

A cult I almost joined

Eric Blair

14 May 2005

Although I had previously commented on how minimalist I want my life to be, I've found my limit—the point at which minimalism becomes madness, and that is lisp.

Lisp (list processing) is a programming language and a cult. Now, I know you're thinking that you've seen many a programmer's favorite whatever become cults, but lisp *actually* shows signs of culthood. Ms TNH of Nueva York, NY has this simple definition of a cult:

If, on appropriate occasions, the members tell, enjoy, trade, and/or devise transgressively funny jokes about their denomination, it's a church.

If such jokes reliably meet with stifling social disapproval, it's a cult.

So, I can't find any self-poking jokes about lisp. Self-deprecating Unix jokes, no problem. Jokes about frigging text editors, hilarious. But lisp has nothing but theological proselytizing. [The "My other car is a cdr" bumper-sticker doesn't count—it's not self-deprecating or transgressively funny, just an in-joke.]

See, we geeks never played sports, and even if we did, nobody would pick us for the team and we'd wind up on the shirts team just because nobody would want to see us on the skins team. And so, we pick teams by programming language. Which language you use doesn't just determine what kind of syntax you're going to stare at for the rest of the project, but also who you'll be associating with and who you're going to be making fun of. One could make Sapir-Whorfesque arguments about how different types of thinking are attracted to different languages, but those only go so far; like a sports team, it's a basically pointless division of a homogenous group of people which people wind up caring far too much about. [Of course, this essay is making fun of the lisp team. What can I say.]

Exhibit B: a guy who explains that lisp is a language for smart people, while everything else is for code monkeys. When I was on the fence between putting too much time into learning lisp and too much time learning python, this tipped me over the fence into python land. Do I really want to be on this guy's team? I'm up to number seven of this delightful computing tutorial right now, and glad I went with python. Have been judiciously avoiding articles entitled "Python vs. [anything]" since.

The next thing that stands out in how people praise lisp is the frequent mention of the fact that Yahoo! Stores run on lisp. When I was in Morocco, several different people told me that Neil Armstrong converted to Islam after seeing the Earth, and that Cat Stevens converted. At first, this was OK, but soon, after the third person or so mentioned this pair, I got to wondering, *maybe they're the only ones*. Islam seems to be doing OK for itself, but I know that lisp is not very commonly used, and it may be true that Yahoo! Stores is indeed one of the only global-scale successes of lisp. Oh, and Mathematica implements a lisp-like syntax, but sort of hides it from the user.

Open source proselytizers, conversely, have it easy: two-thirds of the web (69%) runs the open source Apache, which was written in C. MySQL is taking off, OpenOffice is used by various municipal governments, et friggig cetera. Whatever you're reading this with, be it Firefox or Internet Explorer or Safari, was written in a variant of C.

Which brings us to the fundamental question driving all discussion about lisp: if it's so great, why isn't anybody using it? Here's a story where Lisp follows the MIT/Stanford approach (good), while the rest of computing trundles along with the New Jersey approach (not good enough). [A great deal of what we use today was developed at Bell Labs, in NJ.] It's an amazing read—the archetype of the cult essay—because it explains the overall success of the dominant paradigm as being entirely due to its evilness. This is on par with an essay on how McDonald's sells so much because it sells fattening crap. Partly true, but it also packages blobs of animal fat in a cheap and convenient package, and cheap and convenient is good even if factory animal slaughter isn't. If we rate cult essayists by their ability to maintain the fiction that there are absolutely no good goals in the world but their own, our lisp author gets a gold star. [The rest of the article gives some pretty good self-critique of how to save lisp, and seems to recover from the juvenalia of the subessay I'm bitching about here.] Indeed, from what I can gather, lisp is a good language. Its key feature over the New Jersey languages (C, C++, Java, et cetera) is that it can execute its own output. It's easy to write a program that can modify its own code, or that can write new programs that improve upon itself. You don't have to be into artificial intelligence or computer learning to appreciate the coolness of a program-writing program.

(Don't like lisp's syntax (and ('nobody) does))? Then the first half of your program can be a set of instructions giving new syntax, and the second half can be the actual program, in your new, cozier syntax. This is also kind of cool.

As well as being cool, it is the death knell of the language for serious computing.

For the non-programmers among you, here's the executive summary of how to program: 1. Surf the Net for packages by other people that do what you need to get done. 2. Repeat step one until you've gathered everything you need. 3. Write a program that calls the modules you've collected. This is a great method, because step three is potentially very brief, provided there are enough modules out there. Here is Google's directory page for python; notice that thing in the corner where it says "Modules (249)"; that's a low estimate.

Other comparable languages like Perl or Delphi have a comparable or greater abundance.

And lisp modules? Well, every part of the code can modify every other part. You can write a whole new grammar if you're so inclined. None of this helps with the goal of downloading a random package and easily calling whatever you want from it with minimal hassle. There's just a fundamental conflict between easy modularity on the one hand, which requires lots of restrictions and boring standard forms, and fluid modifiability on the other.

So with most proselytizing. If only you saw it our way, the proselytizers insist, you'd be a happier person. But we wouldn't be, because our goals fundamentally differ. I want efficient albeit boring; lisp programmers want fluid and clean; sales figures show that most people want a dancing paperclip. The more typical proselytizers have the same problem: they want inner peace, while most people just want to get laid.

Exacerbating the failure of communication, everybody lies about what they want: in the working-with-machines context they say they want to work quickly, but their true goals are to make the work go away as quickly as possible. Achieving the first involves finding good tools and reading their manuals while the second typically involves playing some music, not trying anything too hard, and generally tuning the tools out and thinking about sex. And we all want inner peace—provided it doesn't hinder our chances of getting laid.

And this is why lisp will always remain a cult. It provides cleverness and beauty, which we all want on the surface, but it's fundamentally at odds with laziness and transparency. When we're pressed, the surface desire for beauty and elegance evaporate and reveal our base, underlying desire for the easy and fattening.