Better (Small) Scientific Software Teams

SAND2017-12353C

Presented at

Better Scientific Software tutorial

SC17, Denver, Colorado

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

NNSA National Nuclear Security Administration
U.S. DEPARTMENT OF ENERGY | Office of Science
IDEAS productivity

License, citation, and acknowledgments

License and Citation

- Requested citation: Michael A. Heroux, Better (Small) Scientific Software Teams, tutorial, in SC '17: International Conference for High Performance Computing, Networking, Storage and Analysis, Denver, Colorado, 2017. DOI: TBA.

Acknowledgements

- This work was supported by the U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research (ASCR), and 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.
- 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.

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

Outline

- Small Team Models, Challenges.
- Agile workflow management for small teams
  - Intro to terminology and approaches
  - Overview of Kanban
  - Free tools: Trello, GitHub.

Small Teams

Ideas for managing transitions and steady work.

Small team interaction model

- Team composition:
  - Senior staff, faculty:
    - Stable presence, in charge of science questions, experiments.
    - Know the conceptual models well.
    - Spend less time writing code, fuzzy on details.
  - Junior staff, students:
    - Transient, dual focus (science results, next position).
    - Staged experience: New, experienced, departing.
    - Learning conceptual models.
    - Write most code, know details.

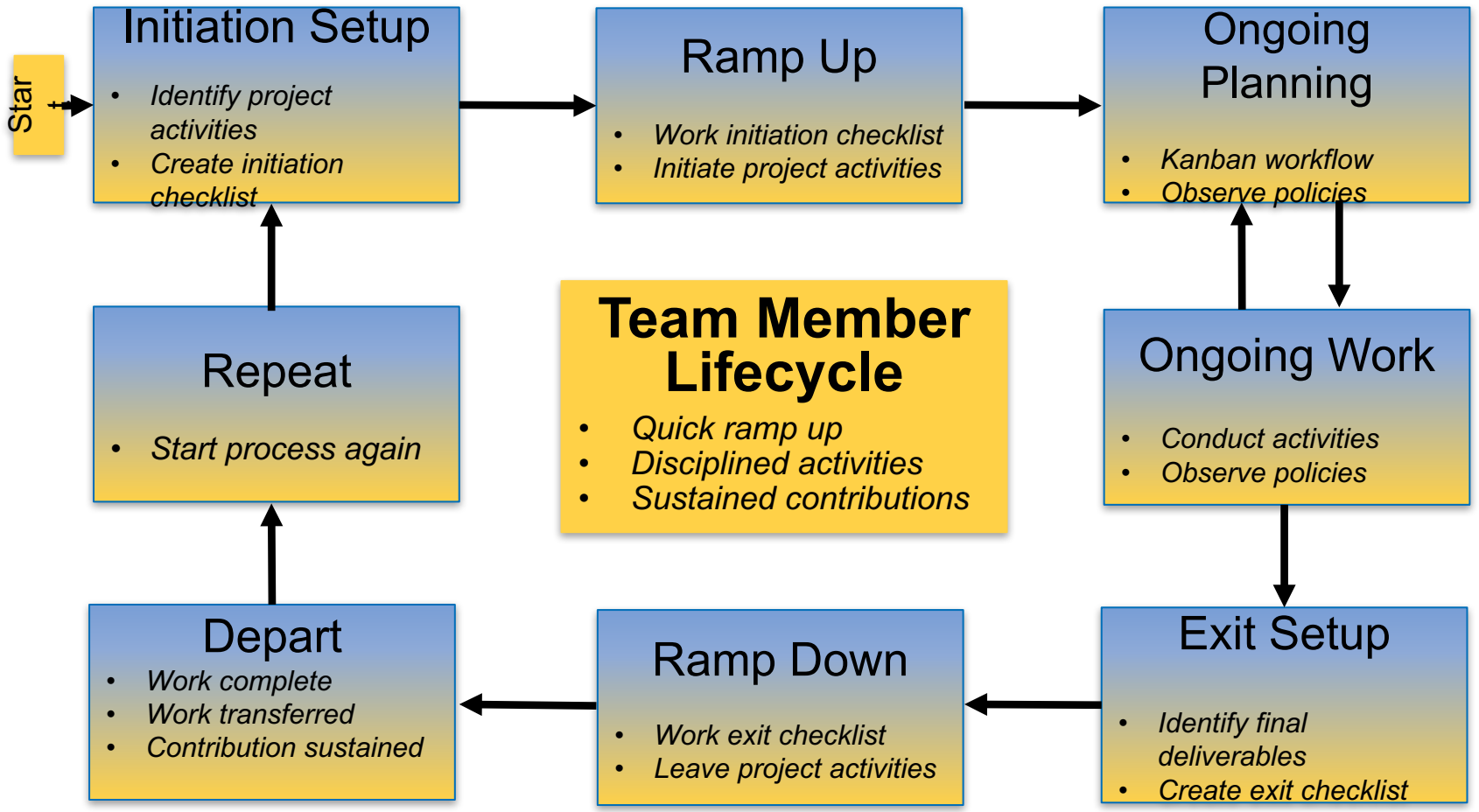
Large team challenges

- Composed of small teams (and all the challenges).
- Additional interaction challenges.
- Policies, regularly cultural exchanges important.

Small team challenges

- Ramping up new junior members:
  - Background.
  - Conceptual models.
  - Software practices, processes, tools.
- Preparing for departure of experienced juniors.
  - Doing today those things needed for retaining work value.
  - Managing dual focus.

Research Team Member Lifecycle



Checklists & Policies

Team Member Phase		
New Team Member	Steady Contributor	Departing Member
Checklist	Policies	Checklist

- New, departing team member checklists:
  - Example: Trilinos New Developer Checklist.
  - https://software.sandia.gov/trilinos/developer/sqp/checklists/index.html
- Steady state: Policy-driven.
  - Example: xSDK Community policies.
  - https://xsdk.info/policies/

Your checklists & policies?

- Checklist: New team member?
- Policies: Ongoing work?
- Checklist: Before someone departs?

Collaborative Work Management

Managing with Kanban

Managing issues: Fundamental software process

Continual improvement

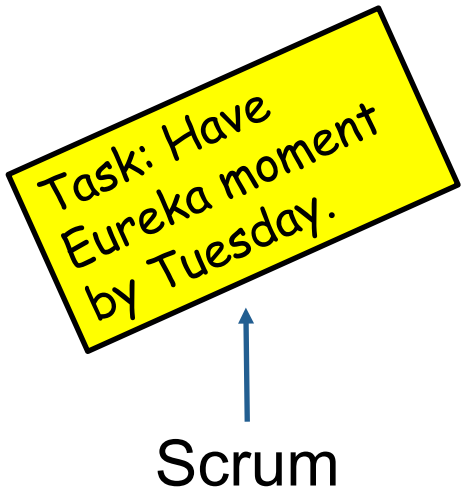
- Issue: Bug report, feature request
- Approaches:
  - Short-term memory, office notepad
  - ToDo.txt on computer desktop (1 person)
  - Issues.txt in repository root (small co-located team)
  - …
  - Web-based tool + Kanban (distributed, larger team)
  - Web-based tool + Scrum (full-time dev team)

Informal, less training

Formal, more training

Kanban principles

- Limit number of “In Progress” tasks
- Productivity improvement:
  - Optimize “flexibility vs swap overhead” balance. No overcommitting.
  - Productivity weakness exposed as bottleneck. Team must identify and fix the bottleneck.
  - Effective in R&D setting. Avoids a deadline-based approach. Deadlines are dealt with in a different way.
- Provides a board for viewing and managing issues



Basic Kanban

Backlog	Ready	In Progress	Done
• Any task idea • Trim occasionally • Source for other columns	• Task + description of how to do it. • Could be pulled when slot opens. • Typically comes from backlog.	• Task you are working on *right now.* • **The only kanban rule: Can have only so many "In Progress" tasks.** • Limit is based on experience, calibration. • **Key: Work is *pulled*. You are in charge!**	• Completed tasks. • Record of your life activities. • Rate of completion is your "velocity".

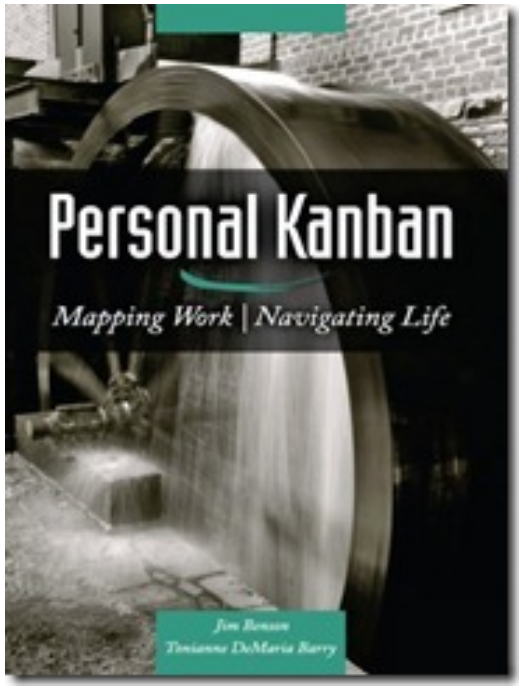
Notes:

- Ready column is not strictly required, sometimes called "Selected for development".
- Other common column: In Review
- Can be creative with columns:
  - Waiting on Advisor Confirmation.
  - Tasks I won't do.

Personal Kanban

- Personal Kanban: Kanban applied to one person.
  - Apply Kanban principles to your life.
  - Fully adaptable.

- Personal Kanban: Commercial book/website.
  - Useful, but not necessary.



http://www.personalkanban.com

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

Kanban tools

- Wall, whiteboard, blackboard: Basic approach.
- Software, cloud-based:
  - Trello, JIRA, GitHub Issues.
  - Many more.
- I use Trello (browser, iPhone, iPad).
  - Can add, view, update, anytime, anywhere.

Big question: How many tasks?

- Personal question.
- Approach: Start with 2 or 3. See how it goes.
- Use a freeway traffic analogy:
  - Does traffic flow best when fully packed? No.
  - Same thing with your effectiveness.
- Spend time consulting board regularly.
  - Brings focus.
  - Enables reflection, retrospection.
  - Use slack time effectively.
  - When you get out of the habit, start up again.

Importance of “In Progress” concept for you

- Junior community members:
  - Less control over task.
  - Given by supervisor.
- In Progress column: Protects you.
  - If asked to take on another task, respond:
    - Is this important enough to become less efficient?
    - Sometimes it is.

Key Team Management Elements

- **Checklists:**
  - Initiation, Transition, Exit
- **Policies:**
  - How team conducts its work
- **Issue tracking system:**
  - All work tracked, visible to team
  - Milestones: Aggregate related issues.
  - Kanban board
  - Regular meetings, updates

Samples from Collegeville Org: Policies, Initiation Checklist

Collegeville / Labora Private

Unwatch 9 | Star 0 | Fork 0

Code | Issues 25 | Pull requests 0 | Projects 1 | Wiki | Settings | Insights

Branch: master ▾ | Labora / TeamPolicy.md | Find file | Copy path

maherou Fix formatting — 51f30e2 a minute ago

1 contributor

21 lines (18 sloc) | 1.53 KB | Raw | Blame | History

Collegeville Research Team Policies

The following policies are meant to guide team members in their activities, establishing expectations for ongoing work.

