CMDV Software:

SW Engineering ~~Education~~ Learning

Mike Heroux
Senior Scientist
Center for Computing Research
Sandia National Laboratories
Collaborators at this meeting:
Andy Salinger
Anshu Dubey

http://tinyurl.com/HerouxAcme2016

Sandia National Laboratories

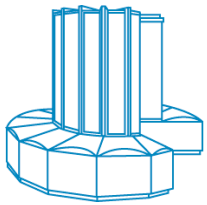
Exceptional service in the national interest

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Some relevant background



CRAY RESEARCH

Libraries & Applications

Miniapps

Scientific Libraries

Benchmark

Saint John's UNIVERSITY

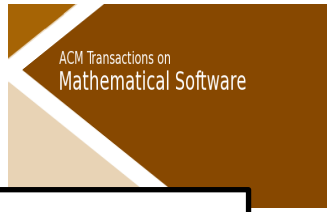
Research Methods

IDEAS productivity

Anshu Dubey

Productivity & Sustainability

Association for Computing Machinery



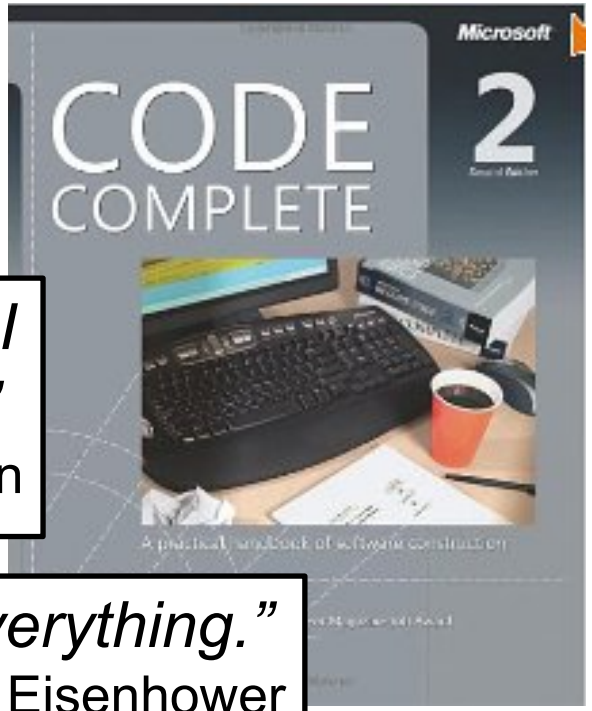
Reproducibility

Libraries & Frameworks

IDEAS2

Code Complete: Useful "Overhead"

- Code Complete: Ultimate value is code.
  - Should we only write code?
  - Some non-coding activities improve code.

