Improving Productivity and Sustainability for Parallel Computing Software

EXASCALE COMPUTING PROJECT

Judy Hill (ORNL)
Rinku Gupta (ANL)
Lois Curfman McInnes (ANL)

February 14, 2020
SIAM Conference on Parallel Processing (PP20)
Seattle, WA

National Nuclear Security Administration

U.S. DEPARTMENT OF ENERGY | Office of Science

IDEAS productivity

Partnering with the community to improve developer productivity and software sustainability as key aspects of increasing overall scientific productivity.

https://ideas-productivity.org

- **Productivity and Sustainability Improvement Planning (PSIP)** https://bssw.io/psip

psip

A lightweight iterative workflow, where teams identify their most urgent software bottlenecks and track progress to overcome them.

- **Resources for Software Development Kits (SDKs)** facilitate work toward a sustainable ECP software ecosystem
  - https://github.com/betterscientificsoftware/SDK-Tools

- **IDEAS-ECP outreach** communicates best practices; partners with synergistic groups
  - Webinar series on *Best Practices for HPC Software Developers*
  - Tutorials on *Better Scientific Software*
    - Software testing, code coverage
    - Improving reproducibility through better scientific software
    - Verification & refactoring
    - Git workflows
    - Introduction to software licensing
    - Better (small) scientific software teams

IDEAS-ECP: Advancing Software Productivity for Exascale Applications

better scientific software

Rate Your Project's Software Practices

Lightweight Software Quality Practice

- **Better Scientific Software site** (https://bssw.io)

better scientific software

Provides a community-based hub for sharing info on practices, techniques, and tools to improve developer productivity & software sustainability

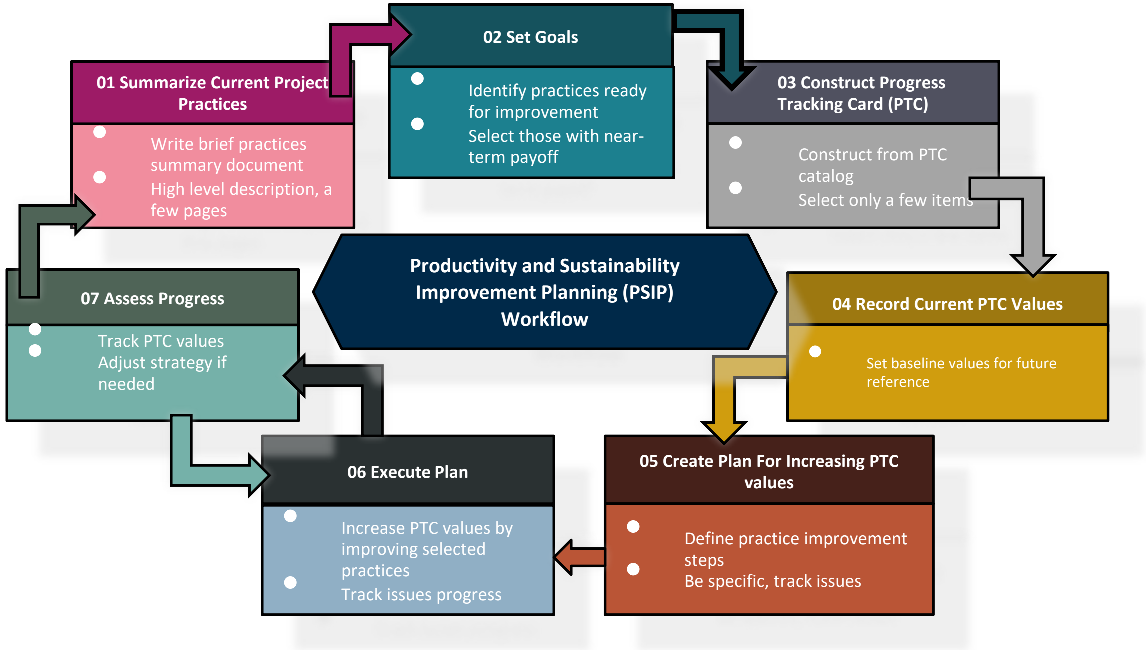
- **BSSw Fellowship Program**
  - Provides recognition & funding to leaders & advocates of high-quality scientific software; provides SWP methodologies for ECP community

Productivity and Sustainability Improvement Planning (PSIP)

psip

A lightweight iterative workflow, where teams identify their most urgent software bottlenecks and track progress to overcome them.

