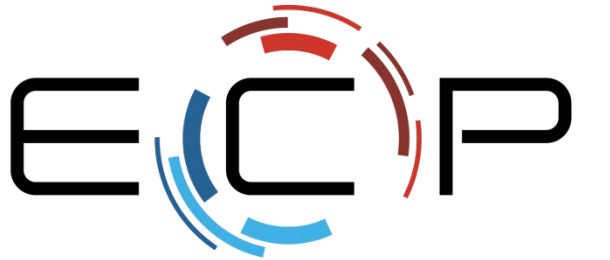
SW Engineering for Computational Science & Engineering
What Can Work and What Will Not

The 2017 International Workshop on Software Engineering for High Performance Computing in Computational and Data-Enabled Science and Engineering (SE-CoDeSE 2017)

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

http://www.users.csbsju.edu/~mheroux/HerouxSE4CSE.pdf



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

Acknowledgments

- 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

- My Perspective
- A Bit about “Barely Sufficient”
- Small Team Models, Challenges
- Agile workflow management for small teams
  - Intro to terminology and approaches
  - Overview of Kanban
  - Checklists, Policies, Issue tracking system
- Planning: Simple better than none.
- Wrap up: Opportunities for you.

My Perspective

- Regarding observations on opportunities to improve:
  - *More like a psychologist than expert.*

- Regarding software tools, processes, practices improvements:
  - *More like a carpenter than expert.*

Entry-level Scientific Software Developer Posting

Apply for Job

Job ID	659272	Full/Part Time	Full-Time
Location	Albuquerque, NM	Regular/Tempor…	Temporary

☆ Add to Favorite Jobs

Email this Job

Qualifications We Require

- PhD, conferred within 5 years prior to employment, in physics, engineering, mathematics, computer science, or a related area
- Experience with computational formulations and solution methods for two or more of the following: electromagnetics, compressible fluid flow, magnetohydrodynamics (MHD) or multi-fluid plasma physics
- Experience with finite element methods for PDEs
- Experience with software development in C++
- Experience in a collaborative research environment on problems comprising diverse application domains
- Evidence of strong research achievements in relevant technical areas as demonstrated in the form of technical publications, presentations, software tools, and/or knowledge of applications

Qualifications We Desire

- Demonstrated strong background and expertise in numerical methods for partial differential equations with particular emphasis on hyperbolic problems and compressible plasma systems
- Experience with computational models for magnetic confinement fusion applications or magnetic implosion dynamics with relevance to z-pinch devices
- Some experience with hybrid continuum / kinetic algorithms and/or modeling for plasma physics systems
- Some experience in the application of uncertainty quantification (UQ) methods to complex computational models
- Strong proficiency with scientific software development
- Excellent communication skills

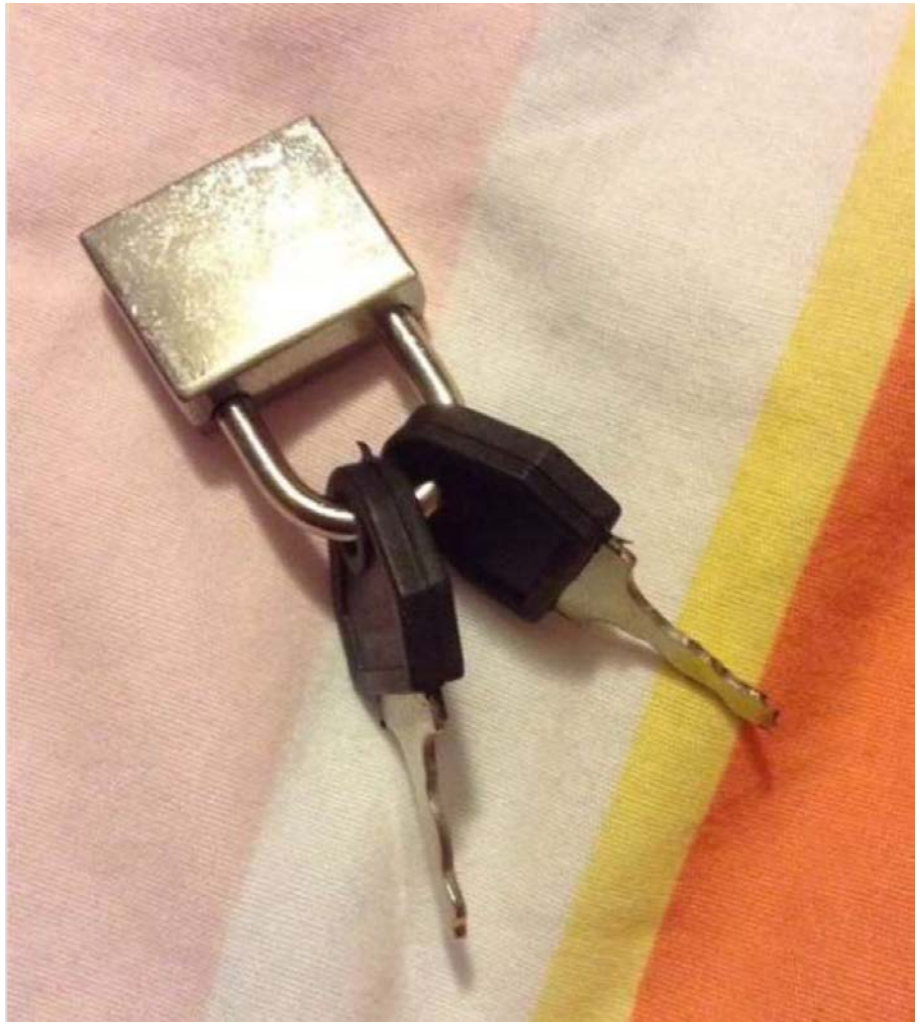
- Hi quality SW ideal:
  - Deep domain knowledge.
  - Deep SW Eng knowledge.
- Possible? Maybe, but hard.
- Next best?
  - Deep domain + some SE.
  - Deep SE + some domain

Observation: Mostly unsuccessful.

