

Personal Systems

IBM'S MAGAZINE FOR PC PROFESSIONALS

JULY/AUGUST 1996



**Computer
Security**

MPEG SOLUTIONS

Growing the Perfect Application

VisualAge C++ NMAKE ✓

COBOL Collection Classes ✓

Object-Oriented C Code ✓

BULK RATE
U.S. POSTAGE
PAID
PERMIT 1016
FORT WORTH, TEXAS

IBM



OS/2® Warp is a proven performer in the Fortune 1000 companies and the world's financial institutions.

Oracle7™ dominates data management with the industry's most advanced scalability, reliability, distributed computing and replication features.

Together, these powerful technologies let everyone in your company work in an efficient and integrated environment. From laptops to distributed servers, Personal Oracle7™, Oracle7 Workgroup Server™ and Oracle7 Server™ for OS/2 Warp give everyone access to

EXPECT AMAZING THINGS FROM ORACLE7 ON OS/2 WARP.

corporate data. Soon, with Oracle Developer/2000™ for OS/2 Warp, you'll have one of the easiest and most productive platforms for enterprise-wide client/server development.

All of these products are backed by a mutual commitment from IBM and Oracle. This provides proven, reliable technology today, along with greater capabilities far into the future. So start running Oracle7 on OS/2 Warp now, and watch miracles blossom in your business.



FREE 60-DAY TRIAL! Call 1-800-633-0747, ext. 9333 now for a free 60-day trial of Oracle's OS/2 databases and IBM's OS/2 Warp products on CD. Or visit the Oracle and IBM web sites for more information: <http://www.oracle.com> or <http://www.ibm.com>

ORACLE®
Enabling the Information Age™

OS/2 WARP

©1996 Oracle Corporation. Oracle is a registered trademark, and Oracle7, Personal Oracle7, Oracle7 Server, Oracle7 Workgroup Server, Oracle Developer/2000 and Enabling the Information Age are trademarks of Oracle Corporation. IBM and OS/2 are registered trademarks of the International Business Machines Corporation. All rights reserved. All other company and product names are the trademarks of their respective owners.

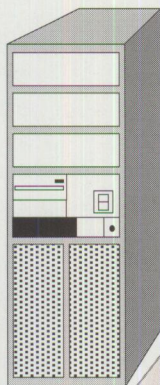
Circle #27 on reader service card.

Lock and Load!

Create, Deliver and Secure OS/2 Desktops over any LAN!

Suggested Retail Price

\$49⁰⁰



- Store/Manage Desktops Centrally! Users get their desktop whenever and wherever they log on!
- Works with any LAN!
- Easily associate Desktops with User ID's!
- Avoid difficult REXX maintenance or INI nightmares!
- Authenticate users' Desktops with your LAN Security!

Suggested Retail Price

\$179⁰⁰



Desktop Commander

A Complete Solution To OS/2 Desktop Control

- Easily take a picture of user's desktops and store it centrally in a small file!
- Standardize any group of workstations or allow users to have their own Desktop wherever they log in! (The "Follow Me™" Desktop)
- Restrict right mouse button options!
- Restore lost or changed desktops instantly!
- Security upgrade available!

Desktop Observatory

A Complete Solution To OS/2 Desktop Security

- Same benefits of the Desktop Commander and more!
- Password protect objects and applications...even the Launch Pad!
- Take background tasks off the Window List!
- Create Security/Audit Logs!
- Drag and Drop File Encryption!
- Prevent Ctrl-Break and Alt-F1 access!
- Inhibit unauthorized file access!
- Prevent clever users from building unauthorized objects!



Information **800.525.1650**

© 1995 Pinnacle Technology, Inc. • PO Box 128, Kirklint, IN 46050 • 317.279.5157

OS/2, OS/2 Ready!, OS/2 WARP and Ready for OS/2 WARP are trademarks of the IBM Corporation.

© 1995 Pinnacle Technology, Inc. All Rights Reserved.

Circle #28 on reader service card.

Let's Get Personal!



When my husband and I first got e-mail between our respective workplaces over a year ago, he told me we needed to be careful not to rely on it too much, thus eliminating our ever talking "face to face." Just the other day I called him about some urgent matter and had to confess that I had trouble remembering his phone number because it had been so long since we had communicated by anything other than e-mail.

Now this is not intended to be a commentary about carrying on personal business at the workplace, nor am I going to elaborate on the necessity of phone conversations between spouses because of extended work hours. What I am becoming more and more concerned about is our impending loss altogether of personal contact with other humans.

How many of you have offended someone with a seemingly innocent comment in an e-mail just because that person couldn't see the expression on your face when you said it? Oh, I know we have a new way of visually "expressing expressions" :-)
<grin>! But there just isn't anything like looking into the eyes of the person to whom you are trying to convey a thought, an idea, a sentiment.

We're already turning to this solitary source of cyber-communication for much of what we used to rely on other media and methods. The *USA Today* recently featured a survey of the "Microprocessor Generation," born since 1971, indicating that 59% thought they would get all their news from the Internet by 2000.

Movies such as *The Net* and, to some extent, *Copycat*, show people who have retreated from society—their only contact with another human comes through a computer display—without happy results.

Scott Adams, in his latest book *The Dilbert Principle*, dramatizes the side effects of working at home, or "telecommuting," with cartoons of Dilbert losing all sense of personal grooming habits and ultimately forfeiting all interpersonal skills. "Day two of telecommuting is going smoothly. I have

eliminated all optional habits of hygiene," proclaims Dilbert as he sits at his computer in his pajamas. "My co-workers are a fading memory. I am losing language skills. I talk to my computer and expect answers." (By the way, Adams' book is a *must read* for the entire corporate world!)

I *like* people! I like being around people. I like stimulating conversation. I like to be challenged. I like to laugh. And I don't want to do all that with a computer screen. IBM's 1996 Technical Interchange in Nashville earlier this year reconfirmed my belief that face to face is better. I enjoyed five days of meeting *Personal Systems'* readers, talking about products, dreams, and the future. I came away with new friends, new ideas, and a sense that I was working with and for a lot of wonderful, real people.

Yes, I know the value of electronic communications in today's demanding, time-constrained world. I know it relieves the frustration of "phone tag." I know "telecommuting" provides more time with family and reduces traffic congestion. But I also see us relying more and more on impersonal communication tools such as voice mail and e-mail (where we can say what we think without anyone interrupting, or where we can deliver bad news without suffering the recipient's immediate reaction). I want to stop withdrawing into a cyber-world before it completely swallows up the personal person that is *me*.

So I'm going to call my husband more often, and I'm going to attend as many conferences as possible where I can meet and talk with my readers and potential readers, and I'm going to get as much real, personal, face-to-face conversation as I possibly can. And when that's not possible, I'll use e-mail and voice mail!

Betty Hawkins, Editor

LAN "Intensive Care Utilities" For IBM LAN Server 3.0/4.0

ICU
FOR YOUR LAN

SUPPORTS
IBM
WARP
SERVER!



New goodies
on our web site:
www.lanicu.com

ORDER TODAY! And receive your copy of LAN Intensive Care Utilities electronically in minutes.

\$970⁰⁰

SITE LICENSING
AVAILABLE AT
SUBSTANTIAL DISCOUNTS

Use your American Express, VISA, MasterCard or Discover Card.



Free demo disk available or download the demo from our BBS, CompuServe or IBMLink.

INCLUDES INDUSTRIAL STRENGTH TOOLS TO:

■ Eliminate flying blind or worrying about corrupted domains

Export your entire domain or just a part into simple and easy to read text files. These text files can then be imported in part or as a whole.

■ Fix your domain using artificial intelligence techniques

Our LAN analyzer goes through your NET.ACC and DCDB files looking for corruption and inconsistencies. Using a rule base, the utility can correct most of the problems encountered. All problems are logged.

■ Put into place a corporate wide security policy on LAN Server

Our LAN analyzer will check each and every account against a rule-based profile and alert you to accounts that contain access and application assignments beyond their privilege and memberships. Accounts that are missing assignments are also flagged. Extra or missing assignments can be corrected on-line immediately. Check hundreds of users in a matter of minutes.

■ Balance your loading

Reports are produced providing cross reference of alias usages as well as permissions by user and group for better control of your LAN.

■ Build and update your LAN FAST!

Rapid add/delete/update of users, groups, aliases, and applications using simple ASCII text files. Templates allow you to define prototype resources or users. Templates allow you to type in only those fields that are unique for users and resources.

Lieberman and Associates Design and Engineering Group

221 N. Robertson Blvd. / Suite C / Beverly Hills, CA 90211

800-829-6263

Phone: (310) 550-8575

IBMMail: USMVHLVH

FAX: (310) 550-1152

CompuServe: 76426,363

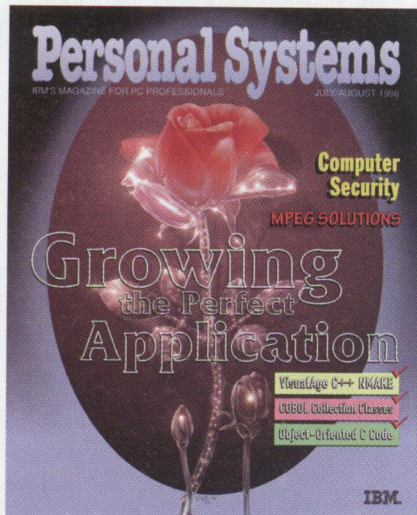
BBS: (310) 550-5980

OS2BBS1: LANUTIL

Internet: 76426.363@compuserve.com

Circle #38 on reader service card.





ABOUT THE COVER

Application development is symbolized in this issue's cover illustration by noted Dallas artist Bill Carr. The crystalline rose, at its base only suggestive of its real world counterpart, transforms into a real flower, just as an application, beginning its existence as formless ideas, evolves into a tangible tool.

Personal Systems Advertising Representatives

Personal Systems accepts paid advertising for applications, products, or services that run on or complement IBM's personal computer hardware and software products. To obtain a media kit and advertising rate information, contact one of the *Personal Systems* advertising sales representatives at the address below.

Lewis Edge & Associates, Inc.
366 Wall Street
Princeton, NJ 08540-1517
(800) ADS-4PSM

Winfield BoyerExt. 124
winfield@edgeassoc.com

Lewis Edge, Jr.Ext. 123
lewis@edgeassoc.com

Peter GriffinExt. 126
peter@edgeassoc.com

Joseph TomaszewskiExt. 125
joseph@edgeassoc.com

Fax (609) 497-0412
CompuServe 72457,3535
Voice (609) 683-7900

Printed in U.S.A.

Contents

FOCUS

- 6 **What's New?**
This issue's "What's New?" brings you a wide variety of products and solutions. You'll find information on Internet technology, personal finance software, server management solutions, backup and recovery solutions, plus much more.
- 17 **Securing Your Communications from Mainframe to Desktop**
How can a corporation balance its strong desire to do business on the information highway and still protect its networked data? What security is available? And how does it work? Find out in this article on network security.

TECHNICAL

- 20 **Creating Applications with VisualAge C++'s NMAKE Facility**
The NMAKE facility, which comes with VisualAge C++, greatly simplifies the process of creating applications. This article discusses NMAKE's advantages and describes how it works.
- 30 **Building Object-Oriented Applications from Existing C Code**
This article shows how the Visual Builder component of VisualAge C++ can extend the usefulness of legacy C code and existing C skills and provide an elegant migration path from C and procedural programming to C++ and OO development. This process is illustrated step-by-step.
- 39 **SOM Collection Classes: A Primer for the VisualAge COBOL Programmer**
This article introduces the concept of SOM collection classes and illustrates the programming techniques for implementing collection classes through a simple application programmed in object-oriented COBOL with IBM's VisualAge for COBOL for OS/2.

LITTLE SOLUTIONS

- 60 **Installing FixPaks via CID**
This little solution discusses setting up a configuration/installation/distribution (CID) installation for FixPaks.
- 63 **Meeting Your Users' IS Needs**
Find out how you can better serve your users, making both your job and theirs easier.

IBM Personal Systems Technical Solutions is published bimonthly by Personal Systems Competency Center, International Business Machines Corporation, Roanoke, Texas, U.S.A. Send any correspondence and address changes to *Personal Systems* at:
IBM Corp. Mail Stop 01-04-60
5 West Kirkwood Blvd.
Roanoke, TX 76299-0015

Personal Systems can be found on the Internet's World-Wide Web at: <http://psc.dfw.ibm.com/psmag/>

IBM customers are eligible to receive the magazine free of charge and can request subscription information by mail or Internet, or by faxing a request to (817) 962-7218. IBMers can subscribe through SLSS (GBOF-7532).

© Copyright 1996 International Business Machines Corporation

19 Going Mobile

I always tell myself, "Remember, it's the data, stupid!"

In this article, the first in a series on going mobile, Bob Angell, consultant with AIMS in Salt Lake City, talks about his journey toward mobile computing—frequently a trial-and-error experience.

50 Choosing the Right MPEG Solution

This article offers helpful advice for choosing an MPEG solution. It covers different MPEG systems, MPEG uses, the Open MPEG Consortium, major decision criteria, and several selection criteria.

56 Computer Security and Implementation

The OS/2 operating system and OS/2 network client software provide adequate security mechanisms if they are implemented properly. This article highlights these mechanisms and shows their strengths and weaknesses, plus presents information from an OS/2 perspective about general security precautions.

65 Corrective Service Information

Refer to this section for the latest maintenance release levels and other software service information.

Editor and Publisher

Betty Hawkins
(817) 962-5799
bhawkins@vnet.ibm.com

Assistant Editor

Lia Wilson
(817) 962-6267
lia@vnet.ibm.com

Business Manager/ Circulation Manager

Van Landrum
(817) 962-5810
vlandrum@vnet.ibm.com

Editorial Assistant

Jeffrey Miller
(817) 962-5823
bonziman@vnet.ibm.com

Subscription Manager

Rose McAlister
(817) 962-5821
rmcalister@vnet.ibm.com

Business Processes Assistants

Dustin Lyon
Terri Sharp-Kubica

Publication Services, Typesetting, and Design

Terry Pinkston/Corporate Graphics
Arlington, Texas

Illustrator

Bill Carr
Dallas, Texas

Printing

Dave Willburn/Motheral Printing
Fort Worth, Texas

Editorial Services

Mike Engelberg/Studio East
Boca Raton, Florida

Reprints

Reprint Management Services
(717) 560-2001

Executive Publishers

Pamela Porter and Beverly Montgomery

Copying or reprinting material from this magazine is strictly prohibited without the express written permission of the editor. Titles and abstracts, but no other portions, of information contained in this publication may be copied and distributed by computer-based and other information service systems.

What's New?

Automated Fax Server



The **T4 MultiLINK** fax server from T4 Systems, Inc. provides sophisticated, mission critical facsimile automation for corporate information system environments from mainframe computer systems to local area networks.

The T4 MultiLINK was designed to perform as a communications processor and offer compatibility with a wide variety of data types including text and graphics, e-mail, document imaging, and voice. You can fax from multiple mainframe host applications, client/server applications, and networks or combinations of these sources simultaneously with ease and efficiency. This flexibility provides a unique solution within the market.

Following are some applications in which T4 MultiLINK can be useful:

- **Communications**—Replace subscription services, or use as a backup communications system for a high speed leased line system.
- **Purchasing**—Broadcast requests for quotes or bids, or send releases against blanket POs from an automated inventory control program.
- **Laboratory/Hospital**—Send patient information, reports, examination results, and memos to physician offices.
- **Billing**—Send statements or inquiries directly to customers.
- **Credit and Lending**—Provide faster response for credit approvals, lease agreements, customer loan applications, and contracts

For more information, circle 1 on the reader service card.

Personal Finance System for OS/2



In Charge! from Spitfire Software is a fully functional personal finance system for OS/2, consisting of multiple systems:

- **Account management**, which handles cash, checking, savings, credit union, and other accounts
- **Budget system**, which supports multi-year budgets
- **Billing system**, which performs payable and receivable functions

Additional systems include tax, securities, insurance, and property management. In Charge! also supports multiple check formats, quick and easy reconciliation, and easy porting from other finance programs.

For more information, circle 2 on the reader service card.

Contract Programming Service



Today's companies are often in the position of providing new graphical, client/server applications even while they support existing legacy systems. **Infrastructure Incorporated**, a consulting firm with years of programming and consulting experience, can now help with both of these demands. Infrastructure can get your new application started by providing education, mentoring, developers, and project leadership for your new client/server project.

Contract programming service addresses the following key issues:

- Hire development skills at a fraction of the costs to hire full time employees. When your project is finished, so is the contract and its associated costs.

- Defer or eliminate the need for additional office space, furniture, and computer systems. Infrastructure can develop your application off site, using their own systems and office space.
- Benefit from a "turnkey" solution. Infrastructure's experienced project managers lead the development effort as well as select and manage a staff to design, develop, model, test, and document your application. They will also train your users and/or maintenance staff.

For more information, circle 3 on the reader service card.

Ultimate RAD Environment for Xbase



OnCmd offers the Xbase developer the ultimate RAD (rapid application) environment and Xbase conversion tool for native OS/2 applications. Developed by On-Line Data, **OnCmd xBase for OS/2** is a powerful application programming system (APS) consisting of tools designed to deliver effective and reliable programs that meet business needs.

OnCmd can be used for new programming as well as for converting applications from other Xbase products such as dBASE, FoxPro, or Clipper. The OnCmd APS provides interactive database creation, a visual design/build utility, project management, report writing, and online help. You also get access to the full OS/2 application programming interfaces (APIs) such as pen, voice, multimedia, and notebooks. OnCmd moves Xbase into the client/server world with an optional transparent index server that results in significant network performance gains.

For more information, circle 4 on the reader service card.

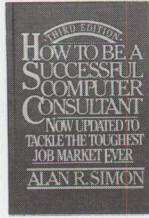
3 BOOKS FOR ONLY \$4.95

when you join the **Computer Professionals' Book Society**

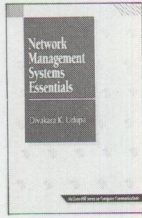
Values to \$150.00



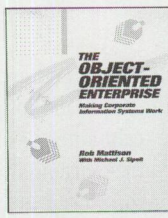
882110X \$22.95
Softcover



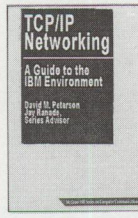
0576173 \$30.00



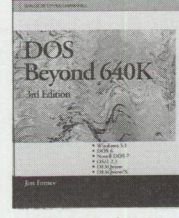
0657661-XX \$50.00
Counts as 2



0410313-XX \$45.00
Counts as 2



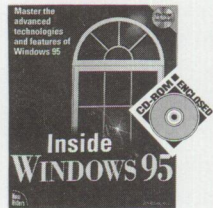
0496633 \$50.00



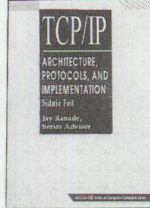
0216096 \$39.95



0110573 \$50.00



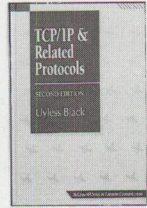
587282X-XX \$40.00
Counts as 2



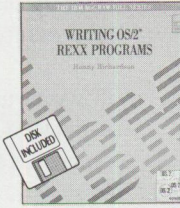
0203466-XX \$45.00
Counts as 2



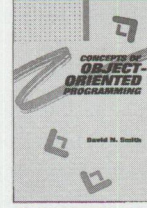
1575472 \$27.95



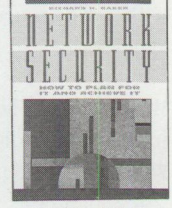
0055602 \$50.00



052372X-XX \$39.95
Counts as 2



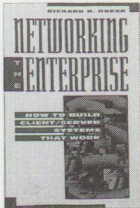
0591776 \$24.95
Softcover



0051194 \$45.00



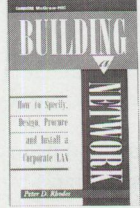
9120288-XX \$39.95
Counts as 2/Softcover



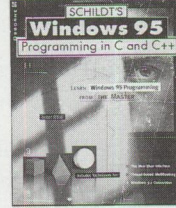
0050899 \$40.00



8820391-XX \$34.95
Counts as 2/Softcover



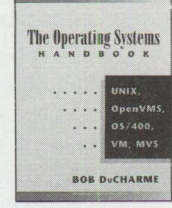
0521344 \$40.00



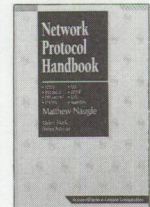
8820812 \$29.95
Softcover



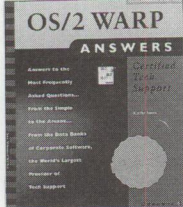
0512795-XX \$40.00
Counts as 2



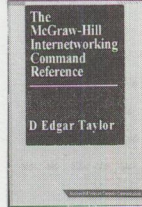
0178917-XX \$49.50
Counts as 2



0464618-XX \$49.50
Counts as 2



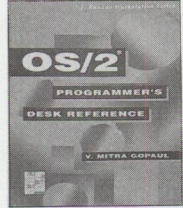
8821150 \$19.95
Softcover



0633010-XX \$60.00
Counts as 2



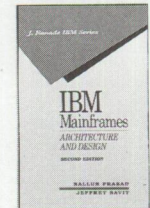
0055920-XX \$45.00
Counts as 2



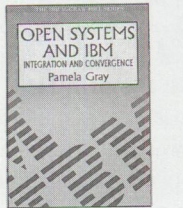
0237484-XX \$44.95
Counts as 2



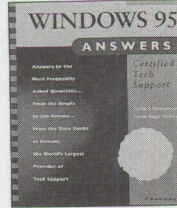
0105162 \$50.00



0506914 \$45.00



7077504-XX \$30.00
Counts as 2



8821282 \$19.95
Softcover

As a member of the Computer Professionals' Book Society...you'll

enjoy receiving Society bulletins every 3-4 weeks containing exciting offers on the latest books in the field at savings of up to 50% off regular publishers' prices. If you want the Main selection, do nothing and it will be shipped automatically. If you want another book, or no book at all, simply return the reply form to us by the date specified. You'll have at least 10 days to decide. If you ever receive a book you don't want due to late delivery of the bulletin, you can return it at our expense. Plus, you'll be eligible for FREE books through our Bonus Book Program. Your only obligation is to purchase 3 more books during the next 2 years, after which you may cancel your membership at any time.

If you select a book that counts as 2 choices, write the book number in one box and XX in the next. All books are softcover unless otherwise noted. Publishers' prices shown. A shipping/handling charge and sales tax will be added to all orders.

If coupon is missing, write to: Computer Professionals' Book Society
A Division of The McGraw-Hill Companies, P.O. Box 549, Blacklick, OH 43004-9918



Computer Professionals' Book Society

A Division of The McGraw-Hill Companies
P.O. Box 549, Blacklick, Ohio 43004-9918

YES! Please rush me the books indicated below for just \$4.95 (plus shipping/handling & sales tax). Enroll me as a member of the Computer Professionals' Book Society according to the terms outlined in this ad. If not satisfied, I may return the books within 10 days for a full refund and my membership will be cancelled. I agree to purchase just 3 more selections at regular Club prices during the next 2 years and may resign anytime thereafter.

Code #'s of my books for \$4.95

--	--	--	--

If you select a book that counts as 2 choices, write the book number in one box and XX in the next.

Name _____

Address/Apt. # _____

City _____ State _____

Zip _____ Phone _____

Offer valid for new members only, subject to acceptance by CPBS. U.S. orders are shipped 4th Class Book Post. Canada must remit in U.S. funds drawn on U.S. banks. Applicants outside the U.S. and Canada will receive special ordering instructions. A shipping/handling charge & sales tax will be added to all orders. © 1996 CPBS IBMPS596

Circle #30 on reader service card

PHONE: 1-614-759-3666
(8:30 a.m. to 5:00 p.m. EST Monday-Friday)

FAX: 1-614-759-3749
(24 hours a day, 7 days a week)

MPEG-1 Playback Card



Visual Circuits has announced an OS/2 driver for the Micro Channel (MCA) and ISA versions of its **ReelTime** MPEG-1 playback card. The card offers true color video playback with CD quality stereo and gives 386-or-greater platforms the ability to play full-screen video at real-time (30 fps) rates. Its conformance to OS/2, OS/2 Warp, DOS, and Windows MCI specifications makes it immediately compatible with a wide range of MPEG-1 titles and applications.

MPEG files can be played back using the standard OS/2 digital video player or Windows media player. Hardware interpolation maintains a high-quality image at any size, and motion-filtering is included to reduce compression artifacts. The MCA interface eliminates installation conflicts and ensures plug-and-play operation. Up to three ISA boards can be installed in one computer to offer multichannel MPEG-1 playback for applications such as video-on-demand, computer-based training, kiosks, and distance learning.

A powerful MPEG utility, the **Visual Circuits MPEG Video Producer (MVP)** is bundled with the ReelTime card. The MVP software will convert AVI format video/audio files into MPEG files, making it possible for existing AVI files to play back in any size window at 30 frames per second in 24-bit color, regardless of the system's VGA capability. The MVP can also convert sequential animation files, such as those produced by morphing or 3-D creation software packages, into full-motion MPEG files. WAV format files can be synchronized with the MPEG video into a single MPEG file that can be played back on any MPEG-compatible platform. An optional CD-ROM recorder lets you create MPEG-encoded disks.

For more information, circle 5 on the reader service card.

Reuse-Based Process Manager for Application Development



The latest addition to the product offerings from Extended Intelligence is **RPM**,

Reuse Process Manager, which is a component-based application assembly process manager tool that enables you to exploit the enormous benefits of reusable code.

You can use RPM to extend an information engineering approach to incorporate software reuse techniques or as a reuse-framework for object-oriented methods. RPM empowers you to capture the maximum benefits from reuse, ensuring dramatic improvements in software productivity and quality.

RPM runs under OS/2, Windows, and Windows NT and operates on a stand-alone PC, LAN, or WAN. Its features include:

- Tool launching to invoke development tools from within RPM
- Fully customizable to adopt methodologies at the enterprise and project levels
- Project planning and management link to estimate and manage projects
- Reuse metrics to predict where to focus reuse efforts for maximum benefits
- Reuse cost models to estimate the cost and savings from reuse

For more information, circle 6 on the reader service card.

Object Management Toolset



From Secant Technologies, the **Secant Persistent Object Manager (POM)** is an object management toolset for building true object-oriented client/server applications. Its powerful mapping services allow relational, ISAM, and stream-oriented data stores to be seamlessly integrated into an application's object model. Object-oriented databases are supported, allowing the integration of legacy databases into object-oriented environments and providing a sensible way to migrate existing client/server systems to object-oriented databases.

An integral component called the **META-DATA COMPILER** compiles an object-oriented schema, including all database mappings, to a working C++ object model. The

increase in quality and reduction in development time are significant since all low-level database programming is eliminated.

POM also provides an integrated object manager that offers powerful transaction services to C++ applications. This allows operations such as rollback, checkpoint, and commit to be applied to in-memory objects as well as to databases.

Running on OS/2, Windows 95, and Windows NT, POM supports database projects by allowing you to efficiently manage persistent business objects using a set of tools and class libraries. It enables support recovery in a system crash and allows users to fully utilize Oracle, DB2, Sybase, and SQL Server to interface objects directly to their databases.

Secant has also announced that it is developing **OpenDoc Control Pack**, a library that will offer more than a dozen control parts that extend the set of the controls supplied by OS/2, Windows, and Macintosh. As general purpose components, these controls can be used by all OpenDoc applications to enter and present information. The Control Pack library consists of the following controls: spreadsheet, calendar, data combo box, cell box, cell combo box, RTF viewer, data field, LED gauge, split bar, hint bubble, control container, thermometer, image button, 3-D panel, GIF control, and more.

For more information, circle 7 on the reader service card.

Cross-Platform Network Tool



From International Software Solutions, **Remote Services Management** (formerly known as PolyPM/2) is an innovative approach to remotely access and manage LAN workstations. It performs a variety of network administrative and maintenance roles, from help desk and LAN Server support to software distribution to end user training.

Remote Services Management supports client workstations running OS/2, Windows, and DOS, as well as manager and gateway workstations. The manager can also run on IBM's new OS/2 Warp

Server, exploiting its power, function, and scalability. A variety of LAN and WAN communication protocols are supported, from serial cables and asynchronous modems to PSTN, NetBIOS, IPX/SPX, TCP/IP, X25, ISDN, and APPC/APPN.

For more information, circle 8 on the reader service card.

OS/2 Security and Desktop Management



Syntegration, Inc.'s release of **The Secure Workplace for OS/2 4.0** features improvements in file and workplace object access control, user identification and authentication, single sign-on, audit trail, keyboard lock and screen blanker, and the Ctrl+Alt+Del plus Alt+F1 disablement facilities.

Workplace object access control allows configuration and enforcement of user- and group-based permissions for files,

directories, and Workplace Shell objects. This version includes Installable Security System (ISS) for IBM's Security Enabling Services for OS/2 Warp (SES), allowing administrators to enforce file access control permissions on a user or group basis.

Identification and authentication is performed internally. At startup, The Secure Workplace for OS/2 presents users with a sign-on window that prompts for user identification and password. Users can also use the sign-on window to change their passwords or shut down the system.

With *single sign-on* to a network operating system or remote host, users identify and authenticate themselves only once. Single sign-on is accomplished by interfacing with the network sign-on coordinator (NSC), user profile manager (UPM), or your own custom program.

Audit trail capability tracks all user activities. You can configure the product to record the audit information locally and/or remotely on a file server.

The *keyboard lock and screen blanker* facility is activated after an administrator defined time-out. After the screen blanker is activated, users must re-enter the sign-on password to regain access to the workstation.

You can disable the *Ctrl+Alt+Del* key combination to prevent unauthorized rebooting of the system, plus you can disable the *Alt+F1* key combination at system startup to prevent users from restoring an unprotected desktop or accessing a command prompt.

For more information, circle 9 on the reader service card.

Add-On for OS/2 Game



Following the release of **Galactic Civilizations 2**, Stardock Systems, Inc. has released **Shipyards 2**, an add-on to Galactic Civilizations that allows you to design your own ships.



Deskman/2 v2.0 by Development Technologies

Deskman/2 2.0 is the next generation of DevTech's award-winning desktop management suite. The release incorporates powerful new features, such as the MultiDesk Personal Desktop Facility, user defined Workspaces, live desktop synchronization and more, all seamlessly integrated into OS/2.

CALL FOR PRICING!

dbfREXX & dbfLIB by dSoft Development New! Version 3

dbfREXX v3:

Simple, affordable database management for OS/2 REXX programs. Now supports stem variables for direct data access. dBASE file management has never been easier.

DSF25 MSRP \$129.00 \$95.00

dbfLIB v3:

Now your C/C++ programs can access dBASE files! dbfLIB provides a consistent set of API's across OS/2, DOS, Windows, Windows NT and Windows 95.

DSF22 MSRP \$195.00 \$159.00

Service, Selection, Smarts and The Best OS/2® Apps!

Remote Services Management Meeting Maker XP for OS/2 PolyPM/2™ by ON Technology

by International Software Solutions, LP



Remote Services Management™ allows an OS/2 "manager" workstation to take control of a DOS, Windows, Windows95, NT or OS/2 "client" workstation across modems, LANs, or WANs. Used for help desks, remote server support, electronic software distribution and automated data collection.

ISS50 Lite MSRP \$199.00 \$165.00
ISS55 Advanced MSRP 349.00 290.00
ISS60 Pro MSRP 499.00 419.00

Why spend hours to schedule a one hour meeting? This client/server group scheduler automates scheduling and rescheduling, even over your network! Runs on OS/2, Windows, Macintosh, UNIX and DOS.

ONT10 MSRP \$349.00 \$319.00

Visual Age C++ by IBM

IBM's powerful C Set++ development environment is now VisualAge C++ v3. Quickly develop 32-bit OS/2 Warp apps or port existing Windows NT apps to OS/2 Warp. Upgrade now from C Set++ version 2.x.

30H1665 (CD-ROM) MSRP \$449 \$375.00



For the latest OS/2 gear, call for our new catalog!

Outfitters for the Information Frontier
1-800-776-8284
<http://www.indelible-blue.com/ib>



OS/2 and IBM are registered trademarks of IBM Corporation. All other trademarks belong to their respective owners. Prices are subject to change. This ad designed on a PC running fast and crash-free on OS/2 Warp.

Circle #31 on reader service card

Now Get **In Charge!**TM

In Charge! is a full function personal and small business finance system for OS/2.

In Charge! supports:

- Multiple sets of financial books
- All types of accounts, from checking to stock margin
- Multiple currencies
- Securities portfolio management system
- Powerful check printing facility
- CheckFree electronic bill payment
- Graphical reports
- Special small business functions

And much more!



In Charge! is available through dealers, or directly from Spitfire Software for only

\$79 + Shipping

(404) 257-0187 • Fax: (404) 255-8032

Circle #2 on reader service card

Galactic Civilizations 2 ships with a few dozen pre-made ship classes for you to build as you achieve the appropriate technology. With Shipyards, you obtain components as you achieve new technologies, which allows you to play the game as you would some of the other games of the genre (Masters of Orion, Ascendancy, etc.). You can put these components, such as warp drive, phasers, Xanthium armor, etc. together to design a brand-new ship. With Shipyards, you have the choice of playing two different types of games (without the expense of paying for two full strategy games).

Also bundled with Shipyards 2 is **Galactic Civilizations 2.1**, which provides enhanced governors who can automatically choose and build social projects in the order you choose from a master governor dialog. Users who prefer to micro-manage their planets still have the option to build their social projects on a planet-by-planet basis.

