Bad QA

by Howard I. Cannon (hic@iname.com)

The automated alert came in at around 2:45PM. I was near the Operations Center and decided to head over there to check out the situation first hand. On my way, I ran into the VP of Sales.

“Hey, I just wanted to say a big thank you for getting the release out so quickly. Our customers were really breathing down my neck. And we’re getting a great response from the reviewers about the new specs.” He grasped my hand and shook it vigorously.

I glared at him. “What an ass,” I thought. He damn well knew my feelings about our shortened QA cycles. “Thanks,” I said, “I’m in a hurry. Catch you later.” I pulled my hand from his and walked away smirking.

I swiped my badge to open the OpCen door. I’d been there many times before, especially just after a release. Still, I always found the darkened room with its walls of screens and banks of consoles to be quite impressive. This time there were more yellow and red alerts than I’d ever seen before.

My company sells the autonomous control software piloting about one third of all makes and models of cars. That accounts for almost two thirds of all cars on the road. We had just pushed out a minor software release with some highly-requested user interface features. I was the Quality Assurance Lead on the project.

We centrally monitored road incidents, almost all of which were near misses or otherwise benign. Not that day. The yellow warnings popped up a few hours after release. By the time I had arrived we were seeing a major crash every ten minutes, thousands of times the normal rate. And then it got worse. Pretty soon we were up to an incident a minute. That’s big, huge, enormous. Reports of serious and fatal accidents started making the news.

A quick check showed that almost all of cars involved were running the new software. “Damn it,” I screamed, “who has the authority to roll back this release.” No one knew. The OpCen director just shrugged. I made a few pointless phone calls then jogged down the hall to the Senior VP of Operations office.

I quickly explained the situation and told her we had to pull this release, now! Even with my constant pleading and cajoling it took over an hour to get all of the required signatures and approvals. It took another hour before the incident rate started to show a meaningful downtick, and another few hours before the OpCen was mostly green again. We all breathed a sigh of relief. Someone passed around the Scotch.

Alas, I knew something like this would happen eventually. Given the number of car models we support, it's tricky to meet the demand for constant updates driven by the immediacy of the web and mobile apps. Imagine the time and cost of running a full QA suite for every minor feature on every combination of car and option package. Early on that's exactly what we did – we tried to jam as many features as possible into each yearly release and QA them together.

Pretty soon, management realized that for minor changes unrelated to the core driving autonomy we really didn't have to test that exhaustively. We moved more of the testing to simulation and only tested on representative vehicles. All parties in the ecosystem quickly realized that this was a huge win-win-win so they gave us a nearly unlimited budget to develop better and better simulation technology.

The thing about a slippery slope is that you generally don't realize how steep it is until you're careening out of control. As the simulations got more and more accurate we did less and less QA on the actual vehicles. After two years of finding exactly zero new failures when running on the real cars we started to send most software releases directly from simulation to the vehicles in the field.

At the time, I had over 30 years of experience testing and releasing sophisticated non-deterministic learning systems. I knew that testing only in simulation is a huge mistake. And yes, I protested loudly enough to get a stern warning to "just shut up and do my job." It was a good job, so I held my nose and certified new releases based solely on the results of the simulations. Still, I tried to test on actual cars as much as I could, even after the software had been released into the wild.

The fateful release had some "minor changes to the way the GUI subsystem commands the driving autonomy subsystem," according to the engineering release notes. The simulations passed with flying colors. No changes were made to the code or data of the autonomy subsystem itself. We even took the time to run the release with a few cars on the standard obstacle course. Everything seemed fine so my team recommended we go ahead. I signed on the electronically-dotted line. The car companies rubber-stamped the rollout and within an hour the system started to push the update to a large fraction of the vehicles on the road. And then the carnage began.

Since I'd approved the release I figured this was it for me – I would make the perfect scapegoat – but circumstances turned in my favor. For one, the public ire was directed towards the car companies, not us. Years ago we decided not to become a brand. We eschewed "powered by" tags and required each car company to skin the software according to their standards. The auto manufacturers were so desperate for our tech we were able to build strong liability limits into the contracts. So, after this debacle, the manufacturers turned to us and said "how can we make sure this never happens again" and "how much money do you need?" I had the credibility to respond.

Then there was the small matter of the protest emails I had sent when we first instituted the simulation-only testing protocols. Executive management really didn't want those to get out. Given how hard I'd fought at the time I'm sure they assumed I had archived the threads and would use them for leverage. They came to me with some stronger confidentiality agreements related to my new promotion to VP.

The final technical report showed the risk of simulation-only testing. It turned out there was an incredibly subtle timing problem with the autonomy interface that was tickled by the new usage pattern, but only in the real cars. They enhanced the simulator and sure enough the bug showed up almost immediately.

I found it both fascinating and stomach-turning.

It will probably take several years before they forget this unfortunate incident, start to trust the simulator again and make the same mistakes. I'll fight hard, but I'll lose. Such is the tech biz.