

## Re: OK, it's the matter of database structure rather

---

*Source:* <http://coding.derkeiler.com/Archive/Tcl/comp.lang.tcl/2007-09/msg00816.html>

---

- *From:* "Larry W. Virden" <[lvirden@xxxxxxxxxx](mailto:lvirden@xxxxxxxxxx)>
  - *Date:* Wed, 19 Sep 2007 12:12:59 -0000
- 

On Sep 18, 3:45 pm, cla...@xxxxxxxxxx (Cameron Laird) wrote:

"Better code with less risk of defect" comes at a cost. There's abundant, even overwhelming, anecdotal evidence that organizations are unwilling to pay many or most of those costs, in many real-world situations.

I think that one can make an argument in fact for stopping at some point. Whether the majority of businesses go that far or not is a different issue. However, attempting to get "all the bugs" out of a program seems to be one of never ending deal (you know, like dividing a non-zero number by 2, over and over again).

There has to be a point where the cost of the software development becomes more than the price paid if something goes wrong.

Unfortunately, there isn't a standard way of determining what the price will be if something goes wrong. From what I've read, there are methods one can attempt to quantify an estimate of the remaining bugs, but I've not seen any work done on quantifying where the remaining bugs are. And even if one "knew" where all the bugs are, I would still have my basic mistrust of the information – in my experience, I see regular occurrences of things that typically are not considered (what will the program do if the hardware memory begins to fail, if a disk drive scribbles over part of the file, if power goes out at some random part of the execution, etc.) As soon as one adds those things, something else comes up (bug in the hardware ROM of the CPU, operating system patch causes new return code – or introduces new bug in behavior of system function, etc.)

.