For more information, circle 10 on the reader service card.

Scalable TCP/IP Switching



Ipsilon Networks has developed the first switching system designed specifically to deliver a quantum leap in price/performance for networks using the Internet protocol (IP). Called an **IP Switch**, this new product combines the intelligence and control of IP routing with the speed and capacity of ATM switching into a single scalable platform. The **IP Switch ATM1600** supports a switching capacity of up to 5.3 million IP packets per second (PPS)—five times the performance of conventional devices.

The IP Switch ATM1600 implements the IP protocol stack on ATM hardware, which operates as a high performance hardware accelerator. Using an intelligent classification scheme, the IP Switch ATM1600 dynamically determines when to switch and when to route based upon the needs of IP conversations, or flows. It maximizes throughput of longer flows by cut-through switching in the ATM hardware and reserves hop-by-hop store-and-forward routing for short-lived traffic.

©1996 Cirrus Technology, Inc.
Unite CD•MakerTM

CD-ROM Mastering for OS/2 Warp

Unite CD•MakerTM lets you create your own custom CDs right from your desktop. Unite CD•Maker 2.0 includes these great features:

- Audio CD reading and writing
- Multi-session capability
- 32-bit APIs
- Long file name support
- Device drivers for many popular CD-recordables
- Beta OpenDocTM interface upon request

Unite CD•Maker combines a multi-threaded design and drag-and-drop features for a powerful and easy-to-use CD mastering tool.

Cirrus Technology
Fourth Floor, 5301 Buckeystown Pike, Frederick, Maryland 21704
email: 102404.3643@compuserve.com internet: http://www.cirrusunite.com

1-301-698-1900

Circle #19 on reader service card

The IP Switch ATM1600 provides a complete backbone solution for network designers who have selected TCP/IP as their strategic protocol suite. In the workgroup, it offers a perfect blend of high performance and infrastructure compatibility.

For more information, circle 11 on the reader service card.

Software Development Tools



JBA International has released the latest version of its software development tool, **JBA Guidelines 3.3**. Guidelines, which runs on OS/2 Warp and OS/2 2.1, is a complete software development environment for generating GUI-based client/server applications anywhere from workgroup to enterprisewide.

Key to the power of Guidelines is the use of JOT, JBA's own platform-independent

programming language, which generates C++ code. Targeted at development teams who currently program in COBOL, BASIC, RPG, C, and C++, as well as the visual programming environments, Guidelines lets you create programs that run on both the PC and server platform. Supported IBM servers are OS/2, AIX, OS/400, and MVS.

Version 3.3 enhances the distributed message layer that provides you with an interface between standardized object calls and the code servicing these calls. Also new for Version 3.3 are enhanced graph control and clipboard access as well as support for two additional compilers: Borland 2.0 C++ and Watcom 10.5 C++, both for OS/2.

For developing graphical interfaces, Guidelines offers a wide selection of controls and time-saving editing functions. Standard Presentation Manager and Windows graphical controls include frames, dialogs, edit fields, and list boxes. Advanced controls include formatted

entry field, notebook, container, grid control, value-set, spin-button, and more.

For more information, circle 12 on the reader service card.

Web Application Server



Deploy live, interactive client/server applications via the Internet with **Amazon**, an Internet Web application server offered by Intelligent Environments. Amazon includes a development tool that connects enterprise corporate applications and databases to the World-Wide Web (WWW). It enables WWW delivery of mainframe-connected client/server business applications, changing today's text-and graphics-based Web pages into interactive applications back-ended by full application program logic, computation, and open architecture database access.

Typical application uses include automated subscription servers, online quote

“Far and away the best prize that life offers is the chance to work hard at work worth doing.”

FOCUS ColoradOS/2 is the only conference of its kind devoted exclusively to OS/2.

RESULTS What you learn will have an immediate impact on the caliber of your work.

QUALITY Conference materials and presenters are widely regarded as unsurpassed.

ANSWERS You'll learn what's happening in OS/2 from the people making it happen.

EXPERTISE Work side-by-side with experts in a hands-on, gloves-off environment.

EXPOSURE Network with a global community of OS/2 experts.

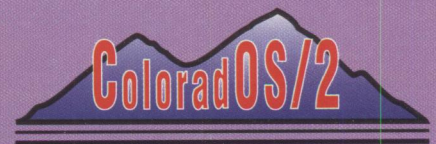
SOLUTIONS Eliminate months of costly trial and error.

EXCHANGE New user or an OS/2 guru, you'll gain and contribute valuable insights.

ENTHUSIASM You'll return with renewed vision, energy and perspective.

Theodore Roosevelt

ColoradOS/2 1996 Go for the prize!



Plan now to attend the Fifth Annual International ColoradOS/2

October 13-October 18, 1996
Keystone Conference Center
Keystone, Colorado

To register or for information call:
(800) 481-3389 US & Canada
(719) 481-3389 International
(719) 481-8069 FAX

Visit our Web page: <http://www.colos2.com>

ColoradOS/2 is a production of Kovsky Conference Productions, Inc. OS/2 is a registered trademark of International Business Machines Corporation.

Circle #32 on reader service card

systems (financial services, retail, etc.), inventory and order tracking systems, in-house information services (Intranet), global technical support databases, and product reference databases (configuration, pricing, availability).

Amazon allows you to program and modify applications with ease—without using low-level languages such as C or PERL—and offers the widest possible range of open connectivity options to existing databases. Amazon links to AM, a powerful corporate client/server programming tool for developing interactive WWW applications.

Working with standard browsers, Web servers, and authoring tools, Amazon accesses and manipulates legacy data and dynamically creates or updates HTML pages. Amazon also enables the use of standard, reusable HTML templates for managing larger sites.

Amazon requires OS/2 Warp or Windows NT 3.51 running a Web server with CGI gateway interface.

For more information, circle 13 on the reader service card.

World-Wide Web Toolkit



Abraxas Software, developer and publisher of language development tools, offers

PCYACC/WEB 1.0, a WWW language toolkit for developing WWW language scripting products. This product addresses the growing WWW market's need for support of a wide variety of scripting languages such as Java, Visual Basic, HTML, VRML, and SGML.

The WWW's popularity is largely due to "scripting languages" that allow non-programmers to create home pages and publish information on the WWW. PCYACC/WEB 1.0 provides the necessary support for all software companies to add these new powerful and simple languages to their products.

Usually, developing WWW scripting languages is a specialized process requiring experts in compiler design. PCYACC/WEB 1.0 provides all the technology needed for any software developer to support these emerging standards in their products. PCYACC/WEB 1.0 is available for OS/2 Warp, Windows 95, Windows NT, Macintosh, and UNIX.

For more information, circle 14 on the reader service card.

Space Monitoring Utility



A low-cost disk space monitoring utility for OS/2 Warp, **DiskStat PLUS!** is now

available from Oberon Software. In addition to providing more textual and graphical display options, this updated edition of Oberon's **DiskStat** has been completely reworked for better Workplace Shell integration, behavior in LAN-based scenarios, and support for multiple execution profiles.

This utility program will appeal to all OS/2 users. Home users can view the status of their system swapper files, while network administrators may use it to monitor multiple disk drives on multiple servers.

DiskStat PLUS! provides four different views of the current state of any disk drive on an OS/2 system or LAN: a textual report, a simple pie chart, a pie chart with legend, plus a historical report. The program also specifically reports on the system swapper file if it is present on the monitored disk.

Also from Oberon Software comes a new personal call tracking and contact management program called **pLog**. This program tracks both incoming and outbound calls, allows for scheduling call backs, and generates multiple reports on the stored data. It features a built-in speed dial list as well as integration with your existing **FaxWorks** phone book.

If you use pLog and FaxWorks together in a LAN environment, pLog can supply each workstation user with his or her own speed dial list while accessing the voice numbers in the corporate FaxWorks phone book.

For more information, circle 15 on the reader service card.

It could be a best seller.



But it's free.

You can't buy the Consumer Information Catalog anywhere. But you can send for it, free! It's your guide to more than 200 free or low-cost government publications about getting federal benefits, saving and investing, staying healthy, and more. Send today for your latest free Consumer Information Catalog.

The Catalog is free. The information is priceless. Send your name and address to:

**Consumer Information Center
Department BEST Pueblo, Colorado 81009**

Backup and Recovery Software



Computer Data Strategies, Inc. has announced the newest release of **Back Again/2 Professional Version 4.0**. To better manage your unattended backups, Back Again/2 includes a new extension to the Workplace Shell folder. You can now schedule a backup program or REXX script to execute at a single date and time, a timed interval, or a combination of days at a chosen time by simply dragging the program icon into the new backup scheduler folder.

With features such as user exits, you can configure any external command to execute before or after a backup is performed. This adds a whole new level of power to your backups. Delete temporary files, log onto or off of a network, broadcast a network message, or shut down and restart a database—all before and/or after a backup.

More robust Workplace Shell integration allows you to back up and restore with drag-and-drop ease. The new, quick backup user interface can dramatically reduce the time to start your backup, and now you can keep your backup sets, message files, and catalogs in their own directories and folders for easy access.

Disaster recovery couldn't be easier. With a new text mode graphical user interface, you can easily restore an entire backup set (including OS/2), subdirectory, or individual file, all from a boot diskette.

For more information, circle 16 on the reader service card.

OS/2 Warp Support for New TR-4 Minicartridge Tape Drives



Seagate Technology, Inc. has announced the availability of three new **Travan TR-4** tape backup solutions for OS/2 Warp. These 1-inch high, 3.5-inch TR-4 minicartridge tape systems provide OS/2 Warp users with 8 GB of compressed capacity, a data transfer rate of up to 60 MB per minute, and backward read/write compatibility with the QIC-3080 format.

Available in either SCSI or IDE interfaces, Seagate's new TR-4 tape backup kits include the internal CTT8000R SCSI, CTT8000R IDE, or external CTT8000E SCSI tape drive with Computer Data Strategies' comprehensive, easy-to-use **Back Again/2** backup and recovery software, a Travan tape, and documentation.

For more information, circle 17 on the reader service card.

Eliminate Network Bottlenecks



NETAnywhere, from Virtual Resources Communication, Inc., uses "Smart Memory," a revolutionary new architecture that extends the reach of existing communications networks, providing a consistent quality of service.

NETAnywhere breaks through the bottleneck experienced in a network environment that is stretched to the limit, taking it to a new level of superior

performance. **NETAnywhere** processes communications data independently of the host's processor. As a result, communications service is consistent, and the host CPU remains free to do what it was designed to do: execute the applications that are critical to your business' success.

The "Smart Memory" architecture as applied in **NETAnywhere** provides a variety of benefits to network servers, including:

- **Scalability**—The modular design allows for growth in port capacities as connection requirements increase.
- **Parallel Processing Capabilities**—The "Smart Memory" bus, running at 240 Mbps, and independent co-processor architecture provide massive parallel communications processing.
- **System Integration**—Any combination up to 135 serial RS232 ports or 45 ISDN BRI ports can be supported by each "Smart Memory" controller through a standard device driver interface.
- **Enhanced Communications**—Each communications co-processor board has three Motorola RISC integrated multiprotocol processors specifically designed for communications applications.
- **Increased Performance**—Not just another multiport controller, **NETAnywhere** is a communications engine that provides high port density and superior performance to meet the communications needs of the emerging era of network-centric computing.

For more information, circle 18 on the reader service card.

ChipChat TxtPager ClientServer

- **Send Text Messages to Wireless Pagers**
Works with all paging service providers...
- **OS/2® based ChipChat® TxtPager Server**
Rock solid, robust, reliable 32-bit code
- **ChipChat Clients for Multiple Platforms**
Includes REXX, C, C++ APIs
- **Transports use TCP/IP or NetBios**
Send pager messages via Internet!



ChipChat Sound Card for PS/2

- **SoundBlaster® compatible for Micro Channel**
Run thousands of multimedia software titles...
- **16-bit CD Quality Record and Playback**
Highest quality sound available today!
- **High Performance 'WaveTable' model**
Makes MIDI music come to life!
- **Supports DOS, Windows® (3.x, 95, NT), OS/2**
Micro Channel Multimedia for all platforms!

ChipChat Technology Group 313-565-4000 <http://www.ChipChat.com>

Circle #34 on reader service card

New Products from Cirrus Technology, Inc.



Several new products are now available from Cirrus Technology, Inc.: **Unite openPage**, **Unite openStore**, and **UniteScan**.

Unite openPage, an OS/2 part for OpenDoc technology, lets you image-enable your legacy host applications, fax images from your PC, scan all your current paper- or folder-based files, and forward documents inside your organization without photocopying. Designed to be integrated into an OpenDoc environment, Unite openPage provides the capability to scan and view document images. Added value in Unite openPage includes OpenDoc methods to control scanning either from a low level or through the image viewer, an easy-to-use scanner configuration notebook, and support for many file formats.

Unite openStore provides native OS/2 device drive support for a variety of SCSI-based optical and CD-recordable jukeboxes. Unite OpenStore's complete set of 32-bit APIs is included to develop flexible and efficient jukebox management applications. A graphical user interface lets you start, control, tune, and stop Unite openStore, as well as perform administrative functions such as importing, labeling, and exporting platters.

Unite openStore can be integrated into the OpenDoc environment and takes full advantage of OS/2's multitasking capabilities to manage thousands of store and retrieve requests while simultaneously reading and writing to all drives.

UniteScan, available for OS/2 Warp, is a production scanning product, ideal for high volume paper input. It offers the latest in high-powered scanning for the Kodak image scanners. UniteScan offers configuration control and is ready to use "right out of the box"; no programming is required. You may select the scanner configuration and the frequency to view images as scanning occurs. It comes with 18 scanner configurations representing mode 01 through mode 18 for Kodak devices.

For more information, circle 19 on the reader service card.

Server Management



Client Server Networking (CSN) recently announced **WATCHIT PLUS**, one of CSN's OS/2 Warp Server utility products that operates across all IBM platforms, including AS/400, AIX, and MVS/VM.

WATCHIT PLUS provides more extensive server management for larger and more complex configurations than those supported by its predecessor, **WATCHIT**. WATCHIT PLUS will not replace WATCHIT since WATCHIT remains the primary tool for single server management.

From one location, WATCHIT PLUS provides the ability to monitor all the servers in a domain. It makes accessible all the information needed to optimize any IBM LAN Server. From a single OS/2 platform, WATCHIT PLUS collects data from OS/2 Warp, AIX, MVS/VM, or AS/400 implementations of IBM LAN Server. WATCHIT PLUS collects statistics about resource and user activity that you can graphically analyze to improve performance and anticipate capacity problems.

Even though WATCHIT PLUS provides instant analysis, you can still examine the data after the collection period. WATCHIT PLUS's analysis programs include graphs of 14 significant LAN Server characteristics. Resource usage levels are compared to the operational IBMLAN.INI control file. Each display of a sample provides 16 items of detail for the collection period.

For more information, circle 20 on the reader service card.

Small Business and Personal Finance Software



Computer Interface Corporation recently announced an upgrade of its OS/2 financial management program, **Check+**. Check+ Version 2.1's many new, powerful features include ASCII import and export; on-screen viewing of predefined and custom reports; simpler, quicker entry for single item deposits; improved Quicken import; an improved user interface with better graphics and quicker response time; and more user-defined options.

These enhancements complement the easy-to-use check writing, credit card, investment, loans, and property management features of earlier versions. Check+ includes sophisticated report features that summarize data from multiple accounts and a user interface that is easy to master and remains consistent throughout the program.

Also available from Computer Interface Corporation is Version 1.2 of **Contact Connection**, a flexible and easy to use native OS/2 contact management program for the home and office.

Contact Connection allows you to enter an unlimited amount of data about your personal and business contacts. You can use it to print form letters, envelopes, and mailing labels, as well as automatically dial phone numbers. Each contact can be assigned user-defined codes, so that contacts can be easily grouped and retrieved. Links can be created between contacts, allowing you to easily move between related individuals and businesses.

For more information, circle 21 on the reader service card.

Software RAID Solution for OS/2 Users



EZRAID PRO and **EZRAID Lite** from Cyranex Corporation support IBM's OS/2 Warp Server. Bringing the benefits of RAID technology to the OS/2 Warp Server platform, EZRAID PRO serves as a powerful add-on to the larger network implementation, while EZRAID Lite provides disk fault tolerance support for smaller networks.

Using disk mirroring, data striping, and disk spanning, EZRAID flexibly creates software-managed disk arrays. It supports all disk interfaces, including Enhanced IDE, Fast Wide and Ultra SCSI, P1394, SSA (Serial Storage Architecture), and Fiber Channel. To avoid system downtime, EZRAID offers hot swap and hot spare capabilities.

For more information, circle 22 on the reader service card.

N O W A V A I L A B L E

"The Power of Many"



In response to an overwhelming number of requests, posters of the stunning cover of the May/June issue of */AIXtra* magazine are now available for purchase. Reproduced to display all of the magnificent colors of the original by highly acclaimed artist Bill Carr, each full-color poster measures 19"x25" and is suitable for framing.

Supplies are limited and we urge you to place your order as soon as possible.

Name: _____

City: _____

Company: _____

State: _____ Zip: _____

Address: _____

Business Telephone: () _____

I have enclosed a check or money order for _____ posters @ \$8.95 each + \$3.95 shipping/handling (per order)
Checks must be drawn on U.S. bank in U.S. dollars. (For quantity orders/discounts, call 218-723-9477.)

Charge to: VISA MC AMEX Expires on: _____ Number: _____

Make checks payable to, and mail to: Superior Fulfillment, 131 W. First Street, Duluth, MN 55802-2065

Server Mirroring Capability for OS/2 Warp



Vinca Corporation offers **StandbyServer for OS/2 Warp**, a transaction-based server mirroring solution that delivers full server fault tolerance to the OS/2 Warp platform. The product allows a hot secondary server to be connected to the main server. If the main server fails, the standby machine takes over automatically. Now mission critical applications running on the OS/2 Warp platform, such as Lotus Notes, can be protected from expensive downtime.

In addition, **StandbyServer for OS/2 Warp** is specifically designed to take full advantage of the remote notification and monitoring features in NetFinity, IBM's powerful systems management software. These user-definable features give the network manager the ability to customize notifications that can be received on a remote workstation or a digital pager when a server fails. Network managers can also monitor critical information during normal operation or during a switch-over from the standby machine.

StandbyServer for OS/2 Warp has been tested and proven compatible with Lotus Notes. In addition, its expanded functionality supports all existing LAN Server networks, including entry level.

For more information, circle 23 on the reader service card.

OS/2 Warp Administrator's Survival Guide



The **OS/2 Warp Administrator's Survival Guide**, from SAMS Publishing, gives OS/2 Warp administrators 1,000+ pages packed with information written by IBM and industry experts. This book provides all you need to effectively install, run, and manage OS/2 Warp, including:

- Details of OS/2 Warp Connect and its components—including IBM Peer for OS/2 and the new integrated install

- Instructions on network printing, DATABASE 2 for OS/2 administration, Communication Manager's SNA Gateway, and systems management topics
- Installation tips and techniques, such as multi-user remote installation and using the NetWare requester as the redirector
- Tuning and tweaking techniques

The book's CD-ROM extends its value further by providing evaluation versions of IBM systems management products; demonstration copies of popular third-party utilities such as Back Again/2 3.1, PartitionMagic 2.0, and Graham Utilities 1.0; shareware and freeware tools; and code from the authors.

OS/2 Warp Administrator's Survival Guide: ISBN 0-672-30744-8.

For more information, circle 24 on the reader service card.

IBM Credit Extends 3.9 Percent and 5.9 Percent Financing



IBM Credit Corporation has extended its 3.9 percent total solution financing and 5.9 percent preferred rate financing offering to additional IBM personal computers.

With the extension, best-credit customers who acquire selected IBM PCs and finance them in conjunction with an eligible IBM AS/400 Advanced Series business computer and IBM software and services as part of their information technology solution will qualify for financing as low as 3.9 percent.

IBM Credit Corporation has also extended its 5.9 percent preferred rate financing program to additional IBM PCs. With this extension, customers acquiring an eligible IBM PC and a qualifying IBM AS/400 or IBM RISC System/6000 processor can finance the IBM PC with these same attractive monthly rates. Previously,

the preferred rate financing program supported customers' acquisitions of selected IBM PCs, including certain IBM ThinkPad notebook computers and IBM PC Servers.

Visit the IBM customer financing home page at <http://www.financing.hosting.ibm.com>.

PR Specialists Simplify Communications



As the number of hardware and software developers increases, so does the caliber of competition, creating an increasing need for public relations within the high-tech industry. These increases are leading developers to seek out firms to help disseminate their messages to trade journals, newspapers, syndicated columnists, Internet bulletin boards and chat rooms, television, and radio.

P.R. Unlimited, Inc. specializes in publicizing high-tech products and services to all media outlets. The firm uses all forms of media as well as the Internet to disseminate information and promote products and services in the OS/2, Windows, HP, UNIX, and Macintosh environments.

High-tech companies face the following dilemma: How do we balance the information illustrating an extremely technical concept to both highly technical and non-technical writers and editors? A solid public relations campaign should be a significant component of a successful new product and/or service release. Feature articles in magazines and newspapers help increase sales by providing new opportunities, raising employee motivation, and enabling small companies to grow into larger markets.

A PR firm must customize each campaign to meet clients' needs and schedules. It must become an extended voice for each company and be a solid part of these companies.

For more information, circle 25 on the reader service card.

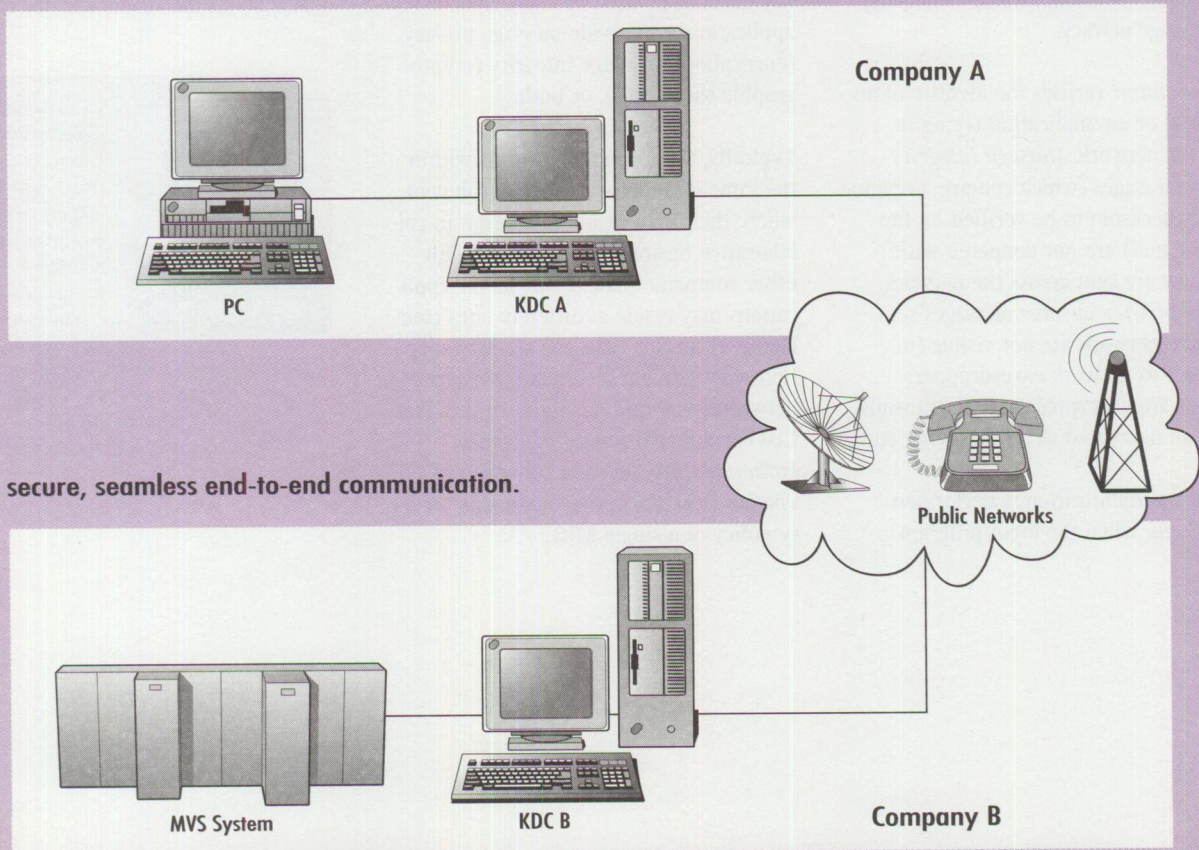
Securing Your Communications from Mainframe to Desktop

By Laurie Anderson

Companies who plan to conduct business on the Internet must also plan for the security risks this new business opportunity creates. While companies can use the Internet's infrastructure and information to access customers, they open themselves up to possible infiltration. People outside a corporation (e.g., corporate spies, hackers) and inside a corporation (e.g., malicious or unhappy employees or contractors) can attempt to gain unauthorized access to its intellectual trade secrets and confidential data. How can a corporation balance its strong desire to do business on the information highway and still protect its networked data? What security is available? And how does it work?

By adding network security services and securing its applications, a corporation can do business securely on the open network, as well as its intranetwork. Benefits include the ability to securely provide services to customers without compromising the corporate network, securely interact with a competing firm, and work with vendors who work closely with its competitors without

Desktop to mainframe . . .



secure, seamless end-to-end communication.

Figure 1. Example of Secured Desktop to Mainframe Communications

providing an avenue into the corporation's network. For example, an airplane manufacturer can collaborate with a competing vendor to build an economical jet engine, or a market research firm can provide an information clearinghouse to only paid subscribers. In the area of financial applications, customers can enjoy secured electronic commerce. These security services safeguard networks for doing business in the 21st century.

Three security features are required for conducting business on an open network without compromising security: authentication, message integrity, and message privacy. For the greatest level of security, these features need to be available at the application layer rather than at a lower layer, such as the network protocol layer.

While there are many proprietary, alternative security solutions available that offer these features, only one is a fully functional Internet security standard. Kerberos, developed by MIT as part of Project Athena and chosen as the basis for Distributed Computing Environment (DCE) security, provides secure authentication, message integrity, and message privacy.

Authentication verifies the identity of an individual or an application trying to access the network. *Message integrity* ensures messages (which contain a cryptographic checksum to be verified on the receiving end) are not tampered with when they are sent across the network. *Message privacy* ensures messages sent across the network are not visible (in clear text) to network eavesdroppers. Messages are encrypted prior to transmission, then decrypted at the receiving end.

Typically, authentication is performed during login when the login program

requests a ticket from the Key Distribution Center (KDC). The KDC receives an encrypted message from the login program, including a known piece of data encrypted in a key derived from the user's password. The KDC stores each user's key and employs it to authenticate users by decrypting messages. If a message can be decrypted, the user identity is verified, and the KDC issues a ticket. This process requires that all KDCs be secured physically (placed in a secure room with minimal physical access) and electronically (only encrypted sessions are allowed into the system) to ensure against compromising the database.

After authentication, users have transparent, yet secured, access to any secured applications to which they have authorized access. When a user executes a secured program, such as a database application, the program uses the original ticket granted by the KDC to transparently request a service ticket from the KDC, which securely identifies the user to the remote server. After some security checks, the KDC grants a ticket, which includes a key that can be used by client and server applications to provide message privacy (encryption), message integrity (cryptographic checksums), or both.

Typically, users run applications within the same local networking environment where their KDC resides; however, in collaborative business relationships with other companies, client and server applications may reside at different sites (see Figure 1). In this case, the security solution must provide secure authentication between corporations. Kerberos provides that capability through *inter-realm authentication*, where a collection of systems (a *realm*) exists under the security policy of a single KDC.

In Figure 1, the PC user gains secure access to data on the MVS system. First, the PC user authenticates with KDC A. When the user launches the database program, it transparently asks KDC A for a ticket to the MVS database program. KDC A cannot grant that ticket and informs the program that it needs to communicate with KDC B. The PC database program then asks KDC B for a ticket. KDC B can accept KDC A's authentication of the PC user and can grant a ticket for secure communications.

With CyberSafe Challenger authentication software, users securely gain access to all their network resources with one password. After authentication, users have transparent and secured access to any applications that have been secured and to which they are authorized, thereby providing secure communications throughout an enterprise. CyberSafe offers developer toolkits for securing in-house applications, and provides consulting and educational services to help large enterprises secure their networks.

Circle 26 on the reader service card.



Laurie Anderson is director of communications at CyberSafe Corporation, Issaquah, WA, where she is responsible for CyberSafe's written materials, such as marketing communications and

technical documentation, as well as their educational services. She can be reached at laurie.anderson@cybersafe.com.

Going Mobile

I always tell myself, "Remember, it's the data, stupid!"

By Bob Angell

Bob Angell, consultant with AIMS in Salt Lake City, talks about his journey toward mobile computing—frequently a trial-and-error experience.

That's right, folks, it's the data that's important! There are literally thousands of appropriate computing solutions in today's business environment geared towards allowing you to gather, analyze, and manipulate your company's data in a timely fashion (or at least they should). Therefore, the bottom line for any computing solution should be centered and focused around the goal of data availability and usability.

Most companies are looking for ways to reduce overhead costs while trying to increase overall profits. Some companies (most notably IBM) have disbanded the traditional office (assigned office space, desk, personal effects, golf clubs in the corner for those tough clients <grin>) for the mobile office—shared office space while in the office with the ability to access, process, and simulate company data at any time while out of the office.

Spurring the move toward mobile computing is the widespread acceptance and affordability of laptop and notebook computers.

The memory of today's laptop computers can even be expanded to address up to 40 MB of RAM. In addition, removable 1.2 GB hard drives are not uncommon. Only the system board and its processor cannot be upgraded. Mobile technology is now quite mature and, in my opinion, ready for those who care to and need to go mobile.

A few months ago, I decided to take my office "mobile" after countless headaches from trying to keep the data on my desktop machine synchronized with my laptop. This was not a first attempt to go

completely mobile; I have learned many lessons since I first tried. At that time, docking stations were just being introduced. I did have an external monitor hooked up to my laptop, although it only had CGA resolution. My first mobile computing attempt was also before PC cards (PCMCIA) were invented.

With the addition of a docking station, I found that I could significantly expand my current laptop configuration (LAN, video, hard drives, etc.).

My laptop is an IBM ThinkPad 755CE with built-in MWave technology, 24 MB of RAM (the standard configuration is 8 MB plus a credit-card-sized 16 MB memory module), and a 540 MB hard drive. I also have one 540 MB and two 360 MB hard drives that I use as backups and with other operating systems. Although these drives are tolerable (because I use a docking station with a 4.3 GB small computer system interface [SCSI] hard drive and an external CD-ROM, tape backup unit, and extra modem), I recommend that you get at least an 810 MB hard drive for day-to-day use. My docking station also has enough room to put in a network card and perhaps use a different video card.

I am slowly (in my spare time <grin>) networking the docking station to a RISC System/6000 and am considering attaching an AS/400 as a client in the next few months. This is a good solution if you don't need all the data that resides on your docking station hard drive; however, I have found that this setup is not quite sufficient for my needs. I will be transferring the docking station's 4.3 GB hard drive to the RISC System/6000 so that when I dial

in from the field (using a product like LAN Distance to log in to the network from a telephone), I can have my data (AIX and OS/2) readily accessible.

A full-time connection to a RISC System/6000 is a great way to exploit a full realm of mobile computing opportunities. Although it can be costly, it might make sense for larger companies. You do not, however, need a RISC System/6000 to take full advantage of mobile computing—a medium to high-end PC loaded with the OS/2 Warp Server family of products will work just as well. With one of these IBM solutions, you can access your data anywhere in the world, 24 hours a day, 7 days a week!

In my next "Going Mobile" article, I will continue to discuss the advantages and pitfalls of accessing data while on the road and discuss how transmission control protocol/internet protocol (TCP/IP) can become your friend.



Bob Angell is a principal with Applied Information and Management Systems (AIMS) in Salt Lake City, Utah. A management consulting firm, AIMS specializes in management information

systems integration, OS/2 and RS/6000 development and integration, total quality improvement engineering, and other related services. Bob's specialties include multiplatform data integration, database design and development, simulation and modeling of complex environments (neural networks), and cross-platform software development. Bob can be reached through the Internet at bange11@cs.utah.edu.

Creating Applications with VisualAge C++'s NMAKE Facility

The NMAKE facility, which comes with VisualAge C++, greatly simplifies the process of creating applications. This article discusses NMAKE's advantages and describes how it works.

While learning C++, I began picking up the user interface part of IBM's VisualAge C++ Open Class Library. I was looking forward to finally being able to write OS/2 Presentation Manager (PM) programs relatively easily.

Mark Fisher
IBM Corporation
Roanoke, Texas

Early in the process, however, I found that writing realistic applications required tracking and properly assembling several different kinds of files. It became increasingly difficult to coordinate independent files as

they grew in number. For example, as you write larger and more complex applications, you will create a number of "human-readable" source files. The source files are used to create different types of binary, "computer-readable" intermediate files. The intermediate files are eventually combined to build the final executable application.

Let's assume you change some of the source files. You have the option of creating all the intermediate files over again, then combining all of the intermediate files into the final executable



file. This results in unnecessary work that can take your computer many minutes or even hours to process. Ideally, you would like to process only the changed source files and then process only the newly created intermediate files.