1. Team members will conduct themselves in a professional manner, observing institutional policies given to them at student and faculty orientation.
2. Initiation, transition and exit events will be guided by creating and following an event checklist.
3. All work will be tracked in the organization issues-only repository Labora.
4. All work, notes and relevant content will be kept in a repository associated with the team GitHub organization.
5. Each team member will have an individual Collegeville repository: Lastname-Firstname-Work. This repo contains:
   i. Thesis or dissertation, as appropriate.
   ii. Annotated bibliography of resources.
   iii. Personal notes from project meetings and research activities.
6. If work is appropriate for one of the team repos, it will be retain there. Otherwise, it is kept in the team member's individual repo.
7. Team members will update project Kanban board prior to team meetings, more frequently if particularly active.
8. Exceptions to these policies are acceptable, but:
   i. Important exceptions should be approved before acting.
   ii. Other exceptions should mentioned at next team meeting or before.
   iii. Exceptions should be infrequent.
   iv. If an exception is frequent, actions or policies should be updated.
9. Any concerns not addressed by team policies should be discussed with Dr. Heroux.

Collegeville / Labora Private

Unwatch 9

Code | Issues 25 | Pull requests 0 | Projects 1 | Wiki | Setting

Neil Lindquist Initiation Checklist #17

Closed maherou opened this issue on Mar 31 · 0 comments

maherou commented on Mar 31 • edited by neil-lindquist

This is the initial checklist for Neil's initiation into the Collegeville research project:

- [x] Create a GitHub account (if you don't have one) and ask Dr Heroux to add you to the Collegeville organization.
- [x] Become a member of all appropriate repositories in the Collegeville organization.
- [x] Identify any new repos that should be created, especially if your research topic is new.
- [x] Learn LaTeX using the https://github.com/Collegeville/Scribe repository.
- [x] At least one of your repos will be a LaTeX collection that will contain your annotated bibliography and the starting point for at least one technical report, which will be an ongoing record of your progress.
- [x] Sign up for a Udacity online learning account at https://www.udacity.com, if you don't have one already. You will use Udacity for some of your introductory training.
- [x] Take the Udacity course Software Development Proces at https://classroom.udacity.com/courses/ud805.
- [x] Take the Udacity course How to Use Git and GitHub at https://classroom.udacity.com/courses/ud775.
- [x] Take the online courses in C++: http://www.cprogramming.com/tutorial/c++-tutorial.html and http://www.cplusplus.com/doc/tutorial
- [x] Redo CS200 lab exercises in C++

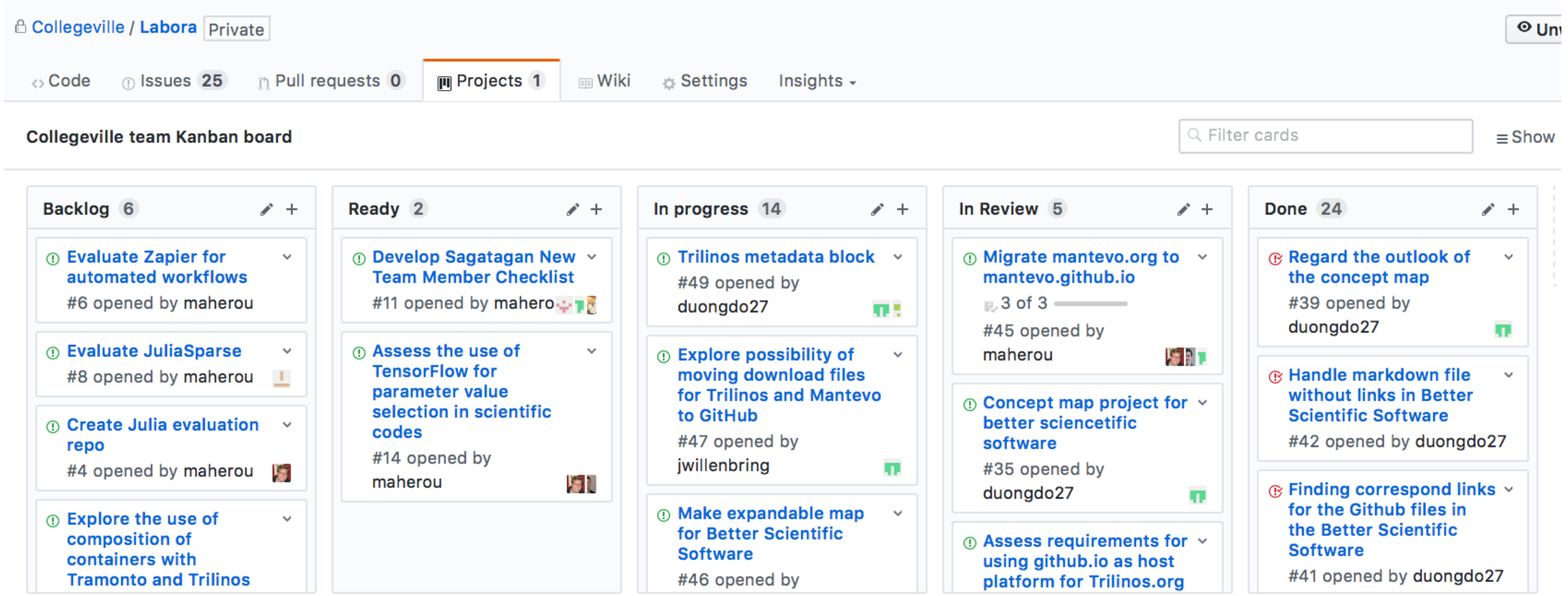
maherou assigned **maherou** and **neil-lindquist** on Mar 31

maherou added this to the **Neil Lindquist Initiation** milestone on Mar 31

maherou added to **Ready** in **Collegeville team Kanban board** on Mar 31

maherou moved from **Ready** to **In progress** in **Collegeville team Kanban board** on May 15

