# A Light-weight Model for Small Scientific Software Teams

Michael A. Heroux
St. John's University, Collegeville, MN

## Introduction

This paper focuses on a simple, lightweight model with tools and practices to help small scientific software teams[1] introduce structure and process to their management efforts. We emphasize a handful of tools and practices we have found to be broadly useful. This model has been used and evolved as part of the author's student research efforts at St. John's University.

## Team Composition

A common characteristic of scientific software teams is the presence of both junior and senior members. Senior members are typically university faculty or permanent research lab staff.

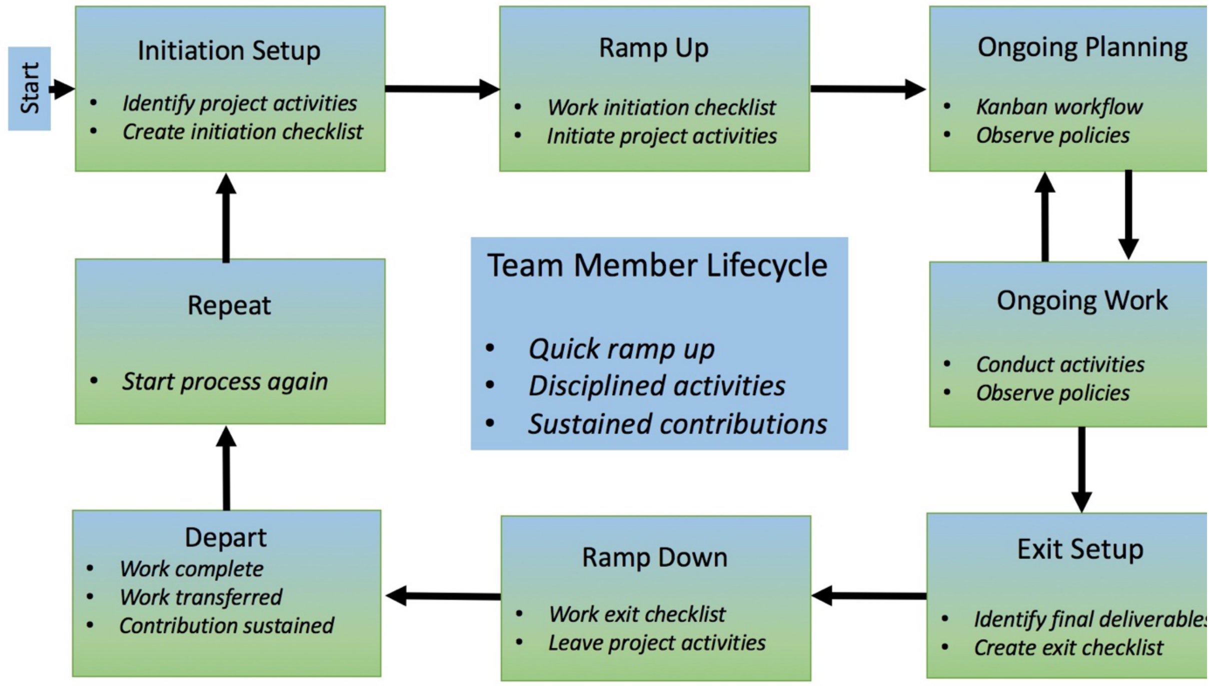

*Figure 1: Membership lifecycle of a typical junior team member. Junior members are typically students or post-docs who are joining, contributing and then departing a team after a few years*

[1] The model, tools and practices described here can be useful for large teams, too. Often large teams are composed of small teams, so these practices can be applied at the small team level and applied similarly to cross-team leadership as a group. Furthermore, team policy becomes even more important on large teams, as do lightweight communication strategies such as project Kanban boards where the status of activities can be referenced asynchronously via online tools.

Junior members can also be faculty or staff on the way to becoming senior members but are typically students or post-doctoral members who are expected to both contribute to the team but also prepare for a future permanent position, often at a different institution. Figure 1 depicts the basic membership lifecycle of a junior team member.

While there are many ways to characterize a research software team, and teams are more diverse than having just junior and senior members as described here, it is useful to characterize members as follows:

**Senior staff, faculty:**

- Stable presence, in charge of research questions, experiments.
- Know the scientific domain conceptual models well.
- May spend less time writing code, may be fuzzy on coding details.

**Junior staff, students:**

- Transient, dual focus (research results, next position).
- Staged experience: New, experienced, departing.
- Learning conceptual models as part of their ramp up.
- Write most code, know code details.

## Team Collaboration Platform

An important component of our small team approach is using a GitHub organization[2]. Our organization is called Collegeville, reflecting the geographical location of St. John's University. Given this organization, which enables coordination of shared repositories, including private repositories that are useful for managing team activities and keeping unpublished content under access control, we have a complete platform for collaboration.

