Defining Policies to Turn a Team and Project Around

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

THE PROBLEM

- 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
- 4 main code repos & 7 auxiliary ones
- ~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
- Confusion as to:
  - Who’s on the team?
  - What are people working on?
  - What needs to be done?
  - How do I get started?

Pushes directly to master

- Minimal testing, code review, documentation, etc.
- Largely every man for himself

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
- 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

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...

- Confusion as to:
  - Who’s on the team?
  - What are people working on?
  - What needs to be done?
  - How do I get started?
- Pushes directly to master
- Minimal testing, code review, documentation, etc.
- Largely every man for himself

COMMON LOOK AND FEEL

- Design discussions happen that would not have otherwise
- Individuals are more knowledgeable about what the team is doing

CODE DOCUMENTATION

- Minimum requirements
  - \{brief, \parameter, \returns\} Enforced by MR reviewers

CODE STYLE GUIDE

- Doesn’t matter what you decide
- Just pick something
- Guaranteed to upset someone
- Team vote on options:
  - Maximize agreement
  - Minimize retraining

AUTOMATED TESTING

- Jenkins Pipelines
- Jobs build/test multiple machines/configurations
- Used to update libraries, develop to master
- Automated emails
- Team aware of where/when things are failing

MORE STABLE CODE BASE

- More official Scrum adoption
- Better defined/enforced rules of engagement
- Formalized communication & documentation of work/decisions
- Better engagement with component teams

PRIMARY ENFORCEMENT MECHANISM = PEER PRESSURE

IMPROVEMENT

MONTHLY RETROSPECTIVE

- 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
- Team room hackathons
  - 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

- Requires project lead open to experimenting with new ideas & individuals passionate about improving the software engineering side of research
- 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