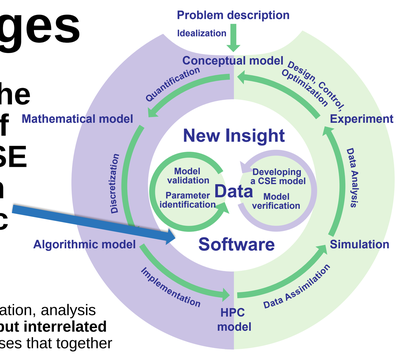
Progress in CSE Software Ecosystems

Lois Curfman McInnes (ANL), Mike Heroux (SNL), Daniel S. Katz (UIUC), Scott Lathrop (UIUC), Jim Willenbring (SNL), Ulrike Meier Yang (LLNL), and the xSDK Community

Challenges

Software is the foundation of sustained CSE collaboration and scientific progress.



CSE cycle: Modeling, simulation, analysis
- **Software: independent but interrelated elements** for various phases that together enable CSE

Ref: **Research and Education in Computational Science and Engineering, *SIAM Review*, 60(3), Dec 2018, https://doi.org/10.1137/16M1096840**

Trends and challenges for CSE software

- **Fundamental trends**
  - Disruptive hardware changes
    - Require algorithm/code refactoring
  - Need coupling, optimization, sensitivities
    - Multiphysics, multiscale, data analytics
- **Challenges**
  - Need refactoring: Really, continuous change
  - Modest funding for app development: No monolithic apps
  - Requirements are unfolding, evolving, not fully known *a priori*

Community software ecosystems require high-quality software.

- **Ecosystem:** A group of independent but interrelated elements comprising a unified whole
  - Diversity is essential for an ecosystem to thrive
  - The whole is greater than the sum of its parts
- **Complex, intertwined challenges**
- **Need community efforts to**
  - Improve software quality
  - Change research culture
  - Promote collaboration

12 scientific software challenges

- Incentives, citation/credit models, and metrics
- Career paths
- Training and education
- Software engineering
- Portability
- Intellectual property
- Publication and peer review
- Software communities and sociology
- Sustainability and funding models
- Software dissemination, catalogs, search, and review
- Multi-disciplinary science
- Reproducibility

All are tied together

Ref: Daniel S. Katz, *Software in Research: Underappreciated and Underrewarded*, 2017 eResearch Australasia conference, Oct 2017, https://doi.org/10.6084/m9.figshare.5518933

Resources and collaborating communities … Get involved!

Collaborative community software ecosystems help improve CSE productivity and sustainability.

BLUE WATERS SUSTAINED PETASCALE COMPUTING

Webinar track: Scientific Software Ecosystems
https://bluewaters.ncsa.illinois.edu/webinars/software-ecosystems

Objectives: Through considering issues in scientific software ecosystems:
- Promote quality **reusable research software** for computational and data-enabled discovery
- Promote community efforts to improve research software quality, culture, credit, collaboration, …

March/April 2019

IEEE
AIP cise.aip.org
IEEE computer society www.computer.org/cise

Theme: Accelerating Scientific Discovery with Reusable Software

Goals:
- Recognize the significant increase in the development and usage of high-quality reusable software that has taken place over the past decade.
- Provide information about the benefits that can be achieved from developing and utilizing reusable software that follows standards of quality and good practices.

Papers: March/April 2019
https://publications.computer.org/cise
- *Community organizations: Changing the culture in which research software is developed and sustained* (Katz et al.)
- *The role of scientific communities in creating reusable software: Lessons from geophysics* (Kellogg et al.)
- *A community of practice around peer-review for long-term research software sustainability* (Ram et al.)
- *Developing a computational chemistry framework for the exascale era* (Richard et al.)
- *How to professionally develop reusable scientific software – and when not to* (Adorf et al.)
- *Fostering reuse in scientific computing with embedded components* (Lanore)

- **Opportunities**
  - Better design, software practices, and tools are available
  - Better software architectures: toolkits, libraries, frameworks
  - Open-source software, community collaboration
- **What makes sense for <u>your</u> software?**
  - Consider resources for improving software quality
  - What community software ecosystems do you want to use and be a part of?

- **URSSI:** Conceptualization of a US Research Software Sustainability Institute: **http://urssi.us**
  - Creating a roadmap for a US software institute to overcome pervasive concerns in research software
- **IDEAS Productivity: https://ideas-productivity.org**
  - Partnering with the community to increase software productivity and sustainability as key aspects of increasing overall scientific productivity
- **Better Scientific Software: https://bssw.io**
  - Central hub for sharing info on practices, techniques, and experiences to improve software productivity and sustainability
- **WSSSPE: http://wssspe.researchcomputing.org.uk**
  - International community-driven organization that promotes sustainable research software
- **NumFOCUS: https://www.numfocus.org**
  - Umbrella nonprofit that supports and promotes open-source scientific computing
- **Research Software Alliance (ReSA): http://researchsoft.org**
  - International organization bringing communities together to collaborate on improving the social and technical environment for research software
- **Software Carpentry: http://software-carpentry.org**
  - Volunteer non-profit organization dedicated to teaching basic computing skills to researchers
  - Lessons: https://software-carpentry.org/lessons
- **Software Sustainability Institute: http://www.software.ac.uk**
  - Institute to support UK's research software community: cultivating better, more sustainable, research software to enable world-class research
  - Guides: https://www.software.ac.uk/resources/guides-everything
- **Computational Science Stack Exchange: SciComp.StackExchange.com**
  - Q&A site for scientists using computers to solve scientific problems
- **IEEE TCHPC Software Engineering Practices Initiative: http://tc.computer.org/tchpc/home-page/software-engineering-practices**
  - Leading a conversation on impactful software engineering practices for HPC

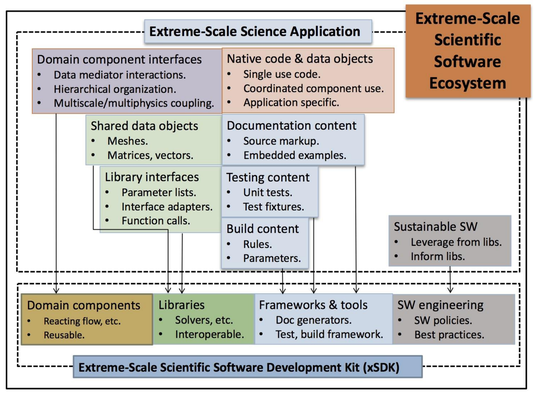
CSE19 Poster: DOI: 10.6084/m9.figshare.7755146.v2

Building the foundation of a highly effective extreme-scale scientific software ecosystem

xSDK **https://xsdk.info**

xSDK Vision and Impact

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.



Impact: Improve access & sustainability of numerical libraries for CSE
- Prior to xSDK, could not build required libraries for multiphysics and multiscale apps into a single executable due to many incompatibilities
- Lay groundwork for addressing broader issues in software interoperability and performance
- Provide ECP SDKs with an example release process

xSDK Community Policies

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

<u>xSDK compatible package</u>: Must satisfy mandatory xSDK policies:

We welcome feedback. What policies make sense for <u>your</u> software?

- **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 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: 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.** Document the versions of packages with which it can work or on which it depends.

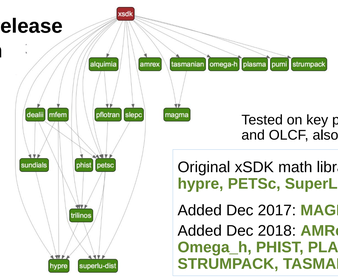
<u>xSDK member package</u>: Must be an xSDK-compatible package, *and* it uses or can be used by another package in the xSDK, and the connecting interface is regularly tested for regressions.

xSDK History and Releases

- Began in ASCR/BER partnership, IDEAS project (Sept 2014)
  - Needed for BER multiscale integrated surface-subsurface hydrology models
- Expanding in DOE Exascale Computing Project (ECP)
  - Many ECP teams rely on math libraries, often in combination
  - ECP teams need sustainable coordinated xSDK releases and increasing interoperability among xSDK packages to achieve good performance and easily access advanced algorithms and data structures

Astro, NWChemEx, WDMAPP, ExaFel, GAMESS, ExaSCD, Subsurface, EXAALT, WarpX, ExaAM, MFIX-Exa, CoPA, AMREX, CEED, CODAR, ATDM (LANL, LLNL, SNL) apps

xSDK-0.4.0 release Spack graph Dec 2018



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

Spack **https://spack.io**

Original xSDK math libraries: April 2016 **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**

Spack/Git Workflow

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

Join the xSDK Community!

We are actively soliciting contributions to the xSDK and feedback on xSDK community policies. See https://xsdk.info/faq

Extreme-Scale Scientific Software Stack

E4S **https://e4s.io**

- **E4S:** Coordinated and interoperable distribution of Exascale Computing Project (ECP) Software Technologies (ST) and related software. While porting of individual scientific software products is challenging, achieving interoperability between packages is even more difficult. E4S uses a dual-pronged approach for achieving software interoperability: Spack and SDKs.
- **Spack:** E4S uses the Spack package manager for software delivery. Spack provides the ability to specify versions of software packages that are and are not interoperable. It is also a common build layer not only to E4S software, but also to an enormous number of software packages outside of ECP ST. These features support achieving and maintaining interoperability between ST software packages.
- **Software Development Kits (SDKs):** An ECP ST Software Development Kit is a collection of related ECP ST 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.

EXASCALE COMPUTING PROJECT

SIAM CSE19, February 2019
Poster in the Minisymposterium on *Software Productivity and Sustainability for CSE and Data Science*

Argonne NATIONAL LABORATORY · BERKELEY LAB · Lawrence Livermore National Laboratory · Sandia National Laboratories · Berkeley UNIVERSITY OF CALIFORNIA · THE UNIVERSITY OF TENNESSEE KNOXVILLE · NCSA · ILLINOIS

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.