## Team Management Elements

While there are many approaches, tools, methodologies and processes to manage a team, we have found checklists and policies (summarized in Figure 2) to be the most effective and minimal management elements:

**Checklists:** Checklists enable a uniform, repeatable, trackable and ever-improving system for managing key events in a team member's participation on a project. The following lists (with links to examples) are particularly useful for our efforts:

- Initiation: Bringing on a new team member occurs often enough to keep a comprehensive reference checklist of potential activities. From the reference list, a custom list can be created as a GitHub issue and assigned to the new team member.
- Transition: Often during a project a team member (in particular a junior member) will need to learn some new material in order to prepare for their next phase of work. A transition

[2] GitHub is one of several platforms that can be used to support the processes and practices described in this paper. Gitlab, Atlassian and even Google Docs can be used effectively, depending on what tools are readily available and already in use by your organization.

checklist provides them with the needed concrete steps. Progress on the checklist is marked by checking off each item as it is completed, showing progress status.

- General Exit, Research Student Exit: A smooth departure of a team member requires planning and sustainability investment throughout project execution. An exit checklist should be established as soon as possible.

**Policies:** The value of team policies cannot be overstated. The act of creating and periodically reviewing policies is a team-building experience, prompting conversation about what team members value most in each other and their work. The policies establish behavior expectations and better assure that the research and software products being developed will be high quality and sustainable, reducing the cost of work and the cost of losing team members who depart. This sample policy illustrates one concrete example of how a team conducts its work.

| Team Member Phase | | |
|---|---|---|
| **New Team Member** | **Steady Contributor** | **Departing Member** |
| Initiation Checklist | Team Policies | Exit Checklist |
| Built from a comprehensive reference checklist and then customized for the individual. This checklist provides the initial set of concrete activities a person can work on as part of on-ramping. Typical items include setting up accounts, learning about team tools and processes, and learning any basic background content. | Statements of expected behavior and practices. Common items include reference to institutional conduct policies, expectations for how to capture work artifacts, how to resolve conflicts and how to modify team policies in the future. A key purpose of team policies is to assure that team member contributions are sustainable. | Built from a comprehensive reference checklist and customized for the individual. Months prior to a member departure, this checklist assures that all research artifacts are preserved and available to others, that others are briefed on current project status. This list can also contain the steps required to schedule and conduct a thesis defense. |

*Figure 2: Initiation and exit checklists, along with team policies, can accelerate on-ramping and better assure the sustainability of junior team member contributions.*

## Team Activity Management

A simple issue tracking system such as GitHub issues provides teams with a transparent way to manage work. While any repository in your team's organization will typically have its own issue database, we also recommend having an *issues-only* repository. This repository can be used for managing team member tasks that are not specific to a particular project. This repository is also where checklists are managed.

A key tool for lightweight task management is a Kanban board, which can be used as both a dashboard and a tool for coordinating meeting agendas:

- **Kanban Board:** A Kanban board is used to manage tasks by placing them in columns indicating their status. On GitHub, the Project board feature can be used to create a Kanban board. Of particular importance is the 'In Progress' column, which lists the presently active tasks. The discipline of Kanban is that any person or team should have only so much work going on at once, in order to optimize progress and spur innovations that increase the rate of task completion. A sample Kanban board shows the five columns our team typically uses: Backlog, Ready, In Progress, In Review and Done.
- **Regular meetings, updates:** Team meetings and project status can be facilitated by regular use and updating of the team Kanban board. Discussion starts with items in the 'In Progress' column, then the 'Ready' column if a slot opens in 'In Progress'. Other columns should be scanned as well. For projects that rely on external contributions, it can be useful to have a 'Blocked' column. Tasks that were in progress but blocked by an external dependence can be moved to 'Blocked'.

## Summary

For the past five years, the author has conducted small-team scientific software research using a simple junior-senior member model, along with the use of checklists, team policies, and a team Kanban board. This lightweight approach as described in this paper enables coordination and collaboration with modest overhead. We have observed rapid on-ramping, steady progress with ongoing work and the ability to capture the value of junior member work for the future.

The positive impacts of this approach include clear communication of what needs to get done and when, and a sense of clarity and steady progress, especially for new students as they ramp up. Finally, we have seen repeatedly that our approach greatly improves the sustainability of work as one junior member departs and a new member joins to carry on the work of the former. Even with a substantial gap between the departure and arrival, we have experienced tremendous value in improved sustainability from our approach.

We recommend our model, tools and practices for consideration by any team that is just getting started with explicit formal approaches. A companion website to this white paper can be found at https://betterscientificsoftware.github.io/A-Team-Tools/.