Patricia Seybold's

UNIX IN THE OFFICE

VOL. 5, NO. 9
ISSN: 0887-3054
SEPTEMBER 1990

INSIDE

COMMERCIAL APPLICATIONS • TOOLS • TRENDS

The Progress 4GL and RDBMS

Shooting for the Mainstream

By Judith R. Davis

THE PROGRESS 4GL and relational database management system (RDBMS) from Progress Software has come a long way since our last review in February 1988. At that time, we complimented Progress on its elegant development environment, but also stressed the hefty challenges it was facing. Progress needed to implement a number of major enhancements to remain competitive—SQL, heterogeneous networking, a multithreaded/multiserver architecture to better support online transaction processing, distributed database, and access to non-Progress databases. *(Continued on page 3.)*

• E D I T O R I A L •

How Much Standardization Is Healthy?

By Judith S. Hurwitz

ONE OF the comments I commonly hear from users is that they want one and only one version of Unix. They are frustrated at the bickering between the Open Software Foundation (OSF) and Unix International. They want the two to merge and get on with the task of achieving one standard version of Unix that all will agree upon for the future. In theory, this would work, but, in reality, such a move would stifle creativity. Even if creativity could be maintained, the process of satisfying every application requirement with one large operating system would overwhelm the industry. In this fast-paced industry, there just isn't enough time. And even if it could be achieved, would it be healthy?

Just look at the Posix specification for a clue to the answer. Posix would serve as that common interface between applications and operating systems. But, though the Posix process is open and publicly accessible, it is painfully slow. Those looking to Posix to solve the industry's problems are becoming disillusioned. The definition of what should be included in Posix continues to expand. What started out as a single committee has grown into at least 23 different committees, each intent on solving another part of the computing puzzle. To the IEEE's credit, two of the key components of Posix have been ratified—the system interface specification (1003.1) and the user shell specification (1003.2). These two specifications will help ensure consistency at the API level.

However, the two specifications only take care of base operating system technology; they do not anticipate new functionality. Therefore, the ability to provide innovation becomes an issue. Now, no one would ever admit to being against innovation. But if users in this industry demand one implementation of an operating system, then innovation must take a back seat. Here's why. Let's say two systems vendors, Vendor A and Vendor B, have both adopted a standard version of an operating system. As long as Vendor A and Vendor B develop the same technology that both have used for the past several years (relational databases, word processing, accounting systems, etc.), everything is fine. But what happens when Vendor A introduces full symmetric multiprocessing—which has not yet found its way into this open operating system? And what if Vendor B develops a more sophisticated way of handling online transaction processing? These vendors have two choices: Either they add extensions to the open operating system, which will make each of their versions incompatible at certain levels with other implementations, or they wait until their advances can be incorporated into the open standard via the open process.

Users would be unhappy if all vendors waited until new technologies filtered through the open process. It would mean that vendors not committed to the open process could produce state-of-the-art software that users need. Ironically, this would mean that proprietary would be the place to go for innovation.

Yes, it is important for users to pressure vendors and standards-oriented organizations to provide as many base-level standards as they need to make porting and interoperability easier. But the last thing the open systems movement needs is another operating system like DOS. Yes, there was, by and large, a single operating system, and it was easy to provide shrink-wrapped, off-the-shelf applications. But when a new generation of applications was needed, the industry was faced with the prospect of moving to a brand new, complex operating system. Those clamoring for one standard operating system must think carefully—they might get just what they ask for. ◐

PATRICIA SEYBOLD'S
UNIX IN THE OFFICE

Editor-in-Chief
Judith S. Hurwitz
MCI: 3538836@mci.com
Internet: hurwitz@dcm1tp.das.net

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

Managing Editor
DOUGLAS A. FREEMAN

News Editor
DAVID S. MARSHAK

Art Director
LAURINDA PHAKOS

Publisher
PATRICIA B. SEYBOLD

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, Suite 612, Boston, Massachusetts 02109
Telephone: (617) 742-5200 FAX: (617) 742-1028 MCI: psocg / 312 2583
Internet: psocg@dcm1sg.das.net TELEX: 6503122583

Patricia Seybold's UNIX in the Office (ISSN 0887-3054) is published monthly for $495 (US), $507 (Canada), and $519 (Foreign) per year by Patricia Seybold's Office Computing Group, 148 State Street, Suite 612, Boston, MA 02109. Second-class postage permit at Boston, MA and additional mailing offices. POSTMASTER: Send address changes to *Patricia Seybold's UNIX in the Office*, 148 State Street, Suite 612, Boston, MA 02109.

• PROGRESS •

(Continued from page 1.) It is impressive that the company has managed to do all of this while maintaining an integrated development environment and keeping up its quality standard.

Progress is still clearly focused on improving the applications development process. While the current tools are not yet what we want for the end user, the company plans to enhance this area shortly. In addition to a new set of challenges (such as supporting graphical user interfaces and providing extensible data types and functions), the major issue for Progress is one of name recognition and market visibility. In terms of technology, Progress is catching up with, and in some cases surpassing, its larger rivals. Now it needs to become better known.

Company Background

Progress Software was founded in 1981 to bring mainframe DBMS functionality to the mini and workstation environments. As a result, Progress incorporated from the very beginning important capabilities such as automatic crash recovery (roll-back) procedures, automatic concurrency control, and a transaction orientation. The Progress RDBMS and application development language was introduced in 1984.

The company is still privately held, with ownership split 50-50 between venture capital participants and employees. Revenues for the 1990 fiscal year are projected to be $40 million, up 60 percent from $25.4 million in 1989. Income figures are not available, but the company states that it has been profitable since the introduction of Progress. Progress currently has 265 employees.

Product Overview

PRODUCT STRATEGY. The company's core product is the Progress fourth-generation application development language (4GL) and RDBMS software. The company also offers layered products on top of Progress, such as the Fast Track applications generator and gateways to non-Progress data.

Progress has always emphasized its 4GL and development toolset with the overriding objective of making the developer/programmer more productive. And the company projects a bright future for the applications development business. Joe Alsop, president, sees "industry trends coming our way. The '80s were focused on relational DBMSs and companies like Oracle. In the '90s, application development tools will be the primary requirement."

The company is also keeping its RDBMS up to snuff. The database engine has always provided a strong underpinning for the 4GL, and Progress has enhanced it significantly over the past two years. In the near term, we do not see Progress packaging its 4GL as a completely separate front-end tool for other DBMSs. The company recognizes that tools designed for one DBMS will never work as efficiently with another. Heterogeneous combinations usually end up paying a performance penalty, however small. The answer, instead, is to provide support for a heterogeneous computing environment (see Illustration 1).

PROGRESS 4GL/RDBMS. In our last review, we looked at Version 4 of Progress. Version 5, introduced in mid-1989, added a multiserver architecture and support for SQL. Version 6, announced in July, provides significant enhancements focused on access to multiple, heterogeneous databases.

Multiserver Architecture. With the advent of client/server architectures, all of the major DBMS vendors have moved toward supporting multiple database servers on a single platform.

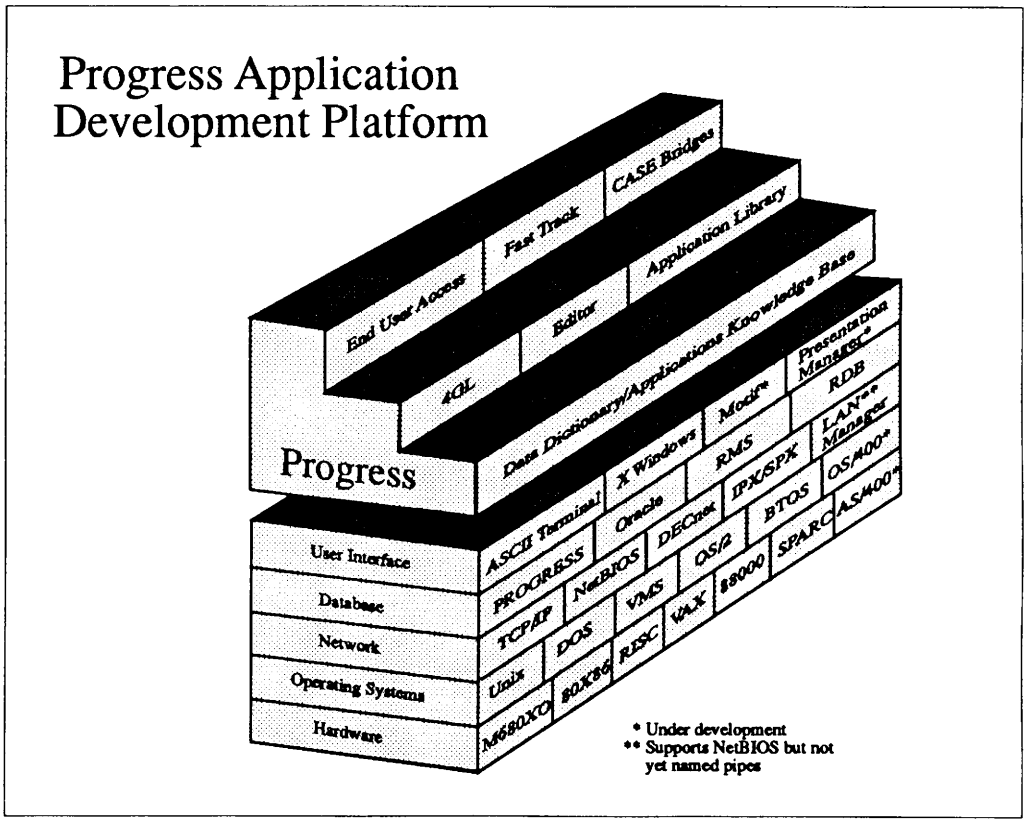

Illustration 1. Progress is working toward a complete application development system that offers independence at all of the five major levels of the computing environment.

This improves performance, increases the number of users that the DBMS can support, and allows the DBMS to exploit symmetric multiprocessing.

On Unix, VMS, and OS/2 systems running shared memory versions of Progress, Progress uses a multiserver architecture for remote client access to the database (see Illustration 2). Local (terminal) users can access the database through servers or directly through shared memory. In the latter case, they do not require a separate server process and are called "self-serving" clients. This improves performance by eliminating the need for any interprocess communication between the client and server. The decision to use servers for local clients is a tuning issue, trading off performance against optimum use of fixed resources such as shared memory and semaphores.

In the case of remote clients, a network service, called a *broker process*, manages the resources shared by remote users and is responsible for launching servers on behalf of these clients. The broker is transparent to the client. If remote clients are accessing the system using multiple network protocols, there is a broker for each protocol (e.g., a TCP/IP broker and a DECnet broker). Each server also handles one protocol and has access to one database at a time (although multiple servers can access the same database). Within these constraints, multiple clients can be assigned to one server. The broker will connect a new client to an existing server or spawn a new server depending on the situation. This is a major difference between Progress and Oracle, for example, which sets up a separate server process for every client.

