Publisher's Note

Return of the Dreaded GoTo

"There are no facts, there is no truth, just data to be manipulated."
--- Don Henley, "In the Garden of Allah," ASCAP.

The biggest trend in computer programming isn't object-oriented development. In fact, the entire concept of structured development is about to be thrown out, with the return of the GoTo as the hottest new concept in programming. Yes, you heard it here first, the GoTo is back and badder than ever.

By now, you're thinking I'm crazy, but consider this sequence of events. We're all being swept up by the Internet craze. Every product from Microsoft, with the possible exception of Solitaire, will become Net enabled.

The hot new tool from Microsoft is VBScript. And the coalition of "everyone else against Microsoft" has anointed Java as its Lancelot to fight the Big Green Dragon from Bellevue.

These tools are exciting. The ability to create download-and-run applications to enable client/server computing across the Internet will have a huge impact. The most visible examples will be public, commercial sites, but the biggest impact will be in design of internal sites—so-called "Intranet" applications.

But, when you look at the code from any Web site, what do you see? Open any site's content in your browser, choose the option to look at the HTML, and what's there? A succession of GoTos. That's all HTML is—a bunch of GoTos, and without even the help of line numbers.

Web sites are huge agglomerations of spaghetti code. Try to map the structure of any site of significant size or content volume and you'll see what I mean. That's why you get URL errors even on sites like Motorola's or General Motors'— sites that cost half a million dollars to design.

They've removed a section, and can't find all the GoTos that called it. When software vendors ask my advice on setting up Web sites, I tell them their major expense will be maintenance because, basically, Web sites violate every design principal and, like any large software project, you pay the price during upkeep.

Next, look at samples of VBScript code. It's HTML mixed with subroutines. My first reaction was amazement at how powerful this will be. And it will. But, after four years of promoting VBA as a modern, structured language, denying its spaghetti-code heritage from GW BASIC days, VB is returning to its roots. This isn't to say that Sun's Java is any better. But the irony is worth enjoying.

Java has earned amazing press coverage, and is credited for redefining the Web—an amazing achievement for a product that was in beta and hadn't yet shipped. It's promise is small, fast, portable appletts. To do this, Java is essentially a specialized version of C++ running atop an interpreter kernel.

Wait a minute! Isn't the primary criticism of VB its use of a pseudocode interpreter? So, the premise here is that if we combine the worst of both worlds—the difficulty of object-oriented programming with its long learning curve (remember, most C++ owners actually still write C code) and the long development times of C++ with the poor performance of a p-code machine runtime DLL, then we'll have an exciting new tool. (Am I missing something?)

That this concept, so highly touted in press from *USA Today* to *ComputerWorld*, has problems is evident in Sun's plans to create a Java compiler. In other words, when you log onto a site, it will have to determine what platform you're using and chose to download OCXs in Intel architecture, PowerPC architecture, SPARC—well, you get it. So, we're back to choosing between portability and performance and the original premise gets tossed out the lowercase window.

To deal with the relative unproductivity of C++, we'll be sold tools like Borland's Latte, to put visual programming on top of C++, which itself runs on top of the p-code machine.

All of this effort just to get back to where Visual Basic starts—while still being able to say you're programming in C++—so Unix-heads will love it. Seems obtuse to me. VBScript, however, with its higher level of abstraction and p-code interpreter works as if it was designed for the Web.

All this is the price of having the whole world networked, which probably makes it worthwhile. The Web's impact is amazing. But let's recognize that this isn't all new, and it isn't all good. From a software perspective, we're reinventing the wheel and all the old problems are bound to resurface, to be solved once again. ☒

Jim Fawcette
Publisher and Editor

YESTERDAY'S PROBLEMS WILL BE RESURRECTED LIKE DRACULA AFTER THE STAKE IS REMOVED.