Wisdom of the Gurus: Facing the Millennium

Diversify

John Vlissides
C++ Report, November/December 1999

A common and generally unwelcome by-product of mastering a skill is the narrowing effect. I don't mean you get thinner, although focusing on something other than food does wonders for my waistline. No, I'm talking about how the single-minded pursuit of a skill affects other areas of your professional life. Mastery has an opportunity cost: You focus on something so intensely that other skills go to seed, or you never acquire them in the first place.

Expertise can become an end in itself. It has a powerful allure, especially when there isn't enough of it to go around. A skill vacuum always accompanies hot topics—the Javas, UMLs, and components of this era or that. What about C++, you ask? That's the thing about vacuums: they can be self-perpetuating. Even though C++ is not the hot property it used to be, the new ANSI Standard guarantees a dearth of C++ expertise well into the next century. Heck, over 30 years have passed since Simula gave us objects, and most people still aren't comfortable with them.

The relevance to the C++ programmer is twofold.

First, don't spend your life learning every last detail of C++—or the UML, or the methodology du jour. Don't limit yourself to pure design, pure implementation, or pure analysis. Leave extreme specialization to the medical profession; ours is too new, too dynamic to focus on any area exclusively. Yes, it's crucial to know your tools and to use them well, but don't obsess over them.

I'm sure there are people who have fun studying the C++ spec for literally seconds on end. But trying to absorb it all is just plain futile. There's no way you're going to appreciate the ins and outs of every language feature without applying them in practice—and by the way, I know not one soul who has exercised all of C++ in practice. There can be value in exploring some nook or cranny even if you never use it, I'll admit. You just have to balance this value against that of learning something relevant outside C++.

When you confront a real problem that resists customary solutions, it makes sense to go on a feature hunt. Yet even then, it's unlikely you're the first to experience the problem. Chances are that someone, somewhere has faced it before, possibly in different guise. Seek out the existing solution. Is it written up as a pattern, perchance? Consult the Pattern Almanac.[1] Browse the Portland Pattern Repository[2] or other online pattern resources. Post a query to a newsgroup or mailing list.[3] When all else fails, go to the language spec and craft a solution from first principles; until then, resist the urge to reinvent the wheel.

Second, spend the time you just saved on other areas of learning. Broaden your horizons. Delve into Java, XML, even (heaven forbid!) Visual Basic. Branch out. Learn about the special needs of software for low-power devices like PDAs and cell phones. Get crazy and study organizational management. Curl up with that long-shelved novel. Read to your kids, for pete's sake. A gear-change can do wonders for your creativity, not to mention the insights that can come from a little cross-pollination.

There *is* life beyond C++. Don't let it pass you by.

References

[1] Rising, L. *The Pattern Almanac*, Addison–Wesley, Reading, MA, 2000.
[2] http://c2.com/ppr.
[3] http://hillside.net/patterns/Lists.html.