Samples from Collegeville Org: Kanban Board



IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

Team Management Example

Team Policy

Checklists

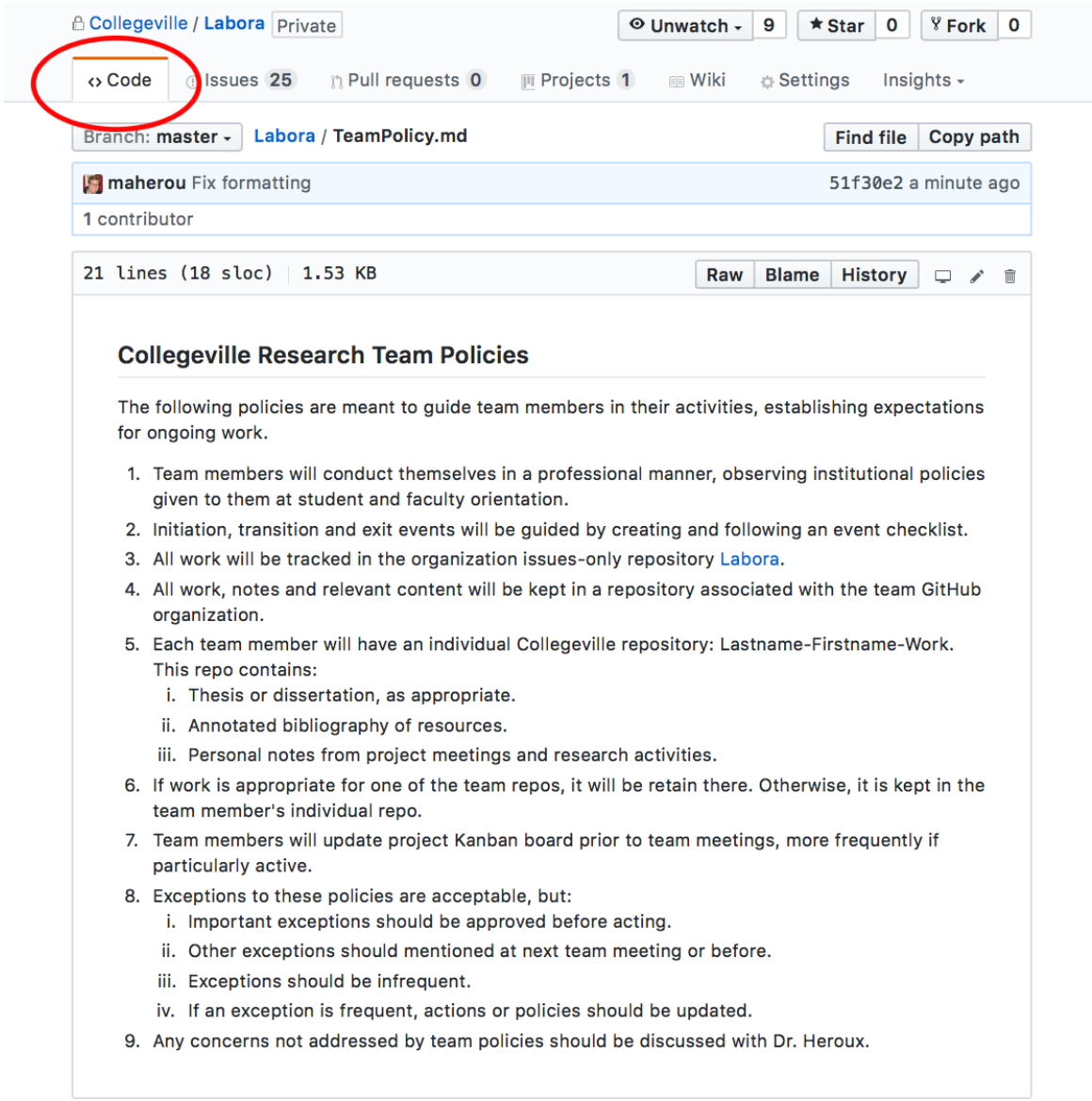
Kanban Board

Step 1: Create Issues-only GitHub repo

- Go to https://github.com/username
  - Example: https://github.com/maherou
- Create new repo:
  - Click on “+” (upper right).
  - Select New repository…
  - Give repo a name, e.g., **Issues**
  - Select Public. In real life, this repo is often private (requires $ or special status)
  - Init with README.
  - Don’t add .gitignore or license.
  - Click Create Repository.

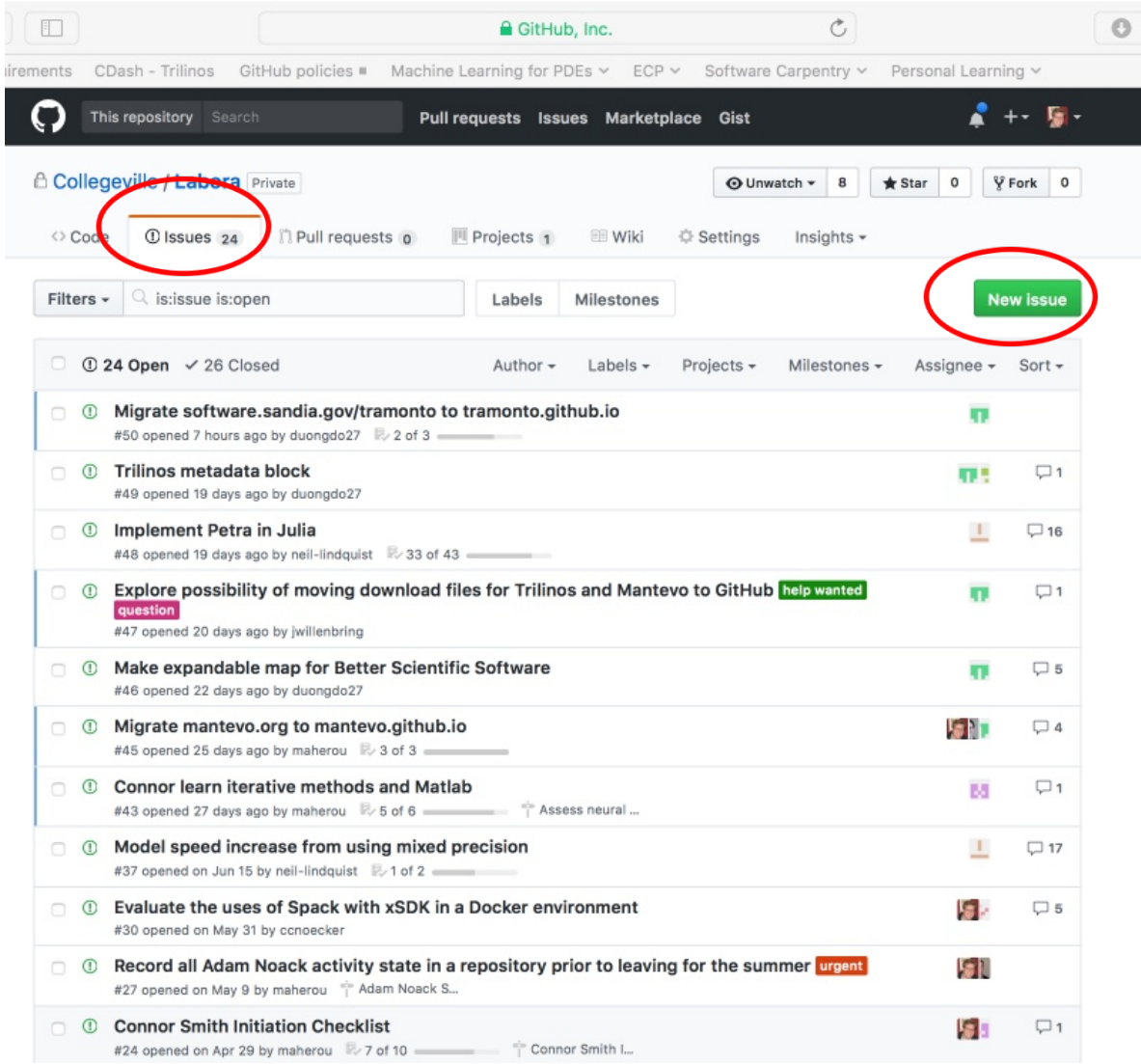
Step 2: Define Team Policy

- Create file:
  - Go to new repo: Issues.
  - Select <> Code tab.
  - Select Create new file TeamPolicy.md
- Questions to address:
  - How members support team?
  - How team supports members?
- Community version:
  - http://contributor-covenant.org
- Policy is living document:
  - Informal good practices added.
  - Avoidable bad situations addressed.



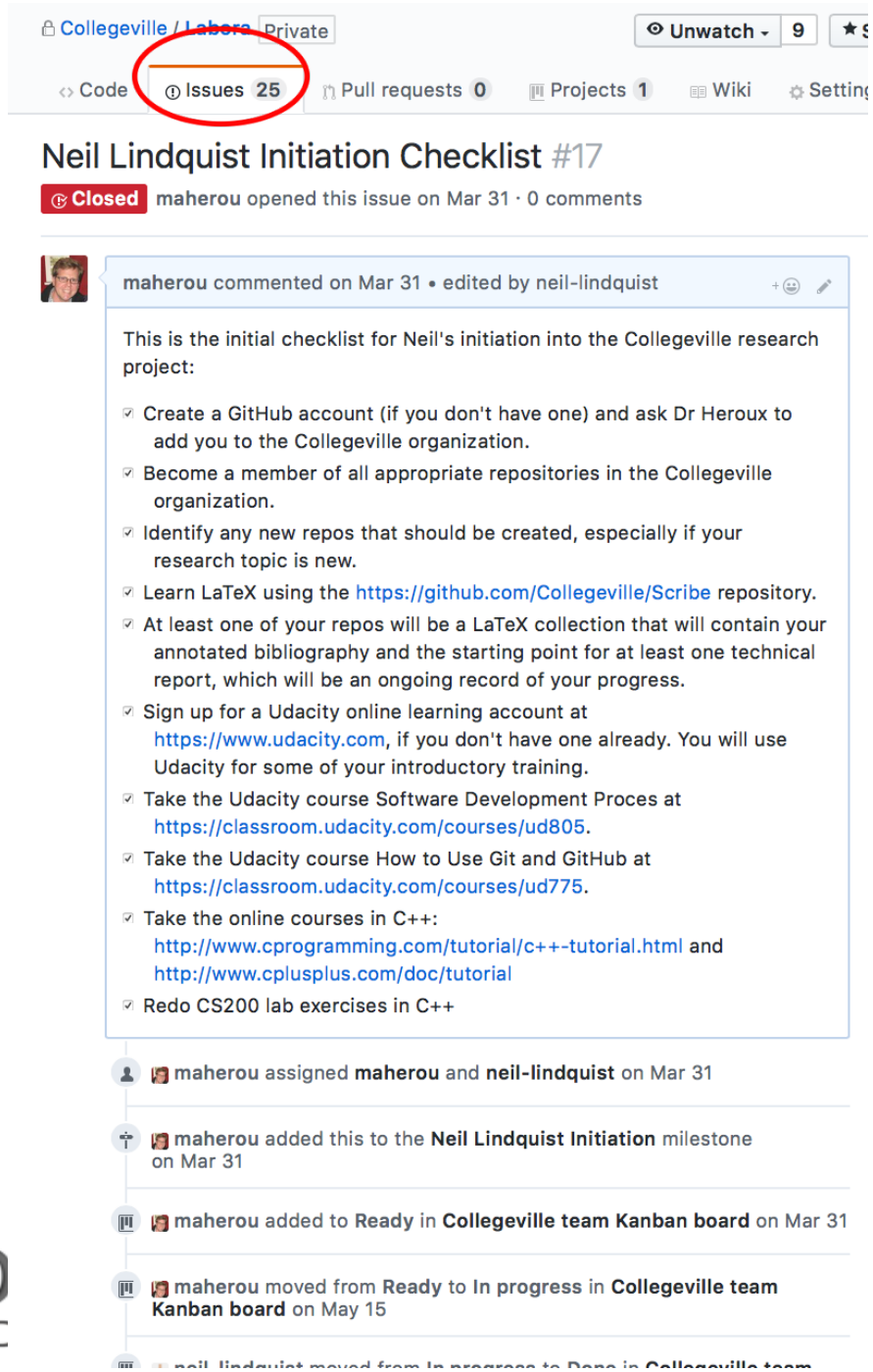
Step 3a: Create Issues

- Select the Issues tab.
- Click on New Issue.
- Type in task statement 1 (from list).
  - Type in title only.
- Click Submit new issue
- Repeat.



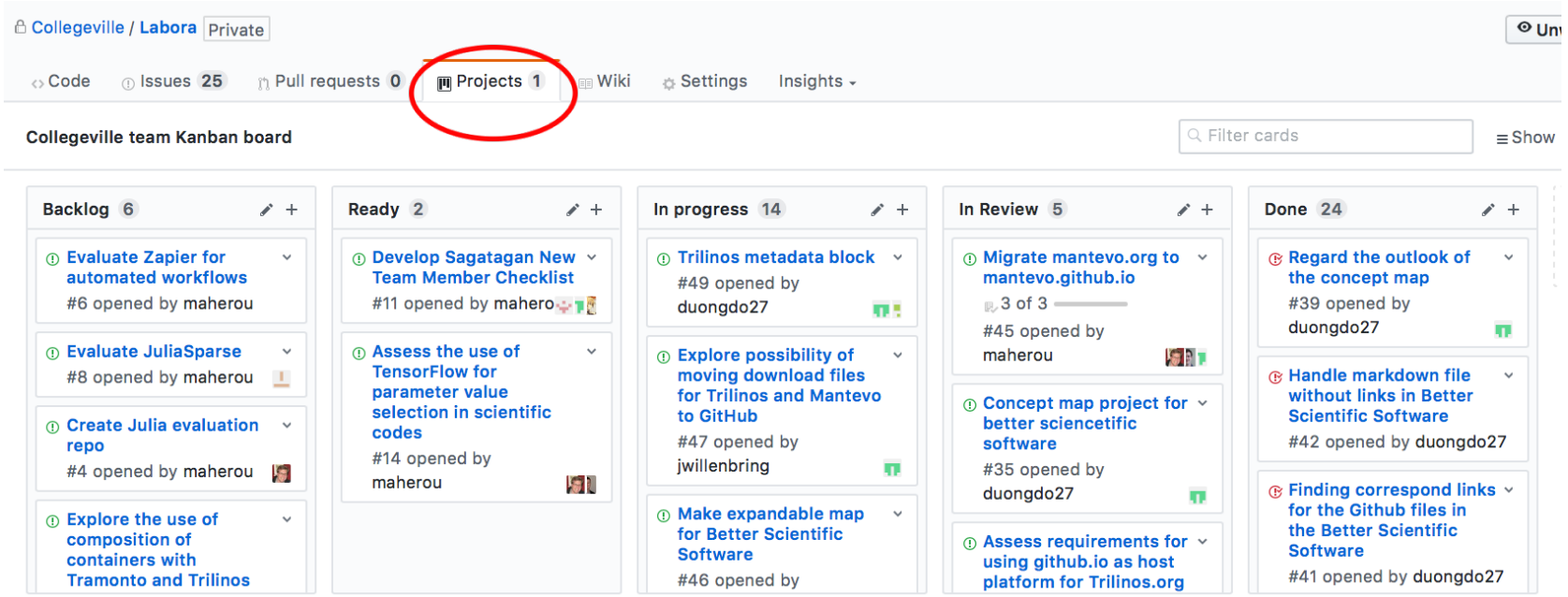
Step 3b: Create Initiation Checklist

- Select the Issues tab.
- Click on New Issue.
- Select a classmate.
- Type in title: Pat Evans Initiation Checklist
- Add checklist items:
  - Use syntax (note the spaces):
    - [ ] Description



Step 4: Create Kanban Board

- Select Projects tab
- Click New Project
- Use title
  - Team Kanban board
- Add these columns:
  - Backlog, Ready, In progress, In review, Done.
- Click on +Add cards (upper right).
  - Move each issue to the proper Kanban column



IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

Next Steps: Real Life

- Create a GitHub Org and set of repos for your team:
  - Each team member has an individual repo.
  - Each project has a repo.
  - One special repo for issues.
- Track all work:
  - Use checklists for initiation, exit, any big new effort.
  - Create Kanban board. Keep it current.
  - Aggregate related issues using milestones.
- Drive meetings using Kanban board.
- Adapt this approach to meet your needs.
- When you start to get sloppy, get back on track.

Other resources

The Agile Samurai: How Agile Masters Deliver Great Software (Pragmatic Programmers), Jonathan Rasmusson. Excellent, readable book on Agile methodologies. https://www.amazon.com/Agile-Samurai-Software-Pragmatic-Programmers/dp/1934356581

Also available on Audible.

Code Complete, Steve McConnell. Great text on software.

Construx website has large collection of content.

Agenda

Tutorial evaluation form: *TBA*

Time	Topic	Speaker
8:30am-8:45am	Why effective software practices are essential for CSE projects	David E. Bernholdt, ORNL
8:45am-9:15am	Introduction to software licensing	David E. Bernholdt, ORNL
9:15am-9:45am	Better (small) scientific software teams	Michael A. Heroux, SNL
9:45am-10:00am	Improving Reproducibility Through Better Software Practices	Michael A. Heroux, SNL
10:00am-10:30am	*Break*	
10:30am-10:45am	Testing of HPC Scientific Software: Introduction	Alicia M. Klinvex, SNL
10:45am-11:15am	Verification	Anshu Dubey, ANL
11:15am-11:45am	Evaluating project testing needs	Anshu Dubey, ANL
11:45am-12:00pm	Code coverage demo and CI demo	Alicia M. Klinvex, SNL

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

Improving Reproducibility Through Better Software Practices

Presented at

Better Scientific Software tutorial

SC17, Denver, Colorado

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

License, citation, and acknowledgments

License and Citation

- Requested citation: Michael A. Heroux, Improving Reproducibility Through Better Software Practices, tutorial, in SC '17: International Conference for High Performance Computing, Networking, Storage and Analysis, Denver, Colorado, 2017. DOI: TBA.

