Making Reproducibility Indispensable: Changing the Incentives that Drive Computational Science

Presented at
Rescue-HPC 2018

SC18, Dallas, TX

Michael A. Heroux
Senior Scientist, Sandia National Laboratories
Scientist in Residence, St. John's University, MN

EXASCALE COMPUTING PROJECT

National Nuclear Security Administration

U.S. DEPARTMENT OF ENERGY | Office of Science

IDEAS productivity

My Background

- 1998 – now: Staff member at Sandia National Labs
  - Lead these projects:
    - ECP SW Technology since Nov 2017.
    - Trilinos: collection of scientific libraries – trilinos.github.io.
    - Mantevo: “Miniapps” project for HPC co-design – mantevo.github.io
    - IDEAS Productivity: Scientific Productivity and sustainability – ideas-productivity.org
    - HPCG Benchmark: Complementary benchmark for Top 500 – hpcg-benchmark.org
    - Better Scientific Software: Portal and content for productivity and sustainability – bssw.io
    - SC18 Reproducibility Chair, SC19 special role
  - Concurrent: Scientist in Residence, St. John’s University, MN USA
- 1988 – 1998: Staff member at Cray Research
  - 88 – 93: Math libraries developer, sparse solvers, LAPACK, BLAS: LIBSCI
  - 93 – 95: Application analyst, computational engineering group: FIDAP, Fluent, Star-CD.
  - 95 – 98: Scalable systems applications specialist: Cray T3E “MPP”.

Outline

- Increasing focus on reproducibility.
- Reproducibility dynamics.
- Publications.
- Software quality.
- Community.
- Personal Productivity Commitment.
- Reproducibility as a Keystone Habit.

Workflows

Other talks in this workshop.

Essential to reproducibility.

Reproducibility is essential

SCIENCE

Many Psychology Findings Not as Strong as Claimed

By BENEDICT CAREY AUG. 27, 2015

Staff of the the Reproducibility Project at the Center for Open Science in Charlottesville, Va., from left: Mallory Kidwell, Courtney Soderberg, Johanna Cohoon and Brian Nosek. Dr. Nosek and his team led an attempt to replicate the findings of 100 social science studies. Andrew Shurtleff for The New York Times

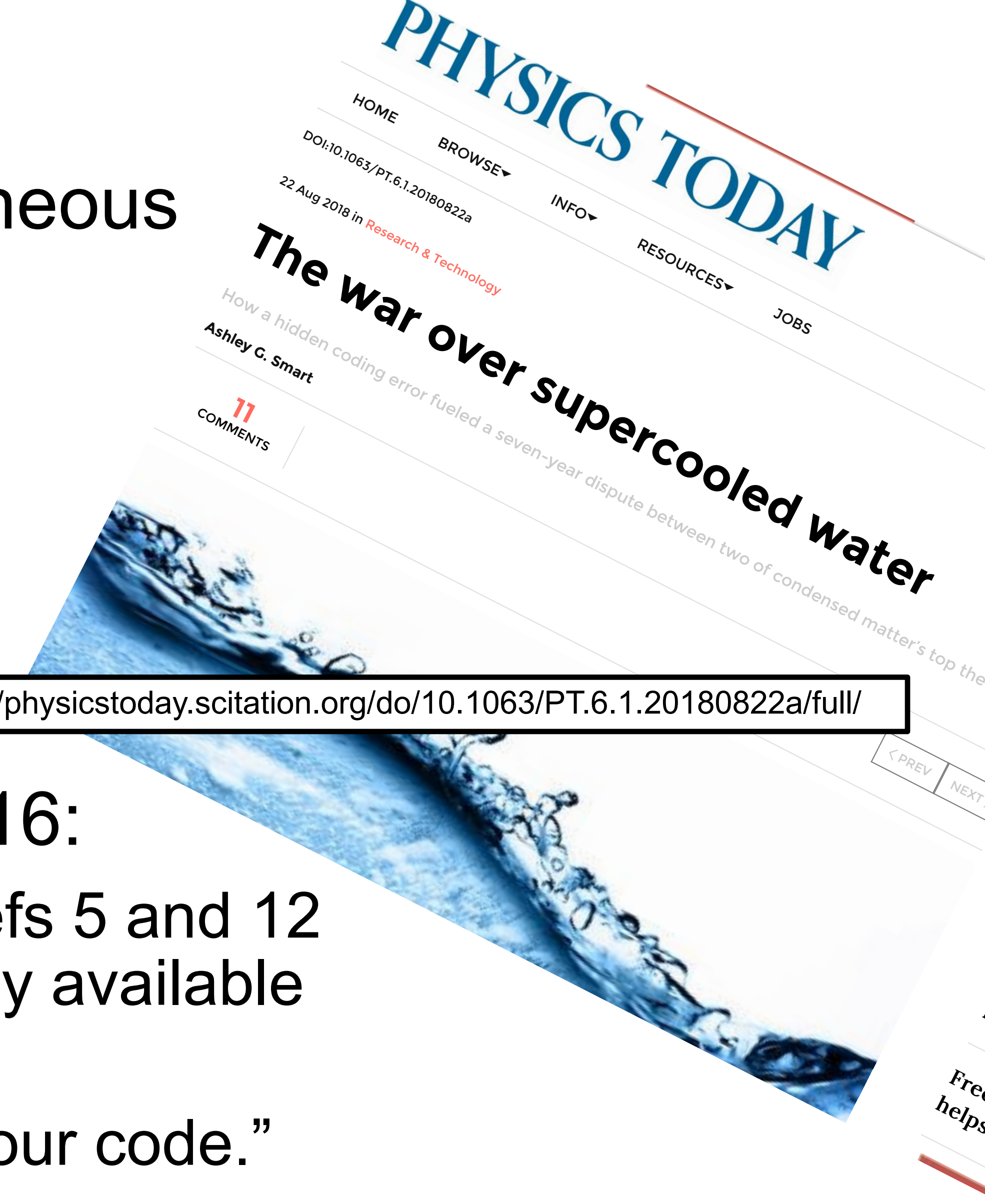
Reproducibility

- NY Times highlights “problems”.
- Only one of many cited examples.
- Computational science ***had*** been spared this “spotlight”.

http://www.nytimes.com/2015/08/28/science/many-social-science-findings-not-as-strong-as-claimed-study-says.html?_r=0

Computational Science Example

- Behavior of pure water just above homogeneous nucleation temperature (~ - 40 C/F).
- Debenedetti/Princeton (2009):
  - 2 possible phases: High or low density.
- Chandler/Berkeley (2011):
  - Only 1 phase: High density.
- No sharing of details across teams until 2016:
  - Chandler in Nature: “LAMMPS codes used in refs 5 and 12 are standard and documented, with scripts freely available upon request.”
  - Debenedetti with colleague Palmer: ”Send us your code.”
  - Received code after requests and appeal to Nature.



Source: https://physicstoday.scitation.org/do/10.1063/PT.6.1.20180822a/full/

Computational Science Example

- Palmer located bug/feature in Berkeley code.
- Used to speed up LAMMPS execution.
- Replaced with more standard approach.
- Obtained result similar to Debenedetti 2009.
- Resolution took 7 years.

Source: https://physicstoday.scitation.org/do/10.1063/PT.6.1.20180822a/full/

For Palmer, the ordeal exemplifies the importance of transparency in scientific research, an issue that has recently drawn heightened attention in the science community. "One of the real travesties," he says, is that "there's no way you could have reproduced [the Berkeley team's] algorithm—the way they had implemented their code—from reading their paper." Presumably, he adds, "if this had been disclosed, this saga might not have gone on for seven years."

Better Productivity and Sustainability

Essential for affordable reproducibility

Tradeoffs: Better, faster, cheaper

- "Better, faster, cheaper: Pick two of the three."
  - Scenario: (Today)
    You are behind in developing a sophisticated new model in your software that you want to use for results in an upcoming paper.
  - Which of these could be reasonable choices?
    - Develop a simpler model for the paper.
    - Set other work aside and spend more time on development.
    - Ask for an extension on the paper deadline.
    - Develop sophisticated model, but don't test its correctness.
    - Develop sophisticated model, but don't document it or check it in.

Improved developer productivity

"Better, faster, cheaper: Pick all three." – Near term.

Scenario: (6 months later)
After investing in **developer productivity improvements**, you are on time in developing a sophisticated new model in your software that you want to use for results in an upcoming paper.

Invest in developer tools, processes, practices.

Improved software sustainability

“Better, faster, cheaper: Pick all three.” – Long term.

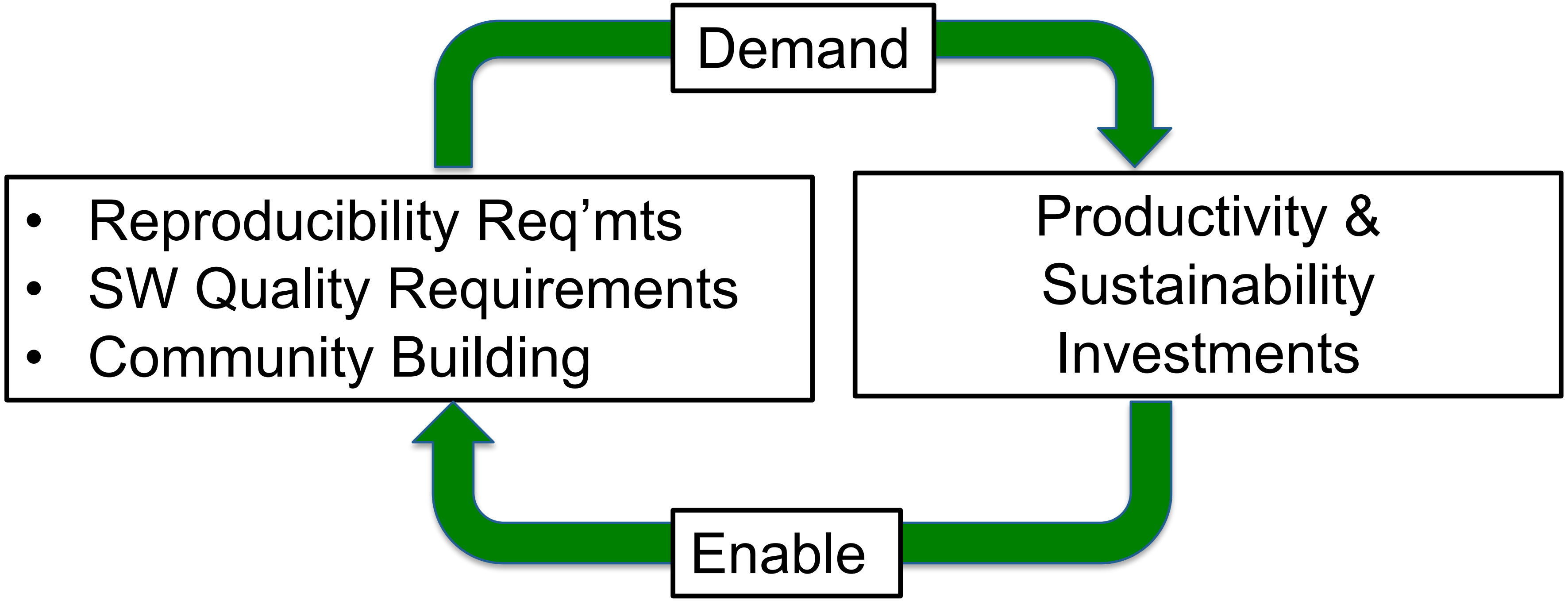
Scenario: (3 years later)
After investing in **software sustainability improvements**, you are on time in developing **several** sophisticated new models in your software that you want to use for results in upcoming papers.

Invest in testing, documentation, integration for long-term software usability.

Which of These Enhance Reproducibility?

- Code written by first-year, untrained grad student.
- Tuning for high performance.
- Dynamic parallelism of modern processors.
- Better software testing.
- Source code and versioning management.
- Investing in developer productivity.
- Investing in software sustainability.

Incentives Demand Investments, Enabled by Investments



Common statement: "I would love to do a better job, but I need to:

- Get this paper submitted.
- Complete this project task.
- Do something my employer values more.

Goal: Change incentives to include value of better software, better science.

