

NICK STOUGHTON

update on standards: the USENIX Standards Project



Nick is the USENIX Standards Liaison, representing USENIX in the POSIX and Programming Language Standards Committees of ISO and ANSI. When he is not busy with that, he is a consultant, with over 25 years of experience in developing portable systems and applications, as well as conformance testing.

nick@usenix.org

USENIX HAS BEEN FUNDING ACTIVITIES in standards for around 20 years. The organization has been involved with POSIX since its inception, as well as with the C and C++ programming languages, the Single UNIX Specification, the Linux Standard Base, and several other projects.

Over those 20-odd years, USENIX has gained a reputation and an esteemed position in the standards community. We are held up as the primary champion of free and open source software within the particular niche of programming language and operating system standardization.

USENIX holds numerous senior positions on the various standards committees: secretary to the IEEE Portable Applications Standards Committee, secretary to the Austin Group, International Representative for the USA on all POSIX and Linux matters, co-editor of the C standard, editor of a technical report for the C committee, editor of the Linux Standard Base, inter-working group liaison for all programming languages and POSIX, and chair of the LSB specification authority, to name just some of them. And let's be shameless: although USENIX holds the positions, since it funds the activities, the person actually doing all of these jobs is me!

The annual expenditure for this work represents a sizable proportion of the organization's Good Works budget (about 34% of the total, a sizable proportion of which goes to international travel expenses). Quite reasonably, the board periodically reviews where it is spending the organization's money, and at a recent Open Board Meeting those present were asked if USENIX's support of standards activities was really benefiting the membership.

I believe it is true that the standards that we are involved in affect in some way or other every single member of the organization, every single day they work. Maybe you never notice, but every key-stroke you type has been touched by code written in C. If you are a UNIX user . . . well, the Single UNIX Specification is one of those standards. It is a superset of the POSIX standard. The fact that the same command does the same thing across all versions of Linux, *BSD, HP/UX, and several other systems is because there's a standard. Maybe you use a GUI . . . probably some C++ there. I regularly receive questions and comments from members of various open source communities about POSIX and

C interpretations that lead me to believe that many of you are not unconscious of the standards you use.

Standards are what make open source software successful. They allow that software to be ported from environment to environment with ease, freeing us from having to reinvent the wheel every time we work on a new project. The Open Source Initiative (OSI) (www.opensource.org) has an “Open Standards Requirement” (OSR). I’d like to quote some parts of their rationale for the OSR:

If interoperability is a grand goal as it relates to software, then standards are the critical tools for achieving this goal. . . . At this point in time, it has become largely intuitive across the industry and among users that broad and widely accepted standards are a Good Thing. . . . The purpose of an open standard is to increase the market for a technology by enabling potential consumers or suppliers of that technology to invest in it without having to either pay monopoly rent or fear litigation on trade secret, copyright, patent, or trademark causes of action. No standard can properly be described as “open” except to the extent it achieves these goals.

The industry has learned by experience that the only software-related standards to fully achieve these goals are those which not only permit but encourage open-source implementations. Open-source implementations are a quality and honesty check for any open standard that might be implemented in software, whether an application programming interface, a hardware interface, a file format, a communication protocol, a specification of user interactions, or any other form of data interchange and program control.

The standards that USENIX is currently helping to develop and maintain are all Open Standards by the definitions used by the OSI.

Let’s consider some current projects in which USENIX is involved. POSIX has just completed a revision (only the third since 1988) that has added a number of new APIs coming from the open source world (and, in particular, from glibc). Many hundreds of other issues were addressed at the same time, and several previously optional features have now been mandated. (An aside: Optional behavior is a real nuisance to the end-application developer. If you cannot rely on a particular interface being available, then you have to code around the possibility that the interface is absent, which bloats your code and makes maintenance harder. So, getting rid of options is definitely a Good Thing in my mind.) The POSIX working group (known as the Austin Group, after their first meeting in that city) is staunchly opposed to invention. Everything that goes into the standard must be based on widespread existing practice. Most of the issues that arise are because there is widespread existing practice that implements an API in a different way from that described in the standard. Much of the new revision has been aimed at removing some of the differences between Linux and UNIX.

The C committee is preparing for its second revision since 1989. Like the POSIX committee, there is a very strong resistance to invention by the majority of members. This revision will likely include features such as attributes for functions, variables, and possibly types, better support for concurrency (probably including a thread API that is at worst a thin layer over POSIX pthreads), and other features already supported by most if not all C compilers in one form or another.

The C++ group is struggling to finish its revision, which will make radical changes to the language. Unlike C and POSIX, in C++ invention is not

feared. USENIX has been one of the leading voices in this effort, trying to ensure implementability and usefulness over cool and sexy with regard to new features! Again, like C, concurrency support is a major driver.

As a spin-off of the C++ concurrency work, and trying to fill in where the necessarily platform-neutral aspects of the programming language itself fall short, a new POSIX/C++ binding working group has just been formed. This group will be producing an IEEE standard that encapsulates POSIX threads, file system, and networking (to name just some of the larger features) in C++ objects.

I hope this report helps you to answer for yourself the “Is this a worthwhile effort?” question raised earlier this year. I’d be interested to hear your opinion.