The Great Debate

Force-fitting objects into a relational database just doesn't work well. The impedance problem is at the root of the incompatibilities.

CRAIG S. MULLINS

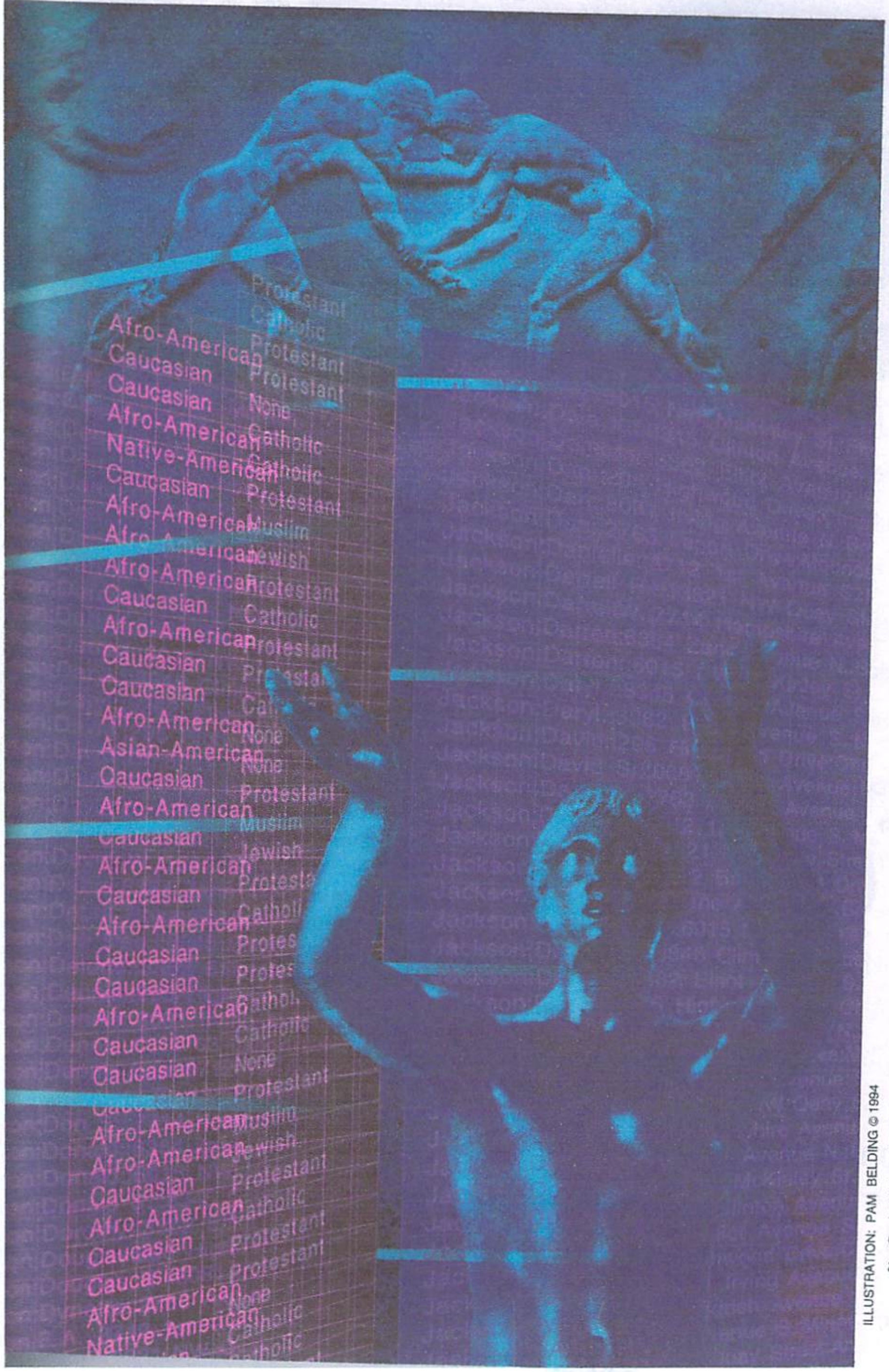


ILLUSTRATION: PAM BELDING © 1994

The relational model has been king of the database hill for the past 10 to 15 years. During this period, few development projects that required a database used anything other than a relational system. But the times they are a-changin'. Object-oriented DBMS products are gaining wide acceptance for their ability to handle complex data in ways that relational products simply can't.

