## Column: Editor's Gossip Column

A Little Light Reading

Whil Hentzen

A number of years ago at a Chicago FUDG (Fox Users/Developers Group), someone in the crowd asked the speaker where he could get "the ultimate library" of functions and utilities, so that he didn't have to write them himself. Wouldn't it have been nice if all the stuff you've written over the years had been available sometime between the first and second app you wrote?

OK, we all know that's not going to happen. But that same need can pop up in a number of places. For instance, an activity my wife and I undertook about ten years ago was to see all of the movies that won a major Oscar – Best Picture, Best Actor/Actress, and Best Director. We didn't feel the need to buy every movie on the list, however, since there were some that we really had to steel ourselves to watch even once. Yet, it was nice to have a ready-made list to use as a checklist.

Another need that many people have that of a "well-stocked library." This would include all of the classics, all of the prize-winning novels over the years, and so on. Realizing this need, select bookstores can actually sell you a complete library, with leather bindings and handsome engraved covers, organized so that your den can look like that of one of those lawyers on TV who stands in front of a bookcase in an attempt to appear learned and trustedworthy.

I heard once that the average software developer owned approximately 0.9 books about his/her profession. Part of that is because the average software developer doesn't really care – they look at their profession as a job, not as a religion. But another reason is that most developer's don't know what they should be reading.

I've been collecting books at the rate of about two a month for a number of years, and have started to accumulate a reasonable library. I thought I'd share some of my favorites, in the hopes that if you're trying to build up or fill in your own personal library, that this would be a helpful starting point. And if you've got your own favorites, please forward them along, and I'll list an addendum in a future column.

Note that these are in no particular order, and certainly are not intended to imply comprehensive coverage. These are just a start:

The five (so far) books in the Microsoft Press series on general software development: Code Complete (Steve McConnell), Writing Solid Code (Steve Maguire), Dynamics of Software Development (Jim McCarthy), Debugging the Development Process (Steve Maguire) and Rapid Development (Steve McConnell). These should be required reading for every software developer, and their bosses. The first two are written with a fair amount of C code examples, but it's OK, because the examples are simple enough for anyone to understand, and serve really only there to illustrate a point made in the text.

I've not met a developer who's been able to read a chapter out of any of these books and say with a straight face, "I already knew all that." You can't help but pick excellent tips out of each of these just by flipping open to a random page and surfing.

A few classics that everyone should be acquainted with include Peopleware by Tom Demarco and Tim Lister (Dorset House) and the Mythical Man-Month by Fred Brooks (Addison Wesley). The first outlines a strategy for creating an environment conducive to professional software development. How do you expect a $60,000/year developer to be able to muster the concentration necessary to visualize and assemble a complex piece of software when the phone rings every four minutes and Joe Bob down the hall keeps coming by to chat about football and "By the way, can you fix my broken printer?"

Brooks discusses the truism that "Adding more programmers to a late project simply makes it later." While you and I know this, it's often difficult to convince management of this. Brooks provides the ammunition.

Constantine on Peopleware by Larry Constantine (Yourden Press) builds on Demarco and Lister's tome. The title irritates slightly if you haven't heard of Constantine before, and seeing his picture on the back prompts you to roll your eyes, muttering "Oh – pul-lease!" But Constantine is to software development what Da Vinci was to the Renaissance. This book is a collection of essays that bounce from Invory Tower Theory to dirt under your fingernails and back, and you can't but help look at software development differently after you finish.

Now that I've tossed Ed Yourden's name out, let's stay on track. His landmark text, The Decline and Fall of the American Programmer, was a sentencing of the American software industry a number of years back, and one you should be familiar with. The sequel, The Rise and Resurrection of the American Programmer, gave the developer a reprieve, but I'm not so sure I agree with his conclusions in the second one. Of course, your mileage may vary.

And his latest, Deathmarch, should be part of every proposal you place in front of a customer. It describes those projects that we've all been in: "What's your estimate for this project?" "If everything goes our way, nine months." "That sounds great, but my boss told me we need it in seven. Can you do it?" "Well, I think we might be able to." "Great…" [Two days later.] "The board met and think your proposal for the project is great – and they've already approved the funding for the new machines you requested. But just one small thing – they need it in five and a half months. Think you can shave a bit more time off?" "Well, I guess we'll just have to. I'm sure that the fact that we're using a brand new tool won't slow us down too much." Ten months later, the project still isn't finished, and you know the rest of the story.

Donald Norman's The Design of Everyday Things (Bantam Doubleday Dell), Bruce Togazzini's (*\\\ spelling?) TOG on Interface (*\\\ TK) and Alan Cooper's About Face – The Essentials of User Interface Design (*\\\TK) are a trio of truly insightful books about the way things work – and the way they should work. I've not been able to open a door since reading Norman without considering whether or not the door handle was appropriate for the type of entrance. TOG was the interface evangleist at Apple during the Macintosh design era. And he writes as well as he designs. 'Nuff said. Cooper's thesis: "No matter how good your interface is, less of it would be an improvement." He lectures, occasionally heavy-handedly, on user interfaces – you can't help but look at your UI design in a new light afterward.

A finally, a few for fun:

Showstopper! (Zachary Pascal, Free Press Inc.) has been mentioned in this column before – it's the saga of the development of Windows NT. You can't help but read this, mumbling to yourself, "Been there, done that." Soul of a New Machine (Tracy Kidder, Avon) traces the development of a new Data General minicomputer that went up against DEC's VAX. Don't dismiss this one with a snort, saying, "Hardware! Who needs it?" The story is compelling and funny (and, poignant, but I don't want to say too much), and paints a picture of the people in our industry in a way few people have before or since. Kidder won a Pulitzer for the work.

And, of course, if you haven't seen it, Robert X. Cringely's Accidental Empires: How The Boys of Silicon Valley Make Their Millions, Battle Foreign Competition, and Still Can't Get a Date (HarperBusiness), is the classic tale of our industry. The last couple of chapters attempt to guess the future, and they look particularly ridiculous in hindsight, but the rest of the story is a fascinating chronology.

Every one of these is an easy read – not one of those books with a thousand equations and a ponderous style that makes you wonder who the author is trying to impress. And I've found that purchasing these has been worthwhile – because I go back to them for re-reads or for reference over and over. They're each highly recommended.