# People-less Requirements and Analysis

**The story goes** that that a general building contractor was looking to cut expenses, and noticed that he had two bricklayers working on the same project. He decided that he could get rid of one, and still finish the project on time. He decided to ask each of the two bricklayers what they though of their jobs.

He asked the first bricklayer "What do you do every day?" The first bricklayer replied "Every morning, I can't wait to get up and come into work, inspired by what I will get to accomplish that day. I prepare my bricks and mortar, and work on raising a cathedral to the sky. My humble bricklaying will help finish this work of art, and the glory and majesty that the cathedral presents will be due, in some small measure, to the quality of my work!" The general contractor, moved beyond words, wiped a tear from his eye, and sought out the second bricklayer.

He asked the second bricklayer the same question - "What do you do every day?" The second bricklayer had a totally different attitude. The bricklayer replied "I come in exactly at 8, no earlier. I mix my mortar, and – except for two breaks and lunch, I pile one stupid brick on top of another. I can't wait for the 5 p.m. bell, so I can clean my trowel and mortar bucket and go home."

The general contractor realized the choice was obvious. Without hesitation, he quickly decided to fire the first bricklayer. You see, the two bricklayers were supposed to be building a small utility shed, not a cathedral.

Back in 1976, I was a young applications programmer working at Offutt AFB. Part of my job involved supporting the programs for handling collection and analysis of satellite data. A Lt. Col, who was one of the more experienced analysts, asked me to write a special program for him to help reduce some data. I don't think the term "data analytics" existed yet – but that was what we were accomplishing. To expedite the program – I wanted to mix the data into a common file and store it on a tape drive (1975. Honeywell 6800. 96K of main memory. 4 tape drives. Card input. What more could you ask for?)

Looking for a way to distinguish one set of data from another, I realized that I would need a record separator to help me analyze the data. I asked the user if "special characters" were part of the input data – and was informed that no special characters were ever used.

Armed with this data, and a copy of Knuth's "The Art of Computer Programming: Volume 3 Sorting and Searching," I started writing code. Because the actual data was classified, I did not have access to the actual input data yet – an unclassified system was used for development and testing. Given a schema (a description of the physical format of the eventual input file), I created a series of dummy records to test my program. In short time, I had a working prototype. Several days of testing confirmed my obvious skill and both a designed and developer. It was efficient, concise, and gave accurate results. My program was ready for real data.

The trial run of my masterpiece was scheduled during the active database downtime, or what we called "night processing." During the day, the analysts needed the database "live", so background processing that modified or manipulated the data ran every night. My program was scheduled for 2 a.m. At 2:01 a.m., I was woken up from a sound sleep to hear an operator tell me that my program has crashed, and in fact had crashed immediately upon starting execution. Not much else he could tell me (remember that classified part?) so I got dressed and headed into the computer room. At about 3 a.m., I was examining the 96K core dump to find the status of registers, files, and program counters (remember the Honeywell 6800? 96K of an octal dump.)

It took quite a while to decipher the dump, but I eventually discovered that the first character of the first input file was a "!". As a matter of fact, the entire classified input file was littered with "!", "#", "#", and every other special character you could imagine.

By this time, it was 6 a.m. – so I just hung around for the Lt. Col to show up. When he finally arrived at his desk, I showed him the input file and the program dump, and reminded him that he was told me that there were no special characters in the input file.

His response? "Exclamation marks? Those aren't special characters. We use them all the time!"

And the moral of the story is that I was a young and inexperienced programmer, who should have known to examine the input files themselves, rather than just a schema of the file.

Or maybe the moral is that users and developers (and analysts and testers and maintainers) all speak a different language – and the same word carries different connotations and meanings for each person.

The hardest part of building large software? Communications. Talking to all the users, and fathering their requirements. Determining what is a "requirement" and what is just a "that would be nice to have, but we could live without it." Determining from the user how to test each requirement. Then explaining to the users how to correctly run the system, including how to handle occasional problems, shortcomings, and failures.

Come to think of it – compared to working with lots of people, coding is probably the easy part.

**David A. Cook**
**Professor of Computer Science**
**Stephen F. Austin State University**
**cookda@sfasu.edu**