Fortunately, the NMAKE facility in VisualAge C++ comes to the rescue. A properly constructed NMAKE file specifies all the parts that make up an application. The NMAKE file can sort through all the source files, find the ones that have been changed, create only the new intermediate files that are needed, and then combine only the files that are needed to build the revised application. You no longer have to keep track of which files have been changed or how they are combined. There is a real sense of relief and pleasure when watching a properly constructed NMAKE file do its thing.

The NMAKE documentation (in the *VisualAge C++ for OS/2 User's Guide*, S25H-6961) is detailed, whereas in this article, I use only a few details to build a series of working examples. I hope to show how NMAKE actually works; this will help you to better understand the detailed documentation. (You may find other products referring to a MAKE facility; NMAKE and MAKE are basically the same thing.)

Use any editor when creating the files shown in the examples in this article. In some examples, you may find it convenient to open several files at once. If you use the 1xpm editor that comes with VisualAge C++, you will find that if you open a file from an OS/2 window, the window remains open. You can then use 1xpm again to open more files from the same OS/2 window.

If you use the epm editor that comes with OS/2, you should use `start` for the first session:

```
start epm foo.cpp
```

This leaves the OS/2 window open. Then use `epm` to edit additional files from the same OS/2 window:

```
epm foofoo.cpp
```

We'll start our examples with the usual "Hello, world" program. (By the way, the

examples were prepared on a system running OS/2 Warp and VisualAge C++ Version 3.)

Example 1

In example 1, you'll enter a simple program to make sure that your computer can compile and link it. The program will be modified in examples 2 and 3 to show how NMAKE works.

Create the C++ program shown in Figure 1, and save it as `x1.cpp`.

You can then compile and link the `x1.cpp` program simply by entering `icc x1.cpp` on the OS/2 command line.

Note: This is not an article about compiling and linking options, but I will explain options when necessary. The defaults create an executable (EXE) file named `x1.exe` that runs in either an OS/2 window or an OS/2 full screen. It is not a PM application.

Finally, enter `x1` on the OS/2 command line to run your application and get the "Hello, world" greeting.

You can automate the compile and link process by putting the `icc` command into an OS/2 command file. To do this, create and save the `do_x1.cmd` file in Figure 2.

Now, to create your EXE file, simply type `do_x1` on the command line.

This is admittedly a simplistic example of the benefits of automation. In the next example, I introduce NMAKE; you'll begin to see the benefits of using NMAKE, even in simplistic cases.

Example 2

In real applications, the code for functions and objects is split out from the `main()` function, so do that first. Create and save the `x2.cpp` file in Figure 3.

Next, create and save the `do_x2.cpp` file in Figure 4, which you use to compile and link `x2.cpp`.

Enter `do_x2` on the command line several times to make sure that the files compile and link. (Every time you enter `do_x2`, the compile and link functions are done again, even though the source `x2.cpp` has not changed and there is no need to compile and link again.)

```

/*****
 * x1.cpp
 *****/
#include <iostream.h>
void main(void)
{
    cout << "Hello, world.\n";
}

```

Figure 1. `x1.cpp`

```

REM *** This is do_x1.cmd ***
icc x1.cpp

```

Figure 2. `do_x1.cmd`

```

/*****
 * x2.cpp
 *****/
#include <iostream.h>
// prototypes -----
void say_hello(void);
// main() -----
void main(void)
{
    say_hello();
}
// functions here -----
void say_hello(void)
{
    cout << "Hello, world.\n";
}

```

Figure 3. `x2.cpp`

```

REM *** This is do_x2.cmd ***
icc x2.cpp

```

Figure 4. `do_x2.cpp`

```

# x2_1.mak
x2.exe : x2.cpp
icc x2.cpp

```

Figure 5. `x2_1.mak`

Doing It with NMAKE

Now let's look at automating the process using NMAKE. Create and save the `x2_1.mak` makefile shown in Figure 5.

The `mak` extension is not necessary, but it helps you to recognize a makefile. A

```
# comment here
targets ... : dependents ...
    command
.
.
.
command
```

Figure 6. Description Block Format

```
#####
# x2_2.mak
# invoke with nmake /f x2_2.mak
#####
# Block 1
x2.exe : x2.cpp
@echo -----
@echo - Block 1
@echo - compiling and
@echo - linking x2.cpp
@echo -
icc x2.cpp
```

Figure 7. x2_2.mak

```
/******
 * x3_a.cpp
 *****/

// prototypes -----
void say_hello(void);

// main() -----
void main(void)
{
    say_hello();
}
```

Figure 8. x3_a.cpp

```
/******
 * x3_b.cpp
 *****/
#include <iostream.h>
// functions here -----
void say_hello(void)
{
    cout << "Hello, world.\n";
}
```

Figure 9. x3_b.cpp

makefile is also commonly called a *description file*, because it contains one or more description blocks. A description block's format is shown in Figure 6.

In x2_1.mak you have one target file (x2.exe), one dependent file (x2.cpp), and one command (icc x2.cpp).

Targets must start at the left margin. Indent a command at least one space and begin a comment with a #, either at the left margin or indented. Comments can be on the same line as the targets and dependents, following the last dependent.

In its basic form, the description file tells NMAKE which files depend upon which other files and which commands need to be executed when a file changes. The use of the term *dependent* may be confusing; it does *not* imply that a file depends upon something else. Instead, dependent files are files that the target depends upon to run. (For that reason, dependent files are sometimes called *prerequisite* files.)

In x2_1.mak in Figure 5, x2.exe depends upon x2.cpp. If the dependent x2.cpp changes (and is therefore newer than the target x2.exe), or if the target x2.exe does not exist, then you want to run the commands in the description block to bring the target up to date. (The commands in a description block are executed *only* if a dependent is newer than one of the targets or if one of the targets does not exist.)

After you create and save the x2_1.mak file, you can tell the NMAKE facility to use it by keying nmake /f x2_1.mak on the OS/2 command line. This starts NMAKE, and the /f indicates to use the description file or makefile that follows.

If you have already run do_x2.cmd a few times, you should see the following message when you try to invoke NMAKE:

```
'x2.exe' is up-to-date
```

This happens because after you run do_x2.cmd, the date/time stamp on the x2.exe file is more recent than the date/time stamp on the x2.cpp file. There is no need to compile and link, as indicated by the date/time stamps, so NMAKE does not do it.

However, if you save x2.cpp from its editor, so that it is newer than x2.exe, or if you erase x2.exe, and then you enter nmake /f x2_1.mak on the command line again, NMAKE will create a new x2.exe file.

You may find it useful to add # comments to your makefile. You can also add the @echo command to have the makefile tell

you what it is doing. Figure 7 shows x2_1.mak updated to x2_2.mak. You may want to run it to see its effect.

As the makefile gets larger, you may miss some of the output as it scrolls off the top of the OS/2 window. You can redirect the output and use an editor to browse through it after NMAKE has completed. The following example illustrates redirecting the output to a file called result.

```
nmake /f x2_2.mak > result
```

Look at the result file after NMAKE completes.

The previous example's use of NMAKE is simple. The next example, however, starts to resemble a real-world application, and it shows how NMAKE can help you manage application creation.

Example 3

Each previous example had only one CPP source file that was compiled and linked in one step into an EXE file. Real-world applications tend to have multiple source files. Furthermore, the source files are compiled separately into intermediate object or OBJ files. In a separate step, the various OBJ files are then linked together into the final EXE file.

You will begin this application by creating multiple source files. Then you'll build the application by invoking the compile/link commands in an unintelligent CMD file. Finally, you'll experiment with intelligent makefiles and see the real advantages of NMAKE.

Split the single source file x2.cpp into two source files, x3_a.cpp and x3_b.cpp, as shown in Figures 8 and 9, respectively. One file has the main() function, while the other has the say_hello() function.

Following the earlier examples, you now create an unintelligent CMD file that creates the application. This file, do_x3.cmd, is shown in Figure 10.

Echo tells you what happens while do_x3.cmd runs.

Note the use of the /c option. (You have icc /c x3_a.cpp instead of icc x3_a.cpp.) The /c option means:

Compile the CPP file into the intermediate OBJ file, but stop there, and do not attempt to create an EXE file. (icc x3_a.cpp would fail anyway, since say_hello() would be an unresolved external during the link.) In Figure 10, steps 1 and 2 create updated OBJ files.

In step 3, icc links the two OBJ files produced in the previous steps into the final EXE file. The /fe"x3.exe" option specifies the name to give the EXE file. Without this explicit option, the name of the EXE would be based upon the name of the first OBJ, resulting in x3_a.exe.

Doing It with NMAKE Again

Every time you run do_x3.cmd, it will create new x3_a.obj and x3_b.obj files and will link them to create x3.exe, even if neither of the CPP source files has been updated. This can be a nuisance if it takes a long time to compile a new x3_a.obj file and the underlying source file x3_a.cpp has not changed (which means that there is no need to compile the new OBJ file).

You might be tempted to take advantage of the conditional nature of NMAKE and create a makefile such as x3_1.mak (see Figure 11) following some of the ideas in do_x3.cmd. You create new OBJ files only if the updated source says it is necessary, and you create an EXE file only if there are new OBJ files. Figure 11 suggests a nice linear logical flow.

In Figure 11, block 1 creates the object file x3_a.obj only if it is out of date with respect to the source file x3_a.cpp. Block 2 creates the object file x3_b.obj only if it is out of date with respect to the source file x3_b.cpp. Finally, block 3 looks at the two dependent files x3_a.obj and x3_b.obj; if either of these files has been updated in block 1 or 2, then one or both will have a date/time stamp newer than the target x3.exe, so a new x3.exe will be created in block 3. (Or, if x3.exe does not exist, then block 3 will create it.)

Warning! You are purposely being led astray here (if you haven't already guessed), in order to prove a point. Create the makefile shown in Figure 11. Delete the x3_a.obj, x3_b.obj, and x3.exe files, since you want to give your new makefile a good test. Then invoke NMAKE: nmake /f x3_1.mak

```
REM *** This is do_x3.cmd ***
@ECHO OFF
REM: Step 1
ECHO -----
ECHO - Step 1
ECHO - Compile x3_a.cpp into OBJ file, but do not link.
ECHO -
icc /c x3_a.cpp
REM: Step 2
ECHO -----
ECHO - Step 2
ECHO - Compile x3_b.cpp into OBJ file, but do not link.
ECHO -
icc /c x3_b.cpp
REM: Step 3
ECHO -----
ECHO - Step 3
ECHO - Now create x3.exe by linking x3_a.obj and x3_b.obj.
ECHO -
icc /fe"x3.exe" x3_a.obj x3_b.obj
```

Figure 10. do_x3.cmd

```
#####
# x3_1.mak
# invoke with nmake /f x3_1.mak
#####
# Block 1
x3_a.obj : x3_a.cpp
@echo -----
@echo - Block 1
@echo - compiling x3_a.cpp into object module.
@echo -
icc /c x3_a.cpp
#####
# Block 2
x3_b.obj : x3_b.cpp
@echo -----
@echo - Block 2
@echo - compiling x3_b.cpp into object module.
@echo -
icc /c x3_b.cpp
#####
# Block 3
x3.exe : x3_a.obj x3_b.obj
@echo -----
@echo - Block 3
@echo - linking x3_a.obj and x3_b.obj into executable.
@echo -
icc /fe"x3.exe" x3_a.obj x3_b.obj
```

Figure 11. x3_1.mak

You find that only block 1 will run! Block 2 is never invoked to create or update x3_b.obj, and block 3 is never invoked to create or update x3.exe. What is wrong? The solution is really pretty simple. Let's look at it first, verify that it works, and then find out why.

Doing It with NMAKE (the Right Way)

All you do is take block 3 in x3_1.mak and move it so that it becomes the first

block. The result is x3_2.mak, shown in Figure 12, with slight changes in the comments.

Once again, you delete the x3_a.obj, x3_b.obj, and x3.exe files, and then invoke NMAKE:

```
nmake /f x3_2.mak
```

As NMAKE runs, it first invokes the commands in block 2, then block 3, and then

```
#####
# x3_2.mak
# invoke with nmake /f x3_2.mak
#####
# Block 1
x3.exe : x3_a.obj x3_b.obj
    @echo -----
    @echo - Block 1
    @echo - linking x3_a.obj and x3_b.obj into executable.
    @echo -
        icc /fe"x3.exe" x3_a.obj x3_b.obj
#####
# Block 2
x3_a.obj : x3_a.cpp
    @echo -----
    @echo - Block 2
    @echo - compiling x3_a.cpp into object module.
    @echo -
        icc /c x3_a.cpp
#####
# Block 3
x3_b.obj : x3_b.cpp
    @echo -----
    @echo - Block 3
    @echo - compiling x3_b.cpp into object module.
    @echo -
        icc /c x3_b.cpp
```

Figure 12. x3_2.mak

block 1. First, x3_a.cpp is compiled into x3_a.obj. Then, x3_b.cpp is compiled into x3_b.obj. Finally, the two new OBJ files are linked into the new x3.exe file.

This is what you wanted, but why does it work that way? Why do the description blocks apparently run out of sequence?

Why It's the Right Way

The first description block is a sort of "master" description block. Generally, its target is ultimately what you want to build. The other description blocks are "secondary" in a sense; they are invoked only by the master description block or by other secondary blocks that were themselves invoked by the master description block.

In block 1 in Figure 12, NMAKE first looks to see if the dependent file x3_a.obj is the target in any of the secondary blocks. Yes, it is the target in block 2, so the commands in block 2 are executed. (The commands are executed because x3_a.obj does not exist. The commands would also be executed if x3_a.cpp were newer than x3_a.obj.) This is why block 2 appears to run first.

Next, NMAKE looks at the next dependent in block 1, x3_a.obj, to see if it is the target of any secondary blocks. It is the

target in block 3, so the commands in block 3 are executed, because there is a valid reason to create a new target x3_b.obj.

Finally, back in block 1 again, you now have all the dependents updated (if updating was appropriate). The dependents are compared against the target x3.exe, and if either dependent is newer than the target, or if the target does not exist, then the commands of block 1 are executed. Since x3.exe does not exist, it is created by the commands in block 1.

If you look once again at x3_1.mak in Figure 11, you should now understand its behavior. In block 1, NMAKE looked at the single dependent x3_a.cpp. The dependent was not the target in any of the secondary blocks (blocks 2 and 3), so none of the commands in any of the secondary blocks was run. The x3_a.obj target in block 1 did not exist, however, so the commands in block 1 were executed, resulting in the creation of a new x3_a.obj file. This is why, at best, only block 1 commands are ever executed.

Experiment with NMAKE

You may want to experiment with the new makefile x3_2.mak and its associated files to better understand how NMAKE works. Try these steps in the indicated order:

1. After successfully running the new makefile, if you try to run it again, you get the message 'x3.exe' is up-to-date. Unnecessary compiles and links are not done. The logic is as follows: The block 1 dependents are targets in blocks 2 and 3. However, the date/time stamps in blocks 2 and 3 result in no updates of the block 2 and 3 targets. Therefore, back in block 1, the two dependents are still older than the target, so the block 1 commands are not executed.
2. Erase x3.exe and run the makefile. Only block 1 executes, creating a new x3.exe from the existing OBJ files. The logic is as follows: The block 1 dependents are targets in blocks 2 and 3. The date/time stamps in blocks 2 and 3 result in no updates of the block 2 and 3 targets. However, back in block 1, the target does not exist, so the block 1 commands are executed, and a new x3.exe is created from the existing OBJ files.
3. Erase x3_a.obj and run the makefile. Because the dependents in block 1 are targets in blocks 2 and 3, NMAKE first determines if block 2 or 3 should be executed. Block 2 executes (because x3_a.obj does not exist), creating a new x3_a.obj from the existing x3_a.cpp file. Block 3 does not execute, because its target is up to date. Then block 1 commands execute (because its target is now out of date), creating a new x3.exe from the new x3_a.obj and the old x3_b.obj.
4. Use the editor to save x3_b.cpp to give it a new date/time stamp. You do not have to change the file; just save it. Then run the makefile. Because the dependents in block 1 are targets in blocks 2 and 3, NMAKE first determines if block 2 or 3 should be executed. Block 2 commands do not execute, because the target is up to date. Block 3 executes (because x3_b.cpp is newer than x3_b.obj), creating a new x3_b.obj to overlay the existing x3_b.obj. Then block 1 executes (because the x3_b.obj dependent is now newer than the target), creating a new x3.exe from the old x3_a.obj and the new x3_b.obj.

Tip: NMAKE does not simply run the description blocks in sequence. It treats the first description block as the "master"

block and the remaining description blocks as “secondary” blocks. NMAKE sees if the dependents in the master block are targets in any secondary blocks.

Depending upon the date/time stamps of a secondary block’s target and dependents, the commands in the secondary block do (or do not) run. Therefore, secondary blocks are run only because their targets are dependents in the master block or dependents in a previous secondary block. Only after the master block’s dependents are ensured to be up to date (by running secondary block commands if appropriate) does NMAKE determine whether to run the commands of the master block. After the commands of the master block are run, NMAKE stops.

A Final Change to the Makefile

There is a final change to make to the x3_1.mak makefile in Figure 11. This change does not improve the behavior, but it introduces an idea that will be useful in the last example.

Normally, using the same target in more than one description block causes NMAKE to end. You can change this behavior by using two colons (::) instead of one colon (:) as the target/dependent separator. (This concept is shown in Figures 13 and 14.) You can therefore extend the first or “master” description block to have it encompass more than one block.

Note the similar changes in the x3_2.mak makefile (see Figure 15).

In Figure 15, note that x3.exe is the target in blocks 1 and 2. Using two colons (::) as the target/dependent separator makes this possible. Note also that block 1 does not have any actual commands (unless you count @echo as a command, and the @echo commands can be removed if you want). Block 1 serves only to trigger any secondary blocks having x3_a.obj or x3_b.obj as targets.

Next, block 2 is a sort of continuation of the master description block. If either of the dependent OBJ files in block 2 is newer than the target EXE file, then the block 2 commands will be executed. Note that the block 2 dependents (which are the same as the block 1 dependents) do not trigger running any of the secondary blocks a second time—block 1 has just

```
# The following causes NMAKE to end
X : A
    command
X : B
    command
```

Figure 13. Reusing a Target with (:)

```
# The following works
X :: A
    command
X :: B
    command
```

Figure 14. Reusing a Target with (::)

```
#####
# x3_2.mak
# invoke with nmake /f x3_2.mak
#####
# Block 1
x3.exe :: x3_a.obj x3_b.obj
    @echo -----
    @echo - Block 1
    @echo - OBJ files are now current
    @echo -
#####
# Block 2
x3.exe :: x3_a.obj x3_b.obj
    @echo -----
    @echo - Block 2
    @echo - linking x3_a.obj and x3_b.obj into executable.
    @echo -
        icc /fe"x3.exe" x3_a.obj x3_b.obj
#####
# Block 3
x3_a.obj : x3_a.cpp
    @echo -----
    @echo - Block 3
    @echo - compiling x3_a.cpp into object module.
    @echo -
        icc /c x3_a.cpp
#####
# Block 4
x3_b.obj : x3_b.cpp
    @echo -----
    @echo - Block 4
    @echo - compiling x3_b.cpp into object module.
    @echo -
        icc /c x3_b.cpp
```

Figure 15. x3_2.mak

ensured that the secondary blocks are up to date.

If you start with no EXE or OBJ files and you run the makefile, you will see block 3 run, then 4, then 1, and finally 2, in that order. This slightly different behavior has no particular benefit in this example, but it is useful in the final example where you build a Presentation Manager application.

Tip: You can use a file as the target in more than one description block by using two colons (::) as the target/dependent separator.

Example 4

You will have to coordinate the use of several additional types of files when building VisualAge C++ Presentation

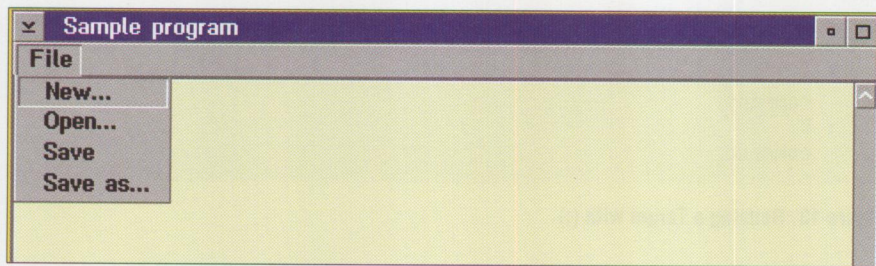


Figure 16. Start of a Simple Editor

```

/*****
 * x4_a.cpp - main() function *
 *****/
#include "x4_b.hpp"
#include "x4.h"
void main()
{
    myWindow appWindow(ID_MAIN);
    IApplication::current().run();
}

```

Figure 17. x4_a.cpp

```

/*****
 * x4_b.cpp - object definition *
 *****/
#include "x4_b.hpp"
#include "x4.h"
// constructor
myWindow :: myWindow(unsigned long Id)
    : IFrameWindow("Sample program",
                  Id,
                  defaultStyle() ]
              menuBar)
    , myMLE(ID_MLE,
            this,
            this)
{
    // Make the MLE control the client area.
    setClient(&myMLE);
    show();
    myMLE.setFocus();
}

```

Figure 18. x4_b.cpp

Manager applications. NMAKE can help you intelligently manage the use of these new files.

The application you will build is the start of a simple editor, as shown in Figure 16. The editor is really simple—nothing happens when you select any of the menu items, except that the menu closes!

The kinds of files you will deal with are:

- **filename.CPP**—These files are the C++ source for your application. You usually have multiple CPP files—the

`main()` function is in one file, and you probably have additional CPP files for the functions and objects you create. In this example, the `main()` function is in the `x4_a.cpp` file, and the window object that appears on the screen is an object of the `myWindow` class defined in the `x4_b.cpp` file.

- **filename.HPP**—These files are the header files for the classes you create. The header file `x4_b.hpp` declares the `myWindow` class that is defined in the `x4_b.cpp` file.

- **filename.RC**—This is the application resource script file. It can contain information about such things as text strings and bitmaps that are to appear in your application. The resource script file also provides one way (there are others) to define menus for your application. The resource script file `x4.rc` defines the menu that you see in Figure 16.

- **filename.H**—This is a header file that typically defines constants used in other application files. The header file `x4.h` defines resource IDs for the menu items, window, and the multi-line edit shown in Figure 16.

The multi-line edit, by the way, is just the big empty area that occupies most of the space within the window. You can type words into the multi-line edit, as with any common editor; however, as suggested here, the menu entries do not allow you to save what you type or to open already existing files.

A New Type of Intermediate File

Intermediate files are created during the application building process. You have already seen the OBJ files. Recall that they are created in a compile step; then in a later step, one or more OBJ files are linked together to create an EXE file.

A new intermediate file in this example is the binary resource file RES. A RES file is created from an RC file using the resource compiler. The compiled RES file contains information about (for example) text strings, bitmaps, or menus that your application uses. Other information may be contained in the RES file as well.

The information in the RES file must be added or bound to the EXE file. The RES file bind is done by the resource compiler (which, as you may recall, created the RES file in the first place). Without this bind step, the EXE file will not run, since it tries to use resources (text strings, for example) that are not properly defined to it.

Therefore, the steps for building the application from the beginning are:

1. Compile the CPP files into OBJ files.
2. Compile the RC file into the RES file.
3. Link the OBJ files to create the EXE file.
4. Bind the RES file to the EXE file.

The Challenge to NMAKE

Some of these steps might take a long time if you are building a large application, so you don't want to do them if they are not necessary. For example, if there are multiple CPP files and only one of them changes, you want to compile only one new OBJ file and link it with the other existing OBJ files to create the new EXE file. As another example, assume that only the RC file has changed. Therefore, you only want to create a new RES file and bind it to the existing EXE file to create the new EXE file. NMAKE and a well designed makefile can help you do this.

The Source Files

The source files that make up this example are:

- x4_a.cpp—Source for main() function (Figure 17).
- x4_b.cpp—Definition of myWindow object (Figure 18).
- x4_b.hpp—Header file or declaration of myWindow object (Figure 19).
- x4.rc—Resource script file (Figure 20).
- x4.h—Header file defining various constants (Figure 21).

The Makefile

Figure 22 shows the makefile. Block 1 serves only to cause NMAKE to examine blocks 4, 5, and 6 to see if the OBJ or RES files need to be brought up to date. Examine block 4 and the source for x4_a.cpp in Figure 17 to make sure you understand why the target x4_a.obj has the x4_a.cpp, x4_b.hpp, and x4.h files as its dependents.

Recall that the /c option says to compile the CPP file into the intermediate OBJ file, but not to link. The /gd option in Figure 22 is new. It says that the final EXE file will dynamically link to VisualAge C++ runtime DLLs. Without this option, various VisualAge functions would be copied into your EXE, thereby expanding it.

Note: The compile steps in blocks 4 and 5 may cause VisualAge C++ to complain about a missing /gm option (the /gm option says to link with multithread versions of the libraries), depending upon your maintenance level. You can add the /gm option as shown here.

For block 4: `icc /c /gd /gm x4_a.cpp`

```
/* x4_b.hpp - object declaration */
#include <iframe.hpp> // for IFrameWindow
#include <imle.hpp> // for IMultiLineEdit
class myWindow : public IFrameWindow
{
public:
    myWindow(unsigned long id); // constructor
private:
    IMultiLineEdit myMLE;
};
```

Figure 19. x4_b.hpp

```
/* x4.rc - resource script file */
#include "x4.h"
MENU ID_MAIN
{
    SUBMENU "File", ID_FILE
    {
        MENUITEM "New...", ID_NEW
        MENUITEM "Open...", ID_OPEN
        MENUITEM "Save", ID_SAVE
        MENUITEM "Save as...", ID_SAVEAS
    }
}
```

Figure 20. x4.rc

For block 5: `icc /c /gd /gm x4_b.cpp`

Examine block 5 and the source for x4_b.cpp in Figure 18 to make sure you understand the target and dependents. Next, examine block 6 and the source for the RC file x4.rc in Figure 20. Notice that the RES file depends upon the RC file x4.rc. Through the RC file, the RES file also depends upon the x4.h file.

In block 6, the rc (resource compiler) command is run. The /r option says to create only a RES file, and do not attempt to bind it to an EXE file. By default, the RES file will take its name from the RC file and become x4.res.

After block 1 has possibly triggered running the commands in blocks 4, 5, and 6, block 2 comes into play. Recall that the two colons (::) enable the same target x4.exe to be used in multiple blocks, effectively extending the "master" description block to multiple blocks. In block 2, if the target does not exist, or if either of the dependents is newer than the target,

```
/* x4.h - header file */
#define ID_MAIN 2
#define ID_FILE 1000
#define ID_NEW 1001
#define ID_OPEN 1002
#define ID_SAVE 1004
#define ID_SAVEAS 1004
#define ID_MLE 2000
```

Figure 21. x4.h

then the block 2 commands are run. There are two commands in the block (other than the @echo commands). The first command links the OBJ files into a new EXE, while the second invokes the resource compiler to bind the RES file to the EXE file.

The new /B"/pmtyp/pm" option in the icc command tells the linking process that the resulting application must run in the PM environment. The syntax of the rc command is slightly different in block 2 than in block 6. In block 2, the syntax says to bind the existing RES file

```
#####
# x4_1.mak
# invoke with nmake /f x4_1.mak
#####
# Block 1
x4.exe :: x4_a.obj x4_b.obj x4.res
@echo -----
@echo - Block 1
@echo - OBJ and/or RES files are now current
@echo -
#####
# Block 2
x4.exe :: x4_a.obj x4_b.obj
@echo -----
@echo - Block 2
@echo - linking OBJ files into EXE
@echo -
icc /B"/pmtyp:pm" /FE"x4.exe" x4_a.obj x4_b.obj
@echo -
@echo - binding RES to EXE
@echo -
rc x4.res x4.exe
#####
# Block 3
x4.exe :: x4.res
@echo -----
@echo - Block 3
@echo - binding RES to EXE
@echo -
rc x4.res x4.exe
#####
# Block 4
x4_a.obj : x4_a.cpp x4_b.hpp x4.h
@echo -----
@echo - Block 4
@echo - compiling x4_A.CPP into x4_A.OBJ ...
@echo -
icc /c /gd x4_a.cpp
#####
# Block 5
x4_b.obj : x4_b.cpp x4_b.hpp x4.h
@echo -----
@echo - Block 5
@echo - compiling x4_B.CPP into x4_B.OBJ ...
@echo -
icc /c /gd x4_b.cpp
#####
# Block 6
x4.res : x4.rc x4.h
@echo -----
@echo - Block 6
@echo - compiling RC file into RES file ...
@echo -
rc /r x4.rc
```

Figure 22. x4_1.mak

(which was created earlier by block 6) to the EXE file.

After the block 2 commands run, you have an application. After block 2, block 3 comes into play because of the two colons (::). What is the point of block 3? Why would you want to do a resource bind again if it was already done in block 2? The idea is this: If block 2 commands run,

then you get an EXE that is newer than the RES, because the EXE was made out of the RES. Therefore, because of the target and dependent date/time stamps in block 3, the block 3 commands do not run.

Suppose, however, that you have already run the makefile to get a running application. Then you change only the RC file. If you run the makefile again, block 1

should trigger the commands in just block 6, creating a new RES file.

Then, because of the date/time stamps, block 2 commands do not run, but the block 3 commands do run. You will have bound a new RES file to an existing EXE file, which is exactly what you wanted to do.

Run the makefile the usual way:

```
nmake /f x4_1.mak
```

Since a lot of material will scroll off the top of your screen, you will probably want to redirect the output, then look at it later to see what happened. In the next command, you redirect the output to the result file, which you can browse later using an editor.

```
nmake /f x4_1.mak > result
```

Starting with no OBJ, RES, or EXE files, you see the commands of block 4 run first, followed by blocks 5, 6, 1, and 2. Block 3 commands do not run, as explained earlier.

Experiment with NMAKE

Now, let's try some experiments:

1. Use an editor to save x4_a.cpp to give it a new date/time stamp, then run the makefile. Commands in blocks 4, 1, and 2 run in that order. Only a single new OBJ file needs to be created, and no new RES file needs to be created. A new EXE is created using the new OBJ and the old OBJ and RES files.
2. Use an editor to save x4.rc to give it a new date/time stamp, then run the makefile. Commands in blocks 6, 1, and 3 run in that order. Only a new RES file needs to be created, and no new OBJ files need to be created. The old EXE is bound to the new RES, resulting in an updated application.
3. Use an editor to save x4.h to give it a new date/time stamp, then run the makefile. Notice that x4.h is a dependent in blocks 4, 5, and 6 (and make sure that you recall why). Both the OBJ files and the RES file depend upon x4.h, so you would expect commands in blocks 4, 5, 6, 1, and 2 to run in that order. This is in fact what happens, but the commands in block 3 also run. The RES file is unnecessarily bound to the EXE file a second time. Hmmm—things are no longer perfect!

The Final Makefile

NMAKE does a lot for you—automating the process of building an application out of many pieces and running only the necessary steps (almost) after files are updated. You can accept the quirk you just found or, being a purist, you can try to make things work a bit better.

Apparently, the date/time relationship of the target and the dependent are not handled as you would like in block 3. Change block 3 as shown in Figure 23, and save it as the new makefile `x4_2.mak`. Give NMAKE a second chance by having `x4_2.mak` call a second makefile from within `x4_2.mak`. This second makefile is shown in Figure 24, where `sup` suggests “supplemental.”

When you run the last scenario (an updated `x4.h` file), things behave as you would like. Block 3 of `x4_2.mak` invokes `x4_sup.mak`, where `x4.exe` is found to be newer than `x4.res`, and therefore the unnecessary resource bind is not done. You will get an ‘`x4.exe`’ is up-to-date message from the target/dependent line of `x4_sup.mak`.

All the other scenarios described for this example work properly as well. In particular, try the case where only `x4.rc` has been updated. There may be another, more elegant solution to this problem—perhaps you can find it!

```
#####  
# Block 3  
x4.exe :: x4.res  
@echo -----  
@echo - Block 3  
@echo - test to see if bind of RES to EXE is needed  
@echo -  
nmake /f x4_sup.mak
```

Figure 23. `x4_2.mak`, Block 3 Only

```
#####  
# x4_sup.mak  
# invoked from x4_2.mak  
#####  
# Block 1  
x4.exe : x4.res  
@echo -----  
@echo - Block 1 (in X4_SUP.MAK)  
@echo - binding RES to EXE  
@echo -  
rc x4.res x4.exe
```

Figure 24. `x4_sup.mak`



Mark Fisher is a senior market support representative in IBM Advanced Decision Support Services in Roanoke, Texas. He provides decision support and data mining services. Mark joined IBM in 1977. He has a BS degree in Physics, an MS in Aeronautical Engineering, and an MBA, all from Stanford University. His Internet ID is `mark_fisher@vnet.ibm.com`.

Building Object-Oriented Applications from Existing C Code

This article discusses how the Visual Builder component of VisualAge C++ can extend the usefulness of legacy C code and existing C skills and provide an elegant migration path from C and procedural programming to C++ and OO development. This process is illustrated step-by-step.

Through computer interaction, users today are enjoying the best of times. This is the age of sophisticated interface controls, such as folders, notebooks, and toolbars. Multimedia and graphic-intensive software are becoming standard, making technology easy and intuitive for computer users.

Unfortunately, while software usability improves, software developers have to pay the price for these impressive graphical user interfaces (GUIs). Requirements are more complex, new concepts have to be learned quickly, and competitive markets are forcing reduced time frames.

Gregory Piamonte
IBM Corporation
Roanoke, Texas

Fortunately, help has arrived in the form of two emerging technologies, *visual programming* and *software construction from parts*. The latter, based mainly upon techniques proven in other industries such as manufacturing, involves building applications from existing, reusable software components called *parts*. Just as a computer is built from prefabricated components, such as the motherboard, power supply, and disk drives, an application can also be assembled from existing software parts. Figure 1 illustrates this construction-from-parts paradigm.

Visual programming hides some of the more complicated concepts involved with graphical user interfaces, such as event-driven programming, plus it greatly facilitates the mundane tasks of drawing dialogs and their graphical controls.

These two approaches allow software builders to concentrate on the mission-critical aspects of applications, such as business logic, communications, and data access.



VisualAge C++ to the Rescue!

Does a product exist that effectively exploits the powerful combination of visual programming and the construction-from-parts methodology?

Allow me to present VisualAge C++, IBM's newest member of its C Set family of products (see Figure 2). The VisualAge C++ package includes enhanced versions of components from previous C Set and C Set ++ products, such as WorkFrame, Performance Analyzer, and User Interface Class Library (now called IBM Open Class Library).

VisualAge C++ also introduces the Visual Builder component, which builds upon the proven application construction technologies of IBM's VisualAge Smalltalk product. This component allows you to visually develop complete object-oriented applications by providing the tools for building, reusing, and assembling all necessary parts. Standard parts, customized parts, or parts built from scratch can be pieced together to build higher-function parts, subsystems, or entire applications.

The Visual Builder is shipped with its own set of prefabricated parts built from the user interface, application support, data access, and collection classes of the Open Class Library. These classes and parts have been developed for maximum reusability and are the main building blocks for our construction-from-parts approach.

The Visual Builder does not restrict the types of existing parts or usable classes. You can reuse any class library supported by the target platform. You can even bring in procedural code such as C functions to use with Visual Builder parts (see Figure 3), allowing a gradual and elegant transition into the world of objects.

In this article, I attempt to illustrate how software investments can be maximized by recycling procedural functions and leveraging existing C skills. The recommended scenario is to initially exploit Visual Builder's screen-painting and object-oriented code generation facilities for your graphical user interface, but then to call existing procedural code for the application logic.

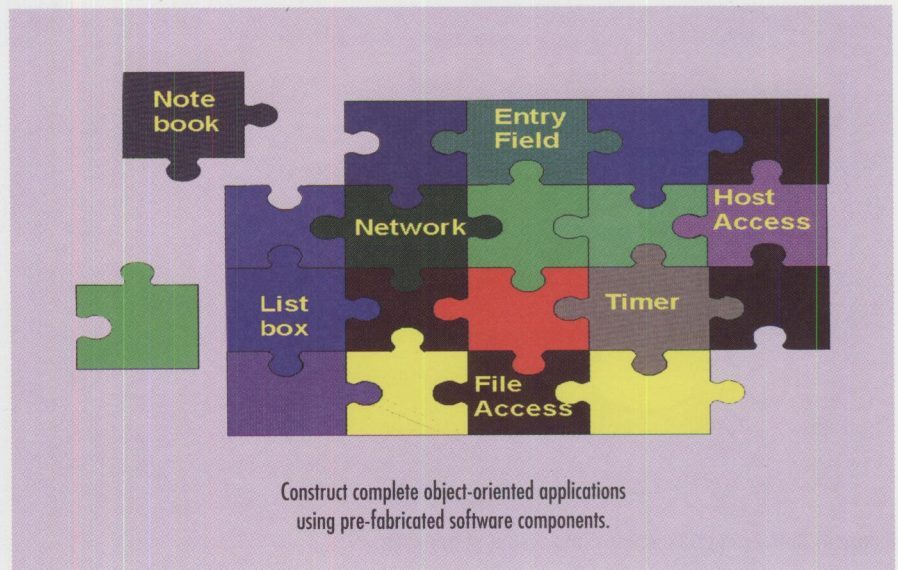


Figure 1. Construction-from-Parts Paradigm

What is VisualAge C++?

Newest member of IBM's C SET++ family of products

Consists of enhanced versions of:

- WorkFrame/2
- Performance Analyzer
- Interactive Debugger
- Class Libraries
- Compiler

Introduces the following new components:

- Visual Builder
- Data Access Builder
- Browser
- Project Smarts
- Build Smarts
- Linker (32-bit)
- Editor

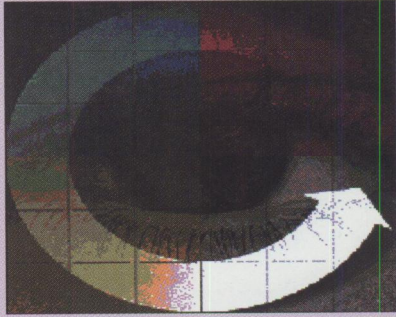


Figure 2. VisualAge C++

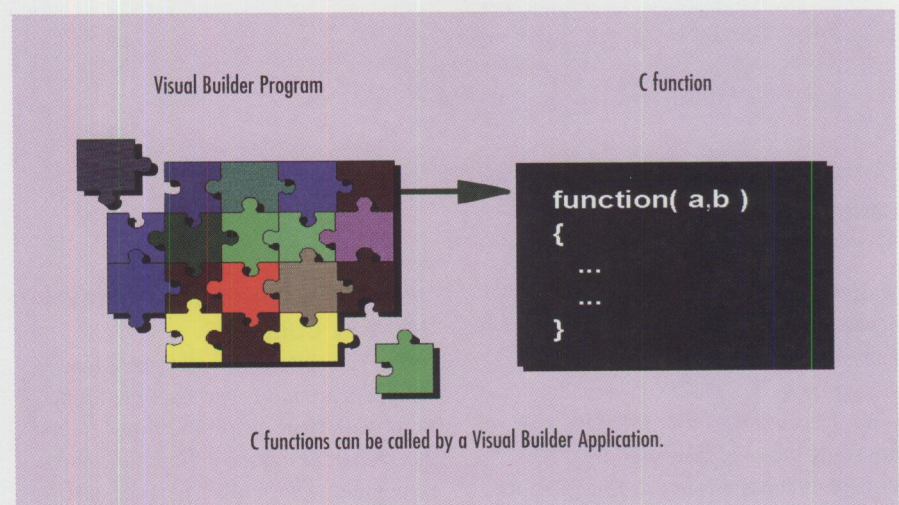


Figure 3. Procedural Code with Object-Oriented Parts

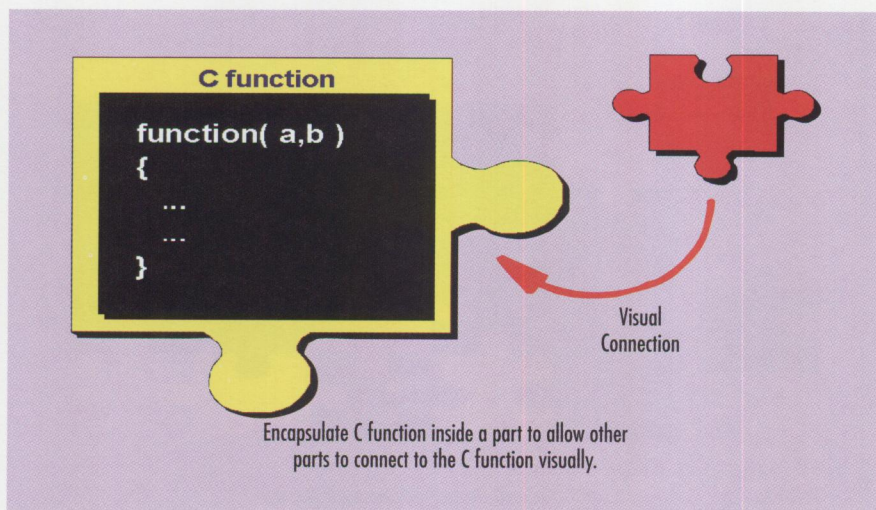


Figure 4. Converting C Functions into Visual Builder Parts

```

/*-----*/
/* CFuncs.c Source file for a C DLL */
/*-----*/
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "CFuncs.h"
/*****/
/* Function Name: convertFahrenToCelsius */
/* Description: Convert Fahrenheit temperature to Celsius */
/* Parameters */
/* */
/*****/
int _Optlink convertFahrenToCelsius( int fahrenheit )
{
    int celsius;
    celsius = ((fahrenheit - 32) * 5) / 9;
    return( celsius );
}
/*****/
/* Function Name: convertCelsiusToFahren */
/* Description: Convert Celsius temperature to Fahrenheit */
/* Parameters */
/* */
/*****/
int _Optlink convertCelsiusToFahren( int celsius )
{
    int fahrenheit;
    fahrenheit = ((celsius * 9) / 5) + 32;
    return( fahrenheit );
}

```

Figure 5. CFUNCS.C

With this strategy, you can exploit the benefits of object-oriented technology immediately, then learn how to use it at your own pace. Once you've built your object-oriented user interface, you can study the code generated by the Visual Builder and learn, as time permits, how C++ and objects are used in the application.

Visual Builder Fundamentals and Key Concepts

In the Visual Builder world, parts are wrappers around C++ classes or C functions, with their own well-defined interface. A *part interface* defines how parts communicate with other parts. A part interface comprises three features: attributes, actions, and events. For C parts,

only *actions* are significant. They are defined as the services or operations that a part can perform. Actions correspond to C++ member functions or C functions.

Attributes are the logical data that map to C++ class member data.

Events are signals that a part sends to interested parts whenever a change has occurred or a specific condition has been met. Examples of these changes or conditions include the contents of an entry field being altered and the user clicking on a pushbutton.

Another key concept involves *visual connections* between parts. These connections are basically the graphical lines of code that implement the logic for the application, and their order dictates the processing flow. A connection is directional, which means it has a source and a target. The direction is determined by the manner in which the connection is drawn. (This will make more sense when we actually start building the sample application.)

The Visual Builder generates C++ code to implement the visual and non-visual parts that have been designed and defined, as well as the connections between these parts. Visual Builder part source code is kept separate from user-defined code to prevent loss of user code if modifications are made. To provide a jump-start, you can generate a code skeleton with stubs for C++ member functions.

All these facilities let you focus on what the application does, rather than how it looks.

Visual Builder Editors

The Visual Builder comes with three editors:

- *Composition Editor*, where you design your graphical interface by selecting and laying out prefabricated visual parts on a work space. You add the non-visual parts that provide the application logic, then make the appropriate connections between the parts to define the flow of processing.
- *Part Interface Editor*, where you define the features of your parts, specifically their actions, attributes, and events. This editor is normally not applicable for C function parts but is crucial for C++ parts.

■ *Class Editor*, where you customize the code generation settings. Here, you provide the specifics of the part to be generated, such as source filenames and file dependencies.

Sample Program Implementation Strategy

This article's sample application will convert temperature units from Fahrenheit to Celsius and vice versa.

There are two techniques for integrating C functions with Visual Builder applications. The first involves calling C functions directly by writing C code using the *custom logic* facility. The second involves wrapping C functions into Visual Builder parts, allowing other parts to visually call these functions through graphical connections (see Figure 4). Because the second solution offers more flexibility and is in line with object-oriented principles, it is the approach we use here.

The sample application uses pre-built C functions packaged in a dynamic link library (DLL) named `CFUNCS.DLL`. We will assume that these functions are our legacy C code. For simplicity, we'll keep the implementation of the C functions primitive by limiting the temperature range to between freezing and boiling points. The source code for the two C functions to be used is shown in Figure 5.

Creating Our WorkFrame Project

To jump-start the initial project creation, we use another new VisualAge C++ component called *Project Smarts*. This component helps us create a development project specifically tailored for Visual Builder applications. The generated project will already have the build, compile, and link options properly customized.

To generate our project using Project Smarts:

1. Open "Project Smarts" in the VisualAge C++ folder to display the Project Smarts catalog, shown in Figure 6.
2. Select "Visual Builder Application" from the list of available projects, then click on the Create pushbutton.
3. During the installation, a Location window appears. Input the directory where the source files will be installed, then

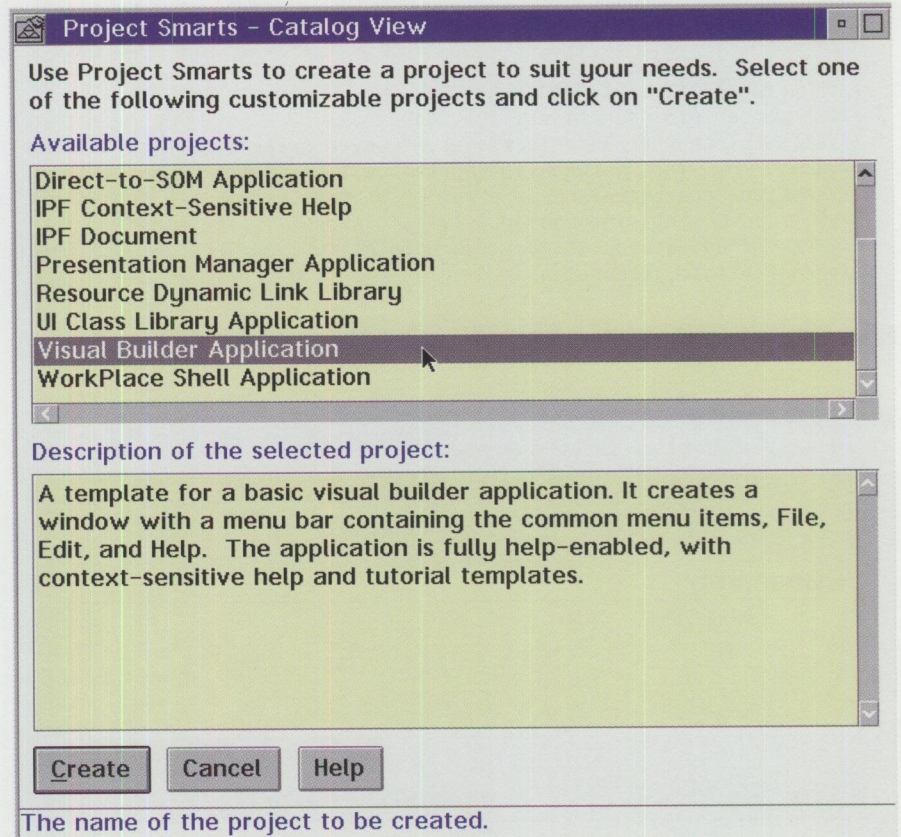


Figure 6. VisualAge C++ Project Smarts Catalog

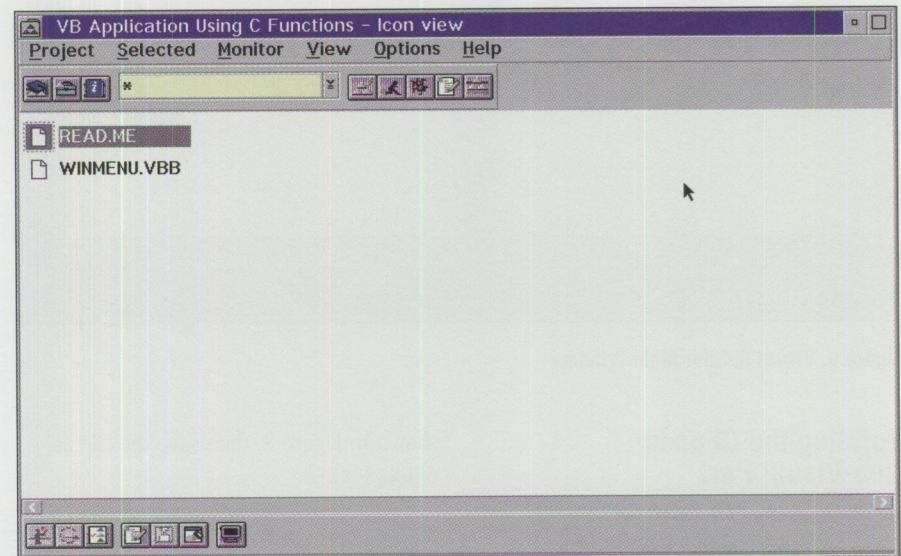


Figure 7. Project Window

4. When the installation is complete, open the project in the specified folder. The Project window will appear, similar to the one in Figure 7.

5. Because we are not using any of the generated files, delete all files within the project by highlighting the files and selecting "Delete . . ." from the context or pop-up menu.

```

//VBBeginPartInfo: CFunctions, "C functions"
//VBIncludes: "cfuncs.h" cfuncs_h
//VBPartDataFile: 'cfuncs.vbb'
//VBComposerInfo: functions, 701, dde4vr30
//VBAction: convertFahrenToCelsius, "Converts Fahrenheit to Celsius",
//VB:      int, int convertFahrenToCelsius( int fahrenheit )
//VBAction: convertCelsiusToFahren, "Converts Celsius to Fahrenheit",
//VB:      int, int convertCelsiusToFahren( int celsius )
//VBPreferredFeatures: convertFahrenToCelsius, convertCelsiusToFahren
//VPEndPartInfo: CFunctions

```

Figure 8. Part Information File for C Functions

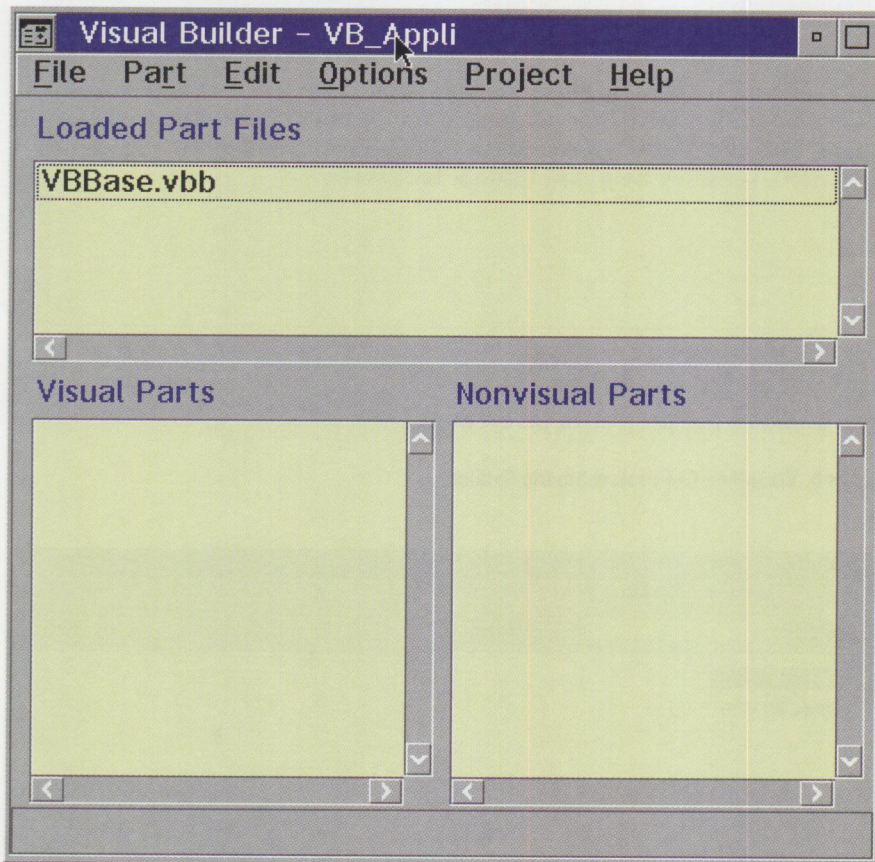


Figure 9. Visual Builder Main Window

Building the CFuncs Non-Visual Part

To wrap C functions as Visual Builder parts, we must create a part information file to define how other parts can communicate or interface with this part. At first glance, a part information file may seem cryptic, but it is not complicated and only requires familiarity with the different statement tags.

Part information files normally have a file extension of .VBE. To create our own file, select "Project->Edit" to display the VisualAge C++ Editor. Type in the statements

shown in Figure 8, then save the file as CFUNCS.VBE.

By importing this file into our sample Visual Builder application, we define the different features of our C part, such as the part name, the include files, the function names as they will be used by other parts, the C function declarations, and the return types.

For more information about the tags and syntax of part information files, refer to the *Visual Builder User's Guide* and the *Building VisualAge C++ Parts for Fun and Profit* documentation. (These

are shipped with the product as online documentation.)

After creating the CFUNCS.VBE file with the statements outlined in Figure 8, close the editor and switch to the Project window.

Importing the CFuncs Non-Visual Part

To import the CFuncs non-visual part:

1. From the Project window menu bar, select "Project->Visual" to display the Visual Builder main window, shown in Figure 9.
2. From the main Visual Builder window menu, select "File->Import part information . . ." to display a file dialog.
3. Select CFUNCS.VBE from the list, then click on the OK pushbutton. If the part information file or VBE was created properly, a confirmation dialog should appear, stating that there are no errors or warnings.
4. Close the confirmation dialog. You should see CFUNCS.VBB in the list of loaded part files. Our non-visual part is now ready to use.

Building the Main Window Visual Part

To build the main window visual part:

1. From the main Visual Builder window menu, select "Part->New" to display the Part-New dialog.
2. Input Sample as the class name. By default, "Visual part" should be the selected part type. The file name will default to the class name. Click on the Open pushbutton.
3. Because we are constructing a visual part, the Composition Editor window will initially appear. You should see only an empty frame window on top of

a white space called the *free-form surface area*. We need to lay out the necessary visual parts on that free-form surface area and build the empty window to look like Figure 10.

This is a good point to become better acquainted with different facilities Composition Editor provides. The *parts palette* on the left side of the Composition Editor has two columns of icons. Icons in the left column represent different part categories. Icons in the right column represent parts within the category selected in the left column. These are pre-fabricated software components that you can use to build your application. The parts palette typically contains the most frequently used visual and non-visual parts. You can customize it to define new categories and include your own parts.

To add parts to the frame window or the free-form surface area, select the appropriate part from the parts palette. At this point, the mouse pointer is loaded and will turn into a crosshair when moved across an area or part where it can be placed. The information area at the bottom of the Composition Editor will indicate the category or part currently selected and will also provide other valuable information when we begin working with connections.

To help you properly position, align, and size the parts being used, tools are provided on the toolbar below the menu bar. There are also special *canvasses*, or containers of parts, that help make life easier. Refer to the online documentation suggested earlier for more detail. For our simple sample, you should have sufficient information to build the window shown in Figure 10.

Our sample application uses only static text controls, numeric spinbuttons, pushbuttons, and entry fields. We will be able to use default settings for all controls except numeric spinbuttons. To alter the settings for the Fahrenheit spinbutton, display its settings notebook by double-clicking on the control or selecting "Open Settings" from its context menu. (Access the context menu through the right mouse button.) Change the lower limit of the numeric range to 32 and the upper limit to 212, then click on the OK pushbutton.

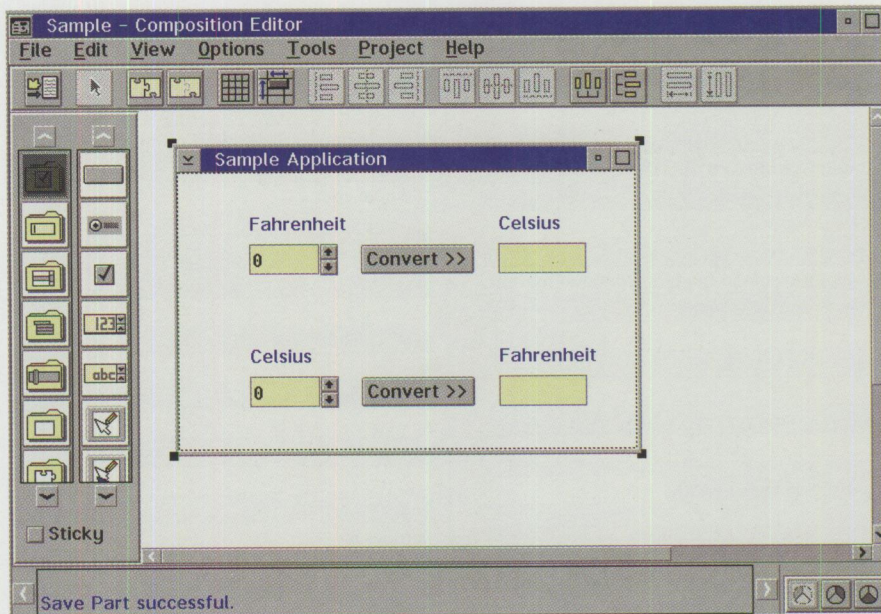


Figure 10. Composition Editor Window

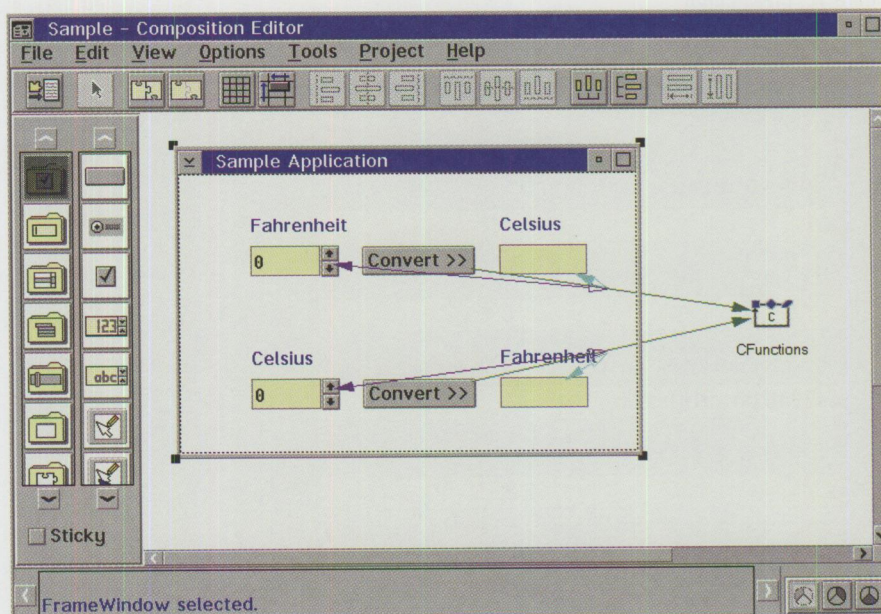


Figure 11. Application Parts with Connections

Now, do the same for the Celsius spinbutton, except keep the lower limit at 0 and change the upper limit to 100. This restricts temperature values to those between freezing and boiling points.

To modify the title on the main window, access its settings notebook by double-clicking on the title bar. At this point, your window should look like Figure 10.

Providing Application Logic Through Connections

Now we need to furnish the logic for the

temperature conversions. For this, we will use C functions contained in CFUNCS.DLL through the CFuncs non-visual part we have defined.

1. From the Composition Editor window menu, select "Options->Add part . . ." to display the Add Part dialog.
2. Input CFunctions* as the part class and CFunctions as the name. Verify that the Part radio button is selected, then click on the Add pushbutton.
3. Place the crosshair anywhere on the free-form surface area, and click on the

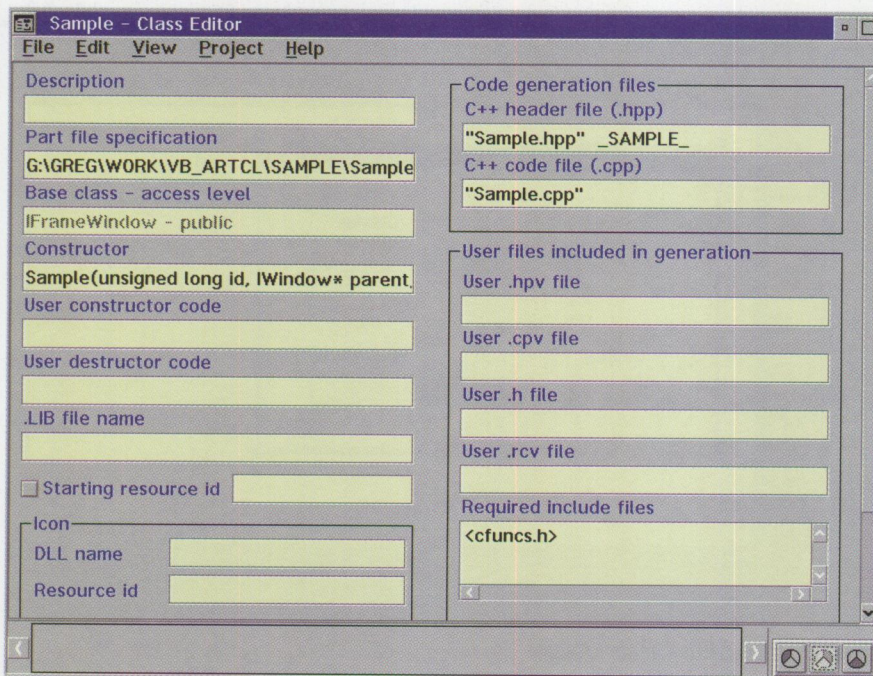


Figure 12. Class Editor

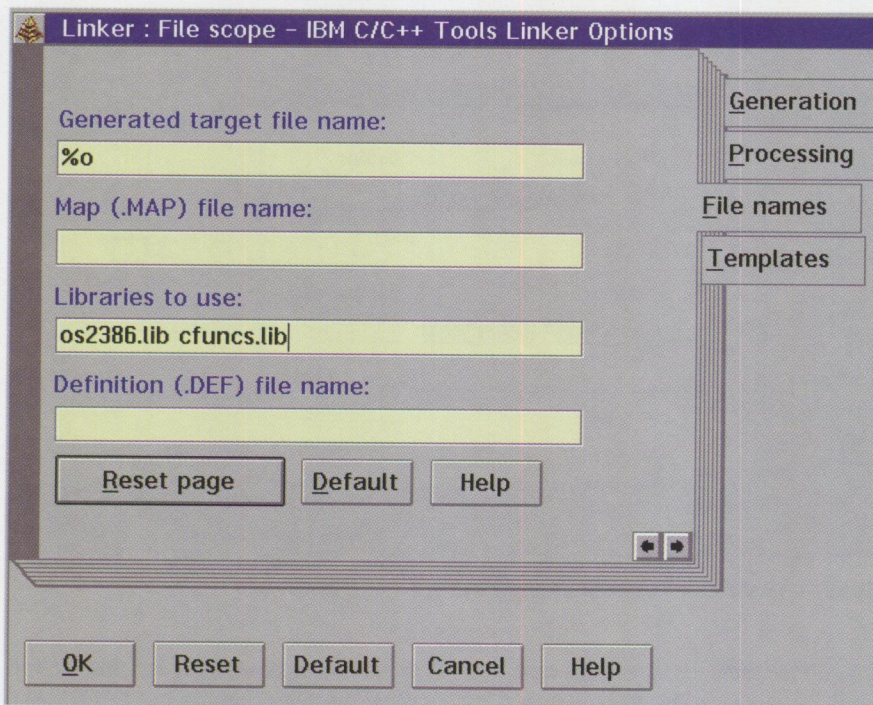


Figure 13. Linker Settings Notebook

4. Access the upper pushbutton's context menu with the right mouse button, then select "Connect->buttonClickEvent." You see a black dotted line with a spider at the end of it.
5. Drop the spider on the CFunctions non-visual part to complete the connection. A pop-up menu of all preferred features displays. In our scenario, this will be a list of C functions that can be invoked by the connection.
6. Select "convertFahrenToCelsius" from the pop-up list. A dotted green line appears between the pushbutton

(source) and the CFunctions part (target). The connection line is dotted to indicate that a parameter is required. In this case, the Fahrenheit value to be converted must be supplied to the C function.

7. Provide the parameter by accessing the context menu of the green connection line, then selecting "Connect->fahrenheit." The spider appears again.
8. Drop the spider onto the Fahrenheit spinbutton and connect to the "value" attribute, completing the parameter connection.
9. Retrieve the result of the function call and initialize the Celsius entry field with this value. Do this by connecting the "actionResult" attribute (from the green connection's list of preferred features) to the "valueAsInt" attribute of the Celsius entry field. This enables the entry field to receive the return value of the C function.

The colors used for the connections vary to indicate the type of connection and can be a valuable clue to whether the logic is being constructed properly. Now repeat steps 4 through 9 for the Celsius to Fahrenheit controls, except call the C function `convertCelsiusToFahren` instead.

I assume you have been saving your work after every major change; if not, do so now. At this point, the Composition Editor should appear similar to the screen shown in Figure 11.

One remaining step provides you with a chance to work with the Class Editor. From the Composition Editor menu, select "View->Class Editor" to display the Class Editor. Look for the multi-line edit field labeled "Required include files" at the bottom right of the window. Type `cfuncs.h` exactly as shown in Figure 12.

Our application is now complete. Select the "Save and Generate->Part source" menu option, then the "Save and Generate->main() for Part" option. Close the Class Editor and the Visual Builder main dialog, and return to the Project window.

Setting the Build Options and Generating the Executable

We are not yet ready to build the

NOW AVAILABLE:

The Best of /AIXtra

VOLUMES I & II

Winner
of STC
Technical
Merit
Award!

The Best of **AIXtra**
An Eclectic UNIX Anthology



Networking and Communications
X.25, TCP/IP, and Internet

Data Storage and Management
RDBMS, Client/Server, and RAID

The PowerPC Architecture

Performance Monitoring and Tuning
Theory and Application

IBM Microkernel Technology

UNIX Batch Scheduling

Edited by Alan E. Hodel

The Best of **AIXtra**
An Eclectic UNIX Anthology



Asymmetric Multi-Processing

ATM Networking

Parallel Database Technology

Distributed Computing

Surf the World-Wide Web
with Mosaic

Edited by Alan E. Hodel

These books are a collection of some of the most popular technical articles from */AIXtra: IBM's Magazine For AIX Professionals*. The books run 300 pages each and are divided into sections that include technical discussions about AIX performance tuning, the Distributed Computing Environment (DCE), networking and communications, relational database management systems, future technological directions, a complete index, and more.

Published in a joint effort with Prentice Hall PTR, the volumes are priced at \$39.00 each and feature

introductions by Donna Van Fleet, vice president of IBM AIX Systems Development, and Irving Wladawsky-Berger, former general manager of the IBM RISC System/6000 Division.

For more information or to order, please contact: BOOKS at the IBM BookStore at (800) IBM-TEACH or (520) 574-4500; FAX: (800) 426-9006 or (520) 574-4501; WWW URL: <http://www.training.ibm.com/cgi-bin/edubin/edubin/8/>. The publication numbers are SR28-5890-00, Vol. I; SR28-5911-00, Vol. II. IBMers may use PUBORDER to order.

executable file for our sample application. We still need to specify the import library of the DLL containing the C functions and the location of this import library and include file. To do this:

1. From the Project window menu, select "Options->Link" to display the Linker Settings notebook.
2. Switch to the Filenames page, and append `cfuns.lib` to the "Libraries to use:" entry field. Do not remove the original contents of the field. (Refer to Figure 13.)
3. Click on the OK pushbutton to accept the changes. It is not necessary to change the default settings for the compilers. The proper options are already set because we used Project Smarts to generate our project.
4. Select "View->Tools setup" from the Project window menu to bring up the window shown in Figure 14.
5. From the menu bar, select "Variables->Add . . .," and the Add Environment Variable dialog appears. Type `lib` into the Name entry field or select it from the drop-down list.
6. Append the path where `CFUNCS.LIB` is stored to the end of the text in the String entry field, and then click on the Add pushbutton.
7. Repeat steps 5 and 6, except specify `include` in the name entry field, and append the path for `CFUNCS.H` to the text in the string entry field.
8. Close the Tools Setup window and return to the Project window. Select the

"Project->Build" menu option to build the .EXE file for our sample application.

Running the Sample Application

That's it! We're done. If everything went smoothly, we should have produced an executable called `VBMAIN.EXE` that works as designed. `VBMAIN.EXE` is the default name, but you can modify it in the Project Settings notebook.

Before you take the sample application for a test drive, make sure that `CFUNCS.DLL` is either in a directory defined in the `LIBPATH` statement of your `CONFIG.SYS` file or in the same subdirectory as the executable. To run the sample from the Project window, select "Project->Run" from the menu or double-click on the `VBMAIN.EXE` file.

Easing the Way With Visual Builder

VisualAge C++ and all of its components, specifically the Visual Builder, were designed to ease the pain and speed up the effort involved in constructing software applications. The Visual Builder promotes reusability not only with logic parts, but with visual parts as well, and is a tool that maximizes rather than limits. It is probably the quickest route to the land of objects, and it enables you to test object-oriented waters without getting wet.

The source code and Visual Builder files used for this sample application can be found on the World-Wide Web at <http://pscc.dfw.ibm.com/psmag/>.

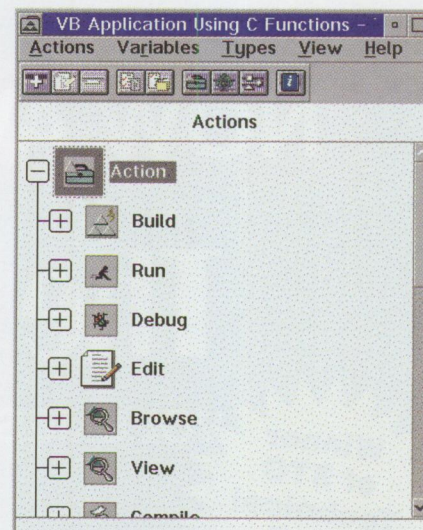
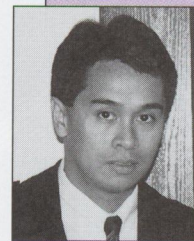


Figure 14. Tools Setup Window



Gregory Piamonte is an OS/2 application development specialist and a member of the Advanced Object Technologies team in the IBM Personal Systems Competency Center in

Roanoke, Texas. Gregory joined IBM in 1994. He has a BS degree in Architecture from the University of Santo Tomas in the Philippines. Gregory's Internet ID is piamonte@vnet.ibm.com.

Article Reprints from

Personal Systems

High-quality article reprints can help your company in many ways. Reprints can increase EXPOSURE for your product or service. They are credible information that consumers TRUST. Reprints make great SALES tools for trade shows, mailings, or media kits.

Call (717) 560-2001 Today!

**REPRINT
MANAGEMENT
SERVICES™**

Duane Dagen
147 W. Airport Road
P.O. Box 5363
Lancaster, PA 17606-5363

SOM Collection Classes: A Primer for the VisualAge COBOL Programmer

The System Object Model (SOM) provides a rich set of collection classes for SOM-enabled applications to use. SOM's collection classes come in many forms, including linked lists, sets, ordered collections, queues, and dictionaries. While samples and documentation exist for using these collection classes in C++, little is currently available to guide the programmer who uses object-oriented COBOL.

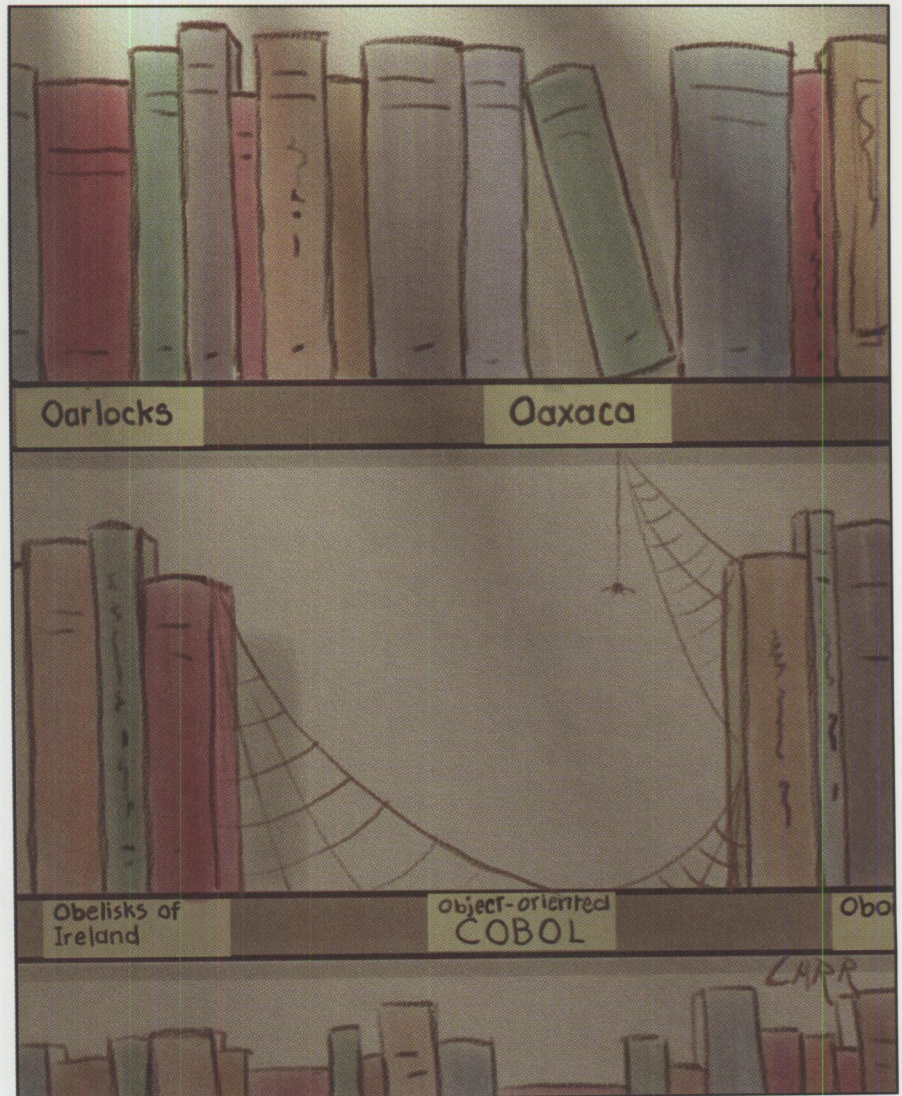
The first part of this article introduces the concept of SOM collection classes, including the various collection classes, iterator classes, and associated abstract classes. The second part illustrates the programming techniques for implementing collection classes through a simple application programmed in object-oriented COBOL with IBM's VisualAge for COBOL for OS/2.

Robert A. Pittman, Jr.
IBM Corporation
Dallas, Texas

Using collections is a common practice in object-oriented programming. *Collection classes* are provided for your convenience. They implement most of the common data structures you encounter in programming, relieving you from the task of coding them.

Collection classes are a set of classes whose purpose is to contain other objects. In

general, a collection may be thought of as an abstract data type of a set of objects that you want to manipulate as a group. A collection is not an array with elements, but a much more sophisticated mechanism for storing and managing objects.



When an element of a collection is referenced, that element is an object, complete with its methods and attributes, not merely data. The collection is considered an object, and it provides various methods for manipulating its elements, as well as itself.

The System Object Model (SOM) collection classes were developed by Taligent, a wholly owned IBM subsidiary. Taligent's systems software is based entirely on object-oriented technology. The company's philosophy, which is to provide open environments that can be extended by software developers, is evident in the structure of the SOM collection classes.

Why Use SOM Collection Classes in an Application?

If collections are implemented in SOM-enabled classes, they can be used unaltered in applications coded in other programming languages supporting SOM, and their reuse by other applications (OO COBOL or otherwise) is facilitated. SOM gives you efficient and reliable implementations of the common abstract data types used in collections, plus it furnishes a framework of properties to guide you in determining the best type of collection to use in your application.

Additionally, collections are unbounded in the number of elements they may contain (at least, any limitations are beyond the practical ones), whereas tables defined in an application program are fixed in size and can require extensive program changes and recompilation when they must be expanded.

Types of Collections

Depending on their type, collections have differing internal structures and differing access methods for manipulation of their elements. The types of collections can be classified into various categories based on the following properties: ordering, keyed access, and uniqueness of entries. *Ordering* means that the elements of the collection may be unordered, sorted, or sequential. *Keyed access* implies a key is used to reference the objects in a collection. *Uniqueness of entries* dictates whether or not the collection may contain duplicate objects.

IBM's System Object Model (SOM) V2.1 implements the following main types of collections:

- **Hash Table**—A collection consisting of (key, value) pairs. The key provides the means for mapping into the collection (or table), and the value is the data element stored in the collection. Hash tables provide fast lookup of a value when given its associated key. Hash tables do not permit two (key, value) pairs to have the same key. Two unique pairs can hash to the same table, but the instantiation of each must be unique. SOM's associated class for this data structure is `somf_THashTable`.
- **Dictionary**—An unordered collection with (key, value) pairs. Equal (key, value) pairs can occur only once. Methods for retrieving a key given its value are provided, but these may be slow. This data structure is implemented in SOM with the `somf_TDictionary` class.

SOM gives you efficient and reliable implementations of the common abstract data types used in collections . . .

- **Set**—An unordered collection of unique objects. If duplicate objects are required, consider using a deque (see the next item) instead. SOM uses `somf_TSet` to implement sets.
- **Deque**—A queue, stack, or deque collection. It is based on the order in which objects are added to, or removed from, the collection. It can be used as a queue or a stack. A *queue* is a list in which elements are inserted, then retrieved via a first-in, first-out (FIFO) approach. A *stack* is a list in which elements are inserted, then retrieved with a last-in, first-out (LIFO) approach. A *deque* is a double-ended queue (hence, its name) that allows insertion and retrieval from either end of the list. Duplicate entries are allowed, and the only ordering of the structure is determined by how elements are inserted into it. Objects can be inserted and removed from any point in the collection. It is the most flexible of the collection classes provided by SOM and is implemented with the `somf_TDeque` class.

- **Primitive Linked List**—A collection in which each element is linked to the element before it and after it. Duplicates are not allowed. The elements of the collection may be traversed using the links between elements. SOM uses `somf_TPrimitiveLinkedList` to define these classes.
- **Sorted Sequence**—A collection whose elements are ordered by size, ranging from largest to smallest. SOM uses the `somf_TSortedSequence` class to implement sorted sequences.
- **Priority Queue**—A special case of the sorted sequence. It keeps the objects of a collection ordered, based on some ordering function. It differs from a queue in that a new element may be inserted and then, say, the largest or smallest deleted (as opposed to the oldest in a straight FIFO queue). SOM uses `somf_TPriorityQueue` to implement priority queues.

Abstract Base Classes

An abstract base class describes general characteristics and cannot be instantiated. Such classes also include pure virtual functions that must be overridden by classes derived from the abstract base classes.

SOM collection classes include the following abstract base classes:

- `somf_TCollection`—Represents a group of objects.
- `somf_TIterator`—Declares the characteristics common to all iterator classes.
- `somf_TSequence`—Declares the characteristics common to all collections with ordered elements.
- `somf_TSequenceIterator`—Declares the characteristics of iterators of collections with ordered elements.

Methods declared in the abstract base classes must be overridden by classes inheriting from them. In some cases, the collection classes and iterator classes provide the override methods; in other cases, you must provide them.

Iterators

An *iterator* for a particular collection allows iteration over each object

contained in the collection. For those conversant with relational databases, it is analogous to a cursor.

Some readers may wonder why iterators are separate and not included in the base collection classes. From an architectural standpoint, this separation is advantageous, because it allows multiple iterators to be defined for a given collection. If iterators were included with the base collection classes, each instantiation of a collection would be limited to a single iterator.

Iterators return objects of the class that is contained in the collection. Note that `somf_TDictionaryIterator` and `somf_THashTableIterator` will return objects of the type `somf_TAssoc`, not simply objects of the `somf_MCollectible` class. `somf_TAssoc` is used to hold a (key, value) pair and is a supporting class that inherits from `somf_MCollectible`.

If a collection changes while an iterator is in use, the iterator is no longer valid. That is, if an iterator is being used to iterate through a collection, and an additional element is added to the collection, the iterator cannot be used to access the remaining elements of the collection. The iterator has to be reset, and iteration has to start over.

If a collection is ordered, the iterator returns the elements of the collection in the correct order. If a collection is unordered, the iterator returns the elements in a random order. Note that iterators are themselves objects, with their own set of methods, and must be instantiated prior to use.

SOM provides the following iterator classes, each of which is associated with one of the main collection classes:

- `somf_THashTableIterator`—Used to iterate over `somf_THashTable` collections.
- `somf_TDictionaryIterator`—Used to iterate over `somf_TDictionary` collections.
- `somf_TSetIterator`—Used to iterate over `somf_TSet` collections.
- `somf_TDequeIterator`—Used to iterate over `somf_TDeque` collections.
- `somf_TPrimitiveLinkedListIterator`—Used to iterate over

Main Collection Class	Mixin Class for Collected Object
<code>somf_THashTable</code>	<code>somf_MCollectible</code>
<code>somf_TDictionary</code>	<code>somf_MCollectible</code>
<code>somf_TSet</code>	<code>somf_MCollectible</code>
<code>somf_TDeque</code>	<code>somf_MCollectible</code>
<code>somf_TPrimitiveLinkedList</code>	<code>somf_MLinkable</code>
<code>somf_TSortedSequence</code>	<code>somf_MOrderableCollectible</code>
<code>somf_TPriorityQueue</code>	<code>somf_MOrderableCollectible</code>

Figure 1. Mixin Classes Used by Objects in Collections

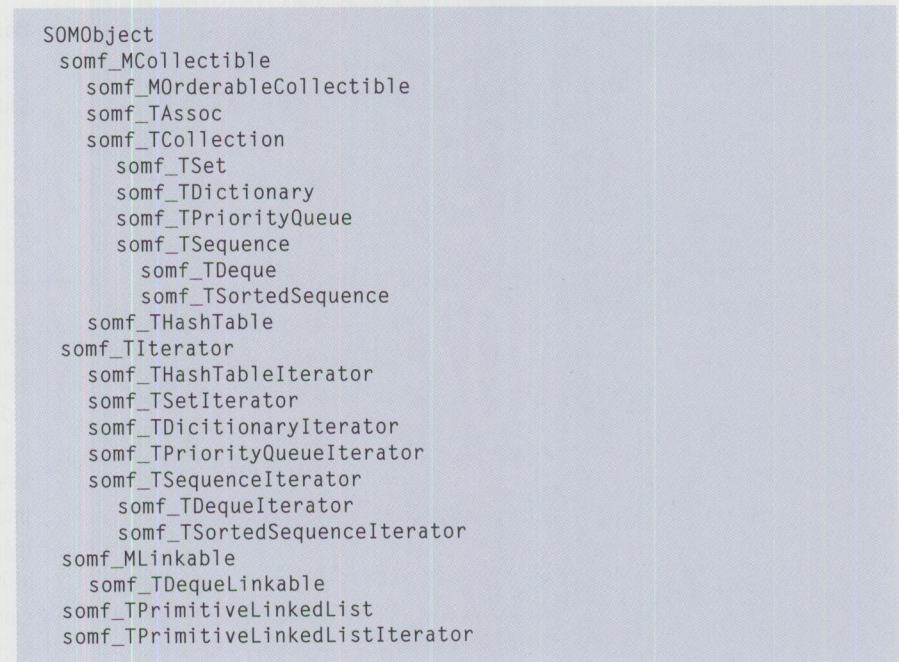


Figure 2. Hierarchy of Collection and Iterator Classes

- `somf_TPrimitiveLinkedList` collections.
- `somf_TSortedSequenceIterator`—Used to iterate over `somf_TSortedSequence` collections.
- `somf_TPriorityQueueIterator`—Used to iterate over `somf_TPriorityQueue` collections.

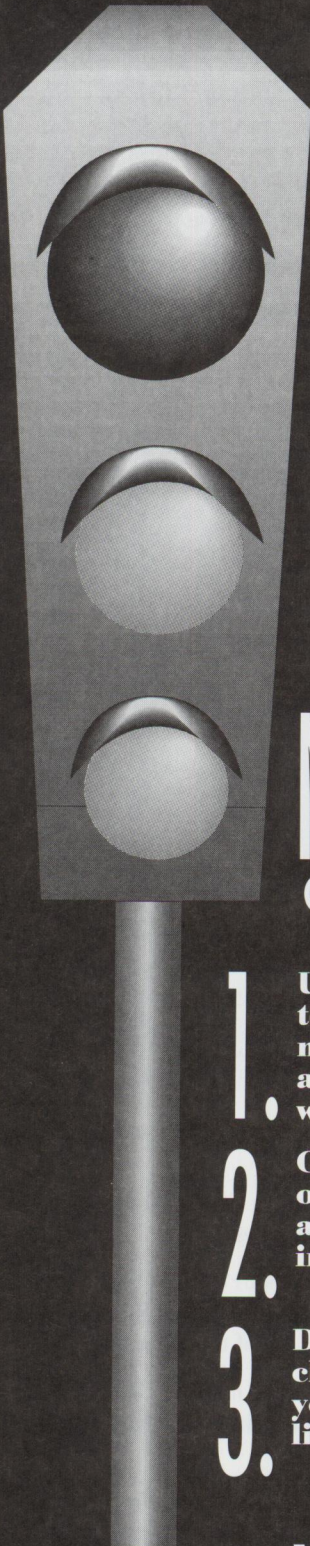
Mixin Classes

Mixin classes are “mixed in” with other classes to produce new classes. For an object to be eligible for use in a collection, it must inherit from a mixin class. The mixin class declares certain characteristics for the element that the collection class requires in order to process the element. Multiple inheritance allows you to inherit from multiple mixin classes to create specialized collectible classes.

SOM’s mixin classes used by the main collection classes are:

- `somf_MCollectible`—Defines the general characteristics of objects inserted into any of the collection classes.
- `somf_MLinkable`—Defines the general characteristics of objects containing links.
- `somf_MOrderableCollectible`—Defines the general characteristics of objects placed in ordered collections.

Figure 1 shows which of the mixin classes must be inherited by an object being placed into each of the main collection classes.



STOP!

Have you used the reader service card to request fast, free information about the products and services advertised in *Personal Systems*?

NO? With the heavy traffic of new technology to choose from in the personal computer market, you need to know about all the most recent developments.

Caution.

1. Use the advertiser's index to get the reader service numbers of the products and services for which you want to receive literature.
2. Circle the same numbers on the reader service card and fill out the necessary information.
3. Drop it in the mail (at no charge!), and we'll give you request the green light!

YES.
Smart move.

Personal Systems

FREE

Product Information Index

Company	RS #	Page #
Abraxas Software	14	12
ChipChat Technology Group	34	13
Cirrus Technology, Inc.	19	10 & 14
Client Server Networking	20	14
ColoradOS/2	32	11
Computer Data Strategies, Inc.	16	13
Computer Interface Corporation	21	14
CyberSafe	26	17
Cyranex Corporation	22	14
Extended Intelligence	6	8
Indelible Blue, Inc.	31	9
Infrastructure Incorporated	3	6
Intelligent Environments	13	11
International Software Solutions	8	8
Ipsilon Networks	11	10
JBA Interntional	12	11
Lieberman and Associates	38	3
McGraw Hill/CPBS	30	7
Micro Channel Dev. Assn.	37	37
Network Telesys.	35	Cover 3
Oberon Software	15	12
On-Line Data	4	6
Oracle	27	Cover 2
Pinnacle Technology	28	1
P.R. Unlimited, Inc.	25	16
SAMS Publishing	24	16
Seagate Technology, Inc.	17	13
Secant Technologies	7	8
SofTouch Systems, Inc.	36	Cover 4
Spitfire Software	2	6 & 10
Stardock Systems, Inc.	10	9
Syntegration, Inc	9	9
T4 Systems, Inc.	1	6
Vinca Corporation	23	16
Virtual Resources Comm., Inc.	18	13
Visual Circuits	5	8 & 55

Inheritance Hierarchy of SOM's Collection Classes

Figure 2 illustrates the hierarchy of the collection and iterator classes.

Methods

In general, the method names are a good indicator of the functions they perform. Many classes have specialized methods. In this article, only the commonly used methods are listed. For more information, particularly regarding method parameters, consult the *SOMObjects Developer Toolkit Collection Classes Reference Manual* (part number 59G5230—also available in soft copy with the SOMObjects Developer Toolkit 2.1, part number 10H9877). As discussed in the previous section on abstract classes, methods defined in the abstract classes must be overridden in derived classes before they can be used.

For classes derived from the `somf_MCollectible` mixin class, you must provide a `somf_IsEqual` method. This makes sense, because SOM cannot know what your application will consider as object equality.

In classes derived from `somf_OrderableCollectible`, you must provide (in addition to a `somf_IsEqual` method) `somf_IsGreaterThan` and `somf_IsLessThan` methods in order to use the `somf_Compare` method. Again, this makes sense, because SOM cannot know how to evaluate the relationships between your application objects.

For the iterator classes, `somf_First` and `somf_Next` methods are furnished. `somf_First` returns the first element of a collection, and `somf_Next` returns the subsequent element. As noted earlier, iterators of dictionaries and hash tables will return objects of the type `somf_TAssoc`. Once you have retrieved a `somf_TAssoc`, you can use `somf_GetKey` and `somf_GetValue` to get the attributes associated with the (key, value) pair.

Iterators of the types `somf_TSortedSequenceIterator`, `somf_TDequeIterator`, and `somf_TPrimitiveLinkedListIterators` supply `somf_Last` and `somf_Previous` methods, in addition to the `somf_First` and `somf_Next` methods common to all iterator classes.

To create an iterator of the type `somf_THashTableIterator`, you must use the `somf_THashTableIteratorInit` method of the `somf_THashTableIterator` class. This is also true for the `somf_TPrimitiveLinkedListIterator` class; to create an iterator, you must use the `somf_TPrimitiveLinkedListIteratorInit` method.

Referring to the inheritance hierarchy above, you will note that `somf_THashTable` inherits from `somf_MCollectible`, and `somf_TPrimitiveLinkedList` inherits directly from `SOMObject` (the topmost class in the inheritance hierarchy). Hence, there is no `somf_CreateIterator` method, as there is for those classes inheriting from `somf_TCollection`. For other types of iterators, the `somf_CreateIterator` method of the collection class is used.

... methods defined in the abstract classes must be overridden in derived classes before they can be used.

For the main collection classes inheriting from the abstract class `somf_TCollection`, overrides are furnished for the commonly used methods `somf_Add`, `somf_Count`, `somf_DeleteAll`, `somf_Remove`, and `somf_CreateIterator`.

The various collection classes may have methods extending those defined by the base class. For example, `somf_TPriorityQueue` adds `somf_Insert`, `somf_Pop`, and `somf_Peek` methods. `somf_TDeque` adds `somf_AddAfter`, `somf_AddBefore`, `somf_AddLast`, and `somf_AddFirst`. It also adds the stack functions `somf_Pop` and `somf_Push`.

The `somf_THashTable` class is an exception to the above, due to its inheritance from `somf_MCollectible` instead of `somf_TCollection`. It has specific methods for manipulating the (key, value) pairs of type `somf_TAssoc`. `somf_TPrimitiveLinkedList` inherits from `SOMObject` and includes

`somf_Count`, `somf_Remove`, `somf_AddBefore`, `somf_AddAfter`, `somf_AddFirst`, and `somf_AddLast` methods. It does not include a `somf_CreateIterator` method.

Refer to the *SOMObjects Developer Toolkit Collection Classes Reference Manual* for additional information about methods specific to particular collection classes.

Implementing SOM Collection Classes in an Application

In general, you'll follow these steps to use a collection class in an application:

1. Determine the type of collection to be used.
2. Define a "collectible" object that is to be placed into the collection by inheriting from the appropriate mixin class.
3. Define a collection object for the type of collection being used.
4. Define an iterator object for the type of collection being used.

An Illustration

An old programmers' adage says that one good example is worth a thousand pages in a manual. The concepts outlined above seem inadequate to prepare you for the task of implementing a SOM collection in a VisualAge COBOL application. To alleviate the shortcomings in the text, I now illustrate the techniques with an example. This example implements a collection of type `set` in which duplicates are not allowed, and their order in the collection is of no consequence. The illustration demonstrates the commonly used methods of `somf_TCollection`, as overridden by `somf_TSet`. The example also shows the use of an iterator on the collection.

The application is a contrived order-entry system consisting of four modules: a client program, an order, order items, and a user interface. *Order items* are the collected objects, and they are contained in the collection defined in the order object. The *client program* instantiates the order, creates the order items, and invokes the appropriate methods in the order object to add the order items.

To keep things simple, the example uses a command-line interface instead of a GUI. Using the command-line interface also serves to make the example portable.

```
CLASS-ID. "OrderItem" INHERITS somf-MCollectible.
```

Figure 3. OrderItem CLASS-ID Statement

```
REPOSITORY.
  CLASS OrderItem          IS "OrderItem"
  CLASS somf-MCollectible IS "somf_MCollectible".
```

Figure 4. REPOSITORY for the OrderItem Class

```
IDENTIFICATION DIVISION.
METHOD-ID. "somfIsEqual"  OVERRIDE.
DATA DIVISION.
LOCAL-STORAGE SECTION.
  01 ItemNumber           PIC X(10).
  01 ItemCost             PIC 999V99.
LINKAGE SECTION.
  01 LS-EV                USAGE POINTER.
  01 theOrderItem         USAGE OBJECT REFERENCE OrderItem.
  01 theEqualFlag         PIC X.
PROCEDURE DIVISION
  USING BY VALUE LS-EV
  BY VALUE theOrderItem
  RETURNING theEqualFlag.
  INVOKE theOrderItem "getNumber" RETURNING ItemNumber.
  INVOKE theOrderItem "getCost"  RETURNING ItemCost.
  IF (Item-Number = ItemNumber) AND
    (Item-Cost = ItemCost)
    THEN MOVE X"01" TO theEqualFlag
  ELSE
    MOVE X"00" TO theEqualFlag.
  EXIT METHOD.
END METHOD "somfIsEqual".
```

Figure 5. somfIsEqual Override

```
CLASS-ID. "Order" INHERITS SOMObject.
```

Figure 6. The Order Class

```
REPOSITORY.
  CLASS SOMObject          IS "SOMObject"
  CLASS SOMCollection      IS "somf_TSet"
  CLASS SOMIterator        IS "somf_TSetIterator"
  CLASS OrderItem         IS "OrderItem".
```

Figure 7. Repository for the Order Class

This is implemented with a user interface object, which is also instantiated and invoked by the client program. The user interface prompts the system user for input and displays the application's output to the user.

I used IBM's VisualAge for COBOL for OS/2 V1.1 with the FixPak 1 Corrective Service Diskette applied. The relevant compiler option was PGMNAME(MIXED),

which allows mixed case names (necessary because SOM uses mixed case). Also required is the SOM Developer's Toolkit, release 2.1, with all the subsequent maintenance applied.

The OrderItem Class

Instantiations of the OrderItem class constitute the collected objects. This class consists of two attributes: an item number and an item cost. It has get and

set methods for each of these attributes and an additional method, somfIsEqual. somfIsEqual is a method in somf_MCollectible that must be overridden.

In the overridden somfIsEqual method, we define the state of equality for our collectible objects. Overriding this method is enforced by SOM, so we will receive a runtime error if the method is not present in our class. In our case, if the item numbers and the item costs of two OrderItem objects are equal, then we consider the two objects equal.

Figure 3 shows the CLASS-ID statement for the OrderItem class definition. Of interest is that it inherits from somf_MCollectible instead of SOMObject, as is typical. Of course, somf_MCollectible inherits from SOMObject, so OrderItem ultimately does as well.

Figure 4 shows the REPOSITORY for the OrderItem class. Notice that somf_MCollectible is defined to be somf-MCollectible, because COBOL does not accept underscores (_) in class names.

Figure 5 shows the overridden somfIsEqual method defined in the class definition of OrderItem. It takes as input the SOM global environment variable (LS-EV, defined as a pointer) and an OrderItem object (theOrderItem, defined as an object reference). It returns a flag indicating if the OrderItem object passed to it as theOrderItem is equal to the OrderItem object upon which the somfIsEqual method is invoked. In other words, we are invoking this method in an OrderItem object, passing to it another OrderItem object, and asking it to compare the two.

Notice also in this method that we are invoking the get methods for the attributes of the passed object (ItemNumber and ItemCost), then comparing them to the attributes of this OrderItem (Item-Number and Item-Cost).

The Order Class

The Order class is straightforward and has the attributes of an order number, an order date, and a collection of OrderItem objects. It also has an iterator defined so that it can iterate through the collection. Its methods include overrides of somDefaultInit and somFree; get

and set methods for the simple attributes of date and number; methods to add and remove OrderItem objects from the collection; and a method that calculates the cost of the order by iterating through the collection.

In Figure 6, we see from the CLASS-ID statement that the order class inherits from the mother of all objects, SOMObject.

In the repository coded in Figure 7, note that the CLASS statement for SOMCollection points to somf_TSet. This indicates that the collection contained in the Order class will be of type set. Further, note that the CLASS statement for SOMIterator points to somf_TSetIterator. This iterator is needed to iterate through the collection.

The working-storage section of the class definition (see Figure 8) shows the simple attributes of Order-Number and Order-Date in addition to the collection, Order-Collection, which is an object reference to SOMCollection. SOMCollection is a somf_TSet collection. The working-storage section also shows the associated iterator, Order-Iterator, which is an object reference to SOMIterator. SOMIterator is a somf_TSetIterator. Finally, WS-EV is a pointer that is used to point to SOM's global environment variable.

In Figure 9, the method somDefaultInit (inherited from SOMObject) is overridden. This override is necessary so that the global environment variable (WS-EV), the collection, and the iterator can be created during the Order object's initialization. The environment variable is obtained by calling somGetGlobalEnvironment.

The collection is created by invoking the somNew method (inherited from SOMObject) on the class SOMCollection (which is a somf_TSet). The collection's handle, Order-Collection, is returned.

After creating the collection, we invoke the somfCreateIterator method (which is inherited from somf_TSet) on its handle. This action creates the iterator of the collection, Order-Iterator, returned from the somfCreateIterator method.

In Figure 10, we override the somFree method inherited from SOMObject. In this

```
WORKING-STORAGE SECTION.
01 Order-Object.
   05 Order-Number          PIC X(5).
   05 Order-Date            PIC X(8).
   05 Order-Collection      USAGE OBJECT REFERENCE SOMCollection.
   05 Order-Iterator        USAGE OBJECT REFERENCE SOMIterator.
01 WS-EV                    USAGE POINTER.
```

Figure 8. Working-Storage Section of Class Definition

```
IDENTIFICATION DIVISION.
METHOD-ID. "somDefaultInit"  OVERRIDE.
DATA DIVISION.
PROCEDURE DIVISION.
   CALL "somGetGlobalEnvironment" RETURNING WS-EV.
   INVOKE SOMCollection "somNew"  RETURNING Order-Collection.
   INVOKE Order-Collection "somfCreateIterator" USING BY VALUE WS-EV
                                           RETURNING OrderIterator.

   EXIT METHOD.
END METHOD "somDefaultInit".
```

Figure 9. Overriding the somDefaultInit Method

```
IDENTIFICATION DIVISION.
METHOD-ID. "somFree"  OVERRIDE.
DATA DIVISION.
PROCEDURE DIVISION.
   INVOKE Order-Collection "somDeleteAll" USING BY VALUE WS-EV.
   INVOKE Order-Iterator  "somFree".
   INVOKE Order-Collection "somFree".
   INVOKE SUPER           "somFree".
   EXIT METHOD.
END METHOD "somFree".
```

Figure 10. Overriding the somFree Method

```
IDENTIFICATION DIVISION.
METHOD-ID. "addOrderItem ".
DATA DIVISION.
LOCAL-STORAGE SECTION.
   01 LSS-Before-Count      PIC S9(8)  COMP.
   01 LSS-After-Count       PIC S9(8)  COMP.
   01 LSS-CollectedOrderItem USAGE OBJECT REFERENCE OrderItem.
   01 LSS-theEqualFlag      PIC X.
   01 LSS-Item-Found-Flag   PIC X.
   01 LSS-Item-Count        PIC S9(8)  COMP.
   01 LSS-Loop-Count        PIC S9(8)  COMP.
```

Figure 11. Working Variables of the addOrderItem Method

```
LINKAGE SECTION.
   01 LS-OrderItem          USAGE OBJECT REFERENCE OrderItem.
   01 LS-Parms.
       05 LS-Item-Count     PIC S9(8)  COMP.
       05 LS-Flag           PIC X.
PROCEDURE DIVISION
   USING LS-OrderItem
   RETURNING LS-Parms.
```

Figure 12. Parameters Passed to the addOrderItem Method

```

MOVE X"00" TO LSS-Item-Found-Flag.
INVOKE Order-Collection "somfCount" USING BY VALUE WS-EV
RETURNING LSS-Before-Count.
MOVE LSS-Before-Count TO LSS-Item-Count.

```

Figure 13. Getting the Count of Objects in the Collection

```

IF LSS-Item-Count NOT = 0
  THEN INVOKE Order-Iterator "somfFirst"
        USING BY VALUE WS-EV
        RETURNING LSS-CollectedOrderItem
        PERFORM CHECK-EQUAL
END-IF.

```

Figure 14. Getting the Collection's First Element

```

SUBTRACT 1 FROM LSS-Item-Count.
IF LSS-Item-Count > 0
  THEN PERFORM VARYING LSS-Loop-Count
        FROM 1 BY 1
        UNTIL LSS-Loop-Count > LSS-Item-Count
              OR LSS-Item-Found-Flag = X"01"
              INVOKE Order-Iterator "somfNext"
                    USING BY VALUE WS-EV
                    RETURNING LSS-CollectedOrderItem
              PERFORM CHECK-EQUAL
        END-PERFORM
END-IF.

```

Figure 15. Getting Subsequent Elements of the Collection

```

IF LSS-Item-Found-Flag = X"00"
  THEN INVOKE Order-Collection "somfAdd"
        USING BY VALUE WS-EV
        BY VALUE LS-OrderItem
END-IF.

```

Figure 16. Adding an Element to the Collection

```

INVOKE Order-Collection "somfCount"
  USING BY VALUE WS-EV
  RETURNING LSS-After-Count.
MOVE LSS-After-Count TO LS-Item-Count.
IF LSS-Before-Count = LSS-After-Count
  THEN MOVE X"01" TO LS-Flag
ELSE
  MOVE X"00" TO LS-Flag
END-IF.
EXIT METHOD.

```

Figure 17. Checking the Element Count After the Add Operation

```

CHECK-EQUAL.
INVOKE LSS-CollectedOrderItem "somfIsEqual"
  USING BY VALUE WS-EV
  BY VALUE LS-OrderItem
  RETURNING LSS-theEqualFlag.
IF LSS-theEqualFlag = X"01"
  THEN MOVE X"01" TO LSS-Item-Found-Flag.
END METHOD "addOrderItem".

```

Figure 18. Invoking the Overridden somfIsEqual Method

method, we are undoing the actions we performed in the `somDefaultInit` override. (If we don't override `somFree` in the `Order` object, the objects in the collection contained in the object won't be destroyed, and a memory leak will occur.) First, the method `somDeleteAll`, which is inherited from `somf_TSet`, is invoked on the collection. This deletes all the objects placed in the collection and frees the storage they occupied.

Next, `somFree`, inherited from `SOMObject`, is invoked to delete the iterator and free its storage. The collection is then freed by `somFree`. Finally, we invoke `somFree` on the `Order` object's super class (in this case, `SOMObject`) to free the `Order` object and the storage it uses.

Figure 11 shows the working variables used in the `addOrderItem` method of the `Order` class. Note that local storage is used in lieu of working storage. Local storage is refreshed each time the method is invoked; working storage will be in the last used state on each invocation.

Figure 12 illustrates the parameters passed to the `addOrderItem` method. An `OrderItem` object is taken as input, and a structure is returned. Because the return clause allows the method to return only a single address, and we are returning two items (`LS-Item-Count` and `LS-Flag`), we group them together under `LS-Parms` to circumvent this limitation.

In Figure 13, the `somfCount` method, inherited from `somf_TSet`, is invoked on the collection. This method returns a count of the number of objects in the collection. Because nothing has been added thus far in the method, this constitutes a "before" count.

In Figure 14, the `somfFirst` method is invoked on the iterator object. This method returns the first item in the collection. Notice that we are testing the count obtained in Figure 13 to verify that the collection contains objects before invoking the `somfFirst` method. Upon the return from the method, the `CHECK-EQUAL` paragraph (described in Figure 18) is performed.

The segment of code in Figure 15 iterates through the collection by invoking `somfNext` on the iterator object. The `somfNext` method returns the next object

in the collection. (Note that we cannot invoke `somfNext` first and expect the first element to be returned.) The `somfFirst` method performs some initialization of the iterator and must be completed before `somfNext` can be used. After the `somfNext` method returns, we perform the CHECK-EQUAL paragraph, described in Figure 18.

In Figure 16, the `somfAdd` method is invoked on the collection. The `somfAdd` method is passed two parameters: a pointer to the global environment variable and an object reference to the object that is to be added to the collection. Both parameters are passed BY VALUE and not BY REFERENCE, the customary manner used by COBOL programs. Notice that we are checking a flag to see if the `OrderItem` object was found during our iteration through the collection.

In Figure 17, we again invoke `somfCount` on the collection. This returns an "after" count of the number of objects in the collection. If the `OrderItem` object was successfully inserted into the collection, the "after" count will be greater than the "before" count, and a flag in the return parameters is set accordingly. If the counts are the same, the object could not be inserted, and the return flag is set appropriately.

Figure 18 shows the paragraph that is performed each time an object is retrieved from the collection. This paragraph invokes the `somfIsEqual` method of the `OrderItem` object. (This method is described in Figure 5.) The method is passed the `OrderItem` object that we want to insert in the collection. In essence, we are retrieving each item in the collection and, for each one returned, we are asking it to compare itself to the object we are attempting to insert.

Figures 11 through 18 represent the `addOrderItem` method that adds an `OrderItem` object to the collection. To remove an item, a `removeOrderItem` method is coded, which works in a similar fashion.

We iterate through the collection, looking for a match with the object we want to remove. Upon finding a match, the `somfRemove` method is invoked on the collection and is passed the same parameters as the `somfAdd` method described above.

```
IDENTIFICATION DIVISION.
METHOD-ID. "calculateCost".
DATA DIVISION.
LOCAL-STORAGE SECTION.
    01  LSS-CollectedOrderItem USAGE Object REFERENCE OrderItem.
    01  LSS-Item-Count          PIC S9(8)  COMP.
    01  LSS-Cost                PIC 999V99.
LINKAGE SECTION.
    01  LS-Cost                 PIC 9(7)V99.
PROCEDURE DIVISION
    RETURNING LS-Cost.
    MOVE ZERO TO LS-Cost.
    INVOKE Order-Collection      "somfCount"
    USING BY VALUE WS-EV
    RETURNING LSS-Item-Count.
    IF LSS-Item-Count > 0
        THEN INVOKE Order-Iterator "somfFirst"
            USING BY VALUE WS-EV
            RETURNING LSS-CollectedOrderItem
            PERFORM GET-COST
    END-IF.
    SUBTRACT 1 FROM LSS-Item-Count.
    IF LSS-Item-Count > 0
        THEN PERFORM LSS-Item-Count TIMES
            INVOKE Order-Iterator "somfNext"
            USING BY VALUE WS-EV
            RETURNING LSS-CollectedOrderItem
            PERFORM GET-COST
        END-PERFORM
    END-IF.
EXIT METHOD.

GET-COST.
    INVOKE LSS-CollectedOrderItem "getCost" RETURNING LSS-Cost.
    ADD LSS-Cost TO LS-Cost.

END METHOD "calculateCost".
```

Figure 19. Iterating through the Collection Elements

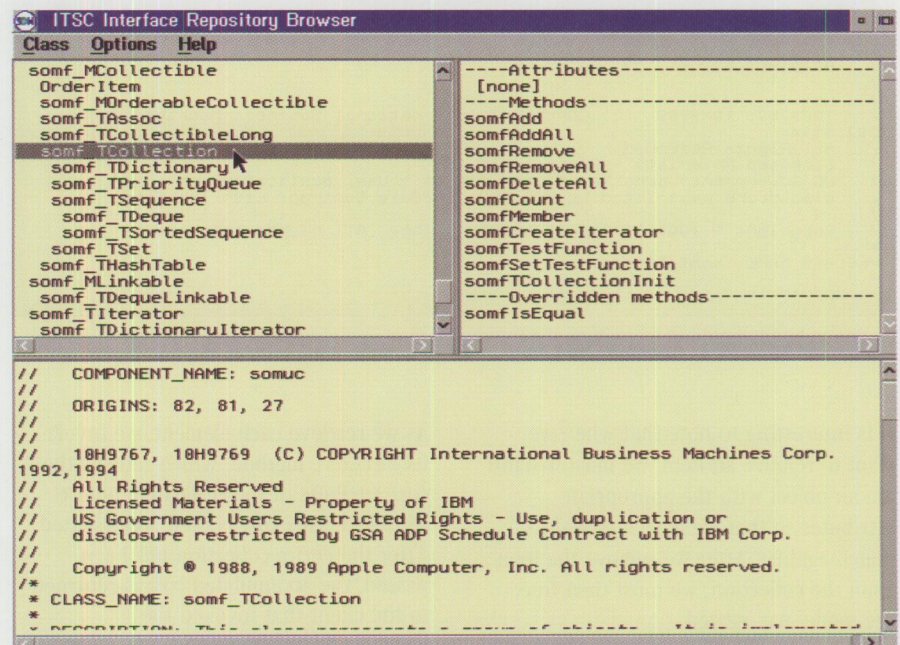


Figure 20. Methods of Abstract Class `somf_TCollection`

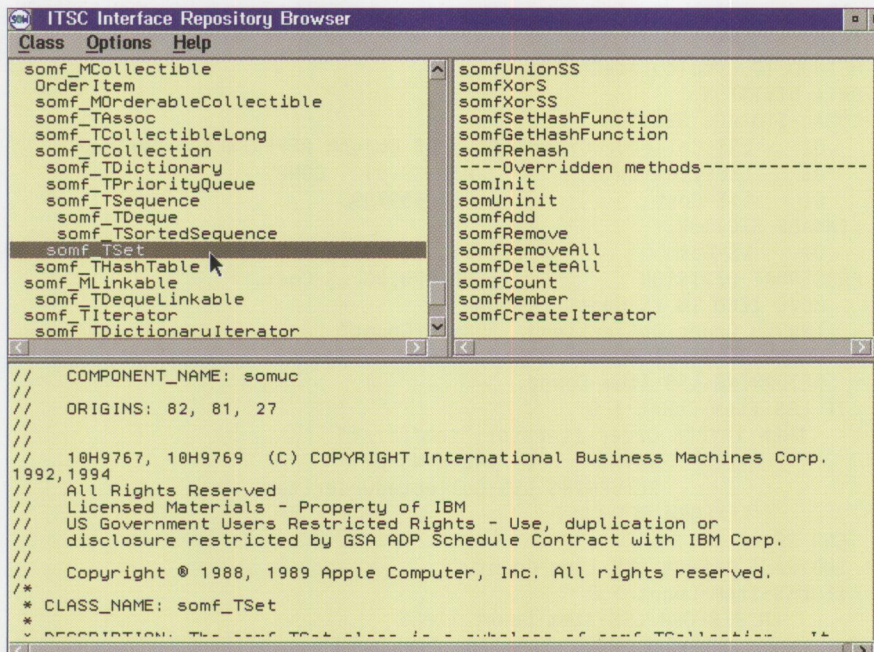


Figure 21. Overriding Methods of Class somf_TSet

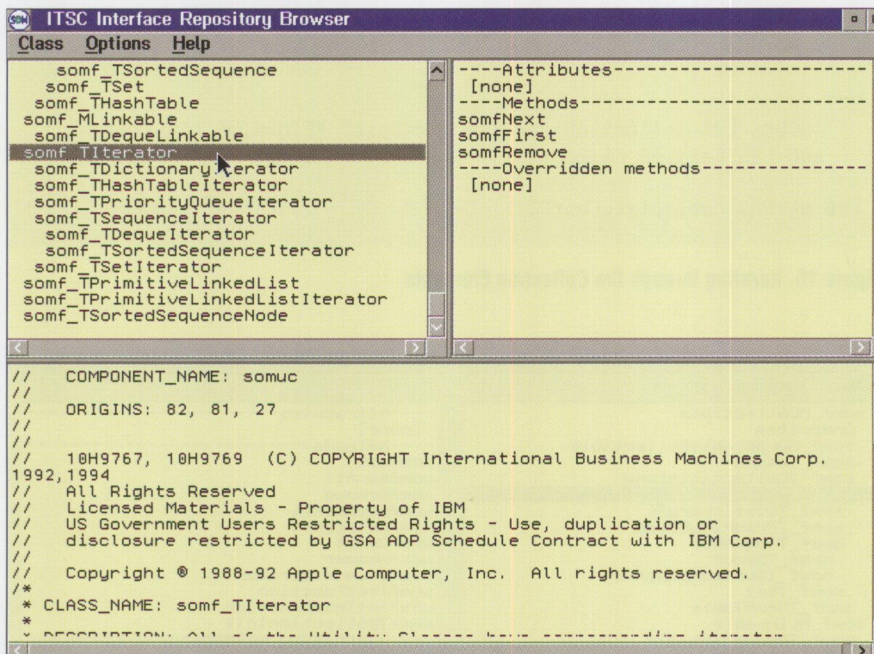


Figure 22. Methods of Abstract Class somf_TIterator

It is interesting to note that when we want to remove an item, we must instantiate an object with the appropriate attributes so that we have a model to match against. After we remove the item from the collection, we must then free the model we created.

Figure 19 shows the calculateCost method of the Order object. This method further illustrates the use of an iterator object to iterate through the collection.

As we retrieve each element, we invoke its getCost method, which returns the cost attribute of the OrderItem object retrieved. This cost is then accumulated. After the entire collection has been passed, the accumulated cost is returned to the client that invoked the calculateCost method.

Figures 20 through 23 illustrate a point mentioned earlier in this article during the brief discussion of abstract base classes.

Figure 20 shows the methods defined by somf_TCollection, an abstract class. Note that the methods somfAdd, somfRemove, somfDeleteAll, somfCount, and somfCreateIterator are all used in our sample application.

Next, contrast these methods with the overridden methods for somf_TSet, shown in Figure 21. somf_TSet inherits from the base class somf_TCollection and overrides these methods defined in the base class. This same case is also true for the iterator classes.

Figure 22 shows the methods defined in the abstract base class somf_TIterator.

Figure 23 shows the overrides for these methods defined in somf_TSetIterator, which inherits from somf_TIterator.

Figures 20 through 23 were taken from the SOM class browser included with the IBM Redbook titled *SOM Objects: Management Utilities for Distributed SOM* (GG24-4479). I strongly recommend this browser for those of you who want to use SOM in your application programs.

The Client Program and the UserInterface Class

The UserInterface class is instantiated by the client program and handles input from and output to the system user. As noted earlier, it is a straight command-line interface, lacking a GUI. It has no attributes, and its methods are concerned with communicating with the user.

The client program drives the system process. Both the UserInterface class and client program are typical object-oriented COBOL programs and have no references to SOM collection classes. For the sake of brevity, I will not explain them here.

The entire sample application discussed in this article consists of a client program and three class definitions. Complete COBOL source code for the Client, Order, OrderItem, and UserInterface modules is available on the World-Wide Web at <http://pscc.dfw.ibm.com/psmag/>.

Putting It All Together

After you build the application, execute it from an OS/2 command line by invoking the client.exe file. Figure 24 shows a sample execution.

If you want to place your classes in a SOM interface repository (IR), the SOMObjects Developer Toolkit 2.1 is required. This product includes the necessary interface definition language (IDL) module (mcollect.idl) and the dynamic link library (DLL) module (somuc.dll) for the collection classes. This DLL is also furnished with OS/2 Warp but not with OS/2 2.1 or 2.11. Building the application in VisualAge COBOL and creating a SOM IR are typical application programming functions outside the scope of this topic.

Collection Classes: Important in Object-Oriented Programming

The techniques illustrated in the second part of this article are typical of the procedures used to implement collections. You can also use alternatives such as placing a C++ interface between the COBOL program and the collections. Insulating the COBOL program (and programmer) in such a manner only delays the learning curve that COBOL programmers must undergo in an object-oriented programming environment.

Using collection classes is an important part of object-oriented programming, and learning and applying the necessary techniques for implementing them should become part of the VisualAge COBOL programmer's repertoire.

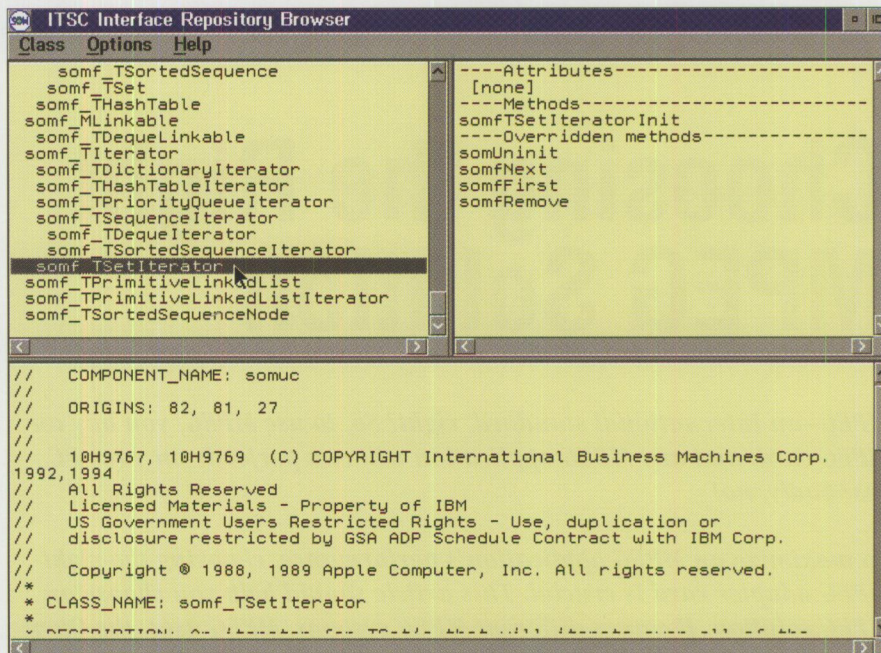


Figure 23. Overriding Methods of Class somf_TSetIterator

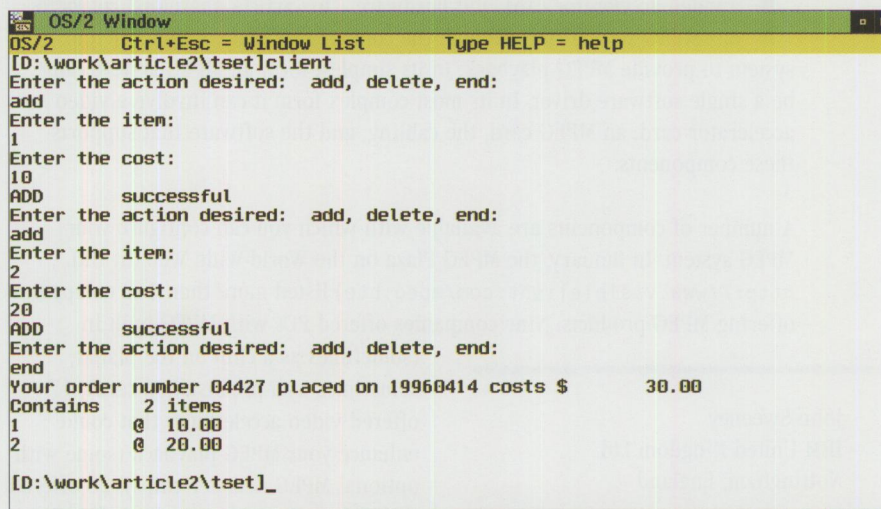


Figure 24. Sample Execution of the Collection Class Application



Robert A. Pittman, Jr. works in the Object Technology University within IBM Education and Training, Dallas, Texas. He provides instruction and develops courses in object orientation and performs on-site consultation in object-oriented development. He is co-author of *IBM VisualAge for COBOL for OS/2: Object-Oriented Programming* (SG24-4606), an IBM Redbook. Rob's prior IBM jobs were in national technical support and software development. He has been involved in the computer industry since 1976. Before joining IBM in 1988, he worked in application development, database administration, and systems programming. Rob has a BBA degree in Accounting from the University of Texas and an MBA in Information Systems from the University of North Texas. He is a Certified Public Accountant licensed in Texas and a member of the American Institute of Certified Public Accountants. His Internet ID is robert_pittman@vnet.ibm.com.

Choosing the Right MPEG Solution

MPEG—an international standard, right? So, to use MPEG, you buy an MPEG card, and any MPEG application will run perfectly on your PC, yes? Sadly, no!

To maximize an MPEG application's performance, choosing the right MPEG adapter card is crucial. This article should help you choose an MPEG solution. It covers different MPEG systems, MPEG uses, the Open MPEG Consortium, major decision criteria, and several selection criteria.

There are three ways to play MPEG: software only, software driver with a video accelerator card, and hardware. This article, therefore, refers to an "MPEG system" meaning all the components that combine in your system to provide MPEG playback. In its simplest form an MPEG system can be a single software driver. In its most complex form it can involve a video accelerator card, an MPEG card, the cabling, and the software that supports these components.

A number of components are available with which you can configure your MPEG system. In January, the MPEG Plaza on the World-Wide Web (at URL <http://www.visiblelight.com/mpeg.htm>) listed more than 140 companies offering MPEG products. Nine companies offered PCs with MPEG built in (sometimes as a chip on the planar, sometimes as a pre-installed card), 17 offered video accelerators that could enhance your MPEG playback (some with optional MPEG decoder chips), 28 offered MPEG decoder cards, and five offered software MPEG decoders.

John Sweeney
IBM United Kingdom Ltd.
Nottingham, England

A key consideration when configuring your MPEG system is whether your system has an actual hardware MPEG decoder chip inside. This chip can be built into the planar, provided as an option on a video accelerator card, or be part of an MPEG decoder card. Another possibility is that your PC has an extra processor, such as an MWave digital signal processor (DSP), which has been programmed to do some of the MPEG decompression and take some of the load off your main processor.

Some MPEG decoders will only play back to a PC screen and others only to a TV screen. Some are more flexible and can do both, or you can add extra hardware to provide more flexibility. This article is primarily about playing back to your PC screen, although most of it is also relevant to TV-screen playback.

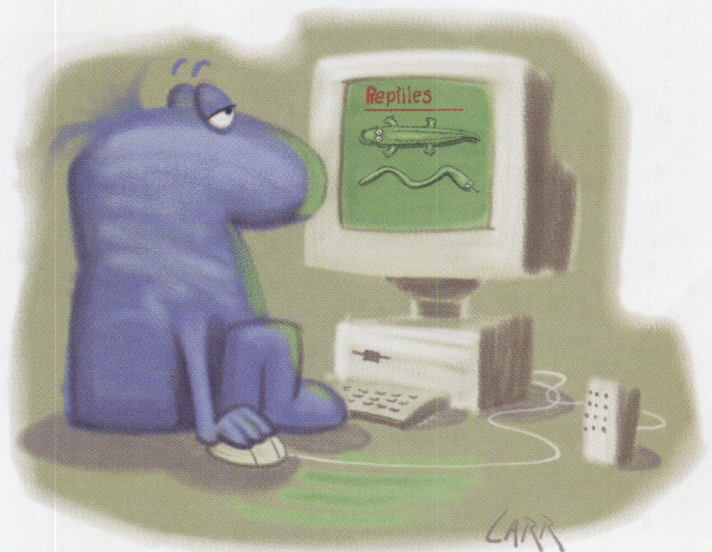
This article assumes that the "device" from which you are playing the MPEG stream is capable of providing the data fast enough. Fragmented hard drives, slow CD-ROM drives, poorly configured systems, slow networks, or overloaded resources can all decrease the playback quality.

MPEG Uses

Apart from specialized uses such as VideoCD, CD-i, or Karaoke CDs (which are not covered in this article), there are two main uses of MPEG on a PC:

- Running the media player that comes with your operating system or with the MPEG card; the MPEG clips play in an operating system window.
- Running an application that sends commands to the MPEG card and plays MPEG as part of the application. In 1995 the chance of an application working with an MPEG card for which it was not designed was quite small.

The compatibility problem is that the relevant international standards cover only compression and decompression of the



video material. They cover neither the details of exactly which commands and parameters the applications can send to the drivers, nor how those commands and parameters should be interpreted.

The standard way of communicating from an application program to a multimedia device driver is via Media Control Interface (MCI) strings. MCI includes a wide range of commands, from simple OPEN and PLAY to complex commands such as PUT DESTINATION AT and SIGNAL RETURN POSITION. MCI covers all multimedia devices, although some of the commands are device-specific.

The Open MPEG Consortium

The Open MPEG Consortium (OM-1) has published standards for MPEG MCI strings based upon the ones that Microsoft specified in its "Digital Video Command Set for the Media Control Interface." Unfortunately there appear to be some variations both in the interpretation of these specifications by various MPEG card manufacturers and also in which subset of the full command and parameter set they have chosen to implement. The Open MPEG Consortium is continuing with this work, and hopefully things will improve in 1996.

For the moment you should never assume that an application written for one MPEG system will work at all on a different MPEG system, let alone behave identically. Even little details such as what happens to the MPEG window when you issue a STOP command—whether the last image freezes, the window goes to a particular color, or any graphics in that part of the screen reappear—can ruin your application.

You can find the Open MPEG Consortium on the World-Wide Web at <http://www.om-1.org>. The MPEG Plaza also links to other useful information.

Major Decision Criteria

The first question you should ask about an MPEG application and a particular MPEG system is: Will it work at all? Various other factors can affect this as well, as detailed ahead.

The second question, even when playing an MPEG clip with a media player, is: Is the quality acceptable for my purposes? Various aspects of quality are discussed below.

You should by now have realized that you will need to test any MPEG system you are planning to use. To accurately analyze the quality of an MPEG system, it is critical to test with high-quality MPEG clips. (The IBM TV commercials available on CD [order number GV31-3886] are a good starting point.) Poor quality MPEG clips played on a good system will exhibit many of the same symptoms as good MPEG clips played on a poor system.

The quality of the MPEG clips depends upon many variables. (See this article's appendix for some considerations.)

The quality of the MPEG clips depends upon many variables.

European, PAL-sourced, MPEG-1 tends to produce 352x288 resolution at 25 frames per second. All of the examples in this article use 352x288 resolution. American, NTSC-sourced, MPEG-1 tends to produce 352x240 at 30 frames per second.

Criteria for Selection

When selecting an MPEG card, some of the criteria you will need to consider are: size, color depth, overlay capability, frame rate, sound playback, audio cabling, lip synch, operating system, MCI command set, controls, multiple tasks, and MPEG-2 versus MPEG-4. Each of these is discussed next.

Size

The resolution to which the screen is set affects the size and quality of the resulting MPEG playback. A 352x288 window will give the best quality.

On a 640x480 display, a 352x288 window is just over a quarter of the screen size and looks quite reasonable for many purposes.

On an 800x600 or 1024x768 display, a 352x288 window is a little small, so you may need to increase the size of the playback window. Virtually all new computers now have the capability to display at either 800x600 with 65 thousand colors or 1024x768 with 256 colors with the standard 1 MB of video RAM; 2 MB or more will give higher capabilities.

There are three main scaling options:

- Full screen
- Double size (704x576)
- Random sizes

Always carefully determine what is meant by "full screen"—some MPEG cards specify this but really mean 640x480. Some cards are optimized to work well at certain "full screen" sizes.

Double size is actually four times as big because it is working in two dimensions. Some cards call it "zoom by 2." Some cards are optimized to work well at double size.

The information in the MPEG file is only for 352x288 pixels. If your window is bigger it contains more pixels, and the system has to guess what the extra pixels should contain. The simplest systems duplicate the adjacent pixels giving a blocky effect; the most sophisticated systems use complex algorithms to generate a best guess at what the extra pixels should contain in order to give the smoothest possible picture. This process is referred to as *interpolation*.

Even if your MPEG system works well at double size or full screen, you may find that the quality is not as good if you randomly choose any other size, because the algorithms may have been optimized for particular scaling options. Always test your system at the planned size.

Color Depth

First consider the color depth to which you set your computer's normal video mode. As mentioned previously, 65 thousand colors (16-bit) or more is rapidly becoming the norm. Check to see whether the MPEG system you are using can function at your chosen color depth.

Even if your base system is capable of 65 thousand colors, some MPEG cards attach via the VGA Feature Connector, which does not support 65 thousand colors on all systems.

Depending upon how the MPEG system works (see "Overlay Capability" next), the MPEG playback may be affected by the color depth to which the video subsystem is set. If you have only 1 MB of video

RAM and you want 1024x768 resolution, then you can only use 256 colors; if this restricts your MPEG playback to 256 colors, then the quality will be poor.

The second consideration, which may be independent of the color depth to which you have set your screen, is the color depth of the MPEG window itself. MPEG cards range from around 16 bits up to a full 24 bits (16 million colors). This factor affects not only the quality of the basic 352x288 window's color reproduction, but also how good the interpolation can be in a larger window. It is no good having a system that can predict exactly what color a certain pixel should be if it is incapable of displaying that color.

You can check your system documentation to determine what chance it has of showing a smooth image at double size or full screen. But the only way to be *sure* is to test it with some MPEG clips of known quality. Check carefully for how well it handles diagonal lines and moving faces.

Overlay Capability

There are many different ways in which the MPEG window's pixels are merged with the pixels for the rest of the screen before they are actually displayed. For example:

- The output from the video subsystem is fed to the MPEG system, which adds the MPEG information and sends the complete image to the screen.
- The MPEG system sends its data to the video subsystem, which combines the two and sends the complete image to the screen.
- The MPEG system generates a PAL or NTSC output which is fed to a TV subsystem that converts it to a PC image and sends it on to either the screen or the video subsystem.

The merging of the MPEG pixels may be done in one of two ways depending upon your system:

- Simply place a rectangle containing *all* the MPEG pixels into a rectangular area of the display, overwriting whatever is there.

- Specify a chromakey color, so that only pixels set to that color within the rectangle are changed to the corresponding pixels from the MPEG frame.

MPEG systems that do not support any variation of chromakeying cannot perform effects such as having a graphic overlay part of the MPEG window or having animation play across the MPEG window. If you need to do more than just play a rectangular window, then check that your MPEG system can support it.

Frame Rate

You should expect a system with a hardware MPEG decoder to be able to play back the video at its full speed, 25 or 30 frames per second.



Systems based upon a software MPEG driver, whether or not assisted by a video accelerator, may not be able to achieve this rate. They are also much more affected by other tasks running simultaneously on your machine than is a pure hardware solution.

The achievable frame rate will depend upon the power of your processor and the size at which you play the MPEG clip. If you are using a video accelerator, the achievable frame rate also depends upon which facilities are supported by the video accelerator and its drivers. For example, Windows requires a Display Control Interface (DCI) to achieve any significant benefits; however, DCI support comes in many flavors, and the quality of MPEG playback can be drastically

affected by whether your video accelerator supports Primary or Offscreen Surface Type, whether it is Stretch-Capable, and which Hardware Color Space Conversion it provides.

Some MPEG subsystems have a diagnostic capability that reports to you how many frames per second they achieve. When the MPEG system cannot cope with the frame rate, there are usually two ways the system can react:

- Play all the frames, even though it will take longer. This, of course, causes major loss of synchronization with any sound track.
- Drop frames so that the clip plays for the right duration and stays in synch with the sound track.

A good system should allow you to choose which way the system will react.

Whether you can accept lower frame rates depends upon the type of material you are playing and your application's requirements.

Further complicating the matter is the fact that MPEG supports a wide range of bit rates and picture sizes. To keep the costs down and avoid over-engineered decoders, a subset of the allowable MPEG parameters, known as the *constrained parameter set*, has been defined. Hopefully, all hardware MPEG decoders will be able to cope with streams that are constrained. The constrained bit rate is 1.856 Mbits per second.

To be sure that your system can cope, you need to test it with MPEG clips that have the same bit rate as the ones you are going to use in your application.

Sound Playback

So far we have concentrated on the visual aspect of an MPEG playback. But a video usually contains an audio track. Although you can encode a .VBS (video bit stream) file that contains only video and a .ABS (audio bit stream) file that contains only audio, this article is primarily concerned with a full .MPG file that contains both video and audio.

Any hardware MPEG decoder should provide sound playback and have an audio line out signal to connect to your speaker system.