Defining the Paradigms

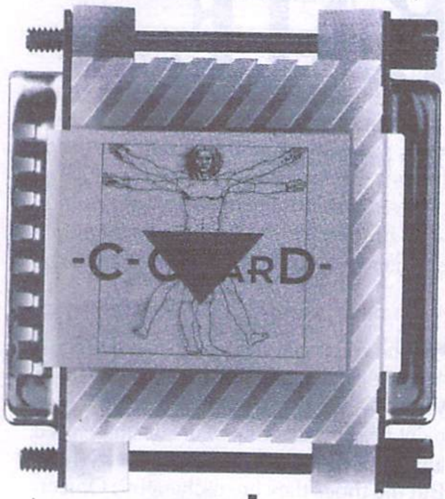
By definition, a *paradigm* is an example that serves as a pattern or model. A *paradigm shift*—a term increasingly seen in the computer trade press—refers to a fundamental change in the basic methods used to accomplish a task.

Most people connected with computer systems realize that the industry changes daily. New technologies are constantly being developed, but most of these merely provide better ways of doing something you have always done. When a paradigm shift occurs, what changes is the essential core of how a task is performed—and perhaps defined.

The object-oriented model represents the latest paradigm for computer programming, after procedural languages and rule-based programming (e.g., in languages such as Prolog). Object-based languages—including Smalltalk, Eiffel, and C++—are based on manipulating objects, which encapsulate complex data structures and processes (usually called *methods*) for manipulating that data. To invoke a method, a message must be sent to the object in which the method is encapsulated. Because each object contains its own methods, most procedural code is eliminated. The object-oriented paradigm, while currently a hot topic in the computer world, is hardly new. The first OOPL (object-oriented programming language), Simula-67, arrived on the scene in 1967. *continued*

for software protection

Count on C-GuarD!



from US $ 12*

Reduce costs, optimize profits:
With a completely new copy protection technology.

- Easy to operate.
- Program installation with C-GuarD.
- Continue working **without** dongle.
- No unauthorized copies.
- Ideal for laptops.

* bulk discounts:
C-GuarD when ordering 1000 : **US $ 12**
C-GuarD when ordering 100 : **US $ 15**

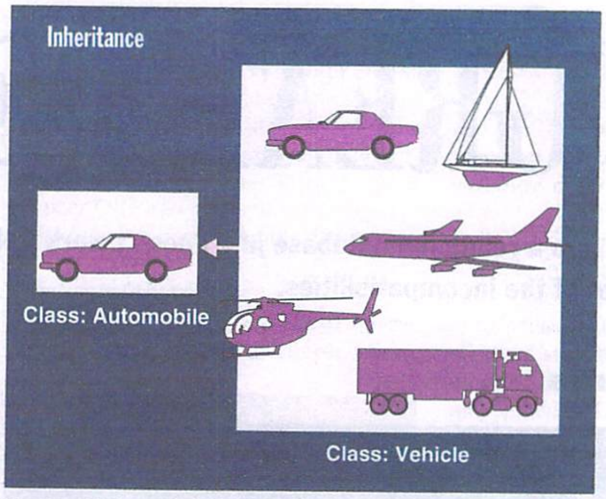
Since 1985 over 1600 customers (as SIEMENS, AEG, IVECO) rely on our software protection products.

VISA

MARX
DATENTECHNIK
DATENSCHUTZ GmbH
Vohburger Straße 68
D-85104 Wackerstein
Tel. ++ 49-8403-1555
Fax ++ 49-8403-1500
E-Mail: 100273.171@ compuserve.com

Hall 4, Booth F56
1st Floor

Circle 162 on Inquiry Card (RESELLERS: 163).



The class Vehicle can include objects of many different subclasses, each with its own special data attributes and methods. Automobile is only one subclass of Vehicle.

Object Database Management Systems

To handle the data that OOPLs create and manipulate and to provide all the fundamental benefits of a DBMS to object-oriented applications, ODBMSes (object-oriented database management systems) were introduced. Benefits of an ODBMS include persistent data, data sharing, concurrent data access, and recovery control.

ODBMS products are designed to supplant relational DBMS products within an object-oriented development environment. Their architectures are designed to understand and utilize object-oriented techniques such as complex objects, abstract data types, encapsulation, and inheritance.