The Progress multiserver architecture can be tuned by defining the maximum number of Progress servers and the minimum and maximum number of users per server. Progress is multithreaded by virtue of its use of shared memory and its multiserver architecture. Each individual server is still single-threaded.

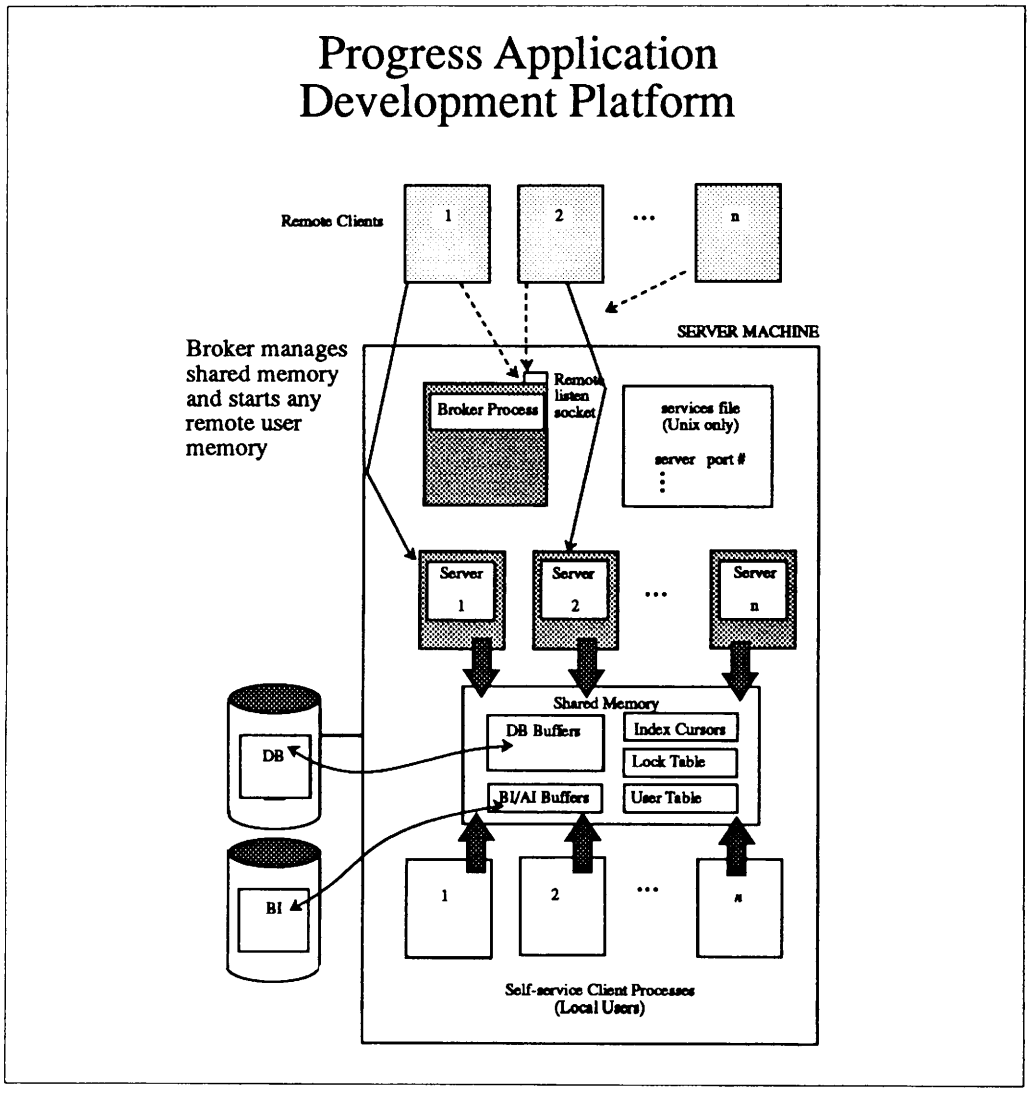

Illustration 2.

SQL. Progress fully supports the ANSI Level 1 SQL standard plus a majority of Level 2. Progress has structured its 4GL to accept SQL statements as well as Progress commands. A developer can use one or the other, or mix and match both types of commands in a procedure. You can also include Progress-type clauses in SQL statements to dress up the format and display of query results. The company has extended SQL to include capabilities such as defining array fields.

Distributed Database. A Progress user/application can now dynamically connect to multiple databases simultaneously (up to 240). These databases can be either local or remote. (Previously, Progress was limited to accessing a single database.) In addition, each Progress client or front-end process can support multiple network protocols concurrently.

Version 6 supports both distributed query processing and distributed transaction processing. The latter requires a two-phase commit (2PC) architecture to provide data integrity during multidatabase update operations. The

Progress 2PC is automatic.

Location independence is achieved by including both the logical and physical database names in the data dictionary at the client location. (Progress does not rely on a central data dictionary/star topology to support distributed database.) With location transparency, developers and end users don't have to know where data resides; they can access it transparently as if it were sitting on the local system. Another benefit is the ability to take advantage of distributed databases without rewriting existing applications.

Progress views the demand for distributed database more as a tool to integrate databases than as a way to take a centralized database and split it up. An organization can optimize DBMS tools and/or engines for each database application or department and then use distributed capabilities to integrate the data where appropriate.

Heterogeneous Databases. Progress now offers gateways to access non-Progress databases—Oracle and Digital's RMS, with support for Digital's Rdb coming later this fall (see Illustration 3). These gateways provide two primary benefits. By offering complete read/write access to data, they enhance the developer's ability to port applications from one database to another. Some of the caveats here are obvious, since Progress supports functionality that Oracle and RMS do not. A classic example is Oracle's lack of support for "find previous". The documentation for the gateways is very clear on these differences in function.

The gateways also enhance the power and flexibility of Progress's distributed database. We have long maintained that this will be a major part of the demand for distributed database, and that the successful vendors will be those recognizing the need to connect to competitive DBMS products.

Progress can perform distributed queries (joins) across all three types of data—Progress, Oracle, and RMS. Heterogeneous distributed updates are much more difficult to implement because neither Oracle nor RMS understand the concept of a two-phase commit. Progress allows Oracle and RMS data to participate in multidatabase update transactions to the extent possible, and clearly documents the risks involved in doing so. Since it is currently impossible to implement a complete two-phase commit across these heterogeneous data managers, the company's objective is to at least dramatically reduce the window of potential failure. If something does fail during this very short window, data can be lost, since Progress cannot extend its crash recovery procedures to Oracle or RMS data.

To access a foreign database, you first create a Progress database as the "schema-holder" for the non-Progress database. This describes the schema (tables and fields) of the foreign database. The DBA can set up the schema-holder or can invoke a utility that examines the foreign schema and configures the Progress version. Progress also provides integration with Digital's Common Data Dictionary to obtain RMS/Rdb schema information. The DBA can then enhance the schema using Progress data dictionary functions, such as adding table and field validation criteria, help messages, etc.

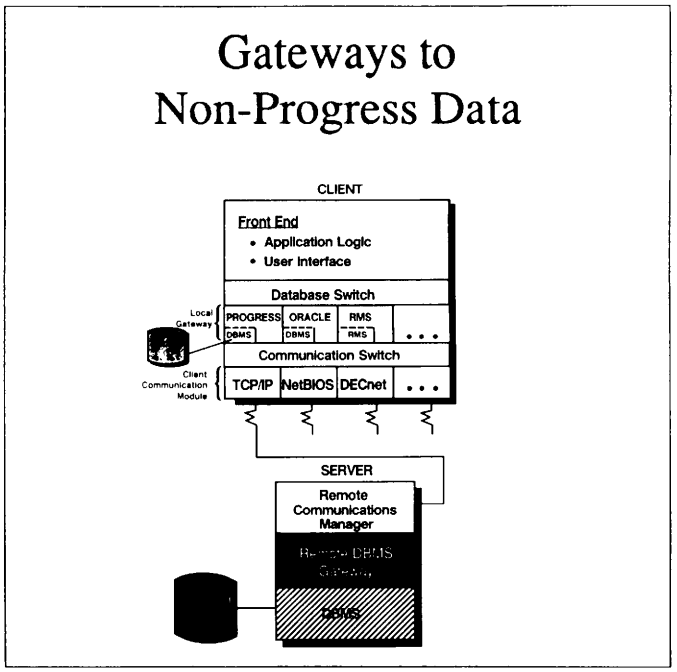

Illustration 3. Progress offers read/write gateways to Oracle and Digital's RMS, with a gateway to Digital's Rdb coming later this fall. Combined with distributed database capabilities, Progress can now begin to provide an applications development solution for heterogeneous data environments.

The Oracle gateway is a joint marketing and technical cooperation effort. Our guess is that this was prompted by Oracle's need, particularly in the Unix environment, for a strong 4GL in order to compete with products like Informix.

New Data Dictionary. The Data Dictionary function has been rewritten with a new interface (strip and pull-down menus); new options to support multiple, alternative, and distributed databases; and many other additions.

SQL Additions. Embedded SQL is now available for C, Cobol, and Pascal in the Progress Host Language Interface (HLI) products. The HLI for C also supports dynamic embedded SQL, letting C programmers create dynamic SQL statements at runtime.

The addition of a dynamic SQL interface will allow developers of other tools (e.g., spreadsheets, CASE, word processing) to more easily access Progress databases as well.

X Window Support. Progress applications can now run under popular graphical user interfaces (GUIs) such as those built atop X Window (OSF/Motif, Open Look, Open Desktop, DECwindows), and OS/2 Presentation Manager (PM). However, support for GUIs is still very limited. While Progress now runs in a window, it still looks like a character interface. Full GUI support will come with Version 7 (see "Futures" below).

Progress Pricing

Unix and VMS multiuser, DOS and OS/2 single user (price depends on platform):

Progress Application Development System (Progress 4GL/RDBMS, Fast Track)	$1,500 - 190,000
Progress 4GL/RDBMS	$1,050 - 154,000
Progress Query/Report	$650 - 93,000
Progress Run-Time	$200 - 38,500
Progress Fast Track (included in Application Development System)	$650 - 48,000
Heterogeneous database gateways: Oracle or RMS gateway	$275 - 38,500
Host Language Interface (HLI) for C (available on all platforms), Cobol or Pascal (available on selected platforms)	$375 - 47,500
Developer's Toolkit	$950 - 77,000
Test Drive (demo version of Progress)	$95

DOS and OS/2 server and client software:

DOS server	$325
OS/2 server	$1,250 (286/386) $1,700 (486)

Prices for DOS or OS/2 client software depend on number of nodes and the level of product desired (Run-time through full 4GL/RDBMS).

Note: All versions of the Progress database products (except Fast Track and the gateways, which are layered products requiring the 4GL/RDBMS) include support for heterogeneous networking—TCP/IP, DECnet, and PC LANs (NetBIOS, Novell SPX/IPX for portable NetWare, OpeNET). TCP/IP is not yet available on OS/2.

