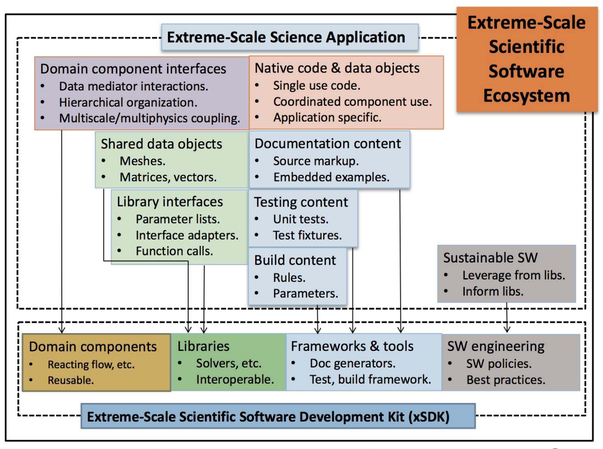
Extreme-scale Scientific Software Development Kit (xSDK) for ECP

xSDK Vision

https://xsdk.info: Building the foundation of a highly effective extreme-scale scientific software ecosystem.



*The vision of the xSDK is to provide infrastructure for and interoperability of a **collection of related and complementary software elements**—developed by diverse, independent teams throughout the high-performance computing (HPC) community—that provide the building blocks, tools, models, processes, and related artifacts for rapid and efficient development of high-quality applications.*

Motivation: Next-generation scientific simulations require combined use of independent packages

- Installing multiple independent software packages is tedious and error prone
  - Need consistency of compiler (+version, options), 3rd-party packages, etc.
  - Namespace and version conflicts make simultaneous build/link of packages difficult
- Multilayer interoperability among packages requires careful design and sustainable coordination

xSDK4ECP Project Overview

https://xsdk.info/ecp: Addressing xSDK challenges for exascale.

Project Scope

Enable seamless combined use of diverse, independently developed software packages as needed by ECP apps

- Coordinated use of on-node resources
- Integrated execution
- Coordinated & sustainable documentation, testing, packaging, distribution

Approach

- Develop **community policies** and **interoperability** layers among xSDK component packages
- Determine xSDK sustainability strategy for ECP
- Work with ECP applications to motivate and test xSDK

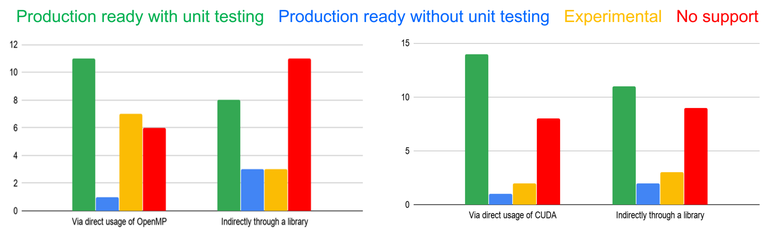
Document on node-level resource management activities within the ECP

Survey status and plans of on-node (and internode) parallel computing capabilities for individual xSDK libraries and prospective additions.

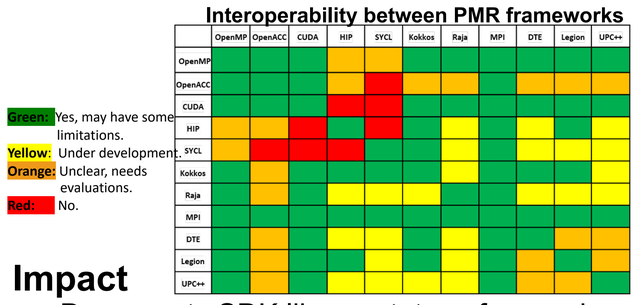
Status of CUDA and OpenMP capabilities of 25 math libraries

Figure 1: Does your library have OpenMP CPU multi-threading capabilities? What is the status?

Figure 3: Does your library have CUDA capabilities? What is the status?



Interoperability between PMR frameworks



Survey programming models and runtime libraries (PMR) projects on status and plans for node-level resource management and other issues that impact mathematical libraries.

Impact

- Document xSDK library status of on-node capabilities, new developments on node-level resource management and its potential impact on xSDK libraries.
- Inform greater ECP community.

Autotuning Parameter Selection for ECP Applications

Objective

Develop autotuning software that learns optimal parameter selection.