Object vs. Relational

What is the difference between an ODBMS and an RDBMS (relational database management system)? The primary difference is the ability of an ODBMS to support complex objects in an efficient and easy-to-manipulate form. A complex object consists of data and processes that manipulate that data. In contrast, RDBMS products provide access to their data only in terms of rows and columns. And, other than triggers, an RDBMS can't store processing logic at the table level.

Other examples of complex objects include bill-of-materials hierarchies, CAD diagrams, and multimedia BLOBs (binary large objects). An ODBMS is ideally suited to store and manipulate these types of objects. Although some relational databases can process these types of objects, it is seldom easy or efficient to do so. Imagine using SQL Server to explode a bill-of-materials hierarchy from a fully normalized table. While technically possible, it is not truly feasible in terms of ease of use or performance.

Greater Abstraction

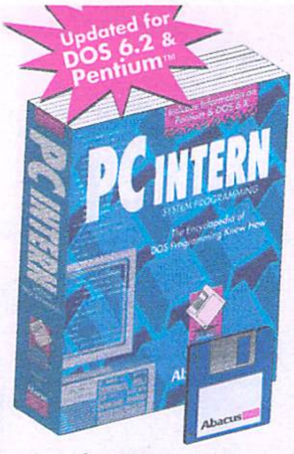
Abstraction is a key component of everyday life. People understand concepts such as an automobile, a light bulb, freedom, and trouble. Automobiles and light bulbs are concrete things, but you need not have them physically present to visualize them. Freedom and trouble are concepts that you can't see or touch, but you understand them in the abstract.

An ODBMS raises the level of abstraction. Objects stored in an ODBMS are organized more closely to the way in which you view and use them in the real world. For example, consider the objects vehicle and automobile. All automobiles are vehicles, but not all vehicles are automobiles. In an ODBMS, you can implement an object of class Vehicle and subclass Automobile that inherits the methods and structure of Vehicle (see the figure "Inheritance"). Using an RDBMS, you would have to implement two separate tables, one for Vehicle and another for Automobile. The Vehicle table can't be used to define Automobile because an RDBMS lacks inheritance. In addition, different algorithms would have to be coded to access each subclass.

The Impedance Mismatch Problem

ODBMS products are sometimes touted as a solution to a problem encountered with RDBMS products. The problem is called *impedance mismatch* and refers to the difference between the declarative, set-level operation of relational-database query languages and the procedural, record-level operation of a typical 3GL (third-generation language). There are two components to impedance mismatch:

1. The difference between the set-at-a-time data manipulation language of the DBMS (e.g., SQL) and the record-at-a-time programming language (e.g., C or COBOL). When the declarative database language is embedded within the procedural language, the system can return multiple rows to a programming language that is not equipped to operate on sets of data (see



Work Smarter

PC Intern System Programming
with Updates for MS-DOS 6.2 and Pentium™

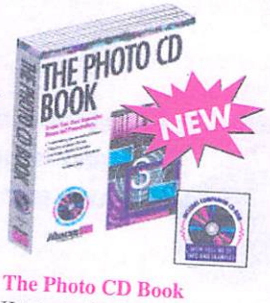
This is a re-release of our all-time best selling PC system programming book. Now PC Intern will include the latest information on programming for DOS, through version 6.2, and programming for the Pentium™ processor. PC Intern is a literal encyclopedia for DOS programmers and contains more than 1300 pages of valuable programming knowledge. The book contains dozens of practical parallel working examples in Assembly language, C, Pascal, and BASIC, for programming video cards, sound, and TSR's. Covers processors, the motherboard, controller, bus and chip sets (NEAT); overview of ports, interrupts, interrupt calls and their significance; the BIOS memory map and services; and much, much more.
$59.95 with companion diskette. **Item #B145**

DOS 6 Complete Special Edition
with Updates for MS-DOS 6.2
The most authoritative and up-to-date DOS reference book available. Your guide to Microsoft's latest version of DOS. It's an encyclopedia of the most recent DOS knowledge, for the computer whiz and the everyday user. Includes many useful tips for outfitting any computer with MS-DOS versions 6 through 6.2.
$39.95 with companion diskette.
Item #B251

Upgrading and Maintaining Your PC
How to turn your PC into a high performance machine. It tells you what you'll see when you open the "hood". How to add a new hard drive, increase memory, upgrade your monitor, or turn an XT into a 486 screamer. Includes utilities and SystemSlueth diagnostics software on companion diskette.
$34.95 with companion diskette.
Item #B167

