NICK STOUGHTON

update on standards: diary of a standards geek, part 2

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

A YEAR OR SO AGO, I WROTE AN ARTIcle here called "Diary of a Standards Geek." It seemed to be very well received, and I was sent a number of follow-on comments and questions after it appeared in *;login:*. This September I spent three solid weeks at standards meetings, and I thought this format seemed the best to summarize that extremely busy period.

Week 1: C, Santa Clara

So, for 17 of the next 19 days I am scheduled to be at a standards meeting of one sort or another. At least for this week and the next, I'm going to be at least relatively close to home. This week, it is C in Santa Clara, hosted by Cisco Systems. If you know the Bay Area, you will appreciate why I chose to stay in a hotel near the meeting room. The meeting may only be 40 miles from my home in Oakland, but it can be easily 60–90 minutes of driving in either direction!

The C standard is being revised. At the last meeting, the Working Group asked me to be the backup editor, as the primary editor has not been able to get to a meeting for several years. However, today he is here! I can take it easy! Someone else is going to do the heavy lifting. Acting as the editor is not a part of the USENIX Standards budget, and anything I do in this role is strictly as a volunteer.

But I have several papers before the committee, and I'm expected to deliver several more before the process is over. Add to that is that the United Kingdom C panel has found itself unable to send anyone to the meeting, so they've asked me, since I'm a UK citizen, to represent it as "Principal UK Expert" (PUKE) and Head of the UK Delegation. This is a great honor, and one for which I am happy to oblige them.

There are 30 or so people attending the meeting—one of the biggest crowds for a while. That's because there are many local people who are able to participate. We even have Apple here for the first time in as long as I can remember. And we have one person attending all week by teleconference, from the East Coast. Cisco is pretty well set up for hosting this kind of meeting, so we have good connectivity, good teleconference facilities, and an attentive host.

C is right at the start of its revision. We don't want to do anything inventive, just build on existing implementations wherever appropriate. Microsoft is here, but their group is humble enough—its C compiler is well known as one of the least compliant in common usage (being barely C89 compliant, and nowhere even close to C99 compliant). But it still has good data points and input. We definitely all want to do attributes in C, and GCC is the model most of us know and understand, but we've all used Microsoft's `_declspec` as well at some point, and we want to do something that fits both. The C++ committee is also working on attributes, so we are closely watching that group and will probably take its lead here, since it is covering the same space, with many of the same players. Its members are (as usual) being a little more inventive, but if they can blaze the way, there may be adequate precedent to follow.

Microsoft is also trying to bring forward an idea it has to "hide" pointers in secure code. By encrypting pointer values Microsoft can make it harder for an attacker to locate key data structures when reverse-engineering applications, and the company thinks this is a great idea to have in the standard. I point out that by specifying it, via explicit calls to a function called `EncodePointer()`, it is not so much making it harder for attackers but more advertises in big bold letters what data structures the implementer considers important enough to hide! This has to be done with inline code—not function calls! I think we'll see another version of this paper soon.

We spend a lot of time discussing "Critical Undefined Behavior." The C standard currently has many places where "undefined behavior" results. Some of these are benign and provide opportunities for some implementation (i.e., compiler) to provide defined behavior (that is not portable). An example is integer overflow on a two's complement architecture. Everyone knows what happens when you add one to `INT_MAX`. But the standard says this behavior is undefined.

Then there's the just plain wrong undefined behavior, such as dereferencing an uninitialized pointer or, worse, writing to such a pointer. This is what we want to move into a new "critical undefined" class. "Out of bounds stores," the largest section of the critical undefined behavior paper, cannot always be seen by the compiler, but where they can be, the result should always be a diagnostic. However, in safety-critical applications, it is important to do nothing that might ever rely on undefined behavior, of any kind. It is unlikely we will be able to make a significant impact here, but anything we can do, we should.

