A Renewed Focus on Lustre Software Structural Quality

By Christopher Morrone
OpenSFS CDWG Lead
August 5, 2014

Software Quality

- Two related but distinct aspects
  - Functional quality (feature works or doesn't)
  - Structural quality (robustness, maintainability, etc.)

Lehman's Laws

- Observations describing a set of behaviors
- Law 8 - “...evolution processes constitute multi-level, multi-loop, multi-agent feedback systems and must be treated as such to achieve significant improvement over any reasonable base”
- New features create positive (destabilizing) feedback
- “Lehman demonstrated that systems continue to evolve over time. As they evolve, they grow more complex unless some action such as code refactoring is taken to reduce the complexity” - http://en.wikipedia.org/wiki/Software_maintenance

http://en.wikipedia.org/wiki/Lehman's_laws_of_software_evolution

Need a Lustre Culture Shift

"...we have come to value: Not only working software, but also **well-crafted software**"

Manifesto for Software Craftmanship
http://manifesto.softwarecraftsmanship.org/

"A quality culture is an organizational environment where quality is viewed as everyone's responsibility"

http://en.wikipedia.org/wiki/Software_quality_management

How OpenSFS Can Change the Lustre Software Culture

- Socialize the new software structural quality expectations through presentations at LUG, LAD, SC, ***Lustre developer meetings***, etc.
- Apply a renewed focus on code quality in all aspects of the Lustre software lifecycle in the Lustre Working Group's* activities
- Leverage the existing “Tree” contract to influence positive cultural changes in core development team
- ***Bootstrap quality into current Lustre code base through financial incentives (targeted contract work)***

Software Quality is a Process, Not a Destination

- Easier to keep code high quality if it starts off high quality

  "Oh, sure, this new code is ugly, but so is the surrounding code and we really need to get this bug fixed/meet this deadline."

- Bootstrap quality software process by raising quality standards in existing 420,000 SLOC (lang:C) in Lustre

More Bootstrapping Quality, Fewer Features

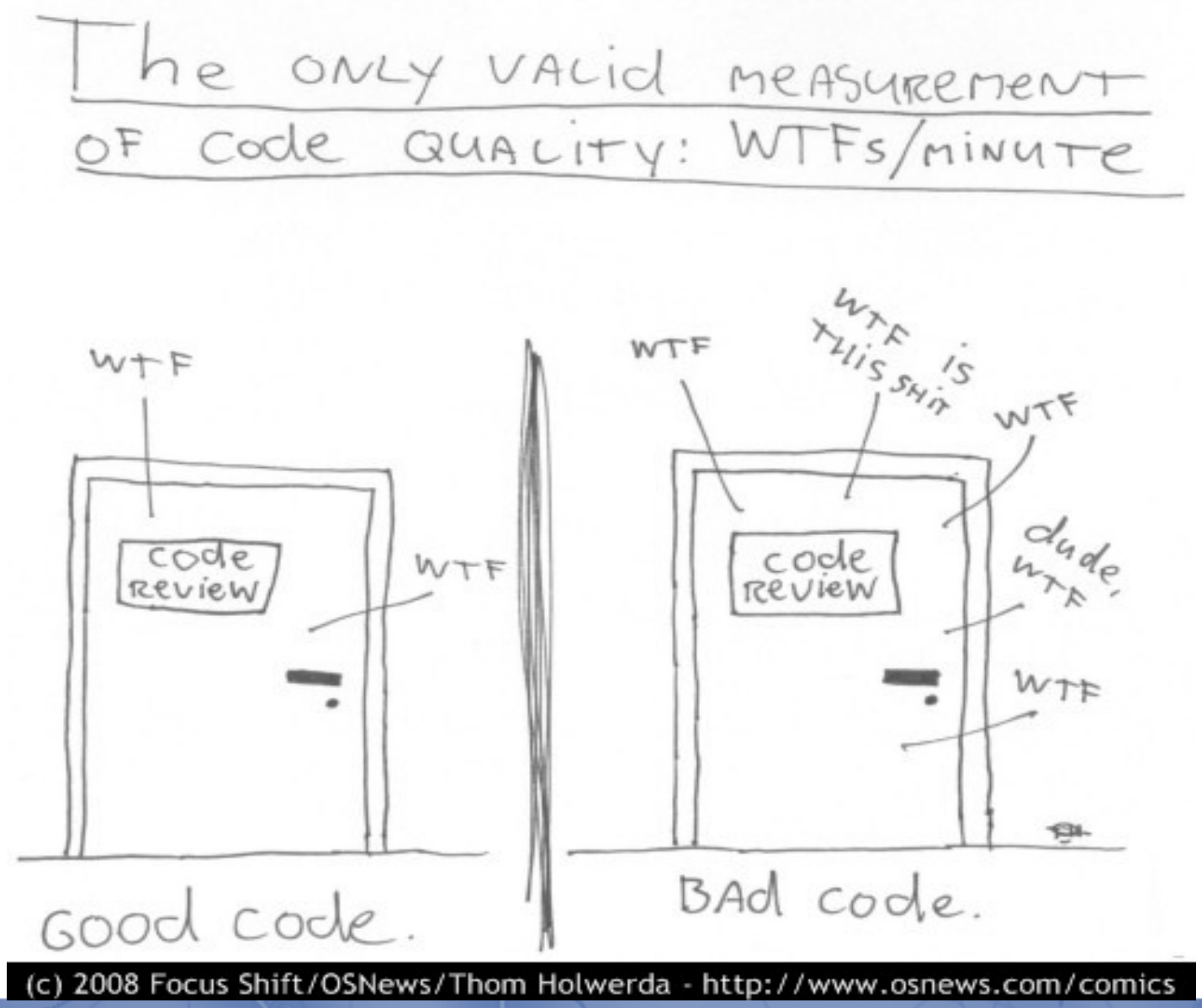
- Contract to document Lustre network protocol (ptlrpc layer and up)
- Series of code refactoring contracts addressing targeted low quality sections of code