The Oracle and RMS gateways are currently available on several Unix platforms and VAX/VMS. They are not yet available on OS/2.

Illustration 4.

PACKAGING. Progress Software maintains a relative simplicity of packaging in refreshing contrast to many of its competitors. The RDBMS and 4GL are packaged together as a single product. The only subsets of Progress available are run-time versions aimed at helping the professional developer deploy applications (see Illustration 4). For example, with Progress Query/Report, the end user can develop new applications (i.e., new queries and reports) but cannot modify the database structure or the data. This offers a nice middle ground for VARs, enabling them to give customers not only a custom-developed application, but also ad hoc development flexibility at a lower cost (and risk) than providing a complete version of Progress. All versions of Progress include heterogeneous networking support.

Another set of tools to assist the developer in packaging and distributing applications is the Developer's Toolkit. The Toolkit now generates encrypted source code (unreadable and uncompiled), and every run-time version of Progress includes an encrypted source code compiler (with no decryption algorithm). Thus, the developer buys the Toolkit only for the platform on which an application is developed, not for every platform on which the application will be deployed. This is a tremendous plus for the developer in terms of reduced cost and easier deployment.

PLATFORMS. Progress is written in C and was developed in the 68000-based Unix environment. In addition to a wide variety of Unix platforms, it also runs on DOS, OS/2, Digital's VMS, Unisys BTOS/CTOS, and LANs (see Illustration 5).

Early availability on the PC AT (1985) provided a significant advantage for Progress developers. They could create a Progress application in a single-user DOS environment and then easily port it to other platforms, including multiuser systems. Unlike some of the competition, Progress on DOS runs within the standard 640K of memory, another strong selling point.

MARKETING STRATEGY. Progress is marketed primarily through value-added resellers (VARs), and the company now has 1,200 VARs worldwide. The company also sells directly to corporate MIS professionals and government agencies. Currently, 60 percent of the company's revenues are from VARs and 40 percent from end users.

Progress is well aware of the need to increase its name recognition vis a vis much larger and better-known competitors like Oracle and Informix. Selling primarily to VARs can often mean that the VAR and its application have a much higher profile than does Progress as the underlying platform. VARs will also benefit from better visibility for Progress since they often find themselves in direct competition with Oracle et al. for business.

In a recent reorganization, Progress established a separate marketing group and plans to gear up its efforts. The overall strategy is to move into a broader market, increasing penetration in the corporate MIS arena while maintaining a strong VAR program. The company feels that it now has the critical mass necessary to focus on both channels, and will increase both the direct sales force and required support groups. Areas the company intends to advertise more are its technology for delivering deployed applications (an issue for large users as well as VARs), its database architecture and support for heterogeneous distributed network computing, its strong positioning in both Unix and VMS, and the breadth of VAR applications available. Future

access to DB2 and availability on the AS/400 will also appeal to the Fortune 1000.

Progress provides consulting services, and expects this aspect of the business to grow. One arena Progress will *not* enter is the applications business. The company has no desire to follow in Oracle's footsteps and end up competing with its own VARs.

Installed Base. The company states that over 40,000 copies of Progress are in use, with over 220,000 users.

The Competition. Progress started on the Unix platform and continues to have its greatest strength here. The company sees positive developments for the Unix market as acceptance in the United States grows and as the platforms improve (e.g., IBM's RS6000). Progress competes most frequently with Informix in head-to-head comparisons of features and function. Oracle is also a strong competitor, since it is so large and well known.

Digital's VAX/VMS environment is newer for Progress, and the company has focused its marketing on VARs rather than trying to get VMS shops to switch to Progress. However, it is starting to see growing interest from user companies. Progress also feels that Version 6 positions it well for the entire Digital environment: The gateways to Oracle, RMS, and Rdb cover a majority of installed VMS databases, Progress runs on all of Digital's platforms (VMS, Ultrix, and DOS), and strong networking support fits well with Digital's emphasis in this area.

DOS is not currently a major database platform for Progress, and the company sells more to organizations looking to deploy applications developed on the PC into multiuser environments. However, emphasis on DOS as a *client* platform is growing, and Progress is packaging its DOS client/server software to appeal to this market. The company also offers both client and server software for OS/2 and will sharpen its focus here as demand dictates. Increased market visibility will become particularly important if Progress intends to compete more directly on the DOS and OS/2 platforms.

The CTOS/BTOS environment is still very pervasive in certain areas, particularly government agencies such as the Coast Guard. Progress is probably the leading applications development platform here.

Using Progress

We evaluated Progress Version 6.2A on a Sun 4/60 under OpenWindows. Progress is virtually identical in all of its operating environments. The accompanying chart provides a summary of Progress's features and functions (see page 12).

The Unix version requires a minimum of 2MB of memory for the server/operating system/shared memory, plus a maximum of .5MB for each user. (In a run-time environment, there can be at least three users per MB of shared memory.)

Progress uses a variable length format to store the information in each field, optimizing the use of storage space. Maximum record size increased from 2K to 32K with Version 6.

In this section, we will describe what it is like to work with the standard Progress 4GL/DBMS product. Then we will cover the Fast Track application development system and provide a sneak preview of a soon-to-be-introduced end-user product.

USER INTERFACE. In the Progress environment, the procedure editor is "home base," so to speak. You start in the procedure editor, and use it to access everything in Progress (including the data dictionary). Once a database is defined using the data dictionary, all other Progress functions and operations are done by entering statements in the procedure editor and running the procedure. This includes adding, updating, and deleting records, querying the database, and generating reports, as well as developing complex applications.

The Procedure Editor. The procedure editor is fully integrated with the data dictionary and the language compiler. It verifies the existence of tables and fields and checks your syntax. One of the best features of the procedure editor is its level of interaction with the user. If you make an error, it positions the cursor over the error and provides an error message. At this point, you can correct the error and immediately rerun the procedure. You can

Progress Operating Platforms

- **Unix**

  Many, including:

Apple A/UX	Nixdorf
AT&T	Olivetti
Bull	Philips
Data General Aviion	Prime
Digital Ultrix	Pyramid
Hewlett-Packard HP-UX	Sequent Symmetry
IBM AIX	Sun
ICL	Unisys
MIPS	Unix 286/386/486
Motorola	Xenix 286/386/486
NCR	

- **Proprietary** — Digital VMS, Unisys BTOS/CTOS
- **DOS** — Single-user, server and client software
- **OS/2** — Single-user, server and client software
- **LANs** — TCP/IP, DECnet, NetBIOS, Novell SPX/IPX, OpeNET

Illustration 5.

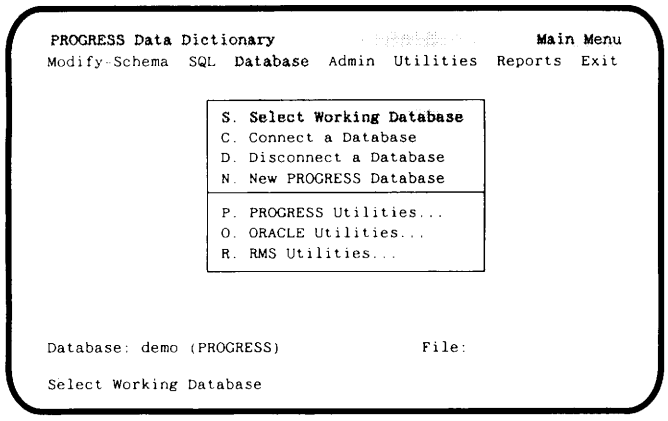

Illustration 6. This is the main menu for the Progress data dictionary showing a pull-down menu.

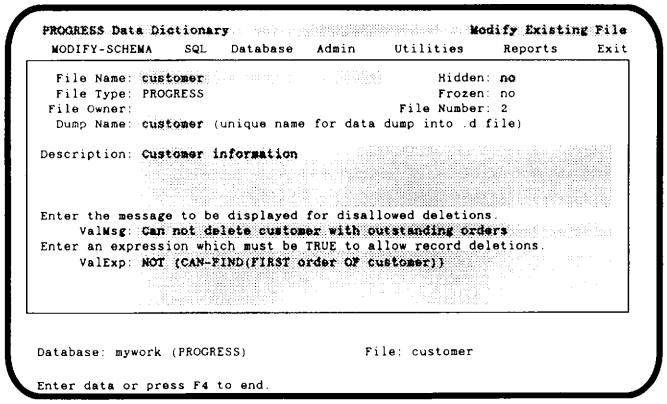

Illustration 7. On this screen, you define a table.

move, delete, and copy text within a single procedure or between procedures. There is also a global search and replace function.

Progress is flexible in creating procedures. Procedures are stored as ASCII files and can be written using other editors. Statements within procedures can take up more than one line. The only requirement is that the entire statement end with a period (as if you were writing sentences). As do most procedural languages, Progress recommends that you indent nested statements and loops within procedures. This formatting convention makes procedures easier to read and understand. A nice touch is the fact that when you indent a line in a procedure, Progress continues to indent subsequent lines until you manually move to a different margin.

Help. From the end-user perspective, we were disappointed that the help function remains unchanged since our last review (the only addition is access to a new procedure library). Help is not context sensitive, it is not as helpful as we would like, and there is no online help for the data dictionary function.

Help is a fixed menu of items from which you choose what you want to know. It is designed primarily for assisting in the procedure debugging process.

Documentation. Version 6 documentation has been redesigned. It consists of several manuals that come packaged with a desktop "bookcase" or frame in which to neatly store them.

The manuals are clear and well-written, including the Programming Handbook, even from the end-user perspective. They are full of examples and make good use of graphics. The writing style is straightforward and does an excellent job of explaining terms and why Progress does something the way it does.

Since Progress looks the same on all systems, the documentation includes instructions for all environments: Unix, DOS, VAX/VMS, and CTOS/BTOS, providing helpful hints about each operating system.There is just one version of the documentation no matter what system you have. The documentation also provides helpful hints about each operating system.

CREATING A DATABASE. Version 6 allows multiple databases to be connected concurrently, so you can create a database from within the Progress data dictionary. Previously, you had to do this at the Unix prompt before entering Progress. Then, if you wanted to create or use another database, you had to actually exit Progress and restart it with a different database name. We found this counter-intuitive, since the database became the "umbrella" under which the user worked rather than the Progress environment itself. This is reversed with Version 6, and it makes much more sense, not to mention the additional flexibility it provides.

Data Dictionary. The Progress data dictionary, itself a Progress application written in the 4GL, is menu driven (see Illustration 6). It stores all information about the distributed database environment, including the structure, type, location, access method, and access restrictions for each database. You create, modify, and delete databases, tables, fields, and indexes by selecting menu items and completing screen forms. Progress has done a nice job of structuring these menus and screens. We found the process relatively intuitive and easy to work with. We especially like the fact that Progress has made defining tables and fields comprehensive and yet not overwhelming.