There are many gray areas in this endeavor, however. The standard currently says that it is "undefined behavior" if pointer addition results in overflow. It turns out that gcc uses this with somewhat surprising results: If you add a pointer and an unsigned integer, the result can never be a pointer to a lower address than you started with. However, there are some embedded system programmers out there who know what happens on their hardware when you add two numbers together and write code like this:

```
// an address somewhere towards the end of memory
void *arena = (void *)HIMEM_ADDR;

void *
myalloc(unsigned int len)
{
  void *ret = arena;
  if ((arena + len) < arena) {
    // overflow happened
    return NULL;
  }
  arena += len;
  return ret;
}
```

However, since it is undefined behavior when a pointer overflows, and this function relies on it to correctly return failure if `len` is too big, the program is nonportable. Worse than that, in gcc's case, is that the `NULL` return is removed as dead code (a perfectly acceptable thing to do for undefined behavior), and the program subsequently crashes when an out-of-bound store happens. We spent a long time discussing this but concluded that this is a case where the compiler was free to do the dead code removal, and the standard should not prevent it. Static analysis would (incorrectly) assume that no out-of-bound store would occur, since this was guarded against.

As we go into the revision process, we want to make sure that we have cleared the decks as far as possible of all of the current open defect reports against C99. We spend a day or so reviewing and finalizing all of the open defects. For the first time since we started the C standard (in 1986 or so), the open defect list is empty!

Week 2: C++, San Francisco

Not one but two distinct meetings are held this week. Sunday: IEEE 1003.27. POSIX-C++ binding. This is a new Working Group looking at the intersection between the standard POSIX operating system environment and the standard C++ programming language. If we can't decide on what thread cancellation means within C++, we can come up with a much narrower definition of what it means when the underlying OS is known to be POSIX. If we do map thread cancellation onto some kind of C++ exception, then we can make it explicitly undefined behavior if an application catches that exception and fails to rethrow it. Of course, that has far-reaching consequences throughout the C++ standard library, since many of those "Throws: Nothing" clauses are now "May throw thread-canceled exception." And, if you catch it, especially in a `catch(...)` clause, you must add that rethrow.

POSIX/C++ is just a one-day meeting, but I'm hosting it (having borrowed a meeting room from one of my other clients). A good crowd of 12 people show up (10 of whom are C++ meeting attendees as well). We agree on a strategy to avoid sending contentious liaison statements back to WG 21, the C++ language committee. We got burnt back in February when we asked its members to take some points into consideration. Let's not do that again! There is nothing contentious here anyway, this time.

Monday: We move on to the C++ language. Both yesterday's and this week's meetings are in San Francisco. At least I can sleep in my own bed! But to get from Oakland to the City for an 8:00 a.m. start I have to leave earlier than usual.

C++ is a big working group (WG). You've heard me complain about its inventiveness and strange procedures in the past. But a number of new things are happening this week:

- Our outgoing convener, who was unable to commit to attending meetings of the parent body (see next week to follow) to represent his group, also has a family emergency this week. This is a pity; we won't be able to give him the send-off he has earned, but he promises to be back next time.
- Our convener in waiting has a longer history and better understanding of the standards development process. With several of us urging him on, he announces that the complex and contentious (and irrelevant) voting procedure this WG has used for the past 15 years is to be simplified into a simple straw poll of all present. This makes understanding consensus (the job of the convener) a thousand times easier, and it removes the most contentious part of the meeting for everyone.
- Hurricane Ike has kept several attendees from Texas at home, at least for the first half of the week. Some strong opinions are missing!

As a result, the meeting is much more relaxed, and consensus is much easier to reach. We are trying desperately to reach a point where we have a draft ready to send to ballot. The published timetable says this is the last meeting before the ballot . . . but are we ready?

The elephant in the room is "concepts." For those following from a distance, "concepts" are a new technique that makes it much easier to write requirements for templated interfaces. And with well-specified requirement statements in your program, you are going to get better diagnostics, easier to understand code, and more portable programs.

But concepts are a *huge* piece of the new standard. They touch *everything* in the library, and the language aspect is also complex, touching many clauses in the core language wording. We have been unable to approve the core language wording for the past 15 months or so, and that has kept the brakes on moving forward with the library aspects.

Given the perceived importance of concepts, we all agreed that even though the core language was not yet approved at the start of the week, library concepts demanded a large subgroup to go off and work solidly, all week, on the various papers for applying concepts to the Standard Template Library.