Systems without a hardware MPEG decoder depend upon your PC's audio capabilities. The standard way for a software MPEG player to handle sound is for it to decode the MPEG audio track, convert it dynamically to a .WAV (wave) file, and send it to your operating system's audio subsystem. The result can be either good-quality sound, fragmented sound, or no sound. Fragmented sound can be worse than no sound!

Your MPEG player may allow you to specify that it should generate a lower-quality wave file, by (for example) reducing the sampling rate of the wave file it creates, or by switching from stereo to mono. This may allow you to play continuous, lower-quality sound.

Using the standard audio subsystem to play back the MPEG audio as a wave file usually prevents the simultaneous playback of any other wave file. If your application requires that you be able to play voice-overs, background music, sound effects, etc. without disrupting the MPEG playback audio, then you will probably need a hardware MPEG decoder as well as your standard audio subsystem.

Audio Cabling

Today's PCs can have multiple sound capabilities:

- CD-audio
- MPEG card
- Audio card (e.g., SoundBlaster or business audio chip on the planar)

Usually your main audio subsystem can accept input from other sources. Sometimes this requires external cabling, but many cards have internal sockets for audio in and out. Careful study of your manuals may enable you to make everything work with no external cabling, although you may need to acquire some internal cables from specialist sources or build them yourself.

Lip Synch

One of the hardest things to do well in MPEG playback is to get exact lip

synchronization. Voice-overs and background music can be quite effective even on degraded MPEG systems; however, if the face of the speaker or singer is visible then you really want lip movements to synchronize with sounds.

The slightest degradation in frame rate is immediately obvious when viewing someone speaking. If you need good lip synch, then you should consider only hardware MPEG decoders.

Operating System

If you are using Windows 3.1, then you should find good support for anything you want to do. If you are using any other operating system you need to check carefully for support. In some cases, it is very limited or even non-existent.

*... you cannot assume
your application will
behave identically on all
MPEG systems.*

MCI Command Set

As previously discussed, if you are planning to use a custom application to play your MPEG clips, you need to check that the commands and parameters that your programmers are planning to use are supported by the MPEG system you choose and that they function as your programmers expect them to. Some companies are more straightforward than others in providing information about what is supported, but even with a list of supported functions you can't guarantee that an existing program will function as expected on a new MPEG system. The only sure way to know is to run some tests.

Here are some examples of what can go wrong:

- *Skipping from one frame to another, i.e., jumping within the stream*—Some cards handle this well, automatically displaying the nearest I-frame (complete frame) when you jump. Others simply display the chromakey color if you have not jumped to an I-frame. Others do not refresh the image buffer until a PLAY command, so effects such as forward and backward jog are not possible.

- *Looping, e.g., in an attract loop*—Not all cards are able to loop a PLAY sequence without flashing either black or the chromakey color at the end of each loop. This makes attract loops look messy.
- *End of MPEG-sequence frame loss*—Some cards do not play the last few frames of a sequence. If your card does this, then the MPEG sequence must be encoded to add extra dummy frames, so that those are the ones lost and the original last frame plays correctly.
- *Audio noises*—Jumping to a different frame can cause a crack or click in the audio when you restart at the new position. Avoid this by lowering the audio level prior to the jump, then raising it once the new position is located and starts playing. Some cards do not allow software to control audio levels; on the other hand, not all cards crack!
- *Extra Features*—Some cards provide software-controlled effects. For example, the image space may be tiled to hold 2x2 images or 4x4 images, similar to a video wall. Although these features may enhance your application, they will tie it down to a single card.
- *New drivers*—Problems such as described above occur not only between different MPEG systems, but also when you get new drivers for your current MPEG system. Hopefully these new drivers will provide some benefit in terms of bug fixes, new function, or performance enhancements, but experience shows that they sometimes change some of the detailed behavior of the MPEG system as well.

It should be obvious by now that you cannot assume your application will behave identically on all MPEG systems. Whenever you start using a new MPEG system, you are likely to encounter problems and idiosyncrasies. You need to allow time for testing on each target MPEG system, plus allow some contingency for programming adjustments and for the extra development time you will need to handle these issues.

Controls

As well as being able to play an MPEG clip, a good MPEG system will also enable you to control various playback characteristics such as brightness, contrast,

saturation, gamma, and audio volume. Control should be available both through configuration options (via utilities or .INI files) and through programming, so that you can dynamically handle variations in the quality of different clips.

Controls are especially important when the MPEG clips are created by different encoding systems, resulting in very different output. Consider, for example, luminance. CCIR601 (the TV industry standard) specifies that luminance varies from 16 to 235. But some encoding and decoding systems mistakenly generate values between 0 and 255. This means that the same clip played back on two different systems appears much brighter on one than the other. The ability to control the picture quality at playback time is therefore critical to receiving the best results.

Multiple Tasks

Even if you have fixed everything else and are receiving perfect MPEG playback (or whatever quality you need for your purposes), things can still go wrong in a live environment. If the MPEG application has to coexist with other tasks on your system, then there is no guarantee that the MPEG playback quality will be maintained. The flow of data to the MPEG window depends upon countless system components, and overloading any single component can stop your MPEG dead. A good MPEG system maintains the audio playback, but drops video frames to keep its resource utilization in line with what is available.

You have a much higher chance of success if you have dedicated MPEG decoding hardware in your system, but even this depends upon receiving a steady flow of data from the device where the MPEG file is stored.

Even the best system cannot cope when all system resources are being used for other purposes. You need to test your MPEG application in the live environment and observe how robust it is when the PC is busy doing other things.

The application program itself must also be carefully designed and tested. Often, once it has started the MPEG playback, it sits in a tight loop, waiting for the clip to finish or for some other event to occur. Poor coding of this loop can result in an overutilization of resources, which can affect the MPEG playback quality.

MPEG-2 Versus MPEG-4

The appearance of MPEG-2 decoders in the consumer marketplace has caused some people to think that MPEG-1 is nearing the end of its life. I do not believe that this is true. MPEG-2 is not a replacement for MPEG-1—they are complementary technologies, designed for different marketplaces. MPEG-2 is a higher bit-rate system that is targeted more at digital television. For most purposes, MPEG-1 is likely to remain the primary high-quality digital video technology on the PC for many years.

MPEG-2 is not a replacement for MPEG-1—they are complementary technologies . . .

MPEG-3 was supposed to offer even higher quality, but was encompassed by MPEG-2 and no longer exists.

MPEG-4, on the other hand, has been focusing on improved coding efficiency to effectively transmit audio/visual data on low-bandwidth channels. This, of course, will also mean more efficient data storage. Most of the work so far has been targeted at finding better video and audio compression techniques to try to achieve a tenfold improvement on current techniques.

But that is not the only new or improved functionality that the current MPEG-4 work addresses. Two of the principal application areas under consideration are:

- Advanced conversational services, such as higher-quality public switched telephone networks (PSTN) video phones and mobile audio/visual terminals

- Interactive TV, including audio/visual databases (e.g., web servers), networked video games, and virtual reality—featuring the ability to interact with audio/visual objects, not just video frames

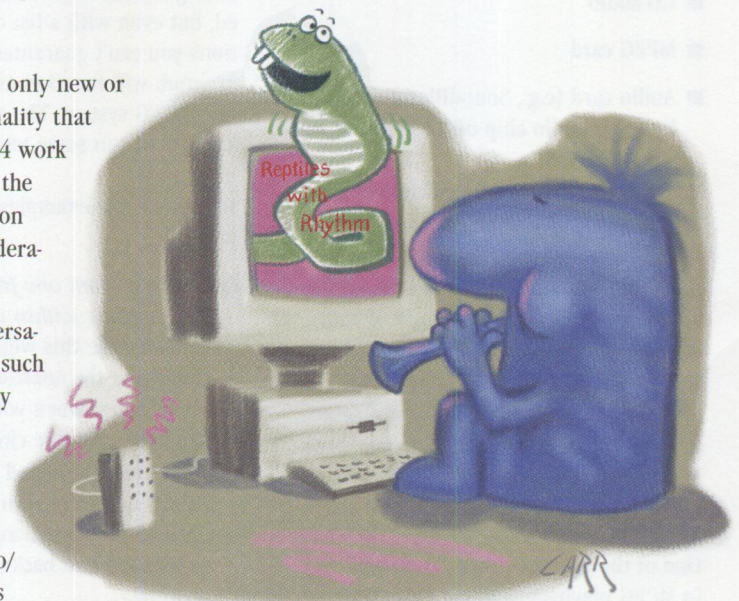
Major enabling work is going on in the areas of the MPEG-4 Syntactic Definition Language (MSDL), Universal Access, and Synthetic/Natural Hybrid Coding. MPEG-4 may prove very interesting when it arrives, but since it is not scheduled to become a standard until the end of 1998, it is not currently relevant to anyone planning to use MPEG-1.

Your To-Do List

With the current state of the art, you must:

- Understand and specify your requirements
- Find an MPEG system that does what you want
- Thoroughly test your potential solution
- Not assume you can switch to another MPEG system without careful selection and testing

But don't be put off. MPEG-1 is excellent, and there are many excellent MPEG systems on the market today. We are still in the early days of MPEG; as MPEG-1 matures many of the current problems will be resolved, and switching between MPEG systems will become less of an issue.



Appendix—Quality of MPEG Clips

The steps to create a high-quality MPEG clip are:

- Choose suitable material. Zooms, pans, and flashing lights are all harder to handle; shots of TVs or computer screens can look terrible unless you use the right synchronization hardware on the camera; and monochrome backgrounds may be boring, but they can result in higher-quality digital videos. If you can affect the shooting of the original material, then get a specialist to help advise the filmmaker about how to shoot material that will digitize well.
- Get a high-quality original. Tape standards include:
 - Betacam SP (the best)
 - Betacam
 - Hi-8
 - S-VHS
 - 8-millimeter
 - Beta
 - VHS (the worst)

This is a rough list and will depend upon many factors, including the quality of your equipment.

Professionally produced material will be better than stuff you copied from the TV. If you have a copy of a copy, then don't waste your time!

- Set up your capture facility. You can purchase an MPEG capture system

(costs range from a couple of thousand dollars up to tens of thousands of dollars), or you can use a bureau which may have hundreds of thousands of dollars worth of equipment. There are two types of facilities:

- A *real-time facility* set up to capture the video as it is played—i.e., it has to do all its processing for each frame in 1/25 of a second, before the next one arrives.
- An *asymmetric facility*, which normally requires specialist hardware so that the PC can control the video source and read one frame at a time. This type of facility gives you better quality because it can spend as long as it likes on each frame.

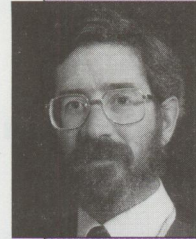
- Perform the capture. Apply parameters and filters to the process to ensure the best results for the quality of the input material and the intended use.

At the moment, MPEG capture is still an art rather than a science, and the expertise of the personnel performing the capture will affect the quality of the resulting MPEG clip. This is another reason for using a bureau if you need very high quality.

Keep in mind that only *one* weak link in the chain will ruin all your work and leave you with low-quality digital video!

Acknowledgments

Many thanks to Andy Burns and Neil Galton for their contributions to this article.



John Sweeney is an information technology consultant within IBM's Information Access, Cross-Industry, Nottingham, England. He performs technical consulting on multi-

media and internet technologies, supporting the marketing and implementation of major projects. Since joining IBM in 1971, he has focused primarily on end-user systems. John has an MA in Mathematics and Computer Science from Cambridge University. He is also a specialist in the fields of panelology, metagrobology, and the Morris. His Internet userid is john_sweeney@uk.ibm.com.

WORLD'S MPEG For
1st Microchannel™

Truecolor, full-screen, full-motion
video playback, CD quality sound

HARD Synchronized
WARE Video and Audio

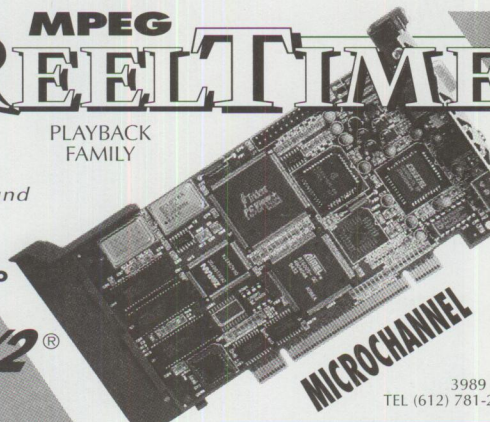
Robust drivers include
Windows™, DOS and... **OS/2**®

All trademarks are the property of their respective owners.

MPEG

REELTIME

PLAYBACK
FAMILY



MICROCHANNEL

ReelTime/M MPEG1 playback board for IBM PS/2® and RS/6000 Microchannel systems

also available **ReelTime/I** MPEG1 playback board for ISA systems

4-ReelTime/P MPEG1 playback board for PCI systems provides 4 independent output channels

VISUAL CIRCUITS

3989 Central Avenue NE, Minneapolis, MN 55421
TEL (612) 781-2186 - FAX (612) 789-6905 - <http://www.vcircuits.com>

Circle #5 on reader service card

Computer Security and Implementation

The OS/2 operating system and OS/2 network client software provide adequate security mechanisms if they are implemented properly. This article highlights these mechanisms and shows their strengths and weaknesses, plus presents information from an OS/2 perspective about general security precautions.

Why bother with security for a single-user operating system? To answer this question, you must examine the environment where the system is used, what network connections exist for it, and what information is stored on the system.

Some reasons to justify using some type of security mechanism are:

- To keep authorized users from accidentally erasing important data
- To keep crackers¹ from accessing your data
- To keep sightseers from using your machine to access other machines on a network
- To keep practical jokers from modifying your system

Richard Potts
Infrastructure, Inc.
Denton, Texas

Security Considerations

This first section discusses several things you should consider when planning for computer security.

Security Through Obscurity

Early in the computer age, many users felt that the security through obscurity (STO) approach was a valid security mechanism. The premise of STO is that if a person does not know about a resource, he or she will not try to access it.

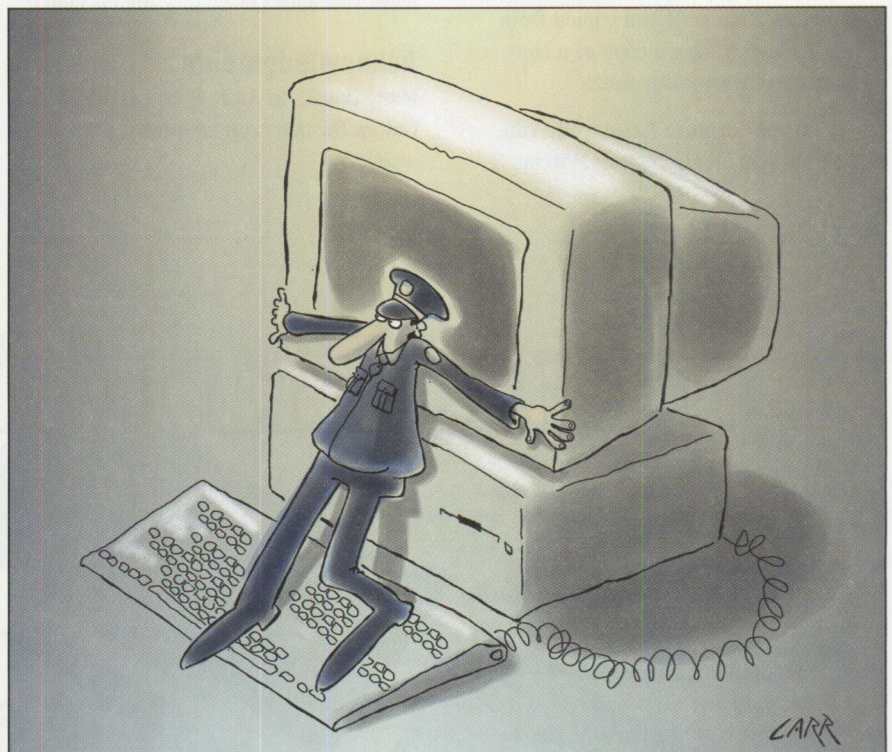
¹A cracker is a computer user who attempts to break into secured computers.

For example, a user may create a hidden directory or a directory name that has a non-keyboard character in it, assuming that other users either will not see the directory or, because the directory contains a non-keyboard character, will not be able to access it. This security measure might pass under DOS, but under OS/2, the drives folder can pull up hidden directories and directories with non-keyboard characters.

Although STO might have worked early on, today most users know too much about computer systems for STO to work. STO mechanisms are faulty because you cannot assume that information is hidden simply because it is not obvious.

DoD Orange Book

In 1983, the U.S. Department of Defense



published the *Department of Defense Trusted Computer Evaluation Criteria*, a specification for computer security. This document became known as the "Orange Book" specification for computer security. It describes the necessary security mechanisms required for a trusted computer system and lists several grades that a system can receive, based upon its compliance with the specifications.

The Orange Book defines the common levels of security found in the marketplace today. One is C2-level security, which consists of the following mechanisms:

- User authentication
- Resource access control
- Auditing capabilities
- Object reuse
- Testing verification
- Documentation

Essentially, the user must use a password to log on to the system before performing any action. All passwords are stored in a system file, and you must protect that file against unauthorized access. Also, actions must be audited and the audit log protected.

All system resources must have access control at either the group or individual level. All unused storage must be empty, not just marked as unused. Verification of the security mechanisms must have taken place, and proper documentation must be available.

With these mechanisms in place and with proper certification, the system complies with the DoD's C2-level security. There are more elements in the specification as the security level increases.

Passwords

The single most important security mechanism is the password. The Computer Emergency Response Team (CERT), a group formed by DARPA (Defense Advanced Research Projects Agency) to study security issues and educate the Internet community, estimates that 80 percent of all network security problems are created by insecure passwords.

Using a password and selecting a good password are not the same thing. If a cracker wants to break your password,

several utilities are available to do it. Most of these utilities work from a dictionary containing common words, proper nouns, and variations on these. Typical dictionary entries and common variations of these entries might be `summer`, `win4ter`, `Richard`, and `Ang3i5e`. Avoid this type of password.

Also, many administrators use an algorithm to generate passwords. While this might sound like a good idea, once a cracker gets one password broken, the rest become easier because a cracker can figure out the algorithm.

The only good passwords are ones that are not generated by a program, do not contain words from a dictionary, and do not contain common proper nouns. Once a good password is selected, it must be kept secret. The more people who know a password, the less secure that password is. Two people knowing a password is one too many.

Use separate passwords for different systems. Then, if a cracker breaks one of

A security exposure that most people overlook is the modem.

your passwords, he or she won't have access to your other systems or accounts.

Security Policy

Every organization should create, maintain, and follow a security policy. Given a set of people and resources, the security policy defines which people can access which resources. The security policy should also contain guidelines for selecting passwords, how often a password must be changed, which resources must be protected and which need not, along with statements outlining which system services, system configurations, and user actions are acceptable and which are not. For instance, trivial file transfer protocol (TFTP) may be banned from your network, while sharing printers is acceptable.

Security With OS/2

OS/2 includes some security mechanisms and provides hooks for third-party

developers to build more robust security software. OS/2's lockup feature, in conjunction with a PC's power-on password, may provide adequate security for your environment. On the other hand, this type of security may not be acceptable within your organization for the reasons described next.

Lockup and Power-On Password

The OS/2 lockup feature provides password protection once the OS/2 operating system is booted and running. Using the desktop settings, you can configure it to lock up the computer during boot time.

The problem with this mechanism is that the password is stored in the OS/2 .INI files. An industrious user can circumvent the password protection by rebuilding the OS/2 .INI files with the MAKEINI.EXE utility provided with OS/2. Rebuilding the .INI files requires rebooting the PC from floppy disk or some type of maintenance partition. To use MAKEINI.EXE, the .INI files cannot be open.

This brings up another issue: The maintenance screen, accessed during OS/2 boot by pressing Alt+F1, which causes a file in \OS2\BOOT called ALT+F1.CMD to execute. By accessing the maintenance screen before the automatic OS/2 lockup is enabled, an unauthorized user can access the system. To eliminate this risk, create a file in \OS2\BOOT called ALT+F1SEC.CMD, with one line: @EXIT. You have now disabled the Alt+F1 maintenance screen, because the ALT+F1SEC.CMD file executes instead.

Because it is possible to keep the maintenance screen from becoming a security issue (so that an intruder is not be able to reboot the computer), the combination of a power-on password for your system plus a lockup password configured to be on at boot time is fairly secure; however, the problem with this mechanism is the ease with which a power-on password can be removed from most PCs.

Modems

A security exposure that most people overlook is the modem. An intruder can call into your modem, and your modem will attach if it is configured to auto-answer. Turning off your modem's auto-answer feature is a simple precaution that is well worth the effort.

Applications

The applications installed on a system present another major security issue, as many newer applications contain an element of network computing and may present an opportunity for a hacker to access your system and resources. Make sure that an application is not leaving a hole in your security.

A classic example of this problem is the much-publicized Internet worm incident. The Internet worm came into UNIX-based systems through a hole in the sendmail (e-mail) software. Most system administrators thought their systems were secure, but they obviously were not. The cracker exploited users' false sense of security about the sendmail application. This example illustrates the point that your system cannot be secure if you do not understand what is running on it.

Removable Media

As we all know, backing up data is very important, not only in case of accidental erasure, but also in case of *intentional* erasure by someone who has broken your security mechanisms. Once you've backed up your data, physically secure the storage media to ensure your backups are safe.

Security Enabling Services

IBM recently delivered the OS/2 security enabling services (SES). These services are hooks into OS/2 that allow third-party developers to work *with* OS/2 in creating strong security software. Although the SES package itself does not secure your computer, it enables your system to be secured with the addition of third-party software.

Third-Party Software

At this time, I am aware of only one security software package that uses the SES hooks. This software is called WATCHDOG PC Data Security for OS/2, from Fischer International. There is another package from SafetyNet called StopLight for OS/2; however, it does not use the SES hooks.

A list of security packages for OS/2 has been compiled and posted on the World-Wide Web (WWW) at <http://www.mfi.com/os2dev/cgi-bin/os2quer?prodcat=Security>.

OS/2 and Network Security

Most security breaches occur on a network. The network allows a cracker

anywhere in the world to gain access to your computer. If your system is not attached to a network, obviously you do not need to be concerned.

The following network clients for OS/2 contain several elements that can help you implement good security mechanisms.

LAN Server

LAN Server offers several mechanisms to provide strong security, including access control lists and password protection, as well as other built-in security features. However, some of the features that add to its robust nature and ease of use can also present security risks if used unwisely.

Default userid and password—Be sure to remove the default userid and password from User Profile Manager (UPM), because they are initially set up at an administrator level. Administrator-level authority is the highest level of authority; it can circumvent a user's security precautions. Pick a good password and use the UPM's password expiration feature.

Guest account—Beware of the guest account; it allows a user without an ID on your server to access files. Typically, a guest ID has no password and can pose a large security threat if you are unaware of this feature.

Peer services—If you are using peer services, be aware that neither user-level nor system-level security provides great security. User-level security requires that each authorized user enter his or her password at each peer computer, which may not be feasible for large LAN environments or remote LAN environments. The other option, system-level security, requires a common userid and password for all users, which is not secure and should be avoided.

TCP/IP

The TCP/IP client for OS/2 contains several potential security administration problems. The TCP/IP software actually stores the password for TELNET in plain text in the CONFIG.SYS file. The passwords for FTP are stored in plain text in the \ETC\TRUSERS file. Similarly, the passwords for REXEC are kept in plain text in the \ETC\NETRC file.

Do not designate these files as shared files, because sharing provides a simple way for a cracker to infiltrate your system. Also, make sure that you are not sharing these files with other network servers. For instance, if you share your boot drive with LAN Server, anyone who has read access to this boot drive can view the CONFIG.SYS file to see your TELNET password.

The FTP configuration notebook page allows you to pick one of two methods for creating access permissions: *directory access lists* and *deny directory access lists*. Directory access gives FTP access to only the directories listed, whereas deny directory access gives full access to the system, excluding the directories in the list.

It is best to use directory access. If the FTP server is attached to another server, a cracker can FTP to another server, because you may have unknowingly given access to those resources.

Do not permit TFTP to be on your network, because TFTP does not require a userid or password. If a server on your network is running TFTP, anyone with a TFTP client on your network can read, write, execute, or delete the files on that server. Additionally, due to the complexity involved in properly setting up access, network file system (NFS) can cause security problems.

Several schemes for setting up FTP can be fairly secure. Here is one scheme for anonymous access: Keep write access to a minimum; set up one specific directory where write access is possible. Keep read access to a minimum, possibly all within one directory tree. It may take more time to set up access to specific directories, but the effort is worthwhile. The directory structure might look like this:

```
\FTP\PUB\OUTGOING (read access only)
\FTP\PUB\INCOMING (write and read access)
```

LAN Distance

The LAN Distance product contains several security mechanisms that, when turned on, provide a broad range of security: user authentication, callback, workstation address identification, logon time

intervals, and passphrase. The key is that the security must be *turned on*.

The easiest mechanism to use is *user authentication*, which requires that all users log on to the LAN Distance Connection Server and pass authentication before traffic is allowed.

Callback means that, when a remote connection is started, the remote workstation dials into the connection server, then hangs up. The server then initiates a call back to the remote workstation. Several configurable options make this feature very robust.

Workstation address identification refers to a specific LAN Distance logical adapter network address stored on both the remote workstation and the connection server. These addresses must match, and, again, there are configuration options that make this mechanism versatile.

You can configure the connection server to allow a user to access the connection server only during specific *logon time intervals*.

The *passphrase* mechanism contains several configurable options that ensure better security. The passphrase can have up to 32 characters, and a user passphrase history can be saved to ensure that a user selects a new passphrase instead of reusing an old one. Additionally, you can configure a passphrase maximum age, minimum age, minimum length, and maximum failed logon attempts.

Lotus Notes

Lotus Notes provides a different mechanism than the standard userid and password combination. An ID file is similar to a key; it is unique to each user and is required to gain access. The ID file is created by the ID certifier and must be present on the workstation where the user wants to log on. Provided that the feature is turned on, there is an encrypted password associated with each user.

Security within Lotus Notes includes database access control lists, mail encryption, user signatures, and time-out. The concept of database access control lists is similar to other access control lists such as those found in LAN Server. In Lotus Notes, the access control lists specify who

is eligible to read, manage, and update a Lotus Notes database.

Encryption allows a user to encrypt a mail message that can be decrypted only by the recipient. User signatures ensure that the mail message sender is truly the author. The time-out feature causes an inactive Lotus Notes logon to be logged off. You can make the time-out feature the default by setting the `AutoLogoffMinutes` parameter in the default `NOTES.INI` file prior to installing over the LAN.

Additional Security and Information

Although screening routers, firewalls, and proxy servers are not specific to OS/2 networks, they are mentioned here because of their increasing popularity and the strong security features they can provide.

Screening Routers

A screening router works at the network and transport layers to let you block certain traffic on the network: specific protocols, source addresses, destination addresses, and packet-control fields. This lets you control the type of traffic on the network, therefore controlling the applications that can exist on the network.

Firewalls

A firewall protects one network from another and is often implemented as a security mechanism between the Internet and a site's internal network. A firewall is similar to the screening router, except the firewall works at the application layer. Specific application-level network traffic can be monitored and rejected.

Proxy Servers

Proxy servers offer security mechanisms that allow you to further limit and thereby secure your sites. The proxy server exists between client workstations and the outside network and simulates both a client and a server. It acts as a client when a client puts a request onto the network; the request is sent to the proxy, which then passes the request out to the network. The proxy acts as a server by receiving incoming requests and passing them on to a client.

All traffic goes through the proxy, which can deny requests in either direction. For sanctioned applications, the proxy simply acts as a relay.

Security Information Sources

Following are some additional sources of information on security:

TCP/IP 2.0 for OS/2 Installation and Interoperability, ITSC Redbook, GG24-3531.

IBM LAN Distance 1.1 Configuration and Customization Guide, ITSC Redbook, GG24-4158.

Siyan, Karanjit, *Internet Firewalls and Network Security*, New Riders Publishing, Indianapolis, Indiana, 1995.

Raymond, Eric, *The New Hackers Dictionary*, The MIT Press, Cambridge, Massachusetts, 1993.

OS/2 Security Enabling Services (SES), ITSC Redbook, SG24-4568.

Rainbow Series—U.S. Government Printing Office (includes the *Department of Defense Orange Book—Trusted Computer System Evaluation Criteria*)

Lotus Notes Database: Lotus Notes Help—Security Section

`security.faq`—maintained by Alec Muffet—can be found at <http://www.cis.ohio-state.edu/hypertext/faq/usenet/security-faq/faq.html>.



Richard Potts is an application development consultant with Infrastructure Incorporated (II), where he helps clients design, develop, and tune their object-oriented business applica-

tions. Richard came to II from the IBM Personal Systems Competency Center in Roanoke, Texas. He holds a BS degree in Computer Science Engineering from the University of Texas at Arlington. He can be reached via Internet at pottsar@airmail.net.

Installing FixPaks via CID

This article assumes that you can do a standard manual installation of a FixPak on your system. Ensure you can do this before you set up a complex configuration/installation/distribution (CID) installation and then find that the FixPak does not work on your system.

Kicker diskettes are required not only when manually installing OS/2 FixPaks and LAN Server (LS) Customer Service Diskettes (called FixPaks here), but also when installing these FixPaks via IBM's CID process.

The installation kicker diskettes are different from the FixPak data diskettes. The same kicker diskettes can be used to install several different FixPaks, including all current OS/2 and LAN Server FixPaks and certain other products such as Distributed Console Access Facility (DCAF) and NetView.

The following information describes how to use these kicker diskettes to install the FixPaks.

Setting Up a CID Server

Step 1. On the CID server, create a new directory for the FixPak. This example names the directory `FIXPAKS`.

Step 2. Use the `XCOPY` command with the `/S` parameter to copy all of the FixPak data diskettes (for either OS/2 or LAN Server) into the `FIXPAKS` directory.

Step 3. `XCOPY` the single LS kicker diskette into the `FIXPAKS` directory. (If you do not have the LS kicker diskette, you can `XCOPY` the second OS/2 FixPak kicker diskette into the `FIXPAKS` directory.)

Creating a Response File

Step 4. To install an OS/2 FixPak, create an `FSERVICE` response file, using the

contents of `RESPONSE.FIL` in the `FIXPAKS` directory as the basis (see Figure 1).

When modifying `RESPONSE.FIL`, be very careful not to re-sequence any of the statements. Incorrect sequencing is a common cause of problems.

Use the `:SYSLEVEL` statement to service a product that resides in a specific partition. For example, `:SYSLEVEL D:\OS2\INSTALL\SYSLEVEL.OS2` will only service the OS/2 product that already resides in the `D:` partition.

There is no need to change the `:SOURCE` statement, because the `/S:` command line parameter overrides the `:SOURCE` keyword in the response file. Also, the `/L1:` command line parameter overrides the `:LOGFILE` keyword.

You can remove or remark out the `:LOGFILE` statement, but the `:SOURCE` statement must always be present, even if it is not used.

The OS/2 response file does not assign drive letters to `:SYSLEVEL` and `:ARCHIVE`. This will cause the `FSERVICE.EXE` file to search all partitions to locate the OS/2 product and update all partitions where it finds the correct OS/2 version for the FixPak.

```
:LOGFILE \OS2\INSTALL\SERVICE.LOG
:FLAGS REPLACE_PROTECTED REPLACE_NEWER
:SOURCE A:\
:SERVICE
:SYSLEVEL \OS2\INSTALL\SYSLEVEL.OS2
:ARCHIVE \ARCHIVE
```

Figure 1. `RESPONSE.FIL` for Installing OS/2 FixPak

Parameter	Description
<code>/S:</code>	the source path pointing to the <code>FIXPAKS</code> directory on the CID server
<code>/R:</code>	drive\path\filename of the response file
<code>/L1:</code>	drive\path\filename of the log file
<code>/T:</code>	(optional) drive\path to use if booting from the <code>SEMAINT</code> environment; you should set it to the same parameter value used in the <code>SEMAINT</code> call
<code>/CID</code>	(optional) to be used for unattended installation
<code>/SF:</code>	do not use for CID installs; use only if source path is removable media

Figure 2. `FSERVICE` Parameters

Executing FSERVICE

Step 5. Execute `FSERVICE` to install the FixPak. Figure 2 lists the available parameters.

The example in Figure 3 shows the `FSERVICE.EXE` file being invoked. Depending upon what the response file contains, use the call in Figure 3 to install a FixPak, back out a FixPak, or commit a FixPak (so the FixPak can't be backed out once it is installed). All code in Figure 3 must be on a single command line.

The /T: Parameter

If you boot OS/2 from a hard drive and the environment variable `REMOTE_INSTALL_STATE` is set to 0 or 1, you must use the `/T:` parameter. If you boot from a hard drive and the `REMOTE_INSTALL_STATE` environment variable is not defined, you should *not* use the `/T:` parameter. If you boot from diskette, `/T:` is ignored.

`FSERVICE` uses the `/T:` parameter to decide whether a file should be updated. If the first part of the path to a file is equal to the `/T:` parameter, then `FSERVICE` will not update that file, because files in the maintenance directory (created by `SEMAINT`) should not be updated.

Whether to Run SEMAINT

The best and safest way to install an OS/2 FixPak is in a maintenance mode. Although you can install an OS/2 FixPak while booted in normal OS/2 PM mode, problems may occur.

There are two standard ways to enter maintenance mode—you can use `SEMAINT` to generate a maintenance boot environment, or you can boot from diskette. If you are going to install several fixes at one time, using `SEMAINT` will save time because it requires fewer reboots and no processing of locked files.

To use `SEMAINT`, you must make changes to the way you invoke `FSERVICE`. `FSERVICE` cleans up a maintenance directory created by `SEMAINT` if you use the `/T:` parameter and boot the system from a hard drive.

You should only use the `/T:` parameter in the first `FSERVICE` call you make. Either remove or do not use the `/T:` parameter in all other calls to `FSERVICE`, because the `/T:` parameter may cause a file that was already updated to be restored to the original level.

If you use `SEMAINT`, do not use the `/S2:` parameter. Use the `/S2:` parameter only when installing OS/2 ServicePak XR_6200 or XR_6300.

Removing an Installed FixPak

To back out a previously installed FixPak, use the response file shown in Figure 4.

When backing out a FixPak, the `FSERVICE` parameter `/S:` should point to the

`FIXPAKS` directory on the CID server, because `FSERVICE` needs to find the product information files for the FixPak that it will back out.

Troubleshooting

If problems occur during a FixPak installation, it may be necessary to delete or rename the `LOGF0000.***` and the

`LOGFSTART.***` files from the target client system and to remove the `ARCHIVE` directory. These steps are frequently necessary when a previous FixPak installation started but did not finish.

The current OS/2 FixPak boot diskettes are based upon OS/2 Warp code. To install the FixPak, you can boot the

```
x:\csd\fixpaks\fservice
/s:x:\csd\fixpaks
/r:x:\csd\fixpaks\response.fil
/cid
/t:c:\service
/l1:x:\log\fixpaks\service.log
```

Figure 3. Invoking `FSERVICE`

```
:LOGFILE C:\OS2\INSTALL\SERVICE.LOG
:TARGET ARCHIVE
:BACKOUT
:SYSLEVEL C:\OS2\INSTALL\SYSLEVEL.OS2
```

Figure 4. Response File for Backing Out a FixPak

```
TargetDir = "C:\SERVICE"
CompNameLen = 4

Section Catalog
Begin
  ObjectType = SOFTWARE
  GlobalName = FIXPAK.WARP.17.C.REF.1.0
  Description = WARP FixPak 17 on C:
End

Section Install
Begin
  Program = SA:\IMG\FIXW17\FSERVICE.EXE
  Pargs = "/S:$(SourceDir) /T:$(TargetDir)
  /R:$(SA)\IMG\FIXW17\SERVICEC.RSP /L1:$(LogFile1)"
  SourceDir = SA:\IMG\FIXW17
  LogFile1 = SB:\LOGS\OS2\$(WorkStatName).FIX
End
```

Figure 5. `NvDM/2` Profile Example

```
x.fixpak = 9
x.9.name='WARP FixPak 17'
x.9.statevar = 'CAS_' || x.9.name
x.9.instprog = 'x:\img\fixw17\fservice.exe ',
  '/S:x:\img\fixw17 ',
  '/L1:y:\logs\os2\' || client || '.fix ',
  '/CID ',
  '/R:'
x.9.rspdir = 'x:\img\fixw17'
x.9.default = 'response.fil'
```

Figure 6. `LCU REXX` Example

system from either OS/2 2.11 or OS/2 Warp; however, a problem may occur if you boot the system with OS/2 2.11 when you apply a FixPak. The problem occurs if FSERVICE finds the OS/2 Warp SHPIINST.DLL file from the FixPak boot diskette. To fix this problem, replace the SHPIINST.DLL file in the FixPak directory with an OS/2 2.11 version.

When installing a FixPak for LS 4.0 or some other product, you may need to change the response file keyword :SYSLEVEL to point to the correct SYSLEVEL.*** file for that product.

You may need to remove or remark out the response file keyword :ARCHIVE if the FixPak has archiving turned off. (This is the case for many LS FixPaks.) You can archive to a LAN drive, and you can archive files for different products to the same directory.

You must always use the same archive directory for the same product. If you install an OS/2 FixPak and a previous OS/2 FixPak was installed on the system, you must use the same archive directory you used before. You can change the archive directory with a special response file keyword, :REDIRECT.

To update several products, you may need several pairs of :SERVICE and :SYSLEVEL parameters for each product.

Using NvDM/2 and LCU

If you are using NetView Distribution Manager for OS/2 (NvDM/2), IBM's premier CID process, refer to Figure 5 for a sample profile. If you are using LAN CID Utility (LCU), the standard CID process, see the REXX command file product definition example in Figure 6.

—Stephen K. Cunningham,
IBM Corporation, Austin, Texas

Personal Systems REPRINTS

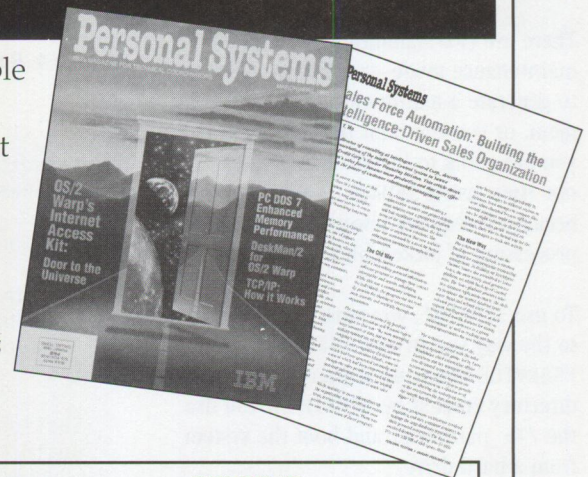
Reprints of any article in PERSONAL SYSTEMS are available from **Reprint Management Services™**, the exclusive reprint agent. High-quality editorial and advertisement reprints can help your company in many ways:

- Increased **exposure** for your product or service
- Credible, believable information that customers **trust**
- Great **sales** tool for trade shows, mailings, and conventions
- Powerful **educational resource** for customers and employees

We tailor the reprint layout and design to your needs.

Reprints are more affordable than you think. Call for quotes on quantities as small as 100 copies.

Your best sales tool is only a phone call away!



Duane Dagen
Reprint Operations Specialist
For additional information
please call (717) 560-2001

**REPRINT
MANAGEMENT
SERVICES™**

147 West Airport Road, Box 5363
Lancaster, PA 17606-5363
Fax (717) 560-2063

Meeting Your Users' IS Needs

Sometimes it's the little things that go a long way toward keeping your end users happy and productive. These "little solutions" take little time on the part of information systems (IS) department personnel but can reap rewards in time saved by having fewer complaints from users and fewer "fires" to put out.

To keep your end users happy campers, your IS team should take the following steps:

- Ensure there are enough people to support your users. Commit more resource where needed.
- Maintain a current list of phone numbers for support contacts.
- Back up your production servers daily.
- Train your users to regularly back up any critical data they keep locally.
- Determine worst-case scenarios and develop contingency plans for solving them.
- Maintain excellent communications with users and management.
- Plan for future needs.

Sufficient Resource

It can be very difficult for an IS department to balance the workload—maintaining enough personnel to meet critical needs while avoiding significant idle time during normal operations. For example, having one expert on an operating system used by 100 people can overwhelm that person and lead to long waits by users if many users experience problems with that operating system at the same time. However, companies are understandably reluctant to hire a lot of people into their IS departments if many of them are idle during normal business operations.

To resolve this "peak and valley" workload problem, the number of people supporting a particular product can be based

on both the number of people using the product and the extent they use it to perform their jobs. For example, if 100 people use Lotus Notes frequently in their jobs, you might need five people in the IS department who are very knowledgeable about Lotus Notes.

Your IS support personnel should be diversified—that is, they should be familiar with more than one product (for example, an expert in DB2/2 might also know OS/2 very well). A person who provides primary support on one product can provide secondary support on other products; however, you should ensure that your technical support personnel provide primary support for only one product each and secondary support for only a few products each.

Support Contact List

While the IS department will be able to meet most of its users' needs without any outside assistance, occasionally it is required. Maintaining a current list of phone numbers for external support contacts will allow your support personnel to get the outside support they need without spending a significant amount of time trying to locate it.

Production Server Backup

A server machine can fail for many reasons: a hard drive may become damaged, the power in the building may be cut off, a virus may be inadvertently introduced. In the event of a failure, the IS team's objective is to rebuild the server, losing as little data as possible. To do this, have a reliable process to back up data files to tape or some other storage medium.

A good method for backing up a production server begins with making a complete backup of all necessary data files to five tapes (or other storage media). One tape holds the initial backup of the server, and the other four contain daily incremental backups. These daily backups

contain only the file changes made in the server during a business day. On the fifth business day (usually Friday), make a full backup on a new tape. Continue this pattern of making incremental backups for four days and a full backup on the fifth day for three more weeks, then store the full backup on the fourth week as the monthly backup. At this point, you will have nine backup tapes: the initial backup, the four daily incremental backups, the three weekly full backups, and the new monthly backup. You can reuse the four daily and three weekly tapes, ensuring that the data in the server can be restored to its previous state on any day of any week of the last backed up month and any day of the last backed up week. As you make new monthly backups, store them with old monthly backups, keeping all the backups as long as desired (six months, one year, etc.).

You can modify this backup method to reflect the usage of the server. Normally, such backups take place after hours in order to minimize the effects backing up the server have on its performance. If the server data changes a great deal during a day, however, it may be necessary to perform incremental backups more than once a day. Also, if the server is used heavily on the weekends, you may also want to perform incremental backups on those days. To reduce the risk of a catastrophic event such as a fire destroying the server along with all the backups of the server data, you should keep a separate copy of the most recent backup offsite.

Local Backups

In some cases, your users may store critical data on their local machines. As with data residing on a production server, this data needs to be preserved so that it can be recovered in the event that a problem with the user's machine causes lost data.

These data files can be preserved several ways. If the files are small enough, they

can be copied to a diskette. If space is allocated on a production server, users can copy files to specially allocated server drives. Whatever method is used to preserve data files, you should encourage your users to back up their files as often as needed to reduce the amount of data that is lost in the event of a machine failure. Try to make it easier for users to back up their files by simplifying the actions they need to take to back up critical data. Many software applications prompt users to back up the data files before exiting. Or you can simplify local backup by providing your users with a single program that performs all the necessary backup actions.

Contingency Plans

The key to recovering as quickly and as painlessly as possible from a catastrophe is to anticipate the catastrophe and have an action plan in place. You need to develop plans that deal with every potential problem and incorporate those plans into an overall recovery strategy. When unanticipated problems occur (and they will be rare if your contingency plans are thorough), you can add the actions taken to resolve the problem to your overall recovery strategy.

User and Management Communication

Your users are "customers" of the IS department. You need your users' input

to make the environment in which they work best meet their needs and to make them as productive as possible. You must effectively communicate changes in their working environment to them in order to minimize the time they need to adapt to these changes.

During normal business operations, IS department representatives and users should meet periodically to discuss suggestions for improving both the IS department and the users' working environments. At times when the interaction between the IS department and the users occurs frequently (such as during operating system upgrades), these meetings should be held weekly or even daily.

Communication between the IS department and management is equally as important as communication between the IS department and end users. Often, IS department decisions to improve the users' working environment have to be approved by management, and management may have ideas for improving the working environment that they want the IS team to evaluate. As with the users, meetings between the IS department and management should take place periodically during normal business operations and should occur more frequently during times of significant IS changes.

Planning for the Future

Technology has made great strides since the first PC was introduced about 15 years ago. Computers and the software that runs on them have become more powerful. Hard drives store more data at less cost, applications are easier for people to use, and access to information repositories such as the World-Wide Web is rapidly growing. This explosion of technological innovation has helped people to become more productive in their jobs, which has helped the companies they work for to be more successful.

Almost as important as supporting your users on the technology they need for doing their jobs today is determining what technology they will need to do their jobs in the future. This requires understanding what your company wants to do a few years from now in order to determine if actions such as purchasing additional machines, upgrading software, or even redesigning the users' working environment are necessary. By learning about new technologies that are being developed, as well as plans your company has for the future, you can plan the best way to implement the hardware and software changes needed to perform the work today as well as in the future.

*—Kelly Westphal, IBM Corporation
Roanoke, Texas*

Corrective Service Information

Figure 1 shows maintenance release levels for the listed products. This information is effective as of June 1, 1996. CSDs may have been updated since press time.

To order all service packages—except for the OS/2 2.0, OS/2 2.1, OS/2 2.1 for Windows, and OS/2 2.0 Toolkit ServicePaks—call IBM Software Solution Services at (800) 992-4777. For the OS/2 2.0 ServicePak (XR06100), OS/2 2.1 ServicePak (XR06200), OS/2 2.1 for Windows ServicePak (XR06300), or the

IBM Developer's Toolkit for OS/2 2.0 ServicePak (XR06110) on diskettes or CD-ROM, call (800) 494-3044. Most OS/2 service packages are also available electronically from the following sources:

- **OS/2 Bulletin Board Service (BBS):** In Software Library, select Option 2. (Corrective services are also listed under the General category on the IBMLink BBS.) To subscribe to the OS/2 BBS, call (800) 547-1283.
- **IBM Personal Computer Company (PCC) BBS:** Call (919) 517-0001.

Service packages are located in Directory 4.

- **CompuServe:** Download service packages from the IBM OS2 FORUM library (GO IBMOS2 IBM DF2).
- **Internet:** Do an anonymous FTP from ps.boulder.ibm.com at /ps/products/. TCP/IP packages are located at software.watson.ibm.com at pub/tcpip/os2.

—Arnie Johnson and Paul Washington, IBM Corporation, Austin, Texas

Product/Component	Release	CSD Level	PTF Number	Change Date	Comments
OS/2 Standard Edition	1.3	XR05150	XR05150	02-10-93	
OS/2 Extended Edition	1.3	WR05200	WR05200	05-12-93	WR05200 replaces WR05050, which can no longer be ordered on diskette
OS/2	2.0	XR06100	XR06100	09-01-93	XR06100 replaces XR06055.
OS/2 2.10 ServicePak	2.1	XR06200	XR06200	03-01-94	This package is not for OS/2 2.1 for Windows.
OS/2 2.11 for Windows ServicePak	2.11	XR06300	XR06300	05-24-94	
OS/2 Toolkit	2.0	XR06110	XR06110	09-01-93	
	1.3	XR05053	XR05053	03-23-92	
OS/2 LAN Server/Requester ServicePak	2.0	IP06030	IP06030	04-25-93	
OS/2 LAN Server/Requester ServicePak	3.0	IP07060	IP07060	05-10-95	Supersedes IP07045.
IBM LAN Server/Requester OS/2 Warp Connect LS 4.0 ServicePak	4.0	IP08222	IP08222	05-14-96	Supersedes IP08152.
IBM Peer for OS/2 Public FixPak	1.0	IP08185	IP08185	03-21-96	Available electronically only.
OS/2 Extended Services Database Manager ServicePak	1.0	WR06035	WR06035	11-18-93	Supersedes WR06001, WR06002, WR06003, WR06004, WR06014, and WR06015.
DB2/2 ServicePak	1.0	WR07042	WR07042	06-08-95	
DB2/2 FixPak	2.1	WR08090	WR08090	05-06-96	
DDCS/2 ServicePak	2.2	WR07046	WR07046	06-06-95	
	2.0	WR07041	WR07041	02-06-95	
Database Manager DB2/2	1.2	WR07047	WR07047	06-06-95	
Client Application Enabler/2 (CAE/2)	1.2	WR07043	WR07043	06-06-95	
Software Developers Kit/2 (SDK/2)	1.2	WR07048	WR07048	06-06-95	

Figure 1. Maintenance Release Levels (continued on next page)

Product/Component	Release	CSD Level	PTF Number	Change Date	Comments
SDK/Windows FixPak	2.1	WR08081	WR08081	01-30-96	
Extended Services Comm Mgr ServicePak	1.0	WR06025	WR06025	11-29-93	
System Performance Monitor (SPM/2) ServicePak	2.0	WR06075	WR06075	12-10-93	
OS/2 LAN Server Macintosh ServicePak	1.0	IP06200	IP06200	03-13-96	
LAN Distance ServicePak	1.1/1.11	IP08205	IP08205	04-17-96	Supersedes IP08175, which superseded IP07050.
OS/2 Network Transport Services/2 SelectPak	2.20.5/2.20.1 2.20.2	WR07060	WR07060	05-10-95	Must be LAPS 2.11 or above. If not, order WR07045 first.
LAN Server 4.0 MPTS ServicePak	4.0	WR08150	WR08150	10-18-95	
LS 4.0 MPTS Warp Connect/Server ServicePak	1.0	WR08210	WR08210	05-15-95	Supersedes WR08152. Requires UN00067 if using TCP/IP 3.0.
Communications Manager/2 Version 1.01 ServicePak	1.01	WR06050	WR06050	06-11-93	Available only on diskette.
CM/2 Version 1.11 ServicePak	1.11	WR06150	WR06150	05-31-94	Available on diskette and CD-ROM.
DOS	4.0/4.01	UR35284	UR35284	09-26-91	
	5.0	UR37387	UR37387	09-22-92	
C Set/2 Compiler	1.0	CS00050	XR06150	06-29-93	
C Set C++ Compiler	2.0/2.01	CTC0010	XR06190	09-15-94	
C Set C++ Utilities	2.01	CTM0006	XR06196	09-15-94	
	2.0	CTL0007	XR06197	09-15-94	
TCP/IP for OS/2 Warp Connect	3.0	UN00067	UN00067	05-15-96	Requires WR08210 for TCP/IP to function properly.
TCP/IP for OS/2 Base and Application Kit	2.0	UN64092	UN64092	08-24-94	
TCP/IP for OS/2 DOS Access	2.0	UN57546	UN57546	08-24-94	
TCP/IP for OS/2 Extended Networking	2.0	UN60005	UN60005	06-21-94	
TCP/IP for OS/2 Programmer's Toolkit	2.0	UN57887	UN57887	06-21-94	
TCP/IP for OS/2 Domain Name Server	2.0	UN60004	UN60004	08-24-94	
TCP/IP for OS/2 Network File System	2.0	UN57064	UN57064	06-21-94	
TCP/IP for OS/2 X-Windows Server	2.0	UN68122	UN68122	01-20-95	
TCP/IP for OS/2 X-Windows Client	2.0	UN59374	UN59374	08-24-94	

Figure 1. Maintenance Release Levels

CSD Naming Conventions

In the past, CSDs were known as ServicePaks and FixPaks. ServicePaks were more complete, cumulative, regression-tested packages. They were large in size and generally available both in diskette and electronically. FixPaks were smaller and more component-oriented than ServicePaks and were generally available electronically only.

All future LAN Server service will adopt the same naming convention for service that is used by OS/2, DB2/2, CM/2, and all the IBM Personal Software Product (PSP) line of products. *FixPak* will be used for all future LS and PSP service offerings; some will be Public FixPaks and some will be Controlled FixPaks. *Public FixPaks* may be total cumulative service

available both in diskette and electronically; or they may be available electronically only. *Controlled FixPaks* will not be generally available until they complete testing and will be available only by contacting Software Solutions Services.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

Advanced Peer-to-Peer Networking, AIX, AIX/6000, APPN, Aptiva, AS/400, BookManager, BookMaster, Common User Access, Communications Manager, C Set ++, CUA, DATABASE 2, DATABASE 2 OS/400, DB2, DB2/2, DB2/400, DB2/6000, Distributed Database Connection Services/2, DProp, DRDA, DSOM, DualStor, IBM, IBMLink, IIN, LAN Distance, LANStreamer, Micro Channel, MVS, MVS/OE, NetFinity, NetView, OS/2, OS/400, Person to Person, PowerPC, Presentation Manager, PS/2, RISC System/6000, ServicePak, SOM, SOMobjects, System/390, TalkLink, ThinkPad, Ultimea, ValuePoint, VisualAge, VisualGen, VM, VoiceType, WebExplorer, WIN-OS/2, Workplace Shell, XGA

Windows is a trademark of Microsoft Corporation.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names may be trademarks or service marks of others.

These back issues of *Personal Systems* are available to provide valuable information. Indicate the desired quantity for the issues you want to order and complete the information on the following page.

May/June 1996

Managing a Systems Management Merger
 IBM's OS/2 Warp Server is HOT . . . HOT . . .HOT!
 What's New?
 Avarice Preview: Software Development as an Audience Participation Sport
 PartitionMagic 2.0—Now Even More Magical
 SystemView in OS/2 Warp Server
 Implementing PC SystemView (NetFinity) in Real-World Environments
 Lotus Notes-Based Meetings
 OS/2 Warp Server: An Architectural Primer
 OS/2 Warp Server: An Installation Primer
 OS/2 Warp Server Performance Highlights and Tuning Tips
 TCP/IP CID Client/Server Setup Procedure

March/April 1996

What's New?
 Seton Hall Students Lead the Way From the Wireless to the Real World
 "Out, Damn Spot!" or How to Rid Your OS/2 Desktop of Pesky Programs
 Why SOM?
 IBM System Object Model—The Wave of the Future (and Now!)
 Building SOM Objects with Native C++
 Distributing Objects with DSOM
 Using OpenDoc and SOM in Application Development
 Enabling Industrial-Strength OO Applications with SOM and CORBAServices
 SOM Language Neutrality: A VisualAge for Smalltalk Perspective
 SOM Language Neutrality: An OO COBOL Perspective

January/February 1996

What's New?
 Tape Backup Products for OS/2
 Fault Tolerance for LAN Server
 Getting Together with cc:Mail
 Sales Force Automation: Building the Intelligence-Driven Sales Organization
 The New Mercantilism
 Designing Lotus Notes Applications That Perform
 Designing a Scalable Lotus Notes Workflow Application
 Lotus Notes for AIX in a Personal Systems Environment
 New Administrative Features and Enhancements in Lotus Notes Release 4
 MQSeries link for Lotus Notes
 Getting Warped and Connected Too!—Part Two

November/December 1995

What's New?
 Road Trip! Shopping the Internet
 Command-Line Commando
 Getting Warped and Connected Too!
 Infrared: LANs Without Wires
 Security and Auditing in IBM LAN Server
 Multi-User Performance Testing in a Client/Server Environment
 DCE Cell Performance: High Water Marks
 Plug and Play in PC DOS 7

September/October 1995

What's New for OS/2?
 Mesa 2 for OS/2
 Manage Your Files with FileStar/2 for OS/2
 PartitionMagic for OS/2

Managing LAN Server Home Directories
 IBM DualStor for OS/2
 Human-Computer Interaction Overview
 User Interface 2000
 IBM's Strategy for OS/2 Platform Products Fix Support
 Road Trip! Back to School
 TalkLink Gets a Facelift
 OpenDoc and Human-Computer Interaction
 Supporting HCI Technologies in Applications
 An Introduction to Speech Recognition with OS/2
 Intelligent Agents: A Primer
 CID Installation of OS/2 and Its Platform Applications
 Creating Your Own INF Hyperlinked Files

July/ August 1995

What's New for OS/2?
 The Soap Box Derby
 Easily Load and Lock Desktops
 Road Trip! Cruisin' to the Olympics
 DB2 for OS/2 V2.1: The Next Generation
 OS/2 Victories from the Data Management Front Lines
 Voting Kiosks: The Future of Electronic Elections
 Performance Enhancements in DB2 for OS/2 V2.1
 DB2 for OS/2 Administrative Tools
 Database Recovery with DB2 for OS/2
 Getting Object-Oriented with DB2 for OS/2 V2.1
 Enhanced SQL in DB2 for OS/2 V2.1
 Enterprisewide Connectivity Using DB2 Visualizer Development
 Performance: DCE RPC as a DB2 for OS/2 and DB2 for AIX Transport
 Remote Program Load of OS/2 Warp from NetWare 3.12

May/June 1995

What's New for OS/2?
 Thanks for the Memory
 Road Trip! Disney on the Internet
 Apache Students Use the Power of the Pen (Light Pen)
 Visualizer: The Conversion Continues
 The Internet: A New Dimension?
 IBM LAN Doctor Services
 Borland C++ 2.0 Brings OWL to the OS/2 Presentation Manager
 LAN Server Logon Internals
 LAN Server 4.0 Performance, Capacity Enhancements, and Tuning Tips
 OS/2 Warp for Developing PC Games
 Controlling the OS/2 Desktop From a File Server
 Jump-Start Your PC with Component Upgrades

March/April 1995

What's New for OS/2?
 Mesa 2: Gaining the Competitive Edge with OS/2
 Managing the Workplace Shell with DeskMan/2
 Circus du COMDEX: The Running of the Geeks
 Road Trip! Touring the Side Roads of the Internet
 What's New in PC DOS 7
 OS/2 Boot and Recovery Options
 TCP/IP: How It Works
 A Guide to OS/2 Warp's Internet Access Kit
 CID Installation of OS/2 Warp and LAPS
 Wrapping Up an OO Experience

January/February 1995

Technical Connection Personal Software Is the Answer!

Visualizer, DB2, and You—An End-User's Perspective
 Insiders' Software Unveiled
 Need a Specialist for Your LAN Server 4.0?
 One-Stop Shopping
 OS/2 Warp
 OS/2 for SMP
 Multimedia File I/O Services
 Need a Fix?
 IBM LAN Server 4.0: New Features and Comparisons with NetWare
 IBM DCE Heterogeneous Enterprise Performance

November/December 1994

Evolution, Not Revolution—Pen Computing Comes of Age
 Handwriting Recognition: The State of the Art
 Pen Digitizing Hardware
 It's HAPENing!
 Bill Carr: Fastest Draw in the West
 Work Management in the Field
 Communicating Without Wires: IBM's Mobile Communications Module
 Tomorrow's Networking Today—from IBM's Personal Systems Competency Center

Customers Speak Out About Consult Line
 New DeScribe 5.0—Leader of the Pack
 Super-Fast PenDOS
 Pen for OS/2
 A Development Environment for Pen-Centric Applications
 Writing DOS Installation Programs for Selective Boot Systems
 OS/2 for PowerPC: Transforming Architecture into Implementation

September/October 1994

"Sneaker Net" or Systems Management?
 Like Father, Like Son
 The Book Shelf
 Cajun Electric Cooks Up OS/2 GUI with VisPro/REXX!
 Application Development by Program Integration
 IBM REXX for NetWare
 GammaTech REXX SuperSet/2—Give Your REXX Programs the Power of C
 BranchCard: A Viable Option to Stand-Alone Hubs
 A Hands-On Primer for REXX
 Visual REXX Development Environments
 CID Installation of OS/2 2.11 and LAPS
 Upgrading from Microsoft LAN Manager to IBM LAN Server 3.0

Send this form with a check or money order, payable to **NCM Enterprise**, to: NCM Enterprise, P.O. Box 165447, Irving, TX 75016-9939. You can also fax both pages of this form to **(214) 518-2507** (please include VISA / MasterCard / AmEx / Diners number and expiration date), or call **(800) 678-8014**. All orders must be prepaid. Checks must be in U.S. dollars.

BACK ISSUE ORDER FORM

NAME _____

COMPANY _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

TELEPHONE (_____) _____

Price is \$12.00 per issue, plus \$3.95 shipping & handling per copy. Overseas orders add \$9.95 shipping & handling per copy.

Texas residents add applicable sales tax.

I have enclosed a: Check Money order
 Charge to: VISA MasterCard AmEx Diners

CREDIT CARD NUMBER _____

SIGNATURE _____ EXPIRES _____

IBM believes the statements contained herein are accurate as of the date of publication of this document. However, IBM hereby disclaims all warranties as to materials and workmanship, either expressed or implied, including without limitation any implied warranty of merchantability or fitness for a particular purpose. In no event will IBM be liable to you for any damages, including any lost profits, lost savings, or other incidental or consequential damage arising out of the use or inability to use any information provided through this service even if IBM has been advised of the possibility of such damages, or for any claim by any other party.

Some states do not allow the limitation or exclusion of liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you.

This publication could contain technical inaccuracies or typographical errors. Also, illustrations contained herein may show prototype equipment. Your system configuration may differ slightly.

IBM has tested the programs contained in this publication. However, IBM does not guarantee that the programs contain no errors.

This information is not intended to be a statement of direction or an assertion of future action. IBM expressly reserves the right to change or withdraw current products that may or may not have the same characteristics or codes listed in this publication. Should IBM modify its products in a way that may affect the information contained in this publication, IBM assumes no obligation whatever to inform any user of the modification.

Some of the information in this magazine concerns future products or future releases of products currently commercially available. The description and discussion of IBM's future products, performance, functions, and availability are based upon IBM's current intent and are subject to change.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not imply giving license to these patents.

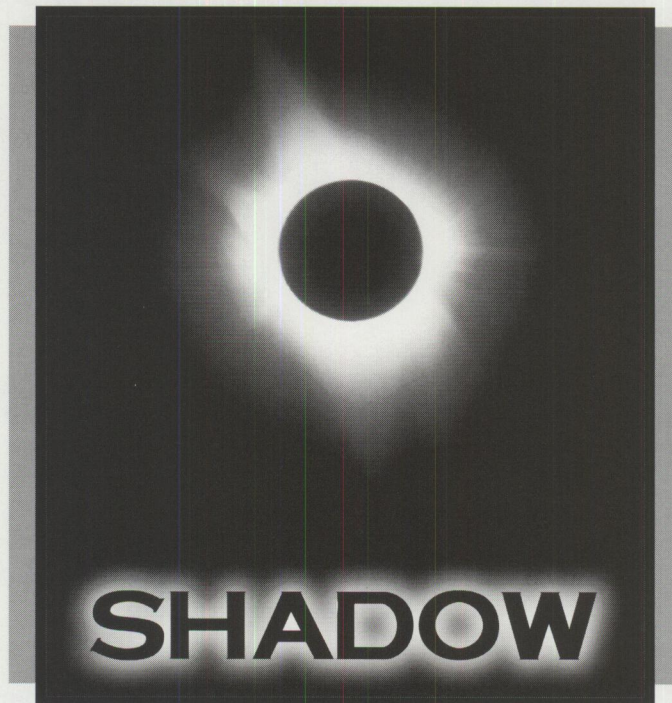
It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such

references or information must not be construed to mean that IBM intends to announce such products, programming, or services in your country.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever.

The articles in this publication represent the views of their authors and do not necessarily represent the views of IBM. This publication may contain articles by non-IBM authors. IBM does not endorse any non-IBM products that may be mentioned. Questions should be directed to the authors.

Publication of advertising material in this magazine does not constitute an expressed or implied recommendation of endorsement of IBM of any particular product, service, company, or technology. IBM takes no responsibility whatsoever with regard to the selection, performance, or use of any advertised products. All understandings, agreements, or warranties must take place directly between the vendor and prospective users.



**IF YOU DON'T HAVE A SHADOW™
MAYBE YOU'RE IN THE DARK
ABOUT THE NEED FOR NBNS**

LET US SHED A LITTLE LIGHT ON THE SUBJECT

Do you need a NetBIOS Name Server? IBM OS/2 WARP and LANServer, Microsoft WFW, Windows 95 and Windows NT all rely on NetBIOS for many Networking operations. Today's enterprise networks, and of course the worldwide Internet, are built on TCP/IP, a protocol that relies heavily on "routing." NetBIOS expects its underlying protocol to find named stations by sending "broadcasts," a method that is not suitable for routed networks. The resolution lies in the use of a NetBIOS Name Server (NBNS), and the perfect integration of NetBIOS and TCP/IP is handled rapidly, reliably and seamlessly by Shadow, the world's best NetBIOS Name Server.

**CALL TODAY FOR
MORE INFORMATION
(800) 990-4776**

Network TeleSystems, Inc. • 550 Del Rey Avenue • Sunnyvale, CA 94086
408/523-8100 • FAX 408/523-8118 • Eastern Region Sales 617/944-3220
online at www.nts.com • e-mail to info@nts.com

Shadow and TCP Pro are trademarks of Network TeleSystems, Inc.
All other trademarks or trade names are the sole property of their respective companies.

NTS
Network TeleSystems

Circle #35 on reader service card.

We've just added another tool
to the power user's toolbox

UNIMAINTM

Version 5.0



● Application Mover

The best desktop repair and maintenance toolkit for OS/2[®] just got a new power tool with UniMaint 5.0's application mover! The application mover moves applications from one partition to another - without the tedious chore of deleting, reinstalling and recustomizing applications. UniMaint also updates system INI and CONFIG.SYS files automatically too. Moving those large applications to the network hard drive is a breeze with UniMaint's application mover.

● Uninstaller

Your hard disk is littered with files left over from the programs you no longer use, or old versions of programs you've since upgraded. The new and improved UniMaint Uninstaller is designed to delete unwanted applications, update INI and CONFIG.SYS files, remove program DLLs, and deregister classes automatically. UniMaint also identifies orphan DLLs for removal.



● Ease of Use

New function wizards make OS/2 maintenance easy for even new OS/2 users.

● Portable Desktop Backup

● Global WPS Settings

● Desktop Repair

"If you run OS/2, you need UniMaint."

Geoffrey Hollander, OS/2 Professional, February 1996

"... a requirement for serious users and I/S departments."

Brian Proffit, OS/2 Magazine, November 1995

Make UniMaint an essential complement to your existing desktop utilities today.

800.944.3028

fax 405.947.8169

405.947.8085



SofTouch Systems[™], Inc.

1300 South Meridian, Suite 600, Oklahoma City, OK 73108-1751

<http://www.softouch.com>

Circle #36 on reader service card.

OS/2 is a registered trademark of IBM corp. SofTouch Systems and UniMaint are trademarks of SofTouch Systems, Inc.