Reproducible vs Replicable

Addressing Confusion in Taxonomies

SANDIA REPORT

SAND2018-11186
Unlimited Release
Printed October 2018

Toward a Compatible Reproducibility Taxonomy for Computational and Computing Sciences

Michael A. Heroux, Lorena A. Barba, Manish Parashar, Victoria Stodden and Michela Taufer

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Table 1: Definitions of Reproducible and Replicable

Table 1: Claerbout/Donoho/Peng (Claerbout) and ACM definitions of Reproducible and Replicable. Claerbout definitions are prevalent in the computational science literature and have been used since the 1990s. The ACM definitions are used by ACM in its Artifact Review and Badging effort and first appeared in February 2013.

Term	Claerbout	ACM
Reproducible	Authors provide all the necessary data and the computer codes to run the analysis again, re-creating the results.	(Different team, different experimental setup.) The measurement can be obtained with stated precision by a different team, a different measuring system, in a different location on multiple trials. For computational experiments, this means that an independent group can obtain the same result using artifacts which they develop completely independently.
Replicable	A new study arrives at the same scientific findings as a previous study, collecting new data (with the same or different methods) and completes new analyses.	(Different team, same experimental setup.) The measurement can be obtained with stated precision by a different team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same or a different location on multiple trials. For computational experiments, this means that an independent group can obtain the same result using the author's own artifacts.

Publications

ACM TOMS Replicated Computational Results (RCR)

- Submission: Optional RCR option.
- Standard reviewer assignment: Nothing changes.
- RCR reviewer assignment:
  - Concurrent with standard reviews.
  - As early as possible in review process.
  - Known to and works with authors during the RCR process.
- RCR process:
  - Multi-faceted approach, Bottom line: Trust the reviewer.
- Publication:
  - Replicated Computational Results Designation.
  - The RCR referee acknowledged.
  - Review report appears with published manuscript.

RCR Process: Two Basic Approaches

1. Independent replication (3 options):
    A. Transfer of, or pointer to, author's software.
    B. Guest account, access to author's software.
    C. Observation of authors replicating results.

Or (Not used with TOMS, but with SC)

2. Review of computational results artifacts:
    - Results may be from an unavailable system.
    - Leadership class computing system.
    - In this situation:
        - Careful documentation of the process.
        - Software should have its own substantial V&V process.

TOMS:

- First RCR paper in TOMS issue 41:3
    - Editorial introduction.
    - van Zee & van de Geijn, BLIS paper.
    - Referee report.
- Second: TOMS 42:1
    - Hogg & Scott.
- Third: TOMS 42:4.
- More in the meantime.

TOMACS

- Similar.

Big Picture of ACM RCR

- Improve science.
  - Quality of prose: Good.
  - Quality of data: Poor.
- So bad now:
  - Trust comes from seeing a “cloud” of similar papers with similar results.
  - Which could still be wrong (built on a common bad piece).
  - Replicability: First step toward improvement.
- Engage a “dark portion” of the R&D community.
  - Reviewers not among typical reviewer pool.
  - Practitioners, users. Expert at use of Math SW.

Thank you for taking the time to consider our paper for your journal.

XXX has agreed to undergo the RCR process should the paper proceed far enough in the review process to qualify. ***To make this easier we have preserved the exact copy of the code used for the results (including additional code for generating detailed statistics that is not in the library version of the code).***

SC18 Reproducibility Initiative

- Two appendices:
  - Artifact description (AD).
    - Blue print for setting up your computational experiment.
    - Makes it easier to rerun computations in future.
    - AD appendix will be mandatory for SC19 paper submissions.
  - Artifact Evaluation (AE).
    - Targets "boutique" environments.
    - Improves trustworthiness when re-running hard, impossible.
- Details:
  - https://collegeville.github.io/sc-reproducibility/

Reproducibility and Supercomputing

Scenario:
You compute a “hero” calculation using 5M core-hours on Mira and submit your results for publication. During the review process, a referee questions the validity of your results. What options are feasible:

- The reviewer re-runs your code on a laptop or cluster.
- The reviewer re-runs your code on Mira.
- You re-run your code on Mira.
- Your results are rejected.
- Your results are accepted, but with risk.

Example: HPCG Benchmark

- Exploit two properties:
  - Spectral properties of CG:
    - Eigenvalue clustering.
    - CG convergence related to number of *distinct* eigenvalues.
  - Operator symmetry:
    - Compact Finite Difference operator is symmetric.
    - Multigrid is symmetric.

Example: HPCG Benchmark

- Symmetry:
  - For any linear operator A, $x^TAy = y^TA^Tx$.
  - If A symmetric $A = A^T$, so $x^TAy = y^TAx$.
  - And $\boldsymbol{x^TAy - y^TAx = 0}$.
- HPCG computes the above expression for:
  - User matrix and the preconditioner.
  - Numerical detail: Need to scale by vector & matrix norms.

Example: HPCG Benchmark

- Eigenvalue clustering:
  - HPCG matrix is 27-point finite difference stencil.
    - -1 off diagonals, diagonally dominant, zero Dirichlet BCs.
    - Max diagonal value – 27.
  - Idea: Temporarily replace diagonal values.
    - For i=1:9 A(i,i) = (i+1)*1.0E6
    - For i>9 A(i,i) = 1.0E6
- Questions:
  - How many distinct diagonal values?
  - How many unpreconditioned CG iterations?
  - How many preconditioned CG iterations?

Sources for Artifact Evalution metrics

- Synthetic operators with known:
  - Spectrum (Huge diagonals).
  - Rank (by constructions).
- Invariant subspaces:
  - Example: Positional/rotational invariance (structures).
- Conservation principles:
  - Example: Flux through a finite volume.
- General:
  - Pre-conditions, post-conditions, invariants.

Can you think of something for your problems?

Reproducibility and Publications

- These conferences expect artifact evaluation appendices (most optionally):
  - CGO, PPoPP, PACT, RTSS and SC.
  - http://fursin.net/reproducibility.html
- ACM Replicated Computational Results (RCR).
  - ACM TOMS, TOMACS.
  - http://toms.acm.org/replicated-computational-results.cfm
- ACM Badging.
  - https://www.acm.org/publications/policies/artifact-review-badging
  - Used with SC technical program.

Software Quality

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

EXASCALE COMPUTING PROJECT

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

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 ST Products: 55 Projects contribute to 89 Unique Products

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

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+

Source Build System	
CMake	44
Configure/Make (autotools)	32
Custom	4

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

Stats collected April 2018

- 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.**

NNSA National Nuclear Security Administration
U.S. DEPARTMENT OF ENERGY | Office of Science
ECP EXASCALE COMPUTING PROJECT

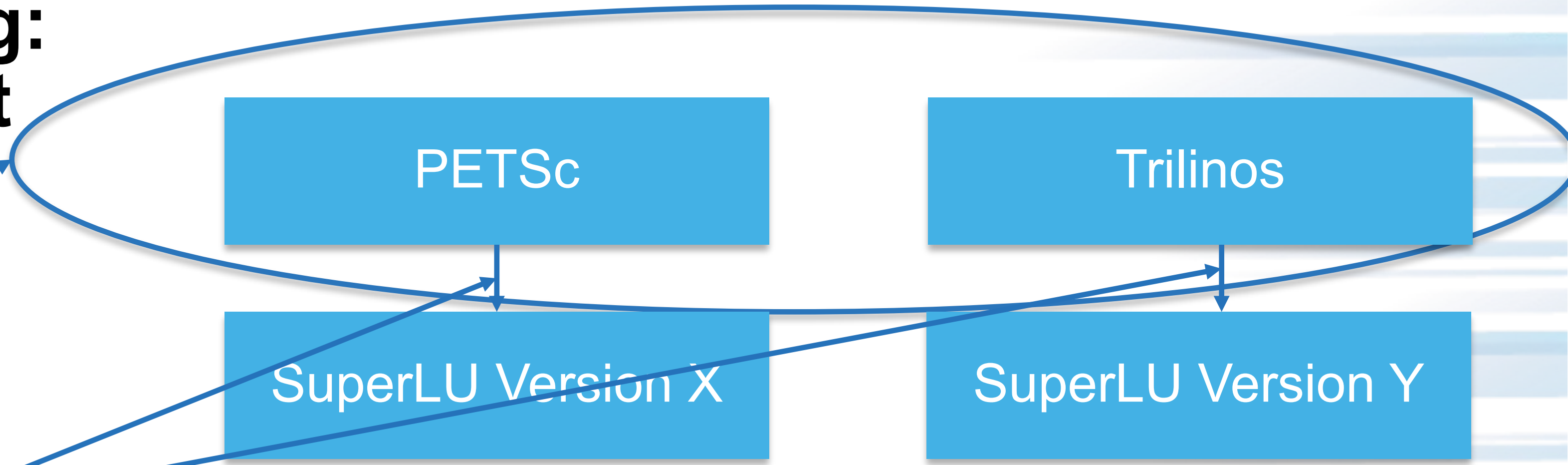
SW Development Kit (SDK) Overview

- 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.

SDKs are a key delivery vehicle for ECP

- 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
- 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

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 the long vertical dependency chain sense
- 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 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 in FY19 Q2, continually refine

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 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.

...

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.

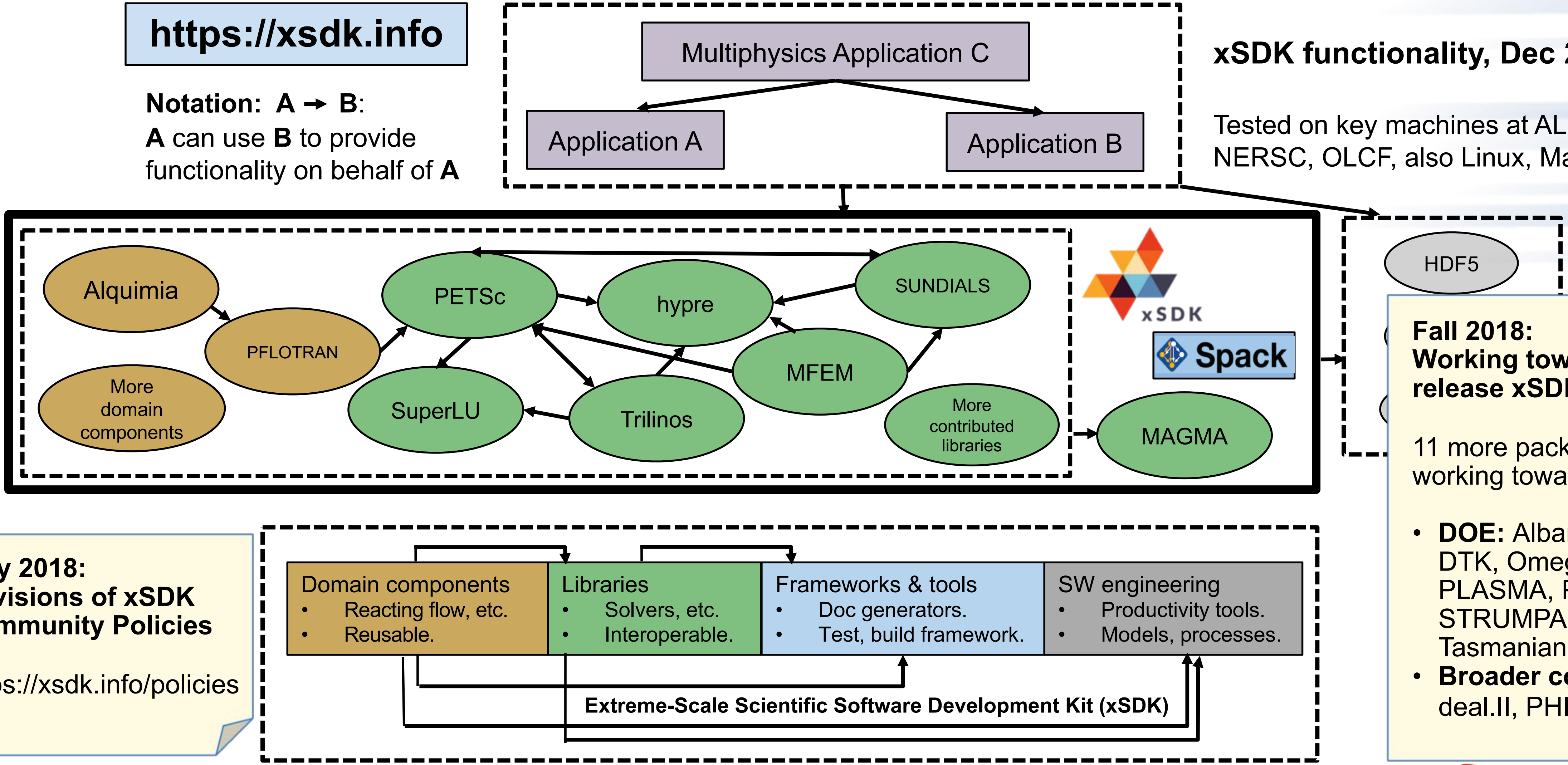
xSDK-0.3.0: Dec 2017 … Now working toward xSDK-0.4.0, FY19-Q1

https://xsdk.info

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

xSDK functionality, Dec 2017

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



July 2018:
Revisions of xSDK Community Policies

https://xsdk.info/policies

Fall 2018:
Working toward release xSDK-0.4.0

11 more packages working toward inclusion:

