Louis Dionne

Education

| Jan 2013 – Dec 2015 | B.Sc. Mathematics , Université Laval, Québec. |
|---------------------|---|
| Sep 2011 – May 2012 | B.Sc. Software Engineering (not completed), Université Laval, Québec. |
| | Experience |
| May 2022 – present | Principal Software Engineer (C++ Standard Library), Apple, Québec City. |
| Jun 2018 – Apr 2022 | Senior Software Engineer (C++ Standard Library), <i>Apple</i> , Québec City. Contributor and then technical lead for the LLVM C++ Standard Library (libc++), see below. The library is the standard toolbox used by all C++ programs on Apple platforms, and on some non-Apple platforms such as Android and FreeBSD. The library is deployed to billions of devices worldwide, making it a performance, safety and stability critical component. |
| Jun 2018 – present | Apple Representative on the C++ Standards Committee. |
| Dec 2020 – present | LLVM C++ Standard Library Technical Lead and Code Owner. |
| | • Make and own difficult decisions about API/ABI changes and other design and evolution |
| | tradeoffs Design key parts of libc++'s implementation of several C++20/23 features like base library concepts and ranges |
| | • Port the library to multiple new platforms with embedded flavors while minimizing tech debt in the codebase and maintaining nearly no downstream difference |
| | Automate Apple's internal process to qualify and release the library for use in Apple SDKs, leading to the library being kept closely in sync with upstream releases in the past few years Held a key role in several security-related initiatives like C++ Standard Library hardening and typed memory allocation |
| | Rebuilt the libc++ community and doubled the number of regular contributors to the project by making various infrastructure and policy changes to make development smoother Mentor new contributors and review almost all contributions |
| Jun 2018 – Nov 2020 | LLVM C++ Standard Library Contributor. |
| | Implement various C++ features, fix bugs in the library and represent libc++ on the C++ Standard Committee Move Apple platforms from a years-old libc++ to a current one |
| | • Rewrite the libc++ testing infrastructure to increase flexibility with focus on embedded and other unconventional platforms |
| May 2017 – Apr 2022 | Member of the Boost Steering Committee, <i>Boost.org</i> . Participate in technical decisions impacting the future of the Boost community, but also day-to-day procedural and policy-related issues. |
| Dec 2016 – Jun 2018 | Amazon Representative on the C++ Standards Committee. |
| Jun 2016 – Jun 2018 | Software Development Engineer , <i>A9.com (an Amazon company)</i> , Palo Alto. Member of the search infrastructure team powering Amazon's search engine. Notable contributions: |
| | Replace parts of the internal key-value store with a lock-free prefix tree backed by a variant of RCU and a custom allocator allowing the structure to be stored in a memory mapped file. Achieved throughput improvement of more than 50% for real-time updates. Replace the component that applies real-time updates to search indices by a new one that |
| | scales better. Played key role in the low-level design of a novel query processing pipeline allowing the space and time cost of any query to be bounded. Achieved significant fleet cost reduction. Pioneer a C++ library of high-quality fundamental utilities for use throughout Amazon. |
| Dec 2014 – Jun 2016 | C++ consulting, (finance, embedded systems). |
| | Development of $C++$ libraries to retain a high level of abstraction in applications where both performance and correctness matter. Also some refactoring of existing systems to add new features and/or improve performance. |

| 2014 – 2015 GSoC student with Boost, Goo | ogle Summer of Cod | e. |
|--|--------------------|----|
|--|--------------------|----|

Work on Boost.Hana during the summers of 2014 and 2015 as part of the Google Summer of Code program. I also received a grant from the Boost Steering Committee to continue working during the winter of 2015, which had never been done before for a GSoC student.

Sep 2012 – Dec 2012 **Software Developer**, *Coveo Solutions*, Québec. Work on a MIME parser in C++. Resigned to pursue a degree in mathematics.

May 2012 – Aug 2012 Intern, Coveo Solutions, Québec.

• Conception and implementation of a deadlock detection system for internal use

- Presentations on C++ techniques and idioms to co-workers:
 - The Boost.ConceptCheck library and associated template metaprogramming techniques
 - C++11 rvalue references

Selected talks (full list)

- 2024 Security in C++: Hardening techniques from the trenches (video coming), C++Now, Aspen.
- 2017 Runtime Polymorphism: Back to the Basics (slides/video), CppCon, Bellevue.
- 2016 Closing keynote on metaprogramming (slides/video), Meeting C++, Berlin. Was voted the best presentation by attendees
- 2015 Metaprogramming: a paradigm shift (slides/video), C++Now, Aspen. Awards for the best presentation and the most inspiring presentation
- 2014 Hana: Expressive metaprogramming (slides/video), CppCon, Bellevue.

Selected Personal Projects

Boost.Hana A Boost library for C++ metaprogramming

Designed and implemented a library to manipulate heterogeneous sequences at compiletime and at runtime. The library introduces a new paradigm for expressing metacomputations allowing a very high level of expressiveness with little to no performance penalty.

Dyno Runtime polymorphism done right

Designed and implemented a library for non-intrusive runtime polymorphism in C++. This is also known as protocols in Swift or traits in Rust, and other languages have similar built-in concepts. Amongst other things, the library allows controlling the layout of vtables and the storage of the runtime object (e.g. the well-known small object optimization).

Metabench A simple framework for doing compile-time benchmarks

Implemented a self-contained CMake module to perform compile-time benchmarks of C++ metaprograms. Such benchmarks are very useful when writing a metaprogramming library, where the performance of the library must be measured in terms of compilation time. The module works by having the user write ERB templates that are then used to generate C++ programs. The C++ programs are compiled and various metrics such as compilation time, link time and executable size are gathered. The module generates HTML5 charts to easily visualize the metrics.

mpl11 Conception and implementation of a C++11 replacement for the Boost.MPL

Reimplemented the functionality of the Boost.MPL library using new template metaprogramming techniques made possible by C++11. Redesigned the API of the library using ideas from Haskell to make it more powerful, easier to use and to extend.

Contributions to other projects

- Contribution of the hawick_circuits algorithm to Boost.Graph
- Occasional patches to Boost (Spirit, Graph, Archive, MPL and others)
- Active on the Boost.Dev mailing list
- CMake port of the FastPFor integer compression library's build system
- $\circ\,$ Too frequent bug reports against the Clang and GCC compilers.