Multimedia Mania
Explores the evolving multimedia explosion. How to set up a complete multimedia system and how to create presentations. Contains terminology and info about popular programs. Covers audio technology, sound boards and sound recording, CD and CD-ROM technology. Companion CD-ROM features example programs and techniques.
$49.95 with companion CD-ROM
Item #B166

EXCEL for Science & Technology
Excel beyond the spreadsheet. Learn about Excel Solver, Scenario Manager, Mathematics Functions, Physics, Chemistry, Technology Conversion and many other topics.
Macros and worksheets on the companion disk allow you to immediately apply what you learn. Excel for Science and Technology is more than a book; it's an indispensable professional work tool.
$34.95 with companion diskette.
Item #B196

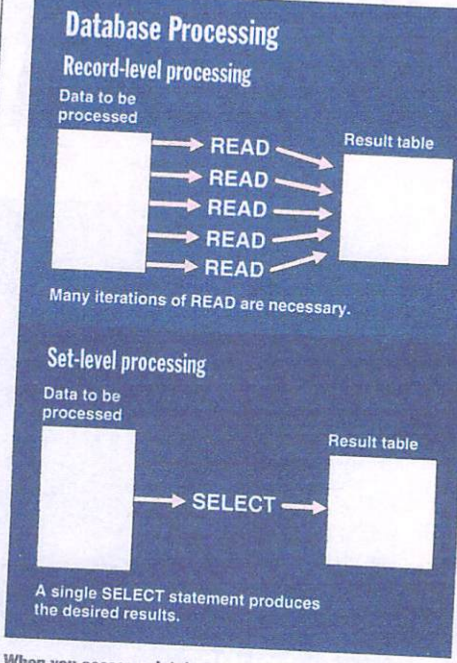


The Photo CD Book
Hot topic for CD-ROM/Multimedia fans, desktop publishers and artists: The most complete guidebook for Photo CD digital image technology. Covers photography, Photo CD system configuration, image processing software, manipulating images, a Photo CD Workshop, creating a home theater, and more. Includes valuable coupons and CD-ROM with examples, photo samplers, demo programs and more!
$29.95 with CD-ROM incl.
Item #B195

Order TOLL FREE 1-800-451-4319 Ext. B4

Ask for our FREE Catalogs of books and software

Abacus
Dept B4, 5370 52nd Street SE
Grand Rapids, MI 49512
1-800-451-4319 Toll Free
Phone 616-698-0330
FAX 616-698-0325



When you access a database one record at a time (top), it takes many READ operations to produce reports or result tables. If the DBMS operates at the set level, it can get the data it needs in one SELECT operation.

the figure "Database Processing").
2. The difference between the typing systems that the DBMS uses and the general-purpose programming language used to develop the rest of the application. For example, many RDBMS products support date types and date arithmetic, but most programming languages don't support them. Therefore, data returned from the DBMS must be transformed into a form that the programming language understands (e.g., by converting a date to a character string).

Typically, when accessing a relational database, you must embed a SQL query within a 3GL program. Because the two languages operate at different levels and some mechanism must resolve the differences (i.e., by copying data from the database language to the programming language and back again), the system incurs extra overhead. On the other hand, you normally access data in an ODBMS by using an OOPL, so no impedance mismatch is encountered. All operations are at the record level.

Why do you need to use two languages? For one thing, the DML (data manipulation language) of the DBMS generally lacks computational completeness and can't handle the nondata manipulation components of an application. On the other hand, most

Circle 61 on Inquiry Card (RESELLERS: 62).

TrueType & Adobe Font Support!
Unlimited Color Support!
Customize your own Toolbox!
Quick-Change Edit Boxes!
On-Line Instruction!
Macros!
Print at any scale!
Cut and Paste to most Documents!
256 layers!
Scores of Dimensioning capabilities!
Over 50 different hatch patterns!
Full Clipboard support!

We Got Tools Out the Wahzoo!

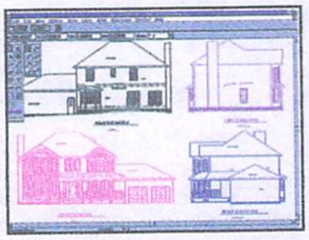
Finally! A CAD System that is all it is cracked up to be. There is nothing extra to buy! DesignCAD 2D for Windows comes complete with all the tools you will need to produce professional designs (from doghouses to Space Shuttles.)