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

CSE & Formal (Heavy) Software Methodologies: Troubled History



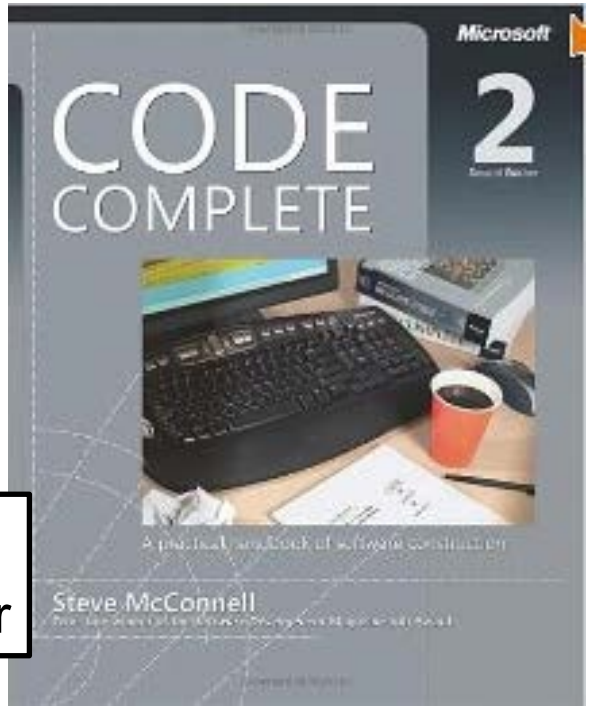
- Cray (1990):
  - Formal Waterfall Method.
- DOE ASCI (2000):
  - CMMI
- Failed to follow own process: Elicit requirements.

CSE Complete: Useful "Overhead"

- Code Complete: Ultimate value is code.
  - Should we only write code?
  - Some non-coding activities improve code.

> *"Give me six hours to chop down a tree and I will spend the first four sharpening the axe."*
> Abraham Lincoln

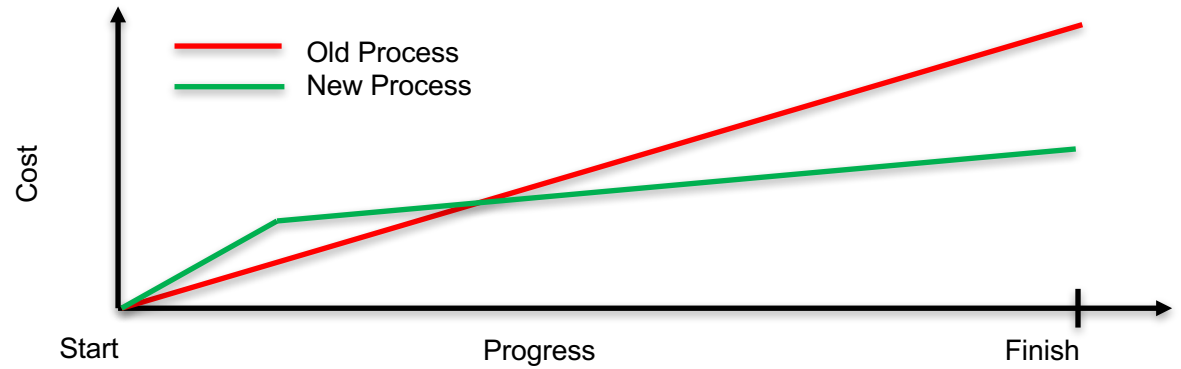
> *"Plans are worthless, but planning is everything."*
> Dwight D. Eisenhower



- CSE Complete: Ultimate value is CSE.
  - Question: What non-coding activities improve CSE?
- Barely Sufficient: Emerges from this philosophy

Incremental Improvement

- Elicit, analyze, prototype, test, revise, deploy. Repeat.
- Realistic: There is a cost.
  - Startup: Overhead
  - Payoff: Best if soon, clear