Acknowledgements

- This work was supported by the U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research (ASCR), and 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.
- 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.

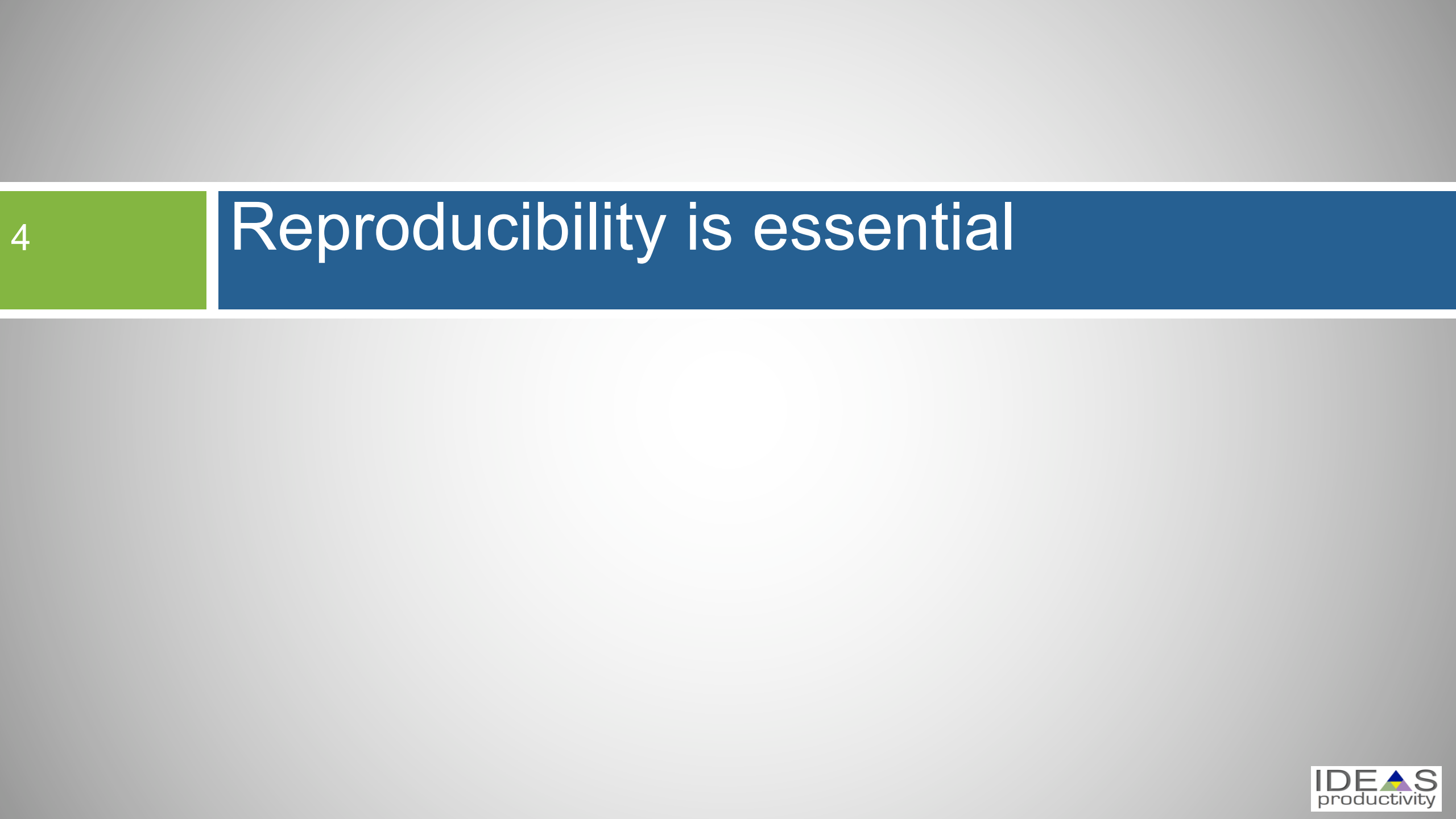
IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

Outline

- Increasing focus on reproducibility.
- Role of better software practices.
- Publication requirements.
- Trustworthiness at Scale.
- Personal Productivity Commitment.

Reproducibility is essential



Reproducibility

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

- NY Times highlights "problems".
- Only one of many cited examples.
- HPC has been spared this "spotlight" (so far).

- Lots of activity:
  - AAAS, ACM initiatives.
  - PPoPP, Supercomputing 2017.
- But what is reproducibility?

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

Productivity and Sustainability

What do we mean?

Objectives

- Productivity – Output per unit input.
- Sustainability – The future cost of usability.
- Goals for today:
  - Learn how to improve
    - Developer productivity.
    - Software sustainability.
  - For the purposes of better scientific productivity,
  - Using tools, processes and practices.

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.

