Progress in CSE Software Ecosystems

Lois Curtinman 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 intertwined elements for variable phases that together enable CSE


Trends and challenges for CSE software

- Fundamental trends
  - Disruptive hardware changes
  - Require algorithm/code retooling
  - Need coupling, optimization, sensitivities
  - Multiphysics, multiscale, data analytics

- Challenges
  - Need retooling: 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 intertwined 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

Resources and collaborating communities ... Get involved!

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,…

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

xSDK Vision and Impact

The vision of the xSDK is to provide infrastructure 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.

https://xSDK.info

xSDK Community Policies
https://xSDK.info/policies: Address challenges in interoperability & sustainability of software development by diverse, independent teams

xSDK History and Releases

- Began in ASC|BERE partnership, IDEAS project (Sept 2014)
- Leaders of xSDK indicate integrated research software sustainability models
- Expanding in DOE Exascale Computing Project (ECP)
  - Many ECP teams rely on xSDK libraries, aim to standardize

- Types of software ecosystems:
  - Research
  - Commercial
  - Intermediate

xSDK 0.4.0 release
Spack graph
Dec 2018

xSDK compatible package: Must satisfy mandatory xSDK policies:

- M1. Support xSDK community GNU Autotools or CMake options
- M2. Provide a comprehensive test suite
- M3. Employ user-provided API communicators
- 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. Portable using 64 bit pointers, 32 bit is optional
- M15. xSDK `make check` should be sustained
- M16. `make check` should be used for xSDK installed tool and package.

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

xSDK History and Releases

- Began in ASC|BERE partnership, IDEAS project (Sept 2014)
- Leaders of xSDK indicate integrated research software sustainability models
- Expanding in DOE Exascale Computing Project (ECP)

- xSDK 0.4.0 release
  - Spack graph

Join the xSDK Community!

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

极端规模软件栈

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 & 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.

Extreme Scale Scientific Software Stack

https://e4s.io

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

© 2019 American Mathematical Society. Released under a Creative Commons Attribution-NonCommercial-NoDerivs 4.0 License.