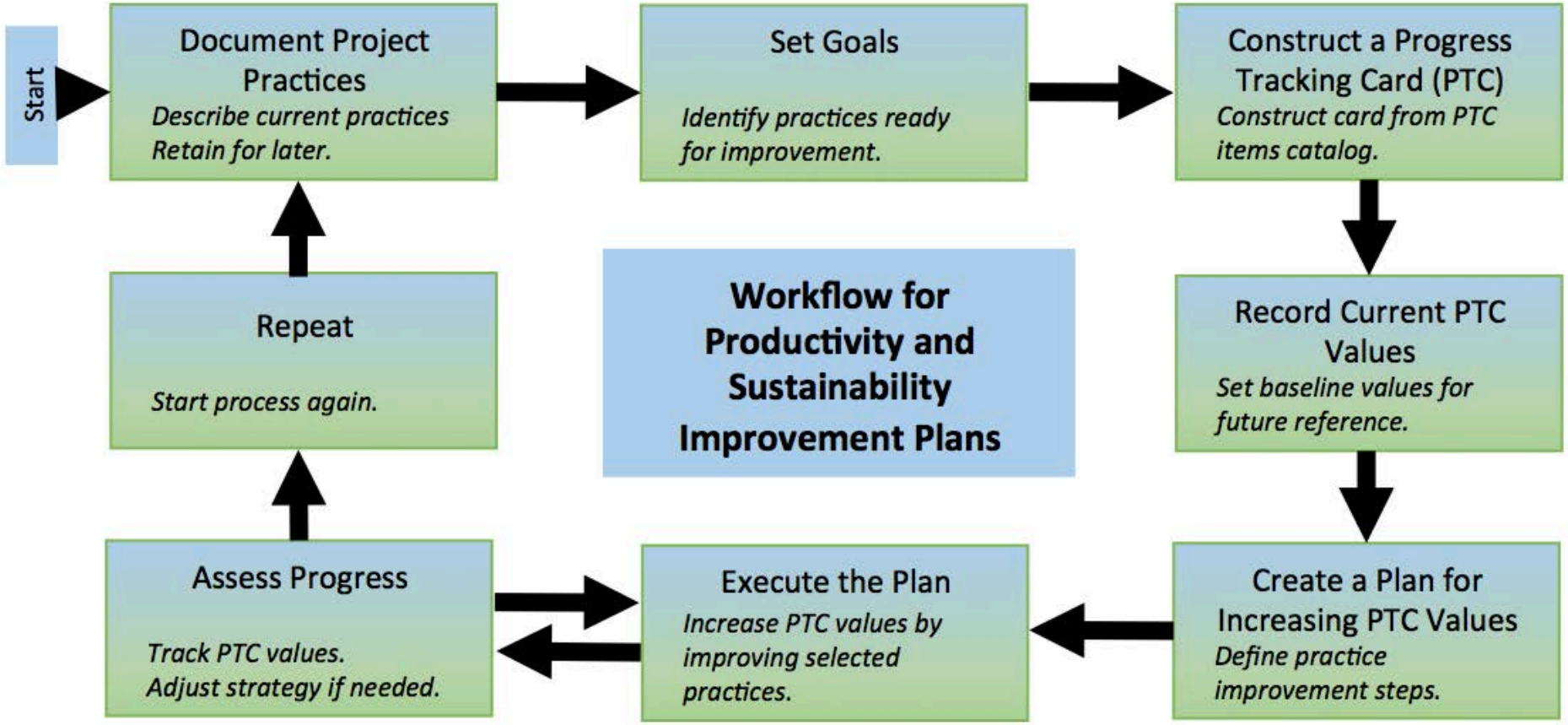
- Working model:
  - Reserve acceptable time/effort for improvement.
  - ***Improve how you do your work while achieving another goal.***
  - Example: Deliver new thread-scalable ILU under new unit testing framework.

Productivity and Sustainability Improvement Planning Tools

Tools for helping a software team to increase software quality while decreasing the effort, time, and cost to develop, deploy, maintain, and extend software over its intended lifetime.

PSIP templates & instructions: https://github.com/betterscientificsoftware/PSIP-Tools

Practice: Test Coverage	Score (0 – 5):
Score Descriptions	
0	No unit testing.
1	Unit testing used for one example refactored class.
2	Two developers using unit testing process.
3	Unit testing developer documentation completed.
4	All developers trained to write unit tests.
5	New policy established: All refactored code is covered by unit tests.



Productivity and Sustainability Improvement Plan (PSIP): a lightweight iterative workflow to identify, plan, and improve selected practices of a software project.

IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

Some SE practices that work for CSE

Team Management Elements

Checklists, Policies, Issue Tracking System

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

Small Teams

Ideas for managing transitions and ongoing 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.

