# Linus Torvalds on C++

From: Linus Torvalds <torvalds <at> linux-foundation.org>
Subject: Re: [RFC] Convert builin-mailinfo.c to use The Better String Library.
Newsgroups: gmane.comp.version-control.git
Date: 2007-09-06 17:50:28 GMT (2 years, 14 weeks, 16 hours and 36 minutes ago)

On Wed, 5 Sep 2007, Dmitry Kakurin wrote:
>
> When I first looked at Git source code two things struck me as odd:
> 1. Pure C as opposed to C++. No idea why. Please don't talk about portability,
> it's BS.

*YOU* are full of bullshit.

C++ is a horrible language. It's made more horrible by the fact that a lot
of substandard programmers use it, to the point where it's much much
easier to generate total and utter crap with it. Quite frankly, even if
the choice of C were to do *nothing* but keep the C++ programmers out,
that in itself would be a huge reason to use C.

In other words: the choice of C is the only sane choice. I know Miles
Bader jokingly said "to piss you off", but it's actually true. I've come
to the conclusion that any programmer that would prefer the project to be
in C++ over C is likely a programmer that I really *would* prefer to piss
off, so that he doesn't come and screw up any project I'm involved with.

C++ leads to really really bad design choices. You invariably start using
the "nice" library features of the language like STL and Boost and other
total and utter crap, that may "help" you program, but causes:

 - infinite amounts of pain when they don't work (and anybody who tells me
   that STL and especially Boost are stable and portable is just so full
   of BS that it's not even funny)

 - inefficient abstracted programming models where two years down the road
   you notice that some abstraction wasn't very efficient, but now all
   your code depends on all the nice object models around it, and you
   cannot fix it without rewriting your app.

In other words, the only way to do good, efficient, and system-level and
portable C++ ends up to limit yourself to all the things that are
basically available in C. And limiting your project to C means that people
don't screw that up, and also means that you get a lot of programmers that
do actually understand low-level issues and don't screw things up with any
idiotic "object model" crap.

So I'm sorry, but for something like git, where efficiency was a primary
objective, the "advantages" of C++ is just a huge mistake. The fact that
we also piss off people who cannot see that is just a big additional
advantage.

If you want a VCS that is written in C++, go play with Monotone. Really.
They use a "real database". They use "nice object-oriented libraries".
They use "nice C++ abstractions". And quite frankly, as a result of all
these design decisions that sound so appealing to some CS people, the end
result is a horrible and unmaintainable mess.

But I'm sure you'd like it more than git.

            Linus

---

From: Linus Torvalds
Subject: Re: Compiling C++ kernel module + Makefile
Date: Mon, 19 Jan 2004 22:46:23 -0800 (PST)

On Tue, 20 Jan 2004, Robin Rosenberg wrote:
>
> This is the "We've always used COBOL^H^H^H^H" argument.

In fact, in Linux we did try C++ once already, back in 1992.

It sucks. Trust me - writing kernel code in C++ is a BLOODY STUPID IDEA.

The fact is, C++ compilers are not trustworthy. They were even worse in
1992, but some fundamental facts haven't changed:

 - the whole C++ exception handling thing is fundamentally broken. It's
   _especially_ broken for kernels.
 - any compiler or language that likes to hide things like memory
   allocations behind your back just isn't a good choice for a kernel.
 - you can write object-oriented code (useful for filesystems etc) in C,
   _without_ the crap that is C++.

In general, I'd say that anybody who designs his kernel modules for C++ is
either
 (a) looking for problems
 (b) a C++ bigot that can't see what he is writing is really just C anyway
 (c) was given an assignment in CS class to do so.

Feel free to make up (d).

        Linus