Patricia Seybold Group

Editor-in-Chief
Michael A. Goulde

INSIDE

EDITORIAL

Page 2

***Benchmarks** provide customers with a way to compare performance of both hardware and software before buying. However, commercial benchmarks are often misleading and misused, bearing little relationship to real-world requirements.*

ANALYSIS

Page 18

*AppWare is **Novell's** strategic thrust to evolve its business to application middleware and development tools. But a critical consideration is whether ISVs will find AppWare a suitable environment in which to invest. • Years of divergence among various Unix offerings may now be resolved with a proposed common Unix System API specification. Constructed from a combination of existing standards and interfaces commonly used by leading applications, the specification will become the dominant criterion for determining what "real" Unix is.*

OPEN INFORMATION SYSTEