Patricia Seybold's Office Computing Group

Editor-in-Chief
Judith S. Hurwitz

UNIX IN THE OFFICE

Guide to Open Systems

Vol. 6, No. 6 • ISSN: 0887-3054 • June 1991

Uniface

Developing Database-Independent Applications

By Judith R. Davis

IN BRIEF: Many organizations are struggling to implement effective database applications across a heterogeneous computing environment. Uniface, a third-party tools vendor, has taken an innovative approach to solving this difficult problem. It gives the developer a single environment and a uniform approach to designing applications regardless of the underlying platform, database management system (DBMS), or native user interface. The resulting applications are not only database independent but can, in fact, access data in multiple DBMSs concurrently. Uniface has generated a great deal of excitement among developers and may help push the entire industry to a new level of functionality.

Report begins on page 3.

EDITORIAL: BY JUDITH S. HURWITZ

Standard Products vs. Standard Interfaces

Is There a Simple Answer? (Is there ever a simple answer?)

I HAD AN INTERESTING thought as I sat talking to Dave House, president of Intel Microcomputer Components Division, the other day. He was explaining the phenomenal success of the 386 chip and the predicted success of the 486 chip. Here was a standard product that was good for the industry. As Intel will explain it, if asked, the company spends a tremendous amount of R&D on its technology. It then has the technology that the industry needs. It licenses either chip technology or configured systems to users, systems integrators who then resell it. The Intel technology becomes pervasive by its sheer size. In 1991, Intel sold 28 million processors versus 2.3 million Macintoshes, 280,000 SPARC machines, and 28,000 MIPS processors—these numbers come from Intel's count. Any company that ships that many processors becomes a de facto standard—at least in some areas of the industry.

So what could be bad about this? If you are a competing chip manufacturer, you probably resent the fact that this one vendor controls the direction of such a large portion of the hardware business. If you buy "standard" products from Intel, you probably worry that the company has too much power to follow its own agenda and can set prices as high as it wishes and decide on characteristics of future products that might not help you—especially if you're an ISV making software that requires special hardware features (i.e., multimedia, graphics, etc.).

Now, what occurred to me while I was talking to Mr. House was that users and ISVs are comforted by the idea of standard software products. They mean stability. A large population of users find solutions that work. The solution may not be precisely what a user was looking for, but at least it will get the job done. One big problem that faces those users committed to standard interfaces is that, even after an entire segment of the market agrees on a standard interface, the standard is not implemented into workable, state-of-the-art products for years.

The Open Software Foundation (OSF) and the Object Management Group (OMG) are excellent examples of the power of standard products. Had OSF decided to take the time to write a series of APIs for its Distributed Computing Environment RFT, they might not have been written, approved, and finally implemented in commercial products for at least two years. This might have been preferable from a long-term perspective, because any vendor with a new idea and more advanced technology would have been able to write to these specifications and provide new implementations. In the long run, users would have been better served. The same can be said for the Object Management Group's search for a standard distributed object management approach. One of the two choices before the OMG (from a joint proposal by Sun Microsystems and Hewlett-Packard) is based on implementations of object broker technology, not on a standard API. Many users, for example, would have preferred that Hewlett-Packard and Sun provide the industry with a distributed object management API. However, we suspect that this would have taken longer and would have upset their installed bases too much for such an approach to have been attempted.

Users will be forced to deal with some serious problems over the coming years because of this trend towards implementing standard products. They will be tied to implementations of technologies that may not age well. They will face the same problems they have always faced when they implemented proprietary technology and then had to start from scratch again when innovation suddenly made the old stuff obsolete.

This issue will not die quietly. User organizations and standards groups will continue to push hard for standard interfaces. Pragmatic system and software suppliers and users with the business model in mind will push for standard products so they can get on with the task of making money. The debate is a difficult one, and there will not be a simple answer. Perhaps the best hope for standard interfaces may come out of the CASE environment, where standards organizations and computer systems vendors are beginning to try to standardize on application development frameworks. ©

UNIX IN THE OFFICE

Editor-in-Chief
Judith S. Hurwitz

MCI:
JHurwitz
Internet:
jhurwitz@mcimail.com

Publisher
PATRICIA B. SEYBOLD

Analysts and Editors
JUDITH R. DAVIS
DAVID S. MARSHAK
RONNI T. MARSHAK
MICHAEL D. MILLIKIN
LAURE B. ROWAN
JOHN R. RYMER

News Editor
DAVID S. MARSHAK

Managing Editor
DOUGLAS A. FREEMAN

Art Director
LAURINDA P. O'CONNOR

Sales Director
RICHARD ALLSBROOK JR.

Circulation Manager
DEBORAH A. HAY

Customer Service Manager
DONALD K. BAILLARGEON

Patricia Seybold's
Office Computing Group
148 State Street, 7th Floor,
Boston, Massachusetts 02109
Telephone: (617) 742-5200
Fax: (617) 742-1028
MCI: PSOCG
Internet: psocg@mcimail.com
TELEX: 6503122583

Unix in the Office (ISSN 0890-4685) is published monthly for $495 (US), $507 (Canada), and $519 (Foreign) per year by Patricia Seybold's Office Computing Group, 148 State Street, 7th Floor, Boston, MA 02109. Second-class postage permit at Boston, MA and additional mailing offices.

POSTMASTER: Send address changes to *Unix in the Office*, 148 State Street, 7th Floor, Boston, MA 02109.

Printed on recycled paper.

FEATURED REPORT: BY JUDITH R. DAVIS

Uniface

Developing Database-Independent Applications

Choosing tools with which to develop applications and access data in a heterogeneous environment is a daunting task. First, you have to understand what it is you are trying to do—what your business requirements are and how they will change in the future. Then you must identify the functionality that will satisfy these requirements and the development tools that offer this level of functionality. Like the choice of a database vendor, a good decision on the application development side involves knowing your application requirements and priorities, and then evaluating the trade-offs you must inevitably make in selecting one solution over another.

In this report, we present a review of Uniface, a development environment targeted specifically for generating database-independent applications. A Uniface application can also access multiple, heterogeneous databases concurrently. Uniface is already popular in Europe, where it got its start in 1987. We expect the company's momentum to continue as it increases its visibility in the United States. Uniface is gaining recognition for its innovative approach to applications development, and it has the potential to make a significant impact on the tools industry.

Introduction to Uniface

Applications That Adapt to Heterogeneous Environments

The Uniface name derived from a combination of syllables in the words *uni*form, *uni*versal, and inter*face*. The basic concept is to provide a single, consistent development environment that is completely independent of the underlying platform. A Uniface application is designed the same way no matter what database manages the data. Uniface has written a series of drivers to integrate the application into a particular computing environment—the hardware and operating system, network protocol, data manager (DBMS or file management system), presentation manager, CASE tool, and 3GL. All that is required to move the application to another environment is a different set of drivers and supporting products. The application itself remains unchanged.

A Migration from Europe to the United States

Uniface B.V. was founded in 1984 in The Netherlands. Its initial funding included participation from several Beta site customers. The product was developed relatively quickly (it is written in C), and it spent an unusually long time—over two years—in Beta test mode. Uniface's founders placed great importance on simply listening to the customer, and the long customer-input phase resulted in a stable product with distinctive appeal in the marketplace. The Uniface product was formally introduced in 1987.

A U.S. subsidiary, Uniface Corporation, was opened in mid-1990. Uniface did not want to introduce the product in the United States until the validity of the concept had been proven in its European installations. It was important to have a significant portfolio of experience and references before hitting the highly competitive U.S. market.

A 1984 Vision That Fits Today's Requirements

THE VISION. The vision that the Uniface founders articulated back in 1984 is one that fits today's requirements quite well. Our initial reaction to Uniface reminds us a little of the one we had when we first learned about Sybase—the product philosophy and design made so much sense that it was hard to believe no one had already implemented it. There is obviously much demand for an application development tool that can accommodate a heterogeneous hardware and software environment, just as there was pent-up demand for what Sybase had to offer when it was introduced back in 1987.

Database Independence. The Uniface vision anticipated several trends in the software development process. One was that DBMS technology would stabilize and become a commodity, eventually to be bundled with operating systems. While this is not true across the board, there are indications that the trend is underway. Several vendors, including Digital Equipment and IBM, are bundling DBMS software with their operating systems.

And while database engines are certainly not a commodity yet, all DBMS vendors are moving in the same general direction with regard to architecture and functionality: multithreaded, multiserver architectures, stored procedures, triggers, declarative referential integrity, distributed database support, full ANSI SQL compliance, etc. We do believe, however, that there will always be demand for value-added functionality around a core of standard features, which will enable at least some of the database engine companies to continue to differentiate themselves. The real issue is whether a database engine company can also be good at front-end application tools. The Uniface users we interviewed were uniformly more impressed with Uniface than with other 4GL tools on the market.

Adapting to New Technology. Uniface also foresaw the difficulty in accommodating new technologies and techniques, such as CASE, client/server architectures, and graphical user interfaces (GUIs), in the application development process and existing tools. We find that many vendors take the easy way out. Rather than extend or redesign products to take advantage of new technology, they develop new products. The customer then often faces the difficult choice of either throwing out the old to implement the new or living with outdated technology. The Uniface approach is to design the product so that new technology can be incorporated without disrupting the installed base.

Major Issues in Application Development

There are several major issues in selecting application development tools.

Mapping Business Needs to a Toolset Isn't Easy

Many organizations have difficulty mapping their business needs to the array of tools available. How does the user know which tools will solve the business requirements at hand? There are several issues here. One is the fact that many tools are available, and it is not easy to determine which are comparable. Another issue is the fact that the tools vendors are not good at selling back to specific business needs; the vendors focus much more on features and functions than on articulating how these features help the customer solve a specific problem. Still a third issue is the potentially erroneous assumption on the part of the user that, if the business problem appears complex, then the answer (i.e., the tools selected) must be comparably complex and sophisticated. How can this difficult problem possibly be solved simply by extracting data into a spreadsheet?

Gaining the Freedom to Choose: Portability and Independence

The primary issue here is how to support a multilevel heterogeneous computing environment. Ideally, the user wants a single development environment and applications that are truly portable across whatever systems are in place. Shielding the developer from the complexities of the operating system and hardware, networking, graphical user interfaces, and even the specific database manager means the developer can focus on the primary objective: good database and application design.

Many think that the DBMS vendor needs its own set of tools to ensure that new back-end features show up in the tools layer on a timely basis (or at all). The question becomes: Can one company be good at both? And what does the customer give up in independence and portability to go with the DBMS vendor across the board? The DBMS vendor doesn't even necessarily roll out client and server software concurrently on each platform. And the customer may want to develop and deploy on a platform that is not yet supported as a client by the DBMS vendor. Tools vendors such as Uniface provide an option for independence and portability at the database level.

Productivity Is Important at All Levels

A major objective is to tighten or compress the development cycle and to increase productivity at all stages of the life cycle.

PROTOTYPING. The ultimate goal is to deploy what you prototype, a "try it, fix it, do it" iterative process. Prototyping criteria include:

- Can I make the application run without having completed everything?
- Are there facilities to generate dummy/test data?
- Instead of writing a subsystem, can I mock it up to test what I am working on?
- Can I paint screen layouts and make them behave the way the system will without doing a lot of work?

Some of these are showing up in 4GL tools, but many vendors haven't really thought through the importance of the prototyping process.

LEARNING CURVE. Another productivity goal is reducing the slope of the learning curve in terms of productivity—reducing the time it takes to go one step up the learning curve in using a tool.

Integration among Tools Is Becoming Critical

All developers are looking for the seamless integration of a variety of tools across the development life cycle. This includes CASE, 4GLs, 3GLs, repositories, DBA management tools, and end-user tools.

Customization and Deployment

An issue for VARs in particular is the ability to customize an application for a user organization while continuing to use the tools for development.

Access to Data: The End User's Objective

The ease and flexibility of accessing data is becoming more critical for all types of users every day. We are beginning to see a new emphasis on easy-to-use end-user tools for ad hoc query and reporting, and a trend toward integration with familiar PC tools.

Migration Path

Users want to take advantage of new technology and functionality while still preserving their investment in existing systems. They are looking for an application development environment that is modular and flexible enough to provide this migration path.