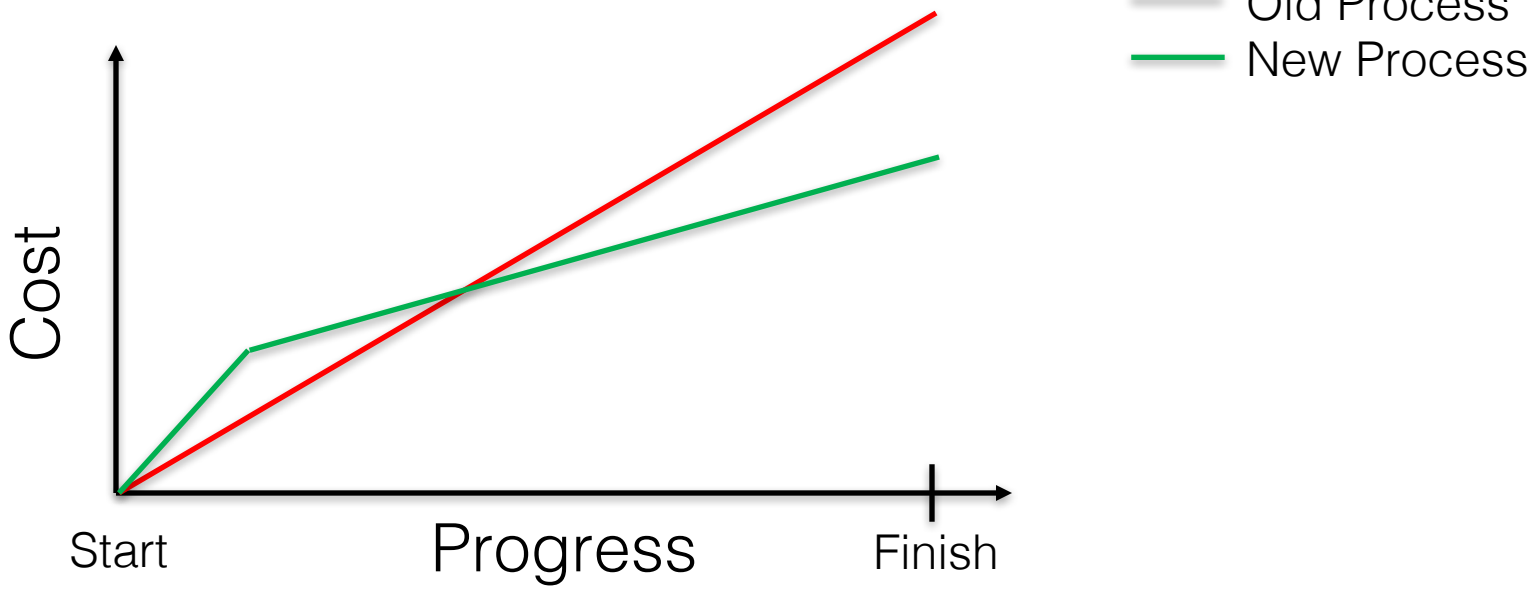


"Give me six hours to chop down a tree and I will spend the first four sharpening the axe."
Abraham Lincoln

"Plans are worthless, but planning is everything."
Dwight D. Eisenhower

General Strategy

- Interview, analyze, prototype, test, revise, deploy. Repeat.
- Realistic: There is a cost.
  - Startup: Overhead.
  - Payoff: Best if soon, clear.



First CMDV SW Activity: Interviews

Sandia National Laboratories

- Interviews + Andy Salinger:
  - Rob Jacob, ANL, ACME SE/CPL. (10/10/16)
  - Balwinder Singh, PNNL, ATM Integrator. (10/13/16)
  - Todd Ringler, LANL, Ocean/Ice lead. (10/13/16)
  - Gautam Bisht, LBNL, Land Integrator. (10/14/16)
  - Steve Ghan, PNNL, ATM, leads other CMDV. (10/19/16)
  - Xiaojuan Yang, ORNL, land, junior member. (10/20/16)
  - Philip Cameron-Smith, LLNL, ATM, senior. (11/2/16)
- Diverse cross-section: Lab, Component, Proximity to SE Group, Experience.
- Notes and summary on Confluence.

Some common interview themes

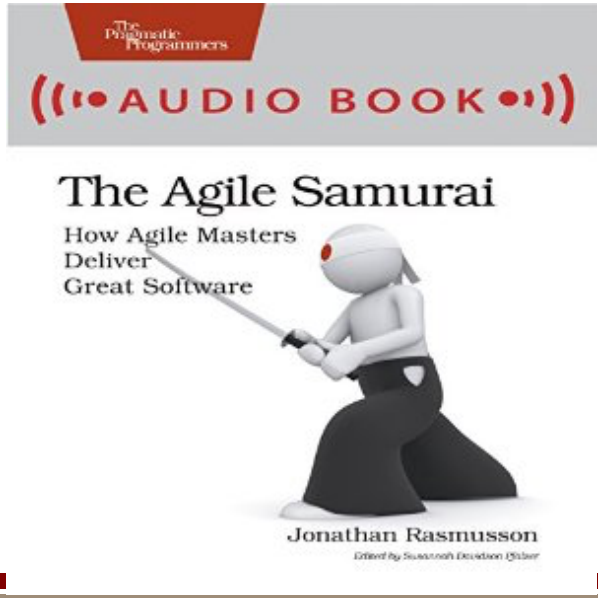
- Software challenges compete with other high priority demands.
  - Urgency of science challenges is paramount.
  - Software improvement must be introduced carefully, with timely and highly probable payoff.
- Challenge working with Git, especially:
  - Efficient management of simultaneous development of shared code.
  - Uncertainty with uncommon but essential processes.
  - Uncertain understanding of how Git really works.
- Testing concerns mentioned often:
  - Testing process not uniform.
  - No standard test harness.
  - Groups evolve own testing approaches.
- Unit testing often mentioned:
  - Desire for quicker, more localized testing, i.e., unit testing.
  - Concern about feasibility of unit testing.

Breakout #2 at 9:50am (Dubey, Wilke)

- Shorten the development cycle:
  - More features with less work.
  - Fewer merge conflicts.
  - Lower barriers for scientist-developers.
- Desire for better, more uniform developer training and minimal skill levels.
  - Basic developer workflows.
  - Coding standards; readable, sustainable source code.
  - Effective commit log messages.
  - Tempered by concerns of too much emphasis.
- Tools and processes should be kept simple, easy-to-use:
  - ACME team is diverse, simplicity is important.
  - External collaborators can more easily contribute to ACME and use product.
- Learning opportunities should be varied:
  - New team member orientation.
  - Face-to-face, webinars, individual learning plans.
  - On-demand access to software experts.
- Programming for performance:
  - Basic performance concepts.
  - Performance portability.
- Challenges using JIRA effectively, especially in the presence of GitHub issues.
  - GitHub issues used daily, considered essential.
  - JIRA used less frequently, often an afterthought.