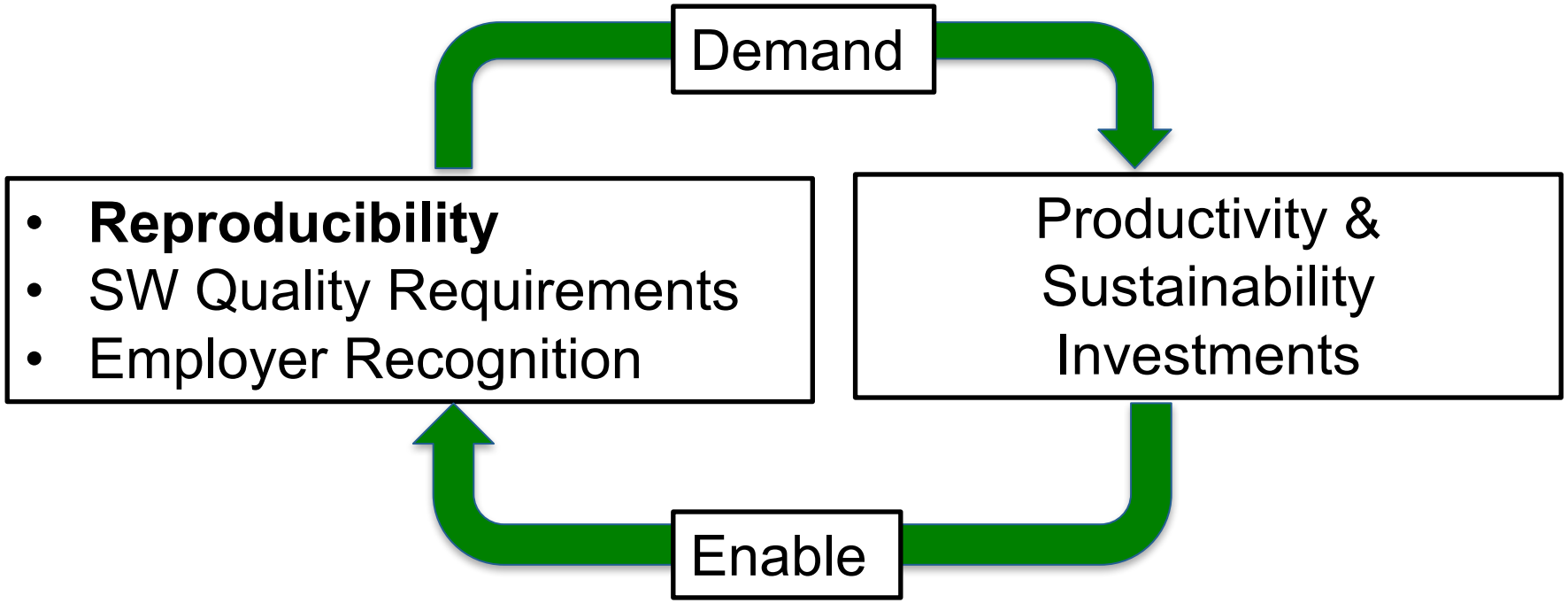
Scenario: (3 years later)
After investing in **software sustainability improvements**, you are on time in developing **several** sophisticated new models in your software that you want to use for results in upcoming papers.

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 To Change



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.

Reproducibility Terminology

V. Stodden, D. H. Bailey, J. Borwein, R. J. LeVeque, W. Rider, and W. Stein. 2013. Setting the Default to Reproducible: Reproducibility in Computational and Experimental Mathematics. (2013). https://icerm.brown.edu/tw12-5-rcem/icerm_report.pdf

- **Reviewable Research.** The descriptions of the research methods can be independently assessed and the results judged credible. (This includes both traditional peer review and community review, and does not necessarily imply reproducibility.)
- **Replicable Research.** Tools are made available that would allow one to duplicate the results of the research, for example by running the authors' code to produce the plots shown in the publication. (Here tools might be limited in scope, e.g., only essential data or executables, and might only be made available to referees or only upon request.)
- **Confirmable Research.** The main conclusions of the research can be attained independently without the use of software provided by the author. (But using the complete description of algorithms and methodology provided in the publication and any supplementary materials.)
- **Auditable Research.** Sufficient records (including data and software) have been archived so that the research can be defended later if necessary or differences between independent confirmations resolved. The archive might be private, as with traditional laboratory notebooks.
- **Open or Reproducible Research.** Auditable research made openly available. This comprised well-documented and fully open code and data that are publicly available that would allow one to (a) fully audit the computational procedure, (b) replicate and also independently reproduce the results of the research, and (c) extend the results or apply the method to new problems.

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.

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

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 (Untested, rare)

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).***

Coming to Your World Soon: Reproducibility Requirements

- 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

How can you prepare?

SC17 Reproducibility Initiative

- Two appendices:
  - Artifact description (AD).
    - Blue print for setting up your computational experiment.
    - Makes it easier to rerun computations in future.
  - Computational Results Analysis (CRA).
    - Targets ”boutique” environments.
    - Improves trustworthiness when re-running hard, impossible.
- Details:
  - http://sc17.supercomputing.org/submitters/technical-papers/reproducibility-initiatives-for-technical-papers/

Improving Trustworthiness at Scale

What if we can re-run a computational experiment?

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.

Sources for CRA 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?

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.

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

(Personal) Productivity++ Initiative

Ask: *Is My Work _____ ?*

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

Summary

- Reproducibility demands are coming.
  - Conferences first, journals slower.
- HPC software is particularly challenging:
  - Hardware variation.
  - Code optimization.
  - Dynamic parallelism.
- Better software practices:
  - Improve chances for reproducibility.
  - Lower its cost.
- Many tools emerging to enable reproducibility.

Other resources

Editorial: ACM TOMS Replicated Computational Results Initiative. Michael A. Heroux. 2015. *ACM Trans. Math. Softw.* 41, 3, Article 13 (June 2015), 5 pages. DOI: http://dx.doi.org/10.1145/2743015

Enhancing Reproducibility for Computational Methods. Victoria Stodden, Marcia McNutt, David H. Bailey, Ewa Deelman, Yolanda Gil, Brooks Hanson, Michael A. Heroux, John P.A. Ioannidis, Michela Taufer Science (09 Dec 2016), pp. 1240-1241

Agenda

Tutorial evaluation form: *TBA*

Time	Topic	Speaker
8:30am-8:45am	Why effective software practices are essential for CSE projects	David E. Bernholdt, ORNL
8:45am-9:15am	Introduction to software licensing	David E. Bernholdt, ORNL
9:15am-9:45am	Better (small) scientific software teams	Michael A. Heroux, SNL
9:45am-10:00am	Improving Reproducibility Through Better Software Practices	Michael A. Heroux, SNL
10:00am-10:30am	*Break*	
10:30am-10:45am	Testing of HPC Scientific Software: Introduction	Alicia M. Klinvex, SNL
10:45am-11:15am	Verification	Anshu Dubey, ANL
11:15am-11:45am	Evaluating project testing needs	Anshu Dubey, ANL
11:45am-12:00pm	Code coverage demo and CI demo	Alicia M. Klinvex, SNL

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT