Boost: A Personal Perspective

Reflections and Discussion on the Past, Present and Future of Boost

Kevlin Henney
kevlin@curbralan.com

Boost Origins and Motives

- The historical motivation for Boost was to avoid the problems of standardisation without prior art
  - This was highlighted by the strain that the STL's use of templates put on compilers of the time
  - Also highlighted by speculative library design flaws, e.g., the allocator model, *valarray*, *char_traits*, locales
- Boost became a proving ground for library ideas and distribution of libraries to the C++ users
  - Conceived of at a standards meeting following C++98 standardisation (named by Herb Sutter, IIRC), based on peer-reviewed acceptance

Boost Presence and Influence

- Boost has grown far beyond the C++ standards committee, both in terms of influence and people
  - Boost is one of the leading open source C++ libraries
  - Boost is preinstalled with a number of Linux distros
  - It is used as a test library by MS Visual Studio team to check their compiler
  - Mentioned as a resource by many sites and books
  - Boost has spawned a number of new techniques
  - It has led to a number of books and has spun off a conference

My Involvement

- Early involvement in Boost and mid-term involvement with the ISO committee
  - Contributed *numeric_cast*, *lexical_cast* and *any* to Boost, which are now maintained by others (including, previously, Terje Slettebø — thanks!)
  - Involved in early threading discussions for Boost and also C++0X
  - Speaker and committee member for BoostCon
- Less involved now
  - But still ongoing casual interest

Influencing the Standard

- In line with the original vision, Boost has influenced the forthcoming C++ standard
  - The obvious influence is in the form of contributed libraries, specifically those included in TR1, which have enjoyed implementation experience and real-world usage and feedback before standardisation
  - The other area of contribution has been in helping to mature the understanding of language support for generic programming and the kind of type model provided by templates, e.g., concept checking, <*static_assert.hpp*> and *enable_if*

Libraries for TR1 and C++0X

- Leaving aside non-Boost libraries, a number of contributions are notable to TR1 and C++0X
  - Memory managing smart pointers: *shared_ptr* and *weak_ptr*
  - Fixed-size containers: *array* and *tuple*
  - Function objects: *function*, *mem_fn* and *bind*
  - Regular expressions: <*regex*>
  - Random number generation: <*random*>
  - Type traits and reference wrappers
- A Boost TR1 implementation is being worked on

Libraries for TR2

- This is still work in progress
  - It is scheduled for completion following C++0X
- However, we can still see what has been submitted for inclusion that is from Boost
  - Filesystem
  - Parts of *asio* in a networking library
  - Memory-mapped and shared files
  - Iterator ranges
  - Date and time facilities
  - *any* and *lexical_cast*

Threading

- In spite of representing prior art, the Boost threading library is not in C++0X
  - Instead, an evolution of the library that has emerged from diverse and extensive discussion
  - Reflects some experience with drawbacks of Boost model, but still reflects constrained ambitions and some issues (N1883 offers a different view)
- Inclusion of threading is more than just a library issue, however, and affects memory model
  - But the effect on the library has not been sufficiently examined, e.g., *shared_ptr*

Some Other Libraries of Interest

- Many libraries have a practical edge to them, but are not necessarily candidates for the standard
  - But they are certainly of day to day use and offer value in the classic way of many open source projects
- For example...
  - The testing framework
  - Preprocessor metaprogramming
  - Formatting
  - BGL (Boost Graph Library)
  - Program options

Some Mind-Melding Libraries

- There are a number of libraries that are of interest from the point of view of techniques
  - But not necessarily for widespread usage of the techniques or general usage of the library
- For example...
  - Spirit
  - Xpressive
  - MPL
  - *enable_if*
  - Lambda

Boost Books and Conferences

- There are books are based on Boost
  - *C++ Template Metaprogramming* by David Abrahams and Aleksey Gurtovoy
  - *The Boost Graph Library* by Jeremy Siek, Lie-Quan Lee and Andrew Lumsdaine
  - *Beyond the C++ Standard Library* by Bjorn Karlsson
- And the BoostCon (*http://www.boostcon.com*) is ready to run a second year
  - This year it ran 14th-18th May in Aspen
  - Next year, same location, 4th-9th May

Boost Blindspots

- However, in spite of many of its obvious benefits and its beneficial influence, it's not all perfect
  - A comprehensive build system with frequent builds and a diverse range of platform support is a plus, but a slow release cycle is a minus
  - It can produce discussions, libraries and code that are overly expert friendly, which can discourage library users and contributors alike
  - Sometimes too much focus on utilities that are in the domain of language design and not on utilities higher up the food chain of modern software architectures