Collaborative Software Management: The LAMMPS Project

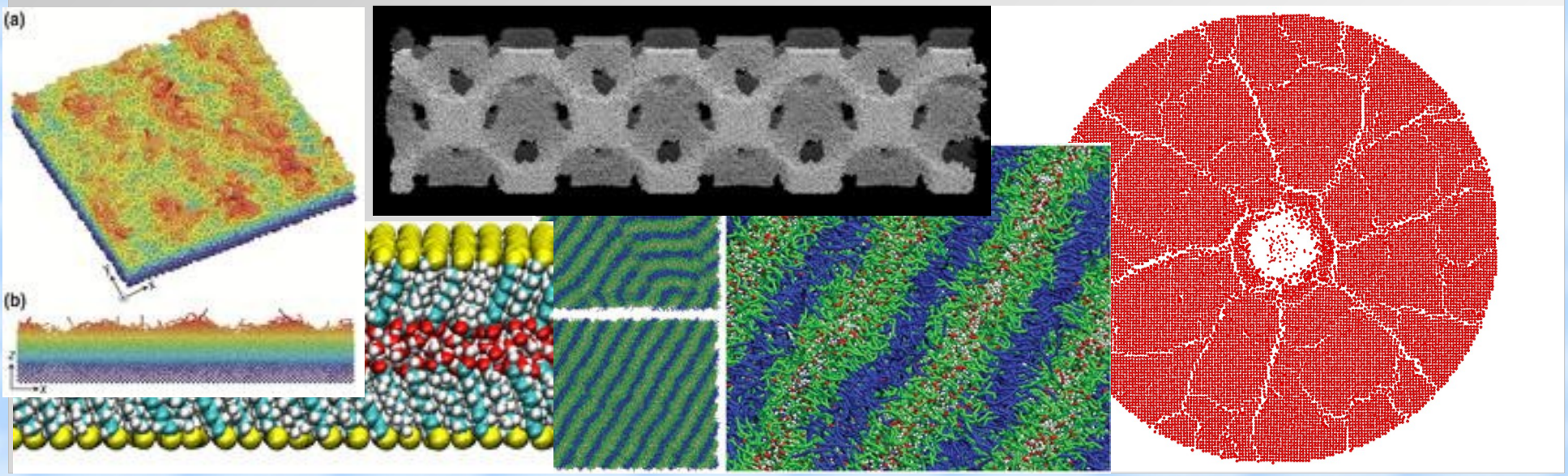
Dr. Axel Kohlmeyer

Assistant Dean for High-Performance Computing
Associate Director, ICMS
Associate Director, TMI
College of Science and Technology
Temple University, Philadelphia
axel.kohlmeyer@temple.edu

LAMMPS Core Developer
developer@lammps.org

What LAMMPS Is

- Large-scale Atomic/Molecular Massively Parallel Simulator
  (each word is an attribute)
- Three-legged stool, supported by force fields and methods:
  one foot in biomolecules and polymers (soft materials)
  one foot in materials science (solids)
  one foot in mesoscale to continuum



LAMMPS is an Extensible Project

- ~3000 C/C++ files with about 1,000,000 lines of code in core executable, plus bundled libs
- Only about 200 files are essential, about 600 files are compiled by default, 2400 are optional
- Optional files are included through derived C++ classes, extra functionality in bundled libraries
- Three levels of “package support”:
  - Core packages (officially supported)
  - USER-<NAME> packages (supported by individuals)
  - USER-MISC package (mixed bag of everything else)

LAMMPS is a Collaborative Project

A few core developers and many contributors:

- Steve Plimpton, Aidan Thompson, Stan Moore at Sandia Lab
- Axel Kohlmeyer, Richard Berger at Temple University

  - Roy Pollock (LLNL), Ewald and PPPM solvers
  - Mike Brown (ORNL/Intel), GPU package, USER-INTEL package
  - Greg Wagner (Sandia), MEAM package for MEAM potential
  - Mike Parks (Sandia), PERI package for Peridynamics
  - Reese Jones (Sandia), USER-ATC package for coupling to continuum
  - Ilya Valuev (JIHT), USER-AWPMD package for wave-packet MD
  - Christian Trott (Sandia), KOKKOS package
  - A. Jaramillo-Botero (Caltech), USER-EFF electron force field package
  - Metin Aktulga (LBL), USER-REAXC package for C version of ReaxFF
  - Georg Gunzenmuller (EMI), USER-SPH, USER-SMD package
  - Ray Shan (Sandia), COMB package, QEQ package
  - Trung Nguyen (ORNL), RIGID package, GPU package
  - Francis Mackay and Coling Denniston (U Western Ontario), USER-LB

- In total over 250 people with significant contributions to LAMMPS

Why Use LAMMPS?