- **DOE:** Albany, AMReX, DTK, Omega_h, PLASMA, PUMI, STRUMPACK, Tasmanian
- **Broader community:** deal.II, PHIST, SLEPc

EXASCALE COMPUTING PROJECT

SDK Definition Scenario

xSDK (16)	PMR Core (14)	Perf Tool and Technology (7)	Compilers & Support (6)	Memory Hierarchy (5)	Ecosystem (12)	Analysis, Instrumentation, and Generation (7)	Data & Vis Analysis and Reduction (9)	I/O Services and Checkpoint Restart (12)
hypre	Legion	TAU	LLVM OpenMP compiler	SICM	mpiFileUtils	Rose	ParaView	FAODEL
FleSCI	Kokkos / Support	HPCToolkit	OpenMP V & V	Umpire	Intel GEOPM	Exascale Code Generation Tool	Catalyst	ROMIO
MFEM	Global Arrays	Dynist Binary Tools	openarc	Papyrus	BEE	Program Database Toolkit	VTK-m	Mercury (part of growing Mochi suite)
Kokkoskernels	RAJA	Gotcha	Kitsune	AML	FSEFI	CHiLL Autotuning Compiler	SZ	HDF5
Trilinos	CHAI	Caliper	LLVM	Argo Containers	Secure JupyterHub	Search using Random Forests	zfp	Parallel netCDF
SUNDIALS	PaRSEC	PAPI	Flang/LLVM Fortran compiler		Kitten Light-weight Kernel	Siboka	VisIt	ADIOS
PETSc/TAO	DARMA	Sonar			COOLR	C2C	ASCENT	Darshan
libEnsemble	GASNet-EX		**Key**		NRM		Cinema	UnifyCR
STRUMPACK	Qthreads		PMR		TriBITS		ROVER	VeloC
SuperLU	BOLT		Tools		Spack			IOSS
ForTrilinos	UPC++		Math Libraries		MarFS			HXHIM
SLATE	MPICH		Data and Vis		GUFI			SCR
MAGMA	Open MPI		Ecosystems & Delivery					
DTK	QUO		Q1 release participant					
Tasmanian			Facility Support					
TuckerMPI			**Support undecided**					

Spack

A flexible package manager for HPC

https://spack.io github.com/spack @spackpm

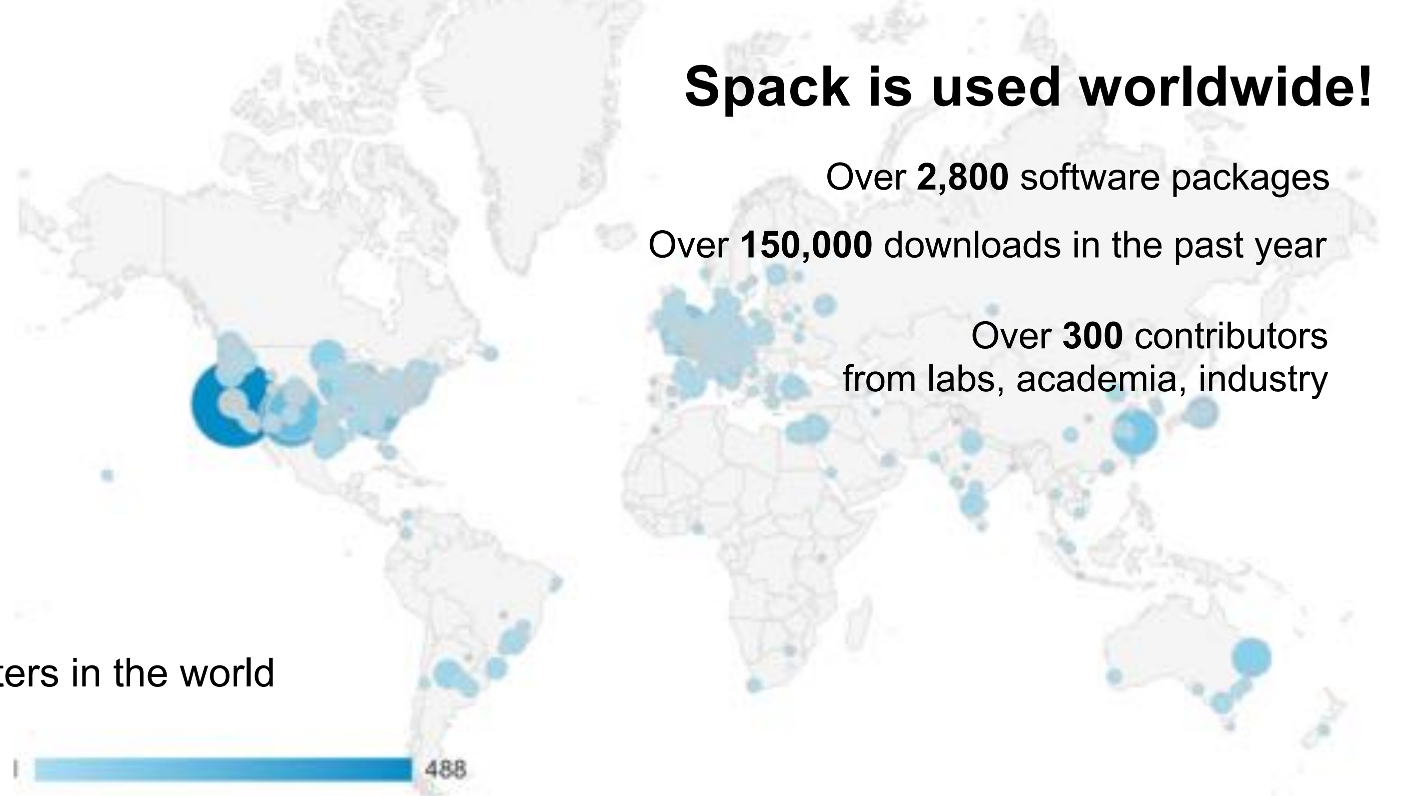
- Inspired by Homebrew, Nix, some others
- Support scientific stacks with multiple languages
- Flexibility:
  - Build packages many different ways
  - Change compilers and flags in builds
  - Swap implementations of libraries (MPI, BLAS, etc.)
- Run on laptops, Linux clusters, and the largest supercomputers in the world

Spack is used worldwide!

Over **2,800** software packages

Over **150,000** downloads in the past year

Over **300** contributors
from labs, academia, industry

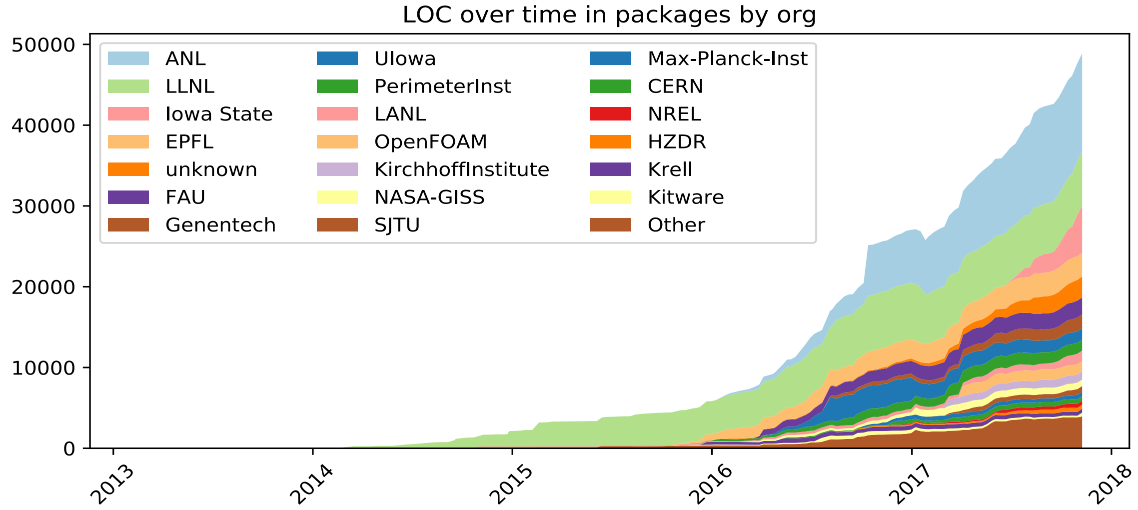


Easy installation

```
$ git clone https://github.com/spack/spack
$ . spack/share/spack/setup-env.sh
$ spack install hdf5
```

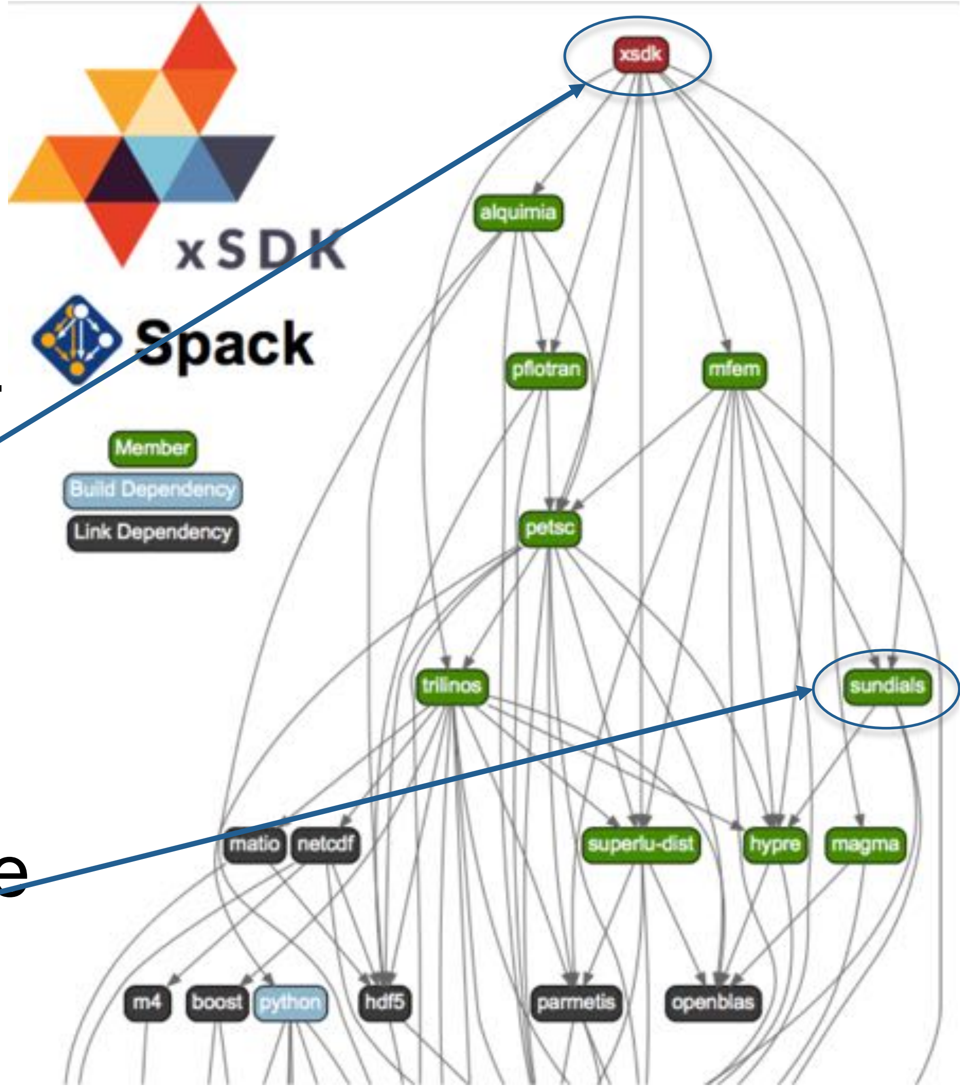
Easy customization

```
$ spack install mpileaks@3.3
$ spack install mpileaks@3.3 %gcc@4.7.3 +threads
$ spack install mpileaks@3.3 cppflags="-O3 –g3"
$ spack install mpileaks@3.3 target=haswell
$ spack install mpileaks@3.3 ^mpich@3.2
```



ECP ST Software Release Goals

- Build All ST Products that are ready.
  - Product readiness is part of success criteria.
  - Number of releasable products increase over time.
- SDKs will provide product suites.
  - Similar products, interoperable.
  - Consistent versions of dependencies.
  - Math SDK (aka, xSDK) is first SDK.
- We build the whole tree, so any subtree will be stable.
  - **spack install xsdk** – Build entire math SDK.
  - **spack install sundials** – Guaranteed to build correctly.

