Did I Say "Koala Tea?"

Question: What do you call a formal afternoon reception or social gathering of furry, bear-like arboreal Australian marsupials, where they drink the boiled glossy leaves of an Asian evergreen shrub?

Answer: A "koala tea." Oh, wait – the issue this month was about quality, not koala tea. That's my mistake. I guess I really didn't know the correct definition of quality, which is "trait, characteristic or property, degree or grade of excellence[1]." But then, perhaps my first definition is just about as good as the second. It depends on the user's needs.

As usual with my columns, let me start with a story. Back in 1974, I was a young airman stationed at wonderful, beautiful Offutt Air Force Base. (Once you get on it, you can't get Offutt – the Air Force base with its own cemetery!) As an applications programmer at SAC (remember the Strategic Air Command?), a user asked me to sort a file of data for him. Being young and eager, I immediately set forth to write the ultimate sort routine to end all sort routines.

I had recently completed an undergraduate course in sorting and searching techniques, and I recalled a particularly cool sort from class (something like a combination polyphase-cascade-merge sort). It required several temporary files, but the machine we were using at the time was pretty limited. Does anyone remember the World Wide Military Command and Control System[2] (WWMCCS) Honeywell H6000? It had 96K of RAM. Not much memory space at all. However, it did have multiple tape drives, so I wrote a program that sorted using four tapes.

Of course with only five tape drives for the entire system, running my job took a bit of scheduling. In fact, it usually took a full day to actually run the job once I had submitted it. After a few weeks of development and testing, I was ready to unveil the program to my long-suffering user.

After explaining that there was a one-day delay between program submission and the results, I saw the look of dismay on his face. Come to find out, the data he wanted sorted consisted of exactly eight 10-character strings. A six-line bubble sort routine would (and eventually did) sort the data online in less than a second. The lesson I learned: Find out exactly what the user needs before writing the program. I had fallen into the trap of letting my biases and experience influence me into giving the user what I thought he needed, rather that giving him what he really wanted.

Well, I'm a bit older and wiser now (as my friends would point out, a lot older, maybe a little wiser) and frequently find myself teaching this lesson to others. It shouldn't be a secret; lots of people know about it. In fact, Simon and Garfunkel must have been software engineers. Back in the '70s, they produced their "Bridge Over Troubled Waters" album. If my memory serves me, on side two song No. 11 was titled "Keep the Customer Satisfied"(words and music by Paul Simon, 1970). Great song. Great message. "Just trying to keep my customers satisfied, satisfied."

You want a definition of quality? OK: Keep the customer satisfied. Do you know what customers want? Do you know what they really need? Are they part of the decision-making process when it comes to making trade-off decisions in the architecture and design? Are they part of the decision process when deciding what is part of a release, and what gets put off until the next release?

You see, you are not really ready to discuss quality until you know the users' needs. If their overwhelming requirement is accuracy, then speed might not impress them. If their overwhelming requirement is consistent uptime, then no matter how accurate and fast the system is, if it crashes frequently, they are not impressed. If their need is for a reliable system, and you forget to implement a quick-and-easy-to-use backup and recovery system, then they won't be impressed. Remember – the latest and greatest languages don't impress them if the resulting system is slow, unreliable, and buggy. A reliable Fortran program might just impress them more than a spiffy (but buggy) program written in Java Plus Plus Sharp (or whatever the language of the month is).

You want quality? Talk to your users. Find out what they need (and what they want). Involve them in trade-off decisions. Write the programs that will make them satisfied. Save the polyphase-cascade-merge sort for another time.

Oh – and yes – I do know that getting the real requirements out of some users is only slightly harder than communicating with the dead. I'll save that discussion for another column!

– David A. Cook
Software Technology Support Center/
Shim Enterprise, Inc.

1 All definitions are from Microsoft Encarta Encyclopedia, 1999 edition.

2. If you remember WWMCCS, then check out "Defense Department Classic Becomes Object of History" at <www.af.mil/news/Jul1997/n19970717_970866.html>.