

developer.***A Web Magazine for Software Developers**

The Human Impact of Software

By Daniel Read

My very first job in the computer software business was as an entry level help desk technician. I had been a computer user for many years (since Dad brought home the family's first Tandy Color Computer), but truly I knew very little about how computers worked. Sure, in high school and college I had written a few BASIC and Pascal programs, but none of that knowledge had stuck with me. At that moment, I was vastly under qualified to support my new employer's vertical market accounting software.

I joined this tiny software firm on the cusp of the 1.0 release of their first application. If I remember correctly, when I came on board they were in the process of running the floppy disk duplicator day and night, printing out address labels, and packaging up the user documentation. As I pitched in to help get this release out the door, little did I know that I was about to learn a lesson about software development that I will never forget.

The shipments all went out (about a thousand of them I think, all advance orders), and we braced ourselves for the phone to start ringing. In the meantime, I was poring over Peter Norton's MS-DOS 5.0 book, which was to become my best friend in the coming months. We knew the software had hit the streets when the phone started ringing off the hook. It was insane. The phone would not stop ringing. Long story short, the release was a disaster.

Many people could not even get it installed, and those people who could were probably less happy than the ones who could not. The software was riddled with bugs. Financial calculations were wrong; line items from one order would mysteriously end up on another order; orders would disappear altogether; the reports would not print; indexes were corrupted; the menus were out of whack; cryptic error messages were popping up everywhere; complete crashes were commonplace; tons of people did not have enough memory to even run the application. It was brutal. Welcome, Dan, to the exciting world of software.

Eventually, we just turned off the phones and let everyone go to voice mail. The mailbox would fill up completely about once an hour, and we would just keep emptying it. We could not answer the phones fast enough, and when we did, people were just screaming and ranting. One guy was so mad that several nights in a row he faxed us page after page after page of solid blackness, killing all the paper and ink in our fax machine.

It took us months to dig us out of this hole. We put out several maintenance releases, all free of charge to our customers. We worked through the night many times, and I slept on the floor of the office more than once. Really the only thing that saved us was our own tenacity and the fact that our customers did not have any other place to go. Our software was somewhat unique.

It was obvious to everyone in our company what caused this disaster: bad code. The company had hired a contract developer to write the software from scratch, and, with some help from a couple of his colleagues, this guy wrote some of the worst code I have ever seen. (Thom, if by some slim chance you're reading this, I'm sorry man, but it was *bad*). It was total spaghetti. As I learned over the years about cohesion, coupling, commenting, naming, layout, clarity, and the rest, it was always immediately apparent to me why these practices would be beneficial. Wading through that code had prepared me to receive this knowledge openly.

I stayed with the company for three years, and we eventually turned the product into something I am still proud of. It was never easy, though. I swear I packed ten years of experience into those three years. My time working with that software, that company, and the people there who mentored me have shaped all of my software development philosophies, standards, and practices ever since.

When I got some distance from the situation, I was able to articulate to myself and others the biggest lesson I learned there: software can have a huge impact on the lives of real people. Software is not just an abstraction that exists in isolation. When I write code, it's not just about me, the code, the operating system, and the database. The impact of what I do when I develop software reaches far beyond those things and into people's lives. Based on my decisions, standards, and commitment to quality (or lack of it), I can have a positive impact or a negative one. Here is a list of all of the people who were effected negatively by that one man's bad code:

- Hundreds of customers, whose businesses were depending on our software to work, and who went through hell because of it.
- The families of those customers, who were deprived of fathers and mothers that had to stay up all night re-entering corrupted data and simply trying to get our software to work at all. (I know, because I was on the phone with them at three in the morning.)
- The employees of these customers who had to go through the same horrible mess.
- The owner of our company (who was not involved in the day-to-day operations), whose reputation and standing was seriously damaged by this disaster, and whose bank account was steadily depleted in the aftermath.
- The prominent business leaders in the vertical market who had blindly endorsed and recommended our software—their reputations were likewise damaged.
- All of the employees of our company, for obvious reasons.
- All of our families, significant others, etc.—again for obvious reasons.
- All of the future employees of the company, who always had to explain and deal with the legacy of that bad code and that disastrous first release.
- The programmer himself, who had to suffer our wrath, and who had to stay up all night for many, many nights trying to fix up his code.
- The family of that programmer (he had several children) who hardly saw him for several weeks.

- The other developers (including myself) who had to maintain and build on that code in the years to follow.

That's a lot of people, numbering in the thousands—but only one developer's code.

###

Daniel Read is editor and publisher of the **developer.*** web magazine. He lives in Atlanta, GA, where he works as a software architect. He is currently at work on a book about software development crafted for a business audience.