# Defining Policies to Turn a Team and Project Around

JASON M. GATES | SANDIA NATIONAL LABORATORIES | ALBUQUERQUE, NM, USA

### THE PROBLEM

# **EMPIRE**







- Code development & process info for next generation electromagnetic/ electrostatic/fluid dynamic codes
- US Department of Energy
  - → Advanced Simulation and Computing
  - → Exascale Computing Project
  - → Advanced Technology Development and Mitigation
- 4 main code repos & 7 auxiliary ones
- Built on Kokkos, TRILINGS

## THE TEAM



- ~20 core developers ≈ 8 full time — —
- ~90 people total interacting on project
- ~6 development teams
- Development team distributed across Albuquerque and elsewhere in the US

# PRIOR TO SUMMER, '17...

- What needs to be done?
- How do I get started?
- Pushes directly to master
- documentation, etc.

## THE SOLUTION

Need to decide how we're going to work as a team and then stick to those decisions. They aren't set in stone forever.

#### GITLAB ISSUES



- All work starts as an issue
- Issue templates

COMMON LOOK AND FEEL

- → More & better information
- Kanban board to organize work-in-progress
- Commits reference issue #'s for traceability

## GITLAB MERGE REQUESTS

- Required to get changes into develop
- Reviewed & approved by > 1 person
- Reviewer tests feature branch
- Code review:
- → Catch problems sooner rather than later
- → Disseminate code knowledge throughout the team



Individuals are more knowledgeable about what the team is doing

#### CODE DOCUMENTATION



- Minimum requirements (\brief, \parameter, \returns)
- Enforced by MR reviewers

Doesn't matter what you decide

Guaranteed to upset someone

→ Maximize agreement

→ Minimize retraining

Just pick something

• Team vote on options:

## GIT WORKFLOW



- master & develop locked down
- Feature branches off develop
- Changes get to develop via MRs
- master updated via nightly testing

#### **AUTOMATED TESTING**



MORE STABLE CODE BASE

- Used to update libraries, develop to master
- things are failing

# MONTHLY RETROSPECTIVE

**IMPROVEMEN** 



- How are we doing as a team?
- Are our policies working well for us?
- Do they need to be amended?

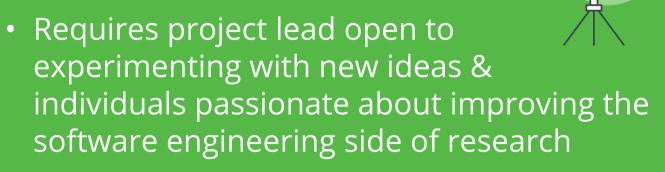
## CURRENT/FUTURE EFFORTS

- Improved automated testing
  - → Better stability
  - → Easier to debug failures



- → Dedicated collaboration time
- → Knowledge transfer
- Onboarding checklist
- More official Scrum adoption
  - → Better defined/enforced rules of engagement
  - → Formalized communication & documentation of work/decisions
  - → Better engagement with component teams

## **OBSERVATIONS**



- High initial commitment to some policies, e.g., GitLab usage, has waned, to an extent, due to real or perceived time pressures
- Requires periodic reevaluation of & recommitment to team policies











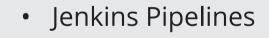


- What are people working on?

Minimal testing, code review,

Largely every man for himself

# CODE STYLE GUIDE





Automated emails

→ Team aware of where/when