# boost && CMake

boostcon, May 2010

David Cole
david.cole@kitware.com

Bill Hoffman
bill.hoffman@kitware.com

How CMake Changes The Way We Build C++

- Boost aims to give C++ a set of useful libraries like Java, Python, and C#

- CMake aims to give C++ compile portability like the compile once and run everywhere of Java, Python, and C#
  - Same build tool and files for all platforms
  - Easy to mix both large and small libraries
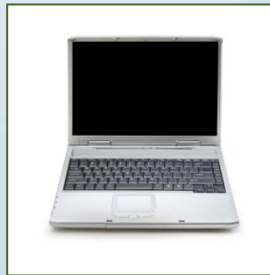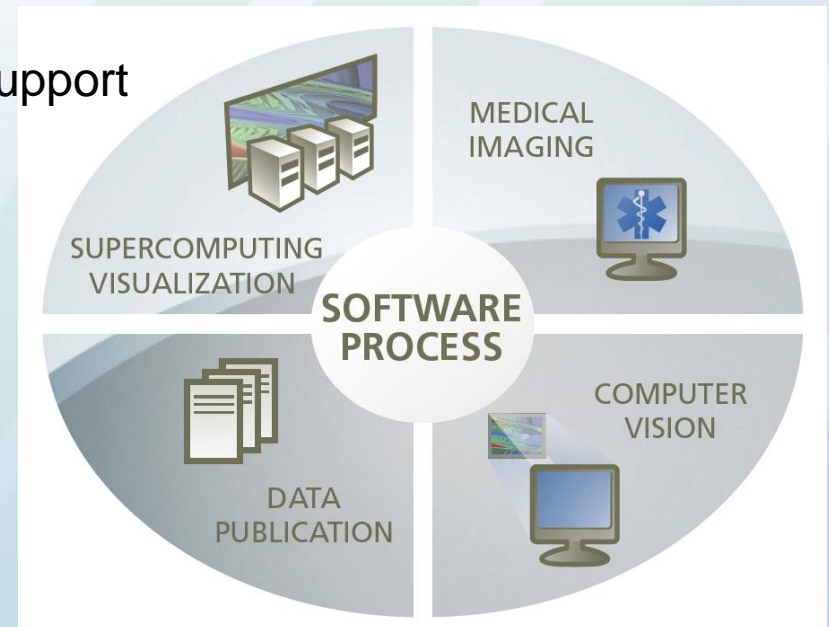
Kitware

Kitware Pledge to Support boost

- one man-month over the next year
  - (this is a guaranteed minimum level of effort)
  - additional effort is very likely, if needed

- sponsored by Kitware clients with
  - direct interest in building boost with CMake
  - significant overlap with proposed new functionality

Kitware

Kitware: the Company

- Founded in 1998

- Founders: 5 previous employees of GE Corporate Research

- Privately held, profitable from creation, no debt
  - Revenues ~$9 million in 2009
  - Principally consulting/grants, with support product revenue

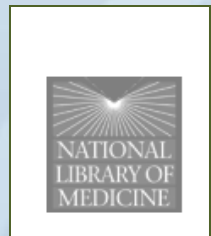- Approximately 65 employees growing rapidly (30% in 2009)
  - > 25 PhD

Kitware

Who Is Involved?

Users

- KDE
- Second Life
- ITK
- VTK
- ParaView
- Trilinos
- Scribus
- Boost (Experimentally)
- MySQL
- LLVM
- many more

Supporters

- Kitware
- ARL
- National Library of Medicine
- Sandia National Labs
- Los Alamos National Labs
- NAMIC

Kitware

Using CMake: Benefits

- CDash, CMake and CTest have lots of direct funding coming from multiple Kitware clients

- Indirect funding is also available when boost community needs overlap with Kitware client needs

- Using CMake means
  - joining a community that is well funded and very interested in C++ software development

