Software Sustainability and the Challenge of Containerization

Henry Neeman, University of Oklahoma

Director, OU Supercomputing Center for Education & Research (OSCER)

Assistant Vice President, Information Technology – Research Strategy Advisor

Associate Professor, Gallogly College of Engineering

Adjunct Associate Professor, School of Computer Science

SC18, Wednesday November 14 2018

Software Sustainability Is Hard!

- Thousands of community codes
  - Dozens of very popular community codes
- Software development cost is typically in the millions per community code.
- Research funding agencies rarely fund software sustainability.
  - So, to fund your software, you have to constantly add groovy new science features -- they'll pay for that.
- Software maintenance is labor-intensive but emotionally underrewarding.

Containers Are Great! (Near Term)

- Containers make it possible to port software to a broad variety of platforms without having to do the tedious refactoring associated with using new compilers, operating system versions etc.
- So, huge savings in labor cost, and makes creating useful/usable STEM research software more attractive for STEM researchers!

Containers Are Awful! (Long Term)

- What happens when a new CPU flavor comes out?
  - New instructions
    - MMX 1996

      → SSE 1999 → SSE2 2001 → SSE3 2004 → SSE4 2006

      → AVX 2008 → FC16 2009 → FMA 2012

      → AVX2 2013 → AVX512 2015

    https://en.wikipedia.org/wiki/SSE4
- Does that new stuff get put into bug fixes for older compiler versions? No!
- If you’re containerized, do you want to port to the next version of the compiler? Won’t that push you to a new OS version too?

Containers Get You Stuck

- Okay, so let's not bother to port this year --
that's a lot of work, and the code is running perfectly well, so let's not mess with it this year.
- How about next year -- will it be less work, or more work? So will you be more likely, or less likely, to go to all that trouble?
- How about five years from now? Ten?
    - The longer you wait, the harder it is.
- But if you don't port the code, then you can't take advantage of those new instructions, so your (CPU-bound) code doesn't speed up!