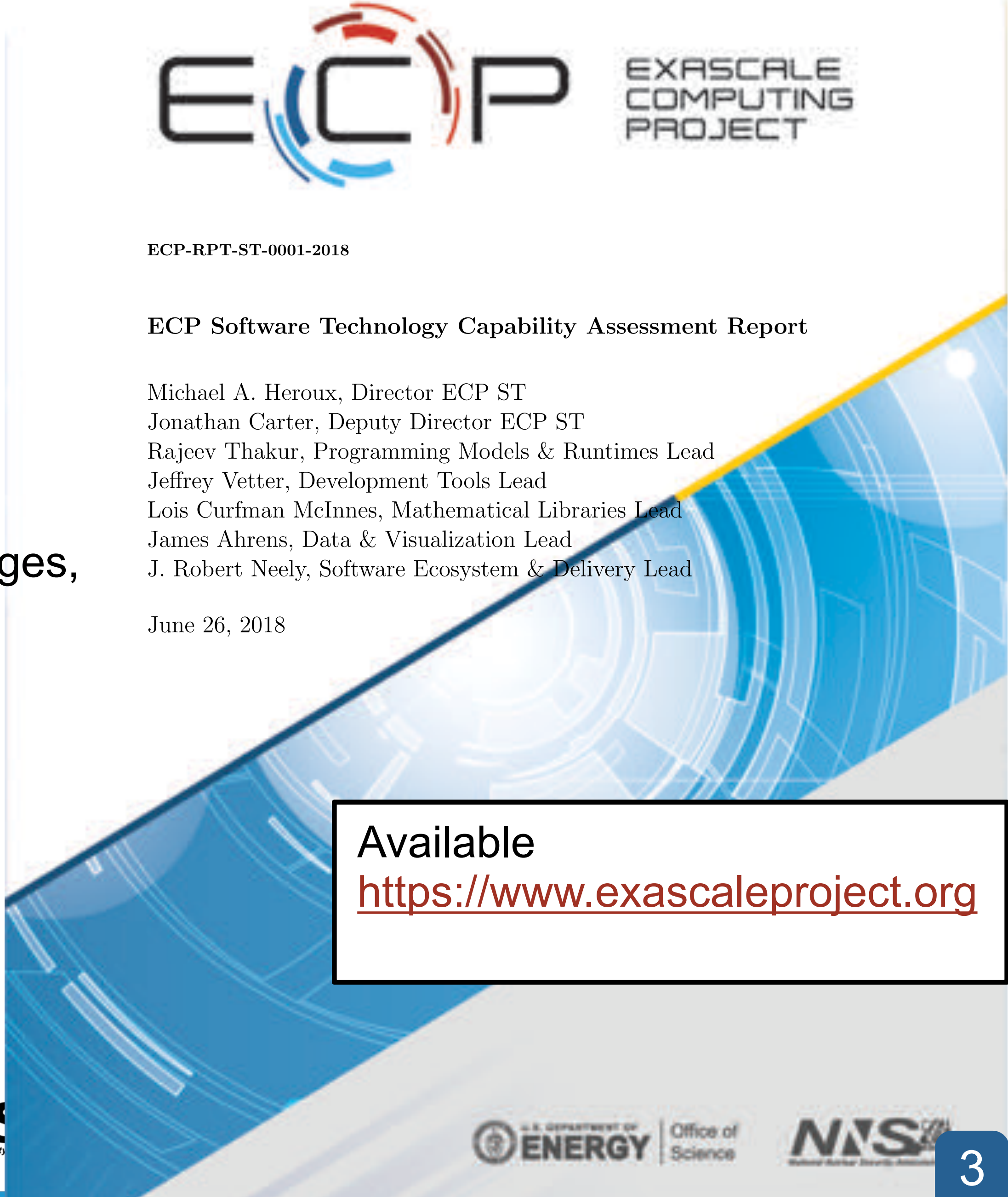


ECP ST Software Release Overview

- ECP ST Delivers Products as Source Code:
  - Tested regularly on target platforms including pre-Exascale.
  - Facilities, Vendors, Users pull from open repositories, coordinated with ECP HI.
- Build from Source
  - Spack build tool:
  - Hierarchical build of independently-developed products.
- Continuous Integration Testing (under construction).
  - ECP ST Products automatically pulled from development repositories.
  - Integrated regression testing performed on key DOE platforms.
- First Release
  - November 8: First Extreme-scale Scientific Software Stack (E4S) Release, Collection of Products Ready for Release http://e4s.io.
- Message: DOE committed to accessible, reusable, evolving software stack.
  - From source using Spack.
  - Via containers (Docker, Singularity, Shifter, CharlieCloud).
  - Laptop to Leadership platforms.
  - Released every six months.

ECP Software Technology Capability Assessment Report

- 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.


ECP 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

Available
https://www.exascaleproject.org

Question 1 – 3

Where is the most detailed description of the design of new capabilities you plan to develop?

Thinking about the ECP capabilities you have developed in the past year for the product, provide the best response.

- In the funded proposal document or statement of work.
- Undocumented. Part of an informal analysis of what needs to get done.
- In a written design document.
- In a prototype implementation.
- In an issue tracking system (Jira, GitHub Issue, etc).
- Other (Describe at end of survey).

Where is the most detailed description of what is required from new capabilities you plan to develop?

Thinking about the ECP capabilities you have developed in the past year for the product, provide the best response.

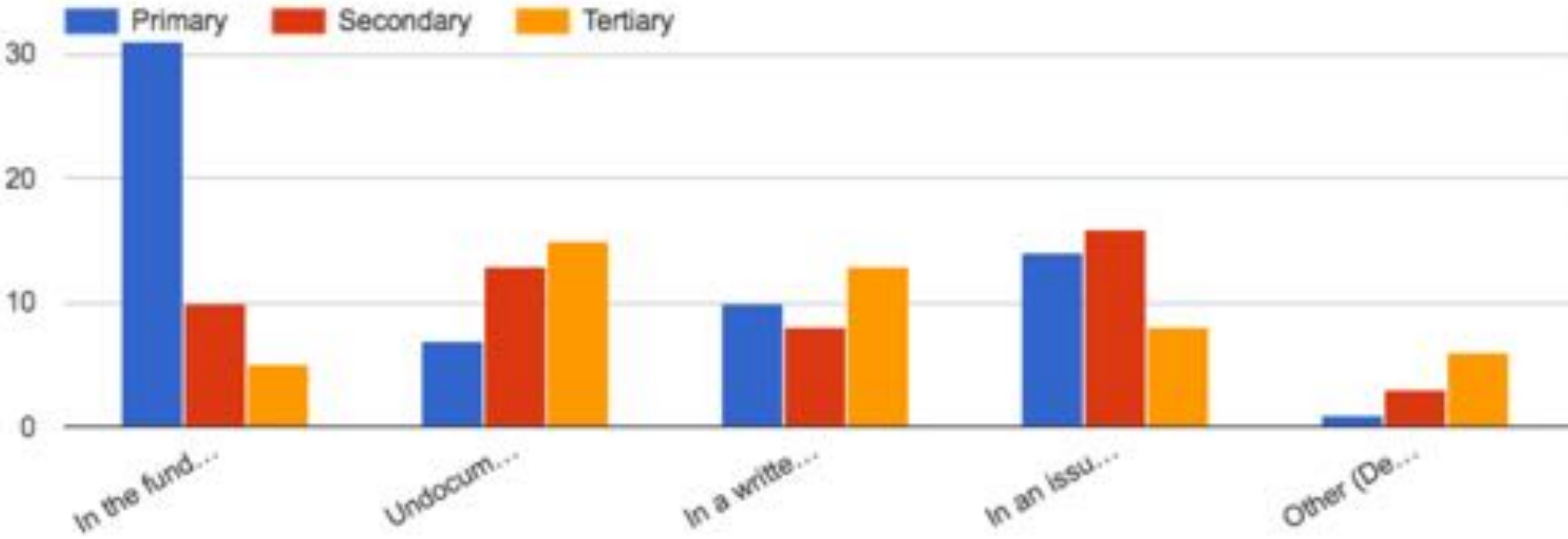
- In the funded proposal document or statement of work.
- Undocumented. Part of an informal analysis of what needs to get done.
- In a written requirements document.
- In an issue tracking system (Jira, GitHub Issue, etc).
- Other (Describe at end of survey).

How do you determine what capabilities you plan to develop?

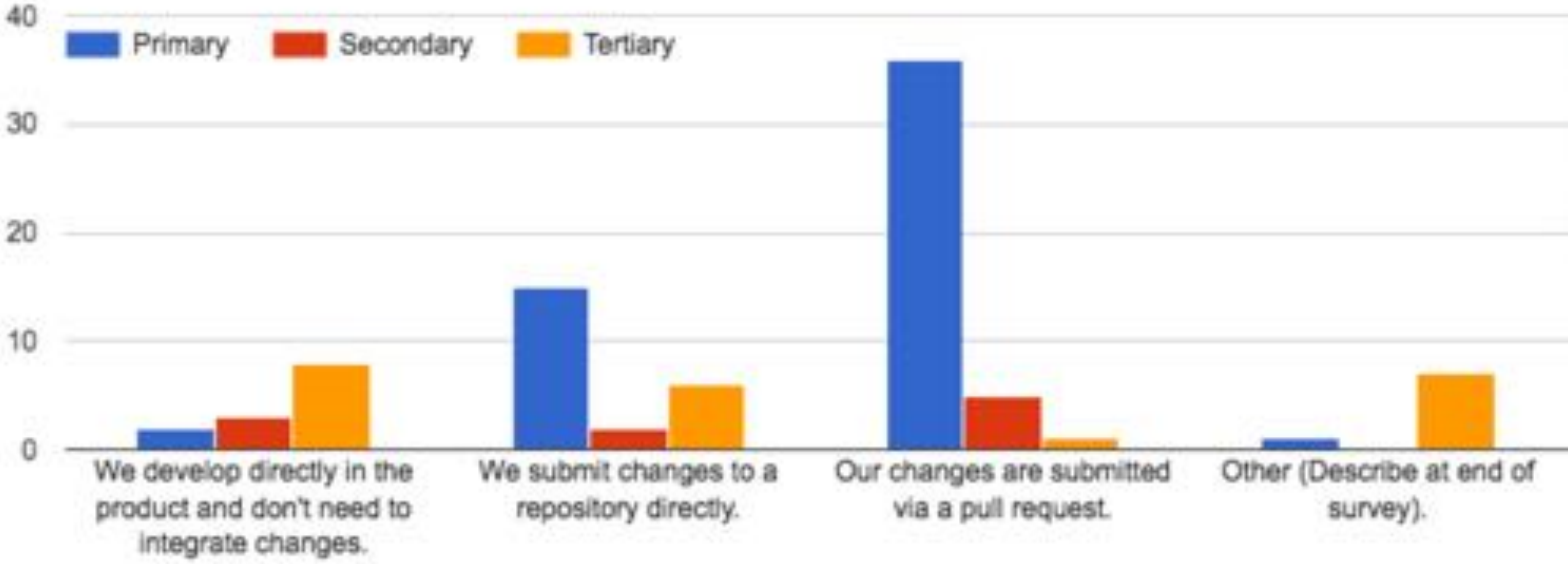
Thinking about the ECP capabilities you have developed in the past year for the product, provide the best response.