One criticism we have of the Progress user interface is that there are minor differences in the way menus have been implemented in the data dictionary, help, and Fast Track modules. This isn't a big problem, but cleaning these up will enhance the product's consistency.

Tables. For each table, you can enter a validation expression, which is a logical expression that must be true before Progress will delete a record from the file(see Illustration 7). An example of a validation test would be disallowing the deletion of a customer master record if there are still outstanding orders for the customer.

More complex validation criteria can be created in an "include" file, which is then referenced as the validation expression. Thus Progress allows you to define a level of referential

integrity in the data dictionary rather than depending on the competence and thoroughness of the developer to preserve referential integrity. This is an important factor in increasing developer productivity and preserving database integrity. Progress cannot, however, cascade referential integrity checks across multiple tables automatically. This will come with the implementation of triggers in Version 7.

Fields. The only required information for a field is its name and datatype. You do not have to indicate a field length because Progress stores the data in each field as a variable length. Progress supports five datatypes: character, integer, decimal, logical, and date. Progress fills in default values for the remaining field characteristics and allows you to change them if you want. Null values and arrays of any datatype are supported. (See Illustration 8 for other field attributes.) We would have liked more information and examples on creating field validation expressions, particularly those where you validate the existence of data in another table.

Modifying the Structure. Progress allows you to change the table specifications (such as validation test and message) as well as the table name, and to delete tables.

You can add, delete, and modify fields in a table. None of these modifications requires the user to unload the data first and load it back in after the database is modified. The only things you cannot change are the datatype or array extent of a field. Doing either requires creating a new field, copying the data into the new field, and deleting the old field.

Only one person can change the data dictionary at one time, and, on a multiuser system, you can't change tables that are in use by either a user or a running procedure.

DESIGNING FORMS. Progress uses "frames," or windows, to display data, and automatically provides a default frame format if one isn't specified. To do this, Progress follows a comprehensive and well-documented set of default options. It is also possible to define custom frames within a procedure. You can specify overall frame characteristics as well as those that affect individual fields in the display (with the ability to override or augment data dictionary definitions). Screen forms can contain multiple frames (windows) and frames can be hidden or overlaid on others (see Illustration 9). Progress also supports color and other video attributes.

Using the "form" statement is helpful if you want to use the frame repeatedly in one or more procedures. The form statement defines the frame and names it. The procedure can then invoke the frame definition whenever appropriate. This is akin to the standard forms-design process in other DBMSs, where you design a specific form, store it, and call it up when needed. Forms can also be designed using the Fast Track screen painter.

ENTERING DATA. To add, update, and delete records in a database, you use the appropriate Progress commands (insert, update, and delete) in the procedure editor.

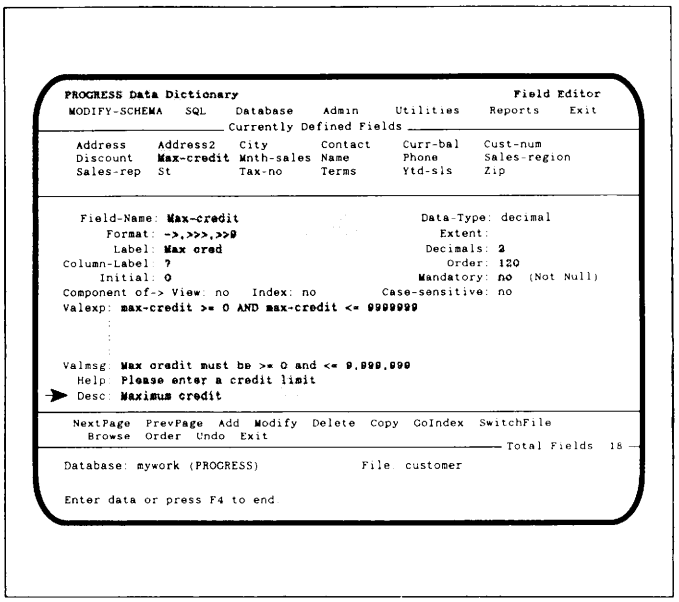

Illustration 8. Here is the field editor showing the max-credit field definition in the customer table.

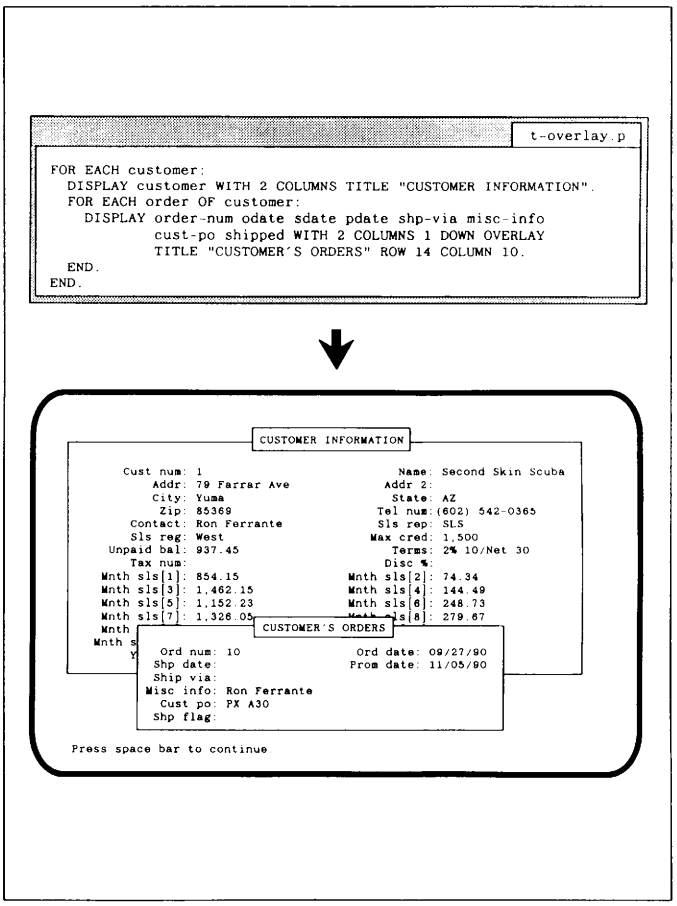

Illustration 9. A sample procedure to define a screen form. This procedure defines two frames, one of which overlays the other. The customer information is displayed first. When the user presses the space bar, the order window pops up, displaying each order for the customer one at a time.

When entering data in a new record, Progress does not always validate a field entry as you leave the field. It may wait until you are done with the record and then go back to invalid field entries. We would rather get immediate feedback that the entry doesn't meet the validation test.

Using the Fast Track query-by-forms function, you can further automate the data entry and editing processes by defining a form and an associated strip menu for add, update, and delete.

INDEXING. It is easy to create indexes with the index editor in the data dictionary. Options include specifying whether an index is primary and/or unique, and the fields on which to build the index.

Progress uses B-tree indexing and allows indexing on up to 16 fields. You can change the name of an index and delete it. You cannot change the components of an index once it is created; you have to create a new index.

Once you define an index, Progress uses it automatically, selecting the most efficient path to the data.

While Progress is not really an end-user development tool for the average user, there is an elegance and ease surrounding much of the Progress functionality that is impressive.

QUERIES. Progress queries are generated from procedures that generally use the "for each" and "display" statements. For example,

```
for each customer where curr-bal >= 1400:
  display name phone curr-bal sales-rep.
end.
```

We found it very easy in Progress to relate tables and display related data on the screen. Displaying information from three files (e.g., line item information for each order for each customer) in three separate windows took only a couple of minutes to generate and refine. And no forms design was necessary, although the display could easily have been spruced up. It is mind-boggling to imagine doing the same thing with SQL and some other forms-design products we have used.

We could not find a way to display aggregates (e.g., totals) without listing all the fields we wanted to see. We also could not go backward through records retrieved by a Progress query (a page-up function to go back to the previous record). The only way to see previous records is to rerun the query. Query-by-forms in Fast Track resolves both of these issues.

REPORTS. A report in Progress is simply another query with control breaks, aggregates, and fancier formatting. As the report gets more complex and customized, so does the procedure required. If you want to override the Progress default layouts, you are into specifying column and row positions. Fast Track provides an easier-to-use WYSIWYG report writer.

APPLICATIONS DEVELOPMENT. The tutorial documentation on writing procedures is excellent, explaining in detail what is actually happening in the database as each statement in a procedure executes. Progress automatically compiles a procedure before running it for the first time, and procedures can be nested. The tutorial describes this process, explaining how to set up subprocedures and include files, and how to pass data and arguments among procedures to build flexibility into an application. The documentation also covers putting the "finishing touches" on an application, such as providing context-sensitive help and preparing the application for use.

While Progress is not really an end-user development tool for the average user, there is an elegance and ease surrounding much of the Progress functionality that is impressive. Much of this has to do with the close integration of the procedure editor and the data dictionary, the interactive nature of the procedure editor, and the fact that Progress essentially provides a single environment to work in. Unfortunately, some basic development steps, such as creating an application menu, require writing what looks like a relatively complex set of procedures. Yet other functions that appear to be complex, such as relating three files and displaying information from each in multiple windows on a single screen, are incredibly easy. You don't even need to design a custom form to get meaningful results in Progress. The default formats are often quite serviceable.

Fast Track provides a more automated tool for generating applications, dramatically improving the accessibility of Progress's power and functionality for the end user. The coming end-user tools go even further in providing easy access to data.

DATA INTEGRITY. One of the real strengths of Progress is its data integrity facilities. You can include data validation and referential integrity in the data dictionary; Progress provides automatic crash recovery and supports roll forward recovery. The company states that crash recovery has always been an important feature of Progress because it uses variable record lengths.

SECURITY. In the Progress data dictionary, you define user IDs and passwords to ensure that only authorized users can access the database. You can also specify separate read, write, create, and delete permissions at the table and field level. Restricting access to specific records in a table can only be accomplished with a procedure.

Within an application, the developer can create an "activities permission" file, a table listing each procedure and the names of users authorized to run the procedure. Security can also be built in at the operating system level in Unix (and VMS) for databases and object code.

Fast Track

Fast Track is a comprehensive, menu-driven, applications builder layered on top of Progress. Introduced in late 1988, it is designed as a productivity tool, speeding the development of applications for both developers and experienced end users. Fast Track, itself an application written in the 4GL, includes a menu editor, screen painter (for creating forms), report writer, and query-by-forms (QBF) generator (see Illustration 10). Once a database is created (using the Progress data dictionary), the developer can design menus, screen forms, reports, and QBF access to the data without having to use the 4GL. The result can be a complete application, or it can provide modules of a more complex application.

Fast Track automatically generates standardized and optimized Progress 4GL code. The application can be modified within Fast Track, or the code can be edited directly using the Progress 4GL editor. Fast Track is self-documenting, including comments in the code it generates.