How do we know it is working?

Classic Code Quality Metrics

- Cyclomatic complexity/McCabe measure (1976)
- Halstead complexity measures (1977), volume, difficulty, effort, time required to program, number of delivered bugs
- Coupling and cohesion (Larry Constantine, 1960)
- SLOC
- Bugs per SLOC
- Comment Density

But Are They What We Want?



“You can't control what you can't measure.”

“Controlling Software Projects:”, Tom DeMarco, 1987

- “The book for me is a curious combination of generally true things written on every page but combined into an overall message that's wrong. It's as though the book's young author had never met a metric he didn't like. The book's deep message seems to be, metrics are good, more would be better, and most would be best. ***Today we all understand that software metrics cost money and time and must be used with careful moderation.***”
  - Tom DeMarco, 2009

The Dilemma

- Quality is subjective
- Metrics are costly
- Simplistic metrics cause more harm than good

The Solution

- Define targeted metrics as needs are identified
- Improve peer review process to ensure quality

The Plan: Part 1

- Grant contract to document all Lustre protocols from the ptrlrpc layer up.
- OpenSFS creates a repository and web site to how the protocol documents.
- Institute peer review system for protocol changes.
- ***No protocol changes permitted to land in tree until protocol documentation change submitted, peer reviewed, and ratified.***

Lustre Protocol Documentation Contract Deliverables

- Complete set of documentation in form appropriate for storage in git repository, and presentation in other forms (web, pdf, etc.)
- A list of files and individual functions needing refactoring, as encountered while reading the code in order to write accurate protocol documents.

The Plan: Part 2

- Grant multiple contracts to refactor specifically identified areas of the code.
- The Lustre Protocol Documentation Contract will be one source of potential areas for refactoring.
- Further refactoring targets identified by Lustre Working Group

The Plan: Part 3

- Continue to evaluate and improve all areas of the Lustre software development lifecycle (requirements, design, implementation, testing, etc.)
- Remember: We are bootstrapping the quality of the existing code base, but software quality is a continuing endeavor.

What We Need From the Board

- Approval to begin development and negotiations of the Lustre Protocol Documentation contract.
- Optional (could start later): Approval to begin development and negotiations of the first Code Refactoring contract.

Final Quote

“...control is ultimately illusory on software development projects. If you want to move your project forward, the only reliable way to do that is to cultivate a deep sense of software craftsmanship and professionalism aroudn it”

- Jeff Atwood