You'll be drawing within minutes thanks to DesignCAD's On-line Instruction and logical interface!

Save time by recording your actions to a Macro so you never have to do the same thing twice. Also, you can put your most-used tools in a "Hot Toolbox" to readily access your most-used commands.

DesignCAD 2D for Windows offers more editing options than other CAD packages. For example, with our "Quick-Change Edit Box" just enter your new specifications and the changes are made for you.

Plus, DesignCAD 2D for Windows offers more than 50 dimensioning options. The dimensioning will even change automatically to reflect your modifications.



Best of all, you can own DesignCAD 2D for Windows for thousands less than comparable Computer Aided Design systems. Call, write or fax to order your copy ***TODAY!***

DesignCAD 2D for Windows
Only $349 or less!
(All Tools Included)

DesignCAD 2D is Also Available in DOS!

Call for a FREE brochure and Demo Disk!

PC MAGAZINE EDITORS' CHOICE
December 11, 1990

Phone (918) 825-7555
Fax (918) 825-6359

American Small Business Computers • One American Way • Pryor • OK • 74361

programming languages lack the facility to handle persistent data other than in the form of files. They also typically lack abstract or high-level data types, constraints, and query capability. To enable the two languages to perform the tasks that each is ideally suited for, they must be able to communicate with each other. This, in a nutshell, is the impedance mismatch problem.

Resolving the Impedance Mismatch

Several different techniques have been tried, with more or less success, to overcome the impedance mismatch:

Query Download. Using this approach, queries are developed that retrieve all objects that the process can require to be executed. The data is translated to a format that the process can read before accessing the object. After execution, the objects are copied back into the ODBMS. This approach results in very efficient program execution at the expense of significant start-up and exit overhead, limited (or no) concurrent data access, and the inability to use the ODBMS during program execution.

BLOBs. Another solution is to store objects in the ODBMS as BLOBs. Instead of identifying each field (i.e., column) and its attributes to the ODBMS, you can store the entire object in a single large field. Once again, the major benefit of this approach is efficiency, but the drawbacks are usually unacceptable. Because the ODBMS is unaware of the object's internal structure, querying is impossible—access requirements can't be coded against individual components (i.e., fields) of the BLOB. Also, the ability to allow concurrent access is problematic.

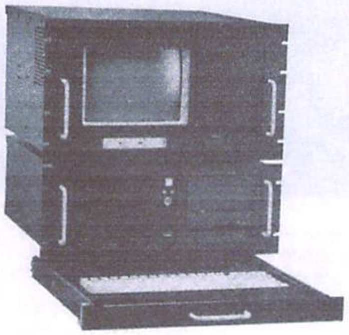
DBMS/Program Affinity. Defining the program's level of interaction with the DBMS to be at a field-by-field (or object-by-object) level removes the set-level versus record-level processing mismatch. However, it also reduces the effectiveness of database access, because all access is one record at a time, thereby crippling the system's ad hoc query capability.

Procedural Database Language. One of the most elegant solutions to the impedance mismatch problem is to augment the database's query language with procedural flow-of-control operations (e.g., WHILE loops and IF...ELSE constructs). Using this approach, more of the application can be written in the query language. A number

Rackmount

- ENCLOSURES
- KEYBOARDS
- MONITORS
- DRIVE ENCLOSURES

Integrand's unique enclosure design uses modular construction. We have 3 basic models for ISA/EISA bus computers. Over 100 interchangable modules allow you to customize them to nearly any requirement. Integrand offers high quality, advanced design hardware and strong support. Why settle for less?



Monitor: 10" Super VGA Color from $650
PC Enclosures from $300
Keyboards: Drawer, Shelf & Panel from $85

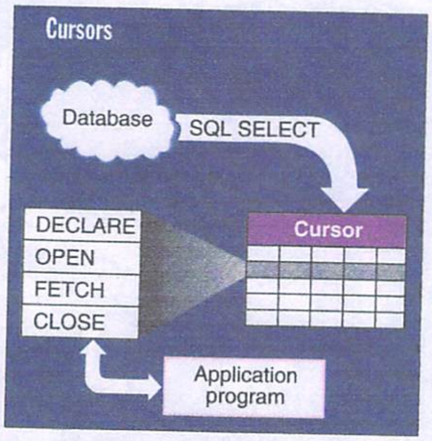
Made in U.S.A.
Rack & Desk Models
Accepts Most Motherboards and Passive Backplanes
Up to 20 Slots
Rugged, Modular Construction
Excellent Air Flow & Cooling
200 & 300 Watt Supplies, UL, CSA, TUV