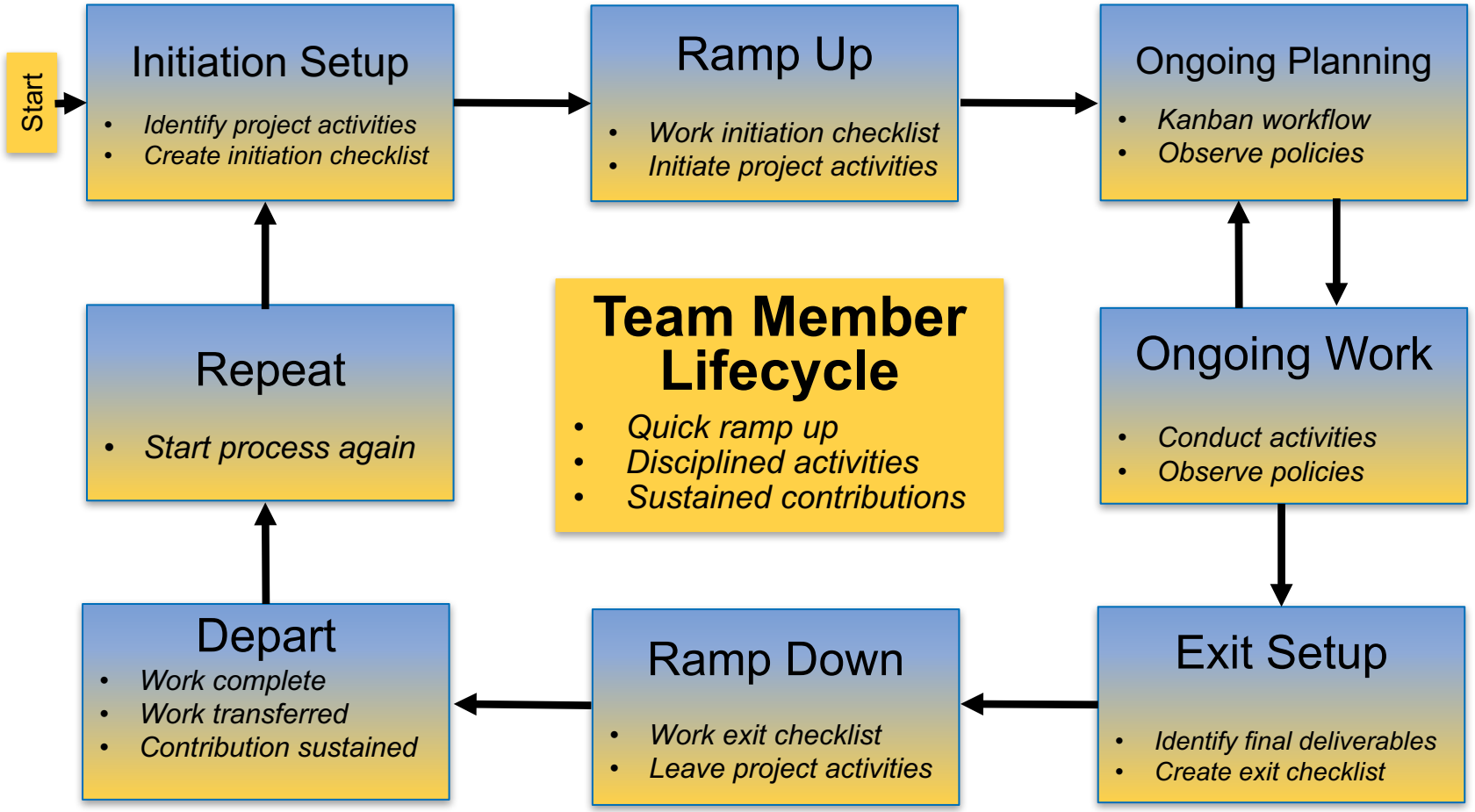
Large team challenges

- Composed of small teams (and all the challenges).
- Additional interaction challenges.
- Policies, regularly cultural exchanges important.
- “Team of Teams” approach is very attractive.