Learning Strategies

- Goal for this meeting:
  - Make contacts:
    - Goto http://tinyurl.com/acme2016learn
    - Pick a time to meet with me.
  - Find out how you learn best.
- Possibilities ways to learn:
  - Real-time, face-to-face? Software Carpentry.
  - Real-time, webinar? Coordinate with LCFs.
  - Recorded, webinar? By-product of real time.
  - MOOC, SPOC? Udacity, etc. Plus local expert.
  - Individualized?
    - Slack, On-demand?
    - Github-based?
    - Audible (my favorite way to learn).
  - What can work for you? Let me know.



Clear learning subject: Git

Sandia National Laboratories

- Powerful, challenging.
- “Defensive” Git Training
- Teach basic worflows: yes.
- Teach also:
  - Prepare to avoid disaster.
  - Prepare for disaster.
- Practice disaster recovery:
  - Create disaster.
  - Recover.
  - In safe setting.
- How to deliver? To whom?

> *With Git as your source management tool, everyone feels stupid.*
>
> *John Cary*

Commitment to Quality

Canadian engineers' oath (taken from Rudyard Kipling):

My Time I will not refuse;
my Thought I will not grudge;
my Care I will not deny
toward the honour, use,
stability and perfection of
any works to which I may be
called to set my hand.

http://commons.bcit.ca/update/2010/11/bcit-engineering-graduates-earn-their-iron-rings

Productivity++ Initiative

Ask: *Is My Work _____ ?*

Sandia National Laboratories

Productivity++

- ✓ Traceable
- ✓ In Progress
- ✓ Sustainable
- ✓ Improved

Version 1.2

https://github.com/trilinos/Trilinos/wiki/Productivity---Initiative

Recent email exchange

XXX,

Please note: Below is a request for information, not scrutiny of your process. Please read from this perspective.

I am wondering if the change you made below is covered by tests. Specifically, is there anything in the test suite for package X to confirm that the changes you made are the changes you intended to make? If not, is there another way you are assuring that the change is covered by tests?

I am not asking for the purposes of questioning your process, but I am probing the accuracy of a simple metric and tool I want to use for monitoring code quality. Specifically, I would like to scan Git commits to see of a source tree change has a corresponding test tree change. But I want to understand the weaknesses of this simple metric if we were to use it. My intention would be to inform developers about their commits to source without commits to test.

I would appreciate your thoughts on this.

Thanks.

Mike

On 8/13/16, 1:07 AM, "Trilinos-checkins on behalf of XXX XXX" <trilinos-checkins-bounces@trilinos.org on behalf of XXX@XXX> wrote:

> Branch: refs/heads/develop
> Home: https://github.com/trilinos/Trilinos
> Commit: xxxxxxxxx
> https://github.com/trilinos/Trilinos/commit/xxxxx
> Author: XXX XXX <XXX@XXX
> Date: 2016-08-12 (Fri, 12 Aug 2016)
>
> Changed paths:
> **M packages/xxx/src/file1.hpp**
> **M packages/xxx/src/file2.hpp**

Sorry Mike. I'll add a test right now.

XXX,

Please don't take my request as a spur to write the test sooner than you would otherwise. I am really just probing to see if the simple metric I have is a good indicator.

So my question is: My metric (committing to source without committing to test) would suggest that the activity decreased software quality in this situation. From your perspective, is this true?

Thanks.

Mike

P.S. I guess the Heisenberg Uncertainty Principle applies to software systems as well: Can't probe for quality without changing behavior ☺

I disagree with your assessment. My fix was hasty so I didn't add a test. I am now remedying that.

Summary

Sandia National Laboratories

- I have experienced the “help” of SW Engineering Experts.
  - Cray circa 1990, ASCI circa 2000
  - Ignored their own process:
    - Failed to elicit, analyze requirements.
    - Slapped on pre-defined solutions.
    - Failed.
- Hopefully realistic: This will not be easy.
- Goals:
  - ID biggest opportunities.
  - Create content and delivery strategies.
  - Work with you.
  - Goto http://tinyurl.com/acme2016learn to sign up to talk.
- Final message:
  - Aspire to improve on your own.
  - I have shelf magnets ☺