One of the most attractive aspects of Fast Track is its flexibility. You can design an application from the top down, creating a menu structure and then defining the underlying processes (reports, data entry forms, etc.) invoked when a menu choice is made. Or you can first design the individual pieces and then integrate these into the menu structure. Fast Track has been designed to easily incorporate procedures written in the Progress 4GL. Thus, an application initially generated with Fast Track and customized further with 4GL procedures can be modified within Fast Track without "clobbering" the embedded procedures.

Other appealing features are the WYSIWYG approach to developing menus, forms, and reports; consistency in the use of function keys and commands; and the ease with which the developer can view objects on the screen exactly as they will be displayed in the application.

While Fast Track cannot completely handle all aspects of a complex application, it can greatly improve productivity in designing menus, forms, and reports while allowing the developer to incorporate procedures written in the 4GL. Fast Track is also a good tool for prototyping applications.

FUTURES. Future plans include extending Fast Track to handle as much of an application as possible, reducing the need to write 4GL code. Progress is also aware that its developers build tools (e.g., menu generators) as well as applications. In an effort to tap this talent and to allow the developer to continue to use familiar tools, the company intends to make Fast Track both modular and tailorable. It will provide a framework and library facilities so the developer can mix and match Progress-developed modules with other development tools of choice.

It is not clear whether Fast Track will ever become a full-fledged CASE development tool, although the company is moving somewhat in that direction. Its current strategy is to encourage third-party development of bridges to existing CASE offerings rather than direct competition. Examples will be connections to both KnowledgeWare and Index CASE products.

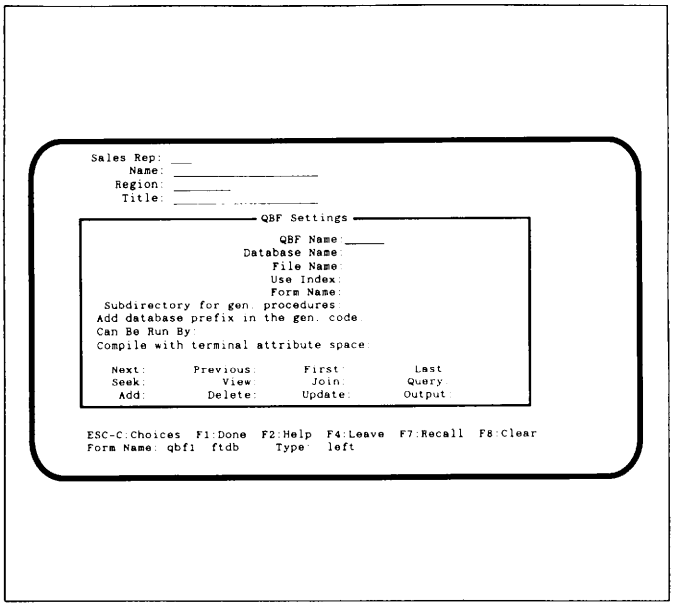

Illustration 10. In this pop-up window, the developer defines parameters for a Fast Track query-by-forms process. The items listed at the bottom of the window are the options that can be included in the ring menu displayed at the bottom of the form in the actual application.

End-User Tools

Progress is getting ready to introduce its first end-user data access product later this fall. We had the opportunity to see a pre-beta-test version and liked several aspects of the product (as yet unnamed). Our initial impression is that Progress has done its homework in assessing what the end user wants, at least in terms of data access. The next step is to provide end-user development tools. The company acknowledges the need for these in the future, but feels that the user still really wants a professional developer to do application development unless the application is very simple.

The company has worked hard to make the end-user product easy to use, reasonably powerful, and well-integrated with other Progress modules. Designed as a set of data access tools, it is complementary to existing development tools rather than an application development tool itself. It supports both the 4GL and Fast Track, generating 4GL code that can be modified by both.

The interface is menu driven and consistent with Progress and Fast Track in the use of function keys. Modules include query, reports, labels, data export, system administration, plus a "user" menu option for accessing custom tools and utilities.

A primary objective is to maintain context for the user, keeping the user from getting lost in either the tool or the database. The product builds its own default interface based on the existing database schema and user permissions. It carries as much context as possible as the user moves between modules (e.g., from query to reports) or to the "user" menu. The help function will also be context sensitive.

QUERIES. The query facility is forms based and offers either a QBF or QBE (query by example) approach. The user can choose from a list of available query forms, and each form has an associated list of tables from which it displays data. The forms themselves must be created in the 4GL or Fast Track. Table relationships (joins) are defined in Fast Track, by the system administrator, or by the end user.

The user has options to enter a query by example (which only supports the "and" operator, as is typical) or more complex query criteria by creating a "where" clause (you simply select "where" from the query menu and fill in the forms on the screen via point and pick). The user can also edit the query statement directly. With permission, the user can access menu options for add, update, and delete.

REPORTS. The report module is very easy and intuitive. You simply pick fields from a list and number them to indicate their order on the report. The fields are displayed in a WYSIWYG layout window as they will appear in the report. You can rearrange the fields by renumbering them (the following fields are automatically renumbered); create totals, subtotals, running totals, percents, and counters; use an expression-builder to create calculated fields; include fields from up to five tables; order on up to five fields and suppress duplicate values; define formats, labels, column spacing, and header justification.

Progress plans to provide direct access to Fast Track so that reports designed in the end-user product can be easily moved to Fast Track and enhanced.

LABELS. A nifty label generator will try to automatically select the appropriate label fields (e.g., name, address, city, etc.) if requested. The administrator can provide a sample list of field names to help this process. The user can modify the selected fields and define several parameters: omit blank lines, number of copies, number of labels across, total height, etc.

Progress Features Chart

Architecture	
Client/server	Yes
Multiserver	Yes
Maximum number of severs/systems	50
Maximum number of users/server	256
Maximum number of users/system	256
Multithreaded server	No
Support for symmetric multiprocessors	Yes
Open architecture (APIs available)	Yes (host language interace and dynamic SQL)
Underlying file structure	Unix
Database parameters	
Database size	200GB
Databases/server	Maximum of 50
Tables/database	1,023
Records/database	Constrained by maximum size of database
Record size	32K
Fields/record	Constrained by record size
Indexes/database	1,023 (no limit per table)
Users/database	2,048
Databases connected to a client	240
User interface	Commands in procedure editor; menus in Data, Dictionary, Help and Fast Track
Menu bypass	No
Contextual help	No, unless custom help designed for an application
Tutorial	Yes
Ability to customize standard menus	Yes, in Help/Data Dictionary; no, in Fast Track
Support for color	Yes
Support for graphical user interface	Limited support for X Window; full GUI support planned for Version 7
Data types	
Character (fixed/variable length)	Variable length; constrained by record size of 32K
Integer	Yes
Float	Yes—decimal datatype (50 digits total; up to 10 decimal places)

Progress Features Chart

Logical	Yes
Currency	Handled through format specification
Date/time	Date—yes; time—no (Progress has limited tools to support time value in a field.)
Long text	Yes (variable length char)
Binary (fixed/variable length)	No; only through C subroutines
Image	No; only through C subroutines
Support for arrays	Yes; can define any data type as an array.
User-defined data types	No (planned for Version 7)
User-defined functions and operators	No (planned for Version 7)
B-tree indexing	
Maximum number of indexes	1,023/database
Maximum number of fields/index	16
Maximum size of index key	126 char
Order options	Ascend/descend
Unique index	Yes
Clustered index	No
Other file access methods (hash, etc.)	None
Screen forms	Forms are created with the 4GL language or with the screen painter in Fast Track
Default form generator	Yes
Customized	Yes
Multiple tables/form	Yes (no limit)
Multiple screens/form	Yes
Embedded processing (if-then-else, display aggregates)	Yes
Field attributes on forms	
Case conversion	Yes
Default value	Yes
Required field	Yes
Acceptable values	Yes
Verification (enter data twice)	Yes, in the 4GL; no, in Fast Track
Formatting of data	Yes
Calculated fields	Yes
Display/read only (no entry/update)	Yes
Hidden	Yes, in the 4GL; no, in Fast Track
Prompt (for data entry)	Yes

The Developer's Perspective

We also discussed Progress with five developers to get their assessment of the strengths and weaknesses of the product. All have extensive experience with Progress, having used it for two to five years. Three developers are using Progress for internal development, and two are VARs.

STRENGTHS. All of the developers are very pleased with Progress. Three major strengths were cited across the board, and everyone had additional positives to report.

Applications Development. Everyone stressed the ease and speed of developing and prototyping applications. The 4GL is also a complete language; none of the developers have had to resort to 3GL to complete an application. "Even when you think you're stuck, you can usually find a way around the problem." (The only exception was storing and manipulating text blobs—see "Other Weaknesses" below).

One internal developer now has 600 Progress applications in place and states that the product has greatly improved productivity. Other positive comments included the conciseness of the language and the ability to build your own development tools using Progress.

Another developer likes the flexibility Progress provides in designing applications. Examples are the ability to design real-time screens (updated every second) and custom interfaces, and strong screen-mapping capabilities.

Integrity. All five developers cited Progress's data integrity capabilities; none has had any problems in this important area. One developer has "never lost data in five years."

Database Architecture. Progress's support for both distributed and heterogeneous databases was mentioned by everyone as providing significant new opportunities in the kinds of applications

that can be developed. One developer stressed the ability to now compete in areas it couldn't previously, such as developing a centralized order processing system that places demands on multiple remote manufacturing sites, or a system in which remote sales offices place demands on a central manufacturing site. Multiple/distributed database access is critical to these types of applications.

One VAR plans to port Progress applications to Oracle in particular. This VAR sees access to Oracle as greatly enhancing its competitive position in vertical markets where Oracle is weak, such as complex manufacturing systems.

Another real benefit of distributed database is in providing both growth paths and performance enhancements. A developer can easily distribute a database among multiple machines if the application outgrows one platform. One developer also improved performance of an application by converting it (without having to modify the application) to a distributed database across three machines. Another developer has worked extensively with Progress's distributed database capabilities and states that it "works and works well, including the two-phase commit."

Portability. Three developers see Progress's high level of portability across a variety of platforms as a major benefit. Applications can be developed on one platform and ported to another simply by recompiling code. This facilitates deployment of applications and provides flexibility in choosing platforms.

Quality. Three developers describe Progress as a very quality-oriented company. Products are not delivered unless they are truly ready for prime time, an important issue for both internal developers and VARs.

Support. Developers complimented the company on its responsiveness to its users ("they listen to us") and on "dependable, knowledgeable" technical support.

Two developers also mentioned the clarity and completeness of the documentation as a strength.