- Flexible choice of per particle attributes
- Large choice of potential functions
- Flexible handling of boundary conditions
- Large choice of ensembles and “manipulators”
- Efficient parallelization (MPI + OpenMP/GPU)
- On-the-fly analysis and powerful scripting
- Easy to add new features or modify code
- Library interface for coupling to other codes

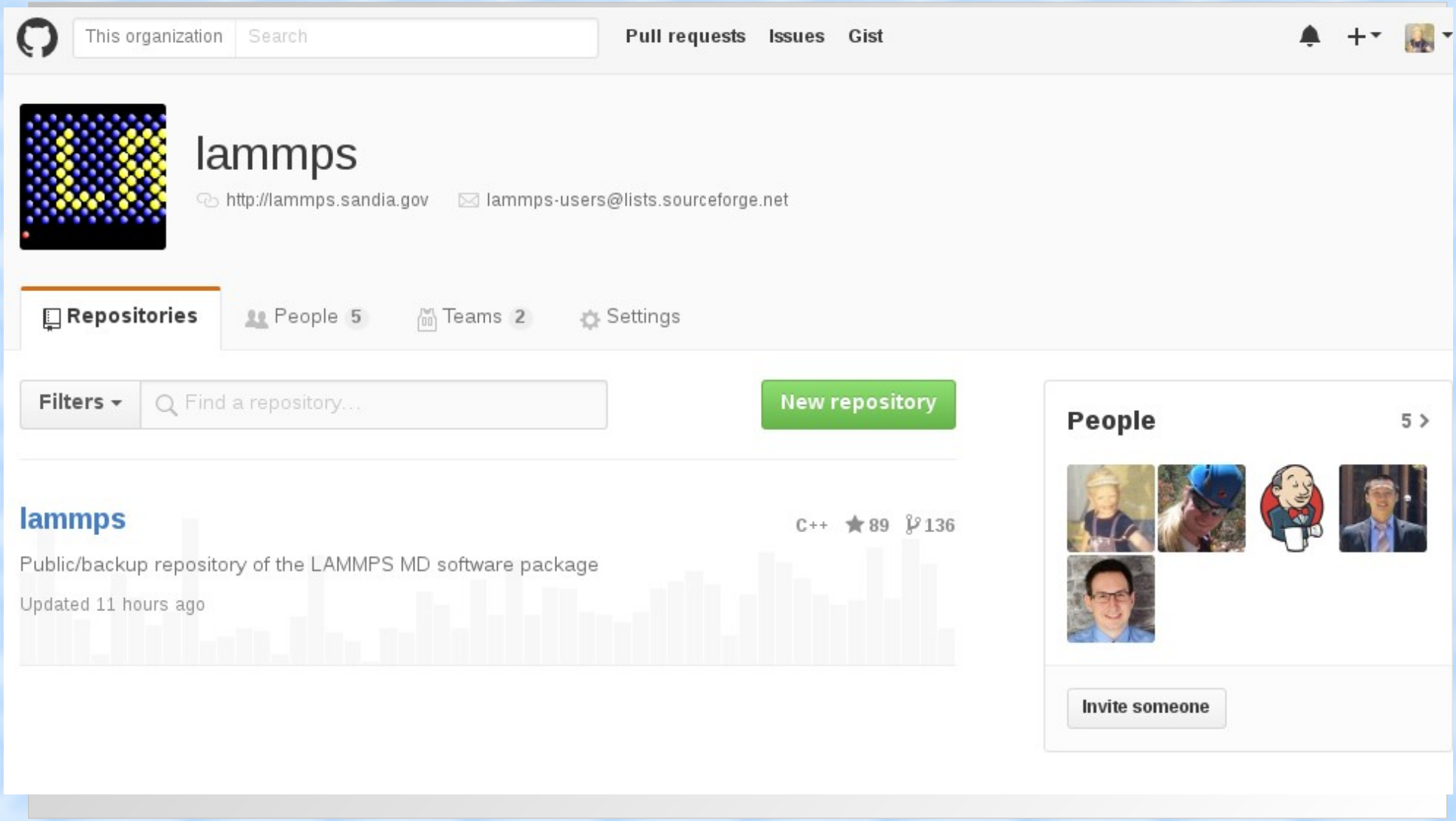
Development Infrastructure

- Public Git repository on GitHub with (manual) backup on Bitbucket
- 3 Branches: master (development), unstable (patch releases), stable (stable versions)
- All changes to LAMMPS *must* be submitted as pull request (even from maintainers)
- Active mailing list for users and developers
- Communication on development also as comments to GitHub issues and pull requests

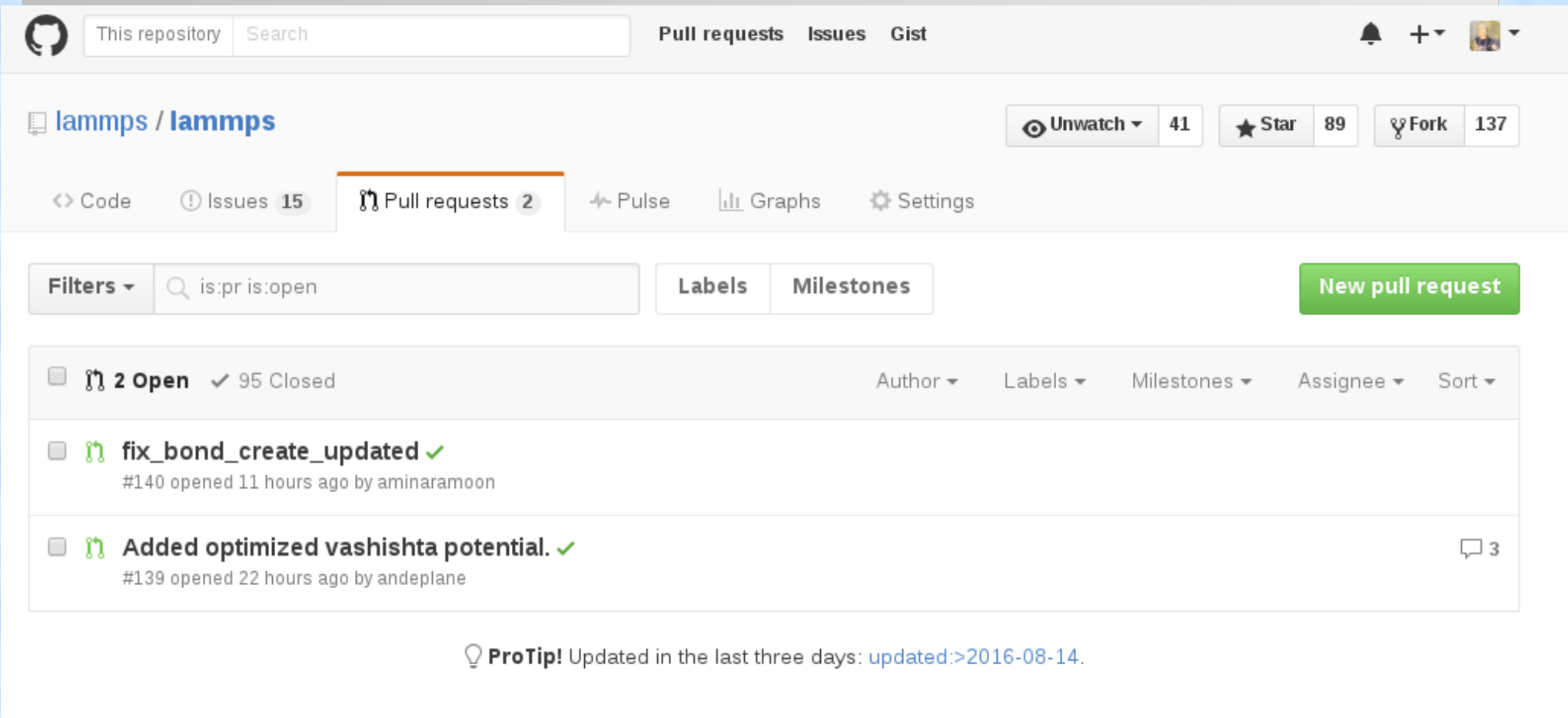
Development Cycle

- Continuous release procedure
- Version indicated by date of release
- Patch releases about ever 4-6 weeks
- 2-3 stable release per year with additional manual testing and compiling
- Continuous integration with Jenkins server tests all pull requests on whether they compile with multiple configurations and settings
- Regression tests after merges to master