Call or write for descriptive brochures, prices or applications assistance:

INTEGRAND
RESEARCH CORP

8620 Roosevelt Ave. • Visalia, CA 93291
209/651-1203
FAX 209/651-1353
We accept VISA and MasterCard
Circle 94 on Inquiry Card.

IBM/XT/AT TM IBM • 286/386/486 TM INTEL. Drives and computer boards not included.



In a relational schema, a cursor is a data structure used to hold multiple rows of data returned by a SELECT statement. The application program can then read data from the cursor, accessing it a row at a time as if it were a sequential file, using the DECLARE, OPEN, FETCH, and CLOSE functions.

of RDBMS products, such as Oracle's PL/SQL and Sybase SQL Server's Transact-SQL, provide procedural extensions to SQL.

The coupling of program logic and data into a set-level procedural language is an attractive benefit of this approach. Likewise, performance is usually enhanced because network traffic is reduced. There are, however, some drawbacks; forcing a programmer to write the application using the database language instead of the general-purpose language limits the options and may pose difficulties. Finally, query languages often lack important computational powers and other important features, such as the ability to define and interact with the user interface.

Persistent Programming Languages. We call programming languages *persistent* when they can specify objects to "keep around" when the program is not executing. However, the ability to make program objects persistent still fails to address the interaction of the program with an ODBMS. Impedance mismatch will still occur because the DBMS's retrieval and data-typing scheme won't match those of the program.

Extended Data Types. Support for handling user-defined, extended data types is another approach. But unless both the DBMS and the programming language can define extended data types using the same methods, impedance mismatch is still inevitable.

Integration. In the end, the most workable solution is to simply remove the mismatch entirely. Integrating ODBMS products and programming languages so that they operate at the same level and use the same data model will eliminate the impedance mismatch problem. Implementing this approach, however, takes much effort. You must consider the following problems:

- the lack of a formal, standard object model
- an apparent compatibility conflict between encapsulation and database query languages
- the difficulty of providing standard interfaces and data models given the reality of different commercial implementations

RDBMSes and Impedance Mismatch

Many RDBMS products handle impedance mismatches through two primary mechanisms: cursors and type translation. You can think of a *cursor* as a kind of pointer (see the figure "Cursors"). The programmer declares a cursor and defines a SQL SELECT statement for that cursor. An application program accessing that RDBMS can navigate, one row at a time, through the set of rows returned by the SQL statement. In essence, the program uses the cursor much like a sequential file. It opens the cursor, fetches one row at a time from the cursor, and then closes the cursor. When processing with cursors, a SQL statement can return zero, one, or many rows. Four distinct operations are available for cursors:

DECLARE. Defines the cursor, gives it a name unique to the program in which it is embedded, and assigns a SQL statement to the cursor name.

OPEN. Readies the cursor for row retrieval. OPEN reads the SQL search fields, executes the SQL statement, and builds the result table. It does not assign values to host variables, however.

FETCH. Returns data from the result table one row at a time and assigns the values to specified variables. If the result table is not built at cursor OPEN time, it is built fetch by fetch.

CLOSE. Releases all resources that the cursor uses.

Type translation occurs when the program attempts to retrieve a column that has been defined as a data type that the interface and hardware, the application cannot be successfully completed.

At present, ODBMS products are maturing and will inevitably eat into the huge market share enjoyed by RDBMS products. Nonetheless, RDBMSes will continue to prosper and thrive in the world of business data processing, where applications such as payroll and accounting do not generally require complex objects. Use of ODBMSes will continue to grow in those fields that require complex objects, such as CAD and manufacturing. There will always be true RDBMSes and true ODBMSes.

Successful RDBMS products will seek to incorporate the best components of object-oriented technology without compromising the relational model. Likewise, successful ODBMS products will seek to incorporate the best features of the relational model, without compromising the benefits derived from classes, inheritance, and encapsulation. This will likely lead to a marriage of the two technologies into a hybrid technology—object relational—that builds on the strengths of both relational and object-oriented concepts.

