Editor: Diomidis Spinellis ■ Athens University of Economics and Business ■ dds@aueb.gr

On Paper

Diomidis Spinellis

A box of crayons and a big sheet of paper provide a more expressive medium for kids than computerized paint programs. —Clifford Stoll

This column came to life as I was trying to devise an algorithm for analyzing initializers for C arrays and structures. At the time I was using the CScout refactoring browser to look for possible differences between closed and open source code. I had already processed the Linux, FreeBSD, and Windows research kernel source code, and only the OpenSolaris kernel remained. Unlike the other three code bases, Sun's code didn't appear to use any exotic compiler extensions, so CScout uncomplainingly devoured one file after the next. Then, after approximately six hours of processing and 80 percent along the way, it reported a syntax error.

Most errors I encounter when processing C code with CScout are easy to handle. I add a macro definition to simulate a compiler built-in function, I fix a corner case in my code, or I add a grammar rule to adjust for a compiler extension. This time it was different. Horrified, I realized that my implementation for handling C's initializers was far from what was actually needed. The requirement to fully evaluate compile-time constants, which I had hid under the carpet 'til then, was the least of my problems. Two days and 550 lines of code later, I found myself struggling with an algorithm to drill down, move along, and climb up a stack of data type stacks matching initialized elements to their initializers. And I was doing this on a sheet of paper.

China power

Why is it that every time I struggle with a tough problem, I turn away from my sophisticated software tools and grab a plain sheet of paper? What's the secret behind the power of this millennia-old Chinese technology? Paper turns out to have many significant advantages.

Its usability makes a computer look embarrassingly clunky. Paper is completely noiseless, it doesn't consume any standby power, it won't crash, it doesn't glare, it doesn't catch viruses, it won't heat the room, and it won't give you a repetitive-stress injury. The resolution with which I can draw on paper with my mechanical pencil is lower than that of my computer's screen. However, on an average day I can address this difference by spreading out about six sheets of paper on my desk.

Its versatility is unmatched. On the same sheet of paper, I can use an arbitrary number of fonts and symbols, even symbols I invent on the spot for my own purposes. I can highlight, I can cross out, I can underline. I can also effortlessly mix text and graphics, annotating my pseudocode with lines and arrows or commenting my diagrams with code excerpts. Try that on your code or diagram editor.

I can easily start a new design from scratch: I just toss the old design away in the round filing bin (also known as a wastebasket) and pull out a fresh sheet of paper. The undo operation

0740-7459/07/$25.00 © 2007 IEEE

(with multiple levels of undo supported) is also trivial, provided I haven't emptied the bin into the recycling dumpster. A sheet of paper also lets me save and restore my state of mind. I can easily file the paper with the rest of the project documentation and then, days later, instantly recover a snapshot of the design by pulling the paper out of the file. On a computer screen, I'd have to spend significant time restoring all editor and graph-drawing windows to the state in which I left them.

Another great advantage is that paper supports multiple abstraction levels. On my computer, the code, a class diagram, and a deployment diagram will live in different files, windows, or even tools. Yet I can combine them on a sheet of paper as naturally as they appear in my mind.

And this isn't the only way paper fits better with my mind's processes. I can scribble on paper a lot faster than I can write or draw on the screen. This means that it's a lot easier to follow my train of thought with handwritten notes than by typing. Once I begin to type, I slow down, and my mind stumbles. If I jot that wonderful idea on paper I have it secured, and I can then spend as much time as I need messing with the C++ STL library to realize it in code.

Missing features are … features

Amazingly, some of the reasons I'm productive on paper are actually due to features that a piece of paper lacks. A piece of paper won't check my syntax while I'm writing code, so I can concentrate on the ideas behind the code rather than on the language's particulars. Also, it won't underline spelling errors, letting me finish my sentences without interruptions.

Furthermore, paper typically lacks a macro command processor, a regular expression search and replace function, and online help. Again, this means more time for thinking about the problem rather than the particulars of writing the code or satisfying a specific API's requirements. I often wish I could look up things on Google or Wikipedia when writing on paper, but this "problem" turns out to also be a terrific efficiency booster (see figure 1).

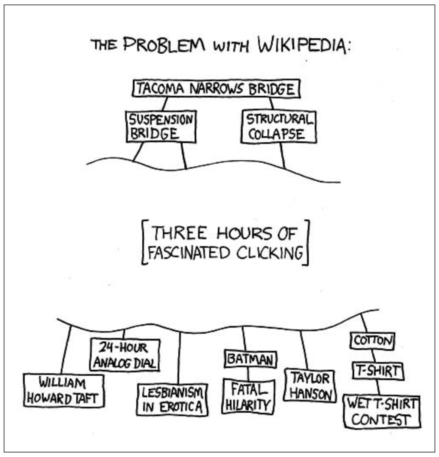

Figure 1. Lack of Wikipedia access considered a timesaver (from http://xkcd.com).

Thankfully, paper is also severely challenged in the area of synchronous and asynchronous communications. No mail pop-ups, chat client interruptions, incoming RSS items, or friends joining a voice-over-IP client. Although getting a notification when a colleague across the globe enters Skype is fascinating, the cost of this interruption is appalling. Years ago, Tom DeMarco and Timothy Lister explained to us in *Peopleware: Productive Projects and Teams* (Dorset House, 1987) that we must enter a mental state called flow to be productive in design, engineering, and other creative endeavors. It can take us more than 15 minutes to enter into such a state and only a trivial interruption ("your imap server has 1 new message") to exit from it.

What's the moral behind this rant? In the short term, make sure you always keep a stack of blank paper within arm's reach. Use those sheets liberally. In the long term, I'm sure that scientists will give our computers more of paper's desirable attributes. Who knows, they might even succeed in disabling those irritating pop-up messages.

Diomidis Spinellis is an associate professor in the Department of Management Science and Technology at the Athens University of Economics and Business and the author of *Code Quality: The Open Source Perspective* (Addison-Wesley, 2006). Contact him at dds@aueb.gr.