Product Line

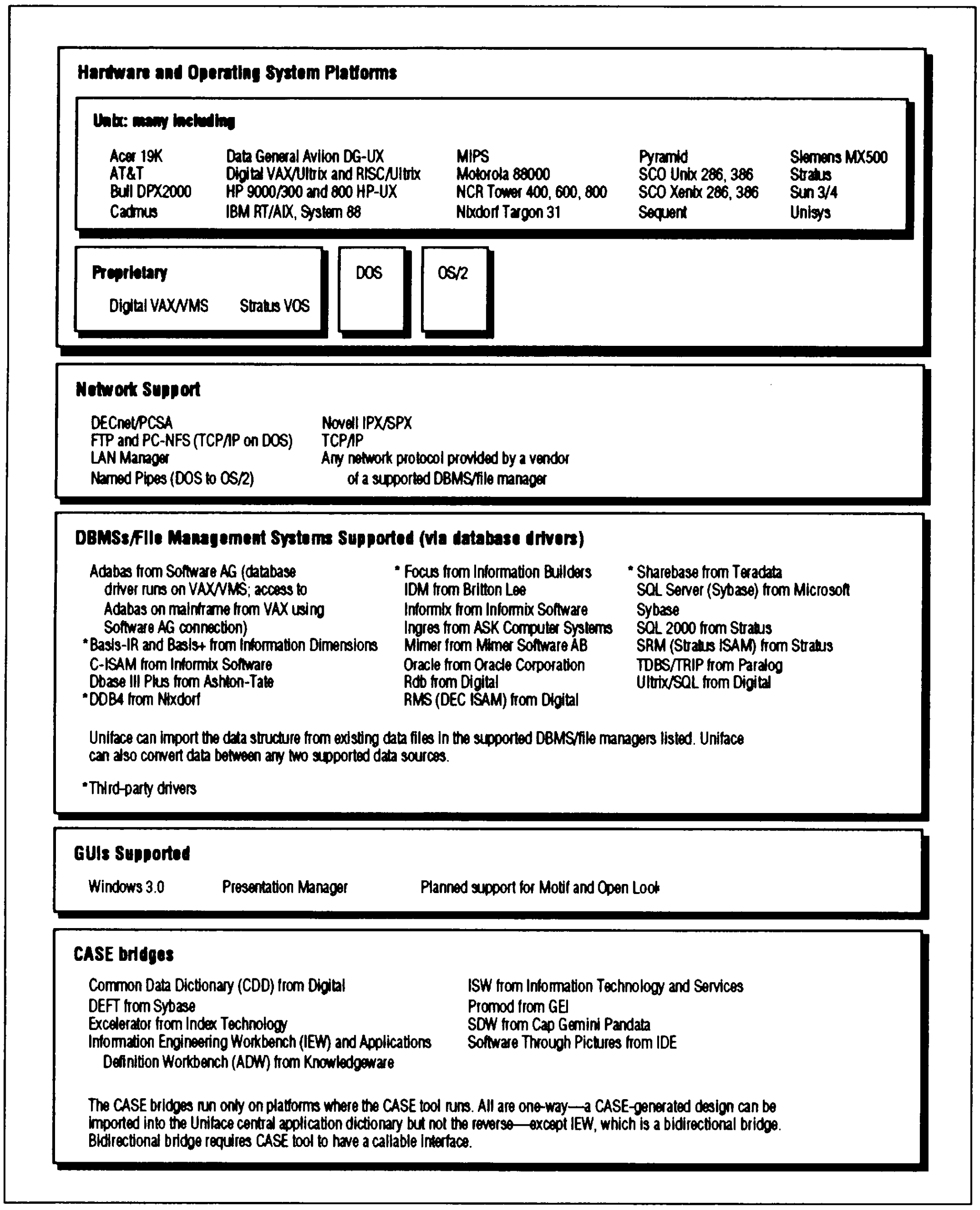
The Uniface product line consists of the Uniface development system, a series of database and interface drivers and networking support. Illustration 1 provides a summary of the Uniface computing environment. The primary development platform is Unix.

Separating Data Definition, User Presentation, and Physical Storage

UNIFACE. The Uniface development environment—called the Information Engineering and Design Facility, or IDF—is based on the ANSI 3-Schema architecture. Uniface claims to be the only company to have implemented this. The 3-schema architecture separates application development into three functions. Data structures, combined with relationships and constraints (such as referential and other data integrity constraints and validation rules), make up the *conceptual schema*. The *external schemas* define how the application will look to the user. In

Uniface, external schemas are equivalent to forms. The *internal schema* defines the physical representation of the data and is based on which data manager(s) will underlie the application.

The Uniface Environment



Hardware and Operating System Platforms

Unix: many including

Acer 19K	Data General Avilon DG-UX	MIPS	Pyramid	Siemens MX500
AT&T	Digital VAX/Ultrix and RISC/Ultrix	Motorola 88000	SCO Unix 286, 386	Stratus
Bull DPX2000	HP 9000/300 and 800 HP-UX	NCR Tower 400, 600, 800	SCO Xenix 286, 386	Sun 3/4
Cadmus	IBM RT/AIX, System 88	Nixdorf Targon 31	Sequent	Unisys

Proprietary

Digital VAX/VMS   Stratus VOS

DOS

OS/2

Network Support

DECnet/PCSA
FTP and PC-NFS (TCP/IP on DOS)
LAN Manager
Named Pipes (DOS to OS/2)
Novell IPX/SPX
TCP/IP
Any network protocol provided by a vendor of a supported DBMS/file manager

DBMSs/File Management Systems Supported (via database drivers)

Adabas from Software AG (database driver runs on VAX/VMS; access to Adabas on mainframe from VAX using Software AG connection)
*Basis-IR and Basis+ from Information Dimensions
C-ISAM from Informix Software
Dbase III Plus from Ashton-Tate
*DDB4 from Nixdorf
* Focus from Information Builders
IDM from Britton Lee
Informix from Informix Software
Ingres from ASK Computer Systems
Mimer from Mimer Software AB
Oracle from Oracle Corporation
Rdb from Digital
RMS (DEC ISAM) from Digital
* Sharebase from Teradata
SQL Server (Sybase) from Microsoft
Sybase
SQL 2000 from Stratus
SRM (Stratus ISAM) from Stratus
TDBS/TRIP from Paralog
Ultrix/SQL from Digital

Uniface can import the data structure from existing data files in the supported DBMS/file managers listed. Uniface can also convert data between any two supported data sources.

*Third-party drivers

GUIs Supported

Windows 3.0   Presentation Manager   Planned support for Motif and Open Look

CASE bridges

Common Data Dictionary (CDD) from Digital
DEFT from Sybase
Excelerator from Index Technology
Information Engineering Workbench (IEW) and Applications Definition Workbench (ADW) from Knowledgeware
ISW from Information Technology and Services
Promod from GEI
SDW from Cap Gemini Pandata
Software Through Pictures from IDE

The CASE bridges run only on platforms where the CASE tool runs. All are one-way—a CASE-generated design can be imported into the Uniface central application dictionary but not the reverse—except IEW, which is a bidirectional bridge. Bidirectional bridge requires CASE tool to have a callable interface.

Illustration 1. Uniface runs on Unix, DOS, and OS/2 plus proprietary platforms from Digital and Stratus. It currently supports over 20 DBMSs and file systems on the back end.

One major benefit of the 3-schema architecture is the fact that, once the data definitions and relationships are defined in the conceptual schema, they are used automatically on external schemas. For example, when designing a form that contains data from multiple tables, the developer does not have to define how the tables are joined if the table relationships are included in the conceptual schema. The relationships between data elements are defined once and then applied throughout the application.

The Data Itself Drives (Defines) the Application

Data-Driven Applications. One aspect of the Uniface philosophy is that an application can be completely defined in terms of the data it processes. Data structures and constraints are all that are needed to actually define an application. The developer defines the application the same way, no matter what the application will look like to the user and which platform it will be implemented on.

Uniface Pricing

Uniface Full development system (IDF) Run-time version (both include the default database driver)	 $5,000 - $250,000 $1,000 - $61,000
Polyserver (includes one network driver and one database driver)	$2,000 - $51,000
Drivers/bridges Network drivers Database drivers Interface drivers CASE bridges	$700 - $40,000 each
"Cookbook" to build database driver or CASE bridge	$1,000
à la Carte	$800 - $40,000

Effective April 30, 1991

Illustration 2. Product pricing varies depending on the specific hardware and operating platform.

A Forms-Driven Development Environment

Uniface is forms based, rather than language based. It has a 4GL for generating procedures, but the procedures are not standalone. You cannot start out writing a procedure and then call up forms, for example, as you can with the Informix, Ingres, or Progress 4GLs. You attach Uniface procedures to an event (e.g., enter form, leave modified field, click with the mouse while in a field, press the menu key) associated with an object. An object can be a form, an entity (table), a field, or a group of fields. This is what Uniface means when it says its environment is *data driven* and *event triggered.*

Extensive Flexibility Results from a Modular Architecture

A Modular Product Architecture. Modularity is a core of the Uniface design center. Not only is the Uniface development environment separate from the database and interface drivers, but Uniface has also implemented a high degree of modularity within the development environment. For example, menu definition is done separately from screen definition. This modularity provides flexibility and reduces the development and maintenance effort.

Targeted for the Professional Developer

As is typical of most 4GL products, Uniface's target user is the professional developer. In terms of the development life cycle, Uniface is designed to cover primarily the application development, testing, and deployment phases of the applications development life cycle, with support in the back end of the life cycle for maintenance from the application perspective, but not from the DBA perspective. Uniface also offers bridges to several CASE environments and has announced à la Carte, an end-user report writer, to help end users more easily access data.

Database Drivers Map the Application to the DBMS

DATABASE DRIVERS. The database driver translates between Uniface and the specific data manager on the back end. Each driver is customized for the data manager and is designed to take advantage of the features and functionality available. Here are just a few examples:

- Automatically generating delete triggers and stored procedures for Sybase.
- Automatically generating BLOB data types for DBMSs that support them, such as Informix and Sybase.
- Supporting two drivers for Oracle, allowing the customer to choose to access the Oracle database with embedded SQL or the call level interface.
- The ability to create Rdb tables even if the customer only has the run-time version of Rdb. (Digital ships a run-time version of Rdb with the VMS operating system, but you cannot create tables with the run-time system alone.)

The database driver typically consists of 1,000 to 3,000 lines of C code for a relational DBMS (RDBMS), and may require more code for nonrelational data managers. Uniface also provides a database driver "cookbook" that specifies how to create a database driver for use by customers or any third party. Uniface comes with one default database driver depending on the operating system (e.g., C-ISAM for Unix and RMS on VAX/VMS).

One potential drawback to Uniface for large companies is that it does not run on the IBM mainframe, and the company does not intend to develop its own drivers for data stored in DB2, IMS, etc. The strategy is to support these data sources through the DBMS vendor's gateways. Uniface will also support the Sybase Open Gateway to DB2 when it is available.

Taking the "Highest Common Denominator" Approach

Highest Common Denominator. One of the real benefits of Uniface is its "highest common denominator" approach. There are two important perspectives here. One is understanding that Uniface supports the same generic application functionality regardless of what back end is used and whether or not the data manager supports that functionality directly. Uniface does not limit the user to a common subset of functionality that can be mapped to every data manager. The second perspective is that Uniface is designed to take advantage of the products it supports, using native functionality in the back end wherever possible.

For example, let's say the developer has defined a referential integrity constraint in the IDF that states that a customer master record cannot be deleted if orders for that customer exist. Uniface will always enforce this constraint and will implement it according to the capabilities of the specific data manager. In the case of Sybase, a Delete trigger is created on the customer table to enforce the constraint (via the database driver). Uniface also supports the option to cascade deletes, so additional Delete triggers may be created depending on how the application is defined in the conceptual schema.

In the case of Oracle, which doesn't support server-enforced integrity, Uniface creates the appropriate code in the application itself to enforce the constraint. And if the application is moved from Sybase to Oracle, Uniface will replace the triggers (and any stored procedures) with application code. Uniface can also convert the data between the two DBMSs.

In each case, the code is generated automatically by Uniface; the developer doesn't have to write or modify any code. This is the type of functionality that gets developers truly excited about Uniface.

Lest you go in with the wrong assumptions, be aware that Uniface isn't quite there yet in automatically generating complete native functionality to support an application. For example, only Delete triggers are generated for Sybase. However, Uniface still enforces whatever declarative referential integrity constraints are defined in the conceptual schema key relationships. For example, it can be instructed not to let the user insert an orders record if there is no corresponding customer record. But it cannot automatically know what additional processing to include in an Insert trigger. Uniface can restrict the user's actions, but cannot automatically cascade processing. One of the issues here is the difference between declarative referential integrity (declared in the data dictionary as primary and foreign key relationships) and integrity constraints that are enforced procedurally. Uniface is currently based on the declarative mode.

You Want to Use an Existing Database? No Problem!

Existing Data Structures. If you already have an existing database, Uniface can pull that information in from the back end and create the application dictionary based on the information contained in the data dictionary—tables, views, fields, indexes, and any declarative relationship information. (Uniface cannot decode existing Sybase triggers, for example, to obtain relationship information.)

Flexibility in Configuring the System

NETWORKING. The user has two options in configuring Uniface. One is to use the APIs and networking support from the DBMS vendor, adding Uniface and a database driver on the client side (see Illustration 3). The second option is to use the Uniface client/server architecture and networking support (see Illustration 4).

