Why are the biggest supercomputers so challenging?
Balancing Software Engineering

Katherine Riley
August 14, 2013
Define a supercomputer

“A supercomputer is a computer at the frontline of contemporary processing capacity - particularly speed of calculation which can happen at speeds of nanoseconds” - Wikipedia

- But not all supercomputers are as ‘frontline’ as others
- This makes the ecosystem ... challenging.
Top 500

Comment

Legend:
4, 6, 8, 10, 12, 16,
Top few vs Top 500

There are not many huge supercomputers

The biggest supercomputers are bleeding edge
  - Hardware is sort of hardened in install
  - Software is flushed out over the first year or two
  - What I consider normal, many might consider HOSTILE

Why is That?
  - Reputation
  - Move science forward
    - Machines are targeted to a small number of users
  - Help steer technology
How is a big DOE supercomputer purchased?

Science Operations  
Politics

$  

Call for Designs  

Science Operations

Choice (Science, Facility, Budget)

$  

Science

Vendor  

Ops

Earliest Hardware

Vendor  

Ops

Science

Usability  
Balance  
Capability  
Future  
Collaboration

Here be Early science

Vendor  

Ops

Shake it out at scale

Vendor  

Ops

Delivery
How is a big DOE supercomputer purchased?

- All over 5 years
- We start the next one before the current one is fully “on the floor”
Collaboration does not yield perfection

- Even with all these iterations with the vendor, the machine will not be perfect
Collaboration does not yield perfection

- Even with all these iterations with the vendor, the machine will not be perfect

(Not perfectly weighted)
What this means for your SW ecosystem?

- Bleeding edge systems are their own fun
- The system is available for production REALLY near to it’s first construction
  - Early science buffers this, but not completely
  - It takes time to move software to a new architecture at a new scale
  - Even earliest access won’t really resolve this
  - Vendor does not have a significant leg up
- As SW on a machine stabilizes, the machine reaches end of life

- The available software will be bare bones
- Complicated requirements can slow you down
- Facilities tries very very hard to expand that as fast as possible
Why Engineering Principles are Crucial

- Testing of your software is crucial
  - You might have a better process than library developers & vendors
  - Defense against other people’s bugs

- Versioning

- Abstraction of functionality
  - To address the speed of change
  - Help debug problems
  - Help with performance

- Document to help us help you

- Running on the biggest systems is a competitive advantage
  - Own all of your code. Even libraries.

- Let the compiler do the work with limits
- Use high level languages with care
Questions