	Primary	Secondary	Tertiary
From informal user interactions and my knowledge of the field.	○	○	○
From formal stakeholder engagement (surveys, user sessions, etc.).	○	○	○
From ideas submitted through issue tracking system (Jira, GitHub Issue, etc).	○	○	○
Other (Describe at end of survey).	○	○	○

Where is the most detailed description of what is required from new capabilities you plan to develop?



How are new capabilities integrated into the existing product?



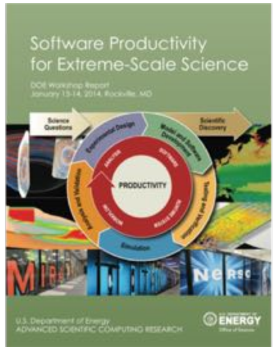
Community

Interoperable Design of Extreme-scale Application Software (IDEAS)

IDEAS productivity
www.ideas-productivity.org

Motivation

Enable ***increased scientific productivity,*** realizing the potential of extreme-scale computing, through ***a new interdisciplinary and agile approach to the scientific software ecosystem***.



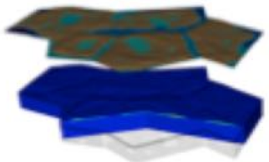
Objectives

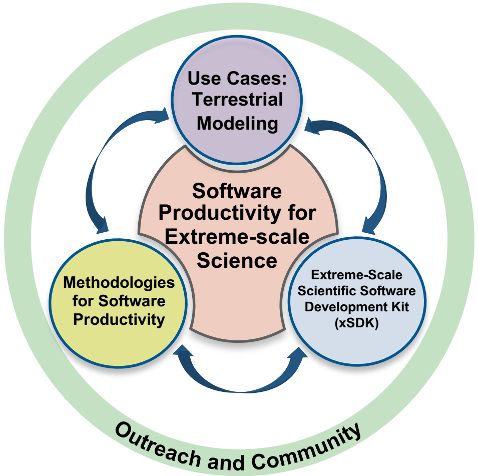
Address confluence of trends in hardware and increasing demands for predictive multiscale, multiphysics simulations.

Respond to trend of continuous refactoring with efficient agile software engineering methodologies and improved software design.

Impact on Applications & Programs

Terrestrial ecosystem ***use cases tie IDEAS to modeling and simulation goals*** in two Science Focus Area (SFA) programs and both Next Generation Ecosystem Experiment (NGEE) programs in DOE Biologic and Environmental Research (BER).





Approach

ASCR/BER partnership ensures delivery of both crosscutting methodologies and metrics with impact on real application and programs.

Interdisciplinary multi-lab team (ANL, LANL, LBNL, LLNL, ORNL, PNNL, SNL)

ASCR Co-Leads: Mike Heroux (SNL) and Lois Curfman McInnes (ANL)
BER Lead: David Moulton (LANL)

Integration and synergistic advances in three communities deliver scientific productivity; outreach establishes a new holistic perspective for the broader scientific community.

IDEAS history

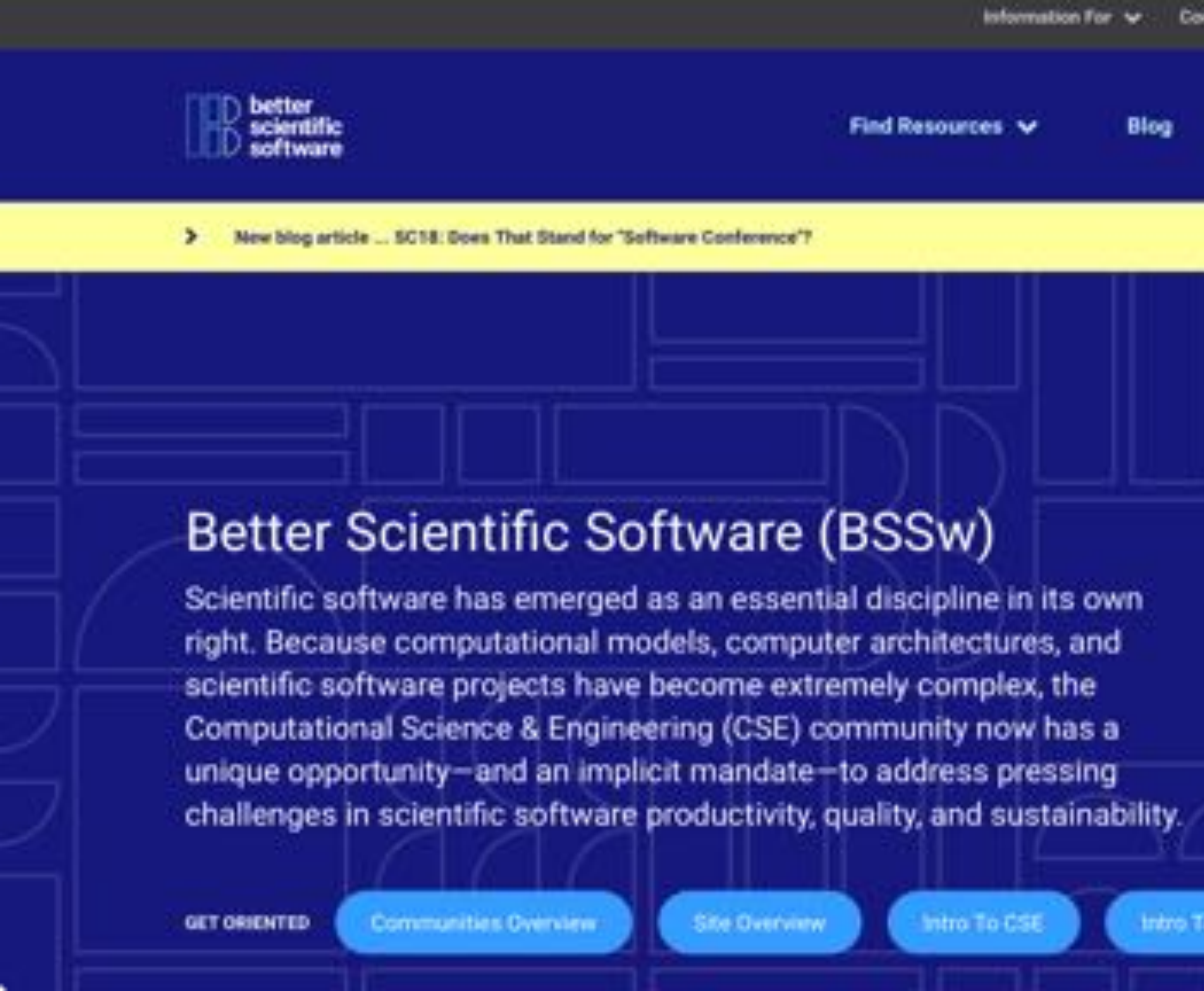
DOE ASCR/BER partnership began in Sept 2014

Program Managers:

- Paul Bayer, David Lesmes (BER)
- Thomas Ndousse-Fetter (ASCR)

First-of-a-kind project: qualitatively new approach based on making productivity and sustainability the explicit and primary principles for guiding our decisions and efforts.

NNSA National Nuclear Security Administration U.S. DEPARTMENT OF ENERGY Office of Science ECP EXASCALE COMPUTING PROJECT



https://bssw.io

Collaborative content development on general topics topics related to developer productivity and software sustainability for CSE

We want and *need* contributions from the community … Join us!

BSSw site history … And an invitation: Join us!

- **BSSw site launched at SC17**
  - **BOF on *Software Engineering and Reuse in Computational Science and Engineering***
    - https://swe-cse.github.io/2017-11-sc17-bof
- **Seeking contributions from US and international CSE community**
  - Researchers, practitioners, and stakeholders from national laboratories, academic institutions, and industry … share your resources, experiences, etc.
- **Over time: Collaborate to build the site to a vibrant community resource**
  - Content and editorial processes provided by volunteers throughout the CSE community
  - **We need your contributions!**

Initiative of the IDEAS Software Productivity Project

- Support from DOE Office of Advanced Scientific Computing Research, DOE Exascale Computing Project
- Thank you to DOE program mangers Thomas Ndousse-Fetter, Paul Bayer, and David Lesmes for encouragement and support

Promoting collaborative content creation through GitHub backend

BSSw Software Platform			
Component	**Backend**		**Frontend**
Technology	**Google Docs**	**GitHub**	**Ruby on Rails**
Location	Google Drive	betterscientificsoftware GitHub organization	https://bssw.io
Purpose	• Rapid collaborative content development • Multi-user typing, suggest edits, comments	• Content creation, refinement, management (from Google Drive) • Content packaging for use with bssw.io	• User-facing portal • Polished backend content • Blogs • Mailing lists
Contributors	Community subject matter experts	Community subject matter experts, BSSw staff	BSSw staff. Web development experts
Consumers	BSSw GitHub Backend	BSSw Frontend	CSE community
Content Notes	Content migrates to GitHub after it stabilizes	Content managed in git repos, markdown	Content from Backend

Contribute! Share your insights on CSE software practices and processes:

- https://github.com/betterscientificsoftware/betterscientificsoftware.github.io/blob/master/README.md
- Or search "github betterscientificsoftware"

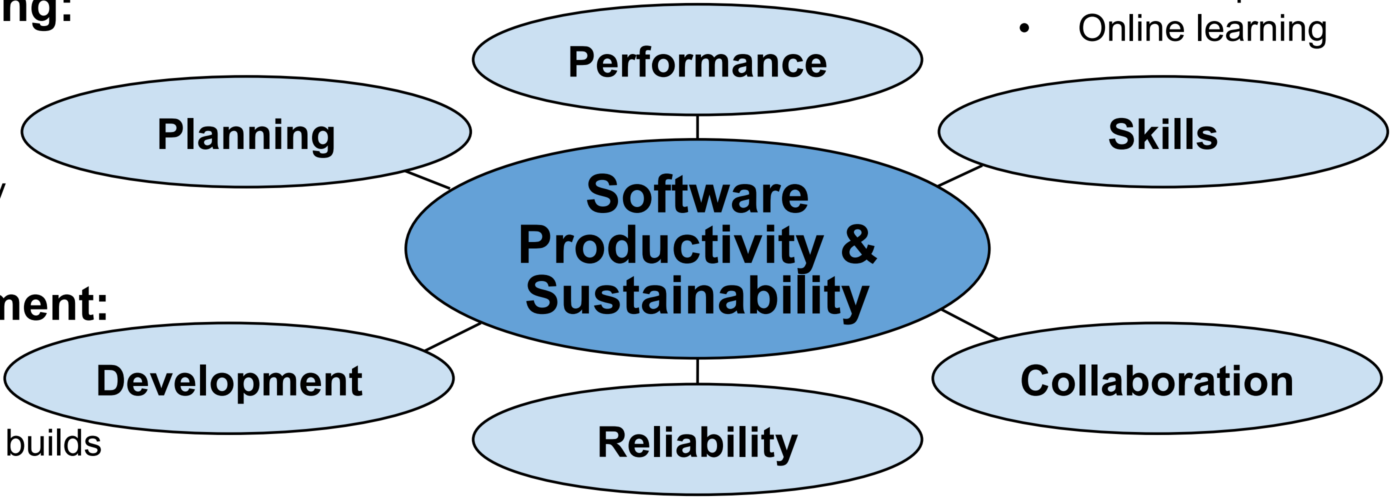
Resource topics

better scientific software

Better Performance:
- High-performance computing
- Performance at LCFs
- Performance portability

Better Skills:
- Personal productivity and sustainability
- Online learning

Better Planning:
- Requirements
- Design
- Software interoperability



Better Development:
- Documentation
- Version control
- Configuration and builds
- Deployment
- Issue tracking
- Refactoring
- Software engineering
- Development tools

Better Collaboration:
- Licensing
- Strategies for more effective teams
- Funding sources and programs
- Projects and organizations
- Software publishing and citation
- Discussion forums, Q&A sites

Better Reliability:
- Testing
- Continuous integration testing
- Reproducibility
- Debugging

Site content spans a broad range of topics.