The Application Can Connect to Heterogeneous Data

Polyserver. Polyserver is essentially Uniface's implementation of a client/server architecture. Polyserver can handle multiple network protocols and database drivers (it comes with one of each), and it runs on the server with the data manager. Polyserver provides two major benefits. The first is the ability to adapt older architectures to a client/server model (e.g., RMS and C-ISAM). The second is the ability to connect to multiple DBMS servers concurrently when the DBMSs are on different servers and the client is not directly connected to both (i.e., server-to-server communications).

Uniface Option 1: Use API/Networking from DBMS/File Manager Vendor

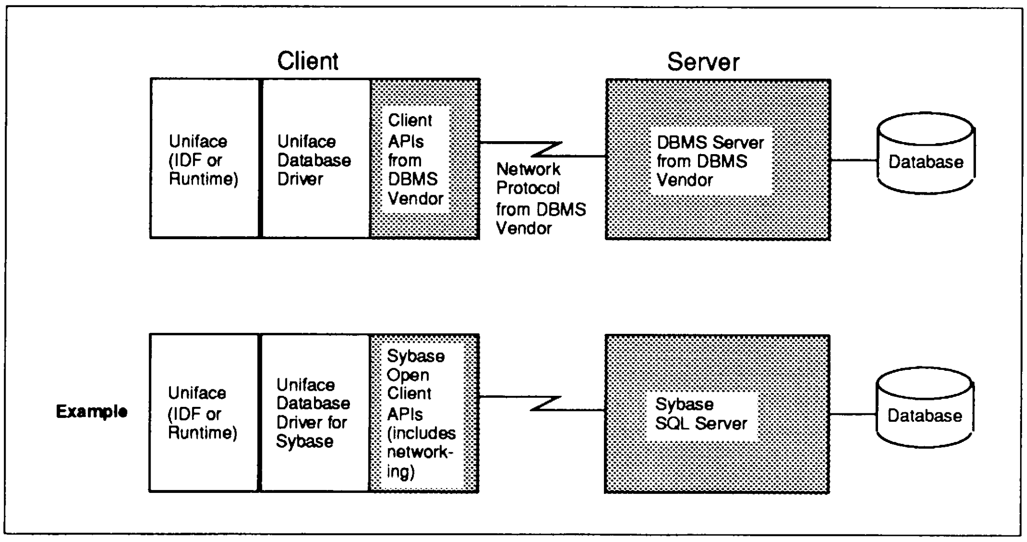

Client
Server
Uniface (IDF or Runtime)
Uniface Database Driver
Client APIs from DBMS Vendor
Network Protocol from DBMS Vendor
DBMS Server from DBMS Vendor
Database
Example
Uniface (IDF or Runtime)
Uniface Database Driver for Sybase
Sybase Open Client APIs (includes networking)
Sybase SQL Server
Database

Illustration 3. The customer can use Uniface with the APIs and networking support from the database vendor. This is important if the customer already has this software installed. Here, Uniface and the database driver(s) run on the client desktop. The user can access multiple databases on different servers if there is a direct connection to each server.

Uniface Option 2: Use API/Networking and Polyserver from Uniface

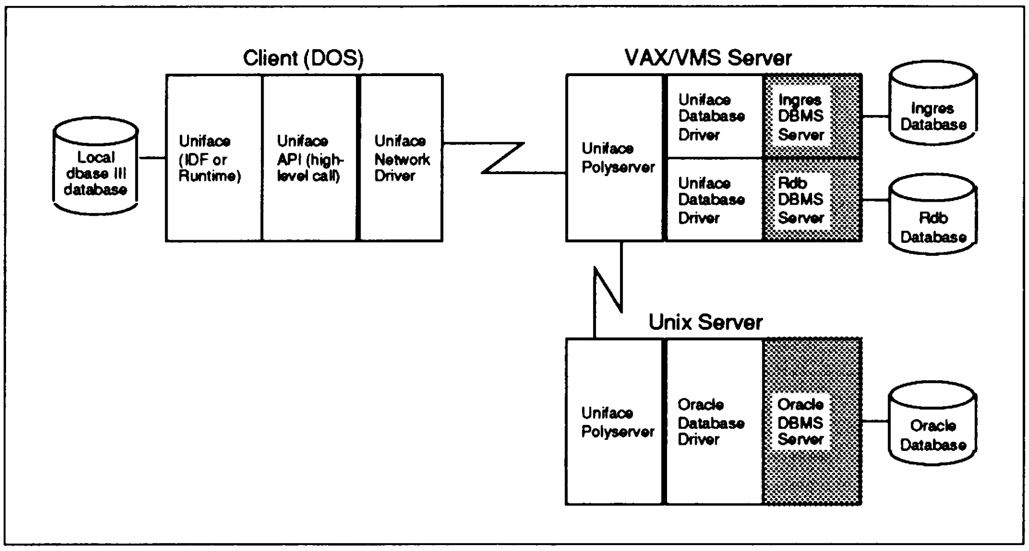


Illustration 4. The Uniface APIs are part of the standard Uniface IDF. Adding a client network driver and Polyserver on the database server platform completes the network connection. (The physical network is acquired from the network vendor.) Benefits of this configuration are server-to-server communications and the ability to adapt older architectures to a client/server model.

Polyserver acts as a router and translator, turning the high-level Uniface API call from the client into SQL or a call to a procedure, etc. Polyserver then relies on the database driver (which resides on the server in this case) for the translation to the DBMS-specific language.

Polyserver spawns one server process per client. One potential concern is the processing load Polyserver puts on the server platform when accessing a data manager with a similar architecture, such as Oracle or Informix. Since these also use the server-per-user model, Polyserver effectively doubles the number of server processes. However, according to Uniface, a Polyserver process doesn't require much in the way of server resources. It is simply a router to connect the client with the data manager through the database driver.

Uniface Relies on the Database Manager for Distributed Database

Distributed Database. Although a Uniface application can access data in multiple DBMSs concurrently, Uniface is careful to stress that it is not a distributed database product. It uses the DBMS vendor's capabilities here. However, Uniface does understand two-phase commit. The user can set an environment variable to turn 2PC on; Uniface will issue "prepare to commit" instructions to the DBMSs that understand it, and will commit them first before committing the non-2PC database(s). The application program performs the coordinator role.

A Good PC Client/Server Architecture

PC Client/Server. On the PC, most customers don't want an application that has the DBMS vendor's look and feel—e.g., an Oracle- or Sybase-like application; they want the application to be PC-like and independent of the DBMS vendor. Uniface has done a good job of providing this with both its character-based and GUI-based interfaces on the PC. The PC can also function as a development platform, with the ability to deploy the application on any other supported platform.

Uniface Is Close to True Presentation Independence

PRESENTATION INDEPENDENCE. Although Uniface was developed in a character-based environment, it has always had its own windowing interface with strip/pull-down menus, multiple overlapping windows, scrollable forms and frames, zoom capability, selection lists, the ability to cut and paste between windows as appropriate, and a WYSIWYG editor. Uniface was also designed from the beginning to provide presentation independence through its Universal Presentation Interface (UPI). Uniface uses the same architectural concept on the front end as it does on the back end with its database drivers, providing a set of customized interface drivers to make the application independent of the presentation environment. Thus, the same Uniface source code can be:

- Simply recompiled to use a different window manager, or
- Run in interpretive mode in a different (supported) GUI environment by specifying the window manager in the assignment table. Uniface will then dynamically redefine the application interface on a run-time basis.

This works not only between different window managers, but also between a character and a GUI interface. You simply write your Uniface source code, and it can subsequently run against any supported interface (or data source) without modification. Uniface claims it is unique in this respect; no one else can provide this level of flexibility without requiring code to be rewritten.

Planned enhancements in the next release of the UPI (third quarter 1991) are a graphical form painter, the ability to display graphic images, and hot links with other applications (e.g., support for DDE on Windows and Presentation Manager).

Windows and PM Are Supported Now; Motif and OpenLook Are Coming

Current GUIs. Uniface currently runs on both Windows 3.0 and Presentation Manager (PM). What's missing are features like an icon manager, dynamic color assignments, and, in the case of Windows, support for Dynamic Data Exchange (DDE).

Future plans here include a second version for Windows and PM to implement the missing pieces described above, plus support for OSF/Motif and OpenLook. The Windows and Motif drivers are due in September with Version 5.2 of Uniface. The OpenLook driver will follow by the end of the year, and PM, in early 1992. Uniface has no plans yet to support the Macintosh.

Bridges Provide Integration with CASE Tools

CASE. Uniface can import data modeling and data definition information from several CASE tools through its CASE bridges (see Illustration 2). As with the database drivers, there is a cookbook for developing CASE bridges.

Using Uniface

A Powerful Application Development Environment

Uniface is a sophisticated, powerful application development environment. We don't have space to cover all of its functionality in detail, but will briefly touch on the basic application development functions, highlight some capabilities that are particularly interesting, and present some issues raised by the architecture.

The Uniface IDF separates the development process into two major tasks: defining the conceptual schema and defining external schemas. Illustration 5, a map of the IDF main menu options, provides a feel for the way Uniface is organized and what the product looks like. The IDF was developed using Uniface and is an example of a Uniface application.

A Steep Learning Curve. Uniface stresses that its product is not a snap to learn, primarily because of its depth and power and the need to understand how to work with the 3-schema architecture. Jim Milbery, a principal consultant with Uniface, put it this way: "Uniface has a relatively high fixed cost, but the benefit is a lower variable cost. Once you learn the product, you become very effective at developing applications." The developers we interviewed echoed this, and everyone agrees that a training program is a must, even for experienced database developers. Uniface does not have a tutorial, which would be a helpful guide for the developer.

Map of the Uniface IDF Main Menu

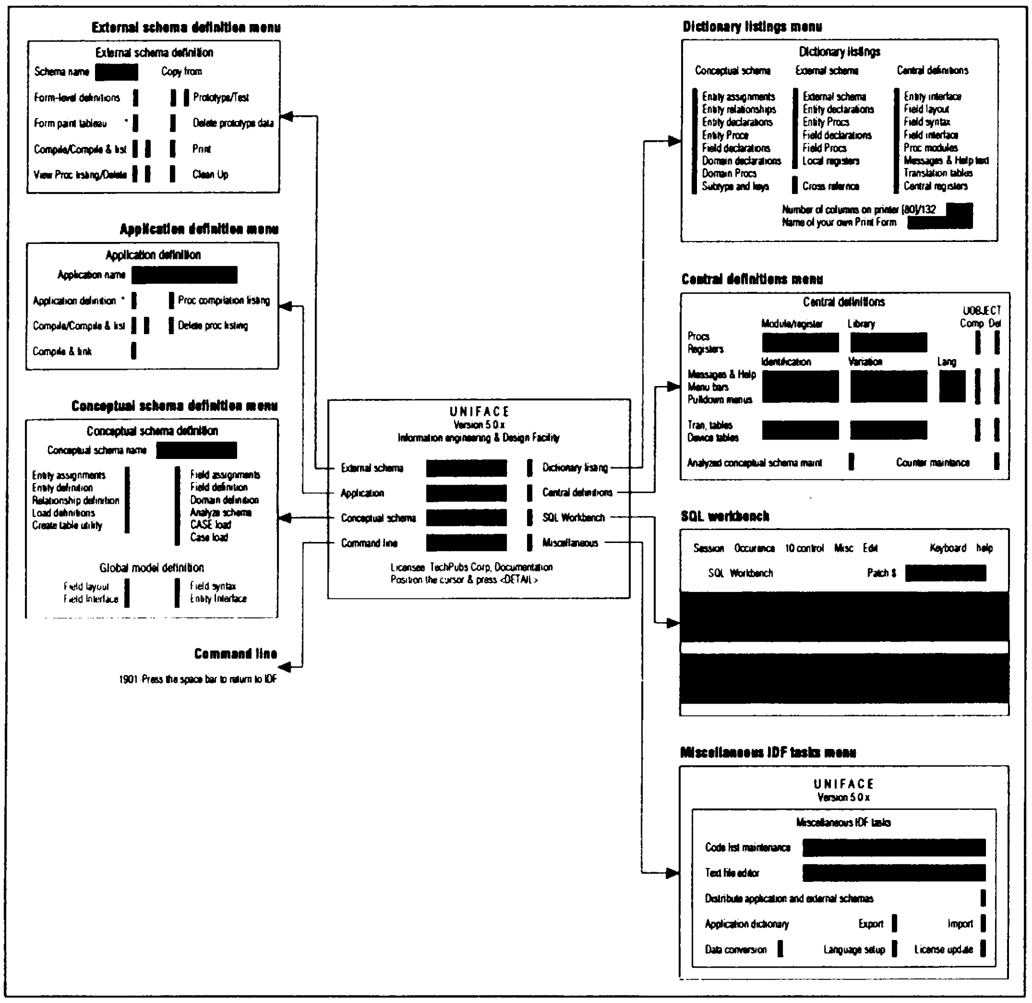


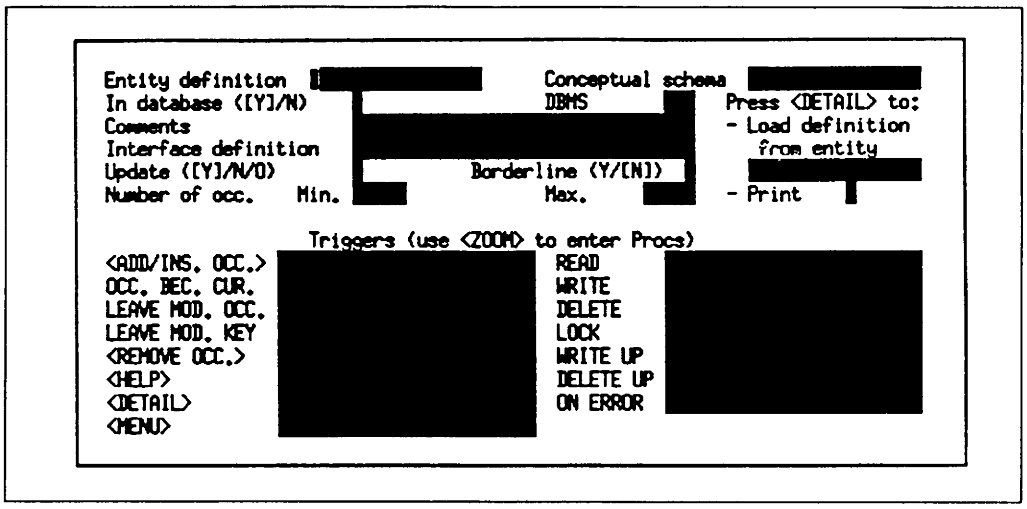
Illustration 5. This map shows the top layer of the Uniface IDF structure.

The Conceptual Schema: Defining Data and Relationships

CONCEPTUAL SCHEMA. The conceptual schema—the core of the Uniface application—is a collection of entity, relationship, and field definitions. An entity is usually equivalent to a database table. A set of entity definition forms allows the developer to specify attributes such as which DBMS controls the entity (DBMS type can be assigned at the table level), and to define event triggers at the entity level (see Illustration 6). These triggers are procedures to be executed when a particular event occurs. There are three levels of triggers: occurrence control (e.g., insert, modify, or delete a row, or a row becomes active); session control (the user presses the Help key or Detail key); and I/O control (e.g., read from or write to the database). When the user presses Help, the developer can call another form, a program, or a text message.

The developer also defines relationships between entities (primary and foreign keys) and referential integrity constraints for each relationship. Constraints are specified for deletes and updates on related tables; deletes and updates can be restricted, cascaded, or nullified (although Uniface supports only the restricted update in the current release).

Defining Entities



```
Entity definition                      Conceptual schema
In database ([Y]/N)                    DBMS                  Press <DETAIL> to:
Comments                                                     - Load definition
Interface definition                                           from entity
Update ([Y]/N/O)                 Borderline (Y/[N])
Number of occ.    Min.                 Max.                  - Print

                       Triggers (use <ZOOM> to enter Procs)
<ADD/INS. OCC.>                        READ
OCC. DEC. CUR.                         WRITE
LEAVE MOD. OCC.                        DELETE
LEAVE MOD. KEY                         LOCK
<REMOVE OCC.>                          WRITE UP
<HELP>                                 DELETE UP
<DETAIL>                               ON ERROR
<MENU>
```

Illustration 6. This screen shows the entity definition form. Triggers define processing that is executed based on the data element and the event that is triggered. In the case of the entity, the default data element is an occurrence, or row. Triggers are then defined based on actions that can be performed on a row in the entity.

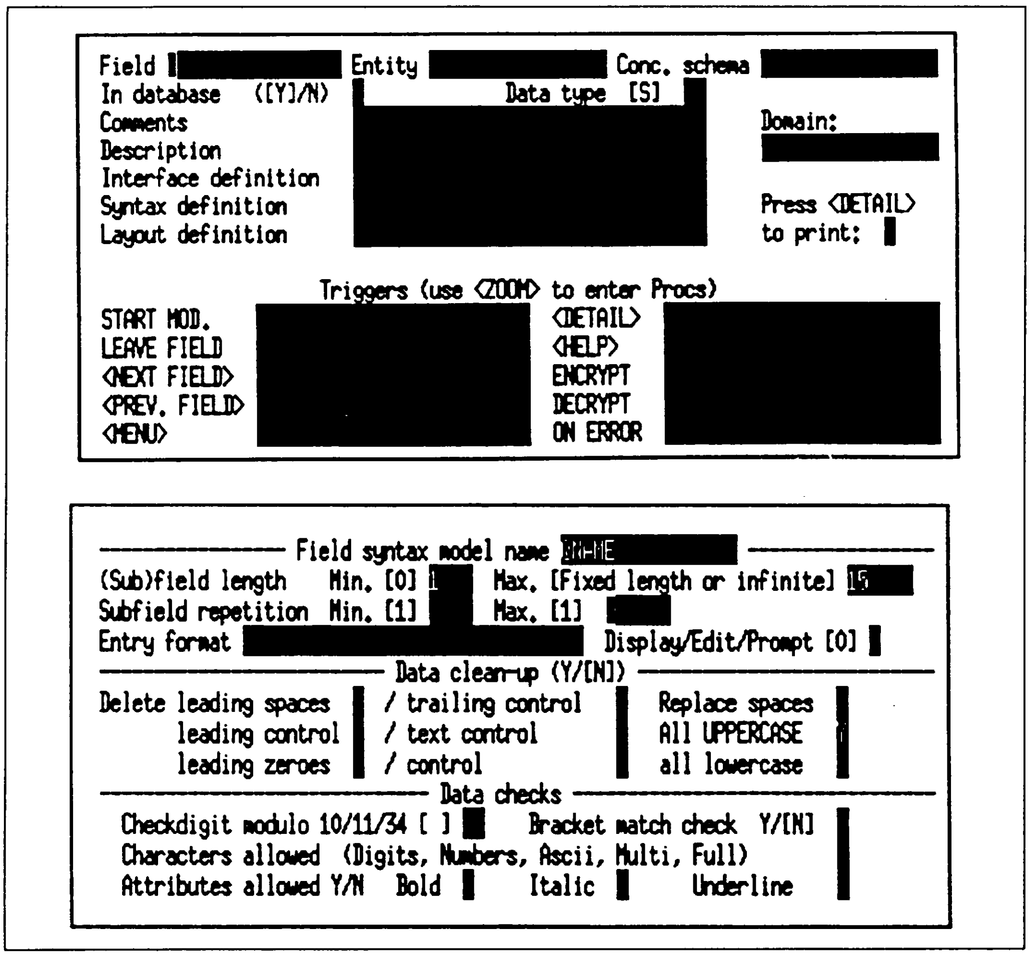
Fields. A set of forms similar to those for an entity are used to specify a field definition. Again, the developer indicates attributes (the data type, whether the field is in the database or calculated, a domain definition, etc.), field-level triggers, and many other aspects of defining a field. A domain is a generic field definition that can be applied to multiple fields. Illustration 7 shows two sample field definition screens.

Central Application Dictionary. It is easy to get confused about the role of the Uniface conceptual schema, or central application dictionary, versus that of the data manager's data dictionary. These are, in fact, two separate dictionaries. However, the conceptual schema is not an extra layer of information that an application must pass through before getting to the data manager. The conceptual schema is created primarily during the development process. Once the application is designed, the conceptual schema is used to create the internal schema—the actual database and its data dictionary—through the database driver. After the application is compiled, it goes directly to the database driver to access data; it does not have to touch the conceptual schema.

While the central application dictionary doesn't pose application performance problems, it does raise maintenance issues. After the internal schema is created, automatically reflecting subsequent changes to the conceptual schema in the database requires regenerating the internal schema. This may not be acceptable if there are already a lot of data in the database. Changes made directly to the database structure are also not reflected in the conceptual schema. In these cases, the two dictionaries must be synchronized manually. The changes to the database are made separately using DBMS tools or the Uniface SQL Workbench.

The conceptual schema is a standard set of tables that can be stored in any supported data manager. It can be centralized or distributed as necessary. All of the 4GL source code is stored in the conceptual schema as well. Uniface does not yet provide version control facilities for the source code. Source code can be stored in operating system files and the appropriate utilities used for version control.

Defining a Field



```
Field                   Entity                  Conc. schema
In database   ([Y]/N)           Data type  [S]
Comments                                                    Domain:
Description
Interface definition
Syntax definition                                           Press <DETAIL>
Layout definition                                           to print:

                        Triggers (use <ZOOM> to enter Procs)
START MOD.                              <DETAIL>
LEAVE FIELD                             <HELP>
<NEXT FIELD>                            ENCRYPT
<PREV. FIELD>                           DECRYPT
<MENU>                                  ON ERROR
```

```
―――――――――――― Field syntax model name NAME ――――――――――――
(Sub)field length     Min. [0] 1   Max. [Fixed length or infinite] 15
Subfield repetition   Min. [1]     Max. [1]
Entry format                           Display/Edit/Prompt [0]
―――――――――――――――――― Data clean-up (Y/[N]) ――――――――――――――――――
Delete leading spaces    / trailing control    Replace spaces
       leading control   / text control        All UPPERCASE
       leading zeroes    / control             all lowercase
―――――――――――――――――――――――― Data checks ――――――――――――――――――――――――
 Checkdigit modulo 10/11/34 [ ]     Bracket match check  Y/[N]
 Characters allowed (Digits, Numbers, Ascii, Multi, Full)
 Attributes allowed Y/N  Bold      Italic        Underline
```

Illustration 7. The top form is used to define a field in the conceptual schema and shows the field-level trigger events, including encryption. When the developer presses the <detail> key in the syntax definition field, the form below pops up.

External Schemas: Extensive Functionality in Forms and Menus

EXTERNAL SCHEMA. An external schema is simply a screen form for presenting data and interacting with the user. Main forms, code lists, detail forms, and menus are all external schemas. Uniface provides extensive functionality and flexibility in form design.

Forms and Frames. A form is separate from the screen, and it can be larger than its actual window on the screen (the form is scrolled within its window). The developer defines the size and position of the form window, and form windows can overlap. A form is made up of frames (windows that present entities and fields) and fixed text. Frames can also be layered within a form (frames within frames), indicating a data relationship. Since relationships are defined in the conceptual schema, they are supported automatically on forms without coding. Multiple independent detail relationships can be displayed on a single form. Frames are scrollable and can be zoomed to a larger size. Uniface offers full color support.

Menu bars and pull-downs are built separately from forms, enabling the developer to easily reuse menus.

Editing Data. The Uniface Structure Editor is a text editor that operates the same way in all external schemas. It provides form and frame travel functions, and text (e.g., a document) is handled the same way as structured data. One feature of Uniface that attracted some of its first customers—pharmaceutical firms such as Hoffman-La Roche and Sandoz—was its ability to store and edit long text fields.

The conceptual schema cannot be modified within the external schemas by the application developer, although the central definitions in the conceptual schema can, in many cases, be overridden (e.g., the spec that a field must be all uppercase). Some defaults cannot be overridden (e.g., a field data type). The developer can specify any definition not included in the conceptual schema. If the developer does override a central definition, Uniface will automatically note this in the documentation. The central definition is automatically restored if the developer removes the override.

Uniface Provides Application Management

DEFINING AN APPLICATION. It is interesting to note that none of the more popular Unix RDBMSs identifies an application and all of its components as an entity called an application. The developer cannot look at the data dictionary and see what reports, forms, procedures, etc. compose a particular application, such as accounts receivable. Yet that's what the developer develops! In contrast, Uniface provides a set of menus and forms for identifying an application. The application definition specifies all of the components of the application, application-level triggers, and a layout that applies to every screen (see Illustration 8). A VAR can easily customize the application for a particular organization without having to change the application itself, a very valuable feature.

Cross-Referencing. Another advantage Uniface provides is cross-referencing of fields, forms, and programs. Cross-referencing allows the developer to see, for example, all of the forms on which a field appears. Cross-referencing is optional and can be done dynamically (a 20 percent performance penalty is the cost of updating another set of catalogues), in batch, or periodically. If the developer changes something in the application, Uniface can specify what needs to be recompiled.

Customizing an Application

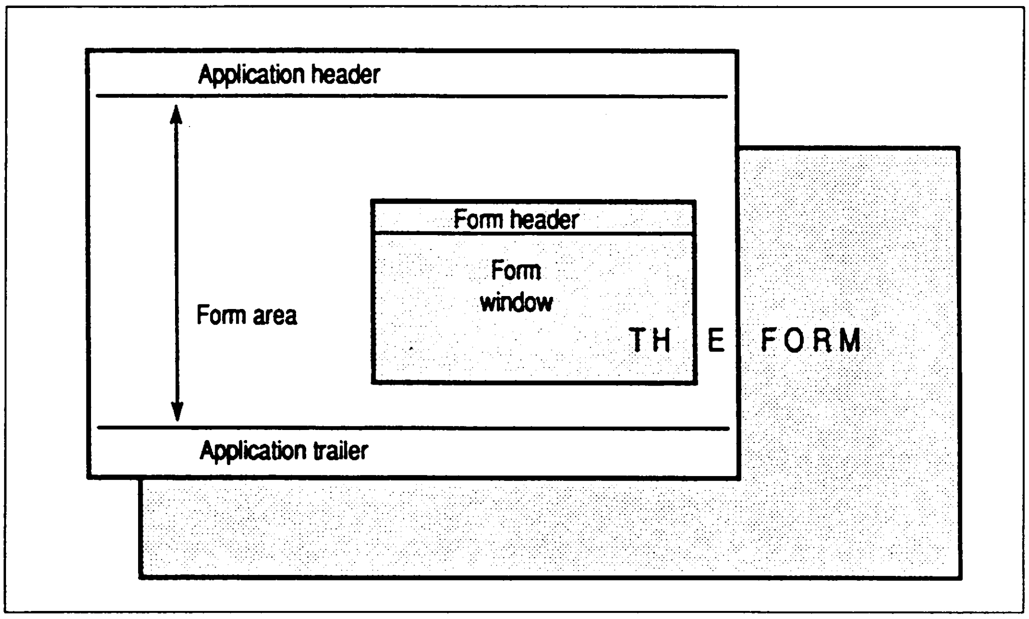


Illustration 8. The Uniface developer can define an application screen consisting of an application header and trailer and a form area. The forms defined in the external schemas will be displayed in the form area, and the application header/trailer can be customized independently from the application for a particular user organization.

The Uniface 4GL Lacks Some Basic Programming Constructs

PROCESSING LANGUAGE. One of the benefits of a good forms-based development environment is the ability to create a sophisticated application without doing much programming. One of the potential limitations is the inability to write standalone procedures and subroutines. This is true of products like Oracle's SQL*Forms, for example, and it can limit the developer's ability to generate complex, customized application processing without resorting to a 3GL program.

The Uniface processing language (called Proc Language) is missing some typical programming constructs, such as "do while" loops and case-branching (additional flow control will be added in Version 5.2). While it is possible to write subroutines in Uniface, they are not standalone procedures. Instead, the subroutine is created inside a trigger, and cannot be easily located or called up as a separate entity. A subroutine is executed with the "call" command; if the subroutine is not contained in the procedure/trigger that calls the subroutine, Uniface looks for a corresponding "entry" command somewhere in the library of triggers for the application.

While the developer can pass SQL statements directly to the RDBMS, there are some significant limitations here. Uniface does no syntax-checking on the SQL and cannot return a set of rows if the SQL statement is a "select" statement. It can only return two scalar values—an error code or the number of hits selected in the $status register, and the value in the first column of the last selected record in the $result register.

Uniface Terminology Supports Independence from DBMS

The Uniface product designers chose to use commands like Read, Write, and Erase instead of the SQL equivalents—Select, Update, Insert, and Delete. A major reason is the desire to hide SQL from the developer to the extent possible, and Uniface has gone out of its way to do this. Another reason is that Uniface runs against nonrelational products as well as relational. The developer defines an application the same way regardless of the back-end DBMS. A third reason is to avoid confusing the developer by using the same command verb but a different set of parameters. The Uniface Read command is not structured exactly the same way as the SQL Select command. Uniface does create SQL automatically from the application definition, and many developers get excited when they see how much is done for them without direct coding of SQL. A developer accustomed to writing SQL, however, may have difficulty with this approach and feel that there are two separate languages within Uniface.

Uniface has an extensive source code debugger and a macro capability. It supports variable indirection and an unlimited number of global and local variables. It has its own text editor in order to provide consistency and portability across all of its supported platforms. The developer can use a different editor, but doing so may lower performance.

Uniface Provides a Helpful Coding Structure

Adding Structure to the Coding Process. One of the nice things that Uniface does is prompt the developer during the coding process. For example, if the developer clicks on a field when painting it on a form, a coding window opens on the screen. It lists all of the events that can be triggered by the user on a field (the type of object the developer clicks on determines the contents of the code window). This is not only a handy reminder of the trigger options available, but it also minimizes the amount of code the developer has to write.

A Migration Strategy for Replacing 3GL Applications

Interface to 3GLs. In addition to the traditional hooks to 3GL programs (i.e., the 4GL can call a 3GL program), Uniface allows a 3GL program to call a Uniface application. Uniface is a subtask to the 3GL program (runs in the same address space) rather than a separate subprocess. The 3GL program can pass instructions and parameters to Uniface as well. One significant benefit is the ability to replace an existing program (Cobol, for example) with Uniface forms and code on a modular basis over time. For shops that are converting slowly, this is a very important feature of Uniface. Uniface supports C, Cobol, Ada, Fortran, and PL/1.

Uniface Performs Joins in Stages

EXECUTION OF SELECTS. When the user selects rows from multiple tables in a single query, Uniface does not perform the traditional relational join. First of all, the developer/user does not have to explicitly define a join. Uniface uses the entity relationships in the conceptual schema to understand how to perform the query. It then decides how to execute the select depending on what the user asks for and how the form is laid out. Basically, it goes to one table and retrieves a default number of rows (e.g., 50 rows); then it goes to the other table and gets the corresponding rows based on the join criteria. It then displays the retrieved records. When the user asks for more data, Uniface repeats the select process for the next set of rows. Uniface calls this a "stepped hit list," and uses a combination of its own optimization techniques plus those of the DBMS vendor to decide on its course of action.

The reason for the stepped hit list is to avoid the performance hit if there are many records in one or more of the tables, since users often abort queries after looking at some of the rows. If, in fact, the user does ask for all the rows, more resources are required to do the query this way, but Uniface thinks it is worth the risk.

Uniface Lacks a Comprehensive Report Writer

REPORTS. Uniface does not have a separate report writer. Generating a report in Uniface involves printing the data that would otherwise be displayed on a form on the screen. So the user defines a form as a series of frames to generate a report. You do have the facility to specify a page header, page trailer, and page numbers, and you can create break frames to show computed totals or subtotals.

Since you generally use a different conceptual model when designing reports than you do when designing forms, the Uniface methodology of using forms to generate reports seems somewhat awkward and complicated, and the output is not necessarily what you ideally want. The bottom line is that the report writer is good at producing a written copy of the data on the screen. Jim Milbery uses the term "online report writer." Printing a letter to confirm an invoice is just part of the application that invokes the print function. It is easy because the developer doesn't have to specify print positions and other time-consuming details. However, printing a long, 132-column, Cobol-type report is better done with a traditional report writer.

It Isn't Clear whether à la Carte Will Fill the Bill

We would like to see a real report writer (one that is easy to use, of course) added to Uniface. Uniface has recently announced à la Carte as an end-user report writer, but it is not clear yet whether that will suffice for the professional developer as well.

À la Carte Is the First Tool Aimed at the End User

À LA CARTE. Designed to be easy for anyone, from inexperienced end users up through application developers, à la Carte uses a "point and pick" approach. As in Uniface, pull-down menus and pop-up windows guide the user in selecting the data and formatting the query or report, and context-sensitive help is available.

À la Carte Eliminates the Need to Specify Data Relationships and Write SQL

Using à la Carte. The database or systems administrator first defines views for the user in à la Carte's Information Management module. À la Carte refers to an existing conceptual schema for data definitions (Uniface and a Uniface application—conceptual schema—must be available somewhere on the system). Since data relationships are defined in the conceptual schema, the administrator doesn't have to worry about this. The administrator also controls permissions to the views. Anyone with access to a view can also access any reports based on that view. In a future release, Uniface plans to implement report-level security.

With the views defined, the user is shielded from having to know how various parts of the database are related or structured, or where they are physically. And SQL is nonexistent in à la Carte. Neither the administrator nor the end user has to know how to write SQL statements.

After selecting a view, the user can choose report items (data fields and calculated fields), print order for columns, selection criteria, sort order, and aggregates (total, subtotal, average, minimum/maximum, and percent). The report can be previewed, edited, printed, and stored for repetitive use. À la Carte offers a list, or columnar, reporting style in the first release.

Behind the scenes, à la Carte generates a read-only Uniface form, or external schema, in standard, compiled 4GL code. The developer has access to the 4GL code, so the report can be included in a Uniface application.

À la Carte Is an Add-on to Uniface

Positioning. À la Carte is positioned as an add-on product to the Uniface environment. Having a separate, easy-to-use tool for creating ad hoc queries and reports is an important feature for both end-user organizations and VARs. Uniface will also offer à la Carte without the Information Management module on DOS and OS/2. This will appeal to VARs, who may not want to give the end user access to the detailed data structures.

À la Carte supports the same set of back-end data sources as Uniface. Built in Version 5.2 of Uniface, it will be available next month on DOS and VMS. Other platforms will follow depending on the porting schedule for 5.2. Initially, the interface will be the Uniface windowed character mode. As soon as the UPI and GUI drivers are available for 5.2, à la Carte will support Windows 3.0, Presentation Manager, OSF/Motif, and Open Look as well.

The Uniface Architecture

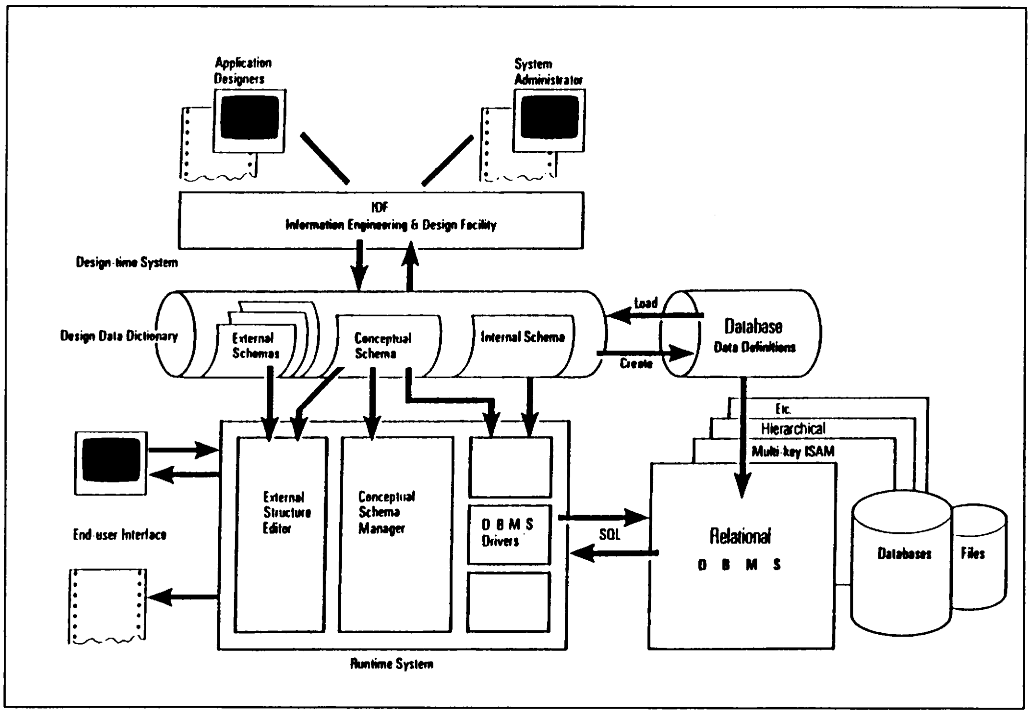


Illustration 9. This shows how the major components of the Uniface architecture—the IDF, the design (central) data dictionary, the run-time system, and the supported data managers—fit together.

Current Focus on Functionality Will Shift to Performance

PERFORMANCE. Through its current release, Uniface has been concerned primarily with functionality. Customers who really need what Uniface offers have been able to live with lower performance if necessary. However, the next release of Uniface (see "Futures" below) will implement a number of features to improve performance. The company points out that performance can also often be improved by taking advantage of additional hardware and memory.

Database Tuning. According to Uniface, in most cases, the DBA or system administrator will continue to use the DBMS vendor's interactive SQL product for database performance tuning. Uniface does not provide customized tools for this task.

Uniface Uses the Data Manager for Locking and Security

LOCKING. Uniface supports three types of locking, depending on performance considerations and the degree of reliability the user wants in the retrieved data. "Optimistic" locking (Uniface attempts to lock a row only when the user issues the Write instruction) increases performance in a multiuser environment. In "cautious" locking, Uniface attempts to lock a row as soon as the user modifies any data in the row. "Paranoid" locking assumes that there will be conflict and locks everything as soon as possible.

Uniface establishes a default locking mode for each data manager; the application uses the data manager's locking system in terms of locking level (e.g., row, page, or table) and lock types.

SECURITY. Uniface uses the security system supported by the underlying data manager.

Marketing Strategy

Marketing Goal: Increased Visibility

In the past, Uniface has not been particularly marketing oriented. The company is still relatively unknown outside Europe, and is working hard to remedy this. One approach is to beef up the direct sales force by adding three more U.S. sales offices by the end of 1991.

Other channels growing in importance are joint marketing partnerships, VARs, and the company's participation in the Sybase Open Tools program. Last month, Uniface joined the Informix InSynch program for ISVs, and we expect to see relationships with other DBMS vendors in the future.

Joint marketing partners include Data General, Digital Equipment, Hewlett-Packard, Index Technology, IBM, MIPS, NCR, Pyramid, Sequent, Stratus, Sun, Unisys, and others. An interesting note here is that these are all U.S. companies. Although the agreements were worked out with Uniface B.V. in Amsterdam, it is up to the U.S. organization to get them going and make them work.

It is to Uniface's benefit to accommodate as many back-end data managers as possible. While Uniface has developed a number of database drivers itself, it cannot possibly expect to do this for every DBMS product out there. So the company plans to make it easy for back-end vendors to develop the interface to Uniface (using the driver specifications and consulting if necessary) and market it themselves. Ideally, Uniface would like to shift responsibility to each back-end vendor for developing and updating its Uniface database driver. Keeping up with developments in multiple database engine products is not an easy task.

An example here is the relationship with Information Dimensions, announced last month. Information Dimensions markets the Basis+ full-text retrieval database product and claims to have over 50 percent of the text retrieval market. The company has developed and will sell its own Uniface driver. Uniface, as the front-end product for Basis+, will be sold by Uniface with a percentage going to Information Dimensions. Uniface benefits by expanding its list of supported DBMSs without the development cost. Information Dimensions gains a comprehensive development environment, a connection to the relational world and SQL, support for GUIs, and a bridge to help customers move from the original Basis-IR product to the newer Basis+.

VAR Program Will Be Enhanced

The ability to support multiple data sources on multiple platforms from within a single development environment has particular appeal to large VARs, system integrators, and large end-user companies. For these folks, the ability to deploy applications across multiple platforms cost-effectively is extremely important, and Uniface is working to provide this support. The first step is the ability to compile a Uniface application in the run-time version of the product; thus, the customer doesn't have to buy a development version of Uniface for every deployment platform. Version 5.2 of Uniface will add distribution utilities aimed at the VAR market.

Relationship with Sybase Is Helping Achieve the Marketing Goal

SYBASE. The relationship with Sybase is very important to Uniface, given both Sybase's popularity as an RDBMS and the fact that Sybase often sells into a heterogeneous environment. As part of the Sybase Open Tools program, the Sybase direct sales force can sell Uniface in addition to the Sybase SQL Toolset and tools from other program participants (Unify is the only other one right now). In fact, two of the four Uniface developers we contacted found out about Uniface through their Sybase salesperson. Uniface indicated that it is not likely to establish a similar direct-sales relationship with any of its other database partners.

Uniface and Sybase have stressed their commitment to a joint support program so the customer won't have to worry about dealing with a different vendor at each end of the application. The two companies have electronic access to each other's support systems in order to pass problem/solution data across, and both will participate in solving the problem until the customer is satisfied.

Support for Multiple Data Managers Is a Key Uniface Advantage

THE COMPETITION. Uniface competes directly with all of the 4GLs offered by the major DBMS vendors—Informix-4GL, Ingres's Application-By-Forms and Windows 4GL, Oracle's SQL*Forms, the Sybase APT-Workbench, Unify's Accell/SQL, Digital's Rally, and others. Uniface's obvious advantage is the ability to support multiple data managers concurrently from within a single development environment. Unify has versions of Accell/SQL for Informix, Oracle, and Sybase in addition to its own Unify 2000, but the Unify application can only access one DBMS at a time. The specific version of Accell/SQL for that DBMS is required as well.

Several of the DBMS vendors have gateways to other DBMSs, but they don't tend to support the direct competition (Progress is one exception, with a gateway to Oracle). The gateways are usually to DBMSs on proprietary platforms, such as Rdb/RMS on VAX/VMS, and DB2/IMS/et al. on the IBM mainframe, and they may be read-only (some vendors have implemented read/write gateways). The gateways also generally use a common subset of SQL as the access language and do not take advantage of proprietary extensions the way Uniface does. Finally, the scope of the data sources covered by a particular vendor's gateways is much narrower than that provided by Uniface.

Uniface also competes with other well-known third-party development products, such as Focus from Information Builders and Powerhouse from Cognos. A number of newer database-independent tools, such as JAM from JYACC, Powerbuilder from Powersoft, and Advanced Revelation from Revelation Technologies, are beginning to address Uniface's market as well.

Customer Concern: Using Third-Party Tools

Uniface is well aware of another area of possible "competition": the customer's concern over using tools from a different vendor from the one providing the back-end data manager. There is always the potential for finger-pointing between the vendors, an unpleasant experience for the poor customer caught in the middle. Uniface is addressing this issue through business relationships with the vendors whose products it supports.

Another concern with third-party tools is the need to keep releases synchronized and to provide access to new database features when they become available. Uniface is strongly committed to the following policy: the company will validate its driver against the new release of a DBMS within three months; if there are new features and functions that Uniface can't support, it will implement these within another three months.

Revenue Goal: 100 Percent Annual Growth Rate

FINANCIALS. Uniface is growing fast, having doubled its revenues and number of employees in each of the past two years. Revenues for fiscal 1990 were under $30 million, already comparable to those of Unify. At the present rate of growth, the company also has the potential to catch up in size with other competitors such as Progress Software ($40 million in 1990). One point to note in comparing Uniface with many of its competitors is the fact that all of Uniface's revenues are from development tools, and none from selling database engines.

Uniface At a Glance

International headquarters	Uniface B. V., Hogehilweg 16, 1101 CD Amsterdam, The Netherlands Telephone: 31 20 6976644
U.S. headquarters	1420 Harbor Bay Parkway, Suite 140, Alameda, California 94501 Telephone: (415) 748-6145
Founded	1984
Product first introduced	1987
Latest release	Uniface Version 5.1 (introduced September, 1990)
Financial	
Ownership	Privately held (employees 20 percent, management 40 percent, venture capital groups 40 percent)
Fiscal year	January 1 - December 31
Revenues	Under $30 million in 1990
Net income	Not available; company states that it has been profitable since 1988 at a rate of 15-20 percent of revenues
Geographic breakdown of revenues	
Europe	70 percent
Other	30 percent
Breakdown of revenues by channel	
Direct Sales	50 percent
Distributors	30 percent
VARs and OEMs	20 percent
Breakdown of revenues by platform	
Unix	40 percent
Proprietary, DOS, and OS/2	60 percent VAX/VMS, OS/2, DOS
Distribution channels	
U.S. sales offices	5 (Alameda, California; Boston; Chicago; Dallas; Bohemia, New York)
International sales locations	International headquarters in Amsterdam; sales offices and distributors in 22 countries
VARs	Number not available
OEMs	2 (Sybase and Delft Technologies)
Installed base	
Number of customers	Not available
Number of sites	Not available
Number of licenses	5,000 development licenses
Number of users	Over 30,000
Number of employees	175 worldwide; 30 in United States

Effective April 30, 1991

Illustration 10.

Futures

Uniface Plans to Enhance Both Performance and Functionality

Version 5.2 of Uniface is scheduled for release in September. It will implement several important enhancements, particularly in the area of performance:

Futures

- Significant performance improvements have been achieved by better tuning of the individual database drivers; Uniface will release new versions of the database driver for every supported product. Asynch interrupt enhancements will also help performance.
- Additional flow control constructs have been added to the 4GL, including "do while" and "repeat until" loops.
- The ability to change field attributes dynamically.
- A VAR distribution utility to make it easier for the VAR to package and deploy an application.
- Kanji support.
- Full support for GUIs.

END-USER STRATEGY. Uniface is considering several approaches to expand its end-user appeal. Two possible options are: integrating à la Carte with PC tools such as Lotus and Excel (another set of drivers, perhaps?), and allowing the user to update data with à la Carte (an "end-user's Uniface"). To some extent, the company is waiting to see what develops in the standards arena (e.g., from the SQL Access Group and others) before committing to a specific strategy.

The Developer's Perspective

Developers Are Excited about Uniface

We interviewed four developers who have chosen Uniface as their development environment. Three are end-user developers, and one is a VAR. Most had not even heard of the company before starting an evaluation of tools and database engines. Two of them were so impressed with Uniface that they made the decision to go with it in a very short period of time (one in under three weeks). One was predisposed not to like 4GLs and has already discarded the idea of using a 4GL. The strengths cited by the developers included:

- Uniface has a sound, solid product and good technical people to support it.
- The Uniface conceptual schema is a very powerful, event-driven development environment that automatically generates much of the SQL for the developer. Developers are able to do things with Uniface that they can't do with many other 4GLs.
- Uniface is truly portable across a wide variety of environments. Developers are impressed with the ability to import existing data definitions and to switch data managers without changing the application code. Uniface obviously understands databases and allows the DBMS vendors to do what they are good at. The way the company has built its database drivers is impressive (one developer ran forms side-by-side in different DBMSs, then put tables from the two DBMSs on the same form with no problems).
- Two developers described performance as good.
- Strong prototyping capability; after you compile an external schema, Uniface takes you right into the test mode.
- Good text-handling, with zoom feature.
- Uniface has few bugs, and, when there are problems, the developer can usually program around them.

The Developer's Perspective

- The ability to call Uniface from a 3GL was mentioned by two developers who need to migrate from 3GL applications.

Weaknesses Include Lack of Marketing Presence

Weaknesses cited included:

- Lack of size and presence in the U.S. market, and lack of a user group base. There was also concern about acceptance in the United States of a product developed in Europe.
- Lack of standard programming constructs (e.g., "do while" loops).
- Product requires training, and it takes a while to get going. One developer didn't like having to learn a new editor.
- There is too much functionality on the keyboard; Uniface needs a keyboard template to get the developer started.

One developer is eager to see GUI support released.

Conclusions

Uniface has created a development concept and product that can be immensely appealing to those struggling with applications development issues in a heterogeneous environment. This becomes clear in discussions with developers who have chosen Uniface as their development tool. A sense of excitement about what Uniface has to offer comes through. There is, in fact, a sense of incredulity that Uniface not only delivers on its promise of portability and independence, but also provides extensive functionality and flexibility in designing applications. As one developer stated, "It's as if we had given them the specs and they went out and wrote the product for us."

There is a large window of opportunity for a product geared so closely to what the developer is looking for, and Uniface has the potential to become a major force in the applications development market. By designing modularity and technology independence into the basic product architecture, the company has built a strong foundation for future success. While Uniface isn't perfect by any means, its strengths will serve to raise the ante in the increasingly competitive, and important, market for application development tools. The company is well aware of the classic challenge it faces—to improve market visibility while keeping up on the product development side. And if Uniface is successful in this endeavor, it will help push the entire tools market in the right direction. ◐

Next month's *Unix in the Office* will address
Digital's Open Systems Strategy.

For reprint information on articles appearing in this issue,
please contact Richard Allsbrook at (617) 742-5200, extension 116.

Evaluating Application Development Tools

Here is a basic framework for evaluating application development tools, a set of five dimensions to consider in making your decision. This framework is not intended to be exhaustive, but rather a set of guidelines that raises important issues and can be extended over time.

Dimension 1: Environments Supported

Make sure you know how the application development environment matches your own in terms of:

- Hardware and operating system(s) on which the product runs.
- Communications/networking protocols supported.
- Database management systems (DBMSs) supported. Are there separate versions for different DBMSs, or does a single version run against data stored in multiple DBMSs? Is there support for a distributed database? What is the method of communication with the DBMS (native SQL, SQL translated to native SQL, common subset of SQL)? What happens if the developer uses enhancements, extensions, or other proprietary functionality in building an application? Is portability lost?
- Graphical user interfaces (GUIs) supported. Does the product have its own GUI, and can the developer create applications that adapt to one or more GUIs within the user environment (presentation independence)?
- Integration with other tools—CASE, 3GLs, end-user tools, etc.

Dimension 2: Development Life Cycle Phases Supported

What phases of the development life cycle are covered by the tool(s)? If there is partial coverage, do you need a broader set of tools to fill in the gaps? If you do, the integration issue becomes an important one to streamline the process and improve productivity.

Several models of the application development life cycle are available, including IBM's AD/Cycle. In general, all of them cover essentially the same phases, although the terms and separation of phases may be somewhat different. These phases include:

- Requirements/analysis and design. This phase encompasses the definition of business requirements and initial system analysis and design. CASE tools and structured design methodologies are often used in this phase of the design cycle.
- Application development and debugging.
- Testing and tuning.
- Deployment.
- Operational control (production management).