Progress Features Chart

Customized help	Yes
Video display	Yes, in the 4GL; no, in Fast Track
Ability to change field attributes dynamically	Yes
Query-By-Forms	Part of Fast Track; all of these features are also included in the Progress 4GL.
Exact match	Yes
Relational operators	Yes
Ranges	Yes
List of values	Yes
Wildcards	Yes
Maximum/minimum values	Yes
Print query results	Yes
Pass results to report writer	No
Text search	No
SQL	Support for ANSI SQL Level 1 and much of Level 2; SQL and Progress 4GL statements can be mixed in a procedure.
Standard SQL statements	
Data definition language (DDL)	Yes
Data manipulation language (DML)	Yes
Extensions to SQL	
Commit/rollback transactions	Yes
Execute operating system commands	Yes
Load/unload data to/from ASCII file	Yes
Additional data definition statements	Yes
Control-of-flow logic	Yes
Support for embedded SQL	Yes (C, Cobol, Pascal)
Support for dynamic SQL	Yes (interactive, C, Cobol, Pascal)
Can create new table with query results	No
Stored queries	No
Case-insensitive (e.g., field names)	Yes
Can call C routines	Yes
How create SQL queries/statements	Procedure editor
Query optimizer	Yes
Syntax independent	Yes

Progress Features Chart

Report writer	Part of Fast Track; all of these features are also included in the Progress 4GL.
Nonprocedural	Yes
Default report generator	No
Interactive report generator using screen forms	Yes, in Fast Track
Interactive debugging	No
Input source	Result of a 4GL procedure or defined within the report
Multiple tables	Yes
Page formatting	Yes
Headers and footers	Yes
Data formatting	Yes
Sort data	Yes
Aggregate functions	Yes
Logical processing (if-then-else logic)	Yes
User variables	Yes
Prompt for input variables at run-time	Yes, with 4GL procedure
Application generator	Progress Fast Track
Ability to design application menus	Yes
Default menu generator	No
Custom help	Yes
Ability to create views	Yes
Integrity	
Transaction logging	Yes
Commit/rollback transaction	Yes
Roll forward	Yes
Referential integrity	Yes; single-level referential integrity defined in data dictionary as an expression
Field validation	Yes
Support for business rules	In application only
Stored procedures (ability to store and execute procedures on the server)	No (planned for Version 7)

Other Strengths. Other strengths included Progress's low cost, minimal memory requirement per user (1/3 to 1/2MB), and its strong VAR program.

WEAKNESSES. None of the developers had any major negatives with respect to Progress, and no single area of improvement was mentioned by all five.

Size of Company. In general, the developers are concerned about the lack of market visibility for Progress. One developer stated that, although the company has capabilities some competitors are still only promising (e.g., distributed database, client/server), it doesn't stress this enough. The consensus is that the company is too conservative in its advertising and needs to do more to compete with Oracle, Ingres, Sybase, and Informix. "The tendency not to overpromise can enhance credibility at the expense of visibility." One VAR described Progress as "second-tier in terms of recognition and sales, although the product itself warrants placement in the top tier" in terms of capabilities.

One developer raised a related issue—the need for a more global presence in Asia and the Pacific. Kanji support, for example, is essential for large global companies; Progress needs to incorporate this to develop a truly worldwide presence.

GUI. Support for graphical user interfaces is becoming important; the developers see this as a requirement for future success.

End-User Requirements. Lack of a good set of end-user tools for query and report writing was a negative for two developers. (These tools will be available shortly.) Another potential solution in this area is the connection of user-friendly tools to Progress via dynamic SQL.

Support. In spite of the overall satisfaction with support, a few problems and cautionary points were raised. One developer experienced some serious problems with Progress when upgrading to a new release of the operating system on one

platform. He described Progress as tending to have a "fly and fix" mentality here (you fly it, we'll fix it). The company needs to be on the leading edge in this area and to work more closely with major hardware vendors.

One developer wants better tools for monitoring database activities and tuning the system. Another was concerned that the heterogeneous database gateways may cause some support problems because they essentially create a multivendor environment. "It is unclear how much this will stretch Progress's support resources."

Another developer doesn't feel that Progress adequately documents "tricks and traps" necessary to more effectively use the language (e.g., definition of record-scoping, why it is relevant, and where it is useful). Progress also needs to make the examples in the documentation consistent in terms of coding (e.g., shared lock vs. no lock).

Default Locking. Two developers were critical of the fact that Progress uses shared locks as the default locking environment. This can create problems (and blow up lock tables) if the developer is not careful on multiuser systems. One mentioned the fact that the documentation doesn't adequately explain this situation. The addition of a startup parameter to change the default to "no lock" will help here.

Other Weaknesses. One developer wants to see enforcement of all of the validation rules in the database and support for primary/foreign keys (coming with Version 7). This becomes increasingly important as companies move toward distributing the applications development process as well as the database. Another developer wants support for large text and image blobs. The database should be able to store large chunks of text and allow editing within the database.

Progress Features Chart

Concurrency control—locking levels	
Database	No
Table	No
Record	Yes
Page	No
Other data isolation levels	None
Database security	
Login password	Yes
Multilevel access control	
User	Yes
Group	Yes
Application	Yes
Database-level access	Yes
Table-level access	Yes
Record-level access	Within application only
Field-level access	Yes
Access by time of day	No
Access by location (workstation)	No
Ability to define resource limits on user queries	No
Availability	
Online backup	Yes
Online database changes	Yes
Raw input/output	No
Database can span multiple physical devices (disks)	Yes

Futures

We have already described enhancements planned for layered products, such as the extension of Fast Track and the introduction of end-user tools. Other developments in key areas will be introduced in future versions of the 4GL/RDBMS.

EXTENSIBLE DATATYPES. Support for multimedia datatypes and peripherals is coming for Progress. The developer will be able to define completely new datatypes by writing C programs (drivers) to describe the necessary formats for I/O, calculation, sorting, storage, and export, as well as functions that can be performed on the datatype. Progress sees two primary markets for this capability: third parties who want to provide tools for the Progress application developer, and hardware manufacturers who want Progress to better support their platforms.

Another planned addition to the data dictionary is support for what Progress calls *domains*, or user-defined datatypes. Here, the user takes a standard Progress datatype and extends it to include parameters that currently must be defined for each instance of the datatype. For example, you could define a "currency" datatype as a decimal datatype with a format of $>>,>>9.99. If you then enter "currency" as the datatype for any new field, the new field will take on these attributes. User-defined datatypes simplify maintenance, support relationships among data, and promote consistency among applications.

Progress Features Chart

Support for CD-ROM, WORM drives	Yes
Network support	TCP/IP, DECnet, PC LANs (NETBIOS, OpeNET, Novell IPX); all are included in Progress 4GL/DBMS.
Import/export capability	
Import formats	DIF, SYLK, ASCII (delimited and fixed length), dBase (definitions and file contents)
Export formats	DIF, SYLK, ASCII, WordStar, MS Word, WordPerfect, BTOS OfisWriter
Distributed database capability	
Location transparency	Yes
Distributed query processing	Yes
Distributed query optimizer	No
Distributed transaction processing (two-phase commit)	Yes
Support for data replication	No
Access to heterogeneous databases	Yes—Oracle, Digital RMS and Rdb; full read/write capability
Maximum number of simultaneously connected databases	240
International language support	
Upper/lower case conversion	Yes
Sorting/collating sequences	Yes
Error messages	Yes
2-byte character set	No (planned for Version 7)

TRIGGERS. Adding triggers to Progress will greatly enhance its appeal to the developer. The company will predefine categories of events that can have triggers attached to them—database events (e.g., create, update, or delete record) and user interface events (e.g., enter or exit field). Triggers will be written in the 4GL and then stored and executed either in the application (on the client side) or on the server. This is different from triggers as implemented by Sybase, since Sybase triggers are always stored and executed on the server.

Triggers can be used for a wide variety of purposes, including cascading referential integrity, record-level security, and complex validation logic.

GATEWAYS. Progress plans to add other DBMS gateways to enhance the migration path it provides for the user. Under consideration are Informix/C-ISAM, Ingres, Sybase, and IBM's DB2. In the case of DB2, Progress is looking to work with a partner that has already invested in a DB2 connection.

Progress would also like to open the development of gateways to third parties. Providing an "open" gateway interface is more a matter of documentation and packaging than development. Extending gateways in this direction also means grappling with new support and quality control issues.

SQL. Version 7 will feature a complete implementation of ANSI Level 2 SQL (e.g., union, support for primary and foreign keys, etc.).

PERFORMANCE. Progress will continue to enhance performance—refining and publishing TP/1 and other benchmarks, and increasing the number of users and the size of the database supported.

WINDOWS. A major effort for the company is full support for industry-standard graphical user interfaces at two primary levels. Progress must take advantage of the appropriate window manager toolkits (Motif, Presentation Manager, et al.) to get its own real GUI. This includes the ability to launch multiple windows from within Progress and to have multiple frames active at the same time. The company must also provide tools that allow the developer to easily add a graphical interface to Progress applications.

Two major objectives for Progress in supporting GUIs are to provide *presentation independence* (so the application can adapt transparently to the native GUI regardless of the development or run-time platform), and support for character-based terminals. The ability to effectively run an application designed for a GUI on a character-based terminal without modification is very important. Developers will not want to use different tools to write applications for terminals and workstations, or to have to write an application twice to accommodate both.

Progress does not yet support a native Macintosh interface and sees support for GUIs in Version 7 as the logical point at which to do this.

OTHER ENHANCEMENTS. Other potential developments include the ability to perform full-text searches within fields; tools, such as a comprehensive debugger, a performance profiler, system tuning tools, and a preprocessor; support for Progress as

a Network Loadable Module (NLM) for Novell's Portable NetWare; and extensive enhancements to multilingual capabilities (such as Kanji support). We also expect to see Progress taking advantage of emerging new technologies such as object orientation.

Conclusion

Progress has a lot going for it—a strong RDBMS, comprehensive development and deployment tools, happy developers, and a broad range of available applications. (Hewlett-Packard's OpenMFG MRP II offering is a Progress VAR application.) The company is right on target with its emphasis on distributed network computing support.

The major negative for Progress is lack of market visibility for both the company and the product. And waiting too long or moving too slowly to fix this problem may mean missing a critical window of opportunity. The selection of a DBMS is becoming a strategic decision for many organizations, and the competition is fierce. The customer is more savvy and is looking for vendors who can back up marketing promises with real products that solve real business problems. The RDBMS is getting more complex, and subtleties in how or to what extent a particular feature or function is implemented can get lost in the marketing noise.

Progress has worked hard to position itself well technically, filling some important gaps with support for heterogeneous distributed databases and SQL. Now it needs to carefully plan and implement a marketing strategy that will enable it to effectively differentiate its offering.

Another concern is how fast the company can grow and still keep everything under control in terms of product quality and effective support. A third requirement for success is an on-going development process to ensure that the product continues to meet market demands. Here, Progress needs to focus on areas we have already discussed: enhancements such as triggers and extensible datatypes, performance improvements, more gateways, and support for windowing environments.

Progress Software again has an impressively full plate in front of it. The pressure this time is as much on the sales and marketing folks as it is on those developing advanced technology. The company is clearly aware of the challenges ahead and appears ready to tackle them with enthusiasm. This could make the RDBMS market even more interesting than it already is.