Kitware

Specific Objectives

- Build boost with CMake (officially)
    - get rid of that "(Experimentally)" on prior slide…

- Enable incremental testing based on dependency analysis: at build time and at test time

- Cloud based testing with CDash & git

Kitware

Architecture / Refactoring

- Presently build time incremental testing is possible with add_custom_command and IMPLICIT_DEPENDS using makefile generators

- Extend that capability to work
  - with non-makefile generators
  - at test time instead of at build time

Other Relevant Recent Work

- Subprojects in CDash

- ExternalProject module added to CMake 2.8
  - Several projects have adopted using this to build their pre-requisites using a "super build" approach

- Parallel testing (ctest –j N) added in version 2.8

Kitware

Why CMake? It's fast

http://blog.qgis.org/?q=node/16

"I was quite surprised with the speed of building Quantum GIS codebase in comparison to Autotools."

Task	CMake	Autotools
Configure	0:08	Automake: 0:41 Configure: 0:20
Make	12:15	21:16
Install	0:20	0:36
Total	12:43	22:43

Kitware

Why CMake? Everyone is using it

KDE 2006 – Tipping Point!
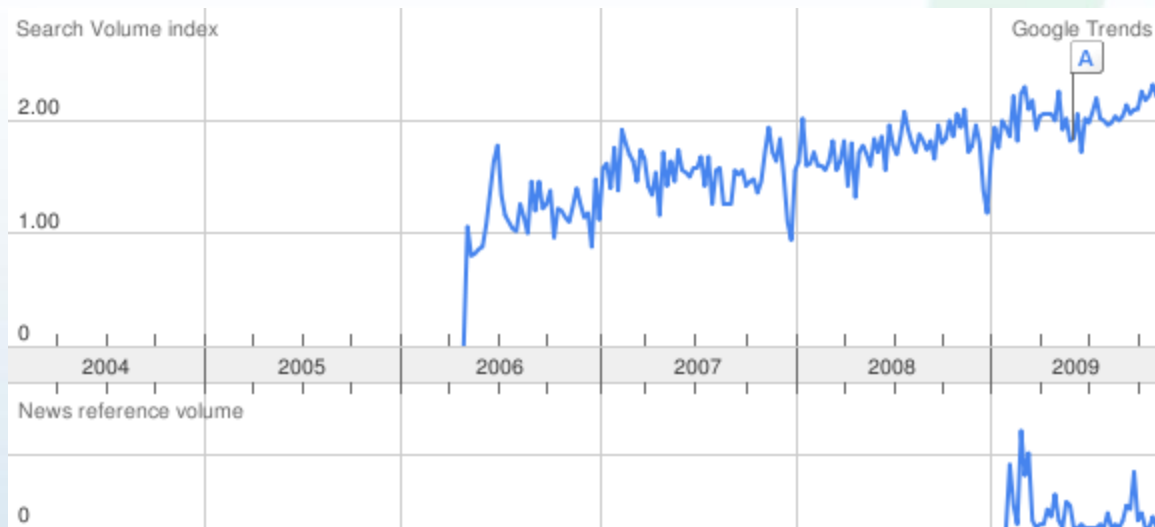
- 1200+ downloads per day from www.cmake.org
- Major Linux distributions and Cygwin provide CMake packages
- KDE, Second Life, Boost (Experimentally), many others

Kitware

Trilinos (Multi-Package Dashboard)
http://trilinos.sandia.gov/cdash/index.php

Query Filters : customize views

Kitware Hosted CDash
http://my.cdash.org

Summary

- Kitware Pledges Support

- Many recently added features are relevant

- More work still to do!

- Links
  - http://www.kitware.com
  - http://www.cmake.org
  - http://www.cdash.org

email
david.cole@kitware.com
bill.hoffman@kitware.com

Kitware

Thank You

Kitware Quality Software Process

CTest/CMake

Software Repository

CDash Dashboards

Developers check in code

Developers review results

Kitware