RC24923 (W0912-125) December 23, 2009
Computer Science

IBM Research Report

The Dark Side of Software Engineering

Grady Booch
IBM Research Division
6533 West Prentice Avenue
Littleton, CO 80123

Essay for *The Dark Side of Software Engineering*
Grady Booch

In 1977, I was declared an officer and a gentleman by act of Congress, having just graduated from the United States Air Force Academy with a bachelor's degree in computer science. I'm no longer an officer – although hopefully still a gentleman – but in my very first assignment as a second lieutenant, I encountered the dark side of software engineering.

By way of context, I need to explain that integrity was a concept that was central to my Academy experience. USAFA enforced an honor code that stated "we will not lie, cheat, or steal, nor tolerate anyone among us who does" and by enforced, I mean that as a cadet we took that code deadly seriously. Cheating on a test was obvious grounds for dismissal, but even fudging about when you signed back on to campus after leave was deemed an ethical breach. Now, attending a service academy is not exactly your typical undergraduate experience (I should also mention that, for all but my senior year, USAFA was a male-only institution) but for the cadets, the code was an important element in shaping us as warfighters.

After graduation, I didn't become a pilot – I knew that I'd always fly a desk because my eyesight was not 20/20 - and indeed I'd planned for that, as I knew that USAFA would offer me an excellent foundation in computer science and I'd be able to work on some infinitely cool software-intensive systems. My first assignment bore that out: I was directed to Vandenberg AFB in central California, attached to what was then called the Space and Missile Test Center (SAMTEC) where I served first as a project engineer for a new telemetry system and then as program manger for a new range safety display system (called, in the spirit of blindingly obvious military acronym-ese, RSDS).

This met my test for infinitely cool. Here I was a twenty-something with a multimillion-dollar budget that pushed the edge of what we imagined we could do in the space program, namely, to build a system that that, in real time, fused sensor data from dozens of radars around the globe and presented it up using one of the earliest color vector displays from Evans and Sutherland. And yes, just like in the movies, there was a big red button that the range safety officer would push if the bird went off mission. This was a complex distributed system, long before the protocols we now take for granted were ever in place.

A brief aside: I got my first email address in 1979 on what was then the ARPANET which at the time had only about two hundred nodes in the entire network, and thus had a printed directory of every email address in the world. So yes this was a wickedly fun time that was ahead of its time.

As was typical for these sorts for things, the Air Force outsourced the development work: in the project office, we set the specification and then went out for bid, planning then to manage the effort from vision to deployment.

And this is where I had a startling collision with the dark side.

The specs were out in the wild; we had received responses from all the bidders, and were then going through our selection process. One day, I received a call from one of the bidders – not unusual; we would often have to field technical questions for it's not like our specifications exhibited the perfect clarity as they do in contemporary systems (yes, I'm being sarcastic) – yet this call was a bit different. After the usual pleasantries and then questions, the bidder made mention that they had expected their stock to rise (red flag number one) and that I should consider investing (red flag number two), and by the way, if as a lieutenant I didn't have the funds to do so, he'd find a way to help me (red flag number three, with all sorts of loud klaxons going off in my head).

I was stunned, having never imagined anyone so blatantly trying to bribe an officer, especially given how USAFA's honor code was a part of me. I had been conditioned to expect the best from my colleagues, that they would always act in an ethical manner, especially those folks involved in pure technology that was clearly free from moral consideration. Even while he was talking, I began to enumerate the dozens of federal laws this bidder had violated. After I got off that call, I immediately created a memo (remember, this is in pre-email days) then called my security officer, who calmly took my report.

And then we planned a sting operation to nail this malfeasant.

I scheduled a call back with the bidder, only this time the call was wiretapped and recorded, and he indeed again made an incredibly clumsy attempt to bribe me, which was more than enough to hoist this man by his own petards.

I understand that he spent some time in a federal prison, and no, his company didn't win the bid.

To frame this in contemporary language, here we had a classic outsourcing deal, tainted by an unethical bidder who, so hungry to win, resorted to an incredible breach of ethics. For me, this was quite an introduction to the dark side of software engineering.

In college, I'd learned all about the craft of software development, and over the years have honed my craft. But, this early experience shaped my understanding of the human element of developing complex systems. Over the years, I have witnessed death marches rising from incompetency, design decisions made out of emotion or avarice, posturing by CEOs and project managers and code warriors out of fear or for political gain or as a means of petty revenge – and even an outright ethical breach of a profoundly stupid bidder.

As I consider what separates us from take the vision of a software-intensive system and then turning that into raw, running, naked code, I recognize that some things, such as the laws of physics – are fundamental constraints, while others – such as how we design a particular algorithm or architect a system of systems – are classical problems of systems engineering. And yet, what this experience tells me – and other more subtle ones over the years confirm – is that engineering software-intensive systems is ultimately a human activity, and thus subject to all the drama associated with the human experience. For me, the dark side of software engineering springs not from the technology itself, but rather from the very human ways we architect our systems, architect our organizations, and attend to the moral and ethical dimensions of how we use the technology we create.