Barely Sufficient Project Management
A few techniques for improving your scientific software development efforts

HPC Best Practices Webinar Series

Michael A. Heroux
Senior Scientist, Sandia National Laboratories
Scientist in Residence, St. John’s University, MN
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.
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
- Example
  - Using all the elements together.
  - Step-by-step guide: One approach to doing it yourself.
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.
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
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.
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

Initiation Setup
- Identify project activities
- Create initiation checklist

Ramp Up
- Work initiation checklist
- Initiate project activities

Repeat
- Start process again

Ongoing Planning
- Kanban workflow
- Observe policies

Ongoing Work
- Conduct activities
- Observe policies

Team Member Lifecycle
- Quick ramp up
- Disciplined activities
- Sustained contributions

Depart
- Work complete
- Work transferred
- Contribution sustained

Ramp Down
- Work exit checklist
- Leave project activities

Exit Setup
- Identify final deliverables
- Create exit checklist

Start
# Checklists & Policies

<table>
<thead>
<tr>
<th>Team Member Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Team Member</td>
</tr>
<tr>
<td>Steady Contributor</td>
</tr>
<tr>
<td>Departing Member</td>
</tr>
<tr>
<td>Checklist</td>
</tr>
<tr>
<td>Policies</td>
</tr>
<tr>
<td>Checklist</td>
</tr>
</tbody>
</table>

- **New, departing team member checklists:**
  - Example: Trilinos New Developer Checklist.

- **Steady state: Policy-driven.**
  - Example: xSDK Community policies.
  - [https://xsdk.info/policies/](https://xsdk.info/policies/)
Your checklists & policies?

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

- Discuss in your local group, type in the Google Doc.
Samples from Collegeville Org: Policies, Initiation Checklist

Collegrove 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.
Questions, comments?
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)
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

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Ready</th>
<th>In Progress</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Any task idea</td>
<td>• Task + description of how to do it.</td>
<td>• Task you are working on right now.</td>
<td>• Completed tasks.</td>
</tr>
<tr>
<td>• Trim occasionally</td>
<td>• Could be pulled when slot opens.</td>
<td>• The only kanban rule: Can have only so many “In Progress” tasks.</td>
<td>• Record of your life activities.</td>
</tr>
<tr>
<td>• Source for other columns</td>
<td>• Typically comes from backlog.</td>
<td>• Limit is based on experience, calibration.</td>
<td>• Rate of completion is your “velocity”.</td>
</tr>
</tbody>
</table>

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:
  – 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

- Evaluate Zapier for automated workflows
  - #6 opened by maherou
- Evaluate JuliaSparse
  - #8 opened by maherou
- Create Julia evaluation repo
  - #4 opened by maherou
- Explore the use of composition of containers with Tramonto and Trilinos
- Develop Sagatagan New Team Member Checklist
  - #11 opened by maherou
- Assess the use of TensorFlow for parameter value selection in scientific codes
  - #14 opened by maherou
- Trilinos metadata block
  - #49 opened by duongdo27
- Explore possibility of moving download files for Trilinos and Mantepo to GitHub
  - #47 opened by jwillenbring
- Make expandable map for Better Scientific Software
  - #46 opened by
- Migrate mantepo.org to mantepo.github.io
  - #3 of 3
  - #45 opened by maherou
- Concept map project for better scientific software
  - #35 opened by duongdo27
- Assess requirements for using github.io as host platform for Trilinos.org
  - #41 opened by duongdo27
- Regard the outlook of the concept map
  - #39 opened by duongdo27
- Handle markdown file without links in Better Scientific Software
  - #42 opened by duongdo27
- Finding correspond links for the Github files in the Better Scientific Software
  - #41 opened by duongdo27
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](https://www.scrumalliance.org)
- [Kanban and Scrum -- Making the Most of Both](https://www.scrumalliance.org), by Henrik Kniberg and Mattias Skarin
Questions, comments?
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 team member, e.g., Pat Evans.
• 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
Wrap Up
Next Steps Summary:

- **Create a GitHub (or similar) 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(s)** – Can easily manage multiple.

- **Adapt this approach to meet your needs.**

- **When you start to get sloppy, get back on track.**
Other resources


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

- [https://www.scrumalliance.org](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.