Memoirs of a Self-Loathing IT Professional

By Bernie Wieser

Legacy

We missed Dean in the meetings. His cutting wit had a way of getting to the heart of issues. Dean had moved on to be an IT Solutions Architect at a competitor. He had a good grasp of how software fit together and was knowledgeable about *tech du jour*, even though his formal training was simply a programming certificate from the Southern Alberta Institute of Technology. Dean's biggest strengths were that he was a good, honest communicator and excellent team player. We also missed the stories about his rock band, and his total disregard for the company dress code.

The new guy, Kenneth Bradley, was not a team player and made us feel Dean's absence even more. In our first group meeting where Anne introduced him, he spent 15 minutes telling us why he was better than us. "I've made my own design patterns," he said at one point, droning on about his achievements. I guess he figured that none of us knew what design patterns were. You don't arbitrarily make them; they are observed abstractions from common solutions. If Dean were there I'm sure he would have pointed out that Kenneth's revelations were reinventing-the-wheel (or completely stupid). I made the mistake of saying, "Hello Ken," once while passing Kenneth in the hall and was quickly dismissed with, "It's Kenneth." He was an odd looking fellow; slightly balding and he seemed to have an affinity for polyester pants that were too short.

At first Todd and I believed we would be divvying up Dean's work, but Anne gave all of it to Todd. She figured because Dean and Todd worked in the same room that Todd would just know what Dean was doing. Todd wasn't very happy about it. "This is what you get from a six week programming course," he would complain when looking over Dean's code. However I knew that Todd appreciated Dean's skills. Dean knew how to make things work and his passion was evident in his craft.

Alan's departure to the PMO left me with everything he was working on. On the one hand it was good because I was coding again on a few projects, but on the other hand it was bad because I had twenty applications to support. I took the support role more seriously than my counterparts and actually tried to become a power user in each system I supported. I wanted to make my clients happy. Anne always said *make the clients happy.*

Support was difficult for a couple apps where the clients didn't let me have elevated access so I could administer the apps properly, or give me the same access they had so I could support the client properly. "It is not necessary for IT to access business functionality", one client told me in a meeting while I was trying to help them troubleshoot a problem. I wanted to ask, *And I can help with my hands tied behind my back?*

It's funny how many business clients think their IT staff are shamans and can work technical magic. It's funnier how many IT people come to believe that they are. More than once I witnessed an IT colleague explain to a client with authority and mystical jargon what was obviously random fiddling. I could not figure out if it was *show no weakness* posturing or faith that the fiddling actually worked. I had the bad habit of being completely honest with my clients. If I couldn't figure something out immediately I'd tell them I'd come back later after I had more time to investigate the problem.

When the business client couldn't figure something out, I would get the ticket from the help desk to go figure it out. Our policy was to discourage direct phone calls and let the help desk record "the incident." A lot of important information never got transcribed into the tickets so it usually ended up with a phone call and a visit anyway. Sometimes this meant watching a client over their shoulder and eventually making a phone call to a vendor, but then some of our systems were built in-house so I didn't have anyone to talk to for those. For the in-house

systems I often had to look at the code to figure out what was going on because the documentation was poor or non-existent.

Two in-house systems that I inherited from Alan gave me a few grey hairs. One was an application that managed gas trading transactions called GASMAN. The other application, called SNUPEA (pronounced *snoo-pee*), allowed our competitive intelligence folks to compare our company against our peers with focus on the *reserves*. Reserves are a special topic in the energy industry because it's the amount of oil or gas that can be economically extracted from a reservoir or well. It was important because you could see future potential, if the economics were right. It was a very exciting topic for the competitive intelligence folks.

GASMAN was originally written by a fellow named Dmitri Isayev. I never met Dmitri but I had heard about him through our client Bob. Bob, the group lead, spoke glowingly about Dmitri and according to him Dmitri was the smartest IT guy around. He had a PhD in mathematics and was "brilliant." He understood every requirement Bob gave him in the three years Dmitri took to write GASMAN, including some of the complex calculations (although I never got to see how complex they actually were). I had the feeling Dmitri could walk on water. Dmitri had left Banana almost a year before I got there. I heard he went to work in the financial industry in Toronto. Alan inherited Dmitri's legacy and everything was going well, at least Alan never heard from Bob nor any of the other traders about GASMAN.

Providence waited for my proverbial shift before Bob called with an unusual dilemma. Dmitri had implemented a sophisticated, granular security on the transactions. Only certain individuals were allowed to see certain data within a transaction and Dmitri had implemented this mathematically inspired security infrastructure. Bob complained that one of the junior traders was seeing information he should not, so I set about trying to figure out why. I anticipated my call to Alan would be unproductive (and it was). It *worked per specification* according to him. So I jumped into the code with my handy-dandy debugger tool. I found the security module. I put in my break points to stop the program when the security code got called. To my surprise, the program whizzed on by without stopping.