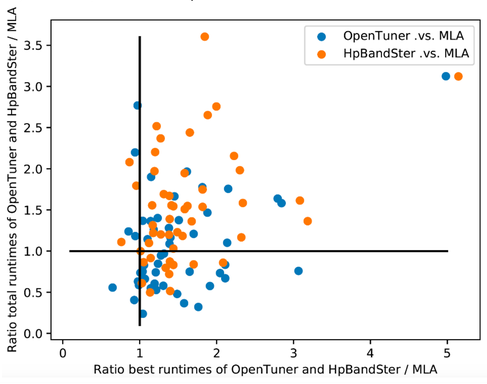
- Enable ECP math libraries and apps to run efficiently on exascale machines.
- New autotuning methods will advance the state of the art in performance optimization research.

Accomplishments

GPTune: https://github.com/xiaoyeli/GPTune

- **Easy integration** - Easy to use Python interface
- **Collaborated** with Y-TUNE team - Developed a ***common interface*** so users can access both tuners and others easily in the same code.
- **Applied** GPTune for parameter search for ScaLAPACK QR & SVD, hypre, and SuperLU

- 128 nodes Edison, 50 tasks (1< m=n < 20000), 4 parameters (mb, nb, #MPIs, #Threads)



GPTune finds **better parameters** in 42 (84%) and 47 (94%) cases compared to OpenTuner and HyBandSter, resp., for ScaLAPACK QR.

xSDK Community Policies

https://xsdk.info/policies: Addressing challenges in interoperability & sustainability of software developed by diverse, independent teams.

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 common platforms.
- **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 OSI-approved, permissive 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.

We welcome feedback. What policies make sense for your software?

Also specify recommended policies: Currently 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.
- **R6.** Each package should document the versions of packages with which it can work and on which it depends.
- **R7**. Have README, SUPPORT, LICENSE, and CHANGELOG files in top directory.

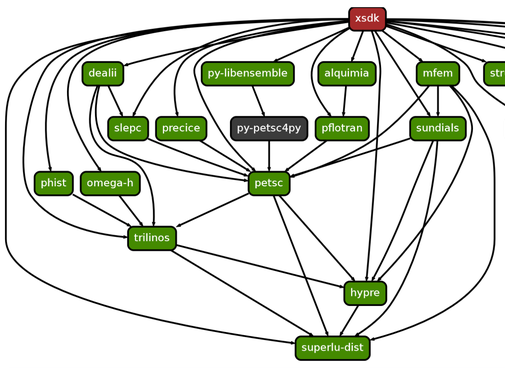
Full version of community policies available on Github: https://github.com/xsdk-project/xsdk-community-policies

New or revised polices can now be proposed with pull requests on Github (https://github.com/xsdk-project/xsdk-community-policies/blob/master/process-for-new-or-revised-policies.md).

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.

xSDK Packages and Releases

https://xsdk.info/packages: Enabling ECP applications to readily access many of the most popular HPC math libraries.



xSDK-0.5.0
released Nov 2019

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

Original xSDK math libraries: **hypre, PETSc, SuperLU, Trilinos**

Added Dec 2017: **MAGMA, MFEM, SUNDIALS**

Added Dec 2018: **AMReX, deal.II, DTK, Omega_h, PHIST, PLASMA, PUMI, SLEPc, STRUMPACK, TASMANIAN**

Added Nov 2019: **ButterflyPACK, Ginkgo, libEnsemble, preCICE**

Impact

- Improve access & sustainability of math libraries for ECP
- Lay groundwork for addressing broader issues in software interoperability and performance portability

Spack/Git Workflow

- **Packages**
  - Follow the standard workflow for a Spack package
  - Submit pull requests with the "xSDK" label
  - Provide package candidate and final xSDK release tags
- **xSDK Meta-package**
  - Depends on xSDK member packages: "**spack install xsdk**"
  - Maintain shared Spack branch for release coordination

History and Plans

- Began in ASCR/BER partnership, IDEAS project (Sept 2014), needed for BER multiscale, multiphysics surface-subsurface hydrology models
- Work toward regular xSDK releases with increased formality of release process.
- Collaboration with broader SDK efforts in ECP

ECP 4th Annual Meeting, February 2020

This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-POST-801849