Resource examples

Curated links: A brief article that highlights other web-based articles or content. Your article should describe why the CSE community might find value.

An Introduction To Software Licensing

Share

This tutorial provides a brief introduction to software copyright and licensing for researchers in computational science and engineering. Explains the difference between closed and open source software, and copyleft and permissive open source licenses. Outlines a variety of factors researchers might want to consider when selecting a software license. Provides links to some key web resources as a starting point for deeper exploration.

Prerequisites

What Is Software Intellectual Property?

PUBLISHED JUNE 20, 2017 CONTRIBUTOR DAVID BERNHOLDT

Tutorial presented at SIAM CSE17: CSE Collaboration through Software: Improving Productivity and Sustainability.

A recording of this tutorial presentation is available at https://www.pathlms.com/siam/courses/4150/sections/5826/video_presentations/42639

https://bssw.io/resources/an-introduction-to-software-licensing

Planning For Better Software: PSIP Tools

Share

Scientific software teams are typically focused on the creation of a new set of features that will enable the next set of computational experiments. Teams seldom have the time to stop development and focus solely on improving productivity or sustainability. However, teams can incorporate improvements *on the way* to developing new science capabilities.

Prerequisites

CSE Software Requirements?

What Are Strategies For More Effective Teams?

PUBLISHED NOVEMBER 21, 2017 CONTRIBUTOR MIKE HEROUX

The Productivity and Sustainability Improvement Planning (PSIP) process recognizes that productivity and sustainability improvements for scientific software benefit from an incremental, iterative approach. The PSIP-Tools GitHub repo is a collection of documents that enable the adoption and use of PSIP for a software team. The PSIP-Tools repo contains everything from a template for the first introduction letter to a complete interview guide, interview prompts and expected timeline.

The PSIP process has been successfully used to help scientific software teams acheive incremental, sustainable process improvement, while still achieving their science goals.

https://bssw.io/resources/planning-for-better-software-psip-tools

BSSw blog articles

better scientific software

Find Resources

BSSw Blog

Better Scientific Software (BSSw) presents articles from expert community members on a topics related to software productivity and sustainability.

Would you like share your ideas through a blog article?

The BSSw blog provides a platform to inform, inspire, and mobilize the community toward better software practices. Please see details on how to contribute to BSSw.

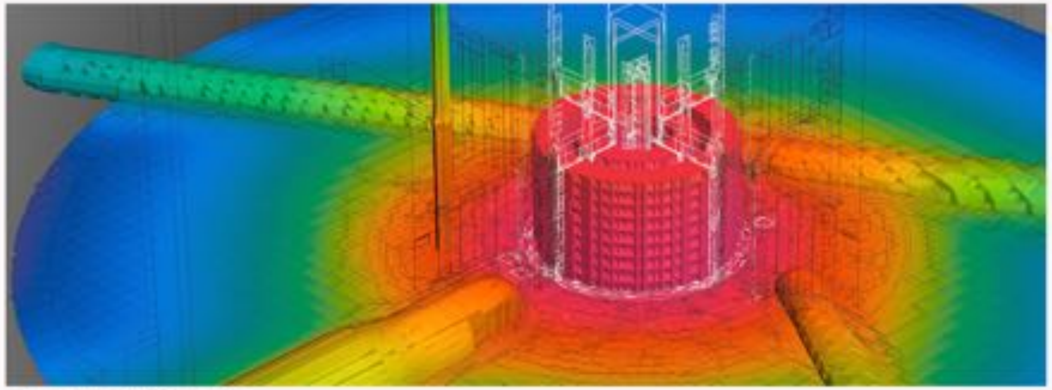
better scientific software

Find Resources | Blog | Events | About

HOME · BLOG · 2018 · BETTER SCIENCE THROUGH SOFTWARE TESTING

Better Science Through Software Testing

Share



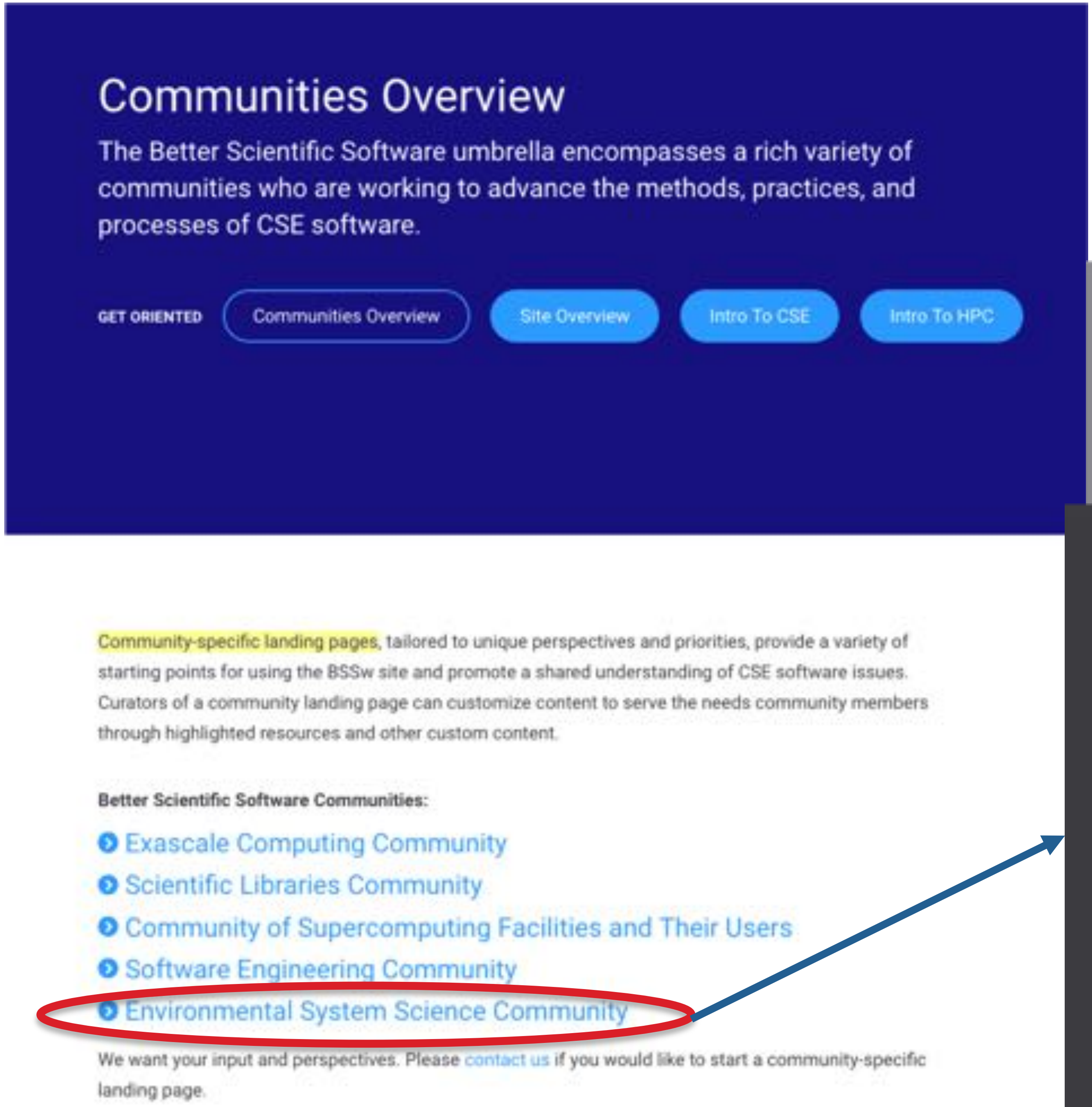
HIGH-FLUX ISOTOPE REACTOR

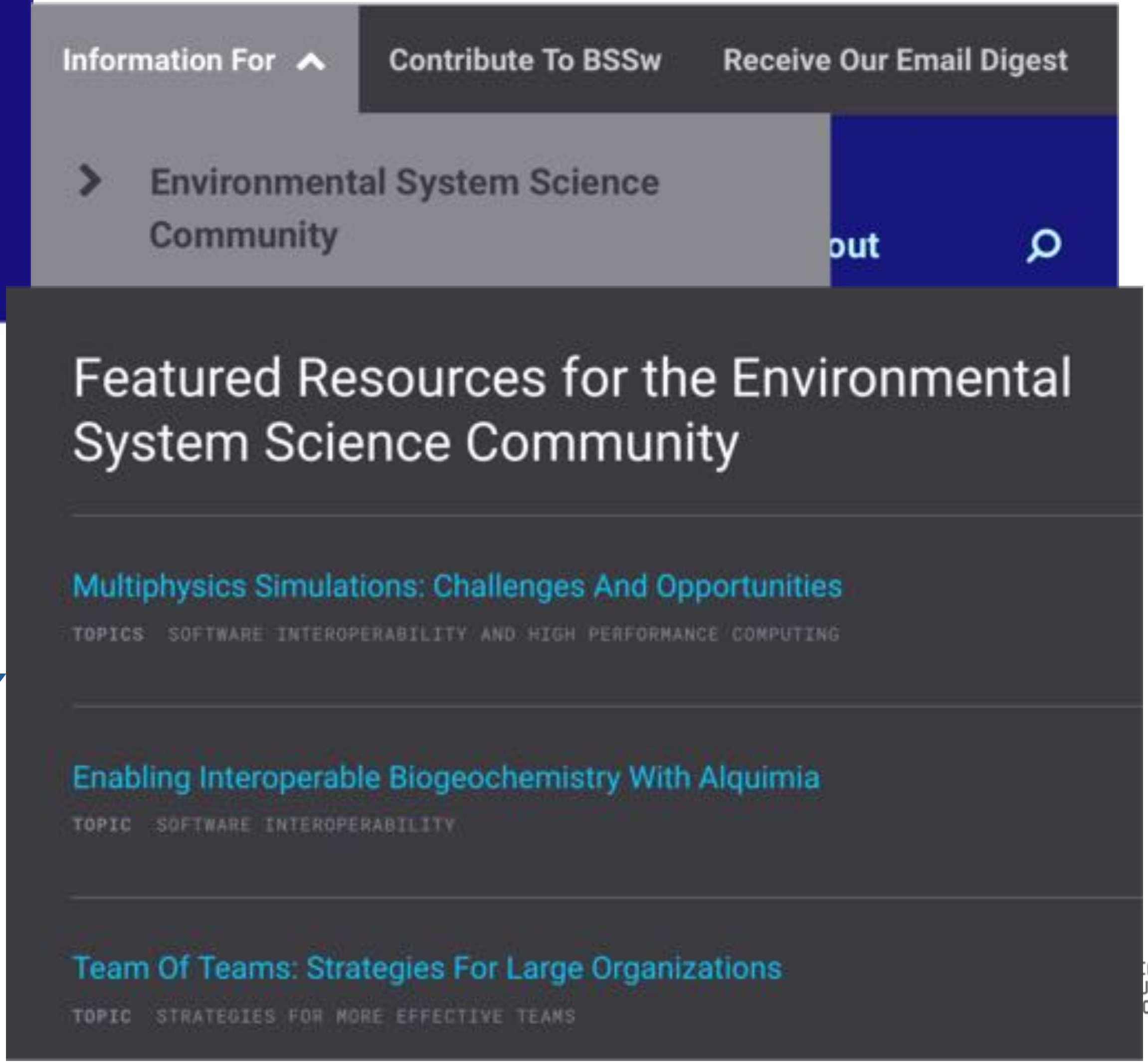
PUBLISHED FEBRUARY 2, 2018 AUTHOR TOM EVANS TOPIC(S) TESTING, REQUIREMENTS, DESIGN

In November 2017, Tom Evans gave a webinar titled "Managing Defects in HPC Software Development" in the series Best Practices for HPC Software Developers. In this article, Tom summarizes how the strategies he employs have helped his teams deliver better science. Tom is a scientist at Oak Ridge National Laboratory; he leads the project "Coupled Monte Carlo Neutronics and Fluid Flow Simulation of Small Modular Reactors," part of the DOE Exascale Computing Project.

