The Untold Story of std::bitset: How a Practical MS-DOS Problem Contributed to C++

By Chuck Allison

std::bitset feels like one of those standard library components that must have existed forever — a compact, efficient, type-safe abstraction for working with fixed-size collections of bits. But behind this ordinary feature lies a very human story: one that begins in an MS-DOS application and winds through the earliest days of C++ standardization.

This is the story of how std::bitset came to be.

A Practical Problem in a Constrained World

In the early 1990s I was working as a technical lead in the Information Systems Department of the Church of Jesus Christ of Latter-day Saints in Salt Lake City. Our Mission Management System ran under MS-DOS and was written in C. Memory was tight (640K RAM was the limit in those days) and every byte mattered.

One of my many tasks was to implement drop-down picklists, and I wanted a compact way to record which entries a user had selected. The solution was simple: associate with each picklist a set of bits stored in an integer and provide an interface that treated that integer as an indexable array of bits.

It solved the problem cleanly, and, as it turned out, planted the seed for a future standard library component.

From Local Utility to Committee Proposal

I had just begun serving on the C++ standards committee (then J16) at the time. At the third committee meeting in Lund, Sweden in June 1991, I volunteered to survey existing third-party C++ libraries. When I reported back at the next meeting, we began assigning follow-up work, and I took on the task of exploring bit-handling classes.

It was natural to generalize the bit-array abstraction I had already built. The result was two classes: **bitset** and **bitstring**. I documented them in a paper titled “Two Bit Handling Classes for C++.” Committee member Stephen Clamage, with characteristic wit, joked that these were just “two-bit classes.”

Solidifying Templates

This was the dawn of templates in C++. Language designer Bjarne Stroustrup had, with incredible vision, specified template syntax and behavior, featuring both type and non-type parameters, in The Annotated C++ Reference Manual, which served as the basis for standardizing the language. The power of templates was manifest, but as is always the case, some details had to be worked out to make them standard-ready, even in those early days before template metaprogramming was discovered.

When I presented my initial bitset design, iostreams designer Jerry Schwarz suggested that the number of bits could be a template parameter. That opened my mind to what templates in C++ were capable of. (This was when we were still using macros to implement templates, and few people knew what was coming. The Standard Template Library/STL had yet to migrate from Alex Stepanov's Lisp implementation.) In any case, my proposal gave the committee a concrete use case for fleshing out the semantics of non-type template parameters.

Acceptance and Standardization

I completed the proposal and reference implementation early in 1993, and the committee accepted it in early 1994, about the same time as STL was revolutionizing what C++ would become. When C++98 was published, std::bitset appeared as a fully integrated part of the standard library. To my knowledge, it was the first class in standard C++ to use a non-type template parameter. Bruce Eckel's foreword in my first book, **C & C++ Code Capsules** (Prentice-Hall, 1998), comments on my authorship of std::bitset, providing a contemporaneous record of its origins and its role in the early standardization process.

The Parallel Lineage: bitstring, vector<bool>, and dynamic_bitset

The second class from my original paper, **bitstring**, followed a different path. It evolved into the well-known (and controversial) specialization **vector<bool>** (*not* my idea!). Years later Jeremy Siek packaged my original reference implementation (which he complimented as clean, tight code) for inclusion into the famous Boost library as **dynamic_bitset**, which still carries structural traces of its origins. Boost::dynamic_bitset has been cited under the Boost license in a number of commercial products, including many from McAfee.

A Small Abstraction with a Long Shadow

std::bitset is a reminder that the C++ standard library was not handed down from a monolithic design team. It was built by individuals solving real problems, sharing their ideas, and shaping the language through practical insight and working code. In my case it was a picklist, a few bits, and someone just showing up willing to participate.

Many opportunities came my way because of my years-long association with the C++ Standards Committee, most notably with P. J. Plauger, who invited me to be a columnist for the C/C++ Users Journal, and with Bruce Eckel, who invited me to be a presenter at Software Development Conference and to later co-author **Thinking in C++, Volume 2** with him. Bjarne Stroustrup himself graciously acknowledged my modest editorial contributions in the second and third editions of his Tour of C++.

I often felt quite ordinary in the presence of the great ones who fashioned such an amazing programming language (Bjarne Stroustrup, Andy Koenig, Jerry Schwarz, Tom Plum, Matt Austern, John Spicer, Pete Becker, and then-chairs Dmitri Lenkov and Stephen Clamage, to name just a few—it's a long, long list…), but I'll always be grateful I was in the room where it happened.