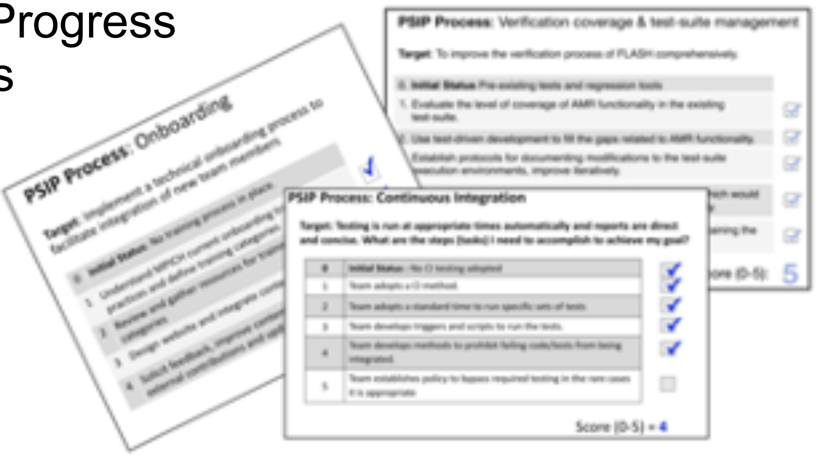
PSIP workflow helps a team identify areas for improvement, select a specific area and topic for a single improvement cycle, and then develop those improvements with specific metrics for success.



02 Set Goals
Identify practices ready for improvement
Select those with near-term payoff
01 Summarize Current Project Practices
Write brief practices summary document
High level description, a few pages
03 Construct Progress Tracking Card (PTC)
Construct from PTC catalog
Select only a few items
Productivity and Sustainability Improvement Planning (PSIP) Workflow
07 Assess Progress
Track PTC values
Adjust strategy if needed
04 Record Current PTC Values
Set baseline values for future reference
06 Execute Plan
Increase PTC values by improving selected practices
Track issues progress
05 Create Plan For Increasing PTC values
Define practice improvement steps
Be specific, track issues

Snapshots of Progress Tracking Cards

By EXAALT, ExaStar, MPICH teams



PSIP Process: Verification coverage & test-suite management

Target: To improve the verification process of FLASH comprehensively.

0. **Initial Status:** Pre-existing tests and regression tools
1. Evaluate the level of coverage of AMR functionality in the existing test-suite.
2. Use test-driven development to fill the gaps related to AMR functionality.
3. Establish protocols for documenting modifications to the test-suite execution environments, improve iteratively.
4. [illegible] which would [illegible]
5. [illegible] aining the [illegible]

Score (0-5): 5

PSIP Process: Onboarding

Target: Implement a technical onboarding process to facilitate integration of new team members.

0. **Initial Status:** No training process in place
1. Understand MPICH current onboarding [illegible] practices and define training categories
2. Review and gather resources for [illegible] categories
3. Design website and integrate [illegible]
4. Solicit feedback, improve [illegible] external contributions and [illegible]

PSIP Process: Continuous Integration

Target: Testing is run at appropriate times automatically and reports are direct and concise. What are the steps (tasks) I need to accomplish to achieve my goal?

0	**Initial Status:** No CI testing adopted	✓
1	Team adopts a CI method	✓
2	Team adopts a standard time to run specific sets of tests	✓
3	Team develops triggers and scripts to run the tests	✓
4	Team develops methods to prohibit failing code/tests from being integrated	✓
5	Team establishes policy to bypass required testing in the rare cases it is appropriate	□

