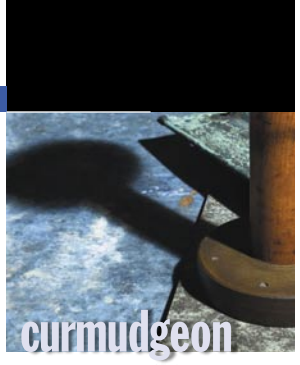

Linguae Francae

Stan Kelly-Bootle

Is programming language a misnomer?

Many linguists are still busy trying to reconstruct the single ur-language presumed to have evolved over untold millennia into the thousands of human tongues—alive and dead, spoken and written—that have since been catalogued and analyzed.[1] The amazing variety and complexity of known languages and dialects seems, at first parse, to gainsay such a singular seed.

Yet many find a deep, mythic confirmation in the Tower of Babel story: "Now the whole world used the same language and the same words" (Genesis 11:1). And, of great computer-theoretic significance, note how Jahweh upsets this Esperantic paradise and launches his logomachic crusade: with a raging, unstructured "GOTO!" followed by "Let us go down and confound their language, that they may not understand one another's speech" (Genesis 11:7). Others, however, especially Dijkstra, see this GOTO as the instruction from hell.[2]

Speaking of "Structure" (with a reverent capital S), we move to a major milestone in linguistic theory, and one that seems to argue against the Babel account. I refer to Ferdinand de Saussure's *Cours de linguistique génerale,* published posthumously in 1916. Irony strikes again. The text, considered to be the genesis of structural linguistics and semiotics, was cobbled together from fragmented lecture notes taken by his students Bally and Sechehaye.[3] To cut one of mankind's most convoluted books short, Saussure's famous gist is the "essential arbitrariness" of the correlation between *signifiants* and *signifiés*—that is, between signifiers (words, for instance) and the objects they signify. Thus, there's nothing particularly tree-like about the word *tree*, or even its French equivalent *arbre*. You may raise the onomatopoeic objection, to which I say: French cats don't "purr," they go "ronron"; and French ducks "ne quackent pas," they prefer to "faire coin-coin."

Saussure's thesis, now universally recognized (with inevitable variations and refinements), can be invoked to cast doubts about whether there was ever one primeval lexicon from which our current "Babel" emerged. Many languages do fall into plausible families and subfamilies. Indeed, the birth of comparative linguistics can be traced to the success in establishing a taxonomy for the IE (Indo-European) super-family. Widespread though IE members are (with English, or rather Englishes, established as the dominants for international discourse), there are some 4,000 more-or-less extant non-IE languages.

Grouping these into larger and larger families is proving to be a rather suspect exercise, since many "creative" (as in "creative accounting") guesses are involved in constructing proto-languages with little or no written evidence. There's no end of wishful leaps based on pure coincidences in shape and sound. And even the same word root found in daughter languages does not imply that that word existed in the proto-parent. As Jared Diamond points out, "Archaeologists [unfairly?] skeptical of linguists' attempt to reconstruct mother tongues love to cite words like 'Coca Cola,' shared among many modern European languages."[4]

Of more immediate concern to my readers is the interaction between NL (natural language) and so-called programming languages. I say "so-called" because many linguists consider programming "languages" to be the most egregious misnomer since the Big Bang (which I parochially date to 1949 when the Cambridge EDSAC I passed the perfect benchmark by listing hundreds of random numbers!). Yet, whether we like it or not, we are stuck with the word *language* in a fresh context: sequences of symbols with artificially "frozen" syntaxes aimed at the precise control of machines. This usage is not entirely new. Galileo hinted at the Platonic algorithms that run our cosmos: "No one will be able to read the great book of the universe if he does not understand its *language*, which is that of mathematics."

We face the familiar, false dichotomy: vague, ambiguous NL versus exact, unambiguous mathematical symbols and axioms. David Lorge Parnas, explaining his techniques for module specification in 1972 when software engineering was emerging with giddy enthusiasm, stressed, "It should be clear that while we cannot afford to use NL specifications, we cannot manage to do without NL *explanations*. Any formal structure is a hollow shell to

Continued on page 78

Continued from page 80
most of us without a description of its intended interpretation."[5] And here, I would like to add the need to spell out the *motivation*. (There's the familiar Euler-Gauss stylistic gulf in mathematical exposition: Euler was lavish in exposing the motives and intermediate arguments in his proofs. Gauss preferred a blunt sequence of steps as if to say, "Follow *that*, dummies!")

Parnas goes in the other direction, from NL to formal, when choosing names for identifiers: "...if one makes use of names with a high mnemonic value, both reader and writer tend to become sloppy and use the intended interpretation implied by the mnemonic name to answer questions which should be answered by the formal statements."[6]

In the early, "unExtended" Basic days, one had little opportunity for concocting memorable, self-explanatory identifier names. `A$` was certainly a string, possibly the employee's name, and her gross pay was the integer `A` (in cents)! Later on came the Hungarian invasion, reminding us that `szTring` both looks and sounds like a Magyar string!

Parnas therefore warns against `gross_pay`, `deductions`, and `net_pay`. But, what if the code reads

```
net_pay := gross_pay + deductions;
```

Whom to believe?

