

In the land of invented languages

Eric Blair

22 June 2009

OK, so this is vaguely a discussion of *In the Land of Invented Languages: Esperanto Rock Stars, Klingon Poets, Loglan Lovers, and the Mad Dreamers Who Tried to Build A Perfect Language* [Okrent, 2009]

To start, for full disclosure, let me tell you why I picked this book up: because Arika and her husband are pals of mine. I'm not going to give you a full review—the NYT¹ and Salon² have already reviewed it, so she doesn't need me. [The Salon review is lazy and represents everything I hate peer review, but that's another rant.]

That said, I'll be discussing Arika's world in the context I know best: coding languages. Arika's book is about human languages, and the people who decided that it's high time for a new language around which a world community would (hopefully) be built. That's never the case with a programming language. Nobody expects to raise their kids in Python.

That said, there's still a lot in common between the new human language and the new coding language.

Personality I'm interested in the coding language thing mostly because I'm sorta forced to. People are darn vehement about this subject, in a scary kind of way, given that we're talking not about abortion or capital punishment but whether to end lines with a semicolon. It's the first point in any discussion of my book: what coding language did I choose? A lot of people never get past that. Others send me letters entreating me to halt my work and redo everything in their favorite language. This happened even after the book was published.

But where does all this vehemence come from?

As Arika tells it, the story of invented languages is a story of personalities: people who (1) felt the need to devise a new way of speaking, (2) did the work of developing a whole new system, and (3) did the harder work of developing a community to use and support that language. All three of these items are tall orders, and require a certain sort of persistence/megalomania.

The stories of the crazy folk who persisted against all odds is what makes Arika's book ready for the popular press. There's certainly conflict: authors whose life's work is torn apart and dismissed, authors expounding on the conspiracy theories holding them down, lawsuits over ownership and control, and yelly people who just yell a lot.

¹<http://www.nytimes.com/2009/05/24/books/review/Blount-t.html>

²http://www.salon.com/books/review/2009/06/03/invented_languages/index.html

Bringing it back to programming, it's just about all there. The personalities are very evident, and there's a name attached to a great many languages: Perl = Larry Wall; Python = Guido van Rossum; C = Kernighan & Ritchie; ruby = Yukihiro "Matz" Matsumoto; C++ = Bjarne Stroustrup; and so on. In the case of C, the authors let go pretty quickly, and we now refer to it as ISO C99, not Brian and Dennis's C. For the others, this is less the case, and the authors are a part of the language. When you're using Perl, you've got Larry Wall over your shoulder, whispering in your ear.

As Arika repeatedly shows, anybody who will devote their life to such an endeavor has to have quite the abundance of self-confidence; or as explained by Larry Wall, who describes himself as "the Perl god", "All language designers are arrogant. Goes with the territory... :-)" ^[wiki³]

So what motivates these folks?

No, we won't achieve world peace by all speaking the same language, but I find it indisputable that language is at some level a unifier and a divider. I've in entry #127 before: to choose a language is to choose a team. So to develop a new language is to (attempt to) form a new team.

The smiley-faced way to say it is that the author wants to build a community. Here's Larry Wall again: "I want to see people using Perl to glue things together creatively, not just technically but also socially." In the world of invented human languages, such thinking goes with the idealists, like the author of Esperanto, Ludwik Zamenhoff.

The cynical way to put it is that the author wants to lead a community, to have followers. These are the ones who wanted more control over things, like the author of Blissymbolics, Charles Bliss, who threatened lawsuits against those who would publicize his language with the wrong line thicknesses, or the author of Loglan, James Brown, who took his followers to court for taking his suggestions and running with them.

Categorizing the authors of the computing languages between community builders and control freaks is left as an exercise for the reader.

Done right, the community actually does form. Arika shows this nicely as she traces how each new language develops its own culture, which repels some people and attracts others. Over and over again, people develop an allegedly culture-free universal language, and then a culture starts to develop around the language, and then reshapes the language to its style.

The grammar and vocabulary of a language inevitably express a point of view, and then those who share that POV gather 'round. In that way, I suppose a new language is a lot like a pop band.

The schisms Some pop bands are entirely derivative, and some say something at least a little different. It's hard to find many that are truly innovative—they still all use rhythms and melodies (except the ones who don't, who all sound alike).

Languages are so cheap because it's always easy to base your new language on an old. Nietzsche described the übermensch—the once-in-a-generation rarity—as one who writes a New Word; the rest of us just recycle and recombine.

³http://en.wikiquote.org/wiki/Larry_wall



Figure 1: Some typical modern C users (or, what happens when you search Flickr for *hipster*).

As a corollary of the fact that all new languages are in some way a derivative of the old, it's hard to keep a language from recombining and digressing. Arika describes the many, many schisms that any successful language will eventually suffer: Esperanto has Interlingua, Ido, Glosa, Globaqa, Novial, Hom-Idyomo [Arika, p 98]. C has C++, C--, C#, C ω , C-smile, D, Objective-C, and lots of others that at least had the sense to not just name their language some variant of *C-increment*. [This site has a list of about 1,200 languages for ya.⁴]

Remember, they're all a set of trade-offs: you can have simple or you can have comprehensive; you can cut out all the redundancy and requirements on the human, or you can fill it with self-checking and human-friendly type agreement rules. But there's no chorus-of-angels true answer that we'll one day discover, and we're instead left wandering the desert using whatever seems useful at the moment.

Hey, I have my complaints about C's syntax, which I already listed in one of my more boring in entry #225. But I chose to stick with the standards rather than talk the world into accepting my personal ideal. Others chose to go the other way, and after the bickering and incompatibilities, that's how progress is made.

So maybe it's a question of improving on things and really getting the syntax right, but maybe it's just a question of personalities. In such a world, the problem with C is not that its variadic function handling isn't so hot, but that we picture the people using it as being cutting-edge in 1979.

How does one update an imagined community? You don't. It's easier to just generate a new one.

⁴<http://www.99-bottles-of-beer.net/>

References

Arika Okrent. *Exploratory Data Analysis*. Spiegel & Grau, 2009.