The title of next month's *Unix in the Office* is **"Sun's OpenWindows: The Workgroup Macintosh."**

For reprint information on articles appearing in this issue, please contact Anne Workman at (617) 742-5200.

NEWS ANALYSIS

PRODUCTS • TRENDS • ISSUES • ANALYSIS

•INSIDE•

•OSF•

DME: Managing to Reach DCE

Throughout the process that ultimately led to the Open Software Foundation's (OSF) Distributed Computing Environment (DCE) selections, one major issue continually surfaced: How do we manage all this stuff? While each of the technologies selected has its own management system, no common framework exists for managing them all.

This concern echoed what we have been hearing from users as their main misgiving about leaving the relative safety of single-vendor solutions to enter the world of open systems and heterogeneous, distributed computing. In every user survey we have taken or seen, ease of management and administration has headed the list of demands for commercial Unix.

RFT FOR DME. Recognizing that distributed management is not only the next logical step in defining the distributing computing environment but also itself the enabler for DCE, OSF has issued an RFT to "begin the process of establishing a vendor-neutral Distributed Management Environment."

In OSF's view, DME has two aspects: a common management framework and specific management applications. The RFT seeks technologies in both areas.

DME Goals. The major aims of DME are to simplify the administration and management of heterogeneous networks and to make it easier to write management applications.

For end users or administrators, DME must do the following:

- Improve the reliability and availability of systems and networks
- Increase the portability of user skills across different platforms
- Reduce the skill level and training required to perform management tasks
- Provide centralized management of distributed systems
- Extend the interoperability of open systems using common management services

ISVs need DME to provide tools for the simplified development of portable applications—tools comparable in sophistication to those found in proprietary systems—and to provide for the development of applications that manage both standalone and distributed systems. DME also should give systems vendors a way to reduce the development costs associated with systems management, as well as an environment for consistent management of heterogeneous systems.

Management Framework. The first step for OSF is to define a conceptual model for systems and network management. This framework defines the managed objects, management services, and management applications, and the relationships between them. While the framework looks a lot like a number of systems, including the Open Systems Interconnect (OSI) model, OSF has worked to make sure that its initial model does not favor any specific vendor. OSF had to make the model broad enough to be usable by its diverse members, such as IBM, Digital, and HP, each of which has its own management framework, as well as keep the model open to contributions by nonmembers with management architectures and technologies.

The model (see Illustration 1) defines four layers and the interfaces between them:

- Managed Objects. Managed Objects are representations of system resources such as devices; users; end-

user applications; and file, print, and mail systems.

- Common Management Services and Management Information Storage Services. Common Management Services include management communication, event, naming and location, and queuing services, which serve as the interface between management applications and managed objects. Management Information Storage Services enable management applications and managed objects to manipulate management information.

- Management Applications. Management Applications are an extensible set of applications that manage the objects through a set of interfaces to the Common Management Services.

- Human Interface. Human Interface is a consistent interface across all management applications. It can be either a graphical, screen-oriented, or command-line interface.

The interfaces between the layers also must be defined. Possible interfaces include the OSI CMIP/CMIS for management information and X Window for human interface information. More than one interface option could be defined between two layers.

Management Applications. The RFT defines certain key administration applications that OSF considers essential for early delivery of DME:

- Accounting
- Backup and restore
- License management
- Notification services
- Object monitoring and control
- Print services
- Software installation and distribution
- User management

All of these applications, and any others developed later, should fit into the finalized framework, making use of the common services and reporting to the human interface layer.

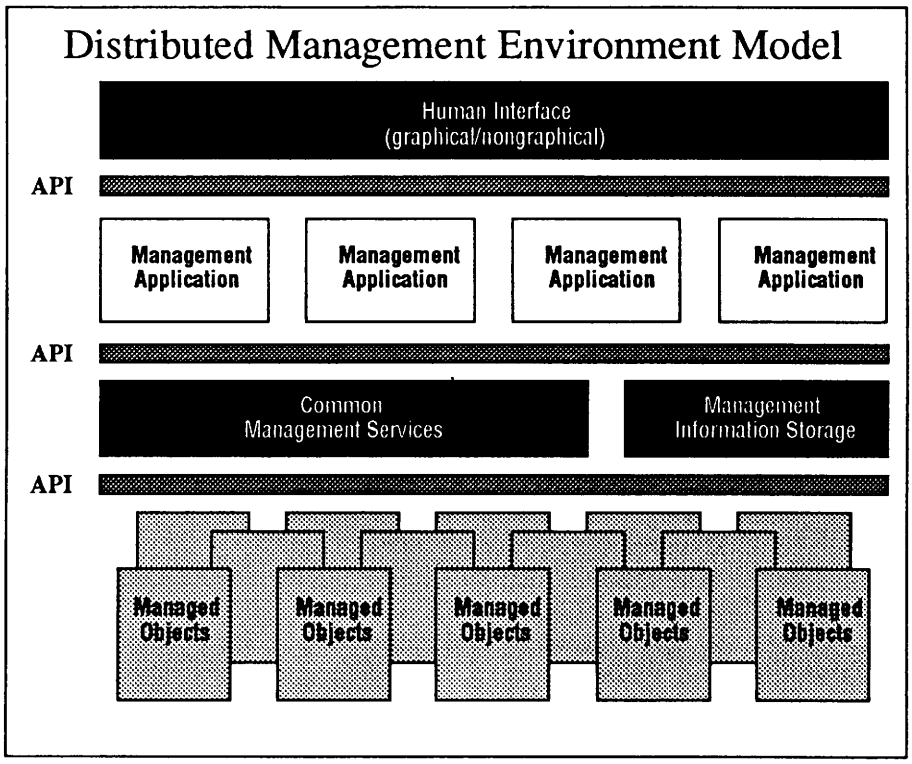

Illustration 1. The DME model defines object types, services, applications, and interfaces. It also defines the APIs and protocols required to isolate and communicate between layers. Managed objects and management applications must be extensible to implement the model.

Standards. OSF claims to be a software organization, not a standards body. It sees its role as offering software technology, specifications, and validation suites. DME is not designed to be a standard but a specific implementation of a set of standards. Thus, the RFT requires all submissions "to be consistent and conformant with industry-accepted standards," including relevant OSI standards, the X/Open Portability Guide, the IEEE Standard 1003.1 (Posix) system interface specification, and the relevant documents of the OSI/Network Management Forum (OSI/NMF) and the Internet Advisory Board (IAB). Implementations are required to be written in ANSI-standard C.

THE RFT PROCEDURE. OSF has modeled the DME request after its successful RFTs that led to Motif and DCE. The DME request grew out of the OSF special-interest group (SIG) on administration and network management. The SIG includes three OSF members that have their own network management architectures—IBM, Digital, and HP—and a number of other members committed to OSI management strategies.

The RFT calls for a letter of intent to respond by September 21, 1990, with full submissions due by December 15, 1990. The technical evaluation for the RFT will be conducted by OSF's European development office in Munich, West Germany. Announcement of the selected technologies and rationales behind the selections are expected to be made in the first half of 1991, with the first DME products possibly available by the end of 1991.

OSF will accept submissions that cover all or parts of the RFT, and it encourages submissions from both

members and nonmembers.

The criteria for acceptance of technologies include:

- Standards conformity
- Portability
- Validation and testing
- Documentation
- Product readiness

VENDOR STRATEGIES. Digital, HP, IBM, and AT&T are in particularly interesting positions regarding DME. Digital and HP each have their own management architectures, based on the OSI model. Both are certain to submit their technologies. The question for Digital and HP is whether the DME selections will be complementary to or in conflict with their own current and planned products.

For IBM, DME may come at just the right time. IBM is talking about making its OSI strategy equal to or even the umbrella for its SNA and NetView strategies. The near-term availability of vendor-neutral OSI implementations will make it much easier for IBM to pursue this course.

While Digital, HP, and IBM were very active in the OSF SIG, AT&T did not participate, making it the only vendor with a major network management architecture not represented. Indeed, it may be argued that AT&T has gone the furthest in network management, with its UNMA and Accumaster Integrator and its leadership in the OSI/NMF. It remains to be seen whether AT&T will submit a response regarding DME and how it will deal with the final framework and applications. OSF—and the industry as a whole—would like to avoid the scenario that led Sun to reject DCE completely. (For more on these vendors' individual network management strategies, see the following *Network Monitor* issues: Vol. 4, No. 9 for AT&T; Vol. 4, No. 10 for Digital; Vol. 5, No. 3 for Hewlett-Packard; and Vol. 4, No. 5 for IBM. We will revisit IBM's strategy in the October issue of *Network Monitor*.)

OUTCOMES. We see two sets of outcomes to the DME process. The first is specific to OSF members and their customers; the second will affect the industry as a whole.

OSF will deliver DME sometime next year, assuming that the technologies to create the whole framework are submitted (OSF is confident that they exist). DME will be delivered initially as a set of specifications, with some example applications. These representative applications—for example, backup and restore or software licensing—will be delivered on OSF/1. Soon after, OSF members (and perhaps others) will begin to deliver DME-compliant applications that run on a number of operating systems and platforms. For customers of these vendors, the move to robust, well-managed, easily administered open systems will have begun.

For the industry as a whole, the impact of DME is likely to be in two areas. First, the process itself should tell us a lot about the state of network management today and the feasibility of building applications to manage heterogeneous systems. For example, we should learn whether OSI's CMIP/CMIS is sufficiently rich to handle day-to-day systems administration.

Second, the process should focus the industry's attention on distributed systems management. Itself the convergence of systems administration and network management, distributed systems management is likely to borrow from both of these disciplines, as do the final DME specifications. This conceptual convergence may be as important as the early applications.

Ultimately, DME will be a success for the very reason we view DCE as a success even before any products have been delivered. Like DCE, DME will be instrumental in setting the industry agenda as we move toward the era of heterogeneous distributed computing. If it isn't now, management of the distributed environment will soon be high on everyone's agenda. We know it is high on ours. —*D. Marshak*

•SUN•

Simplifying Database Access

Last November, we reported the release of Ingres/Simplify, a graphical reporting and querying tool for the Ingres relational database. The product is noteworthy because it brings database access down to the user's level.

Ingres/Simplify was the fruit of a collaboration between Ingres and Sun. Now, Sun has extended the original product to reach other relational databases. Sun's new product—SimplifySQL—provides access to Oracle, Sybase, and, eventually, Informix.

GRAPHICAL QUERY AND REPORTING. SimplifySQL is designed to relieve SQL-phobic users of complex languages and MIS people of SQL-phobic users. It gives users a graphical view of data and a visual query editor. Instead of typing lines of SQL statements, users point and click on the information they need.

SimplifySQL has 4 components:

- SessionManager is used to select and access the proper database.
- DataBrowse is a graphical query editor and data displayer that lets users construct queries and browse through their results visually. Visual queries are automatically generated into SQL, but a standard SQL editor also is included.
- ReportWrite is a graphical report-formatting utility. With a mouse, users can construct a report layout, placing and sizing components such as text, data fields, headers, footers, columns, and subtotals.
- SchemaDesign provides a visual representation of the database. It gives users a view of the data that is available, how it is related, and how

it can be used. The tool also lets users define new data types and create or modify tables and views.

