Removal of Dead Code Scope Statement

Added by Richard Henwood, last edited by Andreas Dilger on Feb 21, 2014

Introduction

The following scope statement applies to the Code Clean-up project within the Technical Proposal by High Performance Data Division of Intel for OpenSFS Solicitation Number W4570 Delivered April 5th 2013.

Problem Statement

Ongoing restructuring of the Lustre* software source code has created large swaths of unreachable code and unused data. At the same time, the layering and complexity of Lustre makes this dead code and data difficult to identify during restructuring, adding unnecessary complexity to ongoing development and maintenance. We propose a concerted effort to address this issue.
This project will identify and remove code from the Lustre codebase which is no longer being used. This will be achieved by using static code analysis tools to isolate areas of dead code. Engineers will then hold inspections to understand why the code is dead and on how to best remove the code with minimal impact to other on-going projects.
Some areas that will be targeted include:

- OBD methods and handlers
- /proc files and structures
- libcfs module APIs
- Utilities
- liblustre

Project Goals

- Clean-up in the order of 10,000 lines of the Lustre software codebase.
- Demonstrate automated checking of code correctness.

In Scope

- Delete unused functions, unused structures, unused structure members.
- Implement a tool to identify 'const' correctness.
- Implement a tool to identify incorrect header usage.
- Perform analysis on the Lustre software code base using the Sparse Static Analyzer.
- Demonstrate a robot that provides 'clean code' metrics into the patch review cycle.
- Removal of code that is judged to be incorrect.
- Open source the CLANG plugins develop for this project.

If time allows:

- Tools development: implement a 'getter' and 'setter' plugin checker.

Out of Scope

- No changes will be made to the existing test infrastructure.
- Running the Sparse Static Analyzer on every build and pushing newly introduced errors beck to Gerrit.

Project Constraints

John Hammond is the strongly preferred engineer for this project and John Hammond is also the strongly preferred engineer for the Layout Enhancement project.

Key Deliverables

- Solution Architecture.
- High-level design document.
- Code patches.
- Demonstration of automated code correctness check.

*Other names and brands may be the property of others.

Like Be the first to like this

5 Comments

Nathan Rutman

What about redundant code (i.e. two functions that perform similar tasks)? Any streamlining involved?

Nathan Rutman

Does this explicitly mean liblustre will be removed? (and other #defined off code)

John Hammond

Unused redundant code will be removed. We have not attempted to identify globally redundant code.

By default, building lustre from source (sh autogen.sh && ./configure && make) builds liblustre as well. It's not #defined off and static analysis identifies most of it as being used. For example, in lustre/liblustre/tests/sanity.c, main() calls __liblustre_setup_(), which calls _sysio_lustre_init(), which calls lllib_init(), which passes llu_fssw_ops to _sysio_fssw_register(). llu_fssw_ops captures references (uses) several functions from liblustre, which in turn call other functions and so on. Statically unused code from liblustre will be deleted.

Cory Spitz

What does "if time allows" mean?

Richard Henwood

If we complete our goals ahead of schedule, we will add this item into scope.