

Re: Crash for Armidillo

Source: <http://sci.tech-archive.net/Archive/sci.space.policy/2004-08/1246.html>

From: Alan Anderson (*aranders_at_insightbb.com*)

Date: 08/13/04

Date: Fri, 13 Aug 2004 01:58:29 GMT

> *sib101@gmail.com (Jack Sprat) wrote:*

> > *Honestly, that's simply an absurd thing to post. If I'm developing
> > medical software where lives are at stake, and I get a compile error,
> > should I wring my hands in despair that I'm not taking it seriously
> > enough?*

> >

> > *No, obviously, because I'm in development. At that stage of the
> > process, errors are expected. It's the end-product that matters, and
> > late-stage testing is different from early-stage testing.*

I'd have to say that your analogy is flawed. A compile error is akin to finding that a part does not fit correctly, an electrical connector is miswired, or some other relatively "static" problem. This was more like a run-time error, a missing bounds check on an array reference or a division by zero.

A compile error catches mistakes before the program can do anything dangerous, and is rarely a cause for worry. A run-time error during development of something like a medical monitoring device is often a minor disaster, since if you **were** taking it more seriously, you'd already have written the code to detect and deal with anomalous events. But it's almost never a **major** disaster, because you can usually get enough information about the error to ensure that it won't happen again.

jamcguir@yahoo.com (Jake McGuire) wrote:

> *If you're developing medical device code, and the attempt to build an
> image for QA fails with a compilation error, you should be **very**
> concerned, as there's no excuse for a developer to check code into
> your version control system that doesn't compile. If he didn't
> compile his changes, he certainly didn't test them.*

ahem

If YOU are the one working on the code, you're obviously making changes that aren't yet checked in. A compile error is usually due to something as simple as a typo, and isn't something I would get worked up over.

(Unless I were writing in C, that is, because a simple typo in a line of C often results in a perfectly valid line of code with a meaning quite different from the intent.)

- > *And it's very hard to retrofit reliability into a design. It's much easier to design it in from the beginning.*

If you don't have a good collection of data on what parts of a design are likely to fail, you don't have a good idea where to concentrate your efforts. Finding the weak links (by breaking them) and redesigning them to be more robust is definitely more effective than analyzing the whole thing to death but never getting around to building it.

- > *What data did he gather, other than "rocket engines don't work so well without propellant", "my vehicle is indeed aerodynamically unstable", and "our vehicle will not survive an unpowered fall from 1000 feet"?*

I saw lots of interesting stuff about how engine performance is affected when the catalytic elements are distributed badly, and how the temperature sensing system can give invalid readings, and how rough acceleration can fool the control system. All can contribute to a more reliable vehicle in the next iteration.

- > *And was that data worth five weeks of downtime?*

Arguably only three. The known flaw was already being addressed, with a part that takes a couple of weeks to fabricate.