My first thought was that I had done something wrong in configuring my debugger, so I fiddled a bit with no change. Then I looked closer at the code through my integrated

development environment. I searched for where the security code was called. I found it was never called from anywhere. I did a deeper dive only to discover that a lot of it was never fully implemented, and what was implemented would *never* work. For all intents and purposes there was no data security in GASMAN.

I tried to find out if GASMAN was ever code reviewed. No one knew, and there were no documents to artifact that anyone aside from Dmitri looked at it. It didn't even seem to have user acceptance testing or client sign off. Somehow it got into production without quality assurance (that really is due diligence). When I asked Anne what I should do, she told me to talk to Bob so I scheduled a meeting.

"Hey Bob, I wanted to talk to you about the security issue in GASMAN," I said. I had met Bob a couple times before. He was a portly fellow and outwardly quite jovial. I hoped this meeting would be the same.

"You found it?" he said somewhat excited.

"Well, yes. The security portion of GASMAN was never completed," I replied.

"Sure it was. I saw it working," Bob replied. "Dmitri showed me the test cases."

I wasn't sure how to respond. Maybe it had worked at some point but Dmitri could have forgotten to check in the code under version control. "I went through the code Bob and traced it. There really is no security function."

"That's simply not possible," Bob said with apparent frustration. "You must be wrong."

"Trust me Bob," I said, "I spent a lot of time going over it and it looks like it was never finished. The security code is not called by any other part of GASMAN."

"Dmitri wouldn't do that," Bob stated flatly. "I saw it working. I want someone else to look at it."

I was stunned into silence for a moment but then I just blurted out, "Listen Bob, believe what you like, but why don't you try it yourself? Any user can see anything, and it's not a bug.

If Dmitri did have it working, he didn't leave the code behind, and it's not what is running in production."

"I will," Bob said gnashing his teeth. Our meeting was over.

The next day I arrived to see a meeting request in my inbox from Anne concerning GASMAN. When I met her in her office she told me that Bob had complained about my unprofessional demeanor: *Why don't I like Dmitri? Why would I lie about Dmitri?* I recalled and recited my meeting with Bob in great detail.

"So you see - one way or another - it doesn't work, and it looks like it never worked. It's dead code," I told Anne. "I don't know how else I could have explained it to Bob."

"I know, I trust you," Anne replied. "The client really holds Dmitri in high regard."

"So what do you want me to do? I could show Bob…" I started to say when Anne interrupted me.

"Bob won't understand what he's looking at. Heck, I wouldn't understand what I'm looking at if you showed me – we're not developers. You need to keep it simple."

"How much simpler is it than *it's plain busted?*" I asked rhetorically.

"Well, you could try by being less technical and more accommodating," Anne recommended.

"We work in IT; T is technical!" I exclaimed in frustration. "I can't shake a magic stick and make it work. Do you want me to validate Bob's feelings by saying what a great guy Dmitri was and make up some story as to why it's FUBAR?"

Anne could see me struggling. "To make things easier, and reduce the heat a little, I can assign this to Kenneth."

You are going to pour gasoline on a freaking fire? I thought. *The guy has no people skills whatsoever.* "That's fine with me," I said. I didn't know I could feel so relieved and so pissed off at the same time.

SNUPEA was not as big a program as GASMAN. I'm not sure why Alan was developing and supporting it; the competitive intelligence team didn't have anything to do with gas marketing. I realized Alan wasn't the original developer when I looked at the documentation; it was good. The requirements were solid and the design cross-referenced them. However SNUPEA, under Alan's reign, changed significantly through the guise of maintenance, and none of these changes aligned with the original requirements. Alan was bolting things on as the client requested them and I could tell that the *kludges* were pushing the original design to its limits.

I called a meeting with Anne to discuss SNUPEA's future.

"Anne, SNUPEA is getting to the end of its life. It has been *enhanced* so many times it's a patchwork. It's going to be harder and harder to maintain or enhance," I said.

"What do you propose?" Anne asked me.

"We should revisit the requirements with the client and do a major upgrade. That really means getting a Business Analyst involved to do a full review, design, and build. I suspect we'll need to significantly change how it works under the hood to enable the new functionality the client wants," I replied.

"I see," Anne started. "I met with the competitive intelligence team last week. They're pretty happy about how it works now. They have a lot of the same functionality in their spreadsheets. All we have to do is make it work like the spreadsheets."

I didn't think that was a good idea. "But Anne, they've been asking for historical calculations and prior period adjustments that you can't easily do in a spreadsheet. SNUPEA wasn't built for that either. We're not managing the changes because we don't have a clear view of the requirements."

"Just make it work like the spreadsheets," Anne instructed me. "We don't need a *do over*."

I was somewhat baffled. "You can't put lipstick on a chicken. I'm not sure I can do that," I said. *It is too complicated*, I thought.