Dimension 3: Target User

What type of user is the product designed for, and what are the user's requirements?

PROFESSIONAL DEVELOPER. The professional developer can be cut a number of different ways. One is the scope of responsibility within the application development life cycle:

- In a small company, one person may end up doing everything—designer, developer, database administrator, system manager. This *generalist* usually wants a lot done automatically (e.g., good defaults) and is willing to trade off the degree of control over the system. He or she simply does not have enough time to do everything.

- In a larger company, there may be several or hundreds of developers. Here, the developer tends to be more of a *specialist*, with responsibility for a specific task or set of tasks—paint screens, develop data dictionary elements, generate math functions, etc. The developer wants separation and independence from other development areas, and tools that provide specific functionality for the task at hand. In this environment, there is the need for fewer defaults, more control over the development and production environments, and management tools for the development process itself (e.g., screen and program locks).

- The value-added reseller (VAR), in addition to standard application development tools, has important requirements in the deployment area as well. Here, there is the need for tools that make it easy to do things such as make global changes to the application for a particular customer and then support those changes. The VAR is also concerned with how ad hoc tools can be deployed to the customer. Can the end user buy just the ad hoc query/reporting tool as an adjunct to the VAR application rather than having to buy the entire development system?

Some of the application development tools aimed at the professional developer/VAR include:

- Forms-based application development environment
- Schema designer or data dictionary tool to create and modify the database structure
- Prototyping tools
- SQL debugger/editor/performance monitor
- Development language (4GL)
- Hooks to 3GL
- Report writer
- Gateways to heterogeneous data sources
- Facility to identify the impact of changes made to an application
- Application management (different people working on different applications, but forms or other parts of the application are shared across applications)

END USER. End-user needs include tools for decision support (ad hoc query and reporting) and applications development. The end user can also be segmented a number of different ways. Here is one example.

- The inputter, who uses a structured application to enter and update data and execute queries.

- The information provider, who may also require an ad hoc query/report tool, but the access to data is still relatively structured. It is also generally defined by those requesting the data.

- The power user, who follows a line of thinking/analysis in examining data. The objective is to arrive at a decision or recommendation and to communicate that result to the appropriate people.

Tools aimed at the end user include natural language interfaces, spreadsheet, ad hoc query/reporting tools, display graphics (which can be a powerful tool in ad hoc analysis), and statistical packages.

DATABASE ADMINISTRATOR. The database administrator tends to be left out in the cold by most RDBMS vendors. This person requires administrative tools for managing a production environment: applications, users, resources, networks, security, and devices. The ability to monitor user-defined thresholds (let me know when I reach a point where a problem might occur, e..g., the disk is 80 percent full), security permissions (who granted access and why), system-wide performance, and the ability to identify the impact of changes are all important.

Dimension 4: Breadth/Depth of Functionality

Broad issues here include:

- Is the application model supported interface driven or language driven?
- Is the data model supported relational tables (now the "traditional" model) or objects? (For more information on this area, see "Object-Oriented Development" in the *Office Computing Report*, March 1991, Vol. 14, No. 3.)
- How do you design and build applications? Does the tool support prototyping? To what extent? How easy is it to prototype an application?
- What are the power and depth of the development environment? Does it cover all required application functionality?
- Is the performance acceptable (ideally, comparable to a 3GL)?
- What international language support is offered?
- What is the quality/reliability of the product?

Dimension 5: Business Issues: Vendor Evaluation

Last, but not least, is the evaluation of the vendor's stability.

- What is the long-term viability of the tools vendor (financial and technological)?
- In the case of a third-party tool, is the tools vendor willing to work closely with the database vendor and its customers? Concerns here are coordination of new releases, platform roll-out, and joint support and maintenance. ◐

Open Systems: Analysis, Issues, & Opinions

DATABASE WATCH

Sybase: A Better Network Player

Sybase recently announced a few developments that should strengthen its distributed network computing strategy. Specifically, the company has increased its support for security and has released a set of networking interfaces that integrate PC applications with Sybase's SQL Server. Neither of these is a major announcement in and of itself, but both are an indication of Sybase's distributed computing directions and progress, and we're encouraged by the company's developments in this area.

Security

Sybase's Secure SQL Toolset was actually released last year to be used with the company's then-new Secure SQL Server, a multilevel security version of Sybase's regular SQL Server. (The Secure version enforces access control, security audits, and separate user and administrator roles.) Recently, the toolset has been made available for additional secure platforms, including Digital's Secure Environment Virtual Monitoring System (SEVMS) and Sun's Multi-Level Secure Operating System (OS/MLS). Both these systems are used mainly for government installations that require at least a C-2 level of security (as specified by the National Computer Security Center's "Orange Book"). Essentially, Sybase has now correctly targeted the toolset to run on the high-security platforms where it makes most sense.

The Secure SQL Toolset can be used to build Sybase applications that have a B-1 level of security. In other words, the tools enforce access control by allowing the developer to prescribe various user security levels. Three secure Sybase application development tools are available:

- APT Workbench, an integrated suite of window-based tools for designing, prototyping, and maintaining forms-based applications

- Data Workbench, decision support and data administration tools

- Open Client Interface, an application programming interface that allows non-Sybase tools and application programs to communicate with the SQL Server

The pricing of these tools can range anywhere from $2,400 to $115,200, depending on product component, CPU size, and hardware platform.

NOT JUST FOR GOVERNMENT. Sybase talks a lot about its secure toolset in conjunction with military and government installations, which makes sense, considering that these sites are the most visible in terms of security demands. However, as distributed network computing and inter- and intra-enterprise communication continue to proliferate, we think that the commercial market will become just as demanding. Products like Secure SQL Server and SQL Toolset could put Sybase in a favorable position when more commercial users realize that they need to protect sensitive RDBMS data better.

Network Interfaces

Sybase's new PC Net-Library is a critical element for integration and downsizing computing environments. It's a set of network modules designed for different network protocols that allow PC applications to be integrated with a Sybase server—which can reside on a number of platforms: OS/2, Unix, VAX, MVS. The product allows developers to build applications on a PC and deploy them across various server platforms and network protocols. It handles all the network communication between the desktop application and the SQL Server platform.

PC Net-Library was designed to be network independent, and it supports over 20 network interfaces, including TCP/IP, DECnet, LAN Manager, NetWare, and LAN Server, as well as implementations from Sun, 3Com, FTP, and AT&T. Each interface is a separately-sold module; your purchase depends on the hardware and network configuration. However, you can run an application over multiple network protocols. You just have to buy and install a different module for each network and specify the appropriate protocol when you run the application. That is, you don't need to maintain a different version of your application for each network your organization supports.

AVAILABILITY. PC Net-Library is available for DOS, Windows, and OS/2, and works with all Sybase-supported applications on those platforms, including Sybase applications, custom C-language applications, and even some third-party applications (such as Lotus, Paradox, Excel, and DataEase). The price is $145 per network interface.

Waiting for Distributed Databases

Sybase's existing Open Client/Open Server architecture, obviously enhanced by these two smaller announcements, gives the company solid footing in the area of distributed computing. But one area in which Sybase lacks competitive functionality is support for distributed databases. Sybase doesn't yet support distributed queries, and features only a *client-based* two-phase commit (2PC) protocol, rather than a server-based 2PC, which is more appropriate for a distributed database environment.

Sybase certainly intends to support these features in the near future, but its distributed database support will extend to non-DBMS data as well—information from a mail server, for example, or a news feed, or a custom application. This support for non-SQL data is where Sybase really excels. Once Sybase couples its functionality with distributed databases, it could have a very convincing distributed network computing story to tell indeed. —*L. Rowan*

DEVELOPMENT TOOLS

New Life for Cobol Applications

Part of the problem in migrating to open systems and new platforms is existing applications. Typically, organizations have sunk such huge investments in their core applications that it's simply not worth the effort or expense to adopt new software solutions that won't work with the applications. There are literally billions of lines of Cobol out there, and, unfortunately, Cobol is even being used for new commercial application development. Rewriting all these applications into a practical, portable language like C is just not an option. Instead, what users need are tools that leverage their existing applications to participate with other technologies that may run on other platforms. Ideally, your mainframe Cobol applications would be able to access information stored in an Ingres database on a Unix file server.

Accessing Databases through Cobol

Therefore, we were particularly interested in a recently-released product from Ryan McFarland (Austin, Texas) that lets users link their Cobol applications to relational databases using standard Cobol indexed files. Ryan McFarland describes RM/plusDB as a "database application enabler." Essentially, the tool provides a transparent database interface to Cobol applications, thereby making the Cobol applications more useful and flexible.

How It Works

RM/plusDB is meant to be used with Ryan McFarland's Cobol-85 system. An RM/plusDB client module is embedded in the Cobol-85 run-time system. An RM/plusDB server acts as a gateway that then translates your application's Cobol indexed file requests into database requests and vice versa. Thus, your Cobol application can reach database files, and your database application can reach Cobol indexed files.

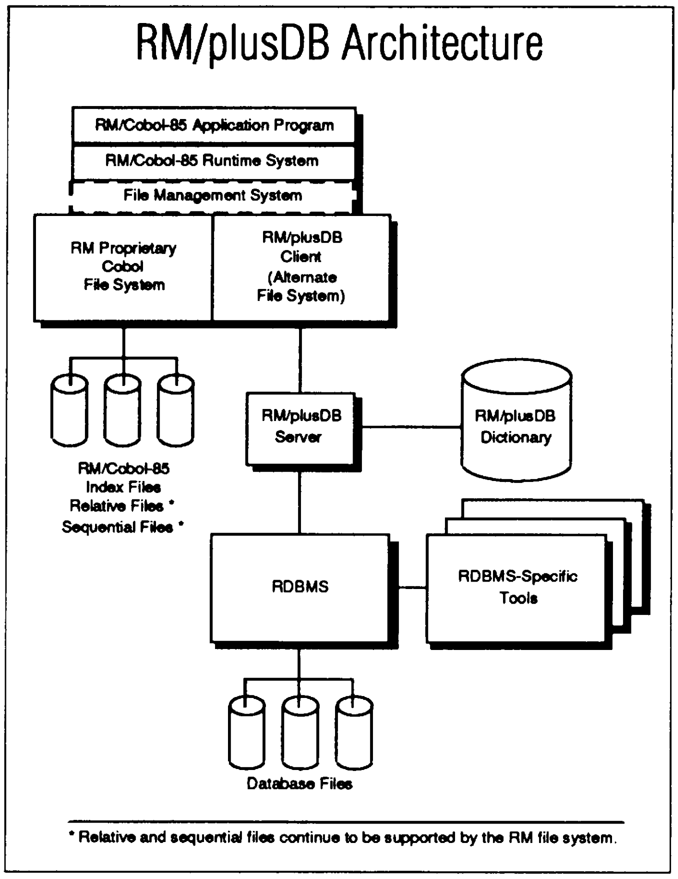

RM/plusDB Architecture
RM/Cobol-85 Application Program
RM/Cobol-85 Runtime System
File Management System
RM Proprietary Cobol File System
RM/plusDB Client (Alternate File System)
RM/Cobol-85 Index Files Relative Files * Sequential Files *
RM/plusDB Server
RM/plusDB Dictionary
RDBMS
RDBMS-Specific Tools
Database Files
* Relative and sequential files continue to be supported by the RM file system.

Illustration 1. An application can access information from a database table or a Cobol indexed file.

The server is a database-specific gateway; you can't access heterogeneous databases simultaneously. (Actually, the product currently only supports Informix, but others are on the way. See "Availability" below.) However, RM/plus DB does allow a single application to draw from multiple database tables as well as Cobol data files (see Illustration 1, page 28). The system creates and stores files in both a database format and a Cobol indexed file format, allowing the application to reach both resources.

The system is fairly extensive and contains a number of utility programs for database creation, modification, duplication, and recovery. These tools are interactive; you can use them to build databases and tables on the fly as well as to access and manipulate existing data.

DIFFERENT FROM EMBEDDED SQL. Ryan McFarland's approach is different from the more traditional means of Cobol/SQL integration, where SQL statements are embedded into the Cobol application, resulting in the maintenance cost of supporting two sets of source code. You actually don't have to know anything about SQL. Users can just stick to their standard Cobol syntax and let the RM/plusDB server take care of the translations. For example, Cobol's concept of a collection of related files is translated into the relational concept of a database; Cobol's file, into a relational table; Cobol's record, into a relational row; Cobol's field, into a relational column; etc., etc.