https://github.com/lammps



Contributing Code via Pull Requests



Reporting Bugs and Suggesting New Features

This repository | Search | Pull requests | Issues | Gist

lammps / lammps

Unwatch 41 | Star 89 | Fork 137

Code | Issues 15 | Pull requests 2 | Pulse | Graphs | Settings

Filters | is:issue is:open | Labels | Milestones | New issue

15 Open | 28 Closed | Author | Labels | Milestones | Assignee | Sort

- USER-DPD rx should not hardcode indices into atom->dname and atom->dvector arrays [enhancement] [user_dpd] — 2
  #131 opened 11 days ago by akohlmey
- implement generic logger class to replace "logfile" and "screen" [enhancement]
  #120 opened 28 days ago by akohlmey
- Make python wrapper examples python3 compatible [enhancement]
  #118 opened 29 days ago by akohlmey
- Enhancements for USER-HADRESS package [documentation] [enhancement] — 1
  #106 opened on Jun 28 by akohlmey
- Merge Gao-Weber Potential implementation. [enhancement] [manybody_package]
  #102 opened on Jun 21 by akohlmey

Public Continuous Integration and Regression Testing at ci.lammps.org

lammps-icms

All

S	W	Name ↓	Last Success	Last Failure	Last Duration
		openmpi	3 days 15 hr - #62	13 days - #49	17 min
		serial	3 days 15 hr - #63	25 days - #34	26 min
		shlib	3 days 15 hr - #65	8 days 2 hr - #60	5 min 28 sec

- Commits to GitHub repository are automatically checked against many inputs for errors
- Pull request contribution tested for compilability
- Advanced checks and pre-compiled packages

Development Procedure

- Clone the Git repository
- Check out the master branch
- Fork a “feature branch” for development
- Modify sources, test, commit
- Create a separate “feature branch” for each new feature or for bugfixes or modifications
- If development cycle has been long: update master from LAMMPS GitHub repo and merge into feature branches or rebase them

Code Submission Process

- Get GitHub account, create fork of LAMMPS
- Push (local) feature branch into forked repo
- Go to LAMMPS GitHub page and create a pull request after comparing branch to master
- Fill out, modify the pull request template text
- Submit either as draft (=more changes coming) or regular pull request (=ready)
- Wait for automated integration tests to clear
- Fix issues with failed CI tests, if any

Code Review Process

- Only designated developers (3) may merge
- Code must pass all automatic CI tests
- Developers may request additional tests
- Review can be done by multiple developers
- Discretion of person doing then merge if a code is sufficiently well reviewed and approved
- Minimum is one approval from a core developer
- Review requests, manual and automatic

Required Code Properties

- Code should follow documented coding style and conventions (not a strict requirement)
- No tabs, no trailing whitespace, no CR-LF
- All new/changed features must be documented
- Manual must build and pass spell-check tests
- Code has to build with legacy make or Cmake
- Added feature must provide some innovation
- No undesired side effects, no performance hit
- Higher scrutiny if changes to core code

More Required Code Properties

- Contributed code should be “valgrind clean”
- Code must work in parallel and serial
- Header files should not include library headers use forward declaration and PIMPL instead
- Limited use of C++ (STL) headers
- Base code has to be C++98 compatible
- Dependency on libraries only for “packages”
- Use of C++11 (or newer) only in “packages”

Collaborative Software Management: The LAMMPS Project

Dr. Axel Kohlmeyer

Assistant Dean for High-Performance Computing
Associate Director, ICMS
Associate Director, TMI
College of Science and Technology
Temple University, Philadelphia
axel.kohlmeyer@temple.edu

LAMMPS Core Developer
developer@lammps.org