Friday: Now it's voting time! One of the key differences this week is that the new convener actually knows the process and procedures that a working group should use. In the ISO process, voting happens in exactly two places: at the JTC 1 level (where standards are finally approved) and at the subcommittee level (where drafts are approved). At the working group level, votes are nothing more than straw polls, an indicator of consensus. At best they serve also as an indicator of how a national body might vote at the subcommittee level. In the past, despite several of us who do understand the process and point out the flaws, we've had an extremely complex voting arrangement, where only certain people in the room are allowed to vote, and numbers are carefully counted. This is always a lengthy and contentious process. This time, everyone puts up a hand (or not) and we can get through things quickly, and with reasonable certainty on the level of consensus reached.

Before we start in on the voting, I give a presentation on the entire ISO balloting process, since one of the primary votes is on whether we are ready to have an official Committee Document ballot on the working draft that results from this meeting. If we say "yes," what exactly does that mean? It is a little surprising to find that most of the 50–60 people in the room didn't know this before we started.

All of the concept papers are accepted! This has taken meeting after meeting after meeting to get right, or at least nearly right.

The document is voted out for CD ballot.

We have had probably the most successful and well-run meetings since I joined this working group.

Week 3: SC 22, Milan

Subcommittee 22 is the parent body within JTC 1 for all programming language and operating system standards. I'm supposed to be part of the U.S. delegation, project editor for two documents in its purview, and liaison for all POSIX-related matters. This is an important committee for me to attend, even though it does little, if anything, technical. I have to defend the position of the Linux Foundation, which, following the OOXML debacle, no longer sees much value in this arena. To make matters worse, the new chair of the subcommittee is none other than the editor of that dreadful OOXML disaster.

Topping it all off, the head of the U.S. delegation, who has been with me all through this series of meetings, fell and concussed himself in San Francisco at the end of last week. Nick, can you please take over as Head of the U.S. delegation? Not bad: I've become head of two distinct national body delegations inside three weeks! And this time, I have a real delegation to head, not just a team of one! This is a singular honor, but one that puts me into a difficult spot. Since the new chair has been appointed by the United States, I can't be too outspoken against him. It's time to go into my behind-the-scenes politicking mode. Just make sure another national body knows the things to say! Perhaps I don't have to say anything in the meeting.

I often hold multiple positions at these meetings, representing different organizations, being technical editor, even representing the national interests of some particular country. It is extremely rare that these multiple positions conflict, but every now and then, it is appropriate to ensure which voice you are using.

You have to "dance with them what brung you." Since USENIX is paying the tab for this meeting, it must always be my first and foremost loyalty. USENIX is a member of the Linux Foundation (LF), and the goals of the LF are very much in line with those of USENIX.

I'm not here to complain about OOXML in any position, except possibly that of the Linux Foundation (being the official liaison between SC 22 and the LF). And OOXML is not a product of this committee. But when the Canadian national body starts getting upset that the LF has not brought LSB 3.2 forward as an ISO standard (3.1 is ISO/IEC 23360), I have to figure out how to respond. It's time to pass the U.S. Head of Delegation hat to one of my fellow delegates and state that I am speaking for the LF and not the United States. The chair gets kind of red and won't meet my eye. He's staring at the papers in front of him. IBM has just announced its new "Open Standards Principles" and everyone knows who and what I'm talking about when I mention "our sponsors' serious concerns about the abuse of the process" and the fact that "every activity under JTC 1 must now be suspect." The Canadians end up withdrawing their motion asking the LF to do anything. It's safer to let things take their natural course.

But that was the only point of any contention for the entire week. There are two reasons to participate in this group: to coordinate all of the various programming language working groups and to ensure that "nothing bad happens." This is the group with the power: It is where most of the important votes actually happen. Almost everyone who attends is there to defend his or her position, to make sure that the work really happens where it is supposed to, in the working groups, and to grease the wheels. It is like a board meeting: This isn't the group that actually makes the company's product (and hence its money), but it is the group who decides how that money is spent.

All our other business was cleared and agreed with unanimity. Nothing bad happened . . . and that really is the point of meetings at this subcommittee level.

I even get off a day early, since we finish the entire agenda ahead of schedule. I can actually get to see Milan!