Score (0-5) = 4

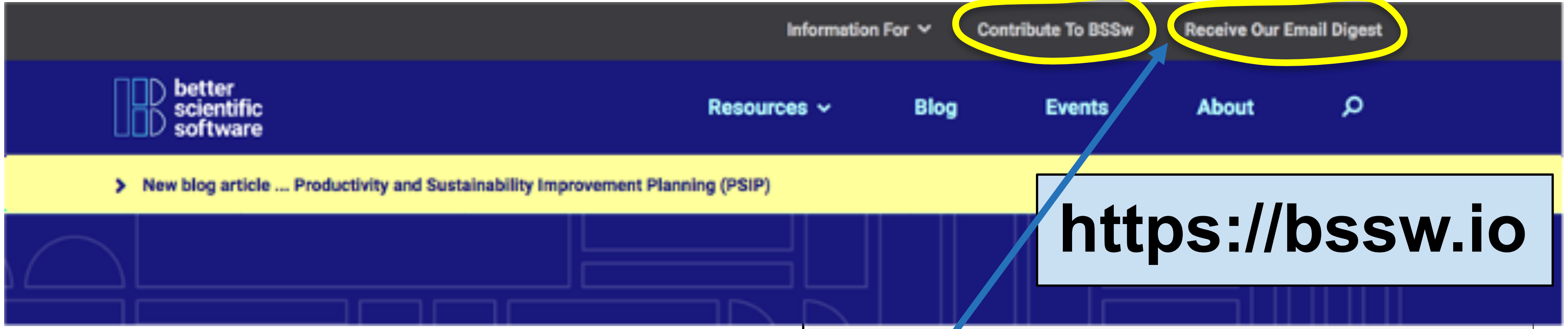
Want more info?

https://bssw.io/psip

ECP EXASCALE COMPUTING PROJECT

better scientific software

So your code will see the future.



New blog article … Productivity and Sustainability Improvement Planning (PSIP)

https://bssw.io

What is BSSw?

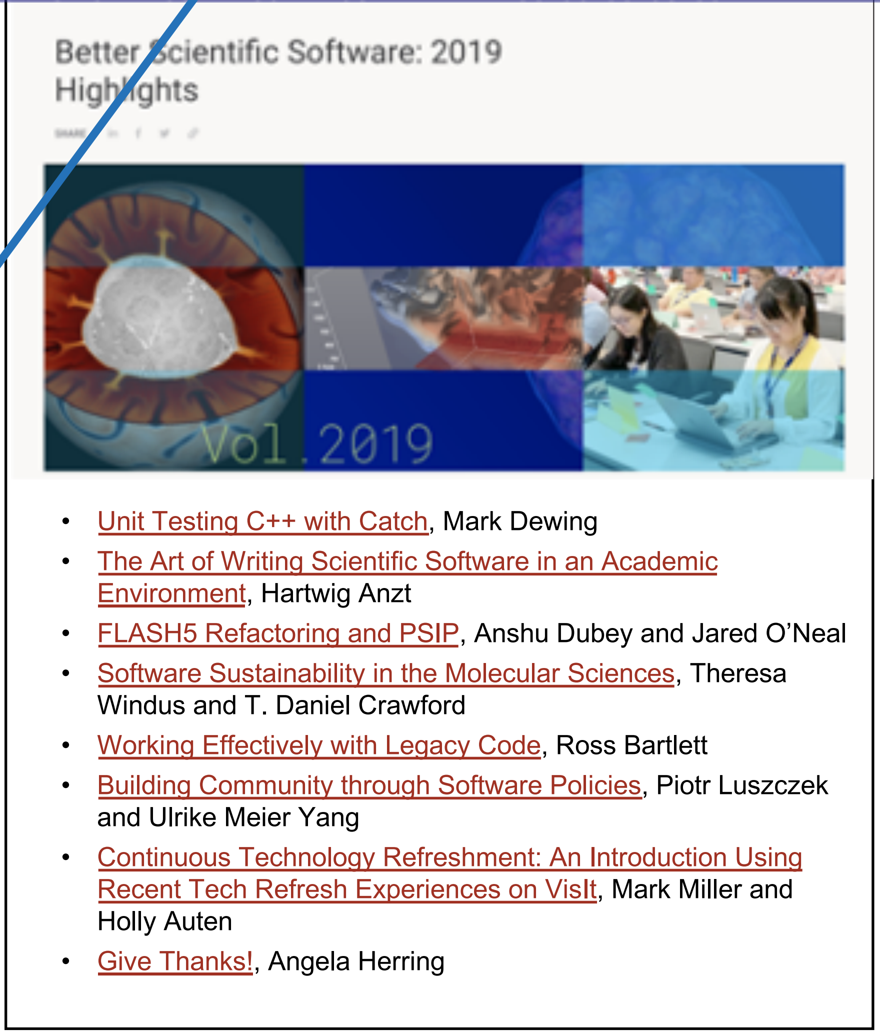
Community-based hub for sharing information on practices, techniques, and tools to improve developer productivity and software sustainability for computational science.