On the other hand, editor Edward Yourdon himself confesses that his "eyes often glaze over" reading page after page of Parnas's Spartan modules. The latter upholds another Parnas structured stricture: "Avoid redundancy. Specifications should tell programmers exactly what they need—no more and no less."

I MEAN TO SAY...

Since we have little choice, the challenge is to make our NL statements as unambiguous as possible...whatever *that* means.

There's much more to "disambiguation" than getting your commas and apostrophes in the "correct" positions. Certainly, as the many guides to proper punctuation point out, misplaced commas and hyphens can leave some doubt as to the writer's intention. Lynn Truss, in her unexpected bestseller, *Eats, Shoots & Leaves*, drives the message home with best-selling humor, yet her examples are quite contrived and ignore the all-important question of context in everyday discourse.[7] As Hank Hanegraaff, CRI's (Christian Research Institute) Bible Answer Man, reminds us, "Text out of context is mere pretext." The punctuation that Truss addresses is really quite modern, and many of the rules that she canonizes are the result of arbitrary decisions made by typesetters and prescriptive grammarians. As Jef Raskin points out, the American rule that places full stops and commas *inside* quotation marks can be quite dangerous when the quotation marks are used to define strings in a computer language.[8]

One can look back in wild surmise to the times when written languages coped not only without punctuation symbols but indeed with no separators to indicate where words or sentences started and ended. And, especially teasing because of the major theological implications, are the earliest New Testament Greek manuscripts written in *scripta continua,* where the uncial letters run on "continuously word after word and sentence after sentence, without a break and with extremely few reading aids."[9] It's one thing to wonder (or care!) what Truss's panda is doing, yet another to confront the long string of uninterrupted Greek uppercase letters in Mark 10:40. That string can be divided into words in two different ways, each giving two completely different meanings. The two interpretations (*allois* or *all ois*) could affect the future seating arrangements in heaven for James and John, the sons of Zebedee.

Trite examples occur regularly in the wonderful world of Web URLs. A recent example finds a company, Expressions Exchange, rashly establishing a URL that could be parsed as expression-sex-change.com.

The really dangerous roots of ambiguity lurk deeper. Polysemy (multiple meanings[10]) is not always obvious from context. Indeed, writers may be less aware than their readers of the semantic spread of the words they produce. And dare I remind you that much ber-lood has been shed over those apparently unequivocal words such as *same*, *some*, *all*, *if*, and, especially, *is*.

Raskin rightly urges that programmers shun ambiguity and cultivate readability (with bonus marks for elegance) in all their writings. In the case of coding for computers, there are compilers and interpreters to "enforce" the rules, yet these rules, in turn, are based on how the compiler writers interpret [sic] the NL documents that purport to represent the intentions of the computer-language designer or standards committee. We cannot escape the daunting challenge of NL—it's really the only language we have, and with the major recursive quirk (*pace* to some Chomskian "formalists") that NLs must be described using NLs. Q

REFERENCES

1. One is tempted to misuse the term *Panglossian*, which,

although etymologically attractive [universal language], has come to mean a follower of Voltaire's character Pangloss in *Candide*, a biting satire of Leibniz's optimism: "All is for the best in the best of all possible worlds."
2. Dijkstra, E. W. 1968. GOTO statement considered harmful. *Communications of the ACM* 11(March): 147-148.
3. Harris, R. 1991. *Reading Saussure—A Critical Commentary on the "Cours de linguistique geńerale"*. La Salle, Illinois: Open Court.
4. Diamond, J. 1992. *The Third Chimpanzee—The Evolution and the Future of the Human Animal*. New York: HarperCollins.
5. Parnas, D. L. 1972. A technique for software module specification with examples. *Communications of the ACM* 15 (May). Reprinted with editorial comments in *Writings of the Revolution—Selected Readings on Software Engineering*, ed. E. Yourdon. 1982. New York: Yourdon Press.
6. See Reference 5.
7. Truss, L. 2004. *Eats, Shoots & Leaves*. New York: Gotham Books.
8. Raskin, J. 2004. For the want of a comma, the meaning was lost. *ACM Queue* 2 (May): 14-16.
9. Aland, K., and B. Aland. 1987. *The Text of the New Testament*. Translated by E. F. Rhodes. Grand Rapids: William B. Eerdmans.
10. The NL equivalent of OOPS (object-oriented programming) polymorphism. The English record for polysemicity seems to be held by the word *set*. This is quite alarming given the fundamental role of Set Theory in mathematics. The Halmos classic Naive Set Theory appeared long before Trussed hyphens became *de rigueur*.

LOVE IT, HATE IT? LET US KNOW

feedback@acmqueue.com or www.acmqueue.com/forums

STAN KELLY-BOOTLE (http://www.feniks.com/skb/; http://www.sarcheck.com), born in Liverpool, England, read pure mathematics at Cambridge in the 1950s before tackling the impurities of computer science on the pioneering EDSAC I. His many books include *The Devil's DP Dictionary* (McGraw-Hill, 1981) and *Understanding Unix* (Sybex, 1994). Under his nom-de-folk, Stan Kelly, he has enjoyed a parallel career as a singer and songwriter.