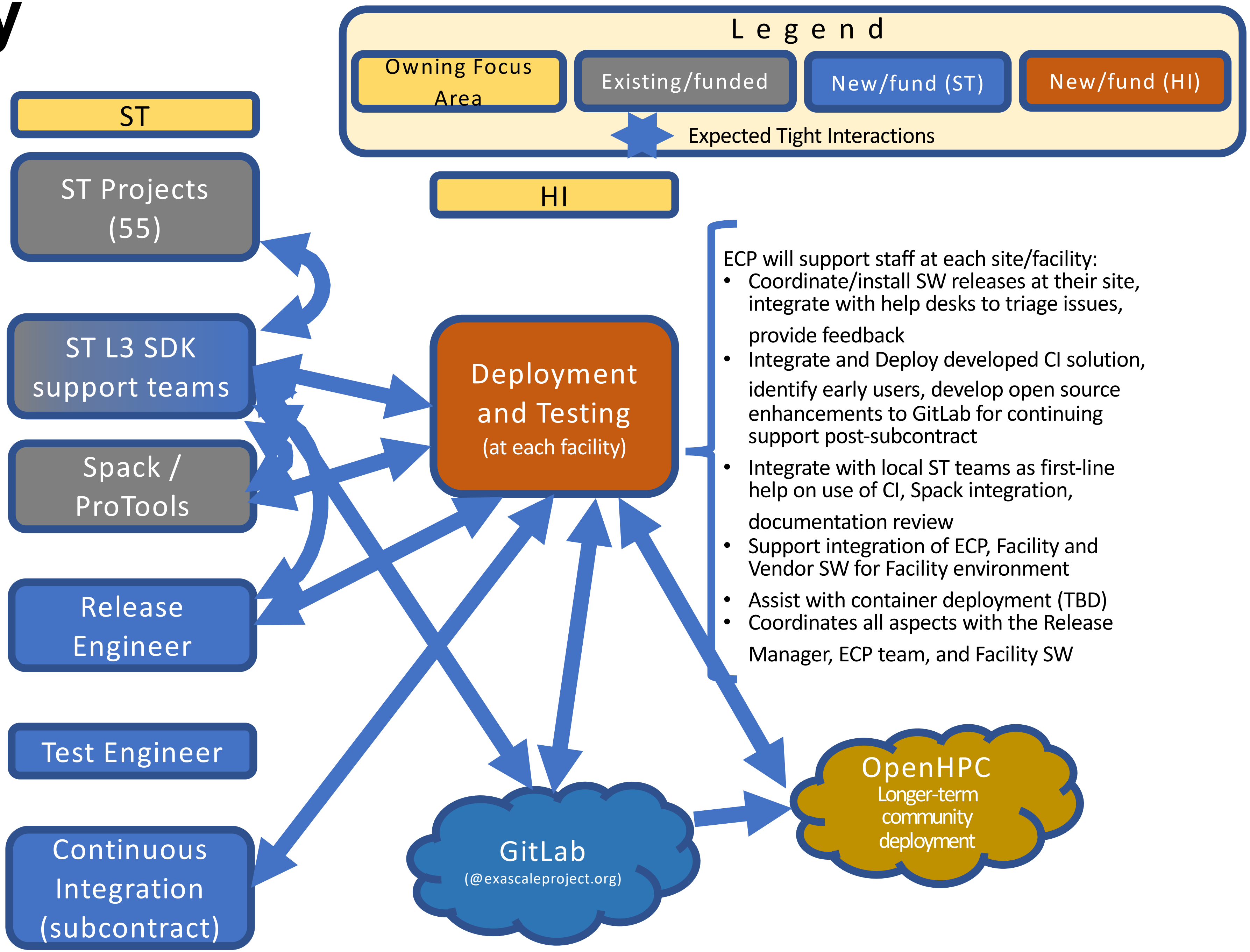
Modeled after xSDK and ECP-IDEAS, each L3 area defines and deploys one or more SDKs based on agreed-upon community policies for this area. Lead will work with ST projects on architecture, documentation, integration testing, and ties to other SDKs, AD, and HI

Existing Spack and ProTools teams at LLNL, supplemented by additional help from deployment team in HI (facilities)

Single POC manager for ST release(s). Identify/leverage commonality across SDKs, establish documentation and testing standards. Work with HI Software Deployment at Facilities team

Work closely with Release Engineer, Spack developers, CI team, and OpenHPC on developing a framework for integrated testing of ST releases

Supplement GitLab CI with features required by facilities to deploy CI as an option to ST projects and other users




ECP will support staff at each site/facility:

- Coordinate/install SW releases at their site, integrate with help desks to triage issues, provide feedback
- Integrate and Deploy developed CI solution, identify early users, develop open source enhancements to GitLab for continuing support post-subcontract
- Integrate with local ST teams as first-line help on use of CI, Spack integration, documentation review
- Support integration of ECP, Facility and Vendor SW for Facility environment
- Assist with container deployment (TBD)
- Coordinates all aspects with the Release Manager, ECP team, and Facility SW

# ECP ST Staff Contribute to ISO and *de facto* standards groups: Assuring Sustainability through standards

| Standards Effort | ECP ST Participants |
|---|---|
| MPI Forum | 15 |
| OpenMP | 15 |
| BLAS | 6 |
| C++ | 4 |
| Fortran | 4 |
| OpenACC | 3 |
| LLVM | 2 |
| PowerAPI | 1 |
| VTK ARB | 1 |

- **MPI/OpenMP:** Several key leadership positions.
- Heavy involvement in all aspects.
- **C++:** Getting HPC requirements considered, contributing working code.
- **Fortran:** Flang front end for LLVM.
- ***De facto:*** Specific HPC efforts.
- **ARB*:** Good model for SDKs.