Availability

RM/plusDB is limited to applications written with RM/Cobol-85. It currently supports Informix databases, but Ryan McFarland should be releasing a version for Oracle in the near term. (Additional RDBMS support is planned, but none has been announced.) The price ranges from $1,400 to $6,500. —*L. Rowan*

HEWLETT-PACKARD

RISC Meets the X Terminal

X terminals have long been criticized for their sluggish performance and their drain on network resources—a valid criticism that has marred their reputation in the commercial market. However, recent developments in X terminal technology may make them a more viable option for large-scale implementation. For example, in January, NCD announced a window manager-resident X terminal that spares the machine from calling on the host for window operations, thereby signficantly reducing network congestion. (We talked about NCD in some depth in the February issue; see Vol. 6, No. 2.) More recently, Hewlett-Packard introduced a new family of X terminals that are based on the Intel i960 RISC processor—a considerable performance benefit.

RISC-Based X Terminals

RISC adds to HP's new line of X terminals the same benefits it has brought to Unix servers and workstations: increased performance. HP claims that its 700/RX family can reach as much as 70,000 xstones at the high end (xstone ratings are a standard benchmark for X server products). 700/RX stations could provide a critical performance margin for compute-intensive graphical applications like CAD/CAM, CASE, statistical modeling, or electronic publishing.

The 700/RX family includes three basic model types that vary in terms of price and power:

- Mi, with a 20 MHz processor, 2MB dynamic memory (DRAM), .25MB video memory (VRAM), and an optional 19-inch monochrome monitor. Base price: $2,395. With monitor: $2,995.
- Ci, with a 20 MHz processor, 4MB DRAM, 1MB VRAM, and an optional 16-inch color monitor. Base price: $2,995. With monitor: $4,495.
- Ca, with a 22.7 MHz processor, 4MB DRAM, 2MB VRAM, and optional 16-inch and 19-inch color monitors. Base price: $3,895. With 19-inch monitor: $5,995; with 16-inch monitor: $5,195.

Comments

HP's new X terminals are pricey. Competitors are selling low-end X terminals for less than half the price of the Mi model. $1,000 is the price to beat in the low-end X terminal market, and, if you include the monitor, the Mi costs almost three times that much. Evidently, HP thinks the performance of the 700/RX is worth the cost. We're not so sure. While users might well find the additional power and performance attractive, justifying a $2,995 monochrome X terminal when they can get a cheap SPARC workstation for under $5,000 might be difficult.

The bottom line: Coupling RISC with X servers is a good idea, and no doubt HP will lure quite a few customers with the sheer power of the 700/RX. However, we think a more competitive pricing structure would make the product line much more successful.

—*L. Rowan*

DATA GENERAL

DG's New Office: CEOMail, Aviion, and NetWare

Data General's AV Object Office is further evidence that the company has returned to its roots to seek success in the '90s. AV Object Office is a large software-integration effort that delivers an office platform built on NewWave 3.0 from Hewlett-Packard, NetWare from Novell, and Aviion Unix servers from Data General.

Data General's value in this equation is its Unix-based Aviion servers, electronic mail software, and software integration. (See Illustration 1.) We like the choices Data General made among available standards upon which to implement AV Object Office. The environment is the first we've seen from a large systems vendor that combines NewWave and NetWare. Competing environments from HP (NewWave Office) and NCR (Cooperation) use LAN Manager as a network operating system.

Where DG AV Object Office Adds Value

Feature	Explanation
Enhanced Filing	1. Addition of public and private folders to NewWave's Folders. 2. Improved concurrency for shared and network filing.
Enhanced E-mail	1. Addition of personal, workgroup inbox, inbox wastebasket, and "postcard" quick message to MHS. 2. Global, local, and personal user directories.
Status Window	1. Real-time mail status in independent window. 2. Server connection status/control.
Enhanced Printing	1. Support for remote printers.
Prebuilt Agent Tasks	1. Empty wastebasket. 2. Open user profile. 3. Open terminal emulation session. 4. Create folder. 5. Others.

Illustration 1.

What Is AV Object Office?

AV Object Office is a client/server office system built on Hewlett-Packard NewWave for DOS clients, Unix-based Aviion servers, and NetWare for Aviion, an implementation of Novell's Portable NetWare. Data General integrated this mix of software, enhanced it, and provided a robust mail system to run on it. (See Illustration 2.)

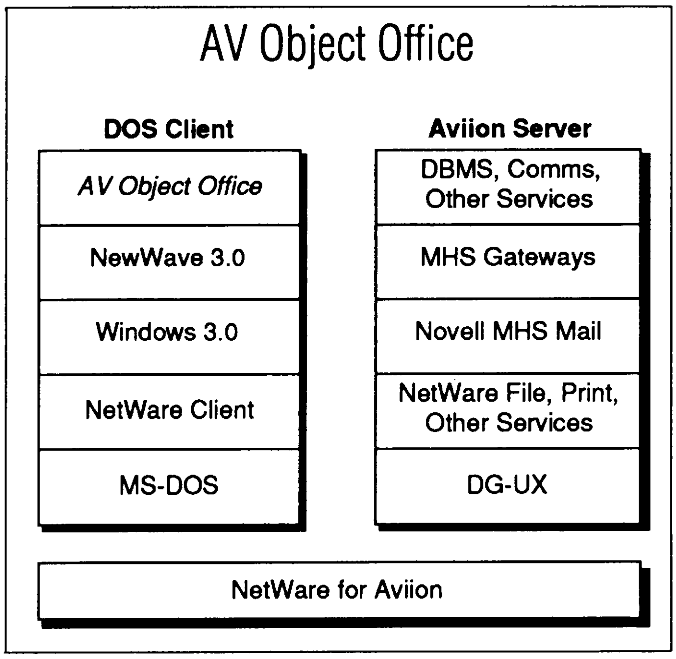


Illustration 2. In building AV Object Office, Data General sought to use as many standard or de facto standard products as possible. Here is the way the software is configured on clients and servers. Data General's value in the software package is its enterprise mail system and extensions to the HP NewWave client environment.

MAIL. The main reason to buy the first release of AV Object Office is mail. AV Object Office Mail is a combination of Message Handling Service (MHS), Novell's store-and-forward protocol, and directory services and a mail client built by Data General.

This is an interesting approach. MHS is widely used by virtue of its inclusion with NetWare platforms. Yet it is just a protocol, and users need much more than a store-and-forward protocol to build robust mail systems. AV Object Office Mail adds two key ingredients to MHS to make it a mail solution for large installations. Both additional ingredients are borrowed from Data General's earlier CEO office automation software for the MV minicomputer series.

First, Data General implemented an "enterprise" directory service atop MHS. The directory service sits on the Aviion server. It is designed to manage large numbers of users—more users than NetWare's current directory service. Novell is working on a larger directory service for NetWare but hasn't shipped it yet.

Second, Novell doesn't provide a state-of-the-art mail client for MHS. It leaves mail clients to third parties. Data General's AV Object Office Mail client implements features from CEO Mail, including workgroup inboxes and "postcards," or short messages, within the NewWave environment.

Data General's use of MHS as its transport also gives it a variety of third-party gateways to other mail systems to deploy on the Aviion server. Later this year or in early 1992, Data General expects to adopt the Unix-based mail gateway under development by Soft•Switch of Wayne, Pennsylvania. Soft•Switch's Unix gateway will link AV Object Office, via MHS, to more than two dozen mail systems, providing directory synchronization services with several major mail systems as well.

ENHANCED FILING AND PRINTING. Data General enhanced NewWave 3.0's filing features in AV Object Office. It added public and private versions of NewWave's Folders and improved on NewWave's network filing concurrency model.

In addition, AV Object Office adds remote printer support to NewWave 3.0's printing features.

APPLICATIONS AND INTEGRATION. The mail client is the most important new application Data General ships with AV Object Office. In addition, Data General provides a "Status Window" application that displays the status of a user's mail inbox in real time and shows the status of the client's connection to the Aviion server. Lastly, Data General wrote a handful of predefined Agent Tasks with NewWave's Agent facility.

AV Object Office gives users access to both the DOS and Unix environments by virtue of its Portable NetWare base. DOS users can gain access to Unix applications via terminal emulation.

Relevance to Customers

AV Object Office is bait to lure users of Data General's MV minis and new customers to bite at Data General's Unix servers—and they just might.

For customers with commitments to DOS PCs as their workstations of choice and to Novell NetWare, AV Object Office is a painless choice. Because it is built on commonly used products, AV Object Office will require little change to current configurations. Users may have to upgrade their PCs to 80386s with at least 4MB of memory to accommodate NewWave, and Data General may be a new supplier of servers. These are small changes compared to competing alternatives from Hewlett-Packard, NCR, and others.

AV Object Office is also a relatively inexpensive solution. Prices for the complete software package range between $350 and $500 per seat.

Data General is seeking an "incremental" approach to building advanced office environments. The company wants to build on existing products, rather than to replace what exists in the office with new hardware and/or software. Many users view this approach as less risky than committing to a new platform that requires wholesale change.

Data General's incremental approach sacrifices some functionality, at least in the short term. What do you really get with AV Object Office? A robust mail system and a platform for NewWave applications. That's it. There are no next-generation object-oriented tools and no workflow automation applications included with the platform. Nor are there new APIs.

By contrast, HP, NCR, IBM, Lotus, and other vendors hawking advanced office systems are each seeking to provide a combination of advanced applications function and APIs or programming environments with which to create new corporate applications. AV Object Office seems to offer a less rich platform for custom applications than such alternatives as NCR Cooperation, Lotus Notes, and HP NewWave Office.

It seems to us that buyers of AV Object Office will meet three criteria:

- They will be committed to NetWare and MHS as their long-term platform for client/server applications.
- They will be comfortable signing on Data General and Soft•Switch as their enterprise mail suppliers.
- They will want risk-free, LAN-based office systems, installing mail and personal productivity applications today while figuring out which advanced applications (groupware, business monitoring, imaging, etc.) make sense for the future.

—*J. Rymer*

Patricia Seybold's Computer Industry Reports

ORDER FORM

Please start my subscription to:

			U.S.A.	Canada	Foreign
☐	*Patricia Seybold's Office Computing Report*	12 issues per year	$385	$397	$409
☐	*Patricia Seybold's Unix in the Office*	12 issues per year	$495	$507	$519
☐	*Patricia Seybold's Network Monitor*	12 issues per year	$495	$507	$519
☐	*Paradigm Shift—Patricia Seybold's Guide to the Information Revolution*	10 issues & tapes per year	$395	$407	$419
☐	*Paradigm Shift—Patricia Seybold's Guide to the Information Revolution*	10 issues per year	$295	$307	$319

Please send me a sample of: ☐ *Network Monitor* ☐ *Office Computing Report* ☐ *Unix in the Office* ☐ *Paradigm Shift—Patricia Seybold's Guide to the Information Revolution*

Please send me information on: ☐ Consulting ☐ Special Reports ☐ Conferences

☐ **My check for $________ is enclosed.** ☐ **Please bill me.** ☐ **Please charge my subscription to: Mastercard/Visa/American Express** (circle one)

Name: ____________ Title: ____________

Company Name: ____________ Dept.: ____________

Address: ____________

City, State, Zip code: ____________

Country: ____________ Bus. Tel. No.: ____________

Card #: ____________

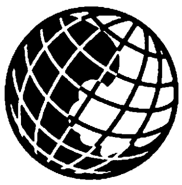
Exp. Date: ____________

Signature: ____________

Checks from Canada and elsewhere outside the United States should be made payable in U.S. dollars. You may transfer funds directly to our bank: Shawmut Bank of Boston, State Street Branch, Boston, MA 02109, into the account of Patricia Seybold's Office Computing Group, account number 20-093-118-6. Please be sure to identify the name of the subscriber and nature of the order if funds are transferred bank-to-bank.

Send to: Patricia Seybold's Office Computing Group: 148 State Street, Boston MA 02109; FAX: 1-617-742-1028; MCI Mail: PSOCG
To order by phone: call (617) 742-5200

IU-0691



Seybold Executive Forum

Workgroup Technologies and Organizational Learning

October 14-16, 1991, Glen Cove, Long Island, New York

Unlike most industry conferences, where you sit on your duff, listening to self-proclaimed "experts" hype themselves and/or their products, this year's Executive Forum will provide exposure to new technolgies, products, and tools in an experiential workshop.

Your objective during the three-day session is to design a solution to a real-world business problem using new workgroup technologies, business process design, and organizational learning methodologies. The problems presented are real; the users living with these problems will present them and will be on hand to act as resources as solutions are being designed.

You'll work together in high-performance learning teams of people who, like yourself, are exploring the intersection of organizational effectiveness and technology. Participants can choose to act as solution seekers (working in teams to design a solution) or solution providers (vendors of either a technology or methodology that can be incorporated as part of the solution)—a case where getting there is most of the fun.

Attendence is by invitation only. If you are interested in receiving an invitation, please fax back:

To: Patty Seybold Fax # (617) 742-1028 Phone # (617) 742-5200

From ____________ Fax # ____________

Title ____________ Address ____________

Organization ____________

Phone # ____________

I am interested in participating in the Executive Forum. I'd be most interested in: (check one):

❐ Responding to the real-world problems as a solution provider (software, consulting, organizational learning methodology)
❐ Being a member of a solution design team