Object-Oriented Comes of Age

Last spring, IBM Corp. shook the programming world when it released its VisualAge object-oriented development environment for its OS/2 operating system. VisualAge represents a new generation of application development environments. It combines sophisticated development tools, object-oriented programming and the elegant object-oriented language, Smalltalk, into a unified environment designed to speed both application development and deployment.

ERIC MEOLA, THE IMAGE BANK

"Smalltalk and VisualAge are turning the market upside-down," says Paul White, vice president of marketing for The Object People, a training and consulting group in Ottawa, Ontario. VisualAge is visual programming that works, says White. On the screen, programmers can build fully featured applications simply by connecting prefabricated program components. Unlike other visual programming environments, VisualAge is rich and expressive. And, most importantly, White says, VisualAge makes it easy to get at Smalltalk, the system's own development language, for programmers with special needs not covered by VisualAge's current generation of visual objects.

New Trends Mean New Opportunities

VisualAge is now available on IBM's OS/2 operating system. Versions for SunOS and Solaris are scheduled for introduction sometime in 1995. The product is a harbinger of two important gains for custom application development in the second half of this decade: the mainstreaming of object-oriented development environments, and the market acceptance of object-oriented languages other than C++, the language invented by Bell Laboratories Inc. and popular on most UNIX platforms.

"It's been accepted for a long time that C++ isn't the best language for many applications," says White. "It was originally conceived for the telecommunications world," where the demands of real-time processing put execution speed at a premium. In the business world, an application program's runtime is often far less important than how long it takes to develop the program itself. Many programmers conversant in both Smalltalk and C++ readily admit that application development under Smalltalk can take significantly less time than under C++ because of faster development tools, better debugging facilities and richer object libraries.

IBM isn't the only company pushing Smalltalk. This spring, Quasar Knowledge Systems Inc., a five-year-old software development company in Bethesda, MD, will be releasing its Smalltalk Agents development environment. The system is a multiplatform object-oriented development environment that will allow developers to create a single application with a graphical user interface and deploy it on a variety of different computers. Smalltalk Agents now runs on the Macintosh and Power Macintosh platforms. QKS plans to ship Smalltalk Agents for Microsoft Corp. Windows, Sun Solaris, Silicon Graphics Inc. Irix and other UNIX platforms during the first half of 1995.

At SunSoft, work continues on Project Distributed Object Environment (DOE) and OpenStep, SunSoft's collaboration with NeXT Computer Inc. An alpha release of the system, which brings NeXT's object-oriented application development environment to Solaris, has been available to select NextStep developers since last October. OpenStep is based on Objective-C, a language that combines the C programming language with the Smalltalk object model, giving it, supporters say, the best of both worlds: speedy program development *and* program execution.

It's been accepted for a long time that C++

There is a growing number of new free software object-oriented systems as well. A version of Smalltalk distributed by the Free Software Foundation is slowly maturing, joining the ranks of the FSF's G++, a C++ compiler developed by Michael Tiemann, president of Cygnus Support and perhaps the most widely used implementation of C++ in the academic world. A number of Project GNU volunteers are also working on a freely available version of the OpenStep system called GNU Step, although the system is far from compilation.

Tcl/Tk is a freely available system for building applications with graphical user interfaces. The system is based on Tcl, a scripting language developed by University of California at Berkeley professor John K. Ousterhout, who created Tcl as a scripting language to control small interactive tools. Tcl runs on most UNIX computers and is freely distributed over the Internet at `ftp://ftp.cs.berkeley.edu/ucb/tcl`. You might also check out *Tcl and the Tk Toolkit*, by Professor Ousterhout, published by Addison-Wesley Publishing Co. Inc. (ISBN 0-201-63337-X).

Tcl isn't without its problems. Some programmers think the language's syntax is idiosyncratic and difficult to understand. Tcl is slow, perhaps 100 to 200 times slower than languages like Forth, and a thousand times slower than C. For this reason, versions of Tk are being developed that are based on Scheme, a Lisp-like language, and Perl, a scripting and systems administration language developed by Larry Wall. Perl itself underwent an object-oriented transformation in 1994, with the release of Perl5, a new version of Perl that adds objects and object-oriented features.

Object-Oriented Databases

Once you've made the transition to an object-oriented development environment, you'll be faced with another question: How do I store the data that my application generates?

Increasingly, programmers are turning to object-oriented databases as an answer. These systems, which have been on the market for years, reinvent the computer's database, turning it from a system that stores tables of data into a system that's fine-tuned for storing the same sorts of objects that programs use to build their applications.

One of the advantages of using an object-oriented database is that data sets created with one program are easily read by another, says William Blundon, vice president of marketing at Object Design Inc., in Burlington, MA, one of the two leading vendors of object-oriented database management systems (OODBMS). "Most of the people using object-oriented tools are using them not just to develop form-based presentation of information, but are using more complex tools because they have additional data-types that are not table-oriented, or they've gotten to the point where creating the table and forms results in so many joins in a relational database that they can't get the performance out of it," Blundon says.