*Architecture Review Board

# External Product Impact

| Product | Contribution |
|---|---|
| MAGMA | ECP ST math libraries eorts inform the design, implementation, and optimization of numerical linear algebra routines on NVIDIA GPUs |
| Vendor/community compilers and runtimes | The Validation and Verication Suite (on-going effort) for the SOLLVE project has helped uncover bugs in OpenMP implementations provided by Cray, LLVM and XL. |
| SWIG | The ECP ST ForTrilinos eorts contributes the capability to generate automatic Fortran bindings from C++ code. |
| TotalView Debugger | ECP ST sta are engaged in co-design of OMPD, the new debugging interface for OpenMP programs, along with RogueWave engineers. This eort helps RogueWave improve their main debugging product, TotalView, by making it aware and compatible with recent advances in OpenMP debugging. |
| MPI Forum | ECP ST sta maintain several chapters of the MPI Forum, effort that require a constant involvement with the other authors, as well as participation to the online discussions related to the chapter and regular attendance of the MPI Forum face-to-face activities. An ECP ST staff member belongs to several working group related to scalability and resilience where, in addition to the discussions, implements proof-of-concept features in OpenMPI. |
| Cray MPICH MPI-IO | As part of the ExaHDF5 ECP project, the ALCF worked with Cray MPI-IO developers to merge the upstream ROMIO code into the downstream proprietary Cray MPICH MPI-IO, leveraging Crays extensive suite of IO performance tests and further tuning the algorithm. Cray is currently targeting its deployment in an experimental release. |
| OpenHPC | An ECP ST staff member serves on the OpenHPC Technical Steering Committee as a Component Development representative. |
| LLVM | An ECP ST staff member is co-leading design discussions around the parallel IR and loop-optimization infrastructure. |

*Some of our best work is to provide input to software we don't productize or support.*

# ECP Software Technology Capability Assessment Report (Release July 2018)

- Three document elements:
  1. Executive summary – Public content.
  2. Project Description - Public content.
     - **SDKs, Delivery strategy, project restructuring, new projects.**
     - Technical areas overview.
     - **Deliverables: Products, Standards committees, contributions to external products.**
     - Project two-pages: 55 with description, activities, challenges, next steps.
  3. **Appendix – ECP/Stakeholder content.**
     - **Impact goals/metrics framework.**
     - **Gaps and Overlaps.**
     - **ASC-ASCR leverage tables.**
- LaTeX, separate contributors, easily updated.
- 212 pages (191 public), update twice a year.


EXASCALE COMPUTING PROJECT

ECP-RPT-ST-0001-2018

ECP Software Technology Capability Assessment Report

Michael A. Heroux, Director ECP ST
Jonathan Carter, Deputy Director ECP ST
Rajeev Thakur, Programming Models & Runtimes Lead
Jeffrey Vetter, Development Tools Lead
Lois Curfman McInnes, Mathematical Libraries Lead
James Ahrens, Data & Visualization Lead
J. Robert Neely, Software Ecosystem & Delivery Lead

June 26, 2018

Will be available
https://www.exascaleproject.org

# ECP Software and Arm

- Arm is not an explicit ECP target, so where are the connections?
- ECP mission need: ”… deliver a capable, sustainable Exascale ecosystem.”
- Exascale Arm feasible:
  - Astra: 2.3 PF on 2600 nodes.
  - Exascale possible with 430X speedup:
    - Combination node/core/VL increases.
    - Example (440X):
      - Nodes: 2600 → 250,000
      - Cores: 56 → 64
      - VL: 4 → 16
  - Arm is part of the ecosystem.
- ECP can provide value to Arm (and the other direction, too).
  - Most ECP-related products are or will be available on Arm platforms.
  - With modest additional attention and coordination, impact, sustainability can improve.

# Summary

- ECP Software Technology contributes to a broad spectrum of HPC software products:
  - 89 products total.
  - 33 broadly used in HPC.
    - Require substantial investment and transformation in preparation for exascale architectures
  - Additional 23 important to some existing applications,
    - Typically represent new capabilities that enable new usage models for realizing the potential that Exascale platforms promise.
  - Remaining products are in early development phases
    - Addressing emerging challenges and opportunities that Exascale platforms present.
  - Almost all ECP efforts will be directly transferrable to Arm platforms.
  - ECP does not explicitly target Arm at this time, but…
  - ECP mission includes a *capable, sustainable software ecosystem.* Must pay attention to Arm.