Better Writing for Software Engineers

Good writing is actually *more* valuable in a technical subject; it makes abstract or complex discussion easier to follow.

Be sure of the basics. Look up words. Don't use words you don't fully understand; don't go past them in reading, either. How are you at grammar and punctuation? Get and use a reference book or two.

Clear perceptive use of figurative language can help communication. Careless clumsy use is annoying and discourteous, and can leave readers baffled.

Use the same words to describe the same thing. If you mention an architectural model in Paragraph 1, but then call it a structural model in Paragraph 3, you will mystify the reader about how many models there are.

Digest your ideas. Work out how to put them into straightforward words and sentences. Short, simple ones are best. Don't make the reader figure out what you probably mean.

Here are some typical mistakes in recent student writing, and some suggestions.

Example 1. The software incorporates both A and B engines in aid of assisting English learning.

a) *In aid of* and *assisting* both mean *help*. Nothing appears to be gained by this redundancy. Note that both are "fancy talk" instead of a simple word. If *assist* backs off from *help*, then *in aid of assisting* backs off a second step — what can be meant by *helping to help*? Just say *help* or *aid*.

Look for the action in what you have to say. Who does what to whom?

b) To *incorporate* both *engines* is bad metaphor. *Incorporate* means *bring into the body of*; the thought of *incorporating an engine* is painful. Why not say *has*? When you use words figuratively, keep them in line with their root meanings.

c) Avoid jingles like *assisting* English *learning*. They distract the reader. This would never have been troublesome if the writer had simply said *The software helps users learn English. It has two engines, A and B.* Note that the parallel structure of these two sentences keeps the reader from taking *it* as a pointer to *English*. (See "parallelism" in a grammar book.)

Example 2. The approach would be applied to the web server to recover its architecture and to add an appropriate evolutionary capability.

a) *Its* points ambiguously to *web server* and *approach*. Keep track of pronouns.

b) *An appropriate evolutionary capability* is vague. Beware of many-syllable words, especially when the words are long because they are modifications of modifications — *proper* to *appropriate*, *evolve* to *evolution* to *evolutionary*, *able* to *capable* to *capability*. Strings of such words are worse.

Look through words to the concepts they stand for. Boil down complicated expressions.

Example 3. The performance of different server architectures is evaluated in the context of a single implementation in order to quantify the impact of a server's concurrency architecture on its performance.

a) *Context* and *impact* are bad metaphor again: *context* is a literary term, meaning *nearby words*, and *impact* means *hit*, what a car does to a telephone pole, neither of which is going on here. Even if they were proper, they're so trite from overuse they should be avoided. *Evaluated* and *quantify* are likewise trite, "fancy talk", and multi-syllable words used in place of simple ones.

b) Beware of noun chains like *concurrency architecture*. They make writing vague and hard to understand. They often indicate vague thought.

c) A few minutes' pondering — which should have been spent by the writer, not imposed upon the reader — suggests that this sentence may mean *Different servers will be tested with the same implementation to show how their architecture affects their performance*, thirty fewer syllables.