For example, the Whitehead Institute for Biomedical Research at the Massachusetts Institute of Technology uses an object-oriented database to store information as part of the U.S. Human Genome Project. The institute's data management system, called MapBase, tracks the progress of laboratory experiments. "The data objects we deal with are extremely complex," says Nathan Goodman, a senior research scientist at the Whitehead, who has led the MapBase development team. The system, based on Object Design's Object Store, stores DNA sequences, genetic map information and nearly 200 kinds of custom objects.

Currently, Object Design's ObjectStore supports objects created in C or C++. A new version of the product, ObjectStore for Smalltalk, is in beta testing and due to be released early this year. ObjectStore Version 4.0, due for release sometime this year, will allow programmers to move objects easily between C++ and Smalltalk, Blundon says. The system also has an SQL interface, allowing it to work with traditional report-writing tools. ODI's principal competitor, Versant Object Technology, already supports Smalltalk and C++ with its OODBMS.

During the next two years, the industry is likely to standardize a system for connecting application programs with each other and with object-oriented databases. That "glue" will be CORBA 2.0, the Common Object Request Broker Architecture, a standard being developed by the Object Management Group. CORBA allows objects to communicate with each other over a network, using a system similar to Sun's Remote Procedure Call system. It's likely to be a fundamental ingredient in Sun's initial DOE offering.

New design tools may even eliminate the

C++: Still Going Strong

Despite the interest in new technology, C++ is likely to be a strong player in the object-oriented arena for some time to come. This is due to a large installed base, its perceived need by MIS directors and the increasing number of companies that are now delivering high-quality C++ tools.

Many people attribute C++'s early success to its backing by Bell Laboratories. Developed by the same research lab as the UNIX operating system, C++ gained instant credibility in the minds of many developers, despite the fact that it followed Smalltalk by several years.

Liant Software Corp. is a development firm in Framingham, MA, whose main product is C++/Views, an object-oriented development environment that allows programmers to write an application with a uniform API and field it on a variety of platforms, including Solaris, Motif, Macintosh and even Microsoft Windows. The system was originally developed in Smalltalk, says Neil Goldman, Liant's director of marketing, and then rewritten in C++ for speed and portability.

"One of the things we have done in our library is try to incorporate the things in Smalltalk that are really good," explains Goldman. "Our architecture is a model-view architecture. Our representations of views are separated from the data representation, and we have one notifying representation that handles message passing between views and models. That separation of views from data is something that Smalltalk does that C++ does not do. What we found are a number of people who have used Smalltalk and like that architecture, but have come to C++ and C++/Views because Smalltalk's performance wasn't acceptable."

Real-time systems is another area where C++ is likely to remain on top. That's because dynamic languages like Objective-C and Smalltalk perform message resolution at runtime, instead of at compile time, as is the case with C++. Smalltalk also has the overhead of automatic garbage collection, which programmers widely believe is slower than C++'s. On the other hand, a 1992 technical report paper from the University of Colorado at Boulder, *The Measured Cost of Conservative Garbage Collection*, by B. Zorn (Technical Report CU-CS-573-92), indicates that automatic garbage collection may be faster than manual systems in many circumstances.

New design tools may even eliminate the need for programmers using C++ to actually type C++ code. "The user never has to edit the derived C++," says Garth Gullekson, vice president of marketing for ObjecTime Ltd., in Kanata, Ontario, which sells an object-oriented development tool for real-time applications. ObjecTime's system is based on the concept of an *executable model*, explains Gullekson. Users edit a graphical representation of their program, which then can be animated while it runs under simulation or in real time. "You see the communication links highlighted, you see the [program] move to various states as the program is executing," says Gullekson. Using any language other than C++, he adds, would simply be too slow.

Where to Go from Here

Moving to object-oriented programming requires a mental shift from conventional programming. If you've already embarked on the transition, stay the course: Programmers seldom become proficient at OOP until they have been practicing the craft for a year or more.

If you haven't yet committed to a particular object-oriented strategy but are strongly considering going with C++, you may wish to hold off on a final decision until Sun releases DOE Version 1.0. Although DOE will support C++, it will also give people the flexibility of using Objective-C, which many programmers seem to prefer because of its simpler syntax.

Meanwhile, the number of Smalltalk offerings for SPARC systems is likely to skyrocket in 1995. IBM, for example, plans to move its Smalltalk system to UNIX sometime in 1995, according to a company insider; the company's VisualAge should follow in late 1995 or early 1996. Meanwhile, numerous other Smalltalks will be appearing on the market within the next 12 months.

All in all, there was never a better time to think objectively. ⊸

Companies Mentioned in this Article

Liant Software Corp.
959 Concord St.
Framingham, MA 01701-4613
Circle 146

Object Design Inc.
25 Mall Road
Burlington, MA 01803
Circle 147

ObjecTime Ltd.
340 March Road, Ste. 200
Kanata, Ontario K2K 2E4
Circle 148

Quasar Knowledge Systems Inc.
9818 Parkwood Drive
Bethesda, MD 20814
Circle 149

Versant Object Technology
1380 Willow Road
Menlo Park, CA 94025
Circle 150