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



IDEAS productivity

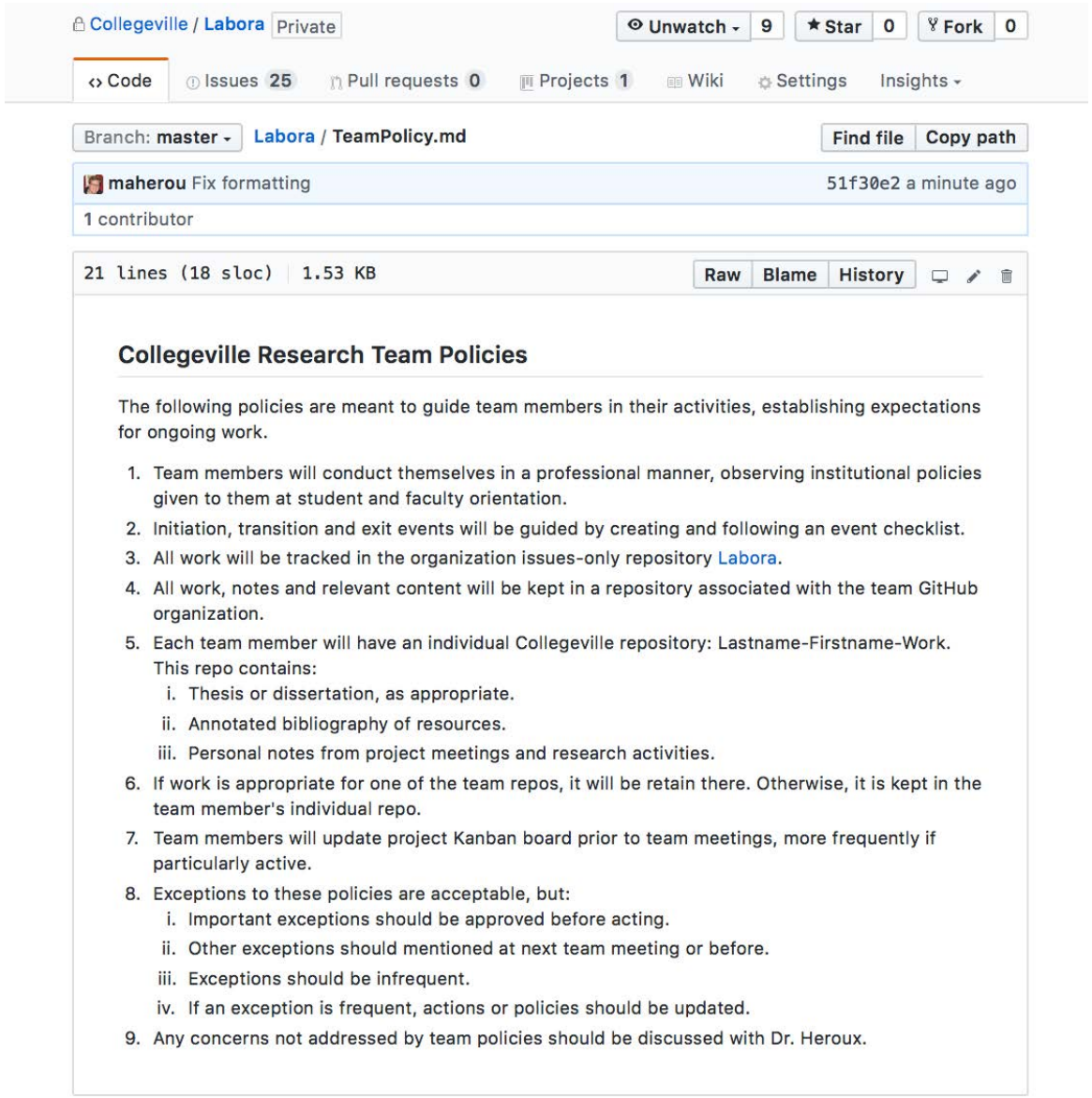
ECP EXASCALE COMPUTING PROJECT

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/

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

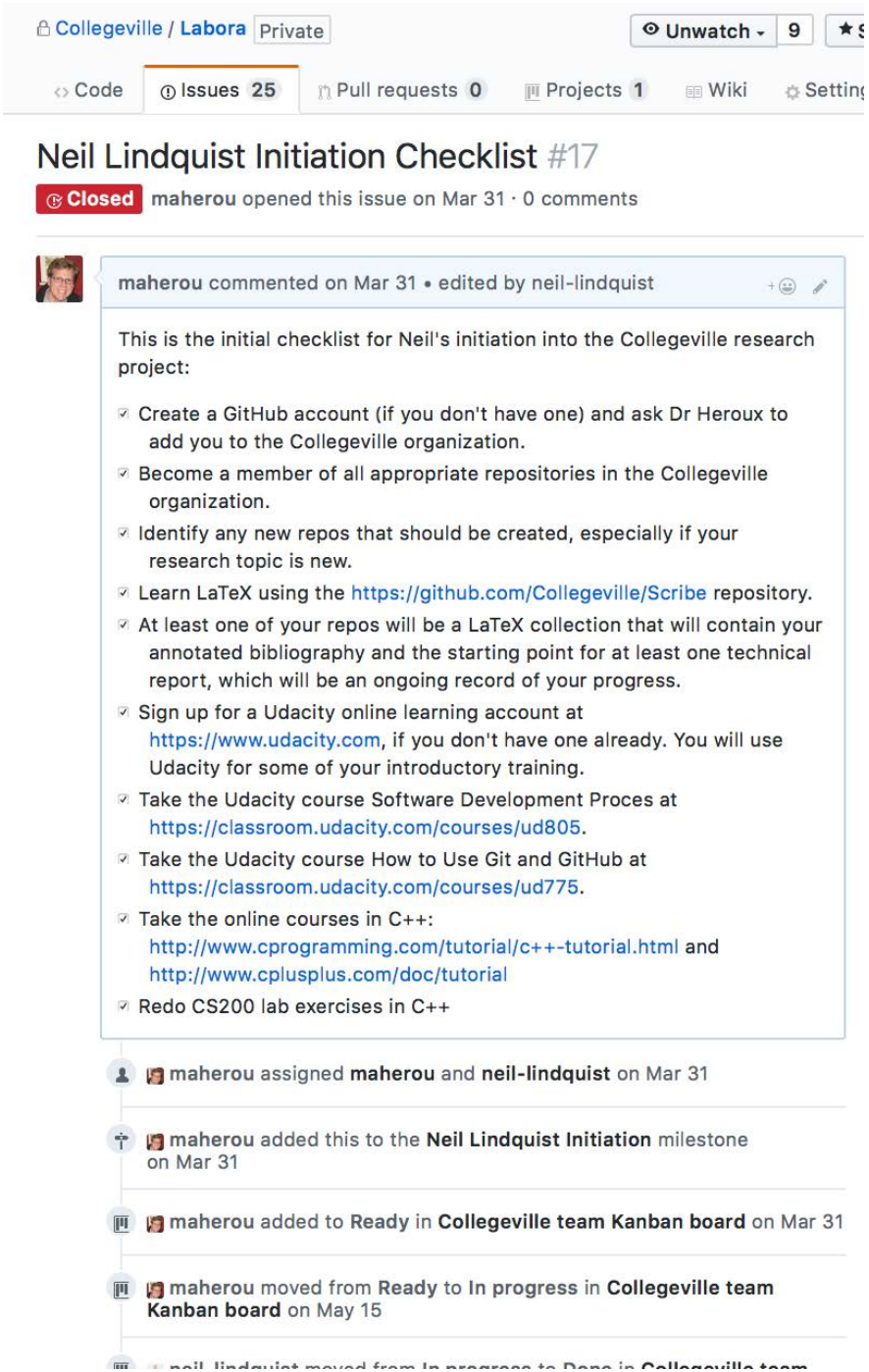
1 contributor

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

Collaborative Work Management

Managing with Kanban

Managing issues: Fundamental software process

Continual improvement

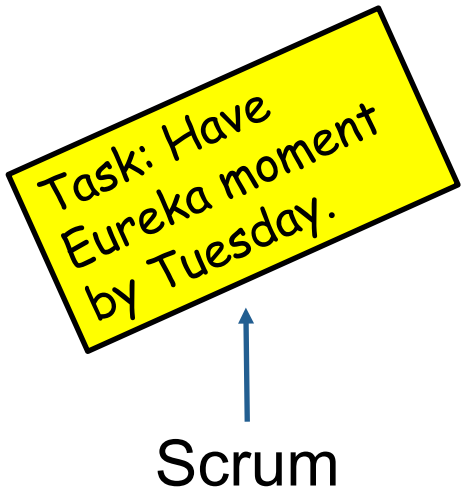
- Issue: Bug report, feature request
- Approaches:
  - Short-term memory, office notepad
  - ToDo.txt on computer desktop (1 person)
  - Issues.txt in repository root (small co-located team)
  - …
  - Web-based tool + Kanban (distributed, larger team)
  - Web-based tool + Scrum (full-time dev team)

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

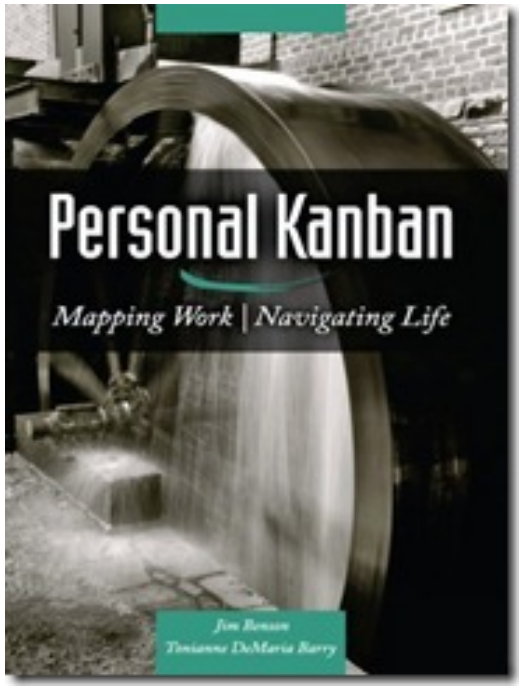
Backlog	Ready	In Progress	Done
• Any task idea • Trim occasionally • Source for other columns	• Task + description of how to do it. • Could be pulled when slot opens. • Typically comes from backlog.	• Task you are working on *right now.* • **The only kanban rule: Can have only so many "In Progress" tasks.** • 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

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.