"Well, Kenneth isn't loaded yet, why don't you transition the project to him?" Anne suggested.

I wasn't sure if this was Anne being helpful or punitive. Even though I had spent significant time figuring out the application, I agreed to pass it on to Kenneth. I booked a meeting to talk about transition planning, and a few other meetings to do a full brain dump of how SNUPEA worked, how it had drifted from its original intent, and how it should be fixed. Kenneth was happy to pick up another programming task. "It will be easy," he told me.

I met him in his office to help him set up the project. I was immediately struck by something strange. His desktop wallpaper was an unflattering picture of a toy poodle. The dog was obviously old and it looked like it had mange. He also had a picture of himself and the dog on his desk, much like a picture some people have of their children. *Eccentric dog lover,* I thought. I asked him, "What's your dog's name?" He replied, "Mr. Gates." I left it alone.

A few months passed and work seemed to be going well. Kenneth continued talking it up in the status meetings. He had made Bob happy by promising to "fix the bugs" in GASMAN. *What bugs?* I thought. The Bob problem went away, but the *defect* remained on Anne's to-do list. Kenneth seemed to be putting more and more effort into SNUPEA. He had a queue of changes but hadn't apparently progressed. On a few occasions I offered help, but Kenneth never asked for help nor took me up on my offer.

Todd, Jeff and I had been lobbying Anne to implement group peer code reviews for our development projects. We also wanted to include folks from other teams in these reviews but she was reluctant; she saw no value in it. However, as timelines slipped, she seemed to be warming to the idea as it might give her more insight as to what certain members of her team

were doing. She suggested SNUPEA be the first candidate for review, and that I should meet with Kenneth to see how things were going.

When I booked a meeting with Kenneth on SNUPEA he was irritated. "I don't need anyone looking over my shoulder," he told me.

I tried a softer, gentler tone to see if I could get more information on the blockers. "We're just trying to help. Sometimes if you get stuck it's easier to talk with people about it."

"I'm not stuck," Kenneth said. "Not anymore."

I looked at him inquisitively. Then he dropped the bombshell on me.

"I couldn't quickly figure it out, so I re-wrote it," Kenneth said.

"You did what?" I asked in disbelief. "You thought it would be faster to re-write than figure out?"

"Well, I wasn't sure where to change SNUPEA to make it do what they were asking, so I rewrote it from scratch. That way I understand it. I made it my own. It almost does exactly what it did before but better," Kenneth said proudly.

"Better in that it does less than it did before?" I asked. *And in more time*, I thought.

"Better in that I can make changes faster," he replied snootily.

"Does Anne know you re-wrote it?" I asked.

"Yes, I told her I was improving it," he said.

"Did you update the requirements or document your design?" I inquired.

"Nah, I don't need to," he replied. "The code is self-documenting."

"Okay," I drawled. I had to leave Kenneth's office. On the way back to my office, with an obvious look of bewilderment on my face, I kept revisiting something I couldn't bring myself to discuss: *You couldn't figure out the code before you got your hands on it but the code you*

wrote is self-documenting. It does less than before and the previous good front-end work is now useless. How is this improvement?

When I told Anne what had happened she said she would have a talk with Kenneth. I'm not sure how she could justify the cost of Kenneth's work to the client or the delays. The client was getting irritated with the lack of progress. I thought, *Maybe this is an example of it's easier to ask for forgiveness than permission.* I wanted to rewrite SNUPEA, but I wanted to do it the right way, by engaging the client and using due process.

I needed to consult the remaining dudes. Because of the office move, they each had their own office and were becoming less attached, so I corralled them into a meeting room. "Hey guys," I started, "I have a bit of a conundrum. I wanted to rewrite SNUPEA but I wanted to do it right by using an analyst, making a new plan and documenting it. Anne told me not to, and I handed it off to Kenneth. Kenneth has rewritten SNUPEA without an apparent plan or talking to the client."

The ever helpful Jeff shared his sage wisdom. "You know John Wooden, the famous basketball coach, once said *if you don't have time to do it right, when do you have time to do it over?*"

Todd was smiling. He folded his hands as he asked, "And with the lack of policy in this regard, what do you consider *right*?"

"I said already," I replied, "I wanted to do it by the book."

"That's my point!" Todd said. "There is no book. You can't hold anyone to a standard if there isn't one."

"Good point," I replied. *Why write a book that no one will read?* I thought.

We spent the rest of the meeting in an academic discussion of how badly we needed a standard for in-house software development, and how bizarre it was to try convincing Alan Chang to be the champion of our best practice.

In the end it didn't matter. The gas marketing group forgot about the security issue in GASMAN, and the competitive intelligence team was so annoyed with the lack of progress they trash-canned SNUPEA and bought an off-the-shelf solution. Kenneth was unaffected by either situation; he just moved on to other projects. One big lesson I learned: the perception of progress is more important than actual progress.