But—and this is no small matter—until the impedance mismatch problem is resolved, inconsistencies will continue to exist between the ways in which the DBMS and the programming language handle data items. ■

BIBLIOGRAPHY

Booch, Grady. *Object-Oriented Design with Applications.* Redwood City, CA: Benjamin-Cummings, 1991.

Braithwaite, Kenmore S. *Object-Oriented Database Design: Concepts and Application.* San Diego, CA: Academic Press, 1993.

Cattell, Roderick G. *Object Data Management.* Reading, MA: Addison-Wesley, 1991.

——, ed. *The Object Database Standard: ODMG-93.* San Mateo, CA: Morgan Kauffman, 1993.

Chorafas, Dimitris, and Heinrich Steinman. *Object-Oriented Databases.* Englewood Cliffs, NJ: Prentice-Hall, 1993.

Hertmo, Elisa, and Lorenzo Martino. *Object-Oriented Database Systems: Concepts and Architectures.* Wokingham, U.K.: Addison-Wesley, 1993.

Khoshafian, Setrag, and Razmik Abnous. *Object Orientation: Concepts, Languages, Databases, User Interfaces.* New York: John Wiley & Sons, 1990.

Kim, Won. *Introduction to Object-Oriented Databases.* Cambridge, MA: MIT Press, 1990.

Runbaugh, James, et al. *Object-Oriented Modeling and Design.* Englewood Cliffs, NJ: Prentice-Hall, 1991.

Taylor, David A. *Object-Oriented Technology: A Manager's Guide.* Reading, MA: Addison-Wesley, 1990.

Zdonik, Stanley B., and David Maier, eds. *Readings in Object-Oriented Database Systems.* San Mateo, CA: Morgan Kaufmann, 1989.

Craig S. Mullins is a technical researcher in database design, client/server technology, object orientation, and relational technology at Platinum Technology (Oakbrook Terrace, IL). He is also the author of DB2 Developer's Guide *(Sams, 1992). You can contact him on Prodigy at WHNX44A, on CompuServe at 70410,237, or on BIX c/o "editors."*

Tripp Lite Smart® Series UPS Out-Networks APC Smart-UPS®

Only one UPS system provides network managers with the most enhanced network protection and control abilities -- Tripp Lite's Smart Series line interactive UPS. Smart Series has a substantial advantage over APC Smart-UPS in critical networking applications.

yes It runs with NetWare

MADE IN USA

NEW!

Tripp Lite Smart Series *OUT-NETWORKS* APC Smart-UPS!

	Tripp Lite Smart Series w/ PowerAlert PLUS	APC Smart-UPS w/ PowerChute Plus®
• Master network log pools power alarms from all UPSs on network to one concise file	STANDARD	*Not available*
• Realtime view of both servers AND WORKSTATION UPS power data from any node	STANDARD	*Not Available*
• Customize alarm points for low voltage, % load, and temperature on each and every UPS	STANDARD	*Not Available*

Tripp Lite Smart Series *OUT-MANAGES* APC Smart-UPS!

	Tripp Lite Smart Series w/ SNMP-2 Adapter	APC Smart-UPS w/ SNMP Adapter
• Redundant AC input on SNMP adapter prevents loss of control of connected UPS	STANDARD	*Not Available*
• "Out of Band" network communication when Ethernet disabled	STANDARD with your modem	*Not Available*
• Ambient temperature and humidity reports	STANDARD	*Extra Cost Option*
• Remote power control of HVAC or other peripherals	STANDARD	*Not Available*
• SNMP adapter controls up to 4 UPSs	4 STANDARD	*Up to 2, only*

Tripp Lite Smart Series *OUT-POWERS* APC Smart-UPS!

• Lowest correctable brownout voltage without battery drain	87 volts	*91 volts*
• Over-voltage correction peak without battery	140 volts	*Not Available*

Find out what else Smart Series UPSs can do.
Call today for more information on the Smart Series advantage!

TRIPP LITE EST 1922
"THE POWER PEOPLE"
500 N. Orleans, Chicago, IL 60610-4188
(312) 755-8741 Fax (312) 644-6505
FaxBack Service (312) 329-9717
Ask for Dept. S20

Smart-UPS and PowerChute Plus are registered trademarks of American Power Conversion, Inc.

Circle 139 on Inquiry Card (RESELLERS: 140).