Contributor Tom Evans, ORNL

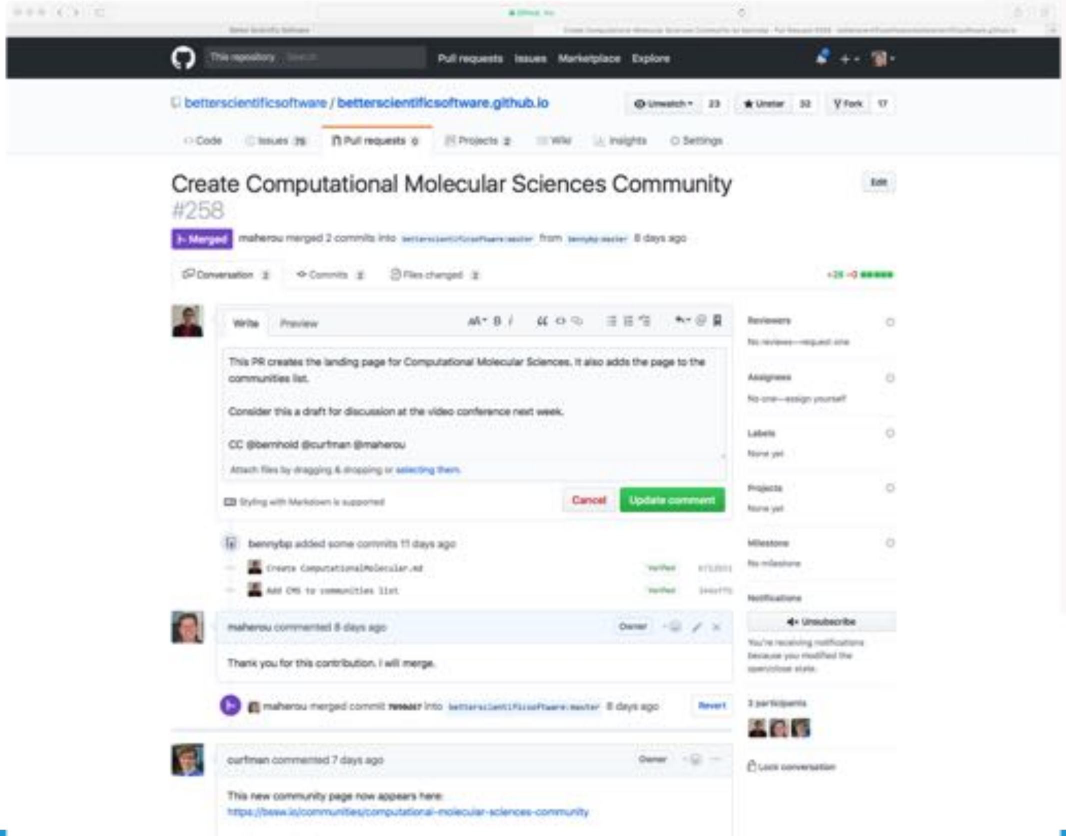
Community landing pages

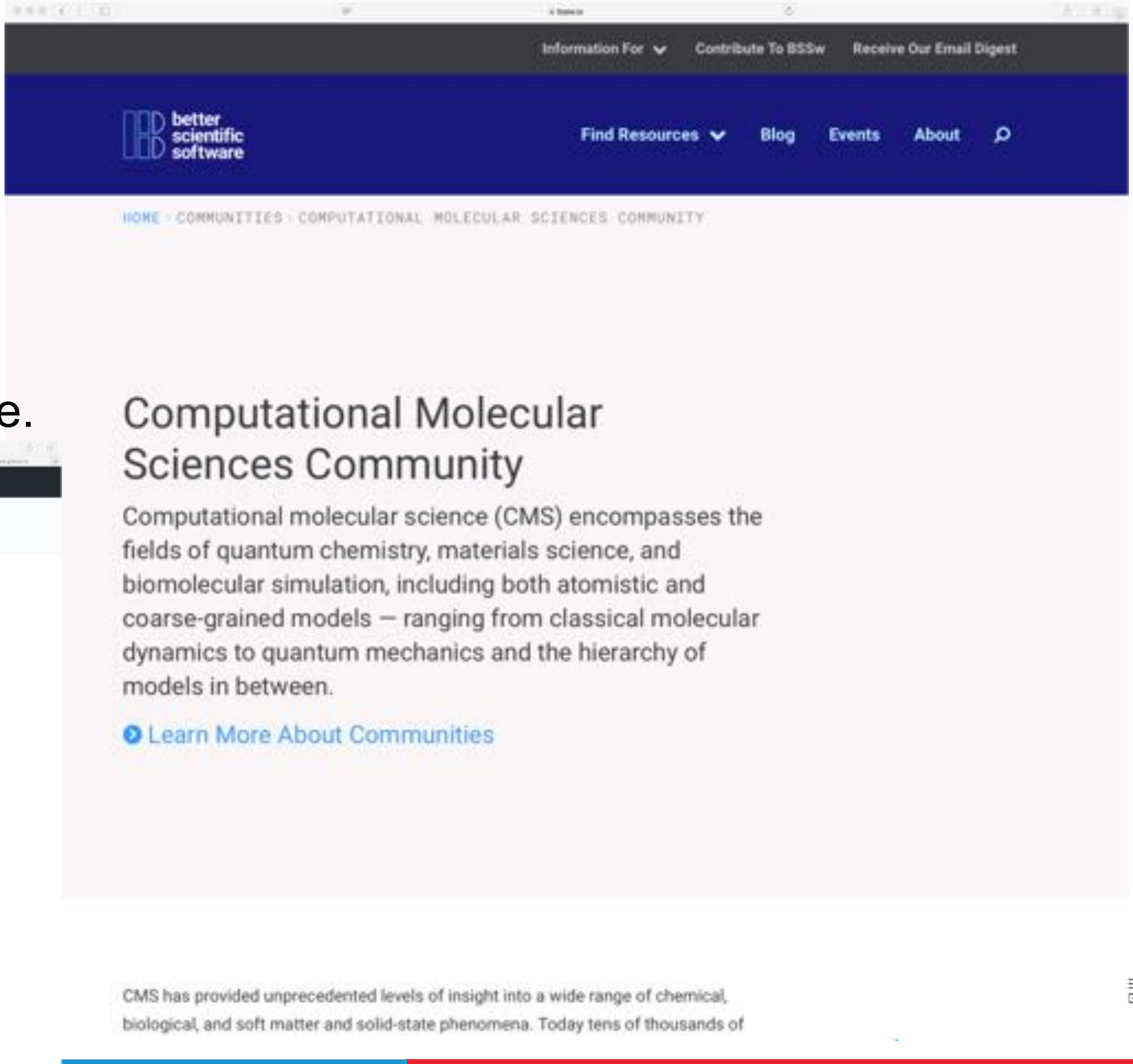




Latest Addition: CMS

- MolSSI (VA-Tech)
- Community Page for Computational Molecular Science.





better scientific software   Menu

BSSw Fellowship Program

The Better Scientific Software (BSSw) Fellowship Program gives recognition and funding to leaders and advocates of high-quality scientific software.

2019 Fellows selected.
Announced soon.

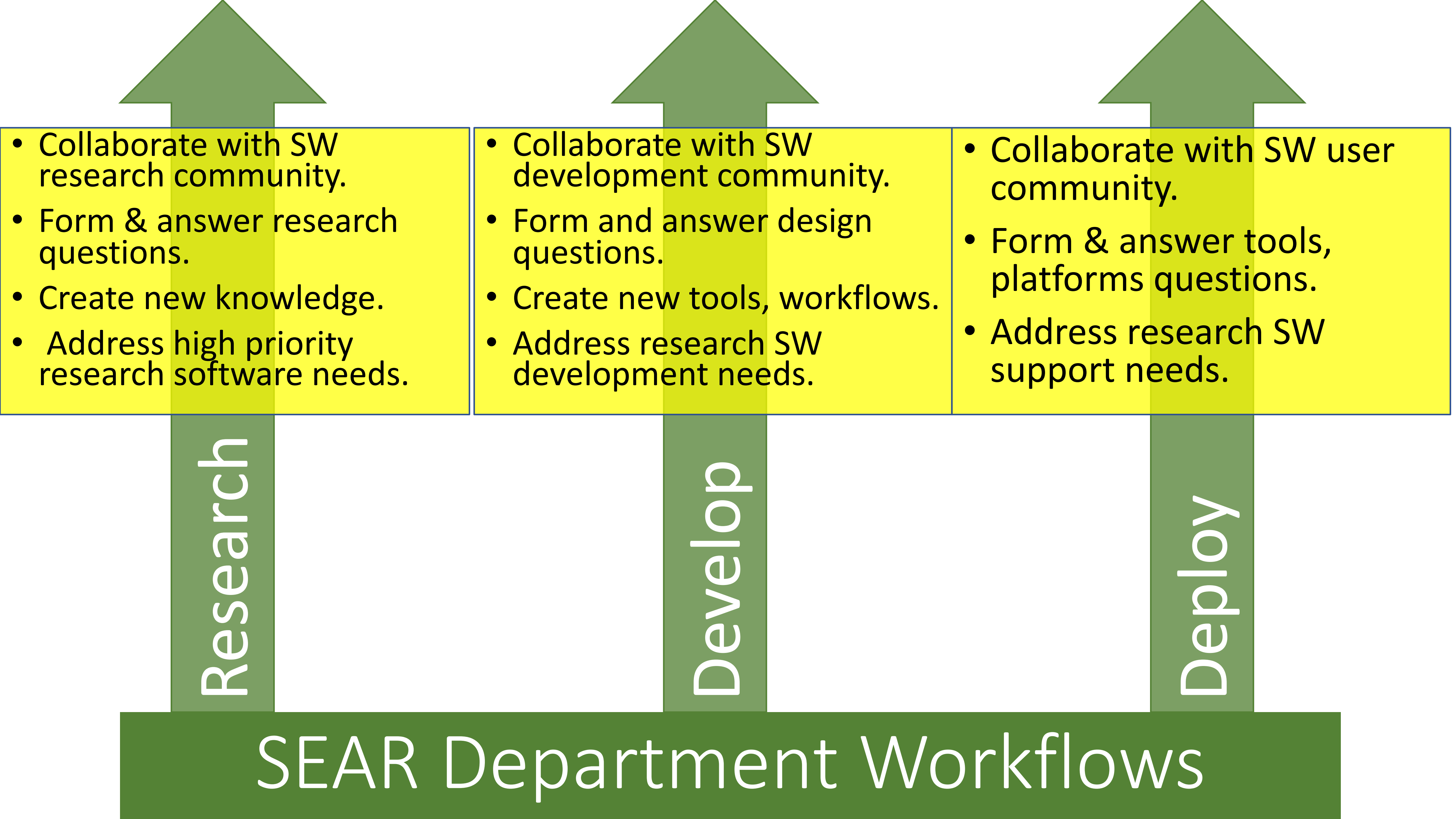
About Our Fellows Program

The main goal of the BSSw Fellowship program is to foster and promote practices, processes, and tools to improve developer productivity and software sustainability of scientific codes. We also anticipate accumulating a growing community of BSSw Fellowship alums who can serve as leaders, mentors, and consultants to increase the visibility of those involved in scientific software production and sustainability in the pursuit of scientific discovery.