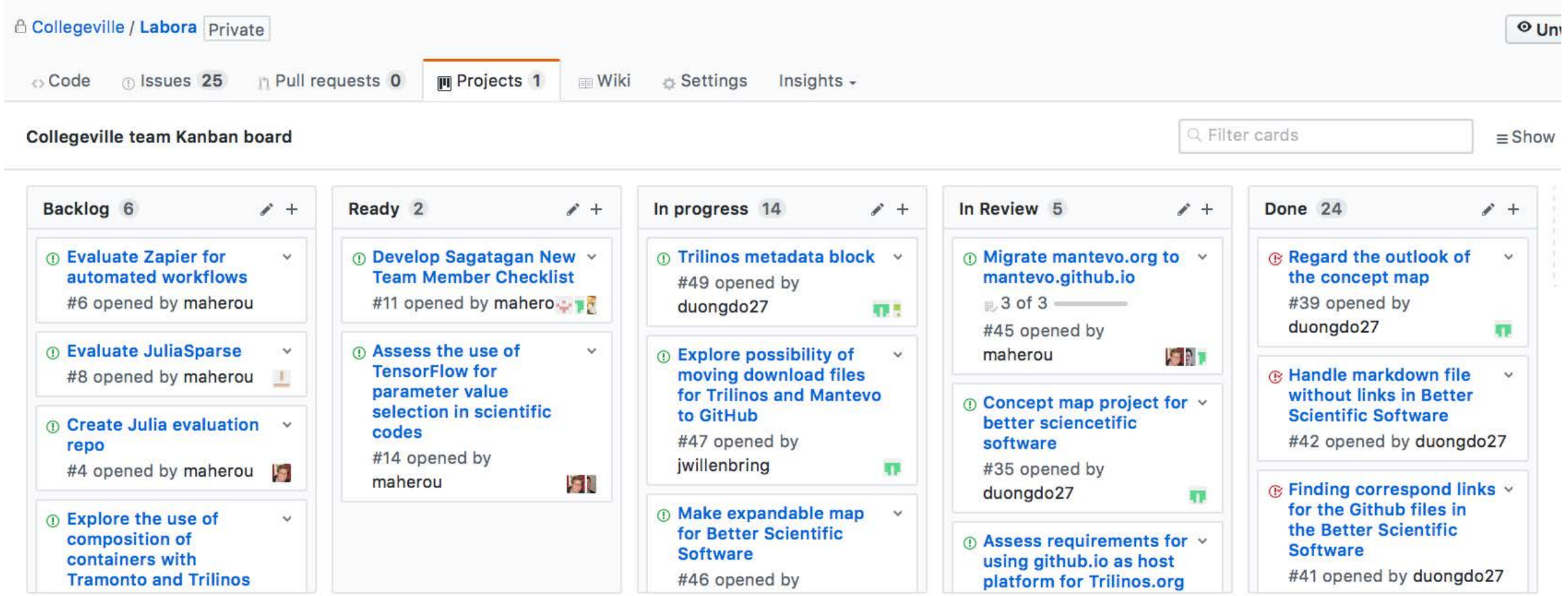
Big question: How many tasks?

- 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 typical situation:
  - 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.

Samples from Collegeville Org: Kanban Board



IDEAS productivity

ECP EXASCALE COMPUTING PROJECT

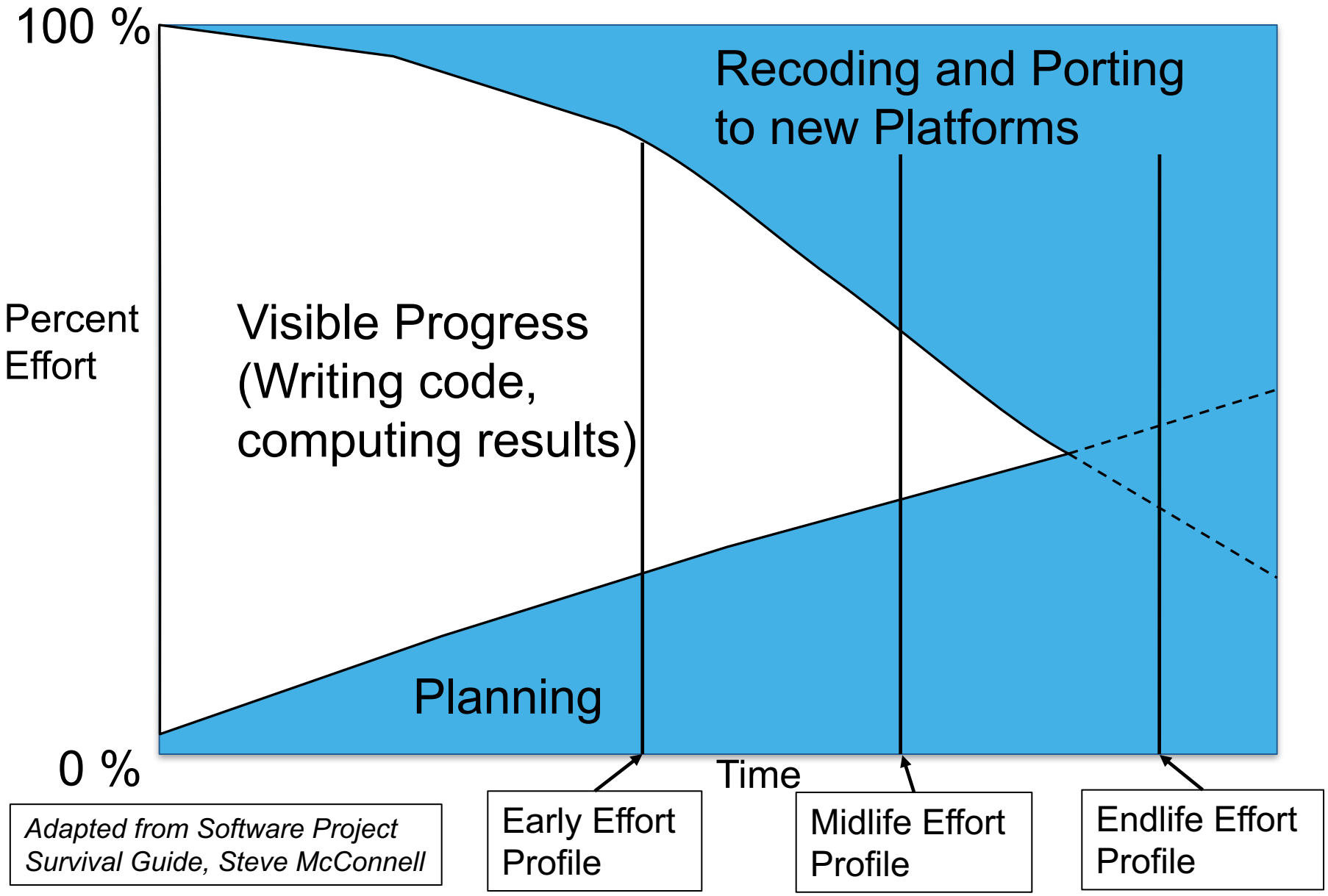
What about Scrum?