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

- **Types of content**
  - Informative articles
  - Curated links
    - Highlight other web-based content
  - Events
  - WhatIs, HowTo docs
  - Blog articles

Receive our email digest

Many contributors



Better Scientific Software: 2019 Highlights

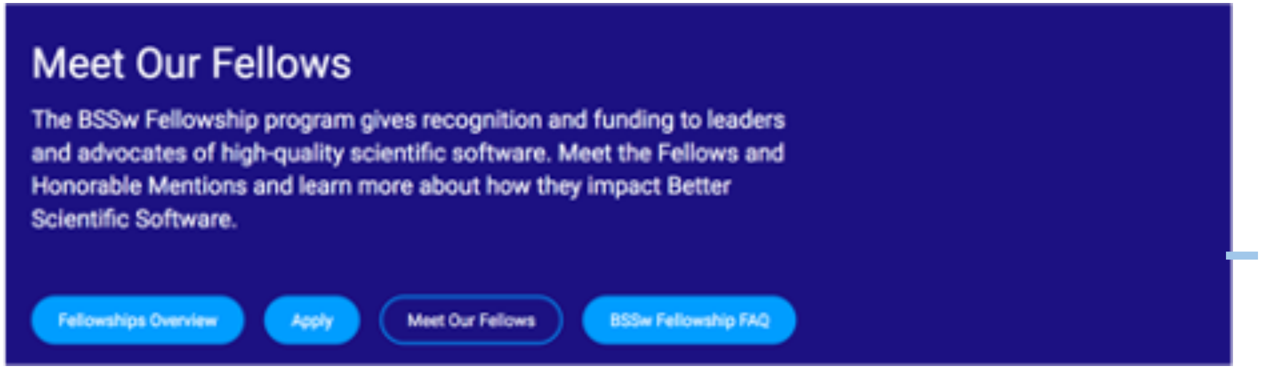
- Unit Testing C++ with Catch, Mark Dewing
- The Art of Writing Scientific Software in an Academic Environment, Hartwig Anzt
- FLASH5 Refactoring and PSIP, Anshu Dubey and Jared O'Neal
- Software Sustainability in the Molecular Sciences, Theresa Windus and T. Daniel Crawford
- Working Effectively with Legacy Code, Ross Bartlett
- Building Community through Software Policies, Piotr Luszczek and Ulrike Meier Yang
- Continuous Technology Refreshment: An Introduction Using Recent Tech Refresh Experiences on VisIt, Mark Miller and Holly Auten
- Give Thanks!, Angela Herring

ECP EXASCALE COMPUTING PROJECT

BSSw Fellowship

https://bssw.io/fellowship

GOAL: Foster and promote practices, processes, and tools to improve developer productivity and software sustainability of scientific codes

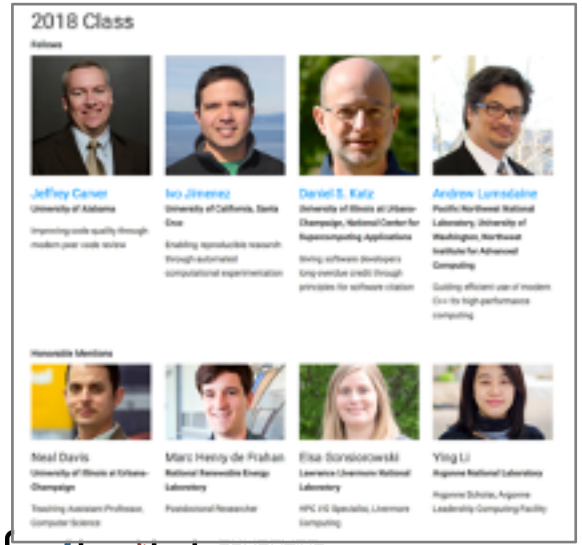


Meet Our Fellows

The BSSw Fellowship program gives recognition and funding to leaders and advocates of high-quality scientific software. Meet the Fellows and Honorable Mentions and learn more about how they impact Better Scientific Software.