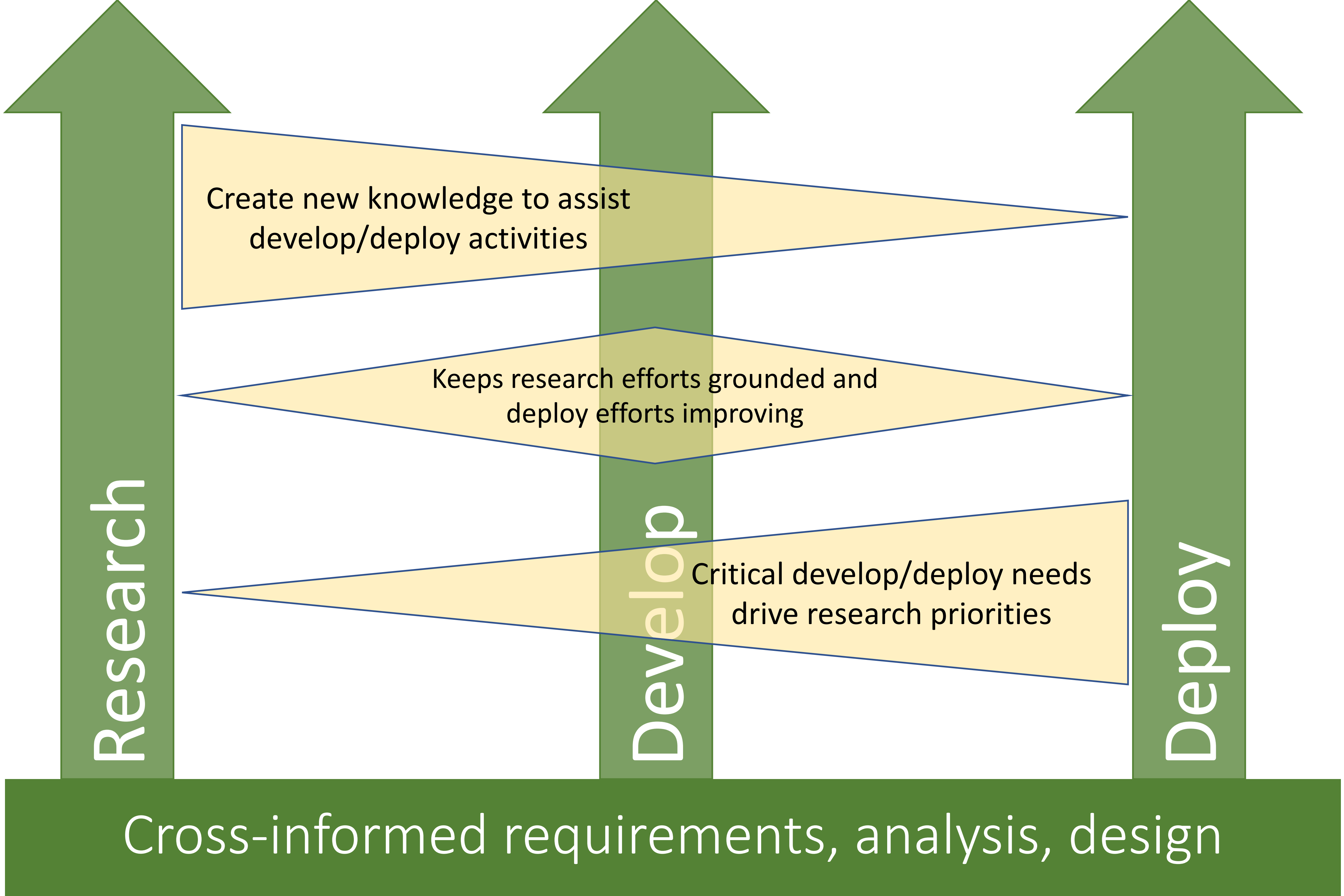
BSSw Fellows are selected annually based on an application process that includes the proposal of a funded activity that promotes better scientific software. We select at least three Fellows per year and honorable mentions as appropriate. Each 2019 BSSw Fellow will receive up to $25,000 for an activity that promotes better scientific software. Activities can include organizing a workshop, preparing a tutorial, or

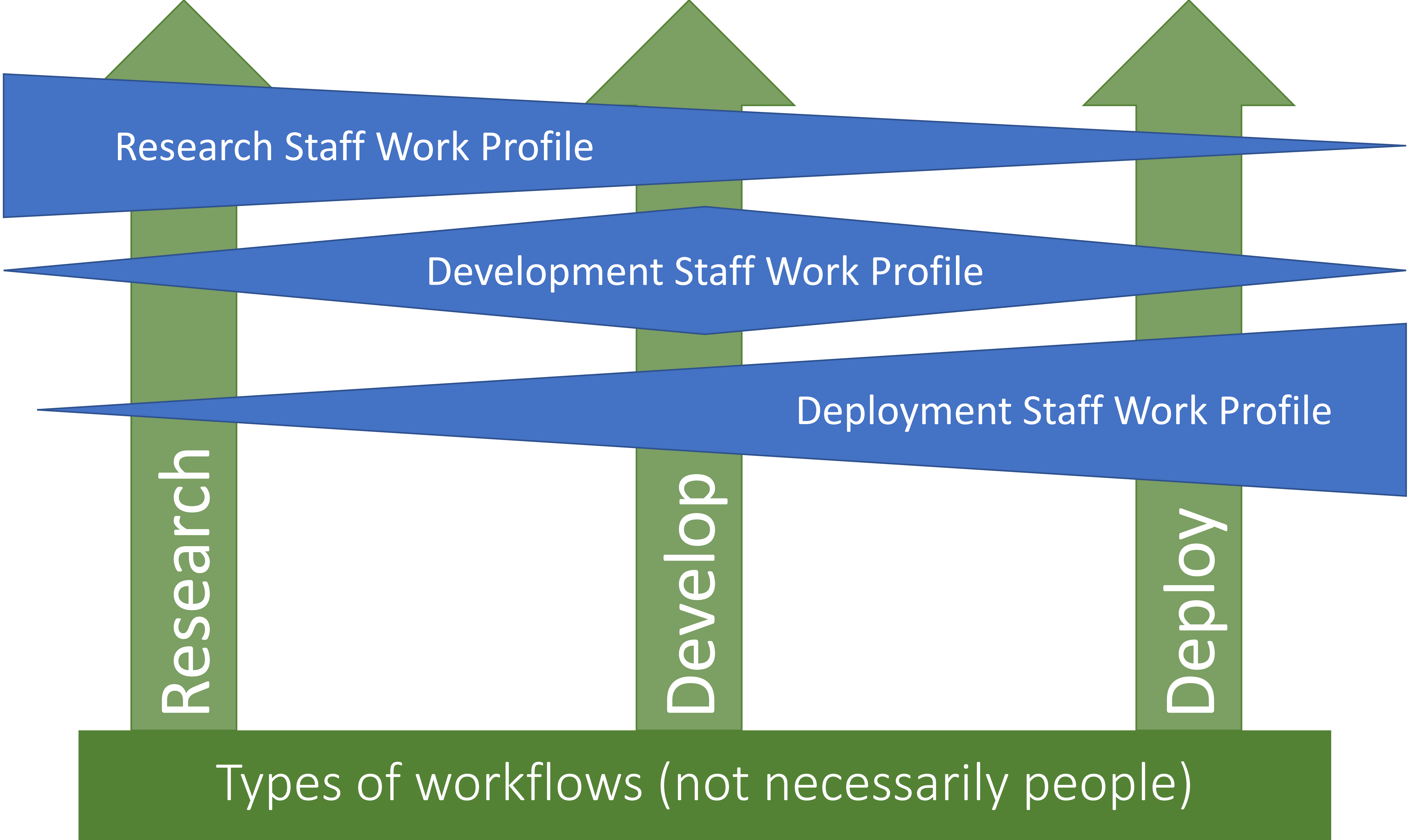
Sandia SW Engineering and Research (SEAR) Department

- New department focused on making CSE Research SW better:
  - Focus on scientific & engineering research software.
  - Improve developer productivity and software sustainability.
  - Bring a critical mass of existing staff (but not necessarily all) into a department.
  - Attract new talent by making SW Eng & Research first-class citizen.
  - Build a community presence visible to DOE and external community.
  - Build on Sandia's native engineering culture.
- Three primary department workflows:
  - **Research:** Participate in research community to understand and create new knowledge for improving CSE research software.
  - **Develop:** Identify, cultivate SW best practices prioritized for CSE research software development (part of SEMS scope).
  - **Deploy:** Provide effective SW tools and environments adapted to CSE research software teams (part of SEMS scope).



Collaborate with SW research community.
Form & answer research questions.
Create new knowledge.
Address high priority research software needs.
Collaborate with SW development community.
Form and answer design questions.
Create new tools, workflows.
Address research SW development needs.
Collaborate with SW user community.
Form & answer tools, platforms questions.
Address research SW support needs.
Research
Develop
Deploy
SEAR Department Workflows


Research
Develop
Deploy
Create new knowledge to assist develop/deploy activities
Keeps research efforts grounded and deploy efforts improving
Critical develop/deploy needs drive research priorities
Cross-informed requirements, analysis, design


Research Staff Work Profile
Development Staff Work Profile
Deployment Staff Work Profile
Research
Develop
Deploy
Types of workflows (not necessarily people)

Why First-Class SW Focus now: The “No CS” Scenario

Scenario: Suppose our research centers had no formally trained computer scientists and CS work had to be done by people who learned it on their own, or just happened to study a bit of CS as part of their other formal training. This situation is undesirable in three ways:

1. We have non-experts doing CS work, making them less available in their expertise.
2. CS work takes a long time to complete compared to other work.
3. We get suboptimal results and pay high ongoing maintenance cost.

Replace ”CS” with “Software” in this scenario and the situation describes computational science and engineering (CSE) software today.

Why focus on software expertise now:

- The role of software has become central to much of our work and the knowledge base is too sophisticated to rely only on non-experts.
- CSE success depends on producing high-quality, sustainable software products.
- Investing in software as a first class pursuit improves the whole CSE ecosystem.

Applying Social Science to Software Teams

- Reed Milewicz – my postdoc.
- Elaine Raybourn – Sandia social scientist recruited to my team.
- New scientific tools to study and improve developer productivity, software sustainability.
- Correlation: Happiness and connectedness.
- Next: Design experiments to detect cause and effect.

Source: https://arxiv.org/pdf/1809.06317.pdf

Personal Expectations

Calling out the best in team members

A Few Concrete Recommendations

Show me the person making the most commits on an undisciplined software project and I will show you the person who is injecting the most technical debt.

- GitHub stats: Easy to find who made the most commits.
  - Some people: Pride in their high ranking.

- Instead, be the person who ranks high in these ways:
  - Writes up requirements, analysis and design, even if simple.
  - Writes good GitHub issues, tracks their progress to completion.
  - Comments on, tests and accepts pull requests.
  - Provide good wiki, gh-pages content, responses to user issues.

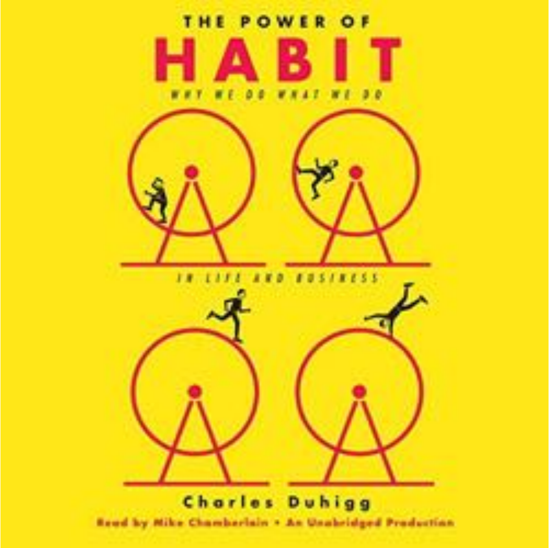
(Personal) Productivity++ Initiative

Ask: *Is My Work _______ ?*

https://github.com/trilinos/Trilinos/wiki/Productivity---Initiative

Reproducibility: A keystone habit

Alcoa and Worker Safety (Duhigg)



- Year: 1987
- Investors concerned about Alcoa.
- Paul O’Neill – Selected Alcoa CEO, not well known.
- First statement: "I want to talk to you about worker safety.”
- Investors panicked. But …
- Executed top-to-bottom safety focus.
- 10X injury drop, 5X revenue growth.

> "I knew I had to transform Alcoa. But you can't order people to change. So I decided I was going to start by focusing on one thing. If I could start disrupting the habits around one thing, it would spread throughout the entire company.”
>
> - Paul O’Neill

Reproducibility and Computational Science

- Aluminum workers:
  - 1500 degree heat, dangerous machines.
  - Safety is key.
- Reproducibility: Key for computational science.
- Can we make reproducibility requirements the keystone habit?
- Experiment:
  - 2020 Sandia LDRD projects: All computational results must be reproducible.
  - Drives rigor, innovation: provenance, tools, practices, communication.
  - Engages Sandia Technical Library.
- Anticipated:
  - Holistic focus on doing the right things.
  - Strong incentive to do things right.

Making Reproducibility Indispensable

- We see heightened focus on:
  - Workflows.
  - Reproducibility requirements.
  - Software quality requirements.
  - Community Incentives.
- We have improved tools, practices, processes.
- Can we expect that all published computational results will be reproducible?
- Let's make it so.