- Scrum: A popular process framework, widely and successfully used.
- Could it work for you? Maybe.
- Emphasis: Regular sprints, reviews, retrospectives, stories, backlog, product owner, scrum master, and more.
- Most people: Scrum-but.
- Alternative: Kanban-and.

- https://www.scrumalliance.org
- Kanban and Scrum -- Making the Most of Both, by Henrik Kniberg and Mattias Skarin

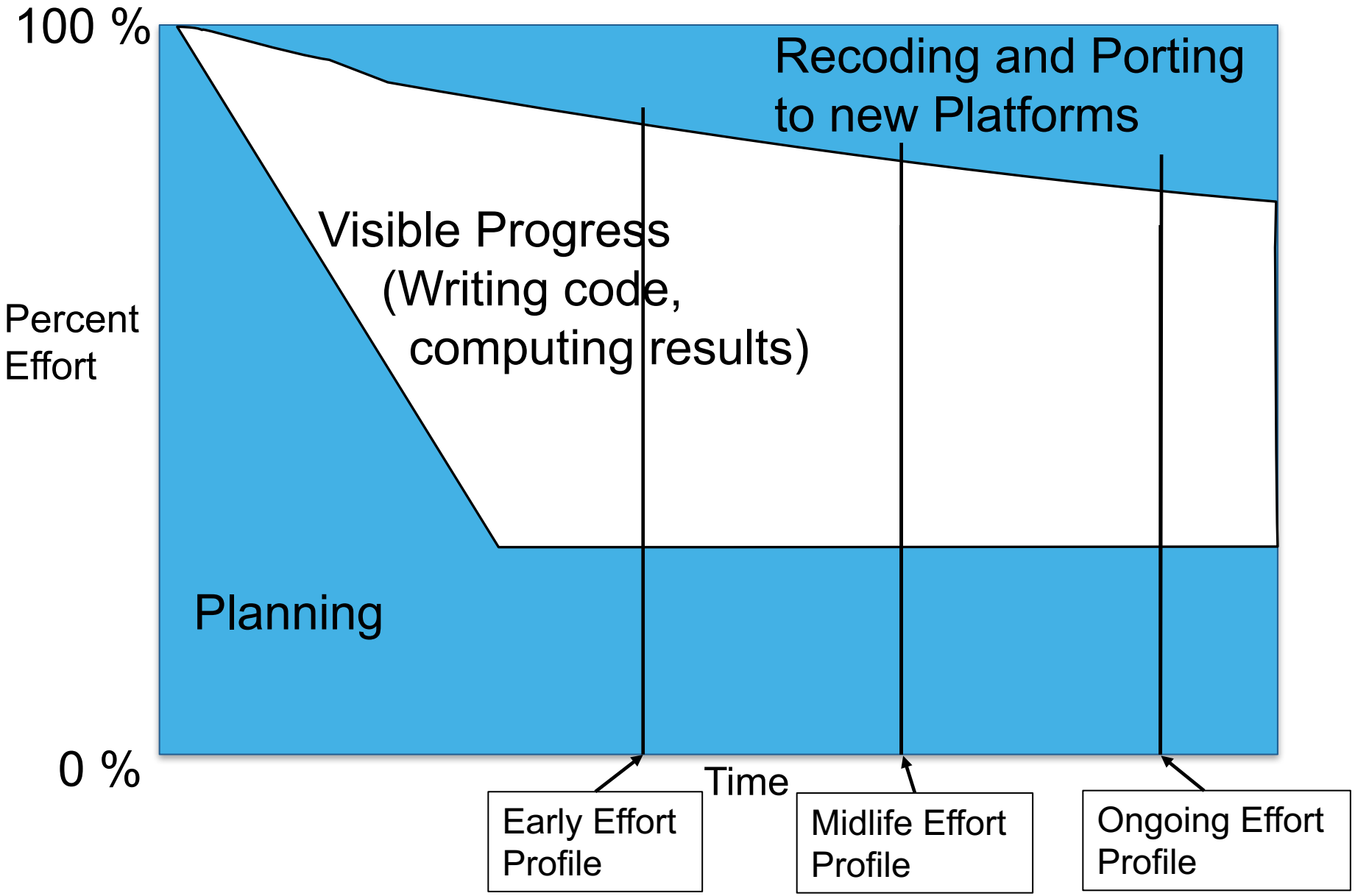
Productivity Improvement: Planning

Code-and-Fix Development Approach



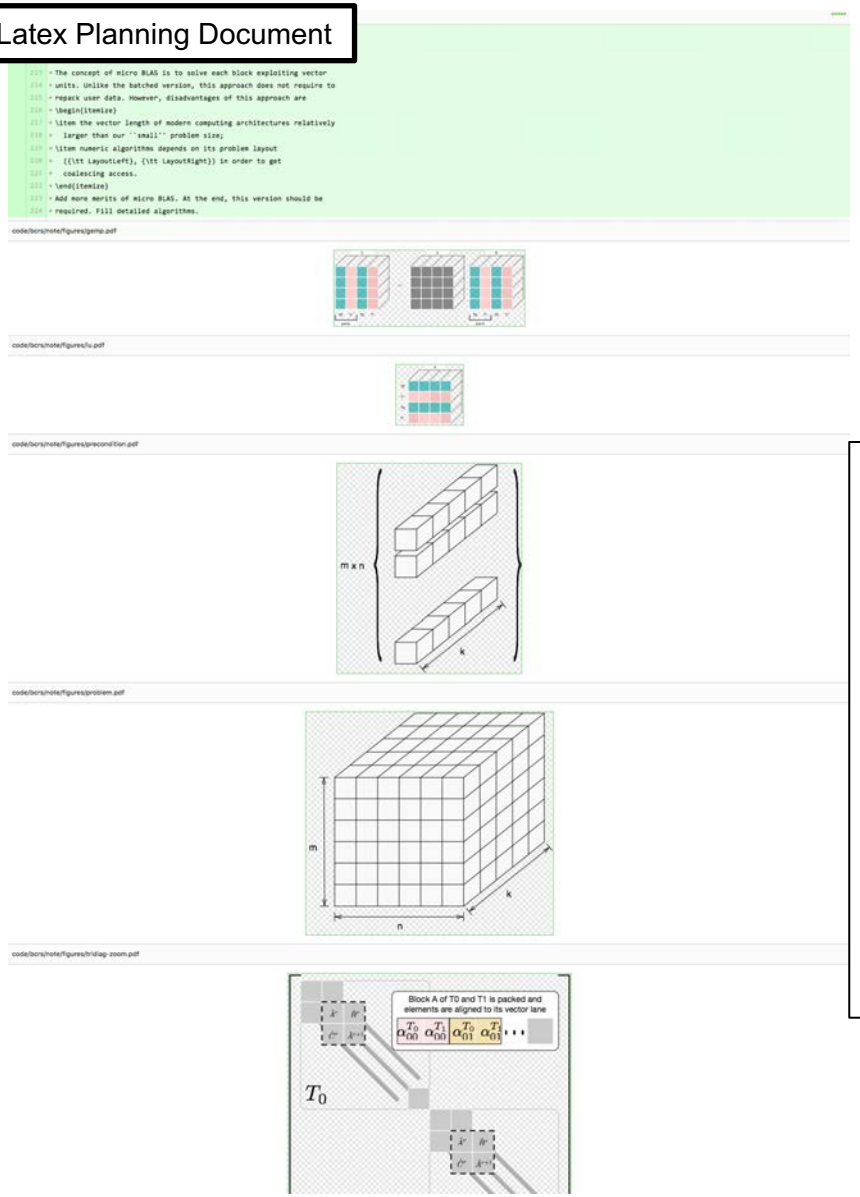
Adapted from Software Project Survival Guide, Steve McConnell

Simple Planned Development Approach



Planning tools: Use what you know

Latex Planning Document



Commit log messages

KokkosKernels - add design note for discussion.
This design note is very informal and working note for discussion.
…

For typeset, "make"

KokkosKernels - add more algorithm variants.
In this algorithm design, I have a few assumptions.
…

KokkosKernels:
Micro & Batched BLAS Design Document

- 6 weeks: Design by LaTeX.
  - Review by diverse experts.
  - Significant design changes: In text only.
- 2 weeks: Write code.

Message: *Use the tools you know.*

Courtesy: KokkosKernels Development Team

"As a <role>, I want <capability> so that <why>" or some variation.

- **IDEAS-ECP Project current activity:** User stories.
- **Brainstorm:** All team members, stakeholders: Create user stories.
  - Easy to generate, starting points for discussion.
- **Discuss:** Discuss each story for scope, understanding and right-sizing.
  - **Out of scope:** Identify stories that are out of scope. Reasons : Not enough expertise, time.
  - **Clarify and right-size:** Clarify stories, split or combine so roughly same "size" and scope.
- **Prioritize, choosing:** Select stories that will be executed.
  - Order the stories based on importance, ability to execute.
  - Only prioritize top set. Keep rest unordered..