SimplifySQL, which is based on OpenLook and runs on Sun workstations, is part of Sun's data management strategy for workgroup computing. The company recently announced a new server strategy to better accommodate workgroup computing. It also has released a database accelerator for improving transaction processing performance and has become an active member of the SQL Access Group. Sun is promoting the fact that SimplifySQL gives users a consistent, intuitive, non-SQL front end to multiple databases on a network. Such tools are becoming more important for the growing number of casual users who need to access and interpret corporate data. *—L. Rowan*

•GUI•

New Faces for Old Applications

The catch in implementing X Window as an interface foundation for distributed environments is existing applications. Applications must be specifically written to X Window to participate, which eliminates a whole slew of existing programs. Nobody's going to scrap these applications and rewrite them for X Window. Instead, tools are needed to migrate existing applications to the standard—and, more specifically, to OpenLook and Motif.

ENTER ALEX. One such tool is Alex, from System Strategies (London, England). Alex is a programming language for writing either Motif or OpenLook interfaces to existing, character-based applications. The language only creates a new interface; it doesn't touch the actual application.

This fits nicely into the X Window architecture. X Window hasn't been embraced as a standard because of its graphical capabilities (you might even say that it was embraced in spite of its graphical capabilities) but because of its networking paradigm. X Window makes a clear distinction between interface and application functionality so that users can manipulate the interface of an application that resides on another workstation or host on the network. With Alex, the interface is accessible to any X Window server, while the original non-X Window application can reside elsewhere on the network—even Cobol applications on mainframes.

How Alex Works. Alex is definitely a programmer's tool. Using it is a matter of defining widgets (such as X Windows with I/O capabilities like scroll bars, dialog boxes, etc.), controlling events (like I/O events and callbacks), and assigning proper application processes and files. Alex runs the application, captures the character-based I/O, and translates it into a graphical I/O. The user then interacts with the graphical version, and Alex translates user input and sends the instructions to the application.

Marketing and Availability. Alex has been available for OpenLook for some time. The announced Motif version hasn't yet been released.

System Strategies is marketing Alex to user organizations that are trying to move to an X Window environment but that already have a big investment in mainframe applications. It has started an OEM campaign as well. Its first major OEM contract is with AT&T, which will market Alex with its 6386 WorkGroup System line.

COMMENTS. Alex is ideal for organizations that have a hefty investment in terminal applications, but System Strategies isn't alone in providing such a tool. UIMX, an interface product from Visual Edge Software (Montreal, Quebec), includes a module for developing X Window-based graphical front ends to existing applications. And last May, IXI Limited (Cambridge, England) announced a tool called X.deskterm, which is similar to Alex.

Each of these solutions, however, caters mainly to terminal-based organizations. If only similar tools were available to migrate workstation and PC applications—both character-based and graphical. (Even a Windows 3.0 application could use the networking advantages that X Window offers.)

Many organizations have been clamoring for ways to integrate Unix with the systems they already have in place. Migration tools like Alex certainly are a welcome technology.

—L. Rowan

•INGRES•

Another Boost for Distributed Database

Distributed database is slowly becoming a reality as major relational DBMS vendors enhance their products' ability to gracefully handle transactions that span multiple databases. Many vendors have for some time offered distributed query processing—the ability to join tables across multiple databases (local and remote) and view the data as if it came from a single local table. The more difficult undertaking has been allowing a single transaction to *update* data across two or more databases. The key is to ensure that a transaction consisting of updates to multiple nodes is committed in its entirety or not at all. If the transaction is not successful at any site, it must be rolled back at every participating site to maintain data integrity and a consistent view of the database.

The common approach to distributed transaction management is a two-phase commit (2PC) protocol. The site controlling the transaction instructs each remote site on the operation it is to perform and issues a "prepare to commit." When each site is ready, it

responds with a message that it is able to commit the requested transaction. When all sites have responded, the originating site sends out messages to commit the transaction.

INGRES NOW HAS 2PC. Ingres now offers an automatic two-phase commit capability, in Release 6.3 of its Ingres/Star software. Ingres/Star can now coordinate an update that spans multiple Ingres databases and can carry out the recovery (rollback) process if any part of the transaction fails.

An important prerequisite of achiev-ing an automatic 2PC was teaching the Ingres RDBMS server how to prepare to commit. This was implemented in Version 6.3 of the server product, announced last fall. With this enhancement, Ingres developers can implement a 2PC protocol manually. Ingres/Star has now caught up with the Ingres server technology, and the 2PC is fully automatic.

Ingres describes its 2PC as heterogeneous, meaning heterogeneous at the hardware, operating system, and networking levels, but not at the database level. Although Ingres offers gateways to other DBMS data (such as Digital's Rdb and RMS), Ingres/Star does not yet implement 2PC across heterogeneous databases. A major stumbling block is the fact that a DBMS must understand how to prepare to commit before it can participate fully in an automatic 2PC process.

Ingres/Star runs on Digital VAX/VMS, Sun Sparc, and Sequent platforms. Release 6.3 will be an automatic update for customers who already have Version 6 of Ingres/Star.

BUT WAS IT FIRST? Although we are impressed with Ingres's technology and its determination to push the state of the art, we were taken aback by its claim to be the first company to offer automatic 2PC. At least two companies offered automatic 2PC before Ingres. InterBase from Interbase Software has had automatic 2PC for a long time, and Progress Software beat Ingres to the punch by three weeks when it announced 2PC in Version 6 of Progress (see the feature article in this issue).

Ingres obviously is focusing on more immediate problems, namely its larger competitors—Oracle, Informix, and Sybase. Oracle and Informix still do not have 2PC; the Sybase 2PC is not automatic and must be programmed into the application. However, in spite of its good technology, Ingres has lost some market momentum and visibility over the past year. One reason may be the fact that Ingres chose to invest so much time and energy in developing specific platform relationships—such as those with Digital (optimizing Ingres as Digital's RDBMS offering for the Ultrix platform) and the Santa Cruz Operation (as part of Open Desktop), and didn't pay enough attention to the general RDBMS market. Any misstep can be costly in such a fiercely competitive environment, and Ingres is working hard to recoup its position on the marketing front. —*J. Davis*

Note: Another important development occurred as we went to press: ASK Computer Systems announced an agreement to acquire Ingres Corporation for $110 million. We will cover the details and potential ramifications of this in a future issue.

•386 UNIX•

En Route to a Standard Unix Desktop

Once again, AT&T and Intel are campaigning for a standard 386 Unix system, but this time, they're wisely teaming up with the Santa Cruz Operation (SCO). AT&T and Intel tried to merge Unix and Xenix to come up with a 386 standard a few years ago—at that time, in a partnership with Microsoft—resulting in Unix/386. Unix/386 was inadequate, however, mostly because it didn't offer a standard binary interface.

Another part of the problem was that SCO wasn't involved in the project until it was too late. You just can't build a successful Unix PC standard without involving the one vendor that owns the overwhelming majority of the Unix PC market.

This time, prospects look brighter. Not only have AT&T, Intel, and SCO defined a binary-compatible Unix operating system standard for 386 and 486 machines, but also other organizations are eager to implement it. Even Interactive Systems, SCO's biggest competitor, has announced support.

SYSTEM DEFINITIONS. The new binary-compatibility specification—iBCS Edition 2—is based on Unix System V Release 3.2 and will include support for NFS, TCP/IP, and X Window. Apparently, AT&T acknowledges Release 3.2 as the appropriate PC standard, which makes us wonder about the validity of Release 4.0 on the desktop. It also makes us wonder how the PC standard will fit into the schemes of OSF and Unix International, because it will be incompatible with the operating systems being offered by both organizations. A PC Unix standard might actually be somewhat of a hindrance for commercial Unix, since it is yet another operating system version that customers and ISVs have to deal with.

THE OPPORTUNITY. Despite its incompatibilities with current Unix standard contenders, iBCS could benefit substantially from the lackluster reception that OS/2 has received from the desktop market. Timing is everything. Just a few months ago, we worried that the Unix community was so busy squabbling over user interface and operating system issues that it didn't notice OS/2 quietly gaining momentum at the desktop. Meanwhile, Windows 3.0 has come along and revived DOS (at least temporarily), and OS/2 popularity has begun to fade. Now Unix has another chance. Perhaps a shrink-wrapped PC Unix may become a reality after all. —*L. Rowan*

Patricia Seybold's Computer Industry Reports

Topics covered in Patricia Seybold's Computer Industry Reports in 1989/1990:

Back issues are available, call (617) 742-5200 for more information.

Office Computing Report

1990—Volume 13

#	Date	Title
1	Jan.	Office Redefined—A New Model for a New Decade
2	Feb.	Visual Programming—Is a Picture Worth 1,000 Lines of Code?
3	Mar.	Word Processing in the '90s—How MS Word for Windows Stacks Up
4	Apr.	Multimedia—The Promise, the Players, the Issues
5	May	Is X the Window? X Window and distributed Applications
6	Jun.	HP's Office Strategy—Extending NewWave Beyond the Desktop
7	July	Lotus Notes—A Platform For Developing Workgroup Applications.
8	Aug.	Odesta's ODMS—The process of Document Management
9	Sept.	Microsoft Mail—An Enabler for Cross-Workgroup Applications

UNIX in the Office

1990—Volume 5

#	Date	Title
2	Feb.	Multiprocessing—Towards Transparent Distributed Computing and Parallel Processing
3	Mar.	IBM's Second Generation RISC—With its new RISC System/6000, IBM claims center stage.
4	Apr.	Hewlett-Packard's NewWave Office—Distributed Object Management in an Open Environment
5	May	Hewlett-Packard's NewWave Office—Distributed Object Management in an Open Environment
6	Jun.	Desktop Unix—Open Desktop Aims to Be the Rallying Point for ISVs
7	July	A/UX 2.0—Apple Targets the Unix Desktop
8	Aug.	Unix OLTP—Getting Ready for Commercial Prime Time
9	Sept.	The Progress 4GL and RDBMS—Shooting for the Mainstream

Network Monitor

1990—Volume 5

#	Date	Title
1	Jan.	Distributed Computing Environment—Interoperability Now!
2	Feb.	LAN Manager 2.0—Jumping Up to the Major Leagues
3	Mar.	A New View of OpenView—HP's Network Management Strategy
4	Apr.	A Study in Contrasts—Unisys's Open Networking Strategy
5	May	Xerox Panos—Delivering on Distributed Networking
6	Jun.	Soft•Switch—Blazing the Trail for Mail in Distributed Computing Environments
7	July	OSF's DCE—Solutions for Heterogeneous Interoperability
8	Aug.	Banyan—Banking on Corporate-Wide PC LANs
9	Sept.	Novell NetWare 386—Platforms for Integration into DCE