AWARD: We select at least 3 Fellows per year and honorable mentions as appropriate. Each 2020 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 creating content to engage the scientific software community.

2018 - 2020



2018 Class

Fellows

Jeffrey Carver
University of Alabama
Improving code quality through modern peer code review

Ivo Jimenez
University of California, Santa Cruz
Enabling reproducible research through automated computational experimentation

Daniel S. Katz
University of Illinois at Urbana-Champaign, National Center for Supercomputing Applications
Giving software developers long-overdue credit through principles for software citation

Andrew Lumsdaine
Pacific Northwest National Laboratory, University of Washington, Northwest Institute for Advanced Computing
Guiding efficient use of modern C++ for high-performance computing

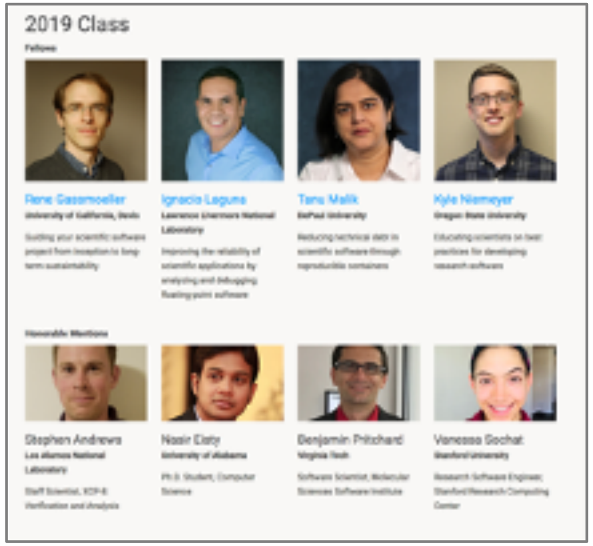
Honorable Mentions

Neal Davis
University of Illinois at Urbana-Champaign
Teaching Assistant Professor, Computer Science

Marc Henry de Frahan
National Renewable Energy Laboratory
Postdoctoral Researcher

Elsa Gonsiorowski
Lawrence Livermore National Laboratory
HPC I/O Specialist, Livermore Computing

Ying Li
Argonne National Laboratory
Argonne Scholar, Argonne Leadership Computing Facility



2019 Class

Fellows

Rene Gassmoeller
University of California, Davis
Guiding your scientific software project from inception to long-term sustainability

Ignacio Laguna
Lawrence Livermore National Laboratory
Improving the reliability of scientific applications by analyzing and debugging floating-point software

Tanu Malik
DePaul University
Reducing technical debt in scientific software through reproducible containers

Kyle Niemeyer
Oregon State University
Educating scientists on best practices for developing research software

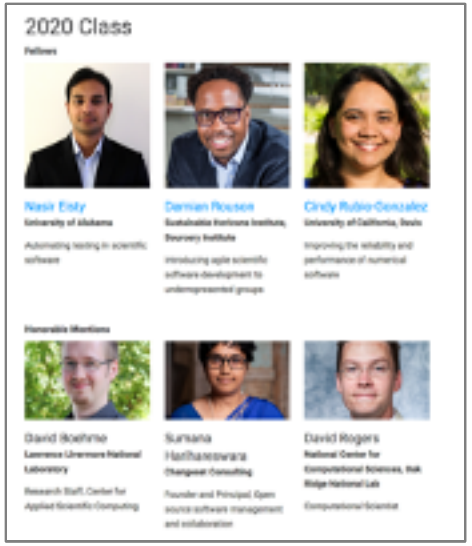
Honorable Mentions

Stephen Andrews
Los Alamos National Laboratory
Staff Scientist, XCP-8 Verification and Analysis

Nasir Eisty
University of Alabama
Ph.D. Student, Computer Science

Benjamin Pritchard
Virginia Tech
Software Scientist, Molecular Sciences Software Institute

Vanessa Sochat
Stanford University
Research Software Engineer, Stanford Research Computing Center



2020 Class

Fellows

Nasir Eisty
University of Alabama
Automating testing in scientific software

Damian Rouson
Sustainable Horizons Institute, Sourcery Institute
Introducing agile scientific software development to underrepresented groups

Cindy Rubio-Gonzalez
University of California, Davis
Improving the reliability and performance of numerical software

Honorable Mentions

David Boehme
Lawrence Livermore National Laboratory
Research Staff, Center for Applied Scientific Computing

Sumana Harihareswara
Changeset Consulting
Founder and Principal, Open source software management and collaboration

David Rogers
National Center for Computational Sciences, Oak Ridge National Lab
Computational Scientist

August 2020: Call opens for applications for the 2021 BSSw Fellowship Program.

