Better (Small) Scientific Software Teams

Presented at
ATPESC 2018

David E. Bernholdt
Distinguished R&D Staff Member and Group Leader
Oak Ridge National Laboratory

See slide 2 for license details
License, citation, and acknowledgments

License and Citation

• This work is licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).


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.

• This work was performed in part at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC for the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.
Outline

• Small Team Models, Challenges.

• Agile workflow management for small teams
  – Intro to terminology and approaches
  – Overview of Kanban
  – Free tools: Trello, GitHub.

• Hands-on example of project management using 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

**Initiation Setup**
- Identify project activities
- Create initiation checklist

**Ramp Up**
- Work initiation checklist
- Initiate project activities

**Ongoing Planning**
- Kanban workflow
- Observe policies

**Ongoing Work**
- Conduct activities
- Observe policies

**Exit Setup**
- Identify final deliverables
- Create exit checklist

**Repeat**
- Start process again

**Depart**
- Work complete
- Work transferred
- Contribution sustained

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

**Ramp Down**
- Work exit checklist
- Leave project activities
Checklists & Policies

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

Managing with Kanban
Managing issues: Fundamental software process

• 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>
<tr>
<td>Notes:</td>
<td></td>
<td>• Key: Work is pulled.</td>
<td></td>
</tr>
<tr>
<td>• Ready column is not strictly required, sometimes called “Selected for development”.</td>
<td></td>
<td>You are in charge!</td>
<td></td>
</tr>
<tr>
<td>• Other common column: In Review</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Can be creative with columns:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>– Waiting on Advisor Confirmation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>– Tasks I won’t do.</td>
<td></td>
<td></td>
<td></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:
  – 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 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.
Samples from Collegeville Org: Kanban Board

- **Backlog**
  - Evaluate Zapier for automated workflows
  - Evaluate JuliaSparse
  - Create Julia evaluation repo
  - Explore the use of composition of containers with Tramonto and Trilinos

- **Ready**
  - Develop Sagatagan New Team Member Checklist
  - Assess the use of TensorFlow for parameter value selection in scientific codes

- **In progress**
  - Trilinos metadata block
  - Explore possibility of moving download files for Trilinos and Mantevo to GitHub

- **In Review**
  - Migrate mantevo.org to mantevo.github.io
  - Concept map project for better scientific software

- **Done**
  - Regard the outlook of the concept map
  - Handle markdown file without links in Better Scientific Software
  - Finding correspond links for the Github files in the Better Scientific Software

---

Better Scientific Software tutorial @ ATPESC 2018-08-08
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: 
    - [] Description

Spaces required
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
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.
  – http://a.co/eUGle95
  – Excellent, readable book on Agile methodologies.
  – Also available on Audible.

• Code Complete: A Practical Handbook of Software Construction, Steve McConnell.
  – http://a.co/eEgWvKj
  – Great text on software.
  – Construx website has large collection of content.

• Getting Things Done: The Art of Stress-Free Productivity, David Allen
  – A classic in the personal productivity literature