

\$9.95 U.S.
Display Until
8/31/93


OS/2 Developer

THE MAGAZINE FOR ADVANCED SOFTWARE DEVELOPMENT

Vol. 5 No. 3



**Spotlight
On
Taligent**



**OS/2
Version
2.1**



**Speech
Recognition
Applications**



WATCOM™

Visual Development Environment For

WATCOM VX•REXX is an easy to use visual development environment for creating applications that leverage the capabilities of OS/2 2.x and exploit the Presentation Manager graphical user interface. VX•REXX combines a project management facility, visual designer and an interactive source-level debugger to deliver a very approachable and highly productive visual development environment.

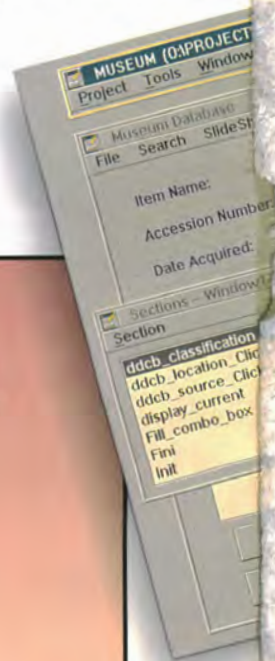
Design Applications Visually Create rich graphical applications quickly and easily using the visual design environment. With the visual designer, you can graphically create Presentation Manager interface objects, quickly customize their properties, and easily attach REXX procedures to the objects.

Integrated Development Environment Build, test and debug your application without leaving the development environment. Then package your application as an EXE file or PM macro for royalty-free redistribution. The power of the integrated development environment and debugger can also be used with your existing REXX applications.

Powerful Open Environment Enjoy the simplicity of event-driven programming together with the global editing capabilities essential for professional project management. WATCOM VX•REXX is open and extensible through IBM's object oriented System Object Model (SOM) technology. You can access all standard REXX API's including DB Manager, because VX•REXX is based on the OS/2 2.x standard system REXX.

Highlights

- ▶ Easy to use visual development environment
- ▶ Create and modify objects dynamically at both edit and run time
- ▶ Powerful project management facility
- ▶ Advanced interactive source-level debugger
- ▶ Package your applications as EXE files or PM macros
- ▶ Access to standard REXX API's including DB Manager
- ▶ System Object Model (SOM) based object manager
- ▶ Support for multi-threaded applications
- ▶ Include OS/2 style help and hints in your applications
- ▶ Supports SAA CUA'91 objects
- ▶ Autosizing and alignment of objects
- ▶ Integrated console window support for existing REXX programs
- ▶ Royalty-free run-time available
- ▶ Multiple modeless window support
- ▶ Create PM macros for applications supporting REXX as a macro language



VX•REXX™

OS/2 REXX

Interactive Debugging

If an error occurs at run-time, VX•REXX will display a traceback pinpointing the source line where the error occurred. A simple click of the mouse will return you to the source edit window to correct the error. The built-in interactive source-level debugger lets you set breakpoints, step through code and watch variables to track down complex problems.

Build Professional Applications

WATCOM VX•REXX allows you to leverage key OS/2 features to create professional applications. Build applications that dynamically create and modify CUA'91 screen objects at both edit and run-time, and include OS/2 style help and hints.

Create Multi-Threaded Applications

Every VX•REXX application contains multiple threads. One thread remains responsive to user input while others continue processing. In addition, VX•REXX provides the ability for advanced applications to easily use additional threads.

Suggested Retail: \$299*

Introductory Offer \$99*

(until September 30, 1993): includes royalty-free runtime

Call Toll Free

1-800-265-4555

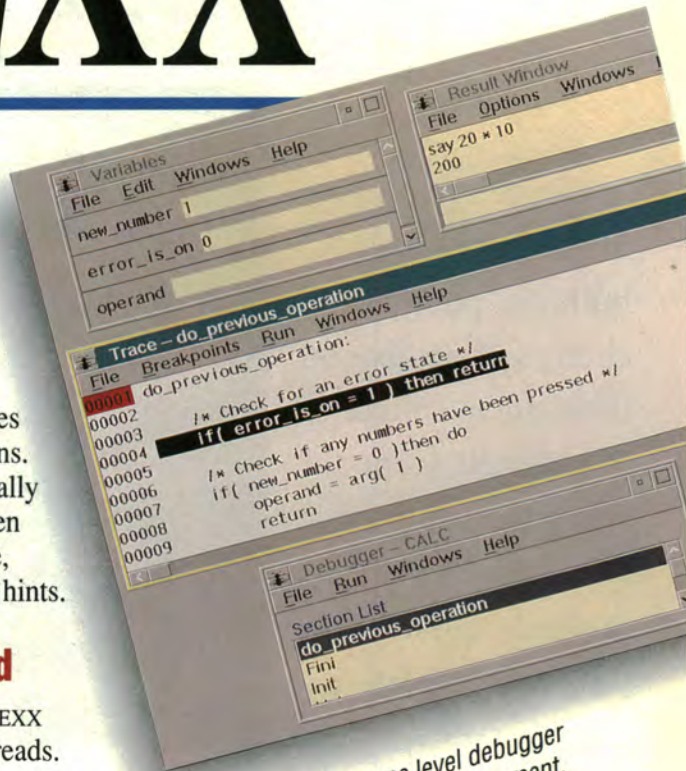
WATCOM

WATCOM International

415 Phillip Street, Waterloo, Ontario, Canada, N2L 3X2

Phone: (519) 886-3700 Fax: (519) 747-4971

*Prices and specification are subject to change without notice. Price does not include freight and taxes where applicable. Prices quoted in US dollars. WATCOM, the Lightning Device, and VX•REXX are trademarks of WATCOM International Corporation. Other trademarks are the properties of their respective owners. © Copyright 1993 WATCOM International Corporation.



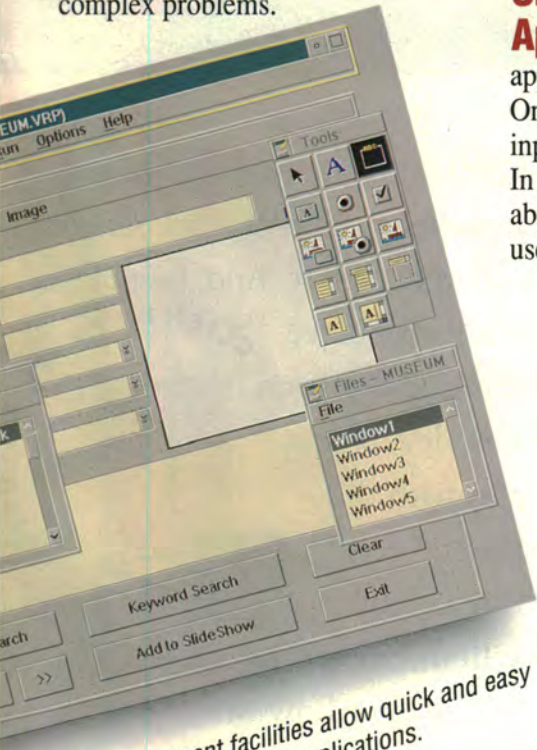
The integrated source level debugger simplifies your project development.



WATCOM VX•REXX:

The integrated visual solution builder for OS/2 2.x.

Project management facilities allow quick and easy development of your OS/2 applications.



You're closer to Client/Server than you think.

There are plenty of choices of GUI builders, plenty of choices of databases, there are even choices of cross-platform communications, but Micro Focus is uniquely qualified to provide complete client/server solutions



for your industrial-strength applications today.

The **Micro Focus Client/Server Solutions** encompass an open framework allowing you to choose Micro Focus' best tools for tasks such as building graphical user interfaces or handling program to program

communications.

Or, instead, plug in the tool of your choice.

Whether you are building

applications with distributed data, distributed application code or distributed presentation



Micro Focus Client/Server Solutions provide middleware components so developers need only know how to plug into the appropriate platform.

services, the Micro Focus Client/Server Solution is right for you. No matter what mix of thin or thick clients, or thin or thick servers, you will find it all in one place. And, best of all, the Micro Focus Client/Server products provide a bridge from the old to the new, leveraging existing resources with the latest technology.

For a brochure on the **Micro Focus Client/Server Solution**, call 800-872-6265. Discover how Micro Focus delivers "A Better Way of Programming.™"

MICRO FOCUS
Micro Focus, Inc., 2465 East Bayshore Road, Palo Alto, CA 94303. Tel. (415) 856 4161.

OS/2 Developer

JULY/AUGUST 1993

THE MAGAZINE FOR ADVANCED SOFTWARE DEVELOPMENT



EDITOR'S COMMENTS

What's New, OS/2? 5

SPOTLIGHT

Taligent: A New Paradigm, A New Approach 6



32-BIT

The Release of OS/2 2.1: An Overview 25
 CD-ROM Support In OS/2 2.1 31



PRINTING

Print Performance Options in OS/2 2.1 Printer Drivers 40



CLIENT/SERVER

LAN NetView: A Programming Overview 48
 32-Bit GUI Client/Server Application Development With Personal AS/2 52
 Client/Server Programming with Distributed Application/2 66
 Easel Workbench: Robust Client/Server Development 73



SPEECH RECOGNITION

Speech-Enabling Your Applications With ICSS 83



RESOURCES

OS/2 32-Bit Porting and
 Technical Consulting Workshops 91



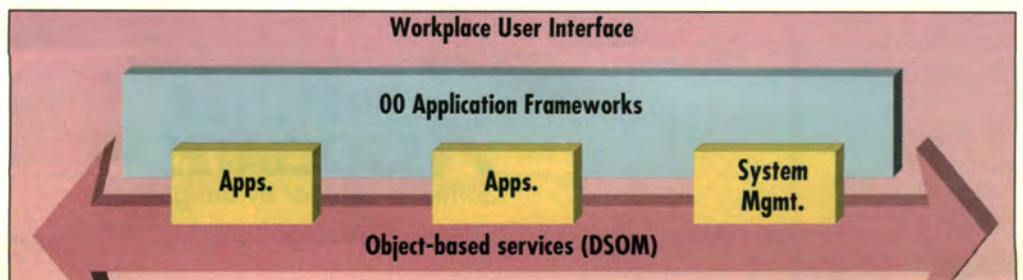
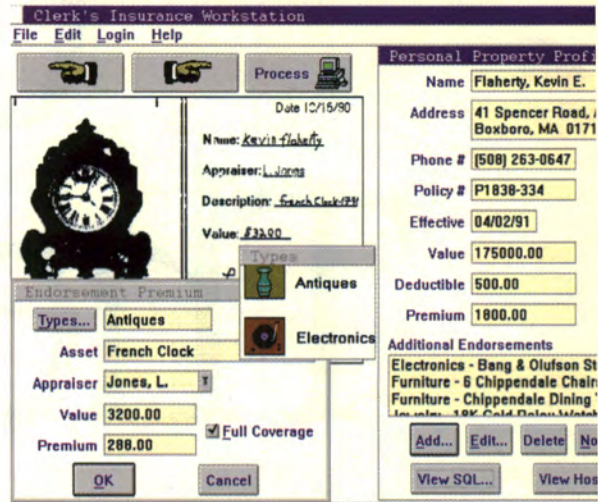
PRODUCT WATCH

New Products, New Support 94

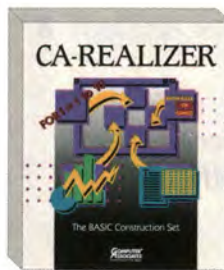


TOOLS

Debugging REXX Applications 97
 Enhancing Your Smalltalk/V Programs 108



OS/2 The Limit.

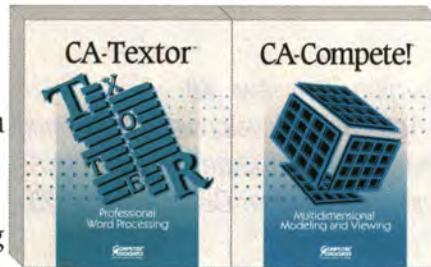


The best-looking BASIC you've ever seen.

This is what OS/2 is all about. Making your PCs do more than ever before.

Which is why Computer Associates, the world's leading software company, is pleased to introduce six souped-up applications for OS/2 users. All are proven winners that fully utilize OS/2's unique strengths. From the Workplace Shell to true multitasking to multimedia.

Businesses large and small will see benefits right out of the box with CA's easy-to-learn, easy-to-use programs. Everything from word



Professional word processing anyone can use. Spreadsheet ease, plus multidimensional modeling.

processing and project management to accounting, spreadsheets and programming becomes faster – and more affordable.

Plus, you'll get the ultimate in flexibility with this unprecedented offer: **Buy the OS/2 version of any of these applications, and we'll include the Windows version absolutely free.**



The most efficient way to get the job done. Small business accounting has never been simpler. Adding a GUI to dBASE apps is a snap.

For More Information,
Call 1-800 CALL CAI Ext. 260.

Congratulations on choosing OS/2. Now add the right apps – and watch it really take off.

COMPUTER ASSOCIATES
Software superior by design.

EDITOR

Dick Conklin

EXECUTIVE EDITOR

Nicole Freeman

PRODUCTION EDITOR

Peter Altenberg

COPY/PRODUCTION EDITOR

Lisa Gluskin

GROUP DIRECTOR

Don Pazour

PUBLISHER

Cathy Passage

ADVERTISING SALESJo Ben-Atar
(203) 498-0329Michele Blake
(212) 626-2322Dave Moreau
(212) 626-2318**MARKETING MANAGER**

Susan McDonald

ART DIRECTOR/MARKETING

Christopher H. Clarke

ADVERTISING PRODUCTION**COORDINATOR**

Tom Marzella

DIRECTOR OF PRODUCTION

Andy Mickus

DIRECTOR OF CIRCULATION

Jerry Okabe

CIRCULATION DIRECTOR

Maira Boyle

SUBSCRIPTION INQUIRIESU.S.: (800) WANT-OS2
Foreign: (708) 647-5960**BY DICK CONKLIN**

Editor's Comments

What's New, OS/2?

Now that OS/2 2.1 is available, we can be more specific about the new release. In this issue, our writers give us an overview of version 2.1's new features, plus a closer look at its CD-ROM and printing support.

SPOTLIGHT ON TALIGENT

What's going on at Taligent, anyway? Many of you have asked that question, so we paid a visit to Mr. Taligent himself, Joe Guglielmi. Joe shares his view of Taligent's world of objects and what it means to OS/2 developers.

SPEECH RECOGNITION

Everyone agrees that speech recognition will play a big role in the future of computing. Maybe we're not quite ready for 2001's HAL but some applications have put this cutting-edge technology to work today. In our first article on this fascinating topic, we'll look at a new OS/2 speech recognition development tool.

PRODUCT WATCH

We're adding a new department in this issue, called Product Watch/Calendar. It will tell you about the latest OS/2 tools and applications, upcoming shows and conferences, and new support programs for developers. Send us your press releases!

NEWS YOU CAN USE

Look for the latest OS/2 publication—the electronic *IBM Personal Software Products Developer Support News*. It's on CompuServe and several other popu-

lar bulletin board systems. In it, you'll find up-to-date news of IBM's technical, business, and marketing programs for developers. For more information, drop an Internet note to dsnews@vnet.ibm.com.



Dick Conklin

MORE OS/2 MORE OFTEN

Does our cover say "July/August"? That's no misprint—*OS/2 Developer* just went bi-monthly. Now we'll be coming to you six times a year instead of four; that means some challenges for our staff but a bonus for our readers. We'll have more news in the months ahead. Watch this space...

Dick Conklin
Editor**NOTICE**

The IBM Communications Manager Client Server/2 product, described in our Spring 1993 issue ("Extending Application Interfaces For Communications: Client/Server," page 27) was withdrawn after our press date. *OS/2 Developer* regrets any inconvenience this may have caused.

IBM OS/2 Developer: Vol. 5, No. 3, July/August 1993

Subscription information: U.S.: One year (six issues), \$39.95. Canada, Mexico, and international surface mail, add \$16 per year for postage. Canadian GST no. 124513185. International air mail, add \$30 per year for postage. Foreign orders must be accompanied by payments in U.S. funds. For new orders and customer service, call (800) WANT-OS2 (800-926-8672) or (708) 647-5960 (fax: 708-647-0537). IBM employees and branch office customers can subscribe to the *IBM OS/2 Developer* through IBM Mechanicsburg's Systems Library Services (SLSS) using the *IBM OS/2 Developer*'s order number: G362-0001. POSTMASTER: Please send address changes to *IBM OS/2 Developer*, P.O. Box 1079, Skokie, IL 60076-8079. Allow eight weeks for subscriptions to begin. The *IBM OS/2 Developer* is published bimonthly by Miller Freeman Inc., 600 Harrison Street, San Francisco, CA 94107, (415) 905-2200. Application to mail at second class postage rates is pending at San Francisco, CA and additional mailing offices.

© Copyright 1993 by Miller Freeman Inc. PRINTED IN THE U.S.A.



Spotlight

Taligent, a joint project of IBM and Apple, is redefining the computer industry and our way of looking at the creation of system software. **BOB ORFALI** and **DAN HARKEY** explore.

Taligent: A New Paradigm, A New Approach



Bob Orfali

To OS/2 developers, Taligent can seem like a "Land of Oz" of operating systems, located "somewhere over the rainbow." Why worry about Taligent when our brain cells are overloaded just trying to digest new OS/2 features like DSOM, the microkernel, DCE, LAN NetView, and motion-video multimedia? Why should OS/2 developers like us need to know about Taligent? At least that was our attitude when we began this article—but it soon became evident that there was a lot about Taligent that OS/2 developers need to know about.

A New Paradigm?

The first thing we discovered about Taligent is that its technology is multifaceted, elusive, and would



Dan Harkey

not fit neatly into our current model of "the way things are." For this article, we were not provided with specifications, product sheets, demos, or an architecture reference model. And this operating system is not like any other; it's all new and different. When we expressed our bewilderment to Mike Potel, Taligent's vice president of technology, he joked, "Joe Guglielmi had the same problem when he became CEO at Taligent. Every time he talked to somebody, he got a different view of this beast."

This article starts by looking at Taligent from an OS/2 perspective and provides a framework for the terminology of the interview. We then turn the microphone over to Joe Guglielmi. We could never match Joe's eloquence and passion for his product.

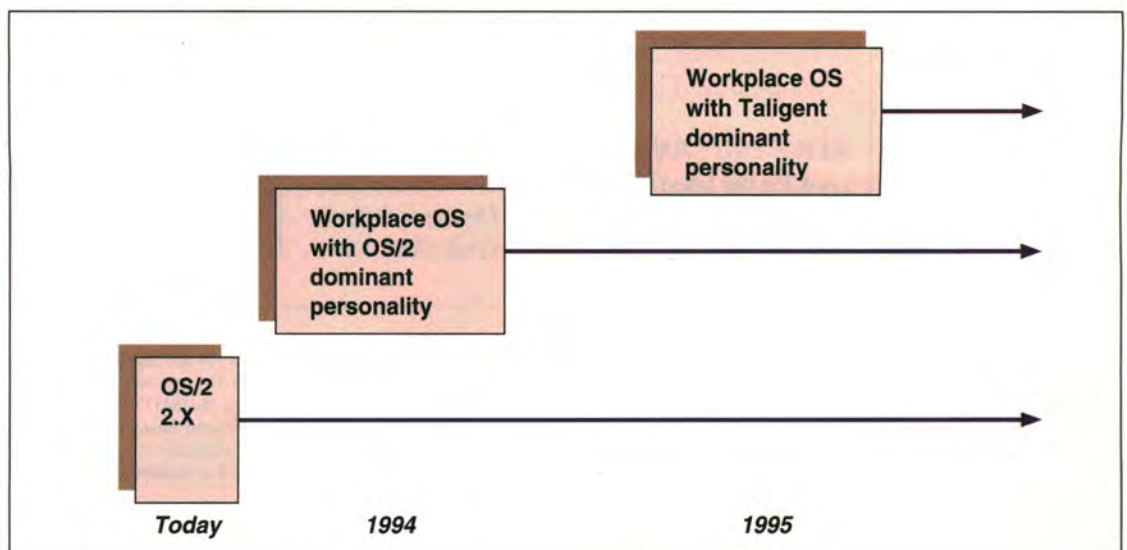


Figure 1. How IBM operating systems evolve and relate



OS/2 and Workplace OS

When does Taligent enter the picture? In Figure 1, we offer a "fearless forecast" of how the desktop operating systems from IBM and Taligent come into the picture over the next few years. In all cases, the future lies in operating systems that embrace object-oriented technology. OS/2 2.x is optimized for Intel processors and will continue to evolve throughout the '90s. The Workplace OS is a portable environment that is designed to run on top of Intel and RISC hardware from a variety of vendors, with the portability layer provided by a common microkernel. The 32-bit OS/2 applications you create should run on all platforms.

In our scenario, objects become very important. Distributed object services based on the System Object Model (SOM) become a vehicle to share objects across operating systems such as OS/2, DOS, Windows, UNIX, AS/400, and MVS. In addition, object-oriented application frameworks containing both IBM and Taligent technology will be offered for OS/2 2.x, UNIX, and Workplace OS.

In Figures 2 and 3, OS/2 2.x and the Workplace OS support DSOM, object services, and the object-oriented application frameworks. OS/2 2.x continues to evolve, providing optimal performance on the large installed base of Intel machines.

A Guide to Microkernels, Personalities, and Frameworks

Figure 3 helps explain all the new terminology and shows where Taligent fits in. Several pieces of the Workplace OS play together: the IBM Microkernel, the operating system personalities, the distributed object layer, the object-oriented frameworks containing Taligent technology, and the Workplace user interface. We will briefly introduce these pieces, then move on to Taligent.

The IBM Microkernel utilizes technology from Mach 3.0 research. Mach 3.0 is a portable microkernel developed by Carnegie Mellon University. The IBM microkernel and Mach 3.0 remove the UNIX-specific elements and provide a few well-defined services—including interprocess communications, virtual memory, ports, task dispatching, and threads—used to build all system services. The rest of the operating system functions—file I/O, user interface services, device drivers, and communications—are implemented outside the microkernel. As a result, the

microkernel can be smaller, faster, and more scalable while providing a higher level of robustness, security, and integrity. Elements outside the microkernel, such as the device drivers and file systems, can be shared by the operating systems that run on top of it. IBM's multithreaded microkernel supports symmetric multiprocessing, a technique that allows programs to run concurrently on multiple processors in tightly-coupled configurations. It is intended to support Intel and a variety of RISC processors.



Taligent At A Glance

Founded in March 1992, Taligent is an independent system software company owned jointly by Apple Computer Inc. and IBM Corp. and headquartered in Cupertino, Calif. The company has 310 employees, who use object-oriented technology to develop system software from the bottom up. The emphasis of the company's charter is on enabling the innovators and entrepreneurs who spawned the desktop revolution, allowing them to take full advantage of object technology's benefits and encouraging a new model for innovation centered around objects. Taligent's system software will be open for extension at all levels by software developers, hardware OEMs, and systems vendors. The company will license, market, and support its software platform worldwide.

Personality modules provide operating system-specific API services. IBM intends to support DOS, Windows 3.x, OS/2 2.x, and UNIX personalities, with the microkernel concurrently executing multiple operating system personalities on one machine. (One personality, designated as domi-

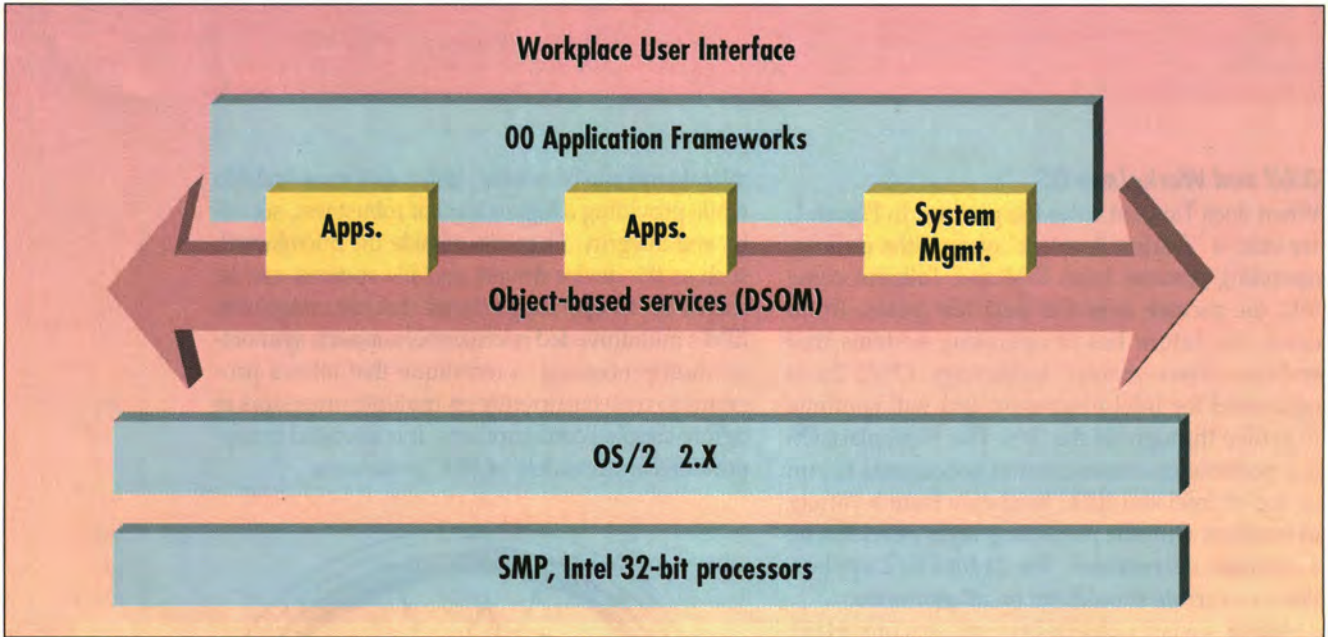


Figure 2. The Workplace Shell OOUI for OS/2 2.x

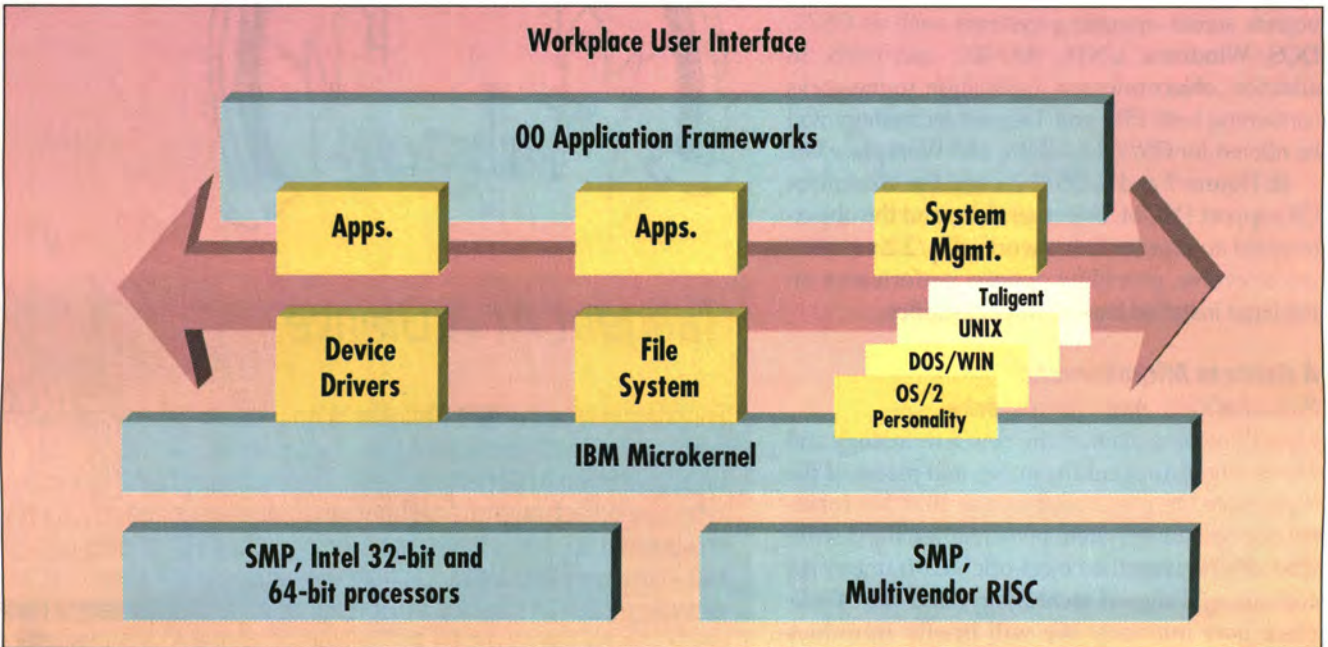


Figure 3. The Workplace OS

nant, controls the appearance of the desktop.) The personalities will preserve investments in code and application packages. The microkernel also provides the means to build specialized servers and operating system replacements.

The distributed object layer, based on IBM's Distributed System Object Model (DSOM) technology, allows objects to operate across networks. This layer also

includes services for storing, replicating, shadowing, creating, destroying, and specifying objects. DSOM is based on standards set by the Object Management Group (OMG), an industry consortium of over 200 member companies. For example, DSOM defines objects using the Common Object Request Broker Architecture (CORBA) interface definition language. DSOM uses OMG's specification of an Object

Request Broker (ORB) to find and invoke objects on different machines. The object services are intended to comply with OMG's object life cycle and persistence specifications when they become available. OMG and IBM are also working on specs for shared objects and object transactions for client/server environments. All this means that OS/2 is already deeply involved in objects, adhering to current

industry standards. These objects are, of course, built on top of a standard operating system.

The object-oriented application frameworks intend to provide a portable distributed set of object services to help create user applications. A framework makes writing object-oriented applications from raw class libraries less tedious; you begin with complete working subsystems, which can be customized for individual applications. You can also create applications by wiring subsystems together with visual application assembly tools. Third party software companies will be able to provide parts or entire components that can be assembled or modified by users or system integrators.

The user interface for IBM's "Workplace" line is (you may have guessed from the name) based on OS/2's Workplace Shell. This object-oriented user interface will be adapted for DOS, UNIX, and OS/2. Users think in terms of directly manipulating objects on the desktop rather than dealing with programs and other computer-based metaphors. IBM is working on making the user interface even easier to use by adding powerful new visual metaphors.

Now that we've introduced some of the terminology, we give the floor to Joe Guglielmi, the chairman and CEO of Taligent. We used the system detailed in Figure 3 as a reference point for our questions.

Developer: Let's start with one of the loftier questions. Taligent was jointly founded by IBM and Apple. What were your original goals, and how have they changed?

Joe: When Apple and IBM got together on this project, it was because both companies had a common vision of the importance that object-oriented technology would have in the marketplace. Both had substantial interest and investments already in that technology. We first looked at a joint development using the Power PC RISC chip and then

it grew into a more encompassing project that would commercialize objects on high-volume platforms and get the industry to make the transition from a procedural world to an object world.

fully integrated into an operating form. So as a kind of a Lucky Strike Extra, you get to do those in an integrated fashion. By the way, in an object system all those things become simply



Joe M. Guglielmi

Joe Guglielmi, 51, is chairman and CEO of Taligent, Inc. He moved to Taligent from IBM, where he was general manager of marketing and business development for the Personal Systems Line of Business. In his 30-year career at IBM, Guglielmi designed the business strategy for IBM's first successful minicomputer line, helped establish a division for marketing IBM's first PC products, and established IBM's first alliances with PC application companies.

Why is that so important?

Because our goal is to provide a dramatic improvement in the application development environment. I talk about it in terms of moving the cycle from years to months and providing a brand new environment, in terms of functionality, that will encourage innovation. It also turns out that, since we're writing an operating system from scratch, we get to do some things right. We now have a rich heritage in multimedia, advanced graphics support, and other capabilities that we know should be

objects that you can deal with. You don't worry about whether you're dealing with full-motion video or whether you're dealing with static data types. Everything is treated as an object structure.

Where do you see this technology going?

We see enormous potential in this technology—not because it's object-oriented, but because it can give the industry a brand new base that will allow us to move from today's environments, which are constrained, we think, by the

operating system capabilities. Just as OS/2 2.0 unleashed the power of a microchip we'd been shipping for years, this will begin to unleash the power and

This is where our strategy has changed dramatically. Initially, the notion was to deliver one large brand-new operating system, top to bottom, in one really

ments, System 7 investments carry forward? How do we ensure that we don't force a change in the marketplace all at once to a new technology?

So how do you do that?

We've focused our strategy over the last year on three or four things. One is moving the project from a research project with two or three technology bases to a product development environment where we now have the focus on delivering products that respond to customers' requirements to the marketplace. Second, we've spent a lot of time staging the project... we can't do it all at once. We'll never be able, in one release, to compare to those established operating systems, so that's a bad goal. So let's focus on areas in which we think the

"What we provide is an environment that lets (developers) do a much better implementation of what they do best."

creativity in the industry in a different way. We won't be constrained by the old paradigms of procedural operating systems.

How will you get this technology into the marketplace?

significant drop. We've become more realistic about that; we've had to deal with very pragmatic issues. How do we, when providing this great step function, deal with current investments? How are we going to make sure the OS/2 investments, AIX invest-

The Path To A Good GUI Isn't Always Clear

Navigate the maze with the NEW 2.1 Professional Developers ToolKit

Promises are cheap, GUI development tools aren't. Gpf 2.1 demo software is FREE. Try Gpf before you buy any tool and you'll agree that Gpf 2.1 DELIVERS.

With this powerful point and click visual programming environment you can create a CUA '91 Graphical User Interface for OS/2 Presentation Manager® or MS DOS/Windows® in as little as 10% of the time required to hand code the same design.

What You See Is What You Get programming reduces weeks of coding to hours. Quickly prototype and test your interface, then let Gpf write the code. Gpf generates error free, structured Object Oriented C or C++ source, complete with embedded SQL for OS/2 DataBase. Custom Help and Logic are added to controls as they are drawn to create full Client/Server or Stand-alone applications!

Gpf is hosted on OS/2 2.x and generates native code for OS/2 2.x, OS/2 1.3.x and MS Windows 3.0 or 3.1 Of course this means maximum performance with no run time module and no royalties!

Seeing Gpf is believing!

- Design 32 or 16 bit GUIs for IBM OS/2® or Microsoft Windows®
- Be creative, WYSIWYG design environment
- Full application and/or DLL generation
- Minimal time to application delivery
- Full CUA '91 Control set, 32 or 16 bit
- Reusable Objects
- Automatic documentation

Gpf

Try us, FREE Demo Software Available
Order Gpf Pro ToolKit today for just \$1440⁰⁰
Separately: Gpf 2.1 for \$1295⁰⁰ & GpfTools for \$195⁰⁰
Or, try: Gpf 2.0 Single Platform, now available for just \$495⁰⁰

Call Gpf Systems Inc. at: (800) 831-0017



* GUI Programming Facility

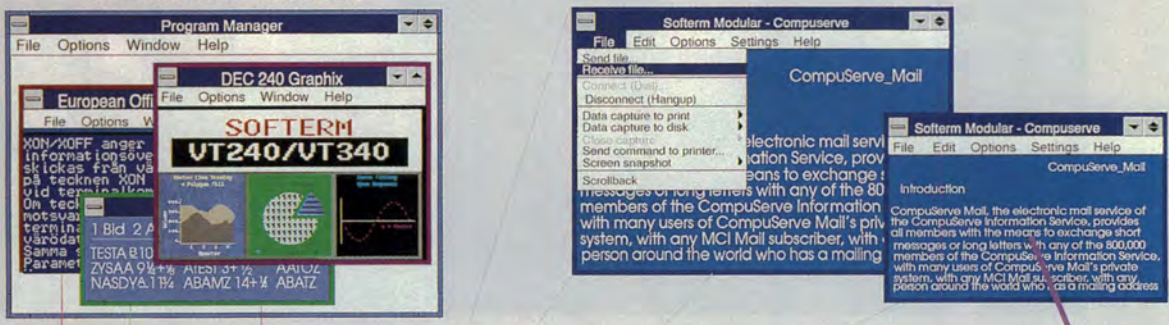


SYSTEMS, INC. Phone (203) 873-3300 • fax (203) 873-3302 30 Falls Road, Moodus, CT 06469

NEW! SOFTERM[®]++ FOR OS/2[®] Only \$50

The replacement for Softterm[®] Custom that is bundled with your OS/2[®]

Concurrent Sessions

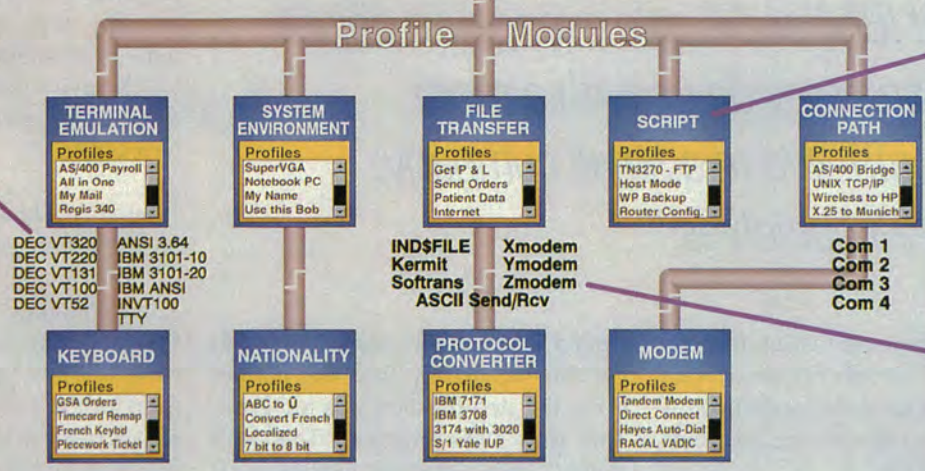


New!
Graphic Terminals
DEC 340
DEC 240, 241
Tektronix 4010, 4014

New!
Dynamic Font
132 Column Support
Procognitive™
The Softterm Graphical User Interface

Also New!
DEC 320,220

New!
Scripts
Genome™, The Softterm Script Language



New!
Zmodem
IND\$FILE

"The only high-performance graphic terminal comm program for OS/2.®"



800-225-8590



Softronic, Inc. 5085 List Drive, Colorado Springs, CO 80919 FAX: 719-548-1878

OS/2 is a registered trademark of IBM Corp.
Softterm is a registered trademark of Softronic, Inc.



technology substantially leverages itself. We're going to stage our technology to focus on both corporate and software developers. What do they need first, to provide value? Taligent's proposition is that the value comes from the developers. We don't compete with them for word processors or spreadsheets, and we don't do databases. What we pro-

operating systems like OS/2, AIX, and System 7. That's a pretty profound change. That's something that's not very natural for a team developing a new operating system to do. A development team wants to make the biggest step function change, one that differentiates between current and future environments. Well, we con-

available to current operating systems?

What does that do for those operating systems?

First of all, if we've done classy implementations of the technology, it will enhance them. OS/2 will be more competitive; System 7 will be more competitive. By the way, we're not limiting it to just those two; our strategy is to go after as many of the current operating systems as possible. Think about all the UNIX environments or—ultimately—any 32-bit system that can carry the technology.

What are the benefits to Taligent of making this technology available to others?

The technology will make OS/2 and System 7 more competitive and they, in turn, will provide us a transition platform for the wide-scale introduction of objects to the marketplace. We benefit if IBM and Apple evangelize object technologies and provide us with a more evolutionary path. We're a small company; we can't do it all ourselves. But by putting pieces out in the marketplace, we begin to provide a transition mechanism—that is, investments made in applications that use that technolo-

"We benefit if IBM and Apple evangelize object technologies and provide us with a more evolutionary path. We're a small company; we can't do it all ourselves."

vide is an environment that lets them do a much better implementation of what they do best. Our delivery channels—Apple, IBM, and others—are also going to add value to this.

Yes, but every new operating system needs to do that kind of stuff...

It does. But the second new notion is that we would take pieces of this technology and make it available to current

cluded that without some strategy to get the technology into the market early, the step function was too great. This isn't 1983 anymore. This isn't about "I have a great technology; we'll just get people to stop what they're using and move over to it." We have example after example in the marketplace today where great technology can't get commercial acceptance. So the notion was, why don't we take pieces of this technology and make them

gy under a mature operating system can be carried forward into Taligent when the time is right. So when Taligent comes, customers can decide whether they want to move to it or not. This is not a forced march. It doesn't mean one Monday morning OS/2 goes out of business or System 7 goes out of business—that's not the strategy.

But if you've made your object technology available on other platforms, why would anybody ever want to move to Taligent?

Taligent will be fully object-oriented with a very consistent object model. When you enter Taligent, everything in the system is an object, with no differentiation between a system object and an application object. We can implement a very consistent object structure: once you're in the object space, you can leverage every element in that environment. You won't ever have to deal with the procedural elements of a system. Take NextStep, for example. Steve (Jobs) has a great user interface tool, but you still have to bounce into UNIX from within your applications.

What are the advantages of having objects at the operating system level?

Envision a growing industry that would provide system frameworks, networking frameworks. Take communications... If they don't like ours, IBM or Novell could replace it with their own implementation. Device adapters can inherit characteristics of a particular device class and customize them to fit their particular needs. It's a very consistent way of taking advantage of new hardware. System software can then keep up with fast-moving hardware technology.

OK, so what's in it for application software developers?

If we do our job right, in the mid-90s application developers will not be writing code but will instead be shopping for objects. If you think about it, there will be thousands of objects; how do I

know which one I want to subclass or reuse? We're building a very consistent development environment with viewers or browsers that let you try out these objects and understand what they can do for you. You'll have a full environment that lets you think of your world in terms of objects.

Will some of that environment be made available on OS/2?

Yes. I didn't mean to imply that the layers of components we provide for OS/2 won't be great; they'll move objects forward and condition the mass market for objects. Taligent has the



MOST COMPUTERS ONLY RESPOND TO TOUCH. WE'RE HELPING THEM DEVELOP A SENSE OF HEARING.




Speech recognition is here!

Introducing ICSS, the IBM Continuous Speech Series. A developer's tool that has the flexibility to speech enable your programs. This software product provides total speaker independence—there's no special voice training required. If someone can talk, they can use your speech enabled program. And ICSS also features "continuous speech"—allowing you to

Speak completely naturally, the way you would to a friend. With a 1,000 word active grammar that is dynamically switchable, the possibilities are limitless. ICSS runs on 486DX machines running OS/2 2.x, as well as RISC/6000 machines running AIX 3.2. For more information, call 1-800-627-8363.