Advancing Scientific Productivity through Better Scientific Software: Developer Productivity and Software Sustainability Report

Disruptive changes in computer architectures and the complexities of tackling new frontiers in extreme-scale modeling, simulation, and analysis present daunting challenges to software productivity and sustainability.

This newly released report explains the IDEAS approach, outcomes, and impact of work (in partnership with the ECP and broader computational science community).

Target readers are all those who care about the quality and integrity of scientific discoveries based on simulation and analysis. While the difficulties of extreme-scale computing intensify software challenges, issues are relevant across all computing scales, given universal increases in complexity and the need to ensure the trustworthiness of computational results.

BETTER SCIENTIFIC PRODUCTIVITY THROUGH BETTER SCIENTIFIC SOFTWARE: THE IDEAS REPORT

01/30/20

ting Project (ECP) provides a unique opportunity to advance computational science
-scale computing. However, disruptive changes in computer architectures and the
-scale modeling, simulation, and analysis present daunting challenges to the
inability of software artifacts.

DEAS project within ECP (called IDEAS-ECP) to foster and advance software
computational science, as a key aspect of improving overall scientific productivity.
es, and impact of work (in partnership with the ECP and broader computational

uality and integrity of scientific discoveries based on simulation and analysis. While
ify software challenges, issues are relevant across all computing scales, given
ensure the trustworthiness of computational results.

e

ECP EXASCALE COMPUTING PROJECT

ECP-U-RPT-2020-0001

Advancing Scientific Productivity through Better Scientific Software:
Developer Productivity and Software Sustainability Report

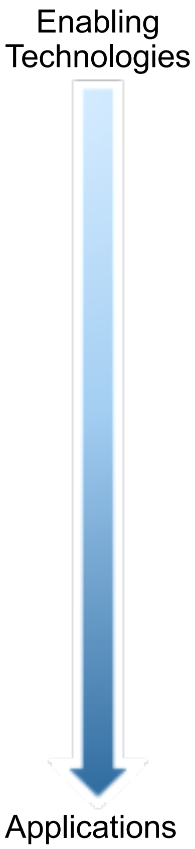
IDEAS-ECP Team and Collaborators

January 28, 2020

U.S. DEPARTMENT OF ENERGY | Office of Science   NNSA

https://exascaleproject.org/better-scientific-productivity-through-better-scientific-software-the-ideas-report

Speakers for this mini-symposium



10:55-11:15	Judy Hill, Oak Ridge National Laboratory Views on Software Sustainability from a Computing Facility Perspective
11:20-11:40	Mark Gates, University of Tennessee The Sustainability Lessons of the SLATE Project
11:45-12:05	Michal Habera , FEniCSX: A Sustainable Future for the FEniCS Project
12:10-12:30	Sunita Chandrasekaran, University of Delaware Training and Best Practices to Develop Portable Yet Performant Code
Lunch Break	
3:20-3:40	Dave Moulton Productivity and Sustainability in a Community-Driven Software Ecosystem for Watershed Science
3:45-4:05	Ken Jansen Software Sustainability Lessons from the Fluid Dynamics Community
4:10-4:30	Ben Pritchard Challenges and Best Practices in the Computational Molecular Sciences
4:35-4:55	Anshu Dubey Experiences with Productivity and Software Sustainability on LCF Machines

Topics to consider during this session

- What does software productivity and sustainability mean? Is there a difference between the application communities and the enabling technologies communities?

- What challenges exist in today’s landscape for ensuring software sustainability and developer productivity?

- What lessons learned and best practices can be shared between communities?

Why should software developers be concerned about sustainability and productivity?