- **Create action plan via Epic-Story-Task framework:**
  - Next step for IDEAS-ECP.
- **So far:** Great team building, shared understanding, important topics.

One More Thing

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.

Code Complete is about more than lines of code.

Wrap Up

- SE for CSE is best improved by incremental training of domain scientists.
  - Too hard to be expert in both. Too hard for SE expert to enter CSE domain.
- Lightweight, iterative improvement and processes work.
  - PSIP - Annotating goals with improving how they are achieved.
- Small teams and “team of teams” can work well.
  - Checklists, policies, issues: potent combo for productive, sustainable research.
  - Drive meetings using Kanban board(s) – Can easily manage multiple.
  - Modern platforms (Atlassian, GitHub, BlueJeans, etc.) enable global collaboration.
- Use the tools you know:
  - Key is exploring requirements and multiple design on paper.
  - Getting input from stakeholders, diverse experts *before* writing code.
- When you start to get sloppy, get back on track.

Contribute!

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

Productivity++ Initiative Ask: *Is My Work _____ ?*

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

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.

- Team of Teams: New Rules of Engagement for a Complex World Audiobook – General Stanley McChrystal,Tantum Collins, David Silverman. *https://www.amazon.com/Team-Teams-Rules-Engagement-Complex/dp/B00UVW4RV0*
  Also available on Audible.

- Code Complete, Steve McConnell. Great text on software.
  Construx website has large collection of content.

- https://www.scrumalliance.org - Portal to Scrum material

- Kanban and Scrum -- Making the Most of Both, by Henrik Kniberg and Mattias Skarin – Easy-to-read intro to Kanban and Scrum.

Questions, comments?

Thank You.