ICSS from IBM. **IBM**
We hear you.



added luxury of being able to put the whole thing together from the bottom to the top.

Speaking of the bottom, both Taligent and OS/2 will be running on the same Mach 3 microkernel—is that correct?

We're on the same microkernel, but IBM is not shipping Mach. They're shipping a microkernel based on Mach technology. Mach is a big, heavyweight thing; IBM and Taligent are creating a much more lightweight version of

try. I'm trying to answer the guy who says, "What are you, nuts? Mach can't run an object system the way it's currently designed." The answer is "It can't, but the kernel we're implementing can." It's really this notion that the core technology is optimized object technology and this new common microkernel works across OS/2, UNIX, and ultimately, hopefully, the industry. That's where the world's going.

So is this microkernel one of the first pieces to be delivered?

mixing objects from a variety of vendors. The fact that the object's interface is the standard defined by the Object Management Group—the CORBA specification—is also very attractive.

Will you support DSOM then, since it's based on CORBA?

Of course. DSOM is a very easy case. Its ORB, which is CORBA compliant, is one way we deal with objects across networks. We know there are objects on this network that are going to be moved around.

So you're depending on DSOM to bring CORBA. Is IBM representing Taligent's position in the OMG?

IBM, like Sun or HP, has put some time into defining the CORBA compliance specs. In our implementation, we support that spec as one of the important distributed object models in the industry. Now take Microsoft: they're doing CAIRO. I don't know if CAIRO is CORBA compliant; the likelihood is not. What's going to be our position on Microsoft's model? If it's a high-volume model in the future, we're going to have to find a way to support it as much as possible. That doesn't mean we like it... or don't like it. Our position is pragmatic. We have to exist with whatever becomes a high-volume platform in the future. We're working with IBM, Apple, and others to try to make the transition as seamless as we can, but we're coming from two different worlds. What we won't do is suboptimize our world. If we end up lowering our goal of full utilization of object technology, we will lose the great differentiation that comes from it.

Which IBM groups do you work with on object technology?

I deal through the PSP division, which means that Larry Loucks, Cliff Reeves, and Lee Reiswig are my contacts in the IBM world. So when I deal with Larry and Cliff, we either agree or don't agree, and I expect them either to bring

"In Taligent, we are dealing with a system optimized for a very large number of small objects.... We don't move a few big objects. We move a lot of very little things around, so we have a tremendous amount of tasking going on."

Mach and creating personality-neutral servers and services. Taligent is adding the mechanisms to make it a first-class object environment. We're working on the performance implications of moving thousands of objects around in a system. IBM will build the microkernel that we'll use as part of our common strategy.

Mach is an open microkernel, something perceived as a benefit. Aren't you losing that by going to your own microkernel?

I'm not commenting on the open aspect. I'm commenting on Mach's poor performance for doing objects. We have over four years of experience writing microkernels for objects, and we know how to do one. Object people don't think you can do objects on the current version of Mach—it may be OK for doing UNIX-type things. The common IBM microkernel solves those technical problems. On the openness question, it's up to IBM how they supply the microkernel back to the indus-

It's actually being developed at IBM. Our operating system development team is working with the IBM team.

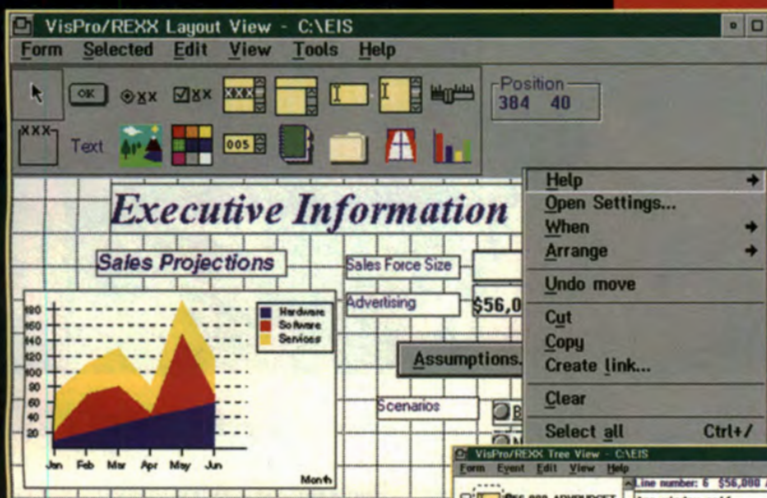
Are your object frameworks built on SOM and CORBA technologies?

I think we've come to appreciate what SOM is and what it isn't. I think there's a greater feeling that SOM's design point and the problem set that it solves are important, very important. Whether or not we can, as a team, adopt all of the goals of SOM in our design point is the discussion we're having with IBM now. The reason is, I think, pretty neat to think about. In Taligent, we are dealing with a system optimized for a very large number of small objects. That's how we get performance. We don't move a few big objects. We move a lot of very little things around, so we have a tremendous amount of tasking going on. We've got the machine burning and we've optimized for performance. We got a lot of value out of that optimization. SOM is best when the primary objective is

OS/2's time is here, thanks to

VisPro/REXX™

- Visual programming
- WYSIWYG editors
- Drag & drop interaction
- Client/server programming



VisPro/REXX is an easy-to-use visual programming tool with a CUA '91 graphical user interface.

By using VisPro/REXX, you can develop OS/2 GUI applications in record time.

Even if you've never programmed in REXX.

And if you're an experienced programmer, you'll love the way VisPro/REXX gives you easy access to OS/2 programming features,

such as buttons, lists, sliders, graphs, charts, graphics, OS/2 Database Manager, APPC, and HLLAPI.

Free demo disk available for a limited time

VisPro/REXX

"... brought the house down at a recent OS/2 conference in Colorado... it is to REXX and OS/2 what Visual Basic is to Windows and DOS."

—Robert X. Cringely
InfoWorld
January 25, 1993

Best of all, it's small

System requirements: OS/2 2.0, 5 Mb memory, 1.5 Mb hard disk space

Sales (919) 387-7391

Tech Support (919) 380-0616

Fax (919) 380-0757

OS/2 2.0 and CUA are registered trademarks of International Business Machines Corporation.

HockWare™

Formerly UCANDU Software

P.O. Box 336, Cary, NC 27512-0336

© 1993 HockWare, Inc.

\$299

Introductory price
plus shipping & handling

Entry level version available for \$99
plus shipping and handling





the rest of the company with them or tell me that I've got to go deal with the situation differently. By the way, they're doing a great job evangelizing Taligent within IBM.

and beyond. This will give you the highest affinity with Taligent. Why do I say that? Because I'm working very hard to make that happen. Can I tell developers that everything they do in every release

as we can with IBM to make it as painless as we can. We hope it's zero pain. That's our goal. But it probably won't be.

How do you balance that with Apple's needs?

The same way. When we talk to Apple, I have the same strategy. The decisions we make optimize on where Taligent is going, mindful of the fact that helping make Apple successful with System 7 and IBM with OS/2, AIX, and Workplace OS is good for us, because every user in those two or three camps is going to be positioned much better to come to Taligent.

Will the Taligent user interface look like Mac, Workplace Shell, or something brand new?

That's unclear, frankly, and let me tell you why. I believe that the opportunity to build the Taligent desktop completely from an object paradigm gives us degrees of freedom that can move the desktop beyond what is possible

"If you're an OS/2 developer, keep working with OS/2—release 2.1 and beyond. This will give you the highest affinity with Taligent.... I'm working very hard to make that happen."

If you were speaking directly to, let's say, leading edge OS/2 application developers who want to do the right thing and position their product for the future, what kind of strategy or recommendation would you map out?

Here's the strategy I think has to be followed. If you're an OS/2 developer, keep working with OS/2—release 2.1

will be cared for seamlessly? No, I can't tell them that. But they should know that we are working very hard with the PSP development group to put as much of this technology as possible into OS/2 and AIX, to enhance their competitiveness wherever we can, because we believe that's good. We believe a big OS/2 base is in Taligent's interest and, therefore, we're going to work as hard

WITT: Instant record,

Now there's a faster, easier, more affordable way to test your applications. Introducing Workstation Interactive Test Tool (WITT) 2.1, for OS/2® 2.0. An AD/Cycle product from IBM Programming Systems.

WITT does it all. Record, edit, playback and screen compare.

Introducing
Workstation
Interactive
Test Tool 2.1

So once you've built a test case, you can use it over and over again.

It works for applications on OS/2 PM, and host-connected MVS, VM, VSE and OS/400®.

Now regression testing is more reliable. New code testing is easier.

• And learning is easier, too. With a friendly online tutorial.

instant replay,



What's more, WITT won't test your budget. Because it's very affordable.


It only takes an instant to call 1 800 426-3346, ext. 64, and ask for more information and our free evaluation demo. Or, better yet, order WITT today with a no-charge two-month testing period and developer hotline assistance.

• The instant you get WITT 2.1, you'll be more productive.

instant relief.

The development team at IBM Santa Teresa created WITT 2.1 to help you improve the quality of your applications.





today. So we're focused on that; we're trying to figure out where we can exploit this technology the best. Let's take one model that could happen. In a Taligent world, since everything is an object, applications don't exist the way they exist today. You turn your system on and it comes up with a set of objects that you deal with, and you arrange them in workflows. The notion of a workflow world centered on the user is much more realistic in a full-object world. Everything on the screen is a full object linked from the top to the bottom of the system. When we are ready to deliver a full-object model to the screen, we're going to go back and work with IBM and others on where they are today. We'll try to make it as evolutionary as we can... we'll find ways to see if we can't evolve multiple releases of the Workplace Shell or the Mac look to where Taligent is heading.

What's your planning horizon?

We're the only ones in town who are worried about 1996. I don't mean that in a negative way. Everybody else is

successful by themselves. They will not have the kind of critical mass that you require. So the design point is to fully exploit the technology and then look backwards to see how we move the current base from here to there, and there are several bases. But the thing people



ought to feel good about is that we're thinking about it. It's not like we're over here doing our best thing and hope that someday it all works. This is the first time there is a group that is allowed to really focus on where we want to be. At

Will Taligent technology bring together the System 7 and OS/2 worlds?

As a separate company, I can talk with a lot of authority about what we're doing, and I can tell you only a little bit about how we're working with these two companies. If you want their view of the world, you've got to go ask IBM and Apple. I will tell you this: both Apple and IBM feel strongly about keeping their current systems viable, and we support that. There's no model here, under any circumstance, where all of a sudden on Monday morning OS/2 or System 7 goes away. That isn't in the cards, because there's no viable scenario where this whole thing can make the transition in one day to a new environment. So there is going to be, as you expect, a continuation of a substantial investment in those current operating systems, and we're going to find a way to make the path to Taligent very attractive.


Are you going to sell a shrink-wrapped Taligent through software stores?

Yes, but that isn't our model. The whole distribution model is changing today. The store distribution model is going away; I mean, who distributes operating systems today? The hardware manufacturers. So our strategy is to get the distributors to accept the technology, to provide the critical mass, the target volumes, so that developers can see it. Shrink-wrap and user sales will come after that. So the whole model is different. It isn't about running big ads in magazines, hoping that users will ask developers to create a great application. It's about going to developers and saying, "Look, you're building a great application. You can keep doing it. We're going to give you the tools to do it in a third of the time and to maintain it more easily by an order of magnitude. You know, by the way, that when you upgrade it, your cycle will go from two years to six months. That's worth a lot by itself. Oh, there's another opportunity coming. Have you thought about spreadsheet

"I will tell you this: both Apple and IBM feel strongly about keeping their current systems viable, and we support that. There's no model here, under any circumstance, where all of a sudden on Monday morning OS/2 or System 7 goes away."

worried about 1993, '94, and '95. I have to stay worried about 1996, or '97: how can this technology, when it's fully exploited, change the playing field in the industry? If we miss the goal of providing dramatic new technology, the industry won't change. If the industry doesn't change, none of the interim operating systems are going to be suc-

cessful by themselves. They will not have the kind of critical mass that you require. So the design point is to fully exploit the technology and then look backwards to see how we move the current base from here to there, and there are several bases. But the thing people ought to feel good about is that we're thinking about it. It's not like we're over here doing our best thing and hope that someday it all works. This is the first time there is a group that is allowed to really focus on where we want to be. At



**Down the road, the shortcomings
of other client/server development tools are
certain to become apparent.**



1-800-OBJECTS

swell and data flow begins to explode? Or when they decide they want to take advantage of the power inherent in the latest release of OS/2? Or when they need to support client/server models beyond a simple Database Server architecture?

It's in situations like these when most application development tools show their weakness. And when Easel's robust solutions display their considerable strength.

ENFIN™ is the perfect solution for organizations that want to use an industrial-strength object-oriented development

environment to build their client/server applications.

Enterprise Workbench™ delivers cross-platform development solutions and unsurpassed connectivity options for building new client/server applications.


EASEL Renovator+™ allows organizations to extend legacy applications into the client/server world.

Importantly, each of these visual development environments offers benefits that go well beyond those of mere GUI builders.

We have a White Paper that describes the five models of client/server computing and explains how Easel products support every one of them. For a free copy, phone us at 1-800-OBJECTS. (Calling from outside the U.S. and Canada: 617-221-3014.)

Otherwise, when you want to develop enduring, enterprisewide solutions, you may find yourself hitting the wall.

EASEL
Corporation



engines, word processor engines, graphics engines? Have you thought about work flow as an application? You can really do it now. Have you thought about it?" So we're creating a different world of applications and when we start this model, all of these characteristics will allow hardware manufacturers to differentiate their products.

We want to sell through the high-

"We want to sell through the high-volume channels. If we don't do that, our adoption curve is going to be seven years. It takes seven years to get an operating system to the market, if you're good. I want to be a success faster than that."

volume channels. If we don't do that, our adoption curve is going to be seven years. It takes seven years to get an operating system to the market, if you're good. I want to be a success faster than that. So all of these strategies are working toward substantially reducing the adoption curve for the new technology.

How do you see the operating system pie divided in 1996?

I'd be presumptuous to give you a target, but I will tell you that existing systems will continue to have a large mar-

ket share. Between now and 1996, there's just not enough time to substantially change the rate flow. I don't know what the division between NT, DOS, OS/2, and Windows will be; who knows? That's completely up to them. We haven't seen NT yet, but they'll have a large share. I believe OS/2 will gain critical mass and have a substantial share. I think Apple will continue

to have a big share. UNIX will probably still own eight or nine percent of a growing market. By 1995 there'll be 35 million micros shipped per year, so even 10% of that is still a big volume.

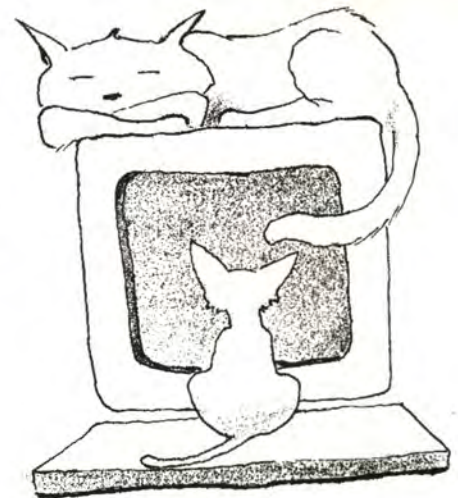
And then Taligent will begin to show up. Now I can't tell you exactly how big our wedge will be, but it will not be zero, because we intend to get to market by the mid 90s; hopefully earlier. And I think that, as Taligent becomes an important environment and we demonstrate its utility, that wedge will grow more rapidly than historical growth rates in new operating systems.

ACKNOWLEDGMENTS

We would like to thank the people who contributed to and reviewed this article, including Mike Potel, D'Ann Ostrom, Cliff Reeves, John Soyering, Lee Reiswig, Larry Loucks, and many others.

Robert Orfali, IBM Corp., 5600 Cottle Road, San Jose, Calif. 95193. Orfali is an advisory programmer in IBM's client/server solutions department. He joined IBM in 1972 and is the architect of Quick Client/Server, an application server platform for OS/2. Orfali is coauthor of Client/Server Programming with OS/2 2.0. He received an M.S. in electrical engineering from the University of California at Berkeley.

Dan Harkey, IBM Corp., 5600 Cottle Road, San Jose, Calif. 95193. Harkey is an advisory programmer in IBM's client/server solutions department. He joined IBM in 1977 and is the lead developer of Quick Client/Server, an application server platform for OS/2. Harkey is coauthor of Client/Server Programming with OS/2 2.0. He received an M.S. in computer science from Santa Clara University.



GET ON A CLIENT/SERVER TRACK WITHOUT DERAILING YOUR DEVELOPMENT

If you're client/server bound, KASE:VIP visual design and code generation tools get your applications on an OS/2 or Windows GUI-based track — while leveraging your current development assets, your existing staff and even your existing application logic.

KASE:VIP automatically generates all the source code you need to build seamless links between graphical interfaces and SQL databases or CICS transactions. In standard languages like C, C++ and COBOL, so you can avoid the derailing overhead and restrictions of proprietary languages, runtimes and royalties.

KASE:VIP also offers an OPEN ARCHITECTURE license that gives you the flexibility to control the code generation process and accelerate your development. You can prototype and automatically generate code unique to your needs — such as specific line-of-business functions, proprietary APIs or your own application "hooks." All of which lets you leverage the expertise of key developers across your entire organization.

Avoid development derailment. Use KASE:VIP to keep your client/server development on the fast track.

GET YOUR GUI-BASED SQL AND CICS client/server development on a COBOL, C or C++ fast track — without proprietary languages or runtimes. Call **1-800-888-4335** for a KASE:VIP demonstration disk, or for information about KASEWORKS seminars in your area.

KASEWORKS™

△△△△ Enabling Client/Server Strategies

(FORMERLY CASEWORKS)

3295 RIVER EXCHANGE DRIVE, SUITE 430

NORCROSS, GA 30092

(800) 888-4335

(404) 448-4240



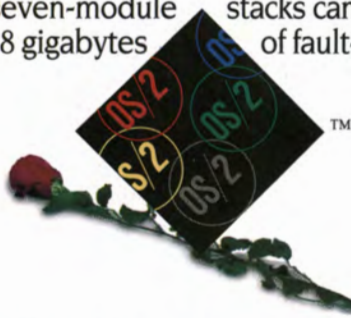
Made For Each Other

RAIDION® LT and OS/2

A Perfect Match. Introducing RAIDION LT fault-tolerant disk arrays for IBM OS/2 or LAN Server based systems. RAIDION LT and RAIDWARE™ management software combine to create the first modular, RAID level 5 fault-tolerant storage subsystem designed specifically for use with OS/2.

Reliability Plus. The LT offers the features you expect of a fault-tolerant storage subsystem, and then some. The Hot Swap of individual modules is fast and easy. An On-Line Spare option enables the subsystem to remain operating in a fault-tolerant mode even if one drive in the array fails. RAIDION LT is designed to give OS/2 users 100% data availability, 100% of the time.

Easy to Install, Easy to Upgrade. Individual disk modules, with either 560 megabytes or 1.0 gigabytes of formatted storage, are combined to make a three-module array of either 1.2 or 2.0 gigabytes. And increasing capacity is simple. Add one or more modules to a three-module stack to increase capacity. A total of 4 seven-module stacks can be combined to provide up to 28 gigabytes of fault-tolerant storage.



*For more product information or the name of your nearest authorized
RAIDION distributor call 1-800-395-3748.*

MICROPOLIS®

IBM and OS/2 are registered trademarks of IBM Corporation.



And
the
winner
is...



AM

BEST
OS/2 EXPLOITATION

BEST
PRODUCTIVITY GAINS

BEST
DATABASE INTEGRATION

BEST
CUSTOMER SATISFACTION

BEST
ADVANCED CLIENT/SERVER TOOL



In Recognition of
your contribution to
a new class of
OS/2 2.0 APPLICATIONS

INTELLIGENT ENVIRONMENTS
CORPORATION
presents an award to

Intelligent Environments

for shipping

Applications Manager

Intelligent Environments
USA 508-640-1080 ♦ EUROPE +44-932-772266

• BOSTON • CHICAGO • HOUSTON • LOS ANGELES • LONDON • FRANKFURT • STOCKHOLM •

INTELLIGENT ENVIRONMENTS INC., 3 HIGHWOOD DRIVE, TINKSBURY, MA. ALL PRODUCTS AND PRODUCT NAMES MENTIONED ARE REGISTERED TRADEMARKS OF THEIR RESPECTIVE COMPANIES.



The release of OS/2 2.1 presents software vendors with a platform for developing 32-bit OS/2 applications and tools as well as DOS and Windows applications. Its new and enhanced features let developers exploit emerging PC technologies. **BY MIKE KAPLY**

The Release of OS/2 2.1: An Overview

OS/2 2.1 has many major and minor enhancements, including better device support (CD-ROM, SCSI, video, and printer), WIN-OS/2 3.1, a 32-bit graphics engine with 32-bit seamless drivers, ISO-compliant fonts, advanced power management, Personal Computer Memory Card International Association (PCMCIA) support, inclusion of Multimedia Presentation Manager/2, and pen and security "hooks." In addition, improvements have been made to the overall performance, specifically to WIN-OS/2 3.1 and the Workplace Shell. Figure 1 is an overview of the OS/2 2.1 system components and the major areas in which service fixes and enhancements were made.

OS/2 2.1 ON CD-ROM

With the success of CD-ROM Professional Developer's Kit, IBM is shipping OS/2 2.1 on a CD-ROM with two accompanying disks to begin installation. The upgrade is also available on 3 1/2" and 5 1/4" disks.

DEVICE SUPPORT

CD-ROM and SCSI support has been improved in OS/2 2.1; OS/2 now provides support for CD-ROMs from eight manufacturers: Hitachi, IBM, NEC, Panasonic, Sony, Texel, Toshiba, and CD-Technology. SCSI adapter support includes Adaptec, Future Domain, DPT, and IBM adapters.

More display drivers are also provided in OS/2 2.1, with seamless video drivers included for 13 SVGA adapters including Tseng, ATI, Western

Digital, Headland, and Trident. 8514/A support is now provided on 32-bit seamless drivers. Other supported video modes are VGA, XGA, and XGA-2. Depending on the driver and the hardware configuration, these drivers can support resolutions as high as 1024-by-768 with 256 colors and 800-by-600 with 65,536 colors.

The 32-bit model removes the 64K segment limits, increasing the limits on most resources including the number of definable points and handles.

PRESENTATION MANAGER

A new 32-bit graphics engine, first available with the Service Pak in October 1992, gives faster, more accurate graphics performance. The major changes are:

- 32-bit model
- Increased limits
- New GpiPolygons API
- Addition of Palette Manager
- ISO-compliant fonts
- Improved performance
- 32-bit seamless drivers.

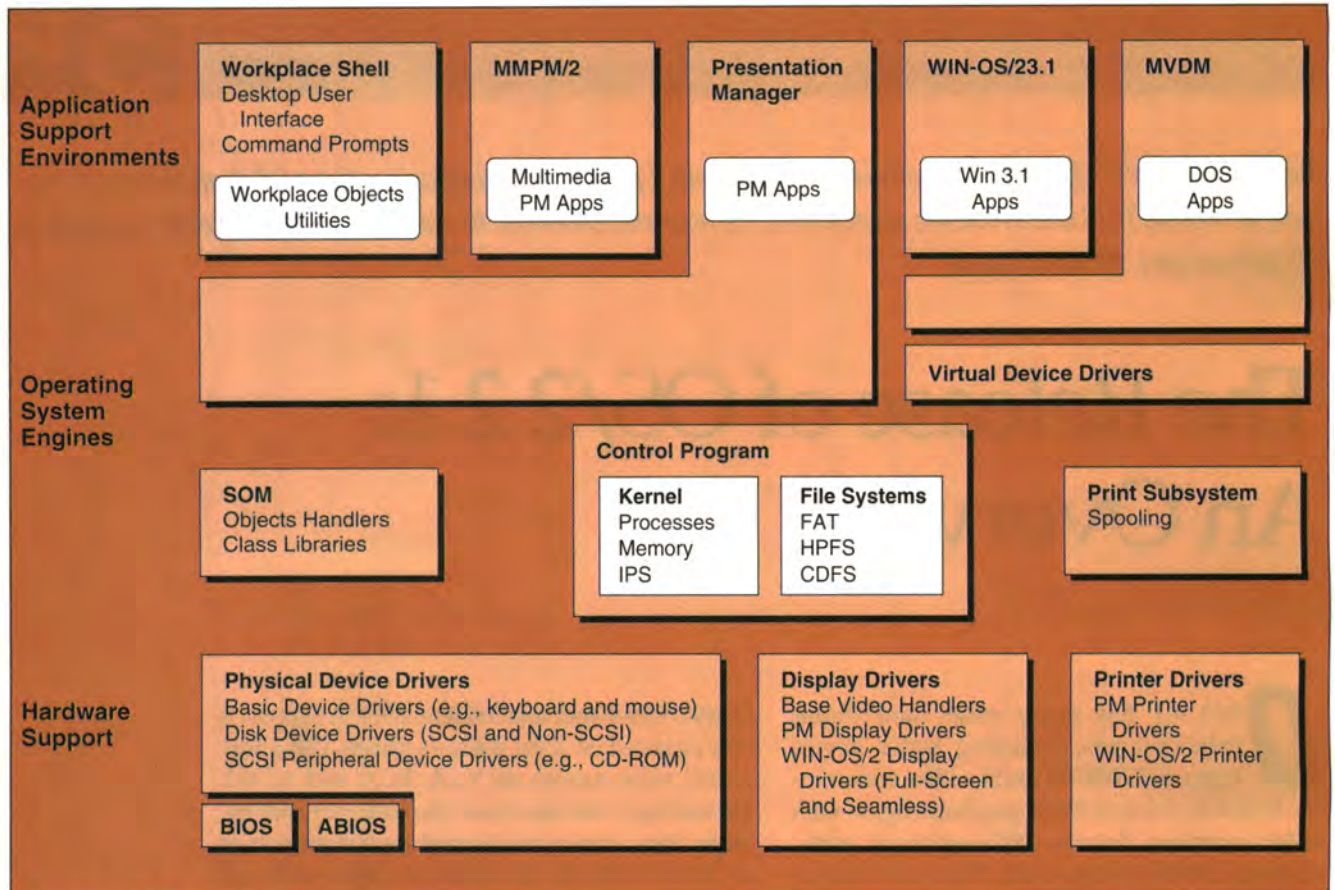


Figure 1. OS/2 2.1 system components overview

The 32-bit model removes the 64K segment limits, increasing the limits on most resources including the number of definable points and handles. Tables 1 and 2 show how some limits have been increased.

The new `GpiPolygons` API improves high-speed drawing of polygons; the 32-bit implementation removes prior restrictions and provides a faster, more powerful mechanism for drawing one or more polygons. (For more information about the `GpiPolygons` API, see the *OS/2 2.0 Programming Guide, Vol. 3*.)

The palette manager allows developers to take full advantage of SVGA and XGA drivers displaying 256 colors. Applications can now define their own colors instead of relying on dithering or closest fit. For more information, see the *OS/2 2.0 Programming Guide, Vol. 3*.

Another change to the graphics engine is the introduction of ISO fonts

that conform to the ISO 9241-3 standard. Use of these fonts is optional; however, the System Proportional font is replaced in OS/2 2.1. All other ISO-compliant fonts are provided in addition to the original OS/2 fonts.

The performance of the graphics engine has been improved by the removal of many "thunks" within the system. Also, many graphics operations were enhanced for better performance.

Another important addition with the new graphics engine is the ability to develop 32-bit seamless drivers. These drivers provide better performance than their 16-bit counterparts. Note that in addition to the new 32-bit drivers, 16-bit drivers are still supported by OS/2 2.1.

WORKPLACE SHELL

Many improvements have been made to the Workplace Shell. The icons of

Workplace Shell objects can now be changed by dragging and dropping any icon or object onto the General Page in the objects' settings. In addition, desktop notebooks now use a spiral binding instead of a solid binding. For security purposes, a new option allows users to lock up OS/2 automatically on system startup. The WPS also has improved memory usage and improved .INI file handling.

Warning: If you have written Workplace Shell drag and drop code under OS/2 2.0, the way the ancestor classes respond to the `_wpDragOver` method has changed for OS/2 2.1. Specifically, the parent `_wpDragOver` method will return `DOR_NEVERDROP` if the ancestor classes cannot accept the source objects. Consequently, code must be written so that the parent `_wpDragOver` method is called first, and if the resultant `Drop` indicator is *not* `DOR_NEVERDROP`, then the method may continue its processing.

Resource Type	16-bit	32-bit
Points in a polyline	64K (data)	128MB (points)
GPI paths	1400 points	8MB points

Table 1. Comparison of limits of 16-bit and 32-bit graphics engine

Resource Type	32-bit
PS/DC handles	16K
Bitmap handles	64K
Region handles	64K
Font handles	64K

Table 2. 32-bit handle limits

DOS SUPPORT

Several enhancements have been made to DOS support in OS/2. A second thread has been introduced into the DOS sessions to allow for better communications and better synchronization of audio and video in multimedia applications. This function can be enabled with the DOS setting `INT_DURATION_ID`. Another new setting, `DOS_AUTOEXEC`, allows individual `AUTOEXEC.BAT` files to be specified for each DOS or WIN-

OS/2 session. This change lets application developers further customize DOS sessions for test scenarios. Other enhancements include the updating of DPMI to be a subset of the 1.0 standard and MSCDEX audio support in the virtual CD-ROM driver.

WIN-OS/2 3.1

WIN-OS/2 support in OS/2 has been updated to support Windows 3.1 applications. An enhanced compatibility mode also lets many enhanced-mode Windows applications run. However, applications that require a VxD-type device driver will not run. Performance in WIN-OS/2 has been improved, especially the clipboard and DDE functions. WIN-OS/2 3.1 also provides most of the applets shipped with Windows 3.1, and

LinkRight by Rightware Inc.

The only file transfer utility for OS/2.

LinkRight is a parallel port and serial port file transfer utility made especially for OS/2. The OS/2 versions are 32 bit, multi-threaded tasks. For a single low price, you can get a PM version, OS/2 command line version, and DOS version.

Copy files and EAs to/from OS/2 to/from DOS systems.

Subdirectories, hidden, system files, and newer files can be selectively copied.

Batch file generation. Mark and immediately send files, or mark files and save them to a list of files for later transfer.

Features compression for faster transfers. Uses CRC checking to insure accurate transfers. Full support for long file names and HPFS.

Copy the OS/2 operating system from one system to another in accordance with your licensing agreement with IBM.

Compatible with LapLink cables.

For purchasing information call your favorite software distributor or Rightware Inc. (301) 762-1151.

OS/2 is a registered trademark of International Business Machines Corporation. LapLink is a registered trademark of Traveling Software. LinkRight is a copyright of Rightware Inc.

Moving up to OS/2 from Windows? Let CUA Controls Library/2 deliver your GUI baggage.

If you're moving up to more powerful OS/2* from Windows™, now you can have the same user interfaces for both new and existing applications.

With CUA™ Controls Library/2 (CCL/2).

It's a 16-bit program that quickly generates familiar OS/2 2.0 GUIs for applications in OS/2 1.3 and Windows 3.0 and 3.1.

It provides consistent interfaces across platforms, so your end users will quickly learn new applications.

CCL/2 contains the code for seven key GUI screens. It's code you won't have to write. It's also code you can reuse. Over and over again.

What's more, CCL/2 will save you money. Because there are no runtime charges for redistribution of DLLs.

And because now it's available at the low price of only \$149. To order, call 1 800 342-6672.

For information, fax 1 919 469-4423.

So get moving. With CCL/2.



IBM and OS/2 are registered trademarks and CUA is a trademark of International Business Machines Corporation. Windows is a trademark of Microsoft Corporation. © 1993 IBM Corp.

a new function allows OS/2 and DOS applications to be started from within a Windows session. WIN-OS/2 3.1 also includes multimedia extensions as well as TrueType font support, provided in addition to Adobe font support.

LAPTOP SUPPORT

On systems that support the APM (Advanced Power Management) standard, OS/2 2.1 provides power-management capabilities and a power applet to monitor battery status. PCMCIA support is also provided; through socket and card services, PCMCIA cards can be removed and replaced while the machine is on. Interfaces allow device drivers to be allocated dynamically.

MMPM/2

OS/2 2.1 includes Multimedia Presentation Manager/2, a set of extensions that adds multimedia functions to OS/2 including image and audio support, a multimedia control interface, support of various devices, system sounds, and several multimedia applets. Applets included with OS/2 2.1 that were not available with MMPM/2 1.0 are a wave editor and the Ultimotion software motion video player. For more information about MMPM/2, see the References section.

PERFORMANCE IMPROVEMENTS

Extensive improvements have been made to OS/2's performance, including a reduction in the working set of OS/2, HPFS improvements, and scheduler and loader changes that provide faster execution. In addition, there have been changes in some CONFIG.SYS parame-

WIN-OS/2 support in OS/2 has been updated to support Windows 3.1 applications.

ters to improve performance in memory-constrained environments. For detailed information about these changes, consult the *OS/2 2.1 Technical Update Redbook*.

The following APIs have been added to OS/2. For more information, consult the online references available with the toolkit.

DosProtectClose	wpAddPowerViewPage
DosProtectEnumAttribute	wpChangePowerState
DosProtectOpen	wpEjectDisk
DosProtectQueryFHState	wpLockDrive
DosProtectQueryFileInfo	wpPower
DosProtectRead	wpQueryAssociatedProgram
DosProtectSetFHState	wpQueryAutoRefresh
DosProtectSetFileInfo	wpQueryDefStatusView
DosProtectSetFileLocks	wpQueryDriveLockStatus
DosProtectSetFilePtr	wpQueryPowerConfirmation
DosProtectSetFileSize	wpQueryPowerManagement
DosProtectWrite	wpQueryRefreshRate
WinCheckInput	wpSetAssociatedFileIcon
WinLockPointerUpdate	wpSetAutoRefresh
WinLockupSystem	wpSetAutoRefresh
WinQuerySysPointerData	wpSetDefStatusView
WinSetPointerOwner	wpSetPowerConfirmation
WinSetSysPointerData	wpSetPowerManagement
WinUnlockSystem	wpSetRefreshRate
wpAddPowerPage	

Table 3. New OS/2 APIs

FOR PROGRAMMERS ONLY

The MMPM/2 Toolkit is now a part of the base OS/2 Toolkit; it does not need to be

SUMMARY

OS/2 2.1 continues to build on the foundation established by OS/2 2.0, providing a solid base for running and developing applications for DOS, OS/2, and Windows. Strengths include new 32-bit interfaces and the object-oriented System Object Model.

ACKNOWLEDGMENTS

Special thanks to Adam Jollans, Khoa Huynh, and Virginia Roarabaugh for their contributions to this article.

purchased separately. Also, several APIs, shown in Table 3, have been added to OS/2. For more information, consult the online references available with the Toolkit.

REFERENCES

OS/2 2.0 Programming Guide, Vol. 3.
(IBM Doc. S10G-6495-00)

OS/2 2.1 Technical Update Redbook
(IBM Doc. GG24-3948-00)

Parsons, John and Val Enright.
"MMPM/2 and OS/2: The Multi-
media Advantage," *OS/2 Developer*,
Fall 1992: 90-99.

Mike Kaply, IBM Corp., 1000 N.W. 51st St., Boca Raton, Fla. 33431. Kaply has worked for IBM for four years in various capacities, including one year as technical assistant to John Soyring, director of software development programs. He is currently

working on the OS/2 Help subsystem. Kaply holds a B.S. in mathematics and computer science from Southern Methodist University.

32



Do these quotes sound familiar?

"It doesn't crash in the debugger!"

"Exactly, what did you do?"

"I can't reproduce it!"

"Where should I put WinGetLastError?"

"Why does WinDefWindowProc generate an error?"

"It must be a configuration problem!"

Introducing the API Debugger which gives Answers, not Guesses

Error Manager

32-bit
Version 2.0
for OS/2 2.X

Include 1 header file.
Link 1 DLL.
Set 1 environment variable.

- Finds intermittent errors without the Debug Kernel.
- Logs messages to a file, pipe or our PM viewer.
- Works with optimized code in realtime situations.
- Notifies your window procedure or callback function
- Supports both PM and Fullscreen programs
- Enables event-driven error handling
- Invaluable for Debug, QA Test and Production cycles

```

Soft & GUI Stream Viewer
File Edit Options
----- Session started on 3/11/93 at 23:17:34 -----
WinDefWindowProc[] generated error 4610 [0x1202] - Invalid switch handle
in file BASEWINL.C at line 75.
Suspect WinCreateStdWindow[] in file BASEWINL.C at line 103.
Error occurred while processing WM_CREATE [0x0, 0x16CA0FCC]

DosRead[] generated error 109 [0x006D] - Broken pipe
in file PIPEHAND.C at line 21.
This occurred 10 consecutive times.
    
```

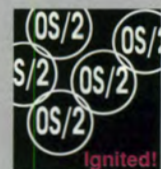
Includes a PM Viewer,
Pointer Validity API and a
Free copy of Command Line

\$225



Soft & GUI

2224 East 21st Street
Brooklyn, New York 11229 (718) 769-8017



SO, YOU HAVE OS/2.



NOW, DO YOU HAVE ENOUGH ROOM TO USE IT?

If you're running OS/2[®], you've probably discovered how much disk space your powerful new operating system demands.

Not to worry.

With Stacker[®] for OS/2 & DOS, you can quickly and safely double the capacity of your hard disk to take full advantage of OS/2's power.

Using Stac[®] Electronics' patented, award-winning Stacker LZS[™] technology, Stacker instantly and transparently compresses all your data, storing it more efficiently so you have more room to work.

That's all there is to it! You can install Stacker in minutes and access all your compressed files at any time, whether you boot from OS/2 or DOS.

And, with the Stacker Optimizer[™] you can quickly defragment your Stacker drives to get the best possible performance. Stacker works on disks as large as 1 gigabyte, giving you up to 2 gigabytes of disk capacity. And, it even comes with a simple Unstack command that returns your system to its original, uncompressed state.

New Stacker for OS/2 & DOS. Think of it as adding space to your OS/2 workplace.

CALL NOW FOR MORE INFORMATION
1-800-522-STAC, ext. 8207
or 619-431-7474
Fax 619-431-9616

STAC[®]

OS/2 2.1 supports many popular CD-ROM drives and enhances application compatibility for DOS CD-ROM applications. A layered CD-ROM device driver design illustrates how SCSI and non-SCSI attached CD-ROM drives are supported. **BY FRANK SCHROEDER and RICK EFRUSS**

CD-ROM Support in OS/2 2.1

The popularity of CD-ROM drives has increased significantly in the past year and the trend is expected to continue. This increased popularity can be attributed to several factors—increasing numbers of CD-ROM software titles, the continuing evolution of multimedia, and the increased use of CD-ROM discs as a convenient storage medium for large quantities of data. Computer software manufacturers have increased their use of CD-ROM discs as a cost-effective means of distributing software products and technical information to customers.

With the growing popularity of CD-ROM, emphasis has been placed on enhancing CD-ROM support in OS/2. While in OS/2 2.0, CD-ROM support is limited in the number of drives supported and the level of compatibility with DOS applications, OS/2 2.1 support has been enhanced. OS/2 now supports many popular CD-ROM drives; the level of DOS CD-ROM application compatibility has also been improved.

This article describes the CD-ROM enhancements in OS/2 2.1. A layered CD-ROM device driver model is described; topics ranging from DOS application compatibility to installation are presented, and a list of supported CD-ROM drives is included.

OS/2 2.0

In OS/2 2.0, support for CD-ROM hardware and software is limited. The CD-ROM device driver (CDROM.SYS) supports only IBM and IBM-compatible CD-ROM drives by Toshiba. Drives from other manufacturers are not supported.

Users with unsupported CD-ROM drives can

work around this limitation with the OS/2 VM boot feature, accessed with the "DOS from Drive A:" icon in the Command Prompts folder. The VM boot feature allows the user to boot a native version of DOS in a full-screen session under OS/2. The booted DOS session uses the native DOS CD-ROM device driver that ships with the CD-ROM drive. Access to the CD-ROM drive is possible only through the booted DOS session; the drive cannot be accessed from an OS/2 or WIN-OS2 session.

The virtual CD-ROM device driver (VCDROM.SYS) in OS/2 2.0 supports only some of the programming interfaces defined in the DOS CD-ROM Extensions (MSCDEX). Although many DOS CD-ROM applications work properly in OS/2 2.0, those that use unsupported DOS CD-ROM extensions will fail. (The terms "DOS CD-ROM Extension" and "Microsoft CD-ROM Extension" [MSCDEX] are used interchangeably and refer to the same set of programming interfaces.)

OS/2 2.1

OS/2 2.1 offers more extensive support for CD-ROM drives and CD-ROM applications. Device driver support is provided for most of the popular SCSI-attached CD-ROM drives. Figure 1 shows a complete list of supported CD-ROM models.

The CD-ROM device drivers in OS/2 2.1 fully exploit the capabilities of each CD-ROM drive. Data and audio support are provided for all models. CD-ROM XA support is provided for those models that support the CD-ROM XA storage standard, which enables the interleaving of audio, video, and standard file system data. It is the storage format used in Kodak's Photo CD technology,



Frank Schroeder



Rick Efruss

Manufacturer	Model	Interface	OS/2 CD-ROM Device Drivers	
Hitachi	CDR-1650S	SCSI-1	OS2CDROM.DMD	HITCDS1.FLT
	CDR-1750S	SCSI-1	OS2CDROM.DMD	HITCDS1.FLT
	CDR-3650	SCSI-1	OS2CDROM.DMD	HITCDS1.FLT
	CDR-3750	SCSI-2	OS2CDROM.DMD	
IBM	CD-ROM I	SCSI-1	OS2CDROM.DMD	TOSHCD1.FLT
	CD-ROM II	SCSI-2	OS2CDROM.DMD	
NEC	CDR-25	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-36	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-37	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-38	SCSI-2	OS2CDROM.DMD	
	CDR-72	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-73	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-74	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-74-1	SCSI-2	OS2CDROM.DMD	
	CDR-82	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-83	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-84	SCSI-1	OS2CDROM.DMD	NECCDS1.FLT
	CDR-84-1	SCSI-2	OS2CDROM.DMD	
Panasonic	CR-501	SCSI-2	OS2CDROM.DMD	
	LK-MC501S	SCSI-2	OS2CDROM.DMD	
Pioneer	DRM-600	SCSI-1	OS2CDROM.DMD	
	DRM-604X	SCSI-2	OS2CDROM.DMD	
Sony	CDU-541	SCSI-2	OS2CDROM.DMD	
	CDU-561	SCSI-2	OS2CDROM.DMD	
	CDU-6111	SCSI-1	OS2CDROM.DMD	SONYCDS1.FLT
	CDU-6211	SCSI-2	OS2CDROM.DMD	
	CDU-7211	SCSI-2	OS2CDROM.DMD	
Texel	DM-3021	SCSI-1	OS2CDROM.DMD	SONYCDS1.FLT
	DM-3024	SCSI-2	OS2CDROM.DMD	
	DM-5021	SCSI-1	OS2CDROM.DMD	SONYCDS1.FLT
	DM-5024	SCSI-2	OS2CDROM.DMD	
Toshiba	XM-3201	SCSI-1	OS2CDROM.DMD	TOSHCD1.FLT
	XM-3301	SCSI-2	OS2CDROM.DMD	
	XM-3401	SCSI-2	OS2CDROM.DMD	

Figure 1. CD-ROM drives supported by OS/2 2.1

which stores photographic images on a CD-ROM disc.

To increase DOS CD-ROM application compatibility, OS/2 supports the full set of application programming interfaces defined by Microsoft CD-ROM Extensions 2.21. A new DOS setting, `INT_DURING_ID`, allows faster performance of DOS multimedia applications.

A layered CD-ROM device driver model simplifies device driver develop-

ment for both SCSI and non-SCSI CD-ROM drives. Several CD-ROM manufacturers use this model to develop OS/2 device drivers that support their popular non-SCSI CD-ROM drives.

A system view of OS/2 2.1 CD-ROM components and their relationships is illustrated in Figure 2. The following sections give an overview of the CD-ROM subsystem and the responsibilities of each component.

The Virtual CD-ROM Device Driver. The virtual CD-ROM device driver (`VCDROM.SYS`) provides the compatibility layer for DOS applications that require CD-ROM API services. It consists of two parts: a generic OS/2 virtual device driver (an operating system extension responsible for separating the activity of multiple DOS sessions) and a DOS device driver. The virtual device driver model consists of a dynamic link

library with separate instance data for each DOS session.

The virtual device driver portion of VCDROM.SYS is responsible for servicing the DOS CD-ROM Extensions software interrupt (INT 2Fh) interface. DOS applications use the AH register for the DOS CD-ROM Extension function code (15h) and the AL register for the command code, as shown in Figure 3. Parameters to the DOS CD-ROM Extensions API are passed in the general-purpose registers. (This API is described in further detail in the *CD-ROM Programmer's Guide for MS-DOS CD-ROM Extensions 2.21*.)

The DOS CD-ROM Extensions enable DOS applications to determine the number of CD-ROM drives installed; to navigate the volume, path, and directory structures; and to manage the files stored on a CD-ROM disc. The virtual device driver routes the DOS CD-ROM Extensions request to the OS/2 CD-ROM file system.

The DOS device-driver portion of VCDROM.SYS handles DOS INT 21h file system requests to the CD-ROM drive. All requests—for example, reading a file from a CD-ROM disc—are routed from the DOS device driver to the OS/2 CD-ROM file system. The virtual CD-ROM device driver passes requests to the OS/2 CD-ROM file system using the virtual device helper services.

Additional information on the virtual CD-ROM device driver and the virtual device helper services can be found in the *Virtual Device Driver Reference* of the OS/2 Technical Library.

The CD-ROM File System. The OS/2 CD-ROM file system (CDFFS.IFS) is a file system driver that understands how data is stored on the CD-ROM disc and uses the installable file system mechanism to attach to and receive requests from the OS/2 file system. The primary method of storage and retrieval, the High Sierra/ISO-9660 format, specifies precisely how volume-, directory-, path-, and file-specific information is arranged on the media.

CDFFS.IFS uses the file system helper services to pass data and audio com-

mands to the CD-ROM device manager. The helper services convert each requested operation into a device driver request packet that is passed to the CD-ROM device manager. For example, if CDFFS.IFS receives a request to read data from a file, it converts the file-system

ves that comply with the American National Standards Institute (ANSI) SCSI-2 standard X3T9.2/86-109 (SCSI-2 draft of proposed ANSI Revision 10g). It provides generic data and audio support for drives that support the command set specified in the

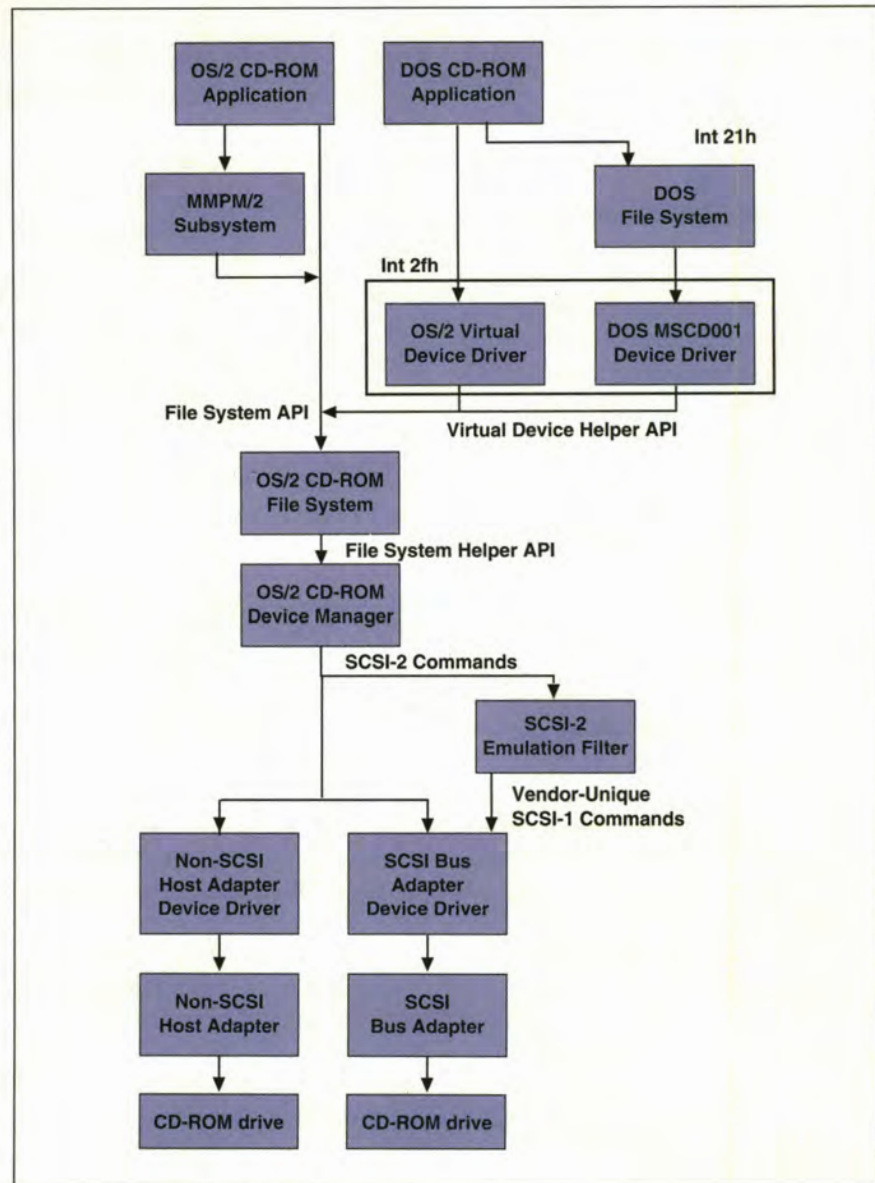


Figure 2. OS/2 2.1 CD-ROM component hierarchy

read request into the appropriate physical sector read request, which is then processed by the device manager.

The CD-ROM Device Manager. The OS/2 CD-ROM device manager (OS2-CDROM.DMD) is a generic block-device driver for CD-ROM dri-

standard. Vendor-unique CD-ROM XA support and multisession Kodak Photo CD support are provided for selected drive models.

The CD-ROM device manager provides a uniform interface between its clients and adapter device drivers. Clients of the device manager include:

- OS/2 applications
- Multimedia Presentation Manager/2 (MMPM/2) applications
- The virtual CD-ROM device driver
- The CD-ROM installable file system.

OS/2 applications and the MMPM/2 subsystem communicate with the device manager using the OS/2 file system API and Categories 80 and 81 IOCTL services, as shown in Figure 4.

DOS applications communicate indirectly with the CD-ROM device manager, going through the virtual CD-ROM device driver. VCDROM.SYS converts DOS CD-ROM extensions and DOS file system requests into an OS/2 file system (or category 80 or 81 IOCTL) request, which is then routed to the CD-ROM device manager using the virtual device helper services.

The CD-ROM file system communicates with the CD-ROM device manager using the request packet interface defined in the *OS/2 2.0 Physical Device Driver Reference* manual.

The interface between the device manager and adapter device drivers adheres to the adapter-device driver interface defined in the *OS/2 2.0 Storage Device Driver Reference Manual* specification. The device manager converts a request from its client into a SCSI-2 command descriptor block and routes the SCSI-2 command to the specified adapter device driver.

The device manager is an installable block-device driver loaded with a DEVICE= statement in CONFIG.SYS. It replaces CDROM.SYS, the CD-ROM device driver shipped in OS/2 2.0.

SCSI-2 Emulation Filters. A SCSI CD-ROM target device with vendor-unique commands not supported in the SCSI-2 standard requires a SCSI-2 emulation filter (.FLT), as shown in Figure 1. The emulation filter maps SCSI-2 commands received from the device manager to the vendor-unique commands supported by the target device. This support is required to enable audio sup-

Command	Function Description
00h	Get number of CD-ROM drive letters
01h	Get CD-ROM drive device list
02h	Get copyright file name
03h	Get abstract file name
04h	Get bibliographic file name
05h	Read VTOC
06h	Reserved
07h	Reserved
08h	Absolute disc read
09h	Absolute disc write
0ah	Reserved
0bh	CD-ROM drive check
0ch	MSCDEX version
0dh	Get CD-ROM drive letters
0eh	Get/set volume descriptor preference
0fh	Get directory entry
10h	Send device request
11h-0FFh	Reserved

Figure 3. MSCDEX Version 2.21 command codes and function descriptions

port on CD-ROM drives that adhere to the SCSI-1 standard because the SCSI-1 standard does not define a standard command set for audio control.

A SCSI-2 emulation filter is required for each vendor-unique CD-ROM drive. Typically, a CD-ROM manufacturer uses the same vendor-unique command set for all its CD-ROM drives; one filter driver is therefore required for each manufacturer.

The filter driver receives SCSI-2 commands from the CD-ROM device manager, converts the command to its vendor-unique equivalent, and routes the filtered command to the SCSI adapter device driver. If data returned with the command needs to be filtered, the filter driver regains control when the request is complete, converts the outgoing data to its SCSI-2 equivalent, and returns to the device manager. The filtering process is transparent to the device manager and the adapter device driver.

SCSI Adapter Device Drivers. A SCSI adapter device driver (.ADD) contains the hardware-specific modules required to program the SCSI adapter that con-

trols the CD-ROM drive. An OS/2 SCSI adapter device driver is needed to support a SCSI-attached CD-ROM drive in OS/2. The generic INT 13 device driver (IBMINT13.I13), which can be used to support a SCSI attached fixed disc, cannot be used to support a CD-ROM drive.

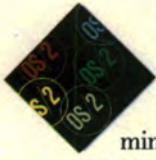
The SCSI-2 emulation filter and the SCSI adapter device driver adhere to the adapter-device driver interface defined in the *OS/2 2.0 Storage Device Driver Reference Manual*.

OS/2 2.1 comes with device drivers for SCSI adapters from IBM, Adaptec, DPT, and Future Domain. In addition, OS/2 device drivers for SCSI adapters from Always Technology, BusLogic, MediaVision, Mylex, Trantor, and UltraStor are available from either the manufacturer or third-party developers.

Non-SCSI CD-ROM Adapter Device Drivers. Several leading CD-ROM drive manufacturers use proprietary, non-SCSI, host adapter interfaces for CD-ROM drives. To support a non-SCSI CD-ROM drive, the adapter device driver must emulate a SCSI-2 target device. This enables the

Upgrade to

C Set ++ for only a



Here's a C Set offer with lots of pluses. From IBM Programming Systems. Because for only \$149, you can upgrade from Work Set/2 or C Set/2 to the new C Set++,[™] the most complete object-oriented application package you can buy for OS/2.[®]

Its 32-bit C/C++ compiler lets you unleash all the power of OS/2 — giving you industrial strength code for your mission critical applications.

It has an extraordinary code optimizer with a full set of options. Even a switch to optimize for the new Pentium[™] processor. Plus a full set of class libraries, including application frameworks for PM, container classes and classes for multitasking, streams and more.

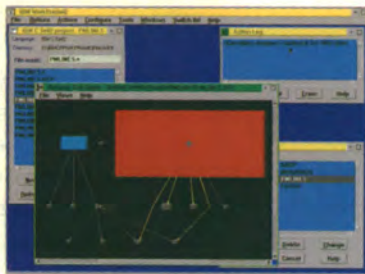


C Set ++

Plus, there's a full complement of other helpful features. Such as an interactive source level debugger. And the unique Execution Trace Analyzer traces the execution of a program, then graphically displays diagrams of the analysis.

Plus, you get Workframe/2,[™] a language-independent tool that lets you customize your own environment.

C Set ++ Technical Features	
Standards	ANSI C X3.159-1989
	NIST validated
	ANSI C++ X3J16 (Full ARM)
	ISO 9899:1990
Optimization	Global
	Inter-module
	Function inlining
	Instruction scheduling



C note +

To order your upgrade at the low price of only \$149, contact your nearest dealer or call **1-800-342-6672** (USA) or **1-800-465-7999** ext. 461 (Canada). But hurry. This offer expires August 31, 1993.



DOS		OS/2		IOctl/Command Description
Category	Function	Category	Function	
02h	00h	***(7)		Return address of device header
02h	01h	80h	70h	Return location of head
02h	04h	81h	60h	Return audio channel information
02h	06h	80h	60h	Return device status
02h	07h	80h	63h	Return sector size
02h	08h	80h	78h	Return volume size
02h	09h	***(8)		Return changed media code
02h	0ah	81h	61h	Return audio disc information
02h	0bh	81h	62h	Return audio track information
02h	0ch	81h	63h	Return audio subchannel Q information
02h	0eh	80h	79h	Return UPC code
02h	15h	81h	65h	Return audio status information
03h	00h	80h	44h	Eject disc
03h	01h	80h	46h	Lock/unlock door
03h	02h	80h	40h	Reset drive
03h	03h	81h	40h	Audio channel control
	***(1)	80h	61h	Identify CD-ROM driver
	***(2)	80h	72h	Read long
	***(3)	80h	50h	Seek
	***(4)	81h	50h	Play audio
	***(5)	81h	51h	Stop audio
	***(6)	81h	52h	Resume audio

Notes

1. Command is not available in DOS.
2. DOS extended device driver command 130h
3. DOS extended device driver command 131h
4. DOS extended device driver command 132h
5. DOS extended device driver command 133h
6. DOS extended device driver command 136h
7. Command is not available in OS/2 2.1
8. Corresponds to OS/2 block device driver command 01h

Figure 4. DOS to OS/2 2.1 CD-ROM IOctl mapping

CD-ROM device manager to issue a common command set to its target devices, regardless of whether a target device directly supports the SCSI-2 command set or the attached host adapter is SCSI or proprietary. The incoming command and the return sense data must be emulated by the driver.

Non-SCSI CD-ROM adapter device drivers adhere to the adapter-device driver interface defined in the *OS/2 2.0 Storage Device Driver Reference Manual* specification.

CD-ROM GENERIC IOCTLS

OS/2 provides an API called device

I/O control (*DosDevIOctl*), which enables applications to communicate directly with OS/2 device drivers. The OS/2 IOctls are specified by a category code and a function code. For CD-ROM support, the category 80h IOctls control data access to the drive, while category 81h IOctls control audio features of the drive. These IOctls are routed from the application to the CD-ROM device manager. The CD-ROM IOctls are further described in the *IBM PS/2 CD-ROM II Technical Reference* manual.

The DOS operating system provides an equivalent application programming interface called I/O Control (IOctl) so

DOS applications can communicate directly with device drivers. The INT 21h function 44h call is translated by the DOS device driver portion of the virtual CD-ROM device driver to a virtual device helper, which is then routed to the CD-ROM device manager. Additional information about the DOS IOctl interface can be found in Appendix C of the *DOS 5.0 Technical Reference* manual.

The DOS IOctls are specified as belonging to either the read (02h) or write (03h) categories and are further subdivided into function codes, which correspond to the activity that the application is requesting of the device

PVCS 5.1
Now Available

Jim Besemer
Director of Research & Development
INTERSOLV

"WE BUILT THREE VERSIONS OF OUR APPLICATION TO RUN ON EIGHT PLATFORMS. THERE'S NO WAY WE COULD HAVE GOT THAT PRODUCT OUT WITHOUT SOFTWARE CONFIGURATION MANAGEMENT. I'D USE PVCS EVEN IF I DIDN'T WORK HERE."

If you're facing an assignment as tough as Jim's, you're ready for The PVCS Series.

If your development team has more than one member, your application has more than one module or you're working on a LAN—you need automated software configuration management.

The PVCS Series is your best choice for LAN-based Software Configuration Management (SCM). We'll help you automate manual processes and avoid the headaches that multi-version, multi-module applications can cause.

You can skip the lost data, the bug fixes that don't stay fixed, the builds that include outdated modules, overwritten code and the frantic midnight phone calls about wrong versions that somehow made their way into production.

Instead of wasting time wrestling with your development environment, you can concentrate on building quality applications, regardless of what language, programmer workbench, methodology or operating system you're using.

PVCS eliminates tedious housekeeping details and automates builds so they are consistent and error free.

PVCS gives you the security of project-wide undo and the confidence to create good code, knowing you've got a complete audit trail to show where you've been—and help plan where you're going.

Operating seamlessly across Windows, NT, MS-DOS, OS/2 and a variety of UNIX environments, PVCS is flexible enough to fit your development style and the most complex new applications.

The PVCS Series is the market leader in SCM for the LAN. It's a complete family of products, designed for the widest variety of distributed client/server development architectures and operating systems.

The PVCS Series includes: PVCS Version Manager, PVCS Configuration Builder, PVCS Production Gateway, PVCS Developer's Toolkit and PVCS Reporter.

Put the power of The PVCS Series to work for you. Call for a free demo disk, evaluation, or a copy of the whitepaper, "Software Configuration Management: Choosing The Correct Interface".



The PVCS Graphical User Interface makes it easy to apply the power of PVCS in your distributed development environment.

CALL 800-547-PVCS EXT. 40

SOFTWARE CONFIGURATION MANAGEMENT FOR HETEROGENEOUS LAN-BASED DEVELOPMENT

INTERSOLV

driver. Figure 4 illustrates the mapping of DOS CD-ROM IOCTLs to their corresponding OS/2 equivalents.

INSTALLATION

Installation of CD-ROM device drivers has been simplified in OS/2 2.1. During the initial installation of OS/2, the installation program automatically detects the presence of the CD-ROM drive and SCSI adapter and installs the required device drivers. The product name of the detected CD-ROM drive and SCSI adapter is displayed, providing visual feedback to the user.

Users who add a CD-ROM drive to their system after OS/2 has been installed can use the Selective Install feature to install the required device drivers.

After installation, the CONFIG.SYS file will contain the following statements:

```
DEVICE=C:\OS2\MDOS\VCDROM.SYS
IFS=C:\OS2\CDFS.IFS /Q
BASEDEV=IBM2SCSI.ADD
DEVICE=C:\OS2\OS2CDROM.DMD /Q
```

In these statements, VCDROM.SYS represents the virtual CD-ROM device driver for DOS application support and CDFS.IFS represents the CD-ROM file system driver. IBM2SCSI.ADD is the adapter device driver for the IBM PS/2 SCSI Adapter. (This statement varies depending on the SCSI adapter installed.) OS2CDROM.DMD is the OS/2 CD-ROM device manager.

The adapter device driver is loaded with BASEDEV=, while the CD-ROM device manager is loaded with the DEVICE= statement. For CD-ROM drives that use the SCSI-1 interface, a BASEDEV= statement is also required to load the SCSI-2 emulation filter. (Figure 1 gives a complete list of supported CD-ROM drives and their associated device drivers.)

DOS SETTINGS FOR CD-ROM APPLICATIONS

In OS/2 2.0, one thread is used in each DOS session for all activity, including hardware interrupt simulation. This sin-

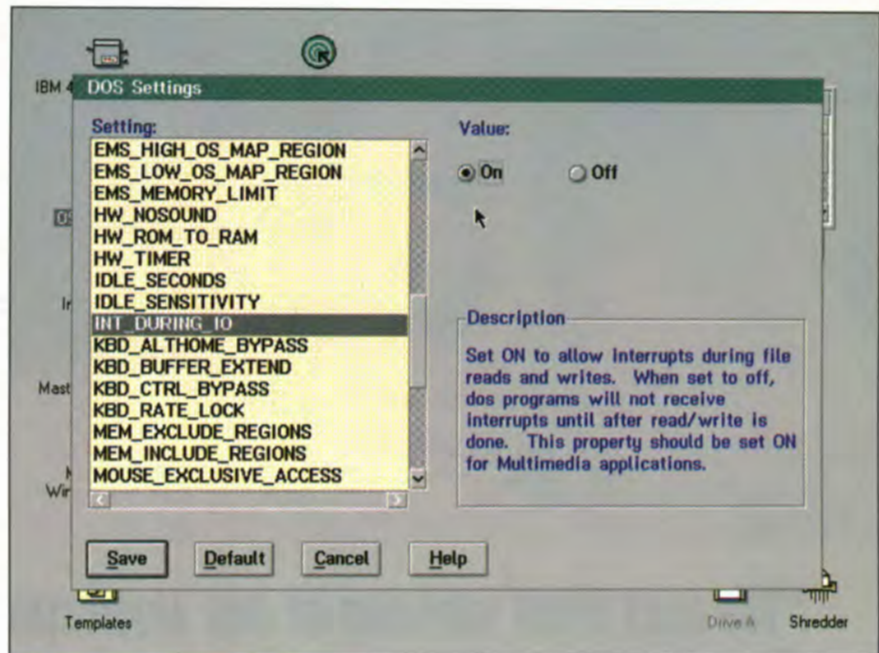


Figure 5. INT_DURING_IO DOS setting

gle-threaded implementation limits the performance of DOS multimedia applications. The limitation is noticeable in DOS applications that require synchronized audio and video; the typical result is breakup of the audio or video stream. A breakup occurs when the single thread is blocked in the device driver while waiting for a read of the CD-ROM to complete. When blocked, the thread cannot process the simulated hardware interrupts of the other stream.

OS/2 2.1 adds a new DOS setting, INT_DURING_IO, which alleviates this synchronization problem, as shown in Figure 5. When INT_DURING_IO is set to ON, OS/2 2.1 creates a second thread so that hardware interrupt simulation can occur concurrently with input and output. The result is smoother audio and video playback for DOS multimedia applications. Because the INT_DURING_IO DOS setting adds some overhead to the system, it should be activated only when needed to make a particular application run successfully.

KODAK PHOTO-CD SUPPORT

Kodak recently introduced its Photo CD technology, which enables users to store photographic images on a CD-ROM

disc. To support the Photo CD standard, a CD-ROM drive must be able to read CD-ROM XA formatted sectors.

Multisession Photo CD allows users to add more sets of photographs to a CD containing an existing set. In OS/2, multisession Photo CD is supported for the following CD-ROM drives:

- Sony CDU-561
- Toshiba 3401
- Telex 3024/5024 (with Telex Photo CD firmware upgrade)
- Pioneer DRM-604X (with Pioneer Photo CD firmware upgrade).

Single-session Photo CD restricts the storage of images to a single set; no photos can be added after the initial set. Single-session Photo CD is supported in OS/2 2.1 for these CD-ROM drives:

- CD Technology Porta-Drive T3301
- Hitachi CDR-3750
- IBM CD-ROM-II
- Sony CDU-541, 6211, 7211
- Telex 3024, 5024
- Toshiba 3301
- NEC Multispin 38, 74-1, 84-1.

REFERENCES

- OS/2 2.0 Technical Library, *Physical Device Driver Reference* (IBM Doc. S10G-6266-00)
- OS/2 2.0 Technical Library, *Virtual Device Driver Reference* (IBM Doc. S10G-6310-00)
- IBM OS/2 2.0 Storage Device Driver Reference Manual (IBM Doc. S71G-1897-00)
- IBM PS/2 CD-ROM II Technical Reference (IBM Doc. S10G-3359-00)
- OS/2 2.1 Technical Update Red Book (IBM Doc. GG24-3948-00)
- DOS 5.0 Technical Reference (IBM Doc. S91F-8614-00)
- CD-ROM Programmer's Guide for MS-DOS CD-ROM Extensions Version 2.21, Microsoft Corp.
- ANSI X3.131 Specification
- ANSI Small Computer System Interface - 2 (SCSI-2) Specification (X3T9.2/86-109 Revision 10g)

Rick M. Efruss, IBM Personal Software Products Division, 1000 N.W. 51st St., Boca Raton, Fla. 33431. Efruss is the CD-ROM development lead in the OS/2 device drivers and multiple virtual DOS machines department. He has worked in OS/2 development since 1986. Efruss holds a B.S. from the University of Virginia and an M.S. from the Georgia Institute of Technology, both in computer science.

Frank J. Schroeder, IBM Personal Software Products Division, 1000 N.W. 51st St., Boca Raton, Fla. 33431. Schroeder is a staff programmer in the OS/2 device drivers and multiple virtual DOS machines department. He has published articles in OS/2 2.x Notebook, OS/2 Developer, and IBM Personal Systems Technical Solutions. He earned a bachelor of technology degree in computer science from Rochester Institute of Technology, New York, and an M.B.A. from the University of Miami, Fla.

GREEN-KEEPER

(noun) 1. One who keeps the environment green. 2. A series of clip-and-save tips for keeping your business and home environments green.

1

How to help
your company
go green.

From the boardroom to the mailroom to the lunchroom to the washroom, there may be more ways than you think to help your company become environmentally responsible:

- Recycled office paper
- In-house recycling programs
- Reduced or recycled product packaging
- Glassine or open window envelopes
- Compact fluorescent light bulbs
- Company coffee mugs
- Weekly departmental newsletters instead of daily memos

But how can you help your company make these changes? Where

can you find energy-saving, recyclable, and recycled materials? How can your company recycle its own waste? And how can you do it at little or no expense—or even *save* money?

Try some of the suggestions in *Keeping Your Company Green*, written by Stefan Bechtel and the editors of Rodale Press. This how-to guidebook is packed with "simple, doable steps your company can take to help keep itself—and the world—green." The book also includes a Directory of Green Products and Services. For more information, contact Rodale Press at 1-800-441-7761.

Permission granted by Rodale Press, Inc.

The Green-Keeper series is sponsored by the MFI Green Project of Miller Freeman Inc.



This article describes enhancements to printing performance in OS/2 2.1 with emphasis on the OS/2 LaserJet printer driver. We will compare and discuss new performance options in the LaserJet driver: printer memory usage, large buffers, printer hardware patterns, and HP-GL/2. **BY SHIXIONG YANG and MONTE COPELAND**

Print Performance Options in OS/2 2.1 Printer Drivers



Shixiong Yang



Monte Copeland

Three things affected printing performance under OS/2 2.0: the addition of the 32-bit graphics engine component (PMGRE.DLL), advances in printer hardware, and changes to 16-bit OS/2 Presentation Manager (PM) print drivers. This article describes changes in OS/2 2.1 that affect printing, with special emphasis on the OS/2 LaserJet printer driver. It compares new performance options in the LaserJet driver: printer memory usage, large buffers, printer hardware patterns, and HP-GL/2.

CHANGES AFFECTING PRINTING PERFORMANCE

32-Bit Graphics Engine Component. The OS/2 graphic engine component (PMGRE.DLL or simply GRE) has been converted to 32-bit in OS/2 2.1 and was first introduced in the Service Pak (previously referred to as the corrective service disk) for OS/2 2.0.

The new GRE allows OS/2 to enable 32-bit PM device drivers while retaining compatibility with existing 16-bit PM device drivers. In OS/2 2.1, most PM video drivers are 32-bit, while most PM printer drivers are 16-bit.

Because of this, printing takes some performance hits because of "thinking." A thinking process is a short piece of code that allows 32-bit functions to call 16-bit worker routines and vice versa. The GRE uses thunks in OS/2 2.1 to retain compatibility with 16-bit PM printer drivers—but at a cost: executing thunk code takes additional time. For example, a commonly run test case caus-

es the GRE to call the driver function `FillPath` over 600,000 times for a single page. Each call must execute thunk code.

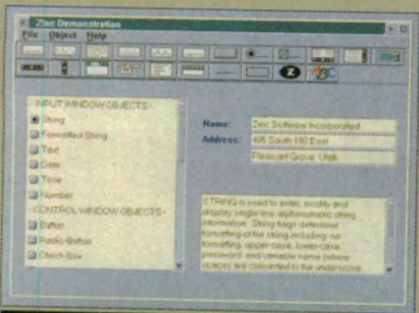
Most laser printers have 300 dots-per-inch (dpi) resolution, but new models boast 600 dpi. This twofold increase in resolution means a fourfold increase in raster data that the GRE, the printer drivers, and the OS/2 kernel must generate and send to the printer. To store a letter-sized monochrome bitmap at 300 dpi, for example, takes about 1MB of memory; at 600 dpi it takes about 4MB of memory.

New Performance Optimization. In OS/2 2.1, the GRE, spooler, and 16-bit PM printer drivers have been optimized to improve printing performance. In the past, the interaction between the GRE and PM printer drivers was not a problem since both GRE and drivers were 16-bit. Now, however, reducing the interaction becomes an important issue. To improve performance, we used dynamic GRE dispatch tables. We also found opportunities in the GRE, the spooler, and PM printer drivers to cache memory rather than freeing and reallocating it later. In several cases, this improved spooling time by as much as 10% to 25%.

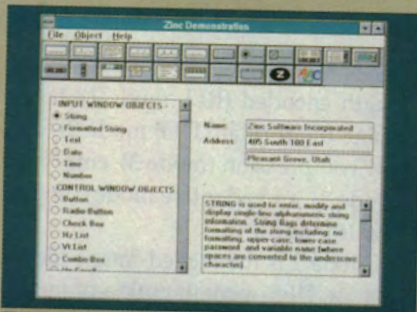
Dynamic GRE Dispatch Tables. One new use of the GRE is for dynamic dispatch table support. The GRE dispatch table (essentially a jump table) is an array of function pointers that, by default, point to functions in the GRE. Device drivers change the table and point to functions within the driver, an activity known as "hooking the engine." Previous-


 THINK
ZINC

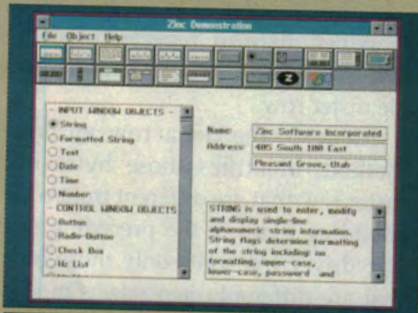

UNIX MOTIF



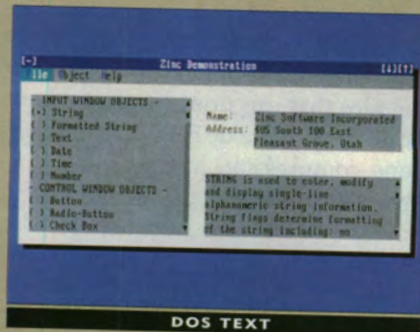
OS/2 2.0



MS WINDOWS & WINDOWS NT



DOS GRAPHICS



DOS TEXT

ZINC™ APPLICATION FRAMEWORK™ 3.5

Multiplatform Flexibility From One Application Framework.

FLEXIBILITY is an essential component in software design. Your development tools should allow you to easily incorporate new features and technologies into your applications without rewriting all of your code. That's a tall order if your development tools are designed like a straight-jacket. Zinc Application Framework 3.5 and Zinc Designer™ give you flexible and extendible support for Microsoft Windows, Windows NT, OS/2 2.0, UNIX Motif, DOS Graphics and DOS Text. And Zinc does it with ONE set of source code. Zinc's multiplatform, object-oriented architecture won't confine your application development options today... or tomorrow.

FOR A FREE ZINC DEMO KIT, CALL US TOLL FREE TODAY AT
1.800.638.8665. IN EUROPE CALL +44 (0) 81 855 9918

Z i n c

ZINC SOFTWARE INCORPORATED, 405 SOUTH 100 EAST, 2ND FLOOR, PLEASANT GROVE, UTAH 84062, TEL 801.785.8900, FAX 801.785.8996, BBS 801.785.8997.

EUROPE: ZINC SOFTWARE (UK) LIMITED 58-60 BERESFORD STREET, LONDON SE18 6BG, TEL +44 (0) 81 855 9918, FAX +44 (0) 81 316 7778, BBS +44 (0) 81 317 2310

© COPYRIGHT 1992 ZINC SOFTWARE INCORPORATED, ALL RIGHTS RESERVED. ZINC, ZINC DESIGNER AND ZINC APPLICATION FRAMEWORK ARE TRADEMARKS OF ZINC SOFTWARE INCORPORATED. OTHER TRADEMARKS ARE OWNED BY THEIR RESPECTIVE COMPANIES.

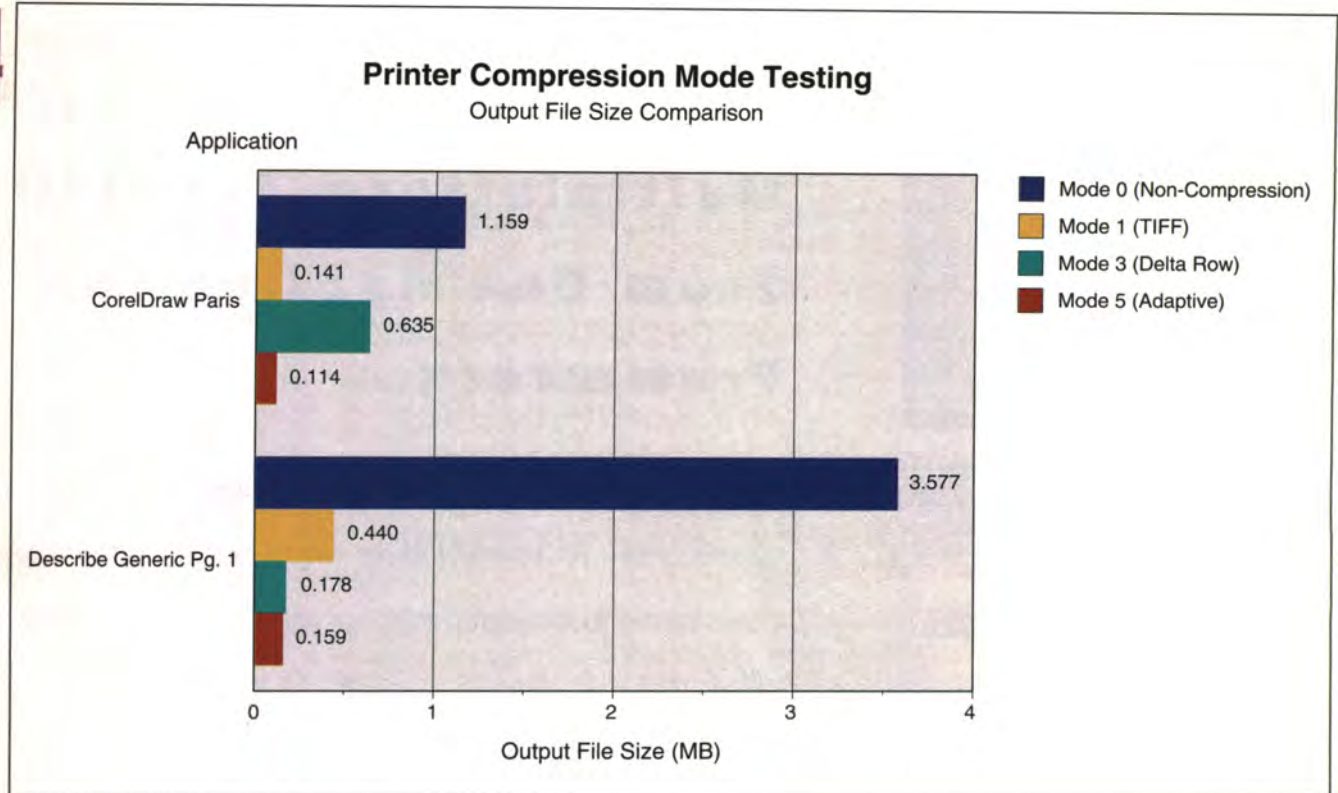


Figure 1. Printer compression mode

ly, a device driver had only one chance to hook the engine. With dynamic dispatch tables, however, a PM driver can now selectively unhook the GRE functions based on its initial settings (job properties, device context type, and so on) when a new printer device context is enabled. Reducing the number of hooked functions reduces the number of thanks to execute; thus, printing performance improves.

PERFORMANCE OPTIONS IN THE PM PRINTER DRIVERS

In the printer properties dialogue, the PM LaserJet driver offers printer memory and printer page protection selections. It also offers four different performance options in the job properties dialogue of the driver. The printer object presents both dialogues, while most applications present the job properties dialogue before printing.

If these options improve performance, it is logical to ask, why not enable them all the time? The answer is that some options may not improve performance every time; it depends on

the application program and the content of the print job.

Printer Memory Selection and Compression Mode. In OS/2 2.1, the LaserJet driver needs to know exactly how much printer memory is available for compressing raster data. On the printer properties dialogue, specify the amount of memory installed in your printer and the page protection value. The memory amount should include standard and expanded memory. If you are unsure about these values, print the self-test page from the printer and enter the values displayed there into the printer properties dialogue box.

The amount of available printer memory affects how the LaserJet driver compresses raster data. PCL5-capable LaserJet printers allow several modes of data compression; the driver chooses the optimal mode of compression based on printer capabilities, available printer memory, and the nature of output data.

The modes of raster data compression are uncompressed (mode 0), run-

length encoded (RLL mode 1), tagged image file format (TIFF mode 2), delta row compression (mode 3), empty row (mode 4), and adaptive mode compression (mode 5).

Mode 2 is the tagged-image file format, or TIFF, a combination of mode 0 (unencoded) and mode 1 (RLL encoded). The driver selects mode 2 encoding when there are repeated or similar bytes adjacent to one another in a single raster row.

Mode 3, the delta row encoding method, identifies those bytes in a raster row that are different from corresponding bytes in the preceding row. The driver transmits only those bytes that are different, then selects mode 3 encoding when there are similar adjacent raster rows.

Mode 5, adaptive compression mode, is a mixture of all compression modes that can print empty or duplicate rows. In mode 5, the driver selects one compression mode per row but bundles up to 32K of rows into a single mode 5 block. Currently, only Hewlett Packard's LaserJet 4 printers support

Install With Ease, Install With Style,
Install With The Best

Software Installer

for OS/2 2.0 & 2.1

Why spend your valuable time developing the installation processes for your applications when Software Installer makes the job much easier? You can use its extensive services to install, update, restore, and delete your products. End users can install your applications from diskette, CD ROM, Local Area Networks (LAN), or from MVS, OS/400, VM or VSE host systems.

Software Installer provides routines for:

- Creating animated Presentation Manager™ installations
- Updating the CONFIG.SYS file
- Adding, deleting, and changing OS/2 files, INI files and Workplace Shell™ classes and objects

Many other features are included, such as:

- Multi-threaded processing
- Disk generation utility
- Open and programmable interfaces
- Response file invocation for non-interactive installations

For information on Software Installer for Windows™ call
1-800-IBM-CARY

List \$349

Software Installer



Developed by
IBM Programming Systems

**Now Creates
CID Enabled
Install Programs!**

for more information call:

Call 1-800-IBM-CARY

to order call either:

Indelible Blue 1-800-776-8284

Business Depot 1-800-844-8448

Let Distributed Application/2
Do The Talking!

Distributed Application/2™

Distributed Application/2 is a set of application programming interfaces (APIs) that provide a consistent way to access interprocess and network communication functions under OS/2 2.0, making client/server applications easier to create than ever. Distributed Application/2 and its APIs:

- Will help you compete in the growing client/server programming arena
- Provide an API set for interprocess and network communications
- Hide the complexity of the supported protocols
- Enable you to select a protocol at run time, without changing your code
- Allow you to focus on your application, and leave the client/server communication to Distributed Application/2

Many other features are included, such as:

- Multiple protocols: APPC, NetBIOS (IBM and Novell®), OS/2 named pipes
- Access to IMS transactions
- C and REXX language interfaces (+ any other language using the C calling conventions)
- Full duplex communications
- Easy-to-use connection definition tool

For information on Distributed Application/2™ call
1-800-IBM-CARY

List \$299

Distributed Application/2™



Developed by
IBM Programming Systems

**Unleash
Client/Server
Programming!**

this mode.

The first step in choosing an optimal compression mode is to check the printer model and available memory. The driver studies the raster data and

Fast System Fonts Option. In OS/2 2.0 and 2.1, there is a fast system fonts option on LaserJet and IBM 4019 printer drivers. When the option is disabled, OS/2 system fonts are sent to the printer as raster

printer driver to allocate substantially more RAM when using the LaserJet 4 or other 600-dpi printers. Test results show that this option works best with at least 8MB of RAM.

With normal buffers, the LaserJet printer driver allocates a 1MB memory buffer to hold raster data for a letter-sized page. (The printable area for letter size is 8.0-by-10.6 inches. At 300 dpi, it takes 11,250 bytes per square inch to hold monochrome raster data. Thus, a 300-dpi letter-sized page fits in 954,000 bytes.)

In higher resolutions or for forms larger than letter size, 1MB is not enough to hold all raster data. In this case, the driver uses a technique called "banding" to partition a page into several 1MB sections, or bands. Graphic orders are first recorded into a journal file, which is played as many times as necessary to write the raster data into each band. Using a 1MB buffer, printing to a LaserJet 4 printer at 600 dpi, the driver must generate four bands for a letter-sized page. The driver must play the journal file four times to render the whole page. Replaying the journal file is time-consuming; printing with it can take up to four times longer than printing a 300-dpi page.

The large buffers option increases the size of the banding buffer to 4MB. A buffer of this size is adequate for a 600 dpi letter-sized page; the driver need play the journal file only once. While this option may adversely affect a low-memory system, it provides a performance boost for users with enough system memory. Figure 2 shows test results for the large buffers option on computers with varying amounts of system memory.

Printer Patterns Option. This performance option was added in OS/2 2.1 for LaserJet and IBM 4019 printer drivers, to speed spreadsheet applications that fill rectangular areas with shading or cross-hatch patterns. It uses pattern-fill capabilities in the printer hardware instead of raster patterns generated by the GRE, reducing the size of the spool file and saving transmission time to the printer.

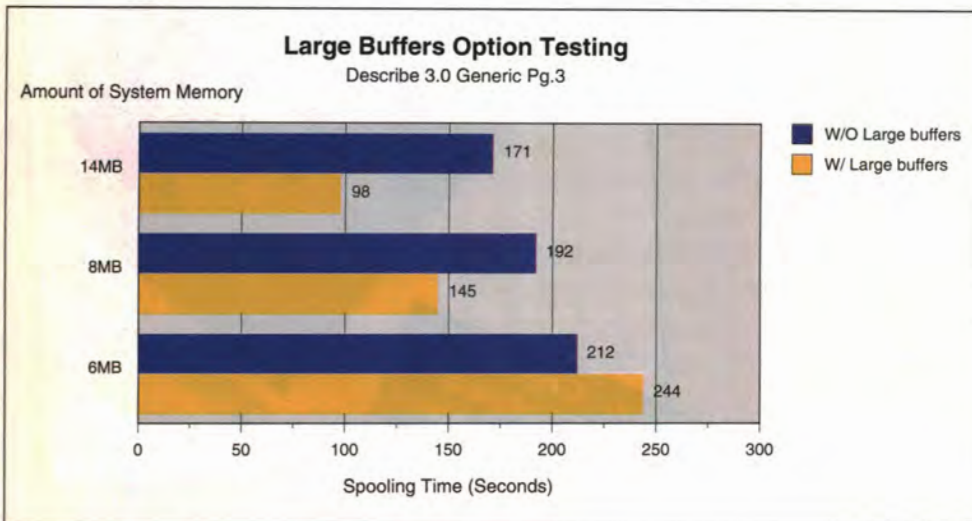


Figure 2. Large buffers option testing

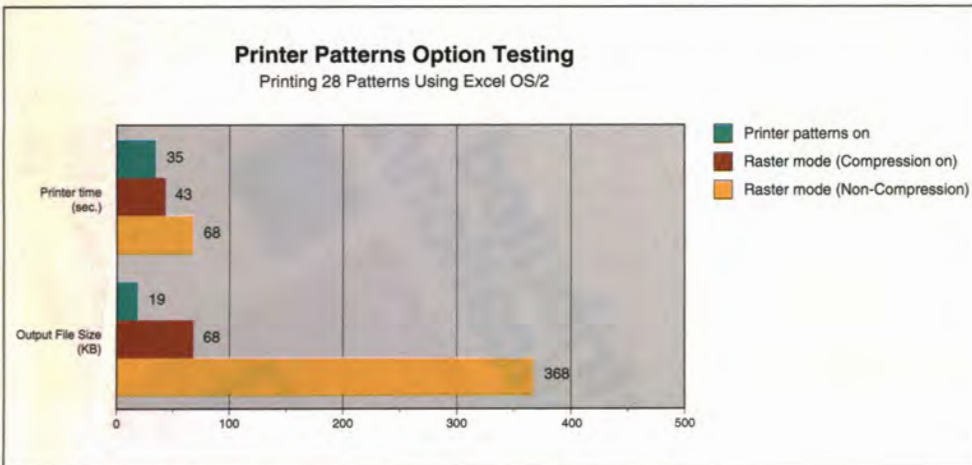


Figure 3. Printer patterns option testing

chooses a suitable compression mode. The best selection is the one that conserves the most spool file space and the time it takes to transmit data to the printer. Figure 1 shows the different spool file sizes of one page of raster data in two applications using different compression modes. The diagram shows that compression mode can have a large impact on file size and that the optimum compression mode varies with different output data.

graphics data. When the option is enabled, however, the printer driver downloads OS/2 system fonts to the printer where they become (essentially) device fonts. With a large text job (for example, a multipage spreadsheet) designed with OS/2 system fonts, printing speed is dramatically improved.

Large Buffers Option. This option refers to buffers of memory allocated from primary computer storage. When selected, it causes the PM LaserJet



When the printer patterns option is enabled, the printer driver hooks the GRE function `FillPath`. When the GRE calls the printer driver to fill a path, it generates the appropriate PCL4 or PCL5 printer control language commands for the target printer. PCL5 printers support the HP print model and allow pattern filling with background mixing modes. This hardware support is adequate for almost all pattern-filling graphic requirements.

The performance gain with the printer patterns option depends on two factors. First, the complexity of paths or regions to fill affects performance. For a small number of large rectangular fills, the option greatly reduces the size of the output file and increases the printing speed. For a large number of rectangular fills, however, it may overburden the printer's engine and decrease performance. Our tests show that this option improves performance when the areas to fill are of moderate complexity. Figure 3 shows the test results printing an Excel for OS/2 spreadsheet with pattern fills of 28 rectangular areas.

Second, the printer processor speed and printer language level affects performance. For PCL4 printers, the option should only be used for simple pattern filling (no overlay or white filling). Further, since the pattern-filling commands are executed in the printer, overall speed increase depends on the speed of the printer microprocessor.

HP-GL/2 Option. HP-GL/2 is the standardized version of the Hewlett-Packard Graphic Language usually found in the company's line of plotters. HP-GL/2 is especially well suited for drawing arcs and lines because the driver can efficiently generate vector drawing orders. Using HP-GL/2 commands reduces the output file size, increasing printing speed. When the HP-GL/2 option is enabled, the LaserJet print driver hooks the GRE functions for line, polyline, arc, box, area fill, box fill, and polygon fill.

HP-GL/2 mode is best suited for print jobs that are mostly vector and pattern-fill commands with few raster

operations. For example, printing a line drawing test case to a LaserJet 4 printer with mode 5 compression produces a 136K spool file without HP-GL/2; the same test case with HP-GL/2 enabled produces a 4K spool file. Figures 4 and 5 show the performance of HP-GL/2 in

with HP-GL/2. At an OS/2 command prompt, change directory (CD) to the `\SPOOL\ and do a directory listing (DIR) of *.SPL with the /OD (sort by date) switch. The last two files contain the PCL5 commands generated by the driver. If the latter (HP-GL/2 enabled)`

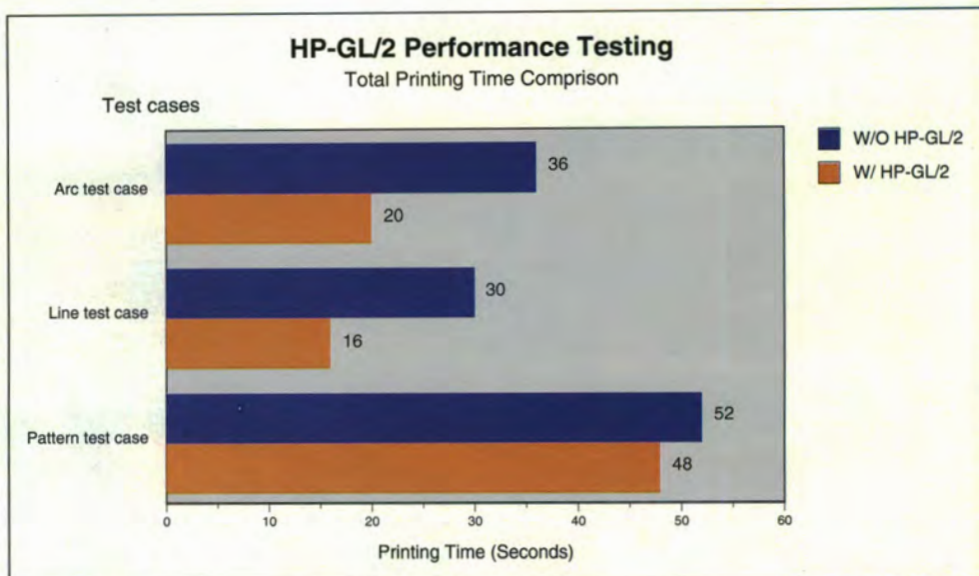


Figure 4. HP-GL/2 performance: printing time comparison

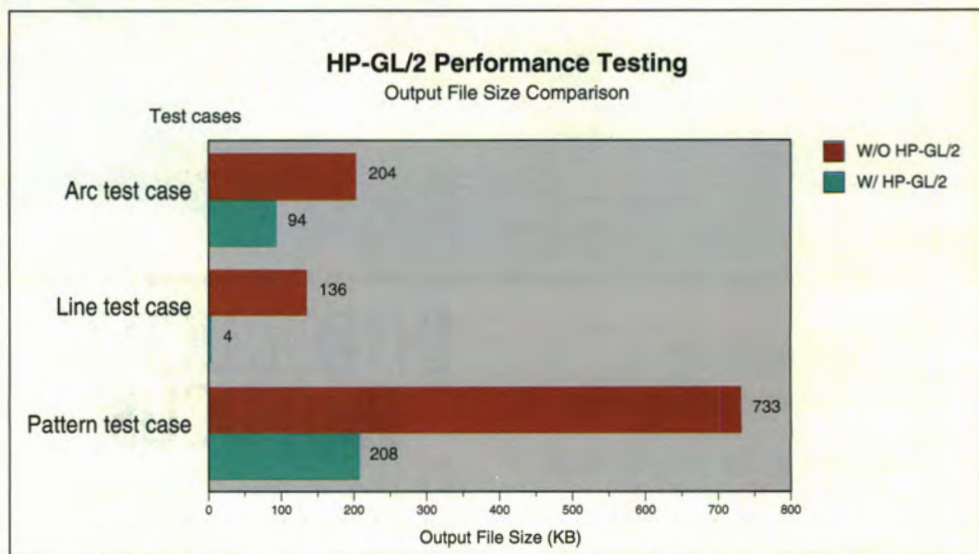


Figure 5. HP-GL/2 performance: output file size comparison

three test cases for arc drawing, line drawing, and pattern filling.

To tell if a job is suited for HP-GL/2, first specify printer-specific format in the LaserJet settings for queue options, then change the status to hold. Print the job two times, first without and then

trial is substantially smaller than the former, then the job is well suited for HP-GL/2.

There are some limitations to the HP-GL/2 option. Mixing HP-GL/2 drawing and raster drawing can lead to incorrect output if one should overlay

Create GUI Applications That Deliver The Promise Of OS/2.



YOUR WAIT IS OVER!

Now there's a graphical development and decision support system that combines unparalleled data access and reporting with the stability and multitasking power of OS/2 2.0. It's called PM/FOCUS from Information Builders, a leader in application development tools for almost every environment.

EXPERIENCE OUR EASE AND SPEED

PM/FOCUS offers a rich graphical toolset for fast, easy application development with a minimum of coding. All application components, including databases, procedures and forms become simple graphical objects that can be

INTRODUCING PM/FOCUS

controlled by mouse-driven "drag and drop."

Built-in list boxes, check boxes, radio buttons, type fonts and other graphical tools allow you to create attractive GUIs that are so intuitive, they're a snap for even the most unsophisticated end users.

POWERFUL REPORTING FEATURES

PM/FOCUS offers the most powerful reporting language of any product on the market today. And building reports is a breeze. You simply

transform database fields, record selections, and report headings and footers into selectable objects. Even complex sorts are available at the touch of your mouse.

DELIVER GREAT OS/2 GUIs

Get the promise of OS/2 now. Make PM/FOCUS your corporate standard for sensational GUI application development. For more information, or to attend a free seminar...

Call 800-969-INFO
In Canada call 416-364-2760

IBI **PM/FOCUS**
Information Builders, Inc.

the other. The order in which they are drawn can affect the final result.

Patterns are treated differently when HP-GL/2 is enabled; the driver essentially ignores the setting of the printer pattern's performance option in this case. When HP-GL/2 is on, the driver will always fill areas using HP-GL/2 commands.

Font output may also differ slightly with HP-GL/2; character strings rendered with it may differ from strings rendered with system fonts, especially when text is clipped, rotated, or mixed with other graphics such as lines or patterns.

Depending on job content, output rendered with HP-GL/2 can differ from the non-HP-GL/2 path. If so, turn off the HP-GL/2 option.

FUTURE DIRECTIONS

New advances in bidirectional ports and printers will eventually enable drivers to query the printer for installed memory and fonts, simplifying driver configuration and saving unnecessary work. We expect more application programs to subclass the Workplace Shell printer object and provide drag-and-drop printing for their particular file formats.

Our strategy is to move away from the 16-bit PM printer drivers and exploit the 32-bit model. The 32-bit print drivers and the GRE will interoperate using the flat memory model. The absence of thinking combined with the new performance techniques described here promise even faster printing on the horizon.

Shixiong Yang, IBM Corp., 1000 N.W. 51st St., Boca Raton, Fla. 33429. Yang is a staff programmer in the OS/2 advanced printing department at IBM, where he is currently working on OS/2 PM printer driver development. Yang holds a D.Eng.Sc. in electrical and computer engineering from the New Jersey Institute of Technology.

Monte Copeland, IBM Corp., 1000 N.W. 51st St., Boca Raton, Fla. 33429. Since 1987, Copeland has worked at IBM as a tester, technical support person, tools writer, and conference presenter. He is currently a presentation print driver developer. Copeland holds a B.S. in computer science from Florida Atlantic University.





Client/Server

As enterprises begin to roll out applications on client/server platforms such as OS/2, system administrators must manage large numbers of distributed resources such as clients, shared printers, and file servers. This presents an opportunity for developers of distributed system management applications. **BY CHUCK MCKELLEY**

LAN NetView: A Programming Overview



Chuck McKelley

Client/server architecture is characterized by applications executing on a large number of workstations and PCs rather than on one time-shared mainframe or minicomputer. In addition to performing traditional network management, the administrator of a client/server network must deal with such problems as distributing corrective service disks and new releases of software products to LAN-connected clients and servers, performing remote failure analysis, and monitoring remote system use. A few vendors have delivered distributed applications that help administrators solve these and other problems.

Most of these distributed system management applications consist of two major pieces. The section of an application that contains the user interface is often known as a manager or managing program, while the PC upon which it runs is called a managing station. Most distributed management programs also contain an agent, a program that runs on the systems to be managed. The systems on which the agent runs are called managed stations.

Management protocols, which managers use to communicate with agents, run over a variety of transport protocols, such as the transmission control protocol/internet protocol (TCP/IP) and the Net-BIOS protocol. Many management applications use private management protocols; some, however, use either the simple network management protocol (SNMP), originally designed to manage TCP/IP networks, or the common management information protocol (CMIP), designed to manage open systems interconnection (OSI) networks. SNMP and CMIP

are quite different and, until recently, most distributed system management application vendors have had to choose between the two. In addition, most vendors concentrate on only a few parts of the distributed system management problem, with the result that different system management applications are incompatible and have very different user interfaces. Further, an application that manages SNMP agents may not be able to communicate with CMIP agents and vice versa.

LAN NETVIEW

With the announcement of the LAN NetView distributed management products (based on Hewlett-Packard's OpenView technology), IBM has made it easier for vendors to integrate distributed management applications on an open-system, standards-based framework with a single object-oriented user interface similar to that of the Workplace Shell. The NetView family supports managing stations on OS/2 and managed stations on OS/2, DOS, and DOS plus Microsoft Windows platforms. The NetView open systems API allows the managing station to communicate with SNMP and CMIP agents on a variety of IBM and non-IBM platforms.

The LAN NetView family, shown in Figure 1, contains products upon which distributed system management applications can be built. The base product set includes a product for managing stations, LAN NetView Manage, and three for managed stations: LAN NetView Enabler, LAN NetView Agents for DOS, and LAN NetView Agents Extended.

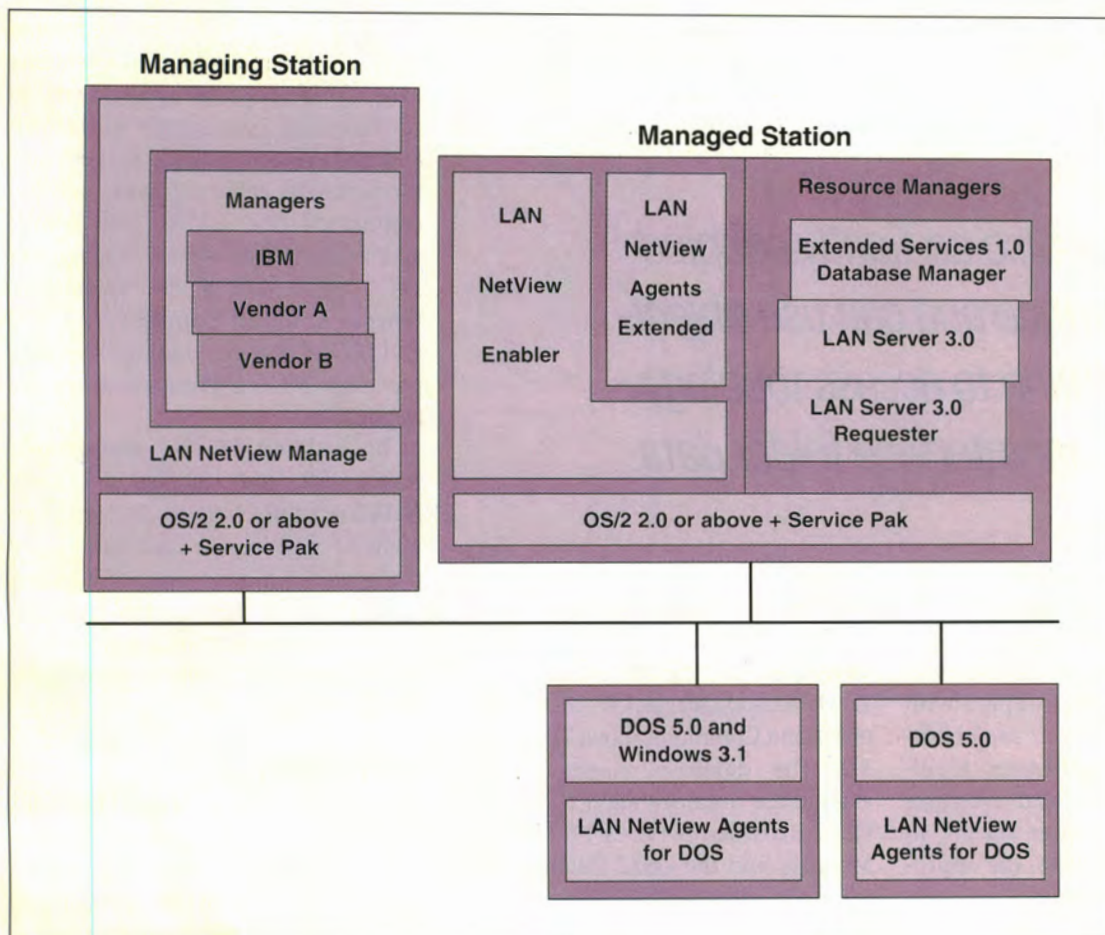
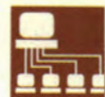


Figure 1. The LAN NetView family

MANAGING STATION COMPONENTS

LAN NetView Manage is installed on OS/2 managing stations. It contains core APIs and services that allow managing applications to communicate with CMIP and SNMP agents on managed stations throughout the network. Manage uses TCP/IP to communicate with SNMP agents. For CMIP agents, it supports CMIP over TCP/IP (CMOT) and CMIP over logical link control (CMOL). Manage also contains several components that assist applications with distributed management.

Event Management Services (EMS) collects CMIP events and SNMP traps issued by agents on remote systems and routes them to any application registered to receive them. EMS's filter service allows applications to register only for events and traps of interest to them. In addition, an EMS agent logs events and retrieves event data from the log.

The Metadata Service manages the NetView management information base (MIB). Agent developers can use the service to store their managed object definitions in the MIB, where they are available for use by application programs. Applications can retrieve object definitions to understand objects that have been added by another application or agent developer.

The Topology Management Service (TMS) contains a set of discovery functions that explore the network and collect information about network nodes and links. TMS stores the information in a network topology database available to management applications. The collected data includes status information, which is constantly updated to reflect the network state. TMS can discover SNMP and CMIP objects accessed via TCP/IP as well as CMIP objects accessed via CMOL.

Application integration and user interface func-



tions are supplied by the **View** component, the user's port into the network topology. An object-oriented user interface much like the OS/2 Workplace Shell, the View component was built using object-oriented programming techniques and the OS/2 system object model (SOM). There are two levels of integration available. Management

tions to receive and respond to requests from the managing station. The Enabler includes the same OS/2 and LAN Server requester agents packaged with Manage.

CMOL agents to manage workstations running DOS 5.0 or DOS 5.0 with Microsoft Windows 3.1 are provided by LAN NetView Agents for DOS.

is used on the managing and managed stations. XOM allows applications and agents on various platforms to encapsulate management data and requests in self-describing packets that can be decoded by any XOM-knowledgeable receiver. Where analogous operations exist, XMP permits applications to communicate consistently with CMIP and SNMP agents. While optimal performance is achieved with specific commands for CMIP and SNMP agents, the LAN NetView Manage and LAN NetView Enabler implementations of XMP can map between CMIP and SNMP requests, hiding the multiple management protocols from the application.

In addition to the management framework and agents, the LAN NetView family includes four management applications: a fault monitor, a performance monitor, a configuration manager, and a manager that transmits LAN notifications to and receives command functions from a mainframe NetView host.

CONCLUSION

The LAN NetView family provides developers of management applications and agents with a robust standards-based framework on which to build distributed management applications. The framework's support of multiple management and transport protocols allows developers to concentrate on management applications rather than the mechanics of communication. The X/Open APIs combined with the LAN NetView management services provide a standard way to supply agents, one that allows applications to manage resources consistently. Developers can share objects and integrate applications with the LAN NetView topology display.

Charles R. McKelley, Jr., IBM LAN Systems, 11400 Burnet Rd., Austin, Texas 78758. McKelley is an advisory programmer on the LAN NetView vendor support team. In his 26 years with IBM, he has held various technical and management positions in field support and software development.

An application can also be tightly coupled with the user interface and can use object-oriented programming to access topology information and user interface folder data.

applications can be launched from the topology display by registering an .EXE file with the View component's application integration service. An application can also be tightly coupled with the user interface and can use object-oriented programming to access topology information and user interface folder data. When a user selects an object in the View folders, the appropriate SOM class is called.

MANAGED STATION COMPONENTS AND PRODUCTS

LAN NetView Manage also includes agents for the LAN Server 3.0 requester and OS/2 2.0 with the service pack and 2.1. These agents can be used to manage the LAN Server requester and operating system on the managing station. For example, the OS/2 agent can retrieve information about the operating system, set information in the CONFIG.SYS file, and emit notifications when exception conditions are detected. Manage also contains a performance subagent that retrieves selected performance information.

The managed station counterpart to Manage is LAN NetView Enabler, which contains the subset of the LAN NetView Manage function needed to allow agents on OS/2-managed sta-

These DOS and Windows agents are installed on managed stations.

LAN NetView Agents Extended provides agents for OS/2 Extended Services 1.0 Database Manager, OS/2 LAN Server 3.0, and Communications Manager/2 1.0. The database manager agent exploits the administration APIs of the OS/2 Extended Services 1.0 Database Manager and the OS/2 Database Distributed Connection Services/2 1.0 on OS/2. Applications can use this agent to update database configuration information, catalogue nodes, or perform any other administrative task for which the database manager provides an API. The LAN Server agent allows management applications to perform remote administration on IBM LAN Server 3.0 requesters and servers on OS/2, with the Communication Manager/2 1.0 agent doing the same for Communication Manager/2 1.0.

LAN NETVIEW MANAGEMENT PROTOCOLS

LAN NetView-based management applications and agents communicate via the X/Open OSI Abstract Data Manipulation API (XOM) and the X/Open Management Protocols API (XMP). The open-system XOM and XMP APIs are symmetric; the same API

VAN NOSTRAND REINHOLD: YOUR BEST POWER SOURCE FOR OS/2® APPLICATIONS

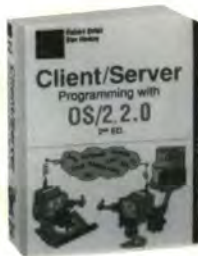
OS/2 Monthly's Book of the Year!
New edition of a best seller!

CLIENT/SERVER PROGRAMMING WITH OS/2® 2.0 Second Edition

By Robert Orfali and
Daniel Harkey

Readers raved about the first edition of this best selling Client/Server-OS/2 resource. Now updated with in-depth tutorials and sample code, this is the ideal guide to client/server in the 32-bit environment. Covers all new 2.0 functions.

\$39.95, 1,026 pages, paper, 0-442-10219-5



New! Power up with the experts!

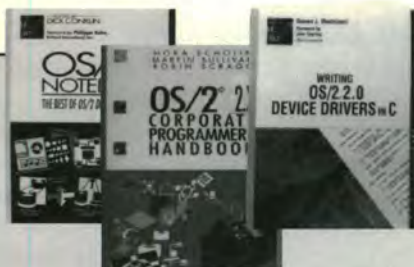
USING WORKPLACE OS/2®

Power User's Guide to IBM's New Operating System/2 Version 2.1

By Lori T. Brown and Jeff Howard

Get the "inside" help of the Workplace Shell's lead designers to • convert easily and quickly from Windows and Mac environments to OS/2 • set up, maintain, and customize your own Workplace environment • power up with OS/2's Multimedia Presentation Manager/2 under Workplace.

\$24.95, 400 pages, paper with disk, 0-442-01590-9



New!

OS/2® 2.X NOTEBOOK The Best of OS/2® Developer Magazine

Edited by Dick Conklin
Foreword by Philippe Kahn,
Borland International, Inc.

The very best articles from OS/2 Developer Magazine, completely revised! Includes product reviews, coding examples, user tips and techniques. Over 1,100 pages covering software tools, graphical user interface, client/server and object oriented programming, multimedia and graphics and lots more.

\$34.95, 1,164 pages, paper,
200 illustrations, 0-442-01522-4

New!

THE OS/2® 2.1 CORPORATE PROGRAM- MER'S HANDBOOK

By Nora Scholin, Martin Sullivan,
and Robin Scragg

Simplify migration from DOS and Windows to OS/2 2.0 and find valuable solutions based on modular subdesigns, from function key assignments to pull-down menus.

\$39.95, 352 pages, cloth,
0-442-01598-4

WRITING OS/2® 2.0 DEVICE DRIVERS IN C

by Steven J. Mastrianni

The first guide to programming 32-bit OS/2 device drivers includes C source code examples, with optional disk. Reduces the difficulty and cost of writing device drivers for OS/2.

\$36.95, 410 pages, paper with
optional disk, 0-442-01141-5

LOOK FOR THESE OS/2® 2.0 LIBRARY BOOKS
from VAN NOSTRAND REINHOLD
at major technical bookstores nationwide.

EASY ORDERING! CALL TOLL-FREE: 1-800-544-0550.
Or Fax 1-606-525-7778



VAN NOSTRAND REINHOLD
Publishing for Professionals Since 1848.

OS/2® is a registered trademark of IBM Corporation.



Client/Server

Accessing and analyzing data in a variety of ways is critical in a business environment. The IBM Personal Application System/2 Version 3 is IBM's new 32-bit client/server product for decision support applications within the Information Warehouse framework architecture. **BY RON DOUGLAS, SHAUN JONES, and WILLIAM KING, JR.**

32-bit GUI Client/Server Application Development With Personal AS/2

Personal Application System/2 Version 3 (Personal AS/2 V3), developed by IBM's Programming Systems (PRGS) division, is a 32-bit application designed for data access, query, data analysis, presentation, and communication of business results. It contains a range of tools to manage and format data, including:

- Table edit and browsing
- Query
- Data analysis and manipulation
- Report writing
- Chart creation
- Task automation via iconic procedures.



Ron Douglas



Shaun Jones



William King, Jr.

users and takes advantage of OS/2 2.1 Workplace Shell functions such as drag and drop, clipboard support, and dynamic data exchange (DDE).

PERSONAL AS/2 V3 PACKAGING

The minimum configuration of Personal AS/2 V3 is the base product, which provides data access, reporting and charting functions, and a graphical procedure function for generating basic applications. The base product also provides run-time support for user applications developed with the optional Builder/2. Both products can be installed on a stand-alone PS/2 or LAN. Personal AS and related products are also available on the DOS Windows platform.

Personal AS/2 V3 is designed for business

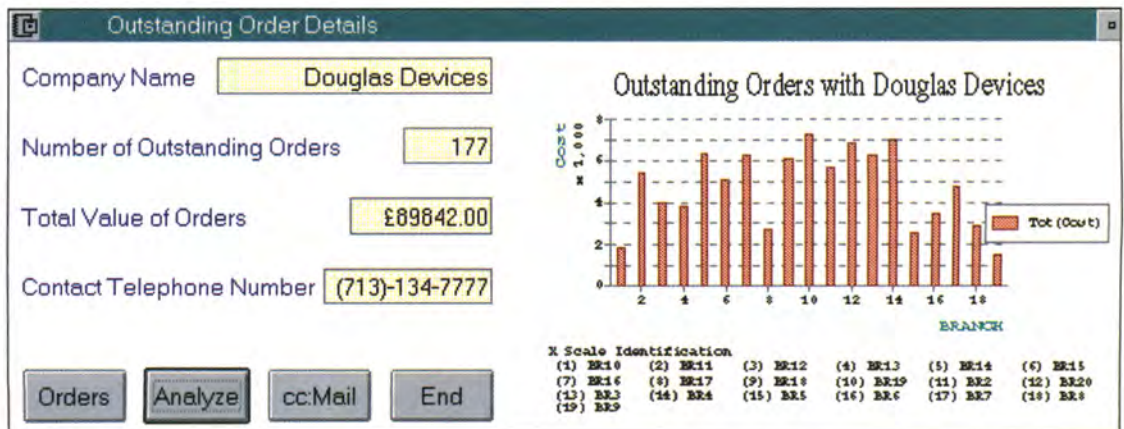


Figure 1. Custom window created with Builder/2

OS/2 PRODUCTS & SERVICES

Consulting **OS/2 PM CUA**

Programming
Management | Testing
Analysis | Installation
Design | Maintenance
Real-Time Controls and device drivers
(708) 380-3584

BUILD YOUR OWN OS/2 LIBRARY

Reprints are now available for **OS/2 DEVELOPER**.

Use your customized reprints for: **Customer support, sales tools, marketing support, educational tool.**

Call today for a custom quote: Laura Pullen, **OS/2 DEVELOPER**
415-358-0126 ext.302 or fax 415-358-9749

Attention Mathematica users:
Everything you need to know to get the most out of your math programming is in

THE MATHEMATICA JOURNAL

Call or fax today for information about subscriptions and special issues:
Phone orders: 415-905-2332
Fax orders: 415-905-2233

NEURAL NETWORK

SPECIAL REPORT

Your comprehensive guide to neural network tools and the companies that make them - all new for 1992!

All for only \$7.95.

To order, call 415-905-2376, or fax to 415-905-2233 now

OCR for OS/2

The most accurate and fastest Optical Character Recognition (OCR) software engine on the market today.

OCR Library and API Toolkits available for OS/2, MS Windows, Macintosh and UNIX.

Cognitive Technology Corp.

Larkspur, CA
Ph. (415) 925-2323 - Fax (415) 461-4010

TCP/2™ tcp/ip for OS/2

"TCP/2™ is, by far, the best implementation of the tcp/ip application suite for DOS and OS/2 available from any source"

Microsoft ITIS

Essex Systems, Inc.

One Central Street Phone (508) 750-6200
Middleton, MA 01949 Fax (508) 750-4699



Custom Entryfields allow formatting for fields like Phone Number, Dates, Social Security Number, Drivers License, etc.

Easy to use. Just change the window class in your resource file.

ENTRYFIELD PICTURE MASKS for OS/2



Download free demo of Impact Entryfields from our BBS.
(818) 879-7405

Add Impact to your application



IMPACT SOFTWARE

To order call or write
(800) 676-9390
(818) 879-5592
FAX (818) 879-5593

5889 Kanon Rd.
Suite 330
Agoura Hills, CA 91301

SUBSCRIBE TODAY!

Only \$39.95 for a full year subscription to the premier source of tools and techniques for the OS/2 software developer: **OS/2 DEVELOPER**

Call today and don't miss a single issue:

1-800-WANT-OS2

“No more



Gary Slattery, Software Developer, Computer Associates

“The best platform for DOS and Windows.”

“I develop software applications for a living and I think OS/2® is a great way to do business.” A 32-bit, virtual memory operating system, OS/2 is the ideal platform for developing your DOS, OS/2, Windows™ and even host-based applications.

With OS/2 you can boost the power of your favorite DOS and Windows tools,

IBM and OS/2 are registered trademarks and Workplace Shell, C++, CommonView, C Set/2 and OS/2 Crash Protection are trademarks of International Business Machines Corporation. All other products are trademarks or registered trademarks of their respective companies. ©1993 IBM Corp.

plus take advantage of over 250 available OS/2 development tools and utilities.

“It’s a productivity thing.”

“I use CA Realizer to prototype new graphical interfaces. I write code using CA C++™ and CommonView™ (integrated with IBM’s C Set/2™ compiler) while I compile in the background. And when designing reports or planning schedules, I use CA-RET.”

OS/2’s pre-emptive multithreaded multitasking dynamically manages CPU time so you can run



DOS, Windows and OS/2 apps concurrently in different sessions with maximum efficiency. That means you can edit in one window, compile in another, link in a third and test in a fourth. With OS/2 Crash Protection,™ if one application goes down due to a bug, the rest you’re working on won’t. “There’s no limit to what you can do with this system.... It’s definitely made me more productive.”

The Development Platform of Choice

- Enhanced development platform for DOS, Windows and OS/2 apps.
- OS/2 Crash Protection for superior reliability.
- Pre-emptive multitasking for increased productivity.
- Virtual memory provides up to 512MB per session.
- Flat memory model eliminates wrestling with segments.
- Multiple virtual DOS machines for concurrent app testing.
- Object-oriented Workplace Shell is easy and intuitive.



arrested development.”

The object-oriented user interface—the Workplace Shell™ (WPS)—gives you easy control with direct manipulation of visual objects on your computer screen. And should you need assistance with anything, IBM's Worldwide Developer Assistance Program is always there to help.

“A developer's dream.”

With OS/2's 32-bit architecture, you can push your 386 and 486 hardware to the limit, and develop spectacular multimedia and enterprise-wide client-server applications. OS/2 lets you create the widest range of applications, for a variety of platforms, for almost any size computer. Here's your chance to develop truly revolutionary 32-bit applications. With OS/2, now there's nothing stopping you.

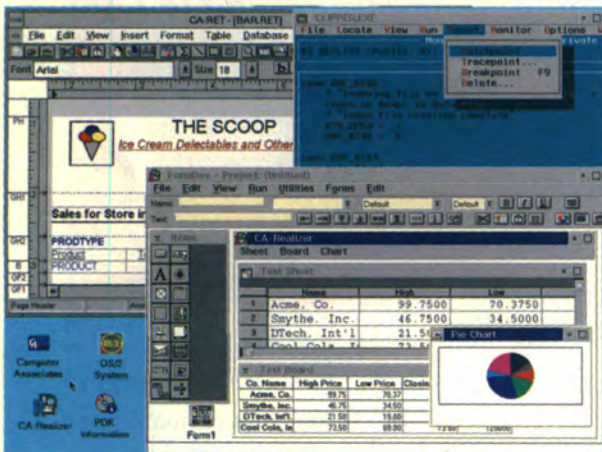
1992
- WINNER -
PC Magazine Award
for Technical Excellence



OPERATING SYSTEMS AND
SOFTWARE STANDARDS
OS/2, Version 2.0
IBM Corporation

To learn more, call 1 407 982-6408 now, and get a free white paper on why OS/2 is the ideal platform for your development efforts.

Operate at a higher level.



Use OS/2 to increase productivity of DOS, Windows and OS/2 application development.





Personal Application System Builder/2

Personal AS/2 V3's vendor-independent messaging interface allows integration with office systems, enabling files and data to be sent to IBM and non-IBM e-mail applications. This capability can be extended with the optional Personal Application System Builder/2 (Builder/2). Personal AS/2 V3's features can also be integrated into applications launched directly from the Workplace Shell.

The Personal AS/2 V3 base product provides data access, reporting and charting of functions, and a graphical procedure function for generating basic applications.

Builder/2 functions that help users who develop customized solutions for others include:

- ASL, an event-driven development language
- Full SQL access to the OS/2 client/server database
- Access to distributed relational data via Distributed Data Connection Services/2 (DDCS/2)
- Support of DDE as client and server
- An interface that allows e-mail capability
- Editors for window layout and control placement
- An intelligent program editor for rapid logic creation
- A C language interface
- OS/2 clipboard support
- An interactive debug facility
- APIs to objects such as Report and Chart.

```
Source program : Untitled
Make File      : DEMOCALC.WIN
Date           : 7-3-92
Owner          : System generated code

Source window  Designated object name
DEMOCALC.WIN  W_DEMOCALC

ON SELECT
This block is executed whenever a selectable control or menu
entry is activated.

ON SELECT
DO

CASE A.System.Object          ! Which screen object?

WHEN "T.W_DEMOCALC.Multiply" ! Menu entry "Multiply"
DO
! Insert your code here
END

WHEN "T.W_DEMOCALC.Divide"   ! Menu entry "Divide"
DO
! Insert your code here
END

WHEN "T.W_DEMOCALC.Add"      ! Menu entry "Add"
DO
! Insert your code here
END

WHEN "T.W_DEMOCALC.Subtract" ! Menu entry "Subtract"
DO
! Insert your code here
END

WHEN "T.W_DEMOCALC.STD_PUSH"
DO
CASE A.System.BoxNumber
WHEN 1                          ! Standard pushbutton "Calculate"
DO
! Insert your code here
END

WHEN 2                          ! Standard pushbutton "Clear"
```

Figure 2. Example of system-generated ASL code (continued on page 57)

Builder/2 can be viewed as an application development tool with a faster learning curve than languages such as C. It combines customization with business tools via its APIs and DDE linkages, providing prototyping


```

DO
  ! Insert your code here
END

WHEN 3          ! Standard pushbutton "Stop"
DO
  ! Insert your code here
END

END
END
END          ! End case, A.System.Object

END

```

Figure 2. Example of system-generated ASL code (continued from page 56)

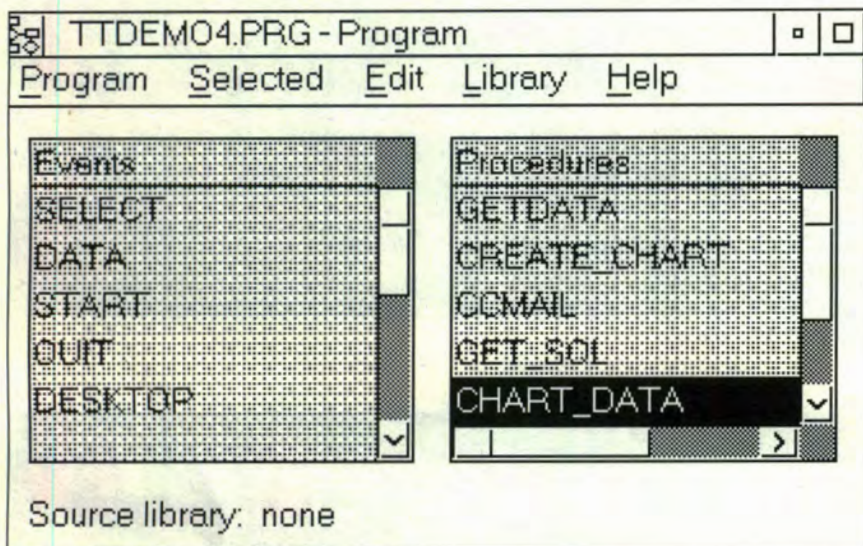


Figure 3. Personal AS/2 V3 ASL editor

and application development features that can adapt and extend the base product to meet specific requirements.

PROVEN PRODUCT FOR MULTINATIONAL BUSINESSES

Personal AS/2 V3 is the third version of the Personal AS product line, first introduced in 1989. The product is used in a variety of industries—retail, banking and finance, health, public sector, and petrochemical.

In addition to English, Personal AS/2 V3 is available in six languages: Dutch, French, German, Italian, Spanish, and Japanese (DBCS). This availability makes it useful for multinational

businesses seeking a single data analysis product for the enterprise.

DATA ACCESS: STRENGTHS

The strength of Personal AS/2 V3 lies

in its built-in data access; it provides access to relational and nonrelational

data at an enterprise, departmental, or personal level in several ways:



- Exploitation of IBM's Distributed Relational Database Architecture (DRDA)
- Support of Information Builders' Enterprise Data Access (EDA/SQL) technology
- Links to AS/400 via PC Support and DDCCS/2
- The Application System S/390 Intelligent Server
- Its own relational-like table structures
- Native ability to read and write data formats including ASCII, DIF, IXF, and DBF.

A decision support component of IBM's Information Warehouse framework architecture, the product directly accesses data stored in the OS/2 Database Manager or the newer IBM Database 2 OS/2 (DB2/2) product. In conjunction with IBM's Distributed Data Connection Services/2 (DDCS/2), Personal AS/2 V3 also provides seamless access to distributed relational data located on remote server databases such as OS/400, SQL/DS, and DB2.

USING BUILDER/2 TO DEVELOP 32-BIT GUI APPLICATIONS

Although Personal AS/2 V3 can be used as packaged, the most successful installations are those in which its functions are launched from a customized application developed with Builder/2. The custom Presentation Manager GUI window shown in Figure 1, for example, was created with Builder/2.

The control of such windows is

Personal AS/2 V3 provides access to relational and nonrelational data at an enterprise, departmental, or personal level.

managed by Personal AS/2's Application System Language (ASL), automati-



APPLICATION	The server application of interest.
CODE	An attribute that can be queried to obtain the error code of the last DDE client operation.
ITEM	The name of the server item.
TIMEOUT	The timeout period, in seconds, beyond which the DDE client object will consider the DDE server unresponsive.
TOPIC	The server topic of interest.

Table 1. Attributes of the DDECLIENT ASL object that can be queried by an ASL application

DATA	This event occurs when the DDECLIENT has issued an ADVISE without the immediate parameter, and the specified item has been modified.
ERROR	This event signifies an error in processing data.
QUIT	This event occurs when the server terminates or no longer services the TOPIC.
SOURCE(pStream)	This event occurs when the DELIVER action is invoked. pStream is a pointer to a STEAM object delivered with the event through which data can be rendered.
TARGET(pStream)	This event occurs when a data delivery happens between a DDESERVER and a DDECLIENT in response to either a REQUEST or ADVISE action with the immediate parameter specified.

Table 2. Discrete events associated with the DDECLIENT ASL client

- ADVISE(Item, Format, Immediate) may be used to register an interest in a specified item so the client is notified whenever the item is updated.
- DELIVER(Item, Format) allows the client to send data to the server in a format of the client's choice.
- EXECUTE(Variable) allows the client to send a command string to the server.
- PROCESSMEMORY(pSMemory, Status, Address, Size) is invoked by an instance of SMEMORY when its FINISHED() action is called.
- QUERYFORMATS(Formats) is used to obtain information about the format capabilities of the server.
- REQUEST(Item, Format) is used to obtain data from the server on the item specified, in the format specified.
- SYSTEM(Item, Variable) is used to obtain information about the capabilities of the server.
- UNADVISE(Item, Format) removes a registration of interest for a prior ADVISE.
- VERIFYFORMAT(Format) may be used as an alternative to QUERYFORMATS; it verifies that the server can render a specific format.

Table 3. Actions that control the function of the ASL DDECLIENT object

Client/Server Database Solutions

It's available now—ready to perform on your desktop. A new function-rich, 32-bit relational database you can really trust with your growing client/server network, your mission-critical data *and* your business.

Introducing IBM DATABASE 2™ OS/2® (DB2/2™) from IBM Programming Systems, the birthplace of relational database technology.

DB2/2 includes an industrial-strength DB engine that supports transaction management, concurrency control, security, integrity, and recovery functions. Designed to exploit the power and open architecture of OS/2, it also supports industry-standard SQL for developing portable applications. And it runs your DOS, DOS Windows™ and OS/2 applications requiring *online* access.

You can access data directly from DB2/2 on your desktop or from a DB2/2 server on your LAN, and with

DB2

goes

DISTRIBUTED DATABASE
CONNECTION SERVICES/2™
from DB2®, SQL/DS™ and OS/400®
databases as if they were on your desktop, too.
This versatility can play a significant role in
an Information Warehouse™ solution
for your business.

We've developed an

desktop.



exciting demo diskette to show you just how well new DB2/2 performs—right on your desktop. Call us today for your free demo, or to order DB2/2: 1 800 342-6672; or fax: 1 800 445-2426. In Canada, call 1 800 465-7999, ext. 850. An upgrade from IBM Extended Edition or Extended Services is also available.

**Manufacturer's suggested retail price is \$425.*

IBM, OS/2, DB2 and OS/400 are registered trademarks and DATABASE 2, DB2/2, DISTRIBUTED DATABASE CONNECTION SERVICES/2, SQL/DS and Information Warehouse are trademarks of International Business Machines Corporation. Windows is a trademark of Microsoft Corporation. © 1993 IBM Corp.

\$199* U.S.
SPECIAL OFFER
INTRODUCTORY OFFER
(UNTIL AUGUST 31, 1993)
FOR SINGLE USER VERSION OF DB2/2
• SEE YOUR NEAREST OS/2 PRODUCT DEALER
OR CALL 1-800-342-6672.



Template for the ON TARGET block. This event is signalled when there has been Direct Manipulation between this application and some other application. This applications Source window has had an Icon dropped onto it from some other application. This application can now ask questions to find out what's been dropped, and take action.

```
ON TARGET(
  pStream)
DO
  DECLARE POINTER pStream
  CASE
    WHEN (?pStream)ˆFORMAT = "Link"
    DO
      DEFINE LinkData[0]
      LET X00 = BINARY(0,1)
```

Figure 4. ASL DDE extentions (continued on page 62)

cally generated by Builder/2 when the window is created. ASL is an event-driven object-oriented language. The system-generated code that determines which window a user has selected is shown in Figure 2.

A key feature of ASL is its combination of object-based language and event handling. It includes a built-in file system; vector handling; and mathematical, string, and graphics facilities. ASL is object-based, with objects that can be used for tasks from handling complex windows to accessing SQL databases, DDE, and reporting and charting functions. Execution control is provided by a set of discrete events (such as START, SELECT, and QUIT) that can be detected by the language for specific processing.

Builder/2 features designed to improve programmer productivity include a window editor with built-in palettes for selecting standard PM controls, a full-function ASL editor, an ASL compiler, and a make facility. The editor, shown in Figure 3, has been redesigned for Personal AS/2 Version 3.

ASL extensions are provided to support access to the OS/2 relational data manager (DB2/2), DDE, emulator high-level language application programming interface (EHLLAPI), and tight coupling with C and REXX.

In Figure 4, ASL extensions for support of DDE automatically link a Personal AS/2 application with an OS/2 2.1 spreadsheet that has been dropped onto the application's source window for processing. (In this example, Personal AS/2 V3 functions as a DDE client).

Five attributes of the DDECLIENT ASL object can be queried by an ASL application; these are shown in Table 1. Five discrete events, shown in Table 2, are associated with the DDECLIENT ASL client. Finally, Table 3 lists nine actions that control the function of the ASL DDECLIENT object.

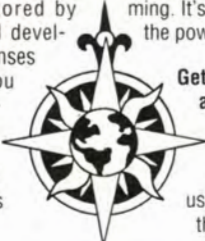
Personal AS/2 V3 can also function as a DDE server where a Personal AS/2 V3 object, such as a report or chart, is linked to another OS/2 2.1 object, such as a word processing document. The attributes, discrete events, actions, and

Looking for fast answers to your development questions?

**You need to do two things:
Go on-line with CompuServe
Information Service. Use Golden
CommPass to do it.**

CompuServe hosts OS/2 developer and user forums monitored by technical personnel. IBM developers are there with responses to anything that's got you stuck. Other OS/2 programmers and enthusiasts are there, too, comparing notes and giving feedback. It's definitely the place to be.

Time is money when you connect to CompuServe, and Golden CommPass saves you both. It lets you compose your questions off-line, send them in a flash and disconnect. It gets your replies while you're busy programming. It's a native OS/2 app, with all the power and features you'd expect.



Get the fastest access to answers. Golden CommPass keeps your development schedule on course. The fact is, the sooner you start using our program, the sooner they'll be talking about yours!

Phone:
(609) 234-1500

Fax:
(609) 234-1920

**Golden
CommPass™**

CompuServe® Access Software

CompuServe:
71511, 151

90 Day Satisfaction
Guarantee

Creative Systems Programming Corporation

Our *Fax/PM*[®] Applet* made OS/2 fax aware...

INVOW

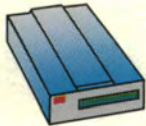
...Our *Fax/PM* makes you fax capable



Fax/PM uses the advanced OS/2 features (drag & drop...), turning your computer into a sophisticated fax machine. Fax/PM enables you to send, receive and print faxes in background mode.

Lan

OS/2 standalone, LAN or Client-Server, as well as Windows, DOS and AS/400 versions: there is a Fax/PM configuration to suit your needs.



Supports Group III fax modems (Class 1 & 2). All internal laptop modems and specific fax boards. Standard fax speed (9600 bps) and high speed (14400 bps).

API

Fax/PM provides APIs for fax integration in custom applications. These APIs can be called by any programming language or REXX procedure.

Dos
Windows
OS/2

Under OS/2: **to fax... just print...** Without leaving your favorite DOS, Windows or OS/2 applications, Fax/PM makes faxing easy for you. All fonts supported, for a "Laser-printed" quality fax.



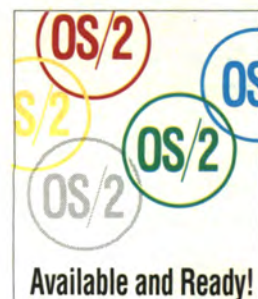
AS/400

Fax without leaving your AS/400. Fax/PM AS/400 provides seamless integration for AS/400 and PC users. Fax from OfficeVision/400, spooled files,...

 **microformatic**

U.S.A.
PO Box 722
610 Niederwerfer Road
South Windsor, CT 06074
Tel: (203) 644-1708
Fax: (203) 648-9587

EUROPE
2, rue Navoiseau
Montreuil-sous-Bois
F-93100
Tel: 33 (1) 48 70 19 00
Fax: 33 (1) 48 70 27 29





events for the DDE Server ASL object are similar to those listed for the DDE Client ASL object.

PERFORMING EVENTS AT SET TIMES

Builder/2 "timers" allow events to occur at predetermined times or after specific time intervals. Timers can be used, for example, to schedule a middle-of-the-night data manipulation, automatically route output to a waiting printer or plotter, or initiate an ASL/EHLLAPI program (like the one described in the previous section) for automated processing with the S/390 server.

Figure 5 shows a sample Personal AS/2 V3 timer application developed with Builder/2, in which a user can specify the date and time a predefined Personal AS/2 V3 procedure, report, or chart is to be executed.

These automatically triggered events allow users to perform regular tasks using the latest corporate data, such as producing a weekly sales report or a chart of the most active customer accounts for a month without human intervention. This level of automation is made possible by Personal AS/2 V3's support for APIs into REPORT and CHART objects called from an ASL program. Figure 6 shows an API call to the CHART object. The type and style of the chart, as well as the X and Y data variables to be used, can be determined under direct ASL program control.

USING PERSONAL AS/2 V3 IN BUSINESS

Personal AS/2 V3 is designed to access enterprise data from the OS/2 desktop; its level of integration with the OS/2 Database Manager, DB2/2 and DRDA make it a strong prototyping tool for new client/server applications. With Builder/2, prototype applications and their CUA 91-compliant windows, applications can be generated in a fraction of the time required to manually code them. In one business application, for example, Personal AS/2 V3 is being used in joint application development projects in which users develop the

```
CALL (?pStream)'GETDATA(LinkData[0])
CALL (?pStream)'FINISHED()
SHUT ?pStream

LET Application = WORDS(LinkData[1], 1, X00)
LET Topic       = WORDS(LinkData[1], 2, X00)
LET Item        = WORDS(LinkData[1], 3, X00)
CALL DDECLient(
  Application,
  Topic,
  Item)
END
OTHERWISE
DO
  DEFINE ServerData[0]
  CALL (?pStream)'GETLINE(ServerData[0])
  DO i = 1:ServerData[0]'ENTRIES
    STAMP LENGTH(ServerData[i])
  END
  CALL (?pStream)'FINISHED()
  SHUT ?pStream

  SHUT ServerDisp
  DEFINE SD_ColForm[0]
  INSERT SD_ColForm[0] = "WIDTH="
    MAX((6*LENGTH(ServerData)),Group'SIZEX - 15)
  DEFINE SD_ColData[0]
  INSERT SD_ColData[0] = ServerData[0]

  OPEN LIST ServerDisp, Window,
    X      = Group'X + 5,
    Y      = Group'Y + 5,
    SIZEX  = Group'SIZEX - 10,
    SIZEY  = Group'SIZEY - 15,
    SELECTION = 1,
    HORZSCROLL = 1,
    EXPRESSION = SD_ColForm[0],
    COLDATA  = SD_ColData[0]

  END
END
END

PROCEDURE DDECLient(
  Application,
  Topic,
  Item)
```

Figure 4. ASL DDE extensions (continued on page 63)

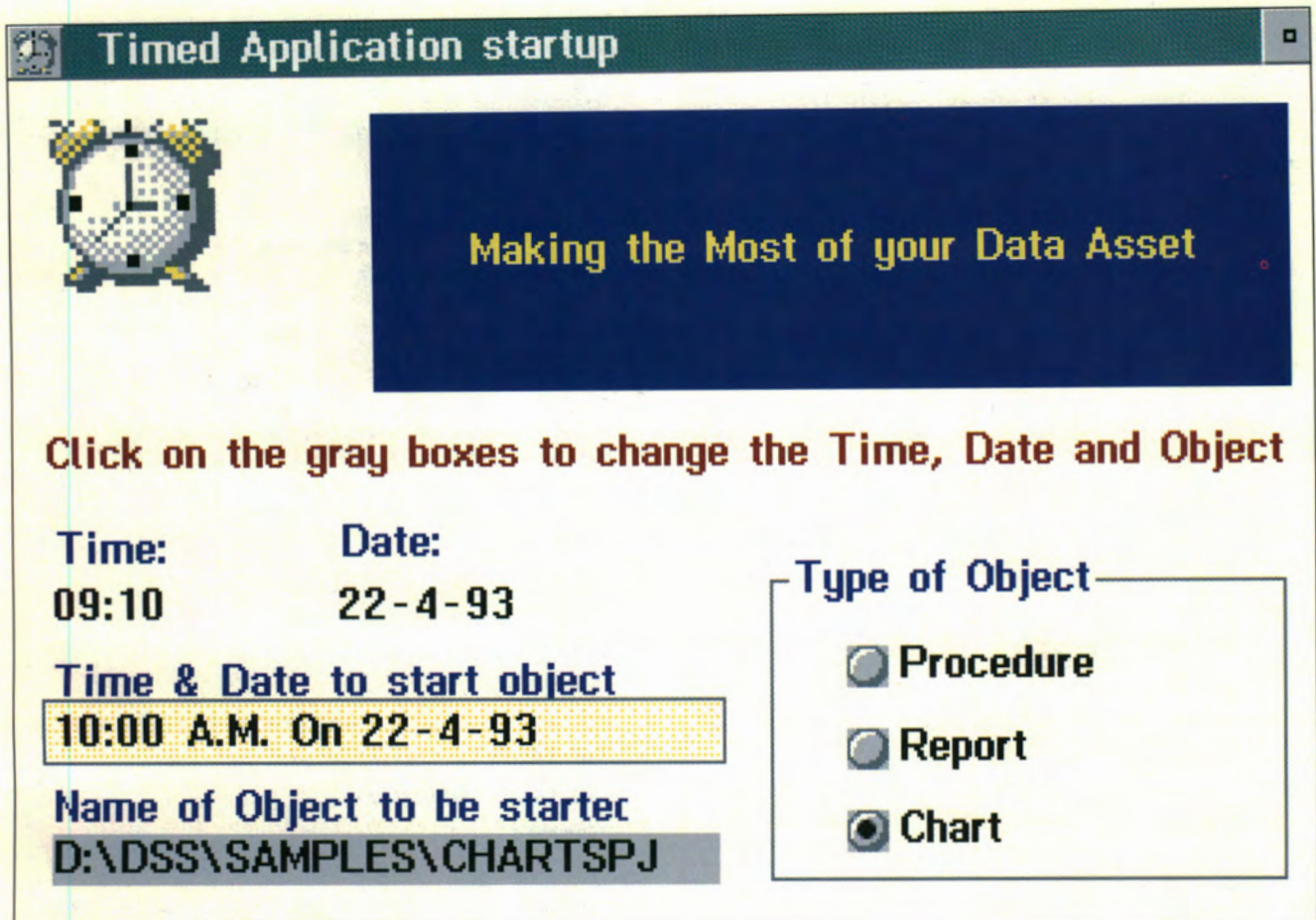


Figure 5. Sample Personal AS/2 V3 timer application

```

DO
  STAMP Application
  STAMP Topic
  STAMP Item
  SHUT DDECLient
  OPEN DDECLIENT      DDECLient,
    APPLICATION = Application,
    TOPIC       = Topic,
    TRACE       = DDETrace
  LET Text`TEXT = STRING(
    "DDECLIENT OPENed for Application: ", Topic: ",
    Application, Topic)

  CALL DDECLient`SYSTEM(SysItems[0], "SysItems")
  LET Request`ENABLED = 0
  LET Advise`ENABLED = 0
  LET Execute`ENABLED = 1
END

```

Figure 4. ASL DDE extentions (continued from page 62)

Several educational offerings on Personal AS/2 V3 are available from Skill Dynamics, an IBM Education company. For a complete list of Personal AS/2 V3 courses available in the U.S., call (800) 426-8322 and request a course schedule, or fax (800) 426-4329 and request the Education List Index.

Disclaimer: This article was prepared prior to the general availability of Personal AS/2 V3 using a development version of the product. Please note that some functions could be added, changed, or deleted in the production version of the product.



```
To use Chart API - CHARTAPI.PGM - ADC 1992
ON START
DO
  !Find out where Base is installed
  LET Basepath=S.Control.Path

  Set the variables that will be used later

  For OPEN IBMCHART
  LET ChartType="Bar"

  For SelectTable
  LET filename=Basepath||"\SAMPLES\EMPDATA"
  LET filesystem="PAS"

  For X SetAxisType
  LET xaxistype="Character"

  For SelDataX
  LET xcolumn="Division"

  For Y SetAxisType
  LET yaxistype="Numeric"

  For SelAnalysis
  LET Column="AnnualSalary"
  LET Function="Avg"
  LET Display="Abs"

  OPEN IBMCHART Mychart,
  Name = "",                !No current name
  Identifier = Basepath||"\Chart", !Name of template
  Type = ChartType,        !Chart Type
  Perspective = 1,         !2D = 0, 3D = 1
  Rotated = 1,             !Rotated = 1 or 0
  AutoRefresh = 0,        !AutoRefresh = 1 or 0
  SavePrompt = 0,         !Prompt = 1 or 0
  Dialog = 1,             !Display dialogs ?
  EUIMode = "Restricted"  !Full or Restricted

  Reset any existing input table
  CALL Mychart'RESETDATA()
```

Establishing the X and Y data before the input table is defined prevents the Select columns dialog from appearing automatically.

```
Set X axis type, variable set above
CALL MyChart'SetAxisType(
  "X",
  xaxistype)
```

```
Set X column, variable set above
CALL MyChart'SelDataX(
  xcolumn)
```

```
Set Y axis type, variable set above
CALL Mychart'SetAxisType(
  "Y",
  yaxistype)
```

```
Set analysis for Y scale, variables set above
CALL MyChart'SelAnalysis(
  Column,
  Function,
  Display)
```

```
Choose the table, variables set above
CALL MyChart'SelectTable(
  filename,
  filesystem)
```

```
Finally open the Chart
CALL MyChart'OPEN()
```

END

```
ON QUIT
STOP PROGRAM
```

Figure 6. Calling a CHART API from ASL

prototype windows themselves using Builder/2.

ACKNOWLEDGMENTS

The authors would like to thank David Bryant for the sample DDE ASL program and for reviewing this article for technical accuracy and Alan Carpenter for the sample CHART API ASL program.

Ronald J. (Ron) Douglas, IBM Programming Systems Division, Warwick Software Development Laboratory, 1 Birmingham Road, Warwick, England, CV34 5JL. Douglas joined IBM in 1973 and has held a variety of jobs within the company, providing support for customers in hardware and software. He currently works in customer support and solutions.

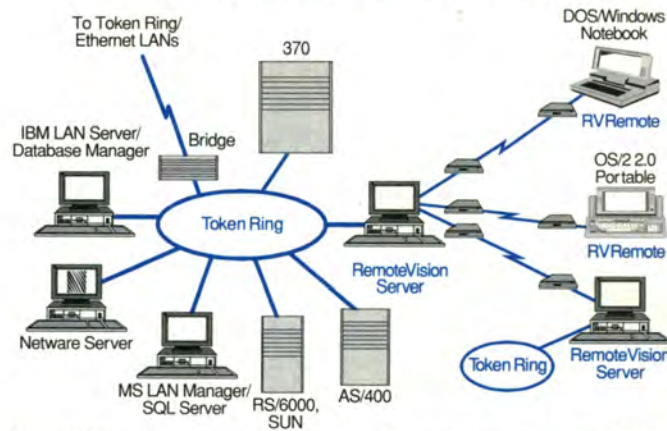
Shaun T. Jones, IBM Programming Systems Division, Warwick Software Development Laboratory, 1 Birmingham Road, Warwick, England, CV34 5JL. Jones joined IBM in 1978 and has held a variety of systems engineering positions in direct software marketing. He is a senior market planner for the Warwick Lab and is currently on international assignment in the U.S.

William G. (Bill) King, Jr., IBM Programming Systems Division, Warwick Software Development Laboratory Site Volumes Center, 3405 W. Dr. Martin Luther King, Jr. Blvd., Tampa, Fla. 33607. King joined IBM in 1979 and has worked with IBM's Federal Systems Division on

NASA's space shuttle program. Currently, he is responsible for implementing the Warwick Lab's business partner technical support program in the U.S. He holds a B.S. in computer and information sciences from the University of Florida.



Introducing RemoteVision.TM The Complete Software Solution For Remote Networking.



- Extend LANs transparently over asynchronous modems to remote workstations and workgroups.
- Run DOS, WindowsTM, & OS/2TM apps written to multiple protocol stacks on remote nodes.
- Operate remote computers as full-function LAN nodes.
- Connect to remote offices with LAN-to-LAN dial-up networking.
- Add remote connectivity services using existing LAN machines.
- Access remote PC servers and hosts without additional host gateways.
- No specialized hardware/adapters.

RemoteVision makes it easy and affordable for DOS, Windows and OS/2 users in remote locations to connect to Token Ring or other LANs using dial-up modems. Response time remains fast, applications don't need changing, and users won't need retraining. So you can turn your remote machines into full-function workstations. With RemoteVision. To immediately find out more, or to take advantage of our "Try Now, Pay Later" 30-day money-back guaranteed trial offer, call today.

REFERENCES

IBM Personal AS/2 V3 Installing and Supporting (IBM Doc. SH45-5501-00)

IBM Personal AS/2 V3 Quickstart: Examples, Hints and Tips (IBM Doc. SH45-5502-00)

IBM Personal AS/2 V3 Getting Started with Builder/2 (IBM Doc. SH45-5503-00)

IBM Personal AS/2 V3 Builder/2 Reference (IBM Doc. SH45-5504-00)



1265 Montecito Ave., Ste. 101, Mountain View, CA 94043
Tel: (415) 965-8607, Fax info: (415) 965-8607, (then press 2)
Trademarks are from their respective companies. ©1993 Token Technology, Inc. 001-0



Client/Server

Distributed Application/2 (DA/2) can help program client/server or interprocess communication in OS/2 2.0 or higher. DA/2 provides a simple set of APIs that allows communication between programs running on OS/2 systems in a network. **BY ED VAN VLIET, STEVE WASLESKI, RICK BLEVINS, and ANDREI MALACINSKI**

Client/Server Programming with Distributed Application/2

Anyone who has written networked client/server applications knows it is difficult to choose and use a specific communication protocol. "Which protocol do I need to code and test my application? If I choose protocol X," developers ask, "will it be available in all the networks I support?"



(l. to r.) Andrei Malacinski, Steve Wasleski, Rick Blevins, and Ed Van Vliet

These and many other questions must be answered while designing and building a client/server application. Once you have chosen a protocol, you must become sufficiently proficient to handle most cases that arise. Communication protocols can

be difficult to learn. Most programmers must endure a steep learning curve before becoming

communications code of an application.

DISTRIBUTED APPLICATION/2

Distributed Application/2 (DA/2) is a layer placed above communication protocols that provides a simple, consistent way to program interprocess and network communication into a client/server application under OS/2 2.0 or higher. Figure 1 shows an overview of the DA/2 structure.

DA/2 is designed to hide the complexity of its supported communications protocols, allowing you to select a communications protocol at run time without changing code and to program in either C or REXX. DA/2 supports the following communication protocols:

- Advanced program-to-program communication (APPC)
- NetBIOS for IBM or Novell LAN configurations
- OS/2 named pipes.

System requirements for DA/2 are OS/2 2.0 or later, 1MB of disk space, and 0.5MB of memory above OS/2.

THE FOUR BASIC INTERFACES

The DA/2 interface is composed of seven APIs. Four APIs support data transfer via `DAOpen`, `DAClose`, `DARead`, and `DAWrite`. The APIs provide a full duplex path between the client and the server, adding flexibility to program design. These four APIs are described in Table 1.

Even after a given protocol is mastered, user requirements could change the underlying communications code.

familiar with the nuances of a given communication protocol. Even after a given protocol is mastered, user requirements could change the network being used, with resulting changes to the underlying com-



Figure 2 shows a simple conversation between a client and server. The client opens the conversation, writes data, then calls `DARRead` to wait for data. In Figure 3, the server accepts the conversation with a call to `DAOpen`, reads the data sent by the client, then writes additional data to the client.

MULTIPLE CLIENT SUPPORT

Three event APIs, described in Table 2, are available to simplify handling multiple conversations. These APIs are especially useful within a server application that supports several clients concurrently.

The `DAWaitOnEvent` API reports either an incoming conversation request or an incoming data event to the caller, allowing you to specify specific event types. For an incoming conversation to be reported, the connection name for the conversation must have been identified on a prior call to `DAIdentifyResource`. Events of each type are queued in FIFO order, with conversation requests having a higher priority than data events. Data events for all conversations are reported by `DAWaitOnEvent`, regardless of whether the current process is the initiator or acceptor of the conversation.

A call to `DAWaitOnEvent` reports the arrival of data or a pending conversation; to actually receive the data, a `DARRead` call is necessary. To start the pending conversation, the `ACCEPT` option must be chosen in the `DAOpen` API.

Figure 4 shows a complete REXX server (minus error checking) that supports multiple concurrent clients. This server can be used with the REXX client example in Figure 2.

ADVANTAGES OF DA/2

- *Ease of use*—Each language binding in DA/2 was designed to accommodate the programming style of that language. Option parameters are character strings for REXX and `UNSIGNED LONG` for C.
- *Run-time protocol selection*—The application is not bound to a specific communication protocol when it is built. Rather, the protocol is chosen with the DA/2 connection profile editor just before the application is run.

- *Single workstation development*—Client/server applications can be built and tested in a single machine without APPC or NetBIOS installed.

NATIVE APPC APPLICATION COMMUNICATION

In addition to transparently supporting named pipes, NetBIOS, and APPC when both partners use DA/2, DA/2 provides a method to talk to native APPC applications.

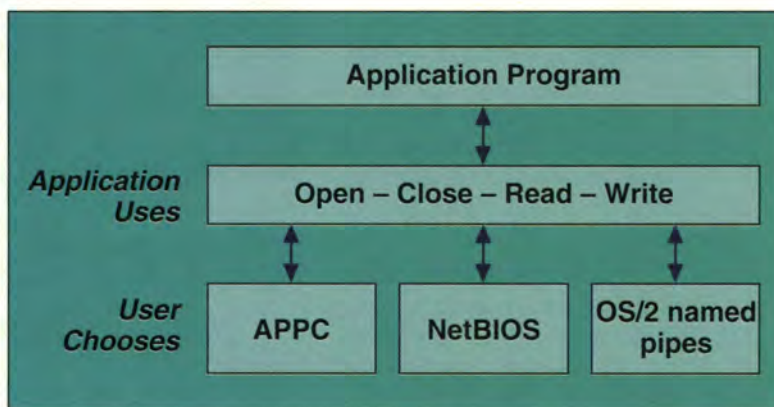


Figure 1. DA/2 approach to communications programming

<code>DAOpen</code>	Makes a connection between two conversation partners
<code>DAClose</code>	Terminates a connection
<code>DARRead</code>	Receives data from a conversation partner
<code>DAWrite</code>	Sends data to a conversation partner

Table 1. Data movement APIs

You can choose native APPC support by selecting either the “Native APPC—Confirmation” or “Native APPC—No Confirmation” radio buttons on the second APPC page of the Connection Profile Editor notebook.

Unlike with normal DA/2 support, when using the native APPC support it is necessary to be aware if a partner is using APPC. In this case, the connection is no longer full duplex, and some APPC half-duplex state considerations arise.

Considerations are made for APPC confirmation processing by providing an option for `DAWrite`



and status information for DAREad.

Regular DA/2 and native APPC connections can appear in the same program. DA/2 event APIs can also be used with native APPC support. For example, the same DAWaitOnEvent call signals data events from any one of a combination of regular DA/2 and native APPC connections.

SUPPORT FOR THE IMS LU 6.1 ADAPTOR FOR LU 6.2 APPLICATIONS

Support for native APPC applications also enables a DA/2 client to communicate with an Information Management System (IMS) transaction. To invoke synchronous IMS transactions, use a simple DAOpen, DAWrite, DAREad, and DAClose sequence.

To invoke asynchronous IMS transactions, use DAOpen, DAWrite, and DAClose. The asynchronous reply will return via the IMSASync transaction program on your workstation. You can write the IMSASync transaction using DAOpen, DAREad, and DAClose. As before, other regular DA/2 and native APPC connections can be active in the same program, and the DA/2 event APIs can be used.

THE CONNECTION PROFILE EDITOR

DA/2 keeps run-time connection information in connection profiles, which are stored in a binary file that must be located in a directory listed on the DPATH of any system that uses DA/2. Profiles can also be shared on a LAN drive.

DA/2 supplies a configuration tool called the Connection Profile Editor to create and maintain connection profiles. This editor, like most, accepts as an optional parameter the name of a file containing connection profiles. During DA/2 installation, a program object is created with an association that lets users double-click on a *.CP file to start the editor. The editor is shown in Figure 5.

The editor uses the CUA 1991 user interface architecture, including the drag-and-drop technique, to facilitate editing and management of the connec-

```

/* Sample REXX Client */
Name = "SAMPLCON" /* Connection name */

/* Load external functions, load variables into variable pool. */
call rxfuncadd 'DALoadFuncs', 'epfdarex', 'DALoadFuncs'
rc = DALoadFuncs() /* load all other DA/2 functions */
rc = DALoadVars() /* load DA/2 return code values */

/* Open a connection to the server. */
rc = DAOpen(handle, Name, 'DA_INIT', "FeedBack") /* Pass in name */
/* get back handle */

/* Writing data to the server. */
Options = 'DA_WRITE_DATA' /* initialize options */
Respid = 0 /* initialize response id */
Buffer = 'This is the data written to server (can be any length)'

rc = DAWrite(handle, Buffer, Options, "Respid", "FeedBack")

/* Read 100 bytes */
rc = DAREad(handle, "buff", 100, 'DA_WAIT', "Respid", "bytesread",
"data", "conn", FeedBack.)

/* Close the connection. */
rc = DAClose(handle, FeedBack.)

exit

```

Figure 2. A sample DA/2 REXX client

DAIdentifyResource	Starts queuing incoming conversation requests for a specific connection name
DAWaitOnEvent	Reports incoming events
DATerminateResources	Discontinues queuing incoming conversation requests

Table 2. Event notification APIs

tion profiles. With each setting, you can access online help by pressing the F1 key when focus is on the desired field.

Once in the editor, you can select a profile to edit, double-clicking on it to bring up a settings notebook for the


```

/* Sample REXX Server */
Name = "SAMPLCON" /* Connection name */
/* Load external functions, load variables into variable pool. */
call rxfuncadd 'DALoadFuncs', 'epfdarex', 'DALoadFuncs'
rc = DALoadFuncs() /* load all other DA/2 functions */
rc = DALoadVars() /* load DA/2 return code values */
/* Open a connection to the client. */
rc = DAOpen(handle, Name, 'DA_ACCEPT', "FeedBack") /* Pass name */
/* Read data from the client. */
Respid = 0
rc = DARead(handle, "buff", 100, 'DA_WAIT', "Respid", "bytesread",

```

Figure 3. A sample DA/2 REXX server (continued on page 70)

specified profiles. The first page of the settings notebook contains general information about the connection, including the name of the connection and the protocol to be used. Each of the notebook pages that follow is devoted to information specific to the protocol selected. Editor settings are shown in Figure 6.

The second page is used to add named pipes and NetBIOS connection information and to specify whether the server is queued or nonqueued. If "nonqueued" is selected, a new server is started each time a connection is opened. If you select "queued," the connection is made to an already running server; in this case, a new server is started only if a queued server is not already running.

The third and fourth notebook



How are you going to test your client/server apps? Check one:

- Manually
- Partially
- The Softbridge
Automated Test Facility

It's not a trick question, but testing client/server applications can be a tricky business. Traditional, manual methods can't stand up to the complexities of client/server. And most automated testing tools provide only a partial answer — concentrating on the GUI desktop, and leaving the rest of the client/server testing puzzle for you to piece together.

Only the Softbridge Automated Test Facility (ATF) is designed to cover all aspects of client/server testing — GUI, distributed systems, legacy applications. If you're doing client/server development, maybe it's time you checked out ATF. To find out how ATF can help with your client/server testing needs, call 1-800-955-9190.



**The Softbridge
Automated Test Facility**

Softbridge, Inc.
125 CambridgePark Dr.
Cambridge, MA 02140
617-576-2257 (Phone)
617-864-7747 (FAX)



pages are devoted to APPC-specific information. This includes configuration information for APPC, such as the remote transaction program name, mode name, local logical unit (LU) name, partner LU name, and other profile information specific to an APPC connection such as data conversion and whether your partner is using DA/2 APIs or native APPC.

With the Connection Profile Editor, you can manage multiple connection profiles using drag-and-drop. An administrator may want to maintain a master file of connection profiles to distribute as needed.

```
        "data", "conn", FeedBack.)

/* Writing data to the client.                               */
Options = 'DA_WRITE_DATA'
Buffer = 'This is data to write to the Client.'

Respid = 0
rc = DAWrite(handle, Buffer, Options, "Respid", "FeedBack")

/* Close the connection.                                    */

rc = DAClose(handle, FeedBack.)
exit
```

Figure 3. A sample DA/2 REXX server (continued from page 69)

```
/* REXX Server supporting multiple clients sample. */
Name = "SAMPLCON" /* Connection name */

/* Load external functions, load variables into variable pool. */
call rxfuncadd 'DALoadFuncs', 'epfdarex', 'DALoadFuncs'
rc = DALoadFuncs() /* load all other DA/2 functions */
rc = DALoadVars() /* load DA/2 return code values */

rc = DAIdentifyResource( Name, "FeedBack");

do FOREVER
  rc = DAWaitOnEvent( "outName", "outHandle", 'DA_ALL_EVENTS', "eventType", ,
                    "dataLength", "responseID", "FeedBack");
  if rc = 0 then
    do
      if eventType = 'DA_CONNECTION' then /* open the incoming conversation */
        do
          rc = DAOpen("handle", outName, 'DA_ACCEPT', "FeedBack");
        end
      else if eventType = 'DA_DATA' then /* handle the incoming data event */
        do
          responseID = 0
          rc = DAREad(outHandle, "buff", dataLength, 'DA_WAIT', "responseID", ,
                    "bytesRead", "dataStatus", "connStatus", "FeedBack");
          if connStatus = 'DA_CLOSE' then
            do
              rc = DAClose(outHandle, "FeedBack");
            end
          else do
            /* ...process the data just read and send back a reply */
            rc = DAWrite(outHandle, buff, 'DA_WRITE_DATA', "responseID", ,
                        "FeedBack");
          end
        end /* end if eventType = 'DA_DATA' */
      end /* end if rc = 0 */
    end /* end do FOREVER loop */
```

Figure 4. Multiple client server

WHY WASTE VALUABLE TIME...

WHEN YOU CAN GO STRAIGHT TO THE SOURCE?



Introducing the only magazine exclusively devoted to software developers working in the OS/2 environment.

- THE SOURCE** for OS/2 tips and techniques, direct from Big Blue to you.
- THE SOURCE** for those working with everything from LANs to multimedia to DBMS architecture.
- THE SOURCE** for valuable news and information for OS/2 application developers.
- THE SOURCE** for new software development ideas and products.

Subscribe now and pay just \$39.95 for six big issues, delivered to your office desk - straight from the source.

YES! Send me a full year of IBM OS/2 Developer. I'll pay just \$39.95 for six issues!

Name _____ Title _____

Company _____

Address _____

City/State/Zip _____

Charge My Credit Card : VISA MC AMEX

Card Number: _____

Expiration Date: _____ Signature: _____

You may pre-pay by enclosing a check or money order made out to IBM OS/2 Developer, or by enclosing VISA, MasterCard or American Express number, expiration date, and your signature. Delivery of first issue will be in 6-8 weeks. Canadian and international surface mail orders, add \$16 per year for postage. International surface mail is an additional \$30 per year for postage. Checks must be in U.S. funds.

SPEED YOUR ORDER!
1-800-WANT-0s2
1-708-647-5960
(OUTSIDE THE U.S.)

FAX:
1-708-647-0537

MAIL THIS FORM TO:
IBM OS/2 DEVELOPER
P.O. Box 1079
SKOKIE, IL, 60606



CONCLUSION

DA/2 provides the communications needed for writing client/server applications in most OS/2 environments. Full duplex paths are provided for APPC, IBM and Novell NetBIOS, and OS/2 named pipes.

Ricky Blevins, IBM Programming Systems, P.O. Box 60000, Cary, N.C. 27512. Blevins is a staff programmer with IBM Application Platform Products, currently working as a member of the DA/2 development team. He joined IBM in 1981 and has since held programming positions in communications and application development tools. Blevins holds a B.A. in mathematics from Berea College and a M.S. in mathematics from the University of Kentucky.

Andrei Malacinski, IBM Programming Systems, P.O. Box 60000, Cary, N.C. 27512. Malacinski is an associate programmer with IBM Application Platform Products, currently working as a member of the DA/2 team. He was responsible for the design and implementation of the DA/2 REXX programming interface and user interface of the DA/2 connection profile editor. Malacinski joined IBM in 1990 after receiving degrees in mathematics and computer science from Indiana University.

Ed Van Vliet, IBM Programming Systems, P.O. Box 60000, Cary, N.C. 27512. Van Vliet is an advisory programmer with IBM Application Platform Products and currently the project leader for the DA/2 team. During his IBM career, Van Vliet has held architectural and project leadership positions for large systems and workstation communications projects.

Steve Wasleski, IBM Programming Systems, P.O. Box 60000, Cary, N.C. 27512. Wasleski joined IBM in 1987 and is currently on the DA/2 development team. His responsibilities include development and support of DA/2. In his six years with IBM, his work has included client/server computing, GUIs, and communications. He holds a B.S. in computer science from the University of Missouri, Rolla.

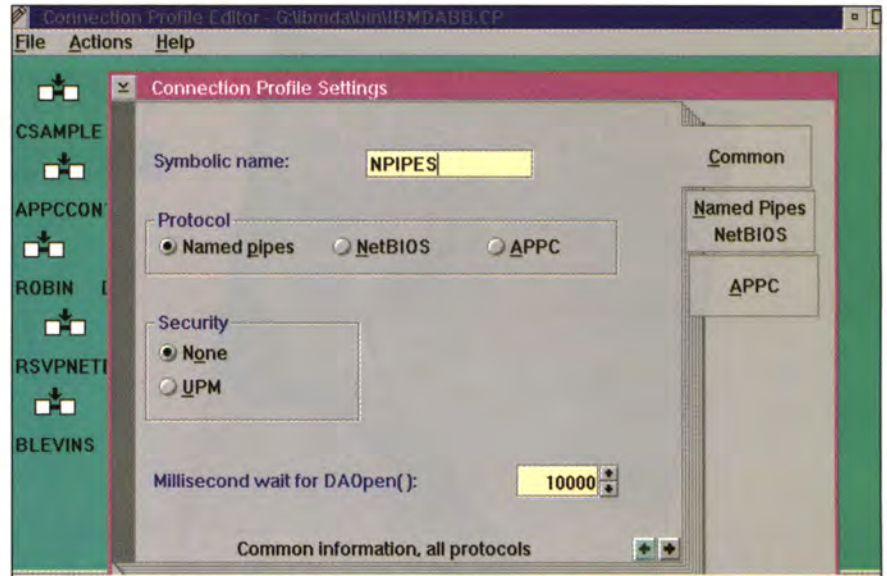


Figure 5. Connection profile editor, common information

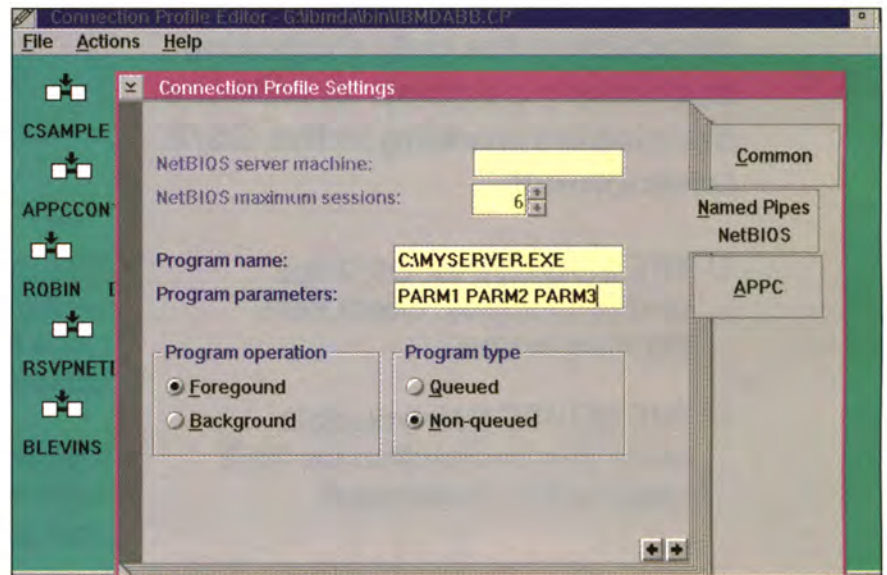


Figure 6. Connection profile editor settings

REFERENCES

More detailed information about the APIs is available from the programmer's guide .INF files contained in the product.

DA/2 is available in American English and Japanese. For more information, fax inquiries to (919) 469-7423 or call the OS/2 BBS TalkLink at (800) 547-1283. Questions are also addressed on the OS/2 Developer forum (OS2DF2) on CompuServe.



As client/server computing becomes widely used, corporations and developers must handle their client/server needs. This article explains Easel Workbench and explores its advantages for corporate developers.

BY CHRIS BROOKINS

Easel Workbench: Robust Client/Server Development

Client/server computing has been emerging in the computer industry for the last several years, but it is only in the past year or so that the idea of client/server has caught on within the ranks of corporate developers. Information systems organizations see client/server as their way of exploiting the computing power and information resources of their enterprise. But increasing application needs, combined with steadily more complex computing environments, has led to a bewildering array of implementation options associated with client/server.

CLIENT/SERVER FRAMEWORK

Defining Client/Server. Client/server technology is a form of computing that divides application function among networked computing resources, some of which are PCs or workstations commonly having a graphical interface. Within this definition there exist five forms or models of client/server computing, shown in Figure 1:

- Database-server
- Transaction-server
- Peer-distributed
- Presentation-distributed
- Presentation and logic

Database Server. For applications that need to access data from a variety of back-end data sources, there is the database-server model of client/server. In this model, SQL is simply embedded in the client application and sent to the database server for process-

ing. Since the interface between the client and the server is strictly SQL-based, processing on the server is limited to what SQL can perform. Often, results are returned and processed further at the client.

Transaction Server. Applications such as order entry, process control, and other mission-critical online transaction processing applications require consistent, minimal response times. The transaction-server model is optimal for OLTP requirements because it allows processing to be distributed and balanced between the client and the server. Data-intensive operations can take place at the server while user intensive tasks can be handled at the client. In addition, this model eliminates unnecessary network traffic, since only the request and the final result are sent between client and server. Data processing that can't be accomplished with SQL, such as conditional logic or complex aggregate functions, is performed on the server, eliminating the need for data to be sent over the network for client processing.

A transaction is a single logical unit of work that may be as simple as an SQL statement or as complex as a combination of SQL statements, processing logic, and non-SQL data manipulation. The client initiates transactions such as `UpdateSystemBalances` by name, hiding the underlying complexity from client developers and allowing transactions to be maintained independently of client applications.

Peer. The peer model contains all the benefits of the transaction-server model with the added flexibility



Chris Brookins



that at any time the client and server can switch roles. This model is useful when an application must respond to unforeseen external events. For example, in a process monitoring system, a stockbroker application can request stock price information from the host server or the host server can send unsolicited data to the desktop if the stock price reaches a specified threshold.

Distributed Presentation And Logic. Existing host-based legacy systems have become easier to use with the advent of PC-based graphical user interfaces, which brought about this form of client/server computing. The pure distributed presentation model can improve usability by simply remapping existing host

screens into equivalent graphical interfaces with pulldown menus, entry fields, and other dialogue controls. Host-based applications in which codes must be entered, selection lists are not available, or the screens are difficult to understand become more user-friendly.

It is possible to transform existing host-based systems into new systems that leverage the PC's power by adopting the distributed presentation and logic model. In this model, a new system is developed that utilizes existing services presented by the host system. Processing logic at the PC can off-load host processing or introduce completely new functionality such as image, sound, or voice. This approach borrows from the database server model in that

the host application is fixed and predefined, although here the interface is through an existing path of host screen-to-screen navigation. Since application logic is split and executed between the server and the client, this model also contains some benefits of the transaction-server model, shown in Figure 2.

Hybrid Applications. Corporate data infrastructures do not change overnight, nor do existing systems. A hybrid approach allows a developer to combine more than one client/server model in an application to meet the real needs of a business. For example, the goal may be to move the entire system to a Database Manager-based solution over time, but it is too time-consuming and expensive to reengineer everything at once. A hybrid application in this case would communicate with the existing system through distributed presentation support, as well as add new application components on the LAN that are accessed through transaction server support.

EASEL WORKBENCH AND CLIENT/SERVER DEVELOPMENT

Easel Workbench was designed to help application developers best use the power of client/server computing. Workbench includes an integrated, 32-bit, OS/2-based development environment, the event-driven EASEL language, an extensive variety of connectivity options, and a production system for the target user platform (OS/2, Windows, or DOS). Because the Workbench supports a powerful client/server framework, its connectivity options can integrate data from nearly any source.

Support for the Database-Server Model. For applications that need to access data from a wide variety of back-end data sources, Workbench provides support for the database-server model of client/server. Like all Easel client/server support, database processing occurs entirely on a separate thread, allowing the Easel client application to continue

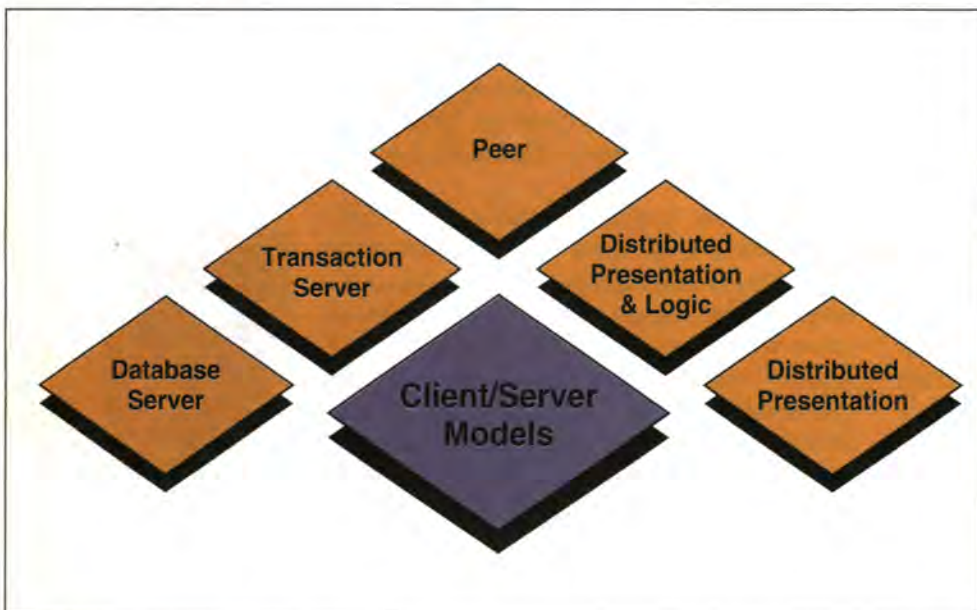


Figure 1. Client/server models

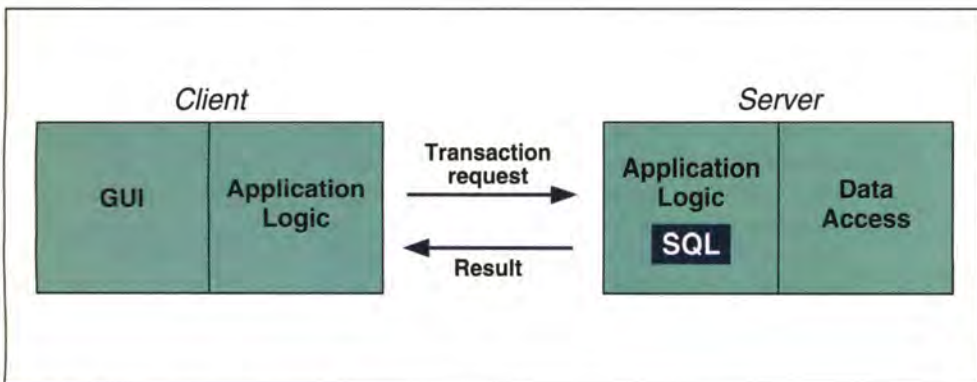


Figure 2. Transaction server model

operation. When results are returned, an Easel event, or "response," is triggered. Database server results can be retrieved record by record, or written directly to a local file for bulk processing.

Easel provides direct database server support for IBM's Database Manager, the Oracle Server, Microsoft/Sybase SQL Server, and all databases supported by IBI's EDA/SQL support. Easel's support for the Database Manager is DRDA-compliant; where DDCS/2 is installed, applications can communicate transparently with all SAA databases including DB2, SQL/400, and SQL/DS. To insulate developers from the complexity of each unique database API, all Easel database drivers provide a high-level common programming interface, which lets a programmer change the back-end database without changing a client application. Multiple connections to the same or different database servers can also exist simultaneously, allowing the integration of information from multiple sources.

Support for the Transaction-Server Model.

Easel provides transaction server support with three main technologies: stored procedures, CICS-OS/2, and the Easel Transaction Server (ETS) Toolkit, shown in Figure 3.

Stored procedures are transactions containing compiled SQL and processing logic stored within the database server. For database servers that support stored procedures such as Microsoft/Sybase and Oracle SQL Server, Easel applications can initiate a procedure on a separate thread. When the procedure has completed processing, the application is notified by way of an Easel event

Easel provides a high-level interface to CICS-OS/2 that makes it easy for an application to initiate transactions residing on CICS-OS/2 workstations or hosts. CICS transactions can be initiated with a simple subroutine call to CICS's Extended Call Interface, including a transaction name and an input and output structure. The request is processed on a separate thread and the

application is notified upon completion. At that time, the output structure is populated with any data returned from the database.

Easel Workbench includes the ETS Toolkit, which allows developers to create and access OS/2-based transactions with a remote procedure call mechanism for all SAA databases

(Database Manager, DB2, SQL/400, and SQL/DS). ETS transactions, based on the Database Manager's application remote interface, are written by creating procedures of SQL and processing logic within an ETS C or COBOL template. Easel applications can initiate a server transaction with the ETSExecute subroutine. The request is delivered to

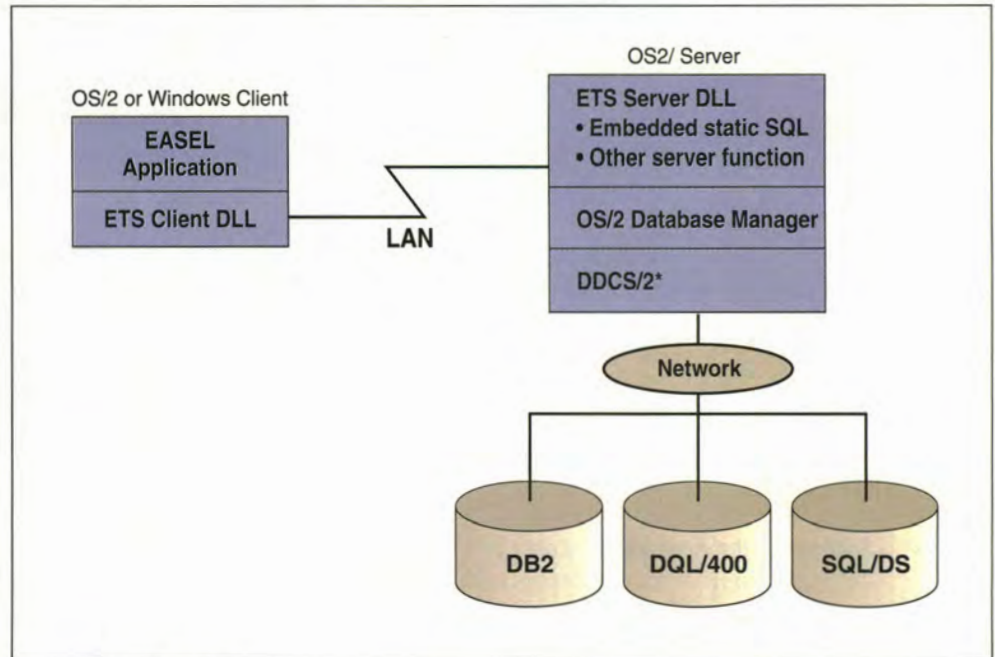


Figure 3. The ETS Toolkit

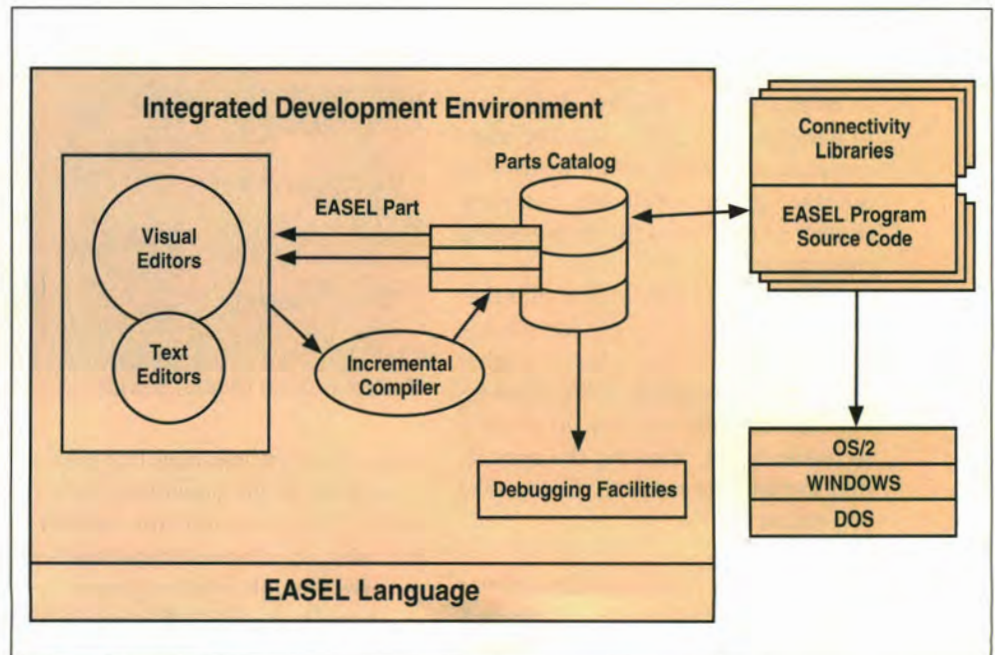


Figure 4. Integrated development environment



EVENT TYPES AND RESPONSES

Event Types	Example	Example EASEL code
User initiated events	User selected a button named OK	response to OK
Communications/ external events	An ETS transaction has completed response to stimulus	ETS_DONE
Time events	5 seconds have elapsed	response to interval 5
Special EASEL events	EASEL application has started	response to start

Table 1. *Easel event types and responses*

the OS/2 server on a separate thread, with NetBIOS on OS/2 Extended Services or the APPC SQL00 protocol with Remote Data Services. The server procedure can execute directly against the Database Manager or, if DDCCS/2 is installed, can connect to DB2, SQL/400, or SQL/DS.

Support for the Peer Model. Easel provides peer support for developers with the APPC protocol. To provide the most control while shielding complexity, there are two APPC interfaces, automatic and controlled. The automatic interface takes care of the necessary acknowledgments and implicitly switches conversation states from send to receive without forcing developers to perform subroutine calls. For more complex applications, the controlled interface gives developers complete control over the details of the APPC conversation.

APPC events appear like all other Easel events—as uniquely identifiable external responses. Like all other client/server support, APPC messaging and communications occur on a separate thread, allowing the application to continue running during peer processing.

Support for the Distributed-Presentation Model. Easel supports the distributed-presentation models with high-level support for HLLAPI, VT-100, and asyn-

USING ACTION STATEMENTS

Modify/query object attributes

Create or delete objects

Call other EASEL subroutines or subroutines within a DLL

Control program flow: for loops, while loops, if/then/else

Perform string processing or integer and floating point arithmetic

Read and write text, image, or structured data to and from disk

Table 2. *Ways to use Easel action statements*

EXTERNAL LIBRARIES PROVIDED WITH WORKBENCH

Local PC Processing	DDE, File I/O
Communications	APPC, CICS OS/2, 3270/5250, VT-100
Database Access	DB Manager (DB2, SQL/DS, SQL/400 via DRDA), Sybase/Microsoft SQL Server, Oracle, and EDA/SQL (optional)
Miscellaneous	Math Functions, Date Functions, Help Manager Functions

Table 3. *External libraries provided with Workbench*

chronous communications that mask the complexity of the underlying API. For example, applications are notified of incoming asynchronous data from CompuServe through Easel responses.

Through the HLLAPI provided by most PC emulators, Easel applications can access 3270 and 5250 applications.

Easel can also access applications on various platforms through other protocols, with Software Corporation of America's TalkThru emulator. The functions available from HLLAPI support include:

- To retrieve host data: ReadField, GetFieldAttribute, ReadScreen, ReadLine.

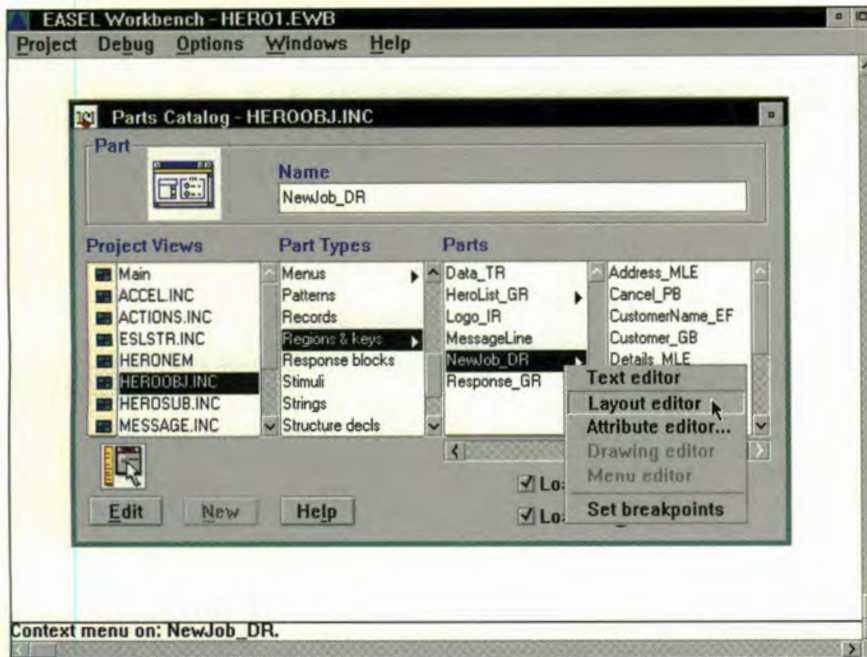


Figure 5. The parts catalogue

- To write data to host: WriteField, PressKey, EnterString.
- For host synchronization: WatchForStringAt, WatchForCustorAt.
- Other: EmulateAndWatch, GetSessionStatus, SetEmulatorWindow.



Support for Hybrid Applications. A single graphical interface can implement any number of client/server models that access data throughout an enterprise. Through its support for multiple models, a single Easel application can update diverse systems with the same data. Many companies use Easel applications to transparently keep multiple systems in sync.

THE EASEL LANGUAGE

The EASEL language is a high-level, event-driven language designed for

BackMaster for OS/2 2.x

32-Bit Backup for OS/2

32-Bit Multi-Threaded PM Backup Utility.
 Supports FAT and HPFS File Systems.
 Handles Long Names, and Extended Attributes.
 Backup WPS, Desktop, and System Files.
 Total/Partial/Incremental/Unattended Backups.
 Backup to Floppy Disk and QIC-40/80 Tape Drives.
 Can Read QIC-40/80 DOS Tapes.
 Easily Migrate your DOS files to OS/2 using Tape.
 We Support CMS Jumbo & Generic QIC-40/80
 Runs in the Background.
 Uses Industry Standard STAC LZS Data Compression.

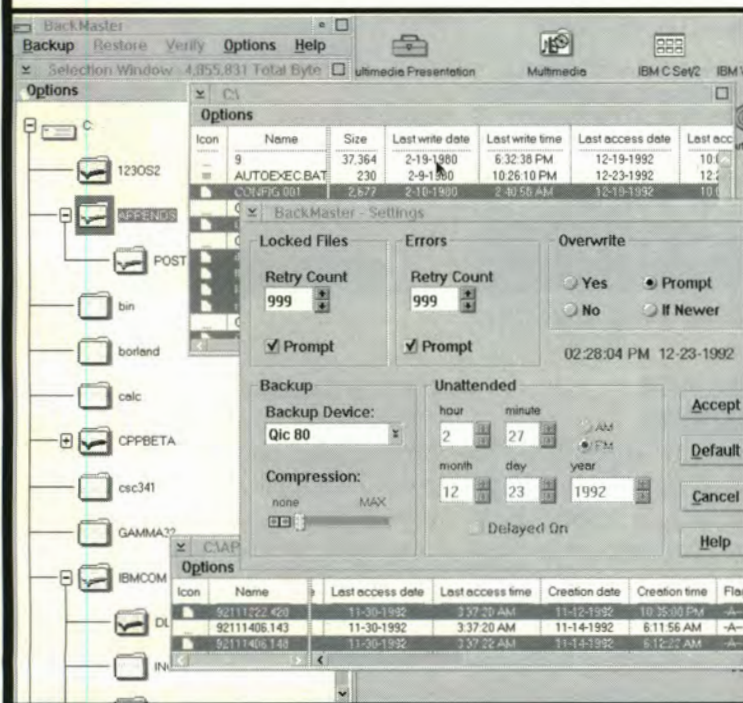
Only \$79.95

**Texas Residents add 8 1/4% Sales Tax
 VISA/MC/COD Welcome**

Call our BBS for a Free Demo Version

MSR Development

Rt 7 #6409, Nacogdoches, TX 75961
 Ph (409) 564-1862, BBS (409)560-5970





developing applications with a graphical interface for all client/server models. Easel applications can run under OS/2, Windows and DOS. Under OS/2 2.0, they use the 32-bit architecture and graphics engine.

EASEL shields developers from the complexities of (and differences between) the underlying operating platform and connectivity APIs. Its complete set of programming constructs provides facilities for developing and controlling an application's graphical interface, communications, database access, and processing logic.

EASEL Objects. At the core of the EASEL language are objects that correspond to other objects with visual attributes. These objects include familiar OS/2 and Windows dialogue boxes and controls, as well as special Easel objects such as image regions (windows that hold bitmap images) and graphical regions (windows that hold scalable vector graphics). All objects have attributes such as visibility, color, and size that can be queried and changed at runtime. Objects can be created either visually in Easel Workbench or dynamically, based on data received during runtime. Objects can also be grouped into classes to which a single action can be performed (such as making them all invisible).

Event-Driven Programming. Easel's non-procedural, event-driven nature allows developers to quickly prototype and incrementally add new functionality as needed. To respond to user and system activities, developers write responses to events of the types shown in Table 1.

Robust Language Constructs. Within a response, developers can choose from a set of action statements that control application behavior. Table 2 lists ways to use these actions.

Extensible. Developers can also access routines in external libraries or modules. Easel provides a complete set of libraries, but developers can create their own. Table 3 lists some of the external libraries provided with Workbench.

VISUAL EDITORS	
Layout Editor	Builds all windows and dialogue controls using the tool palette and various alignment facilities
Menu Editor	Defines CUA menus including accelerators and mnemonics
Attribute Editor	Specifies all object attributes such as color and visibility
Drawing Editor	Creates custom vector graphics using the line, box, ellipse, arc and shape tools in conjunction with zoom and alignment facilities

Table 4. Easel Workbench tools for all phases of the development cycle

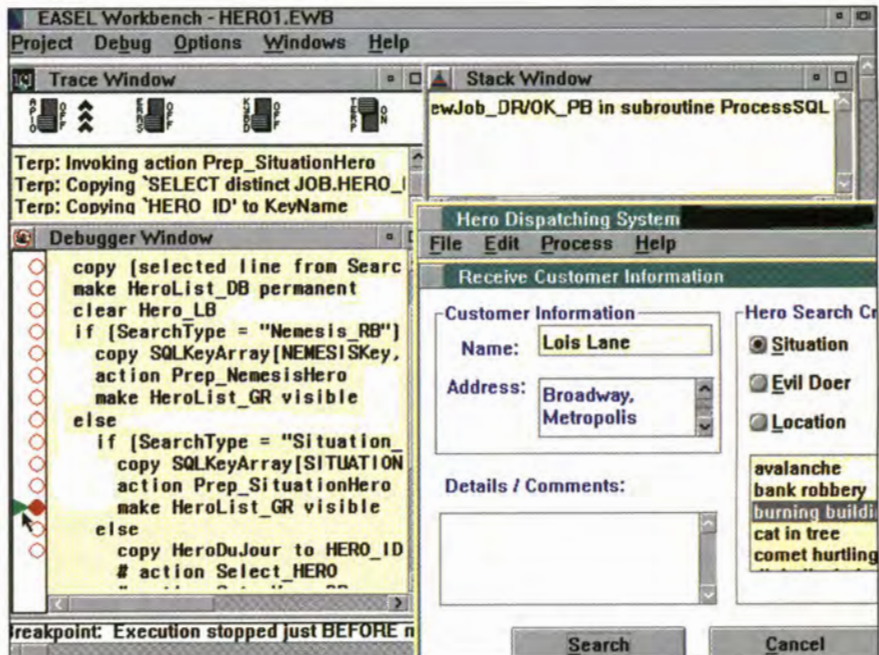


Figure 6. Workbench debugging facilities

IMPLEMENTING WITH EASEL WORKBENCH

EASEL, along with support for its client/server models, is the foundation for the Workbench, a 32-bit, OS/2-based integrated development environment in which developers can visually define a user interface, add application logic, and test and debug Easel applications. This environment is shown in Figure 4.

Parts Catalogue. Central to the Work-

bench is the parts catalogue (shown in Figure 5), which allows developers to browse, edit, and create the components of an application with a listing of project views, or modules, and a hierarchical presentation of application parts. Selecting a part loads and launches the associated editor. Double-clicking on any part, for example, launches the appropriate visual editor for quick modifications. All modifications are incorporated into the application by an incremental compiler that speeds up

the development cycle. The parts catalogue also displays project views, which are stored in ASCII files that provide simple integration with version control systems.

Visual Editors. Workbench's visual editors allow you to define the interface in a WYSIWYG environment. CUA monitoring can check an interface against Common User Access rules of interface design. The visual editors are listed in Table 4.

Debugging Facilities. Application logic is added with Easel's integrated text editor. Debugging facilities, shown in Figure 6, include:

- **Source-Level Debugger.** This facility displays executing source code.

Developers can visually step through statement-by-statement or by individual procedure, set breakpoints, and change runtime values.

- **Trace Windows.** This facility displays a detailed record of an application's execution. Developers may open several trace windows and set filters to customize the type of information monitored in each window.
- **Stack Window.** This facility displays the level of nesting that occurs during program execution. Developers may inspect and change variables on the stack.

Example. The insurance representative workstation shown in Figure 7 integrates data from two client/server

models and makes extensive use of images. When compiled for OS/2, the application automatically exploits the 32-bit API.

The work of this application was formerly a paper-based process in which a representative entered appraisal forms into an aging host-based system. The organization wanted to transfer part of the system to Database Manager and reengineer the entire process with PC and imaging technology.

The Easel system developed to address this problem controls an image bank on a minicomputer, which responds to the clerk pushing a button by issuing commands to download the next scanned appraisal form and item photograph in the processing queue. After reviewing the form in a window,



dbfLIB

Programmer's Library

Includes
32-Bit
OS/2 2.0
DLL

Do you need uniform access to dBASE files from your custom DOS, Windows, and OS/2 applications?

Introducing **dbfLIB**, the dBASE file access library for DOS, Windows, and OS/2 programmers.

Features:

- * A single set of functions for DOS, Windows, OS/2 1.3, OS/2 2.0
- * Royalty free Dynamic Link Libraries for:
 - Windows 3.0 and Windows 3.1
 - OS/2 1.3 (16-bit DLL)
 - OS/2 2.0 (32-bit DLL)
- * Static libraries for DOS and OS/2 applications.
- * Meaningful function names familiar to the xBASE programmer.
- * File and record locking.
- * Handle-based calls allows multiple instances of the same dbf file (very useful in multi-threaded OS/2 programs).
- * Create, update, and index dBASE files.

dbfLIB includes Windows DLL, OS/2 1.3 DLL, OS/2 2.0 DLL, and static libraries for DOS and OS/2.



dSOFT Development Inc.
4710 Innsbruck Drive
Houston, TX 77066
(713) 537-0318



TLIBTM Version Control For DOS, OS/2 and Windows-NT

• The experts loved TLIB 4:

"...amazingly fast... TLIB is a great system." **PC Tech Journal**
 "TLIB has features and power to spare... TLIB is easy to use and the fastest of the reviewed packages." **Computer Language**
 "I will not program without it." **Uptime Magazine**

• Now TLIB 5.0 adds:

Automatic branching. Automatic version labeling across branches. User defined **promote** structures, for staged development. Exclusive whole-level **change migration** for customized software. N-way-tree version numbers. Branch and full locking. OS/2 & NT support.

• Plus the features they loved in TLIB 4:

Check-in/out locking. Branching, for parallel development. Keywords. Full binary file support (does not depend upon CRs in the file like other products). Wildcard and list-of-file support; can create lists by scanning source code for includes. Can merge (reconcile) multiple simultaneous changes and undo intermediate revisions. Network and WORM optical disk support. Mainframe-compatible delta generator for Pansophic, ADR, IBM, Sperry formats. Includes integrated PD MAKE by L. Dyer; also integrates with OpusTM MAKE, SlickTM MAKE, others.

MS-DOS **\$139**, OS/2 & NT (with MS-DOS) **\$195** + shipping.
 5 station network: MS-DOS \$419, OS/2 \$595. Call for other sizes.



Burton Systems Software

PO Box 4156, Cary, NC 27519 (919) 233-8128
 FAX: 233-0716



1993 CONFERENCE & EXHIBITION



Business Software
Solutions

CONFERENCE & EXHIBITION

OUR NEW GOOD LOOKS ARE NICE, BUT IT'S WHAT'S INSIDE THAT COUNTS.

The software industry is changing.

Businesses are rightsizing, new GUI platforms are in the news, and software and operating platforms are becoming ever more important variables in today's business computing equation.

You have a lot to keep up with today. So, we're retooling two great industry events to help you do just that.

WINDOWS & OS/2 BECOMES BUSINESS SOFTWARE SOLUTIONS

First, Windows & OS/2 has changed its name to Business Software Solutions. Why? To let you know we're keeping up

with your growing need to bring *cost effective GUI-based solutions to your business.*

At Business Software Solutions you'll get the *best of the old Windows & OS/2*



Conference. You'll still find the hottest software for today's key GUI-based platforms. You'll find test drive centers where you can try out apps before you buy. And you'll find many of the biggest name *industry gurus* telling you like it is about

what's hot and what's not.

But we've added so much more, you'll barely recognize the place!

First, we've *rebuilt the conference program from the ground up* to provide you with much more value. Whether you manage your own business, or work for one of the Fortune 500, you'll find the answers you need about today's GUI-based platforms and software.

You'll discover how to make *sound buying decisions* that insure your investment as technology changes. You'll learn whether or not it's a must to settle on just one platform for your business. You'll

learn how to set up and manage multiple platforms over a network.

Then, you'll find how to use the most *cost-effective training tools* and techniques to support and train your end-users. You'll discover how to give your enterprise easy access to *legacy data* on the desktop. And you'll learn how to build Windows and OS/2 solutions that meet the real needs of your end users.

On top of that, we've added a *Solutions Interchange Room* where you can trade ideas with peers, a *Solutions Theatre* for head-to-head vendor demos, and much more!

All in all, you'll find the whole event focused on how to help you *solve your biggest end-user computing problems* using the best of today's desktop technology PLUS you'll get a glimpse into what's coming next after Windows and OS/2.

SOFTWARE DEVELOPMENT '93 JOINS BUSINESS SOFTWARE SOLUTIONS

Take the newly refocused Business Software Solutions Conference and add *the largest and most respected technical conference* and show for anyone involved in building desktop or client-server applications. What have you got? A dynamic industry event that reflects the entire desktop software process from development to final applications and solutions!

SD has long been seen as the best place to get *unbiased, here's-how information on developing for the desktop*. With

over *150 courses* in tracks ranging from Windows, OS/2 and UNIX development, to Analysis & Design, C++, Client-Server development, User Interface Design and Management, you'll have a hard time choosing which sessions to attend.

And, again, that's not all. On the show floor you'll see the *best available*



COME SEE FOR YOURSELF. AUGUST 23-27 IN BOSTON AT THE HYNES CONVENTION CENTER.

development products from leading tools vendors. SD was the site of Borland's launch of Borland C++ last year, and Microsoft's Visual C++ and 40 other new products this February. This August you'll find more of the *hottest tools on the market*. And, you'll get an extra bonus. That's because your SD ticket also gets you into Business Software Solutions, the software solutions showcase of the summer.

WE CHANGED OUR LOOKS TO GET YOUR ATTENTION...

But we know that it's what's inside that counts. And whether you're interested in the solutions side or the development side of the desktop software equation, at Software Development and Business Software Solutions *you'll find answers* to more of your business software questions than ever before in two quality conferences and more than 250 exhibits.

To get detailed information on what you'll find when you arrive, return this

coupon, phone us at 415-905-2784 or fax us at 905-8100 *today*. We'll send you a complete conference catalog, and a ticket for a free show pass.

YES! TELL ME MORE!

Please send me more information about your two conferences & exhibitions being held concurrently in Boston, August 23-27, 1993.

- Business Software Solutions
 Software Development

NAME _____

TITLE _____

MAIL STOP _____

COMPANY _____

ADDRESS _____

CITY _____

STATE _____

ZIP _____

PHONE _____

FAX _____

Miller Freeman, Inc.
600 Harrison Street • San Francisco, CA 94107
415-905-2784 • FAX 415-905-8100

OS2D



the clerk enters the first few letters of the owner's last name into a dialogue that invokes an ETS transaction against the Database Manager to present a list of matching names. When the clerk selects a name, the system issues a transaction that returns the owner's complete profile, including a list of appraised items. The clerk then adds the item by entering information into the dialogue box, and Easel updates the existing host-based system through the distributed presentation and logic model. With other transactions, it updates the database with the new insurance premium. Finally, the clerk can send a form letter by pressing the Correspondence button (which initiates Microsoft Word, loads a form, and enters the owner's name and updated list of appraised items via DDE).

Chris Brookins, Easel Corp., 25 Corporate Drive, Burlington, Mass. 01803. Brookins is a product manager at Easel. He has been actively involved with the EASEL language and Easel Workbench for three years. Prior to joining Easel, Brookins was a manager of systems planning at General Electric. He holds a B.S. in computer science from Lehigh University.

Figure 7. Clerk's insurance workstation application

REFERENCES

Programming Guide for the EASEL Language, Easel Corp.

Using Easel Workbench, Easel Corp.

Orfali, Robert and Dan Harkey. *Client/Server Programming With OS/2 2.0*, 2d ed. New York: Van Nostrand Reinhold, 1992.





The ICSS Speech Recognition API helps programmers develop applications that recognize and process a continuous flow of spoken English words. This article explores the main functions and features of the ICSS software and provides a programming example. **BY VINCENT STANFORD and BRANDON BOOTH**

Speech-Enabling Your Applications With ICSS

The ICSS Developer's Program consists of prerelease software currently being system-tested by IBM. It is fully supported for program members; members give opinions, report bugs, and submit design change requests, which leads to improvements in process and product.

The ICSS Speech Recognition API, currently being tested under the program, helps programmers develop applications that recognize and process a continuous flow of spoken English words.

ICSS PRODUCT FEATURES

Speaker Independence. Though there are variations in human speech from person to person, unlike other speech recognition systems ICSS requires no speaker enrollment or user training. ICSS has been tested with a variety of accents and under high background noise conditions. Though performance varies, good recognition is possible in spite of a variety of accents and background conditions.

Continuous Speech. Users can speak at natural or rapid rates with successful recognition. This is important for command and transaction applications that require high-speed control of user interfaces as well as for high-speed data entry by those who find keyboards difficult or uncomfortable to use.

Large Vocabulary. ICSS provides recognition under multiple switchable contexts of up to 1,000 words each. With the context-switching feature, pre-loaded multiple contexts can extend the practical

reach of the system to complex real-world applications. The base dictionary contains over 20,000 words; users can add words without gathering additional speech samples or retraining the system.

Client/Server Metaphor. The recognition server can support multiple applications running on the same machine. The recognition engine serves as a serially reusable resource for a desktop of applications.

Unlike other speech recognition systems, ICSS requires no speaker enrollment or user training.

Grammar-Guided Recognition. ICSS is designed to listen for and process specific sets of words and phrases based on predefined grammars. Its recognition engine uses context-related language models to guide the search process that decodes text from speech. These language models can be bigrams (word-to-word transition pairs and probabilities) or finite state grammars. The models can be derived from either Backus-Naur form (BNF) grammars or example texts.

Additional Product Information. ICSS provides development and run-time support for speech recogni-

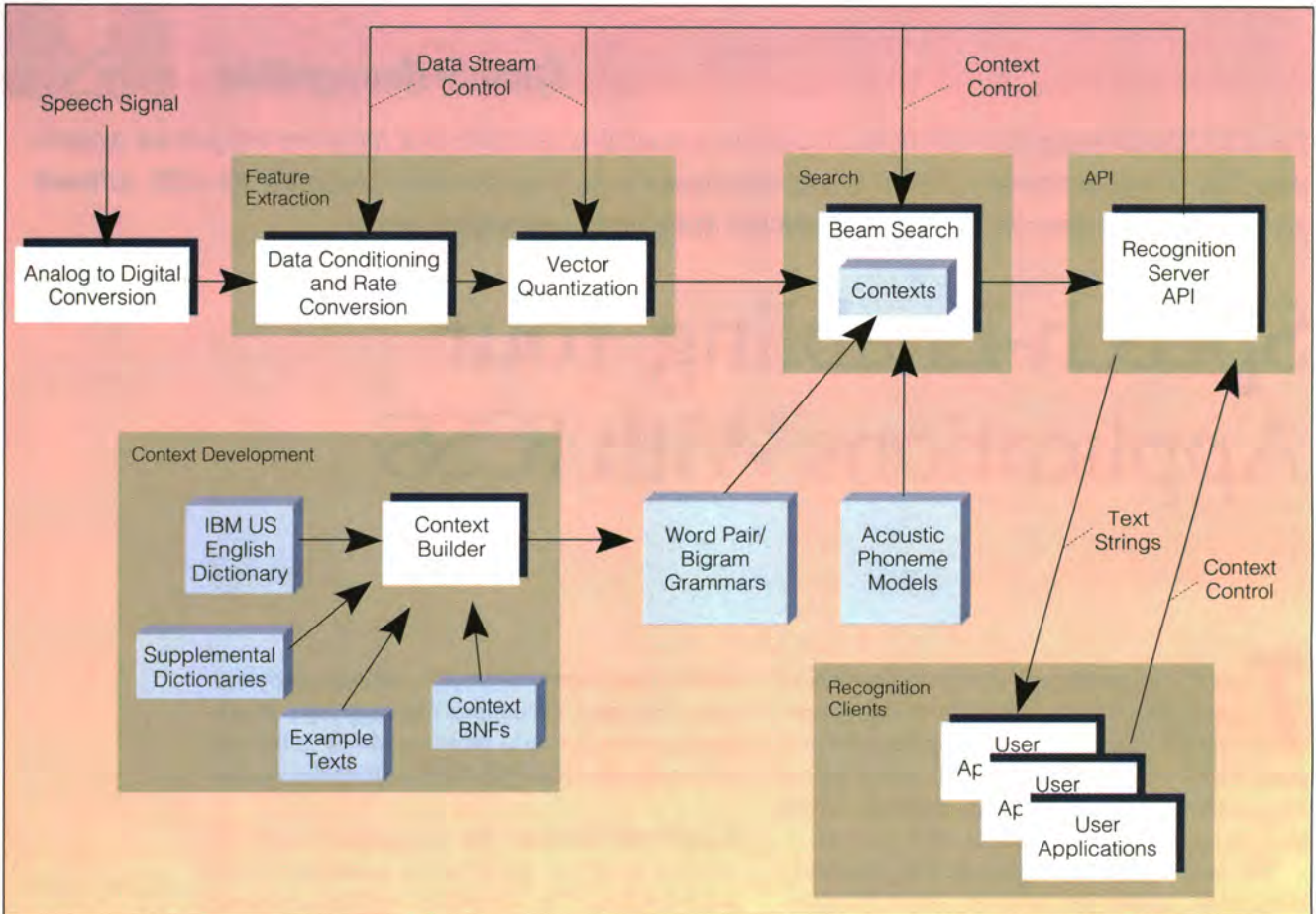


Figure 1. ICSS logical architecture

tion-capable applications in the OS/2 and AIX environments. For many tasks, recognition is performed at or near real time on contemporary 80486 processors with five additional megabytes of main memory above that required for other purposes. Because it takes advantage of OS/2 2.1's preemptive multitasking, ICSS begins to process speech data before the speaker finishes talking, increasing response time.

What Kinds of Applications Can Use ICSS?

ICSS can be applied to situations in which user productivity, space, or hand motion is at a premium. It is best suited for applications that have clearly defined sets of actions or data entry requirements that can be addressed by large vocabulary-continuous speech recognition. Users must be familiar with the command structure of their applica-

tions, but more freeform applications can be supported for limited domains.

Users adapt to speech recognition quickly, and recognition rates improve as speakers use the system. The continuous speech mode supported by ICSS lets developers support high-speed multifield data entry (using a connected utterance). Situations that are well suited for ICSS voice-enabled applications include:

- Processing transactions using GUI-based applications
- Executing hands-free data entry so an operator can simultaneously handle and catalogue an object
- Retrieving data from information systems such as SQL databases
- Supporting help desk operations
- Controlling simulators

- Controlling instruments
- Navigating through user interfaces
- Data entry.

HOW DOES ICSS WORK?

The logical architecture of ICSS is depicted in Figure 1, which shows the major system components that make up the development and run-time systems.

Analog to Digital Conversion. This component currently consists of an IBM audio capture-and-playback adapter (ACPA) card and its associated software driver. The adapter is available in Micro Channel (MCA) and IBM AT Bus architectures. ICSS supports data acquisition using the ACPA at rates of 11KHz and 22KHz in linear 16-bit PCM samples. Of the two rates, 22KHz provides the higher accuracy but is computationally more demanding. 11KHz is adequate for most, if not all, applications.


```

<main_win_cmd> ::= FILE | EDIT | VIEW |
                TEXT | STYLE | PAGE |
                FRAME | TOOLS | WINDOW |
                HELP | PAGE-UP | PAGE-DOWN .

```

Figure 2. "Window dressing" context supporting a command structure

```

<pin> ::= <digit> <digit> <digit> <digit> .
<digit> ::= ZERO | OH | ONE | TWO | ... | NINE .

```

Figure 3. Setting up the BNF form of a PIN context

Data Conditioning and Rate Conversion. This step converts speech data from external to internal formats for use during the feature extraction phase of the recognition process.

Vector Quantization. Once the speech data samples have been converted to the appropriate format, the feature extraction phase of the recognition process begins. The feature extraction

step significantly compresses the speech data.



Recognition Search. After data has been vector-quantized, the feature vectors, context word-pair grammars, and acoustic phoneme models are used by the recognition engine for actual recognition of the speech signal. This is done with Hidden Markov models for pattern matching of the acoustic data stream.

Speech Recognition API. The speech recognition API lets recognition clients communicate with the ICSS recognition engine. This allows several features:

- Context-loading
- Context-switching
- Control of data acquisition
- Return of text decoded from speech
- Control of various runtime

Now Shipping....

NDP OS/2 2.1 Developer's Pack

\$595 Includes:

- 32-bit Globally Optimized Code
- 32-bit Graphics and API Access
- IBM Toolkit and Workframe
- Integrated Development Environment
- 1500 pages of documentation
- Your Choice of Compiler:

NDP C/C++, NDP Fortran-77
or NDP Pascal

Call for our White Papers on OS/2, i860
Parallel Processing on PCs, Fortran 90 and
our port of LAPACK to the 486 and i860.

Microway®

Research Park, Kingston, MA 02364 USA (508) 746-7341
U.K., 081-541-5466 USA FAX (508) 746-4678

Waldenbooks

Waldenbooks® and Que Present the Official Object-Oriented Design Guide



Object-Oriented Interface
Design Guidelines

IBM® Corporation

Uncover the secrets of interface design! This fantastic tutorial and reference describes the fundamental concepts of the Common User Access™ (CUA™) interface.

- Learn about the user interfaces and object-oriented environments
- Explore a CUA designer model
- Examine the components of the CUA interface
- Apply instructions on how to design a product with a CUA interface

que

ISBN# 1-56529-170-0
IBM Pub# SC34-4399-0
\$29.95 USA
Waldenbooks Item# 0967

Available at Waldenbooks®,
Waldensoftware®, & Brentano's®

To Order Direct, Call 1-800-443-7359 Dept# 768



ICSSStart	Initializes the ICSS API code and data and prepares to start communications with the ICSS server.
ICSSStartConversation	Establishes communications with the requested ICSS server facility.
ICSSDefineContext	Requests the ICSS server to load a context into memory in preparation for guiding the ICSS listening process.
ICSSGetValue	Retrieves the setting of a specified ICSS control variable so the application program can verify the setting.
ICSSSetValue	Sets the value of a specified ICSS control variable so the application can customize ICSS to its requirements.
ICSSListen	Causes ICSS to begin listening to the specified source for speech. Once listening has started, ICSS returns control to the application program.
ICSSGetSpokenWords	Causes the ICSS server to retrieve the text representing the recognized speech, in order to return it to the application program. This text is not returned until the listening and recognition process is complete.
ICSSPlayback	Causes the ICSS server to play a specified digitized audio file on a specified output device. The application can use ICSSPlayback to play audio responses or prompts to the user.
ICSSRemoveContext	Removes a previously defined context from memory. ICSSRemoveContext can be executed to conserve memory.
ICSSEndConversation	Removes remaining active contexts and terminates communication with the ICSS server.
ICSSEnd	Cleans up ICSS API memory and resources.

Table 1. Functions that may be invoked in the ICSS API

parameters

- Playback of prerecorded audio files.

The context-switching capability of ICSS allows user applications, operating as speech recognition clients, to load numerous contexts and switch rapidly among them using the speech recognition API calls. The process involves activating a grammar for use during the recognition search process. The API allows multiple applications on the same machine to treat the speech recognition engine as a successively reusable resource; context switching is therefore a particularly important technique when using multiple applications in that each application has several screens or commands and data entry processes.

Context Development. The context devel-

```

/*****
/* Program name: reco.c */
/* OS/2 Developer Magazine, Issue: July/August 1993 */
/* Article: Speech Enabling Your Applications With ICSS Is Easy */
/* Author: Brandon Booth Phone: 301-240-3671 FAX: 301-240-3377 */
/* Description: Program for performing speech recognition. Loads */
/*              a context and executes it, returning the spoken */
/*              words, which best match within the context. */
/* Program Requirements: Software : OS/2 Version 2.0 or better */
/*                      ICSS */
/*                      IBM ACPA V1.02 Software support*/
/*                      Hardware : 486 DX Processor */
/*                      Microchannel architecture */
/*                      5 Meg RAM for ICSS */
/*                      25 Meg DASD for ICSS */
/*                      IBM M-ACPA card */
/*                      Microphone */
*****/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

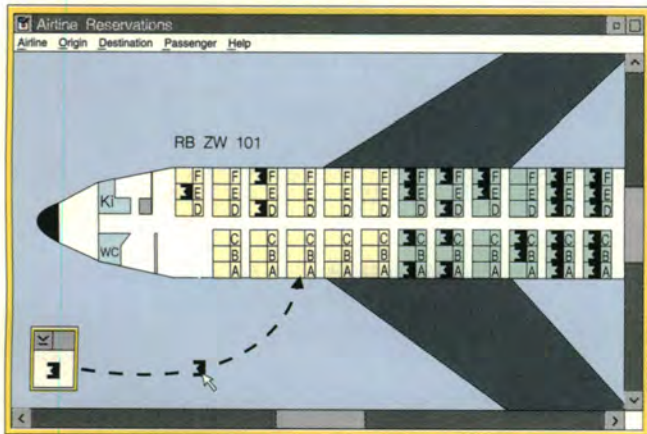
#include "icssapi.h"

```

Figure 4. Sample program for speech recognition (continued on page 88)

This program contains scenes of a graphic nature.

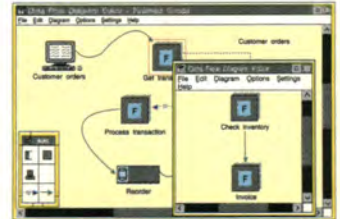
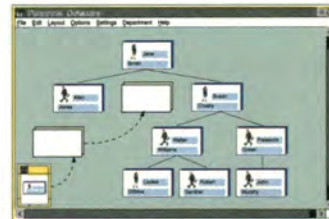
Introducing
Graphics
Interface
Kit/2



But don't worry. It's all in good clean fun. Because now there's a fast, cost-effective way to create graphic, easy-to-use interfaces for OS/2® 2.0 applications and tools. Introducing Graphics Interface Kit/2 (GIK/2), an AD/Cycle® product from IBM Programming Systems.

GIK/2 makes it easy to produce and manipulate symbolic representations of data without writing a single line of Presentation Manager® code. It automatically generates C code, where you can link the graphic symbols to the application data they represent. This not only speeds up your development cycle, it also helps you create applications that are intuitive and easy-to-use. Simplify transaction systems. Make databases easy to access. Bring organizational charts to life. Track inventory with simple icons. Any application can be made easier with the right graphic front-end. And GIK/2 can help you make it happen.

For orders or additional information, please call 1 800 IBM-CALL and ask for Department S71. If you need graphic proof, ask for our evaluation demo which, by the way, is rated G.





opment environment provides tools for activities required to build a speech recognition context, including:

- A context BNF compiler that creates word pair, or bigram, language models and finite-state language models
- A bigram-by-example language model builder
- A supplemental dictionary processor
- An add-word utility
- A context test and checkout function.

The ICSS API. Applications interact with ICSS through the ICSS API, which consists of eleven 32-bit C-language function calls. A concept important to understanding the API is that application programs interact with ICSS as both client and server. In this case, the application is the client and ICSS is the server that provides speech-recognition functions. As each API function is called, the request is passed to the ICSS speech server for processing and the results are passed back to the ICSS API and returned to the invoking application.

In general, it is necessary to perform the following functions to use ICSS:

- Establish communications with the ICSS server facility
- Establish a conversation with that server
- Define any necessary contexts to the conversation to guide speech recognition
- Start the ICSS listening process
- Wait for any spoken words to be returned as text.

To start associating these basic functions, Table 1 describes the functions that may be invoked in the ICSS API.

To ensure that the ICSS API interacts correctly with the ICSS server, the API tracks the order in which these functions are invoked and the success or failure of each function. This hierarchy of function invocation is tracked by maintaining a state for the ICSS API for each client application. For instance,

```
#define SIZEBUFFER 256

void quit (char *msg, long rc);
void message (char *msg, long rc);

/*****
 * main
 *****/
int main(int argc, char *argv )
{
    char    inputbuf`SIZEBUFFER ;
    char    buffer`SIZEBUFFER ;
    long    rc    = -1;
    long    context = 0;
    long    notdone = 1;
    long    spoke_too_soon;
    float   utt_len;
    float   acq_time;
    float   reco_time;

    if(argc != 2)
    {
        printf("The reco example program requires one argument.\n");
        printf("The name of a context to do recognition against.\n");
        printf("Example: reco /icss/appl/samples/rl/context/rl.m.ctx\n");
        return 0;
    }
    puts("Welcome to RECO, the ICSS api example program.");

    rc = ICSSStart(NULL);
    if (rc != ICSS_SUCCESS) quit("Error: Start Failed",rc);

    rc = ICSSStartConversation(ICSS_IBMACPA, /* ADCType */
                              ICSS_DEFAULT_INT, /* ADCIFNumber */
                              ICSS_DEFAULT_STRING, /* FESystemName */
                              ICSS_DIRECT, /* FELinkType */
                              ICSS_DEFAULT_STRING, /* RPSYSTEMName */
                              ICSS_DIRECT); /* RPLinkType */
    if (rc != ICSS_SUCCESS) quit("Error: StartConversation Failed",rc);

    rc = ICSSDefineContext(&context, /* ContextHandle */
                          argv`1, /* ContextName */
                          ICSS_DEFAULT_FLOAT, /* InsertionPenalty*/
                          ICSS_DEFAULT_FLOAT, /* LanguageWeight */
                          ICSS_DEFAULT_FLOAT); /* GrammarWeight */
    if (rc != ICSS_SUCCESS)
    { message("Error: Define Context failed",rc); notdone = 0; }

    while(notdone)
    {
        printf("\nListen?`yes,no >");
        fflush(stdout);
        gets(inputbuf);

        if( (strstr(inputbuf,"NO" ) != NULL) ||
            (strstr(inputbuf,"no" ) != NULL) ) { break; }
        else
        {
            rc = ICSSListen(context, /* ContextHandle */
                            ICSS_DEFAULT_INT, /* SpeechInputType */
                            ICSS_LIVE_MIKE, /* SpeechInputSource*/

```

Figure 4. Sample program for speech recognition (continued from page 87)

ICSS must be in a "listen" state to successfully perform ICSSListen and in a "conversation" state to perform ICSSGetSpokenWords.

A BNF FOR A CONTEXT

ICSS uses contexts, derived from compiled grammars or bigrams, to guide the speech-recognition process. The Speech Development Environment (SDE) tool set in OS/2 2.1 lets developers create and test speech recognition contexts.

The most important step in creating a context is application design. ICSS's context-switching capability allows speech-recognition contexts to be designed to mirror application structure. The windowing message structure enforced by most popular GUIs can provide guidance on organizing speech-recognition contexts. The ICSS recognition server can switch between the preloaded contexts defined for each major window. When a particular window is focused on, it can use the speech-recognition API to switch to the appropriate speech context. For example, a word processor's main window might have a "window dressing" context supporting a command structure, as shown in Figure 2. This context allows a user to speak any of the menu options (File, Edit, and so on).

Developing several small contexts makes it possible to reduce the search space, or perplexity, during functions performed by the application. It also reduces error rates when the speech recognizer is operating under sub-optimal conditions such as high background noise.

A banking application, for example, might ask which service is requested (loan rates, checking services, new accounts, current account status, and so on). A context would then be defined for each of the expected replies to the application's verbal or visual prompts. Once the user replies, the context is switched and a prompt (such as a request for a personal identification number, or PIN) is given. The BNF form of a PIN context could be set up as shown in Figure 3.

This grammar in Figure 3 defines what can be said during a "listen" request. The engine will recognize the closest phrase in the grammar that matches an utterance. The rather simple context in Figure 3 recognizes only

digit strings; if the context is active, a phrase like "One, two, trees, six" would probably be recognized as ONE TWO THREE SIX. The format works well, however, for inputs that do not contain "trick" sequences.



os/2 Developer

THE MAGAZINE FOR ADVANCED SOFTWARE DEVELOPMENT

OS/2 Developer is owned and published by Miller Freeman Inc., 600 Harrison St., San Francisco, CA 94107, (415) 905-2200. Beginning with the July 1993 issue, the editorial content of *OS/2 Developer* is controlled by Miller Freeman Inc.

OS/2 Developer, originally published as the *IBM Personal Systems Developer* and later as the *IBM OS/2 Developer*, was founded in 1988 by IBM Corp.

The term "OS/2" is used in this publication as an abbreviation for the full term "OS/2 operating system." OS/2 is a registered trademark of IBM Corp.

To correspond with *OS/2 Developer*, please write to:

Editor

Miller Freeman Inc.

600 Harrison St.

San Francisco, CA 94107

tel: (415) 905-2387

fax (415) 905 2234

Internet: os2mag@vnet.ibm.com

CompuServe: 76711,1005.



ICSS DEVELOPER'S PROGRAM

The ICSS Developer's Program offers three classes on speech recognition. Call (800) 627-8363 for an information package. There are three packages available: \$3995.00 for a bundle of 23 licenses and \$299.00- and \$699.00 for the small and large microphone packages, respectively.

The program consists of a developer's license and tools that enable applications for speech, as well as the runtime licenses needed to execute these programs. The package also includes a developer's guide and a microphone.

Technical support and development assistance is provided via the 800 number. While there is currently no technical support provided on CompuServe, ICSS specialists monitor the forums regularly and will respond to questions. Call (800) 627-8363 for information on ICSS demonstrations at trade shows and IBM briefing centers.

REFERENCES

IBM Continuous Speech Series Developer's Guide, Developer's Program Beta 1, December 1992.

A SIMPLE PROGRAMMING EXAMPLE

Figure 4 shows a sample program for speech recognition. The program loads a context and executes it, returning the spoken words that best match within the context. A copy of this code can be downloaded from library 13 in the OS2DF2 forum on CompuServe under the name "RECO.C." The code will execute successfully only if ICSS is on the system running it.

```
        NULL); /* SpeechInputName */
if(rc != ICSS_SUCCESS) { message("Error: Listen Failed",rc);
                        break; }

printf("\nBEGIN SPEAKING\n");

rc = ICSSGetSpokenWords(SIZEBUFFER-1, /* TextBufferLen */
                        buffer, /* TextBuffer */
                        &spoke_too_soon, /* SpokeTooSoon */
                        &utt_len, /* UtteranceLength */
                        &acq_time, /* AcquisitionTime */
                        &reco_time); /* RecognitionTime */

if (rc == ICSS_SUCCESS)
{
    printf("The spoken text was: '%s \n",buffer);
    printf("Utterance Length %4.2f\n",utt_len);
    printf("Acquisition Time %4.2f\n",acq_time);
    printf("Recognition Time %4.2f\n",reco_time);
}
else
{
    message("Error: GetSpokenWords Failed",rc);
    break;
}
}

rc = ICSSEndConversation();
if (rc != ICSS_SUCCESS) message("Error: Conversation End Failed",rc);

ICSSEnd(ICSS_LOCAL+ICSS_NOFORCE);

puts("RECO Exiting.");
return 0;
}

void quit(char *msg, long rc)
{
    printf("%s rc=%d %s\n",msg,rc,ICSSErrorString`rc );
    exit(-1);
    return;
}

void message(char *msg, long rc)
{
    printf("%s rc=%d %s\n",msg,rc,ICSSErrorString`rc );
    return;
}
```

Figure 4. Sample program for speech recognition (continued from page 88)

Vincent Stanford, IBM Corp., 100 Lakeforest Blvd., Gaithersburg, MD. 20877. Stanford is a senior programmer and lead engineer for the IBM Continuous Speech Series. He has twenty years of experience developing DSP-based systems for speech recognition, sonar, biomedical, seismic, speech, and acoustic signal classification systems.

Brandon J. Booth, IBM Corp., 100 Lakeforest Blvd. Gaithersburg, MD 20877. Booth, a senior associate programmer, has worked on various parts of ICSS (including the API and I/O subsystem) since 1991. Before that, he worked on the CUA Controls Library/2 and the Personal Communications/3270 product. Booth has been with IBM since 1988, when he graduated from American University with an M.S. in computer science.



The IBM OS/2 32-Bit Porting and Technical Consulting Workshops give developers hands-on training and assistance in application porting. **BY MARILYN JOHNSON**

OS/2 32-Bit Porting and Technical Consulting Workshops

IBM is offering developers a way to port existing DOS, Windows, UNIX, and 16-bit OS/2 applications to 32-bit OS/2 with the IBM OS/2 32-Bit Porting and Technical Consulting Workshops, which provide hands-on training and assistance in application porting. The workshops focus on OS/2 32-bit features such as thread management, multitasking, robust memory management, and full 32-bit APIs, with a focus on positioning applications for portability.

The workshops are led by industry experts who maintain expertise in current and future technology and have extensive experience in porting to 32-bit platforms. Workshop leaders' experience with application porting allows:

- Accurate sizing of the migration effort
- A minimized learning curve
- Architectural integrity
- A shortened development cycle
- 35% to 100% of code ported.

All workshops include extensive hands-on porting of code brought in by the participants. The price of each workshop includes meals, the use of lab hardware, and software to take home. Participants also receive class materials, the OS/2 2.1 operating system, and C Developers' Workset/2 and Multimedia Presentation Manager/2 (MMPM/2).

WINDOWS 3.X TO OS/2 32-BIT PRESENTATION MANAGER NATIVE WORKSHOP

This five-day workshop is for Windows 3.x application developers who want to expedite porting

Windows code to the 32-bit OS/2 platform, allowing Windows 3.x applications to exploit OS/2 memory management, multithreaded design, and the Presentation Manager (PM) interface. The workshop involves extensive hands-on porting of developers' source code, including instruction and assistance from a porting expert.

The workshop highlights architectural changes necessary to port, including comparisons between Windows 3.x and OS/2 program design and discussion of the Windows 3.x 16-bit segmented

The workshops focus on features such as thread management, multitasking, robust memory management, and full 32-bit APIs, with a focus on positioning applications for portability.

memory management versus the 32-bit flat address space of OS/2 2.1. Students explore the benefits of OS/2 memory pools, guard pages, shared memory, page allocation techniques, and multi-threaded design.

The workshop also covers programming to PM under OS/2 2.1. Participants will discuss forms of interprocess communication that allow OS/2 processes to share and pass data and also compare

the OS/2 DOS APIs to functions provided with the C run-time environment. The workshop covers common controls (such as the container, notebook slider, and value set) and common dialogues (such as the file and font dialogues),

now a part of the OS/2 operating system. Finally, the class looks at the MMPM/2 Toolkit and its features.

As part of the Windows 3.x to OS/2 32-bit workshop, attendees use new porting tools that help convert applica-

tions to the OS/2 2.1 PM platform.

Prerequisites for this workshop are Windows 3.x and C programming experience as well as an existing Windows 3.x application.

DOS TO OS/2 32-BIT PRESENTATION MANAGER WORKSHOP

This seven-day workshop is for DOS application developers who want to port existing applications to the OS/2 32-bit PM platform, allowing DOS applications to exploit OS/2 memory management, multithreaded design, and the CUA PM interface. The workshop involves extensive hands-on porting of the developer's source code, including instruction and assistance from a porting expert.

During the week, experts discuss how to map BIOS-based functions to PM APIs and how to redesign the user interface to take advantage of PM programming. It also covers those 32-bit OS/2 functions equivalent to DOS INT functions.

OS/2 32-bit flat addressing memory management, making the best use of OS/2's virtual address space, OS/2 memory pools, guard pages, shared memory, and page allocation techniques are all covered.

The concepts of multitasking with OS/2 processes and threads are discussed in depth. Topics include inter-process communication and offloading work to multiple processes and threads of execution.

Common controls that allow DOS programmers to spend more time on code specific to their application are reviewed, then the instructor demonstrates how to incorporate MMPM/2 into newly ported OS/2 applications.

Prerequisites for this workshop are DOS and C programming experience as well as an existing DOS-based C application.

SYSTEM OBJECT MODEL/WORKPLACE SHELL WORKSHOP

This five-day workshop provides detailed information about OS/2 32-bit System Object Model (SOM) program-

SINGLE KEY-STROKE SCREEN PRINTING



CLIPBOARD VIEWER

SCREEN CAPTURE

- Create **Quality Black & White** copies of color images.
- Screen Saver prevents monitor burn-in.
- Date & Time Display for Time-Stamping images.
- LAN compatible.
- Entity & OEM licensing available.
- Economical combination of important utilities.
- The most stable and reliable product available.
- Includes both OS/2 2 (32-bit) and OS/2 1.3 (16-bit) versions.

THE LEADER: PrntScrn™ Screen Image Utility for OS/2



MITNOR Software
Phone: (918) 357-1628
Fax: (918) 357-2869

TM



ming. It shows how to interact with the Workplace Shell and takes advantage of future SOM enhancements with minimal impact on source code. The workshop features extensive hands-on application development.

During this week, the instructors discuss similarities and differences of SOM and C++ as well as differences between Workplace Shell and non-Workplace Shell applications, identifying which modules should or should not become Workplace Shell objects.

Attendees learn how to integrate existing OS/2 applications into the Workplace Shell by sub-classing existing SOM and Workplace Shell objects, enabling applications to take advantage of the methods and default actions provided by the object-oriented Workplace desktop.

Prerequisites for this workshop are 32-bit OS/2 and C programming experience, as well as knowledge of the principles of object-oriented programming.

OS/2 16-BIT PRESENTATION MANAGER TO 32-BIT PRESEN- TATION MANAGER WORKSHOP

This five-day workshop enables OS/2 developers to port their existing C applications to the OS/2 32-bit PM platform. This includes taking advantage of the flat memory model, better interprocess communication, and 32-bit execution vs. "thunked" 16-bit code. The workshop involves hands-on porting of the developer's source code, complemented by instruction and assistance from a porting expert.

Differences between 16-bit and 32-bit OS/2 subsystems, focusing on areas such as the new memory model, are reviewed. OS/2 2.1 flat memory addresses vs. 16-bit segmented memory architecture are discussed, as are new memory features such as sparse memory objects and page guards. The class explores how 16-bit modules communicate with 32-bit modules and the ramifications of "thunking" to 16-bit code.

The instructors then address thread-

ing features available in OS/2 and OS/2 2.1, including suspending and resuming threads of execution and the system's ability to manage stack memory for applications using threading. It then covers the changes and enhancements to interprocess communication between processes and threads.

The workshop focuses on the new and changed Base Services APIs and their C run-time counterparts; it also introduces OS/2's common controls and standard dialogues, which reduce the code required to perform common interfaces (for example, providing a dialogue of drives, directories, and files). Finally, it covers performance enhancement aids and tools, an introduction to MMPM/2, and adding multimedia to OS/2 applications.

Prerequisites for this workshop are experience in OS/2 1.x and C programming and an existing OS/2 1.3 applica-

UNIX TO OS/2 32-BIT WORKSHOP

This five-day workshop is for UNIX/X-Windows application programmers who want to port or reimplement applications on the OS/2 32-bit platform and also for programmers who want to include OS/2 in their UNIX environment. The workshop fosters a broad-based understanding of OS/2 components by comparing UNIX and OS/2, which helps developers understand how to provide existing application function in the OS/2 environment and how to exploit unique features of OS/2 to extend or enhance existing function. Capabilities and interfaces common to both UNIX and OS/2 are

emphasized to expedite porting and preserve application architecture and design wherever possible.

During this workshop, instructors give an overview of OS/2 and identify the programming environment and development process. The class compares OS/2 and UNIX-based operating system concepts, interprocess communication using shared memory, pipes, queues, semaphores, transmission con-

*All workshops include
extensive hands-on
porting of code brought
in by the participants*

trol protocol/internet protocol (TCP/IP) sockets and RPC, NetBIOS, and SNA basics. The class discusses the interoperability of X-Windows and PM and OS/2 security in standalone and LAN environments.

Prerequisites for this workshop are UNIX/X-Windows and C programming experience as well as familiarity with standard development tools.

Marilyn Johnson, IBM Corp., 1000 N.W. 51st St., Boca Raton, Fla. 33431. Johnson has been with IBM since 1980 and with the worldwide Developer Assistance Program since 1989. She is the program manager for the porting and technical consulting workshops.

WORKSHOP LOCATIONS AND INFORMATION

Workshops are held in West Palm Beach, Fla. and Dallas, Texas. The UNIX to OS/2 32-bit workshop is held at the IBM site in Austin, Texas.

For more information, call One Up Corp. at (800) 678-3187 Monday through Friday, 8:00 a.m. to 5:00 p.m. Central time. Refer to the IBM OS/2 Porting and Technical Consulting Workshop.



Since 32-bit OS/2 was released last year, more than 1,200 OS/2 applications have appeared on the market. Each issue, this column will note new product releases in the OS/2 world.

New Products, New Support



TOOLS

Borland C++ for OS2. Borland's new C++ for OS/2 makes quick creation of powerful C and C++ applications easy with a graphical environment and a suite of visual tools, including Turbo Debugger GX. C++ includes an optimizer for 32-bit code generation and is designed with ANSI C Certification and AT&T C++ standards.

Borland International Inc.
Phone: (800) 331-0877 Fax: (408) 439-9103

Digitalk PARTS Workbench for OS/2. The PARTS Workbench is a 32-bit client/server technology integration tool that lets programmers assemble and reuse parts. The product comes with 67 visual parts (Window, Button, and Text Pane) and nonvisual parts (dynamic data exchange, launch pad, and Btrieve). These components, plus others from Digitalk and third-party vendors, can be linked together to form applications.

Digitalk
Phone: (800) 531-2344 Fax: (310) 645-1306

HockWare VisPro/REXX. VisPro/REXX helps programmers create powerful and professional OS/2 2.x programs with REXX in a GUI environment. VisPro/REXX exploits all OS/2 controls while enabling all the REXX features, including interactive debugging, APPC, HLLAPI, and OS/2 Database Manager interfaces.

HockWare Inc.
Phone: (919) 387-7391 Fax: (919) 380-0757

IBM C Developer's Workset/2. The C Developer's Workset/2 consists of several programs designed

to maximize developer productivity. Included in this package is the IBM high-performance C Set/2 compiler and debugger, the IBM WorkFrame/2 project-oriented development environment, and the IBM Developer's Toolkit for OS/2, which features a collection of language-independent build tools, productivity tools, sample programs, reference information, and a kernel debugger. IBM also offers a free quarterly compiler newsletter that can be ordered by fax.

IBM Corp.
Phone: (800) 342-6672
Fax: (416) 448-6057 (for newsletter only)

PSS Tape Solution Series. Parallel Storage Solutions offers their Tape Solution Series, the only tape-backup on the market specifically designed for use with OS/2. For independent or network users, it protects data quickly and efficiently. Coupled with a tape backup device, the software provides the fastest backup speed on the market.

Parallel Storage Solutions
Phone: (800) 998-7839 Fax: (914) 347-4646

WATCOM C/C++32. C/C++32 is a 32-bit multiplatform optimizing compiler package with a comprehensive toolset, including a debugging system, profile, linker, make utility, and other tools. C/C++32 features include full C++ implementation with templates, exceptions and nested types, and an advanced code generator with 486 and Pentium optimization.

WATCOM
Phone: (519) 886-3700 Fax: (519) 747-4971



APPLICATIONS

Cirrus Unite 2.0. Unite is an integrated client/server-based document imaging solution designed to reduce paper handling while increasing productivity. A true multitasking application for LAN use, Unite 2.0 supports optical jukebox technology for storage, fax, and print output.

Cirrus Technology Inc.
Phone: (301) 698-1900 Fax: (301) 698-1909

Lotus Applications. Lotus has recently shipped several of its most popular applications rewritten for OS/2 2.x to take advantage of multithreaded full 32-bit support. The OS/2 2.x versions of Ami Pro, 1-2-3, Freelance Graphics, Notes, and cc:Mail all

offer drag and drop capabilities and Workplace Shell integration.

Lotus Development Corp.
Phone: (800) 872-3387

WordPerfect 5.2 for OS/2. WordPerfect 5.2 is a graphical word processing program that includes popular features such as the Button Bar, ruler, QuickFinder, and QuickMenu, as well as drag-and-drop capabilities and new macros. It supports all documents created in previous versions of WordPerfect.

WordPerfect Corp.
Phone: (800) 451-5151 Fax: (801) 222-5077



DEVELOPER SUPPORT

IBM Developer Connection for OS/2. The IBM Developer Connection for OS/2 is a new service designed to provide OS/2 developers with the technical information needed to create OS/2 applications. The Developer Connection will also feature updated information, tips, techniques, sample applications, and beta code for upcoming IBM applications.

The service, which consists of quarterly CDs and newsletters as well as electronically-accessible developer support, costs \$199.95 per year. However, developers who sign up for the Developer Connection before September 30, 1993 are eligible for an introductory price of \$149.95 per year.

For more information about receiving the Developer Connection, call 1-800-633-8266.

New I.V. League Catalog. The IBM OS/2 Independent Vendor (I.V.) League showcases hundreds of books, magazines, courseware, and consulting services that support OS/2. Now, select member products are featured in the all new *I.V. League Product Catalog*. To order a free copy, call 1-800-342-6672.

OS/2 CALENDAR

August 24-26	Business Software Solutions Conference (formerly Windows & OS/2 Conference), Boston (415) 905-2784
August 29-September 2	IBM Technical Interchange, Orlando (800) 872-7109
September 9	C.A.M.P., Chicago (708) 291-1360
October 5-8	NetWorld, Dallas (800) 829-3976
October 19-21	PC Expo, Chicago (800) 829-3976
October 31-November 5	ColoradOS/2, Colorado Springs (719) 481-3389

If you would like to contribute to this column, please contact Product Watch, OS/2 Developer, 600 Harrison St., San Francisco, CA 94107, fax: (415) 905 2234.

YOU WOULDN'T THINK ABOUT REINVENTING THE WHEEL... WHY REINVENT REASONING?



Take a quantum leap forward with The Integrated Reasoning Shell (TIRS). TIRS offers:



Speed

Fast performance with 32-bit processing



Productivity

Reduce development and maintenance time



Embeddability

Fully integrates with your application



Flexibility

Do it your way- Use your favorite DBMS

TIRS features include multiple platforms (OS/2, VM, and MVS), frame-based reasoning with inheritance, a code generator (C and PL/I), and a GUI development environment.

For more information about TIRS, our other OEM offerings, and our two month product trial period, call:

1-800-IBM-SOLV

Outside the US: (408) 463-4381



IBM, TIRS, and OS/2 are trademarks of the International Business Machines Corporation.

Advertiser Index

COMPANY AND FAX NO.

R.S. NO.

Borland International Inc.: (408) 439-0115	COV4
Burton Systems Software: (908) 233-0716	79
Cognitive Software Inc.	53
Computer Associates	4
Creative Systems Programming: (609) 234-1920	60
dSoft Development: (713) 378-7448	79
Essex Systems Inc.: (508) 750-4699	53
Easel Corp.: (617) 221-3099	19
Gpf Systems Inc.: (203)873-2171	10
Hookware: (919) 380-0757	15
IBM Canada: (800) 445-2426	35
IBM Programming Systems Cary Labs: (919) 469-7423	43
IBM CUA Controls Library: (919) 469-4423	27
IBM (DB2)—Programming Systems	59
IBM GIK: 011-43-1-21145-4490	87
IBM IV League	32a
IBM Personal Software Products: (407) 982-6408	54-55
IBM Speech Recognition: (301) 240-3377	13
IBM TIRS	96
IBM U.S. Marketing and Services: (904) 238-3442	48a
IBM Programming Systems (WITT)	17
Information Builders Inc.: (212) 629-8819	46
Intersolv	37
Impact Software: (818) 879-5593	53
Intelligent Environments: (508) 750-4699	24
Kaseworks: (404) 448-4163	21
Microformatic: (203) 648-9587	61
Micropolis: (612) 341-2114	22-23
Microway: (508) 746-4678	85
Mitnor Software: (918) 357-2869	92
MSR Development: (409) 560-5964	77
Micro Focus: (415) 856-6134	2
Prentice Hall: (800) 448-3804	85
Real Time Technologies: (708) 446-6886	53
Rightware Inc.	27
Sequiter: (403) 436-2999	COV3
Softbridge: (617) 864-7747	69
Softouch Systems Inc.	99
Software Development Conference 1993	80-81
Soft and GUI: (718) 934-2133	29
Softronics Inc: (719) 548-1878	11
Stac Electronics: (619) 431-1001	30
Token Technology: (408) 559-8661	65
Van Nostrand Reinhold	51, 101
Watcom C for IBM PCs: (519) 747-4971	COV2,1
Zinc Software: (801) 785-8996	41



This article provides an in-depth walk-through of debugging routines using the REXX programming language's trace command. Programs can be tested entirely inside REXX without the use of external debugging tools.

BY STEVE CLARK

Debugging REXX Applications

In *The REXX Language: A Practical Approach to Programming*, author Michael Cowlshaw states that the single design objective of REXX was "to make programming easier than it was before." The simplicity of REXX allows programmers to deal more directly with the logic of a program.

The same design objective was applied to debugging REXX code. The REXX language includes many debugging features to make testing and debugging easier. Understanding where and how these debugging aids are used can improve the quality of your code and the efficiency of the development, testing, debugging, and documentation process.

Most of the information and examples here are appropriate for all REXX operating environments. Specific OS/2 features are clearly noted.

DEFINING TRACE ACTION

The REXX language processor offers many trace actions for use in debugging. Setting a trace action determines the amount of trace information to be displayed and whether the trace is run interactively. This can be controlled within a REXX program by the TRACE instruction or the built-in TRACE() function. These share most of the same parameters and can be used interchangeably in many situations.

The trace action is defined by a single parameter made up of either a whole number or an alphabetic character(word) option. The number trace parameter is only valid during interactive tracing. The most commonly used alphabetic character(word) options are:

- **Normal.** Failed commands are traced after

execution. This is the default setting.

- **Off.** All tracing is turned off.
- **Results.** All clauses are traced before execution and results are displayed. This is the most common option.
- **Intermediates.** This option is similar to Results, but intermediate results are also displayed.
- **Labels.** All labels are traced as they are passed.

REXX recognizes an option by the first letter of its name so it is not necessary to type the full name. The alphabetic character(word) option may also include one or more question mark (?) prefix characters. The question mark prefix acts as a switch turning interactive tracing on and off. Examples of setting the REXX trace action using the trace instruction are shown in Figure 1.

REXX trace parameters are described in detail in the *OS/2 2.0 Procedures Language 2/REXX* reference manual.

DIFFERENCES BETWEEN TRACE AND TRACE()

If the TRACE instruction and the TRACE() function are so similar, why is it necessary to have both? In practice, the TRACE instruction is sufficient for most debugging tasks, and its simpler syntax is quicker to use. There are differences, however, between the two commands. The TRACE() function has some unique features that make it a necessary and valuable debugging aid.

The most obvious difference between TRACE and TRACE() is that the TRACE() function returns a result. The result is the current TRACE action. You can invoke the TRACE() function directly within an



Steve Clark



expression, and the result will be evaluated as part of the expression. Alternatively, the TRACE() function can be invoked with the CALL instruction and the returned result will be assigned to the special variable RESULT. Examples of calling the TRACE() function are shown in Figure 2.

Also, like other functions, TRACE() evaluates its option as an expression, whereas the TRACE instruction evaluates its option as a constant. The TRACE instruction is defined this way to simplify the syntax of the command and avoid surprises to the programmer. The instructions TRACE I or TRACE OFF will always set the expected trace action. However, using CALL TRACE I or CALL TRACE OFF may generate some unexpected trace behavior if the symbols i or off have been assigned values. Figure 3 illustrates that it is necessary to be explicit when using the TRACE() function and delimit the option with quotes.

The intent in Figure 3 is to set the trace action to *Intermediates*. Instead of doing so, the language processor evaluates the variable i, which has been incremented to 11 by the do i=1 to 10 loop. What is actually executed is CALL TRACE 11.

Interactive trace allows programmers to monitor program execution, change variables, reexecute clauses, execute program routines, and even alter program flow.

This example points out another difference. The number parameter, while valid for the TRACE instruction, is not valid for the TRACE() function. Using the number parameter with a call to the TRACE() function will generate a syntax error.

The less obvious differences between the TRACE instruction and the TRACE() function become apparent during interactive tracing; during interactive trace, the REXX processor pauses

```
TRACE OFF      /* Trace off          */
TRACE 0        /* Trace off          */
TRACE ?R       /* Trace results - interactive */
TRACE I        /* Trace intermediate results */
```

Figure 1. Sample TRACE instructions

```
trace_value = TRACE()      /* Get trace option      */
trace_value = TRACE('R')  /* Get option and set results */
CALL TRACE 'R'             /* Set results - interactive */
```

Figure 2. Sample TRACE() function calls

after most clauses for input from the keyboard. One exception is the TRACE instruction. Following a TRACE instruction, the language processor will continue to execute until the next pause. This is true whether the TRACE instruction is in the file or entered from the

summarized in Figure 4.

TRACE OUTPUT

As each clause is executed, the trace output is displayed to the user. The display contains the executed source line number and text. A special symbol is also displayed to indicate the type of data being displayed. The ** symbol indicates that the displayed data is the source text of the executed clause. The +++ symbol indicates a trace message from the language processor.

When results are displayed, they are indicated by symbols that begin and end with the greater than sign (>). These symbols are shown in Figure 5.

INTERACTIVE TRACE

Interactive trace allows programmers to monitor program execution, change variables, reexecute clauses, execute program routines, and even alter program flow. The REXX processor pauses following the execution of most clauses and allows programmers to enter instructions from the console. In fact, all REXX instructions may be entered from the console during interactive trace.

In addition, there are two special

In addition, there are two special


```
do i=1 to 10
.
.
end
call trace i
```

Figure 3. Invalid TRACE() function call

instructions that control execution during interactive trace. These two instructions are the null line and the equal sign (=). Entering a null line by pressing the enter key will cause the processor to execute the next clause and continue executing until it reaches the next pause.

Entering an = will reexecute the last clause traced in the current routine. This is an extremely useful command when used in conjunction with the available REXX instructions. If you are

using the Results trace action and come across a complex line of code, you can switch the tracing action to Intermediates by executing the TRACE() function from the keyboard and then reexecuting the clause. The TRACE() function is used so the language processor will pause and not continue to the next clause.

Figure 6 demonstrates this scenario. In the first execution of the clause, you can see that the variable x was compared to the result of the VERIFY() function and the result of that comparison is 0 or false. But since the values of the variables in this clause are unknown, it is difficult to see how the result was determined. So the trace action is changed with the TRACE() function and reexecuted to get more detail. The Results symbols now show that the variable (>V>) 'x' equals 1, string contains a date, delLim_list contains a list of delimit-

ers, and that there is a literal (>L>) string M. The VERIFY function(>F>) has returned 3, the comparison operation (>D>) returned 0, and the final result is 0.

During an interactive trace, variables can be examined with the SAY instruction. The semicolon can be used to enter multiple clauses on a line. In fact, the processor requires that all instructions such as DO...END and IF...THEN be complete when entered, as in Figure 7.

While monitoring the program execution and examining the variables with the SAY instruction or through trace results, changing a variable's value is invariably useful. Variables can be modified with an ASSIGNMENT instruction, as in Figure 8.

Program subroutines can be executed directly from the keyboard with the CALL instruction. Subroutines can also be



Reprints available from **OS/2 DEVELOPER**

Full or partial reprints are available for all articles appearing in OS/2 DEVELOPER.

Reprints can be customized to your specific needs. You choose color, size, layout, graphics, quantity, etc.

Use your customized reprints for :

- *customer support*
- *sales tools*
- *marketing support*
- *education*

Call or fax today for a custom quote:

Laura Pullen
OS/2 Developer
Phone; 415-358-0126 ext 302
FAX: 415-358-9749

Maximize Performance with true 32-Bit Utilities for OS/2 Version 2

- Recover damaged HPFS Volumes
- Undelete Erased Files
- Optimize HPFS Volumes
- Edit Disk Sectors
- Test Disk Media
- Mass Erase Sensitive Files
- Display Volume Information
- Display System Information
- Display File Fragmentation Information
- Display Directory Information
- Add Comments to Files
- Alter File Timestamps
- Locate Files
- Monitor Boot Sectors
- Alter File Attributes
- Protect files

Gamma Tech Utilities
SofTouch Systems, Inc.

Workstation Division
405/947-8080 • FAX 405/632-6537



exited at any point with the RETURN instruction. This means that a programmer can dramatically alter program flow.

Program execution can also be halted with the EXIT instruction, although it is better to exit by calling the program's exit subroutine (for example, CALL EXIT_PROG). The exit subroutine would have been coded to perform any necessary cleanup, such as dropping queues or deleting temporary work files.

CONTROLLING INTERACTIVE PAUSES

Two trace options control pausing during interactive tracing. One of these is the Labels option. This option changes the trace action to only trace and pause at label clauses. This is useful for limiting the amount of trace output and quickly progressing to the next subroutine of interest. Once a desired subroutine is reached, changing the trace action to Results or Intermediates will resume tracing of all clauses.

TRACE L is also useful when examining unfamiliar programs. Since only labels are traced, it can be used to generate an outline of the program flow showing the order in which each subroutine is executed.

The other trace option used for controlling interactive pauses is the number option. When a positive whole number is used as a trace option, the designated number of pauses will be ignored. Using a negative whole number has the same effect on pausing, but trace output is suspended for the designated number of pauses.

The number option is most useful for debugging loops. After stepping through a loop once, the number option can be used to initiate a full execution of the loop without pausing.

INTERACTIVE TRACE OF SUBROUTINES

When a subroutine is to be traced, the programmer can start interactive tracing by adding a TRACE instruction to the source either within the subroutine or prior to the call to the subroutine. In either case, the subroutine will be

	TRACE	TRACE()
Alter trace action from keyboard	Y	Y
Result returned	N	Y
Option is taken as a constant	Y	N
Pause after TRACE clause	N*	Y*
Ignored in file during interactive tracing	Y	N
Accept number option	Y	N
No parameter sets default trace action	Y	N

* Note: The pause behavior of the TRACE instruction and the TRACE() function when entered at the keyboard during interactive trace is not consistent in all implementations or versions of REXX.

Figure 4. TRACE vs. TRACE() behavior

When the trace action is set to 'Results' with the 'R' trace option, the results are preceded by the following symbols:

- >>> Results of an expression including assignments made with the parse instruction.
- >.> Value assigned to the period ('.') placeholder during parsing.

When the trace action is set to display 'Intermediates' with the 'I' trace option, then the following additional result symbols are used to identify the intermediate results:

- >C> Resolution of a compound symbol (symbols which contain a period). The value shown is not the value assigned the symbol but the stem portion of the symbol and the resolved components of the tail. Compound symbols are described in depth in the OS/2 2.0 Procedures Language 2/REXX Reference manual.
- >F> Result of a function call.
- >L> Literal.
- >O> Result of an operation on two terms. Operations include arithmetic, comparison and concatenation.
- >P> Result of a prefix operation (+ or -).
- >V> Contents of a variable.

Figure 5. Symbols used in trace output


```

9 ** If x = verify(string, delim_list, 'M');
>>> "0"

```

Enter: CALL TRACE 'I'

Enter: =

```

9 ** If x = verify(string, delim_list, 'M');
>V> "1"
>V> "01/17/92"
>V> "/- ::"
>L> "M"
>F> "3"
>D> "0"
>>> "0"

```

Figure 6. Reexecuting a clause to get intermediate results

Inserting a trace instruction within a subroutine will cause tracing to begin after entering the subroutine. There is no need to change the trace action when the subroutine ends. That happens automatically. The trace action of the caller is saved, and when control returns to the caller, that trace action is restored.

If a subroutine is called from numerous points in a program, it is not necessary to locate each call and add a TRACE instruction. Instead, a single TRACE instruction within the subroutine will turn tracing on each time the subroutine is executed. However, because tracing does not begin until after entry into the subroutine, it is not apparent where the call to the subroutine was made or what the source of the subroutine's parameters were. Fortunately,

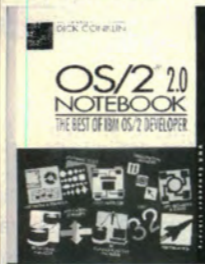
traced. The difference is the programmer's control prior to and following the subroutine. Both approaches have their advantages.

The best of IBM OS/2 2.x

NOW AVAILABLE!

THE OS/2 2.X NOTEBOOK

Second Edition



edited by
Dick Conklin

Every article completely revised! This indispensable operating aid

covers software tools, LAN and communications, plus important advice on presentation manager, client server, developer assistance programs and more. 0-442-01522-4, 1,008 pages, paper, \$34.95

THE BEST SELLER—

CLIENT/SERVER PROGRAMMING WITH OS/2 2.0

Second Edition

by Robert Orfali and Dan Harkey

In-depth tutorials and sample code make this the ideal guide to client/server in the 32-bit environment. Covers all new 2.0 functions. 0-442-01219-5, 1,152 pages, \$39.95

TO COMPLETE YOUR LIBRARY...

For End-Users:

NOW THAT I HAVE OS/2 2.0 ON MY COMPUTER — WHAT DO I DO NEXT?
Levenson—304 pages, \$22.95

THE OS/2 2.0 HANDBOOK
Zack—512 pages, \$34.95

For Programmers:

WRITING OS/2 2.0 DEVICE DRIVERS IN C
Mastrianni—410 pages, \$36.95

COBOL PRESENTATION MANAGER PROGRAMMING GUIDE
Dill—408 pages, \$36.95

LEARNING TO PROGRAM OS/2 2.0 PRESENTATION MANAGER BY EXAMPLE
Knight—240 pages plus 3.5" disk, \$39.95

COMPREHENSIVE DATABASE PERFORMANCE FOR OS/2 2.0'S EXTENDED SERVICES
Tate, et al.—200 pages, \$34.95

OS/2 PRESENTATION MANAGER GPI
Winn—320 pages, \$39.95

Look for these OS/2 2.0 Library Books from  VAN NOSTRAND REINHOLD at technical bookstores nationwide.

Easy Ordering! Call Toll-Free 1-800-842-3636. Or Fax 1-606-525-7778.

1481

6/93



REXX provides a special variable and a built-in function for retrieving this information. The special variable SIGL contains the line number of the clause that transferred control to the subroutine, as shown in Figure 9. The built-in function SOURCELINE() used with the special variable SIGL will return the exact text of the clause that called the active subroutine.

Adding a trace instruction prior to the call to a subroutine establishes the

By modifying the variables that make up the calling parameters and repeatedly calling a subroutine, subroutines can be fully tested in a single execution of the program.

trace action prior to, during, and after the execution of the subroutine. This allows a programmer to see exactly where the call takes place. It provides an opportunity to see and modify the subroutine's calling parameters and to see and modify the result when the subroutine returns. It also allows the programmer to easily reexecute the subroutine.

Since the REXX processor does not pause following a trace instruction, it is recommended that a NOP instruction be added following the TRACE instruction. The language processor will pause after the NOP instruction, which provides an opportunity to examine and possibly modify variables before entering the subroutine. The DEBUG_RX.COMD program illustrates this with the clause TRACE ?R; NOP just before the call to the

```
Enter: say 'The value of x is' x
      The value of x is 34

Enter: do i=0 to var.0; say 'Var.'i 'is' var.i; end
      Var.0 is 3
      Var.1 is 25
      Var.2 is 12
      Var.3 is 58
```

Figure 7. Examining variables with SAY

```
Example:

6 ** If x = y;
  >>> "0" /* Result indicates false */

Enter: x=y /* Modify x with assignment */
Enter: = /* Re-execute */

6 ** If x = y;
  >>> "1" /* Result is now true */
```

Figure 8. Altering program flow with an ASSIGNMENT instruction

```
Enter: say sigl sourceline(sigl)
      126 call subr1 parm_a parm_b /* Line which called subr1 */
```

Figure 9. Displaying a subroutines caller

DIFF subroutine. The TRACE instruction is used instead of the TRACE() function because it is ignored if interactive trace is already switched on.

When the tracing action is set prior to calling a subroutine, that action will be in effect after the subroutine ends. The programmer can take advantage of this by using the equal sign (=) trace command to re-execute the call to the subroutine. The = reexecutes the last clause traced in the current routine. After control is returned to the caller by the RETURN instruction, the last clause traced in the current routine is the clause that called the subroutine. Therefore, entering = will reexecute the

call to the subroutine, as shown in Figure 10.

This is an extremely efficient debugging technique. By modifying the variables that make up the calling parameters and repeatedly calling a subroutine in this manner, subroutines can be fully tested in a single execution of the program.

When using this technique, it may be desirable to quickly return to the calling routine and reexecute the subroutine. There are two ways to do this. Entering TRACE OFF will turn off tracing until the subroutine returns to the caller. Entering the RETURN instruction will change the program flow and force

CSI - The Leader for 20 Years

20TH
ANNUAL
COMPUTER SECURITY
CONFERENCE &
EXHIBITION

Twenty years is a long time in the computer business. And throughout those years, Computer Security Institute has remained the undisputed leader by evolving along with the industry.

For twenty years, we have been refining and improving our annual conference. Because of this, we have become the highest-rated computer security conference in the world.

CSI

COMPUTER
SECURITY
INSTITUTE

The
Computer
Security
Event of
the Year

Join us at CSI's 20th Annual International Computer Security Conference and Exhibition and profit from our years of experience.

Call now to register for the Security Event of the Year.

(415) 905-2626

052793

an immediate return to the caller. If the RETURN instruction is used in a subroutine called as a function, it must return a result, such as RETURN 0, or a syntax error will be generated.

Once a tracing action is set, it will apply to any called subroutines. If there is no interest in tracing a subroutine, entering TRACE OFF will switch off tracing until that subroutine ends. But if a program contains a subroutine that is called repeatedly, such as a logging or messaging routine, it can be an annoyance to constantly be switching off tracing from the keyboard. In this instance, it is preferable to programmatically switch trace off with a clause

During development, it can be very efficient to write and test many routines externally.

in the offending subroutine. TRACE instructions in the program cannot be used for this purpose because they are ignored during interactive trace. But the TRACE() function can be added to a subroutine to programmatically switch tracing off.

USING EXTERNAL ROUTINES

Re-executing subroutine calls with the equal sign (=) trace command works very well with external routines. Whereas an internal routine cannot be changed and retested without ending and restarting the program, an external routine can be modified between calls without ending the calling program. During development, it can be very efficient to write and test many routines externally. Once testing is completed, they can be converted to internal routines. If the file name of the external routine is kept the same as the

```

/* When DEBUG_RX.COMD is executed with the trace instruction */
/* suggested above, the trace scenario might proceed as shown here. */

12 ** Nop;
   +++ Interactive trace. "Trace Off" to end debug, ENTER to Continue.

Enter: say yr1 yr2 sz           /* Examine variables */
      00 10 2

Enter:                          /* Step through subroutine */

13 ** call diff yr1, yr2, sz;
24 ** Diff:                      /* Start of subroutine */

Enter: trace -6                 /* Suspend trace for 6 lines */

32 **      ans = x - y;
   >>>      "-10"

33 **      If abs(ans) >= abs(( x + z) - y );
   >>>      "0"

34 **      If abs(ans) >= abs(x - ( y + z) );
   >>>      "0"

35 **      Return rc;           /* Return to caller */
   >>>      "-10"
   >>>      "-10"

Enter: yr1 = 50                 /* Change calling parameter */
Enter: =                         /* Re-execute subroutine */

13 ** call diff yr1, yr2, sz;
24 ** Diff:                      /* Start of subroutine */

```

Figure 10. Reexecuting a subroutine in interactive trace

label, then the only conversion is to copy the routine into the program.

In determining if this development technique is appropriate, keep in mind that there are no external variables in REXX. All of the caller's variables are hidden from an external routine just as if a PROCEDURE instruction were used. The DIFF subroutine in the DEBUG_RX.COMD sample program is an example of a routine that can be used unchanged either as an internal routine or as an external REXX routine. DIFF does not use any variables from the caller's environment.

It receives data through the calling arguments and returns data to the program as a result. To convert DIFF to an external routine simply copy the routine into a file named DIFF.COMD and delete it from the program.

SAVING TRACE OUTPUT

In the OS/2 operating environment, the REXX processor generally uses the default input and output streams STDIN and STDOUT to read from the keyboard and write to the display. Trace output, on the other hand, is written to the



```

/*****
** REVERSE.CMD
** Filter to reverse lines in a file
** - Uses standard piping/redirection
*****/

trace_value = trace('OFF')          /* interactive trace */
if trace_value = '?R' then trash = trace('R') /* must be off */

do while lines('STDIN')
  line = linein('STDIN')             /* read line from STDIN */
  line = reverse(line)               /* reverse line */
  call lineout 'STDOUT', line        /* write line to STDOUT */
end

```

Figure 11. Resetting trace action in a filter

STDERR output stream. Thus it is very easy to save or print trace output by executing the program with STDERR redirected to a file or a printer. The language processor detects when STDERR has been redirected and will not allow interactive trace to be switched on. When using redirection, STDERR is referenced by the internal file ID (handle) 2. When entered at the OS/2 command prompt, this example will execute the program MYAPP and redirect any trace output to the file TRACE.OUT:

```
MYAPP 2>TRACE.OUT
```

The OS/2 PMREXX utility allows execution and debugging of REXX programs within a Presentation Manager window. It can also be used to save trace output. Among the many features of PMREXX are scrolling and saving output, selective copying to the clipboard, pasting to input, and switching interactive trace on and off with the mouse.

PMREXX is an optional system utility in OS/2 2.0 and 2.1. If it was not originally installed it can be installed from the OS/2 desktop by selecting OS/2 System, System Setup, Selective Install, System Configuration (click on 'OK'), OS/2 Setup and Installation, Optional System Utilities in OS/2 Installation, and PMREXX.

USING RXTRACE ENVIRONMENT VARIABLE

The REXX processor provides an external means of switching on interactive trace in the OS/2 operating environment. If the environment variable RXTRACE is set to ON, the language processor will automatically set the trace action to ?R. It is, therefore, not necessary to modify a REXX source file to switch on interactive trace. The RXTRACE environment variable is most helpful for remotely debugging user problems that cannot be duplicated. The user can easily provide the trace information needed by switching trace on with SET RXTRACE=ON, executing the program with the trace output (STDERR) redirected to a file or printer, and then sending the output to the programmer.

Using the RXTRACE environment variable works quite well unless the ?R trace action is not the preferred or appropriate setting. For instance, ?I might be preferred because it provides more detailed trace information. Or, if the program is a filter, then interactive trace is not appropriate because the language processor will try and interpret the STDIN input stream as REXX instructions. In these cases, some additional code can correctly reset the trace action.

The REVERSE.CMD sample program demonstrates how to program-

matically test and reset the trace action. REVERSE.CMD is a simple filter that takes lines from STDIN, reverses them with the REVERSE() built-in function and then writes to STDOUT. Because it is a filter, REVERSE.CMD will not execute correctly in interactive trace mode. If interactive trace has been switched on using the RXTRACE environment variable, the first two lines of the program will reset the trace action. The first line calls the TRACE() function to query the current trace action and immediately turn tracing off. The current trace action is stored in the TRACE_VALUE variable. The second line checks TRACE_VALUE; if it was set to ?R, the trace action is reset to R, as shown in Figure 11.

AUTOMATING TEST CASES

An important part of application development and maintenance is verifying that a program works as designed in

With the commands available in REXX, it is simple to automate running test cases that will test the complete program or specific routines within the program.

the original release and in successive generations of enhancements and maintenance. This validation is done with test cases. Two problems in running test cases is the time it takes to execute them and the difficulty of designing tests that fully validate all of the program's routines. With the com-

mands available in REXX, it is simple to automate the running of test cases that will test the complete program or specific routines within the program.

It was discussed earlier that testing can be done during interactive trace by entering REXX instructions from the keyboard. Those instructions are interpreted immediately by the language processor. The INTERPRET instruction will also process REXX instructions. Creative use of the INTERPRET instruction allows programmers to exercise control, similar to interactive trace, from outside the program.

The DEBUG_RX.COMD program, shown in Figure 12, demonstrates how to take instructions from a file of test cases and execute them with the INTERPRET instruction. The RUN_TEST_CASES routine in DEBUG_RX.COMD provides this function. If the program is started with a parameter of TEST, then the SIGNAL instruction transfers control to the RUN_TEST_CASES routine, which reads the DEBUG_RX.TST file. As each line is read, it is processed by the INTERPRET instruction.

DEBUG_RX.TST, shown in Figure 13, contains instructions that test the DIFF() routine in DEBUG_RX.COMD. This file looks similar to a REXX file because it contains REXX instructions; however, each line represents a literal string that will be processed by the INTERPRET instruction. INTERPRET requires that all instructions, such as DO...END and IF...THEN, be complete. Therefore, instructions and comments in the test case file cannot span multiple lines. Long instructions are pieced together in multiple assignment statements with a semicolon separating the clauses. The basic test scenario for DIFF() is assigned to the variable TEST in this manner. The test cases are run by changing the values of the variables used in TEST and giving the instruction INTERPRET TEST (RUN_TEST_CASES actually executes INTERPRET INTERPRET TEST). All messages are written to STDERR so they can be easily redirected to a file.

Incorporating this method of saving and running test cases into the develop-

```

/*****
** DEBUG_RX.COMD program to demonstrate automated test cases
*****/
parse upper arg parms
if parms = 'TEST' then signal Run_Test_Cases

trace o
yr1 = 00
yr2 = 10
sz = 2

trace ?r; nop          /* trace DIFF subroutine */
call diff yr1, yr2, sz
ans = result
say yr1 '-' yr2 '=' ans

call exit

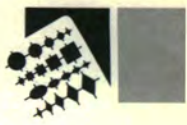
/*****
** DIFF() returns the difference between a pair of counters that
** are reset to zero after they reach maximum value.
** ex. last two digits of the year DIFF(92,00,2) -> -8
*****/
Diff: procedure
  parse arg x, y, z .
  if datatype(x,'W') <> 1 ,
    | datatype(y,'W') <> 1 ,
    | datatype(z,'W') <> 1 then ans = ''
  else do
    z=10**z
    ans=x-y
    if abs(ans)>=abs((x+z)-y) then ans=(x+z)-y
    if abs(ans)>=abs(x-(y+z)) then ans=x-(y+z)
  end
  return ans

/*****
** Read and execute test cases
*****/
Run_Test_Cases:
  TC_input = 'DEBUG_RX.TST'
  TC_test_case = ''
  do while lines(TC_input)<>0
    TC_Test_case = linein(TC_input)
    interpret TC_test_case
    call trace 'OFF'
  end
  call exit

/*****
** Exit
*****/
Exit:
  exit

```

Figure 12. Program that reads and executes its own test cases



improved because there is a record of the actual test cases that have been run. The efficiency of development improves because test cases are captured as each function is developed rather than as an afterthought once coding is complete. The level of documentation improves because the test cases illustrate exactly what a piece of code is expected to do.

SUMMARY

Debugging REXX programs has a number of similarities with other languages and debug utilities. Adding a trace command into a REXX program is equivalent to setting a breakpoint into other languages. Other languages or utilities may also have the ability to examine variables, set variables, and even step through program execution interactively. In addition, REXX has a number of unique debugging features such as the ability to change the trace action, execute instructions, and call subroutines interactively. It can also reexecute a line of code or a subroutine. These features combine to create a very efficient debugging environment.

Steve Clark, *ISSC Corp., Internal Zip WC7D, P.O. Box 2150, Atlanta, Ga. 30301-2150. Clark is a staff programmer in the Programmable Workstation Applications Development department at the ISSC Solution Center, Atlanta. He joined IBM in 1979 and has worked in application development in IBM and ISSC since 1984. He has a B.A. in accounting from Georgia State University.*

REFERENCES

Cowlishaw, M.F. *The REXX Language: A Practical Approach to Programming*, Englewood Cliffs, N.J.: Prentice Hall, 1990. (IBM Doc. ZB35-5100)

OS/2 2.0 Procedures Language 2/REXX Reference Manual (IBM Doc. S10G-6268).

```

/*****
/* DEBUG_RX.TST Test case file for DEBUG_RX.CMD */
/*****

call lineout STDERR, 'Beginning of test case file'
call lineout STDERR, ' '

call lineout STDERR, 'Test cases for DIFF() function'
call lineout STDERR, ' '

/* DIFF() returns the difference between a pair of counters that */
/* are reset to zero after they reach maximum value. */
/* ex. last two digits of the year DIFF(92,00,2) -> -8 */

/* define test (note: all instructions must end with semi-colon) */
test = "call lineout STDERR, 'DIFF('yr1','yr2','sz')';"
test = test "ans = diff(yr1, yr2, sz);"
test = test "if ans == expect then call lineout STDERR, ans '-> OK';"
test = test "else do;"
test = test " call lineout STDERR, 'Test case failed';"
test = test " call lineout STDERR, 'Expected result is: ' expect;"
test = test " call lineout STDERR, 'Actual result is: ' ans;"
test = test " call exit;"
test = test "end;"
test = test "call lineout STDERR, ' ';"

/* run test cases */
yr1 = 10; yr2 = 00; sz = 2; expect = 10
interpret test

yr1 = 90; yr2 = 00; sz = 2; expect = -10
interpret test

yr1 = 49; yr2 = 00; sz = 2; expect = 49
interpret test

say
call lineout STDERR, 'All test cases completed successfully'
call lineout STDERR, 'End of test case file'

```

Figure 13. Sample test cases for DEBUG_RX.CMD

ment process improves the quality of development, and the level of documentation. The quality of the code is



Smalltalk/V is a pure object-oriented language unencumbered by the need to support procedural code. With it, you can create robust production code more quickly than if you were to program from a C source.

BY THEODORE SHRADER

Enhancing Your Smalltalk/V Programs



Theodore Shrader

Unlike C++, Smalltalk/V is a pure object-oriented language unencumbered by the need to support procedural code. With it, you can quickly prototype code, particularly that used to design and test graphical applications. The code can then be further refined to create robust production code, more quickly than if you were to develop from a C source.

INTRODUCTION

Making the transition from C (or another procedural language) to Smalltalk takes time. In a first attempt, developers are likely to shape the Smalltalk classes and methods to follow C programming concepts. For example, a new Smalltalk

oriented programming, there are many ways to refine code, improve performance, and make development and maintenance easier. Unlike a compiled language, Smalltalk allows you to modify code while the program is running and view your changes immediately thereafter. Finally, you can improve program performance by becoming aware of how Smalltalk objects and methods interact. This article focuses on intermediate improvements you can make toward the design and implementation of your Smalltalk/V for OS/2 code.

DESIGNING

One of the best ways to maximize code reuse is to focus on designing the class structure for your program. It helps to first analyze the classes shipped with Smalltalk; examining existing classes and methods can give you an idea of where to place your own classes and which classes can be recycled by your application. Planning ahead doesn't mean that your initial class structure can't be changed, but it will save you from having to rearrange classes and methods in the future.

Wherever you place your classes, determine if it would be better to create a new class as a subclass of an existing one (thus taking advantage of inheritance) or if you should simply add to the methods of one of the base classes. A new class of an object, such as `Corps`, should be made a separate class, whereas new string methods are better off placed in the `String` class since it allows them to be shared by all `String` objects. Make a log of those methods added to existing classes; methods can be forgotten as the change log and source files are

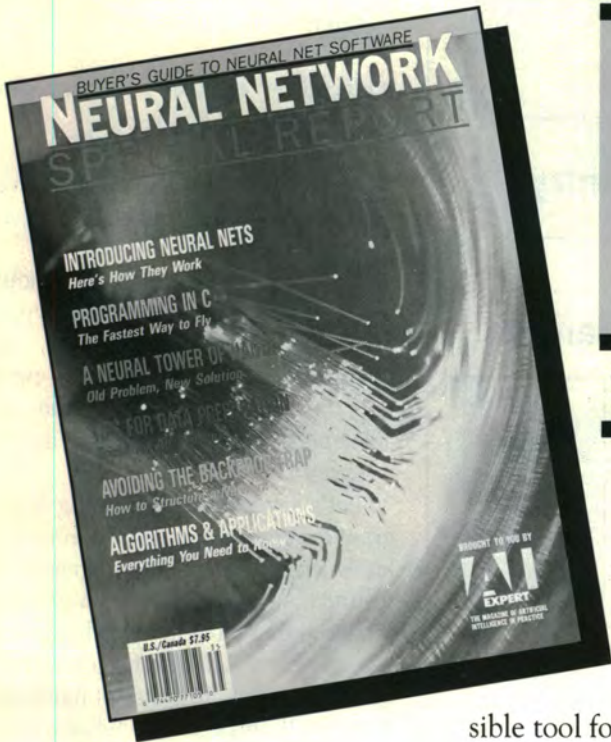
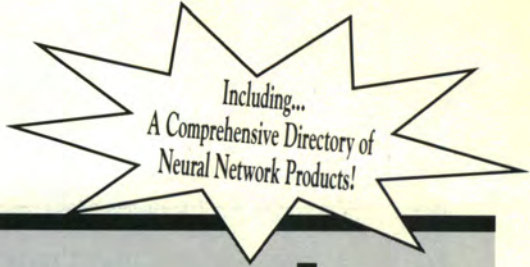
Unlike a compiled language, Smalltalk allows you to modify code while the program is running and view your changes immediately.

developer may contain almost an entire program's methods under one class, rather than subdividing the work among all the classes and reusing as much code as possible. One of Smalltalk's great strengths is its ability to allow code to be subdivided and then frequently and easily reused.

During the transition from procedural to object-



PRESENTS...



Neural Network

SPECIAL REPORT

This publication features in-depth articles on neural networks by industry leaders such as Maureen Caudill and Jeannette Lawrence. The special report includes a comprehensive directory of neural network products.

The Neural Network Special Report is an indispensable tool for professionals who need to integrate neural nets into a business solution today.



GET YOUR COPY TODAY!

TO ORDER

- Telephone 1-800-444-4881
- FAX 1-415-905-2233
- Mail this order form and payment to:

AI Expert
 Miller Freeman
 P.O. Box 105448
 Atlanta, GA 30348-5448

Send _____ copies at \$7.95 each = \$ _____
 Foreign (add \$3.00 each) = \$ _____
 Total Due = \$ _____

Check enclosed

Charge my: VISA MasterCard AMEX

Card # _____ Exp. _____

Signature _____

Name _____

Company _____

Address _____

City _____ State _____ Zip _____

Make check payable to Miller Freeman, Inc. Orders must be prepaid in U.S. dollars. Please allow 3-4 weeks for delivery.

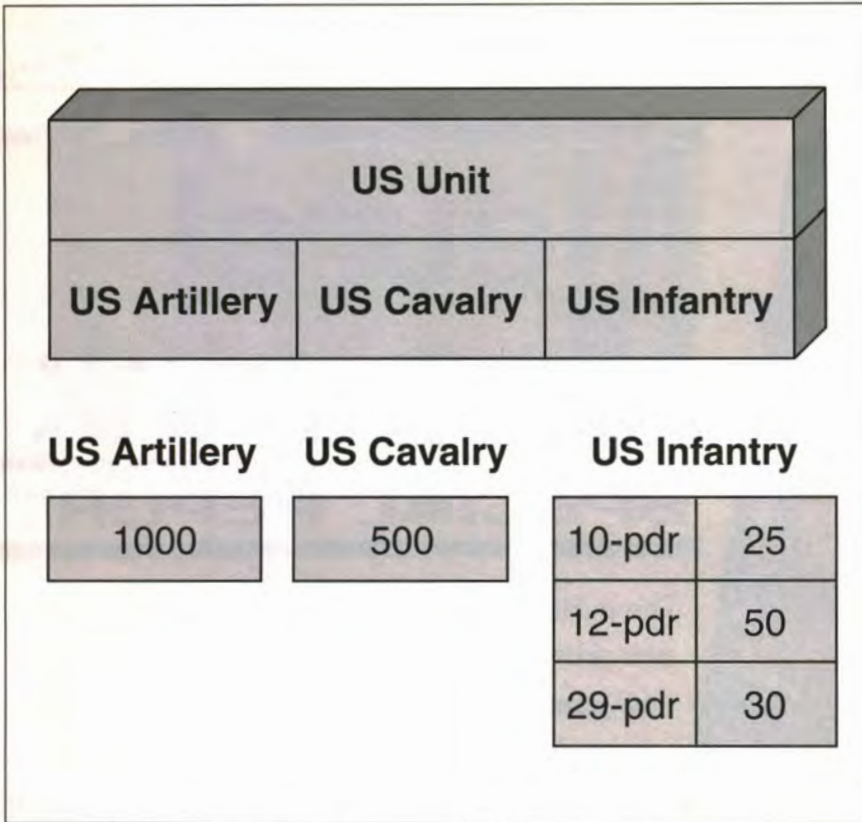


Figure 1. USUnit object subclasses with instances of the strength variable

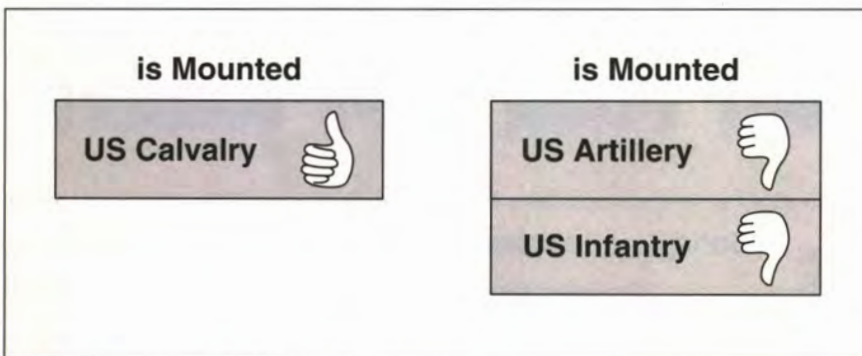


Figure 2. Responses to the isMounted message

compressed. The log will allow you to move the code from one version of Smalltalk to another or from one image to another more easily.

When adding a new class, it is a good idea to prefix the name of the class with a two- or three-letter prefix common to all the classes in your program (for example, USInfantry). This will keep classes sorted together or in clusters in the Class Hierarchy Browser, immediately identifying them to you or

to other developers as those belonging to your program. (Some development environments for Smalltalk separate out the classes used for your application, shielding you from unused parts of the class hierarchy. This makes navigation easier, but with a good naming scheme you can minimize scrolling between classes and retain the same functionality.)

Methods and variables should follow a naming scheme that visually

indicates parameters and object type. For methods, this means placing an abbreviated object name before each variable input. For example, the String method:

```
replaceChar: aOldChar withChar: aNewChar
```

tells the unfamiliar or forgetful developer that character (not string) objects are needed as input.

Instance and class variables should also be named according to the type of object, as with aString, aChar, and strengthDict. The latter variable name tells the reader not only that the variable holds a dictionary, but also that it's a strength dictionary (Oooh).

Let's say you are designing a program to represent units on a map. You know that there will be different kinds of units, such as Infantry, Cavalry, and Artillery, that have unique characteristics, but there will also be commonalities such as the attributes of name and strength. In this example, while a unit's name is a string object for each unit, the strength instance variable takes on a different value for each unit. Whereas for Infantry this variable stores the number of troops, for Cavalry it stores the number of mounted troops. An Artillery's strength, finally, is expressed in the number of its gun batteries.

In fact, instead of an integer value for Artillery, strength could be expressed in terms of a dictionary keyed on the different types of guns, with an associated value of numbers of each. Querying messages sent to each unit could return a different composite strength value for each, with Infantry and Cavalry returning their strength instance values unmodified and Artillery needing to weigh each gun with a strength multiplier, multiplying this value by the number of that type of gun. (Infantry and Artillery are considered polymorphic classes, since they respond to the same message with different behaviors.) The sum of the three

values would be returned as a unit's composite strength.

Everything in Smalltalk is an object; object design can be extended further to make each gun an object, so the strength variable becomes a collection of `USGun` objects. With our base design, we add the following subclasses underneath the `Object` class in the hierarchy:

```
USUnit
  USArtillery
  USCavalry
  USInfantry
```

Figure 1 shows the `USUnit` class structure. The `USUnit` class, which holds the name and strength instance variables, is considered an abstract object because no one will ever create an instance of it. Instead, it is meant to share the methods and variables inherited by its subclasses, which will have instances created of them. After placing dummy methods that return a default answer to common methods in this class, the subclasses can choose to implement class-specific corresponding methods. While `USCavalry` could then answer true to the `isMounted` method, `USArtillery` and `USInfantry` wouldn't have to implement this method because the default method `isMounted` would already exist in the `USUnit` class and would always return false. Figure 2 illustrates responses to the `isMounted` message (method).

USING COLLECTIONS

Collections, with their many incarnations, are one of the most used objects in Smalltalk, yet because of their flexibility and availability, developers tend to overuse them; their proliferation can easily create a drag on program performance. Much of this degradation can be traced back to the overhead need to support each collection, especially the more complex ones, as well as the search algorithms used to access or retrieve data from their vessels. A careful examination of the use of instances of these objects can dramatically increase the speed, efficiency, and capacity requirements of your program. If you know the maximum size a collection will be, create it with that size, as with the statement:

```
dataLineClt := OrderedCollection new:
  175.
```

This is better than using `OrderedCollection new`, which supplies a default of only twelve entries. When the objects to be added exceed the size of the collection, the collection is forced to grow. Even though Smalltalk handles this situation automatically, growing is an expensive, time-consuming task, especially for large collections. Every time a collection grows, it must recalculate its new size, create a new object with that size, and copy the contents of the old collection into the new one. The default growth size for `IndexedCollections`, which includes `OrderedCollections`, is:

```
self size + (self size // 3 + 10)
```

Initializing a collection to its maximum size doesn't bloat the memory needed to store the collections as it waits to be populated, because new entries are initialized to nil values.

Dictionaries are useful collections because they retrieve values immediately. This feature comes with a price, however; Smalltalk must create and maintain their complex structure. Keys can be referenced to retrieve values quickly, as with:

```
unitDict at: '5th Reg'
```

rather than using other structures or stepping through a collection of associations:

```
unitAssocClt do: [ :anAssoc |
  (anAssoc key = '5th Reg')
  ifTrue: [^(anAssoc value)]].
```

Don't shy away from using dictionaries, but be sure to use a collection right for the job at hand. `OrderedCollections` are cheaper to use than dictionaries; arrays are cheaper still.

If a method creates a collection every time it is called, make the collection an instance or class variable. This saves resources otherwise necessary to create and return the collection every time the method is called, especially if

the frequency of calls is high. For example, `USArtillery` could have a class method that initializes and returns a dictionary of guns to be used by the strength instance variable:

```
initStrengthDict
^((Dictionary new)
  at: '10-pdr. Parrott' put: 0;
  at: '12-pdr. Whitworth' put: 0;
  at: '24-pdr. James' put: 0)
```

Now when an instance of `USArtillery` sends a message to itself with `self class initStrengthDict`, it can use the returned dictionary as its base. Since dictionary

Garbage collecting gives Smalltalk a major advantage over C++... it frees developers from having to allocate and deallocate space when interacting with Smalltalk methods and objects.

ies take time to construct, it's a better idea to add the following methods that will create the base strength dictionary once, store it in a class variable, and allow the variable to be retrieved by instances of the object or other objects as well:

```
initClassVars
  strengthDict := self
  initStrengthDict.
freeClassVars
  strengthDict := nil.
strengthDict
  ^strengthDict
```

However, if `strengthDict` changes, it will return the latest changes rather than a fresh dictionary until it is reini-





tialized. Senders of this method should assign their returned dictionary to another value if they want to make distinct modifications.

You can streamline searches of a collection with existing methods such as `select` and `collect` or with subsets that remain stable through a number of searches. Try to reduce the number of times a collection is searched for a value. It is easy to use collections, yet the search time naturally and dramatically grows the larger the collection.

CREATING VARIABLES

Whether local, instance, or class, variables are easy to set in Smalltalk, since their object types need not be declared. As with previous examples, with this flexibility comes the need for restraint. Keeping the number of temporary variables down will keep your stack size from ballooning unnecessarily, especial-

For the most part, with Smalltalk you don't waste time tracking down memory errors.

ly if the methods in which the variables reside are called recursively. Limiting the use of local variables and keeping methods small will speed up your program, since Smalltalk can compile methods as the program runs and store more of them in a cache before having to swap them out for other methods.

Block statements can be handy, especially when setting up error checking rules. The following example sets up a block statement that can be stored in a dictionary and used with a collection of blocks to check values stored in the `ErrorCheck` object. `ErrorCheck` will vary according to implementation, but in this case it stores the values to be checked and uses an instance variable

to store the result of the check:

```
block := [ :anErrorCheck |
  anErrorCheck result: (anErrorCheck
    value > 300) ]
```

If temporary variables are used within a block, however, be sure to set them to nil after they are used. Otherwise, Smalltalk will not be able to garbage collect them from the block's stack and reclaim their space. This will also cause problems if the variable points to an object that you think was already deleted.

GARBAGE COLLECTING

Even those new to Smalltalk are probably aware that Smalltalk busies itself at times, scurrying around to reclaim unused space. Objects that no longer have references are cleaned up and their space is made available to future objects. Objects can be unreferenced either when their local variables are no longer needed after a method has completed or when the variables are set to nil. It's good practice to set all unneeded instance or class variables to nil. You may also want to create `initInstVars`, `freeInstVars`, `initClassVars`, and `freeClassVars` methods and call them upon the initialization and freeing (if postprocessing needs to be done before the object is unreferenced) of an object.

Garbage collecting gives Smalltalk a major advantage over C++ because it frees developers from having to allocate and deallocate space when interacting with Smalltalk methods and objects. However, when dealing with PM structures and API calls, you will need to copy your objects to non-Smalltalk memory to prevent the garbage collection of objects that span the Smalltalk realm. For the most part, with Smalltalk you don't waste time tracking down memory errors.

CONCLUSION

This article has presented a few tips and strategies to help tune your code and recommend possible areas for enhancing your Smalltalk programming style

pertaining to the design and development of Smalltalk/V for OS/2 32-bit applications. The moral of Smalltalk is, "Don't be afraid to rewrite." Most code is not perfect the first time, and Smalltalk allows changes to be made and tested quickly. Your code will become tighter with a hindsight examination, and you'll be able to add to your own list of development and performance enhancements.

ACKNOWLEDGMENTS

The author would like to thank Strike Commander Phil Clem for his time and assistance.

Theodore Shrader, IBM Personal Systems Line of Business, 11400 Burnet Rd., Austin, Texas 78758. Shrader is a senior associate programmer at IBM. He joined the company in 1989, working on graphical interfaces to the Database Manager product. He has worked in graphical application development since then, including work on LAN programs. Shrader holds a B.S. in computer science from the University of Texas at El Paso (Viva Miners!).

REFERENCES

Smalltalk/V for OS/2 Programming Reference

CodeBase 5.0

New for OS/2

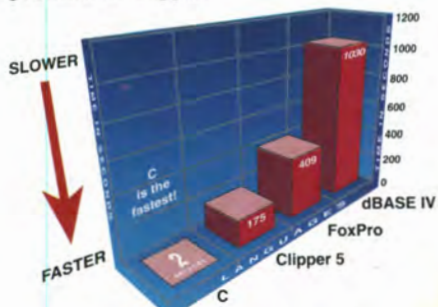
Discover why FoxPro, Clipper, and dBASE were all written in C.

There is a good reason why your database language was developed in C. In fact, there are many good reasons.

C code is small. C code is fast. C code is portable. C code is flexible. C is the language of choice for today's professional developer. With the growing complexity of database applications, C is a realistic alternative. Now with CodeBase 5.0, you can have all the functionality, simplicity and power of traditional database languages together with the benefits of C/C++.

C speed - fast code, true executables...

FoxPro, Clipper, and dBASE were written in C primarily for speed. But those compilers don't really compile, they combine imbedded language interpreters into your .EXE. Now that's slow. For dazzling performance you need the true executables of C. With CodeBase you get the real thing, C code. Consider the following statistics, from the publisher of Clipper:



"Sieve of Eratosthenes" Benchmark for Prime Number Generation Shows C to be incredibly faster!

C size - small executables, no added overhead...

FoxPro, Clipper and dBASE would like you to believe you need their entire development system to build database applications. But

remember, those products are all written in C. So why do you need to lug all their extra code around? You don't. CodeBase is a complete DBMS, in C. No fat executables stuffed with unused code. No runtime modules. No royalties. Just quality C code. CodeBase is just what you need.

C portability - ANSI C/C++ on every hardware platform...

No other language exists on more platforms than C/C++. Why rewrite your entire application for DOS, Windows, Windows NT, OS/2 or UNIX? With CodeBase the complete C source code is included, so you can port to any platform with an ANSI C or C++ compiler. Now and in the future.

dBASE Compatible data, index and memo files...

You want the industry standard. You need compatibility. Sure, dBASE is the standard, but every dBASE compatible DBMS product uses its own unique index and memo file formats. Only CodeBase has them all: FoxPro (.cdx), Clipper (.ntx), dBASE IV (.mdx) and dBASE III (.ndx). Now it's your choice, we're compatible with you.

Announcing CodeBase 5.0

The power of a complete DBMS, the benefits of C

NEW - Multi-user sharing with FoxPro, Clipper and dBASE...

Now your multi-user C/C++ programs can share data, index and memo files at the same time as concurrently running FoxPro, Clipper and dBASE programs. No incompatibilities. No waiting.

NEW - Queries & Relations 1000 times faster...

CodeBase 5.0 now lets you query related

data files with any logical dBASE expression. Our new Bit Optimization Technology (similar to FoxPro's Rushmore technology) uses index files to return a query on a 1/2 million record data file in just a second. Automatically take advantage of this query performance by using our new CodeReporter:

Month of:	Nov, 1992	
Product	Quantity	Value
Crabbase	83	\$75,137.00
Spread Sheet	58	\$2,888.00
Monthly Summary	121	\$47,893.00

Month of:	Dec, 1992	
Product	Quantity	Value
Crabbase	62	\$24,862.00
Spread Sheet	53	\$19,875.00
Monthly Summary	115	\$44,737.00

Summary:	236	\$91,748.00
----------	-----	-------------

To use CodeReporter, simply draw your report, then include it in any program you write. Call 403/437-2410 now for your FREE working model of CodeReporter.

New - Design complex reports in just minutes...

Our new CodeReporter takes the painstaking work out of reports. Now simply design and draw reports interactively under Windows 3.1, then print or display them from any DOS, Windows or UNIX application.

SPECIAL - FREE CodeReporter

Order CodeBase 5 before August 30, 1993 and receive CodeReporter for free! This offer includes our no-risk, 90-day money back guarantee, so order today!



CodeBase 5.0
The C/C++ Library for DataBase Management

Call Now
403-437-2410

SEQUITER SOFTWARE INC. FAX 403-436-2999
Europe 33.20.24.20.14
#209,9644-54 AVE., EDMONTON, AB, CANADA T6E-5V1

OBJECT COMPUTING NOW!



Visual Reality

While the competition plays catch-up, Borland continues to lead with the most graphical C and C++ for Windows, DOS, and OS/2.

A more comfortable place to program

Borland® C++ gives you the most intuitive Integrated Development Environment (IDE) for DOS, Windows, and OS/2.® This ease of use, combined with the fastest C++ compilation speed, means you get your work done faster. And Borland C++ includes time-saving C and C++ code generation and Application Frameworks.™

To set standards you must meet standards

Borland C++ is the only popular compiler that meets the certified ANSI C* and AT&T C++ standards. You can be confident that as the language evolves to make programming easier and more powerful, your Borland C++ code will be ready to take advantage of future versions across all platforms. What's more, applications built

with Borland C++ will deliver faster because you'll have the most advanced features—like templates that make your code smaller, safer, and more reusable.

#1 for real-world application development

With more than one million copies in active use, Borland's C++ is proven, polished, and finely tuned to meet the needs of C and C++ programmers. And Borland has the C++ that's just right for you, from the new Turbo C++ Visual Edition for beginning programmers to powerful Borland C++ for professionals.



Special introductory offer
\$79.95
(regularly \$199⁰⁰)
Offer ends July 31, 1993

90-day, money-back guarantee!
**See your dealer or call now,
1-800-336-6464, ext. 5822**
In Canada call, 1-800-461-3327.

Special introductory offer
\$149.95
(regularly \$495⁰⁰)
Offer ends July 31, 1993

Borland
Power made easy
6362-0001-18



*ANSI C certified by the British Standards Institute. Copyright © 1993 Borland International, Inc. All rights reserved. All Borland product names are trademarks of Borland International, Inc. Offer good in the United States and Canada only. All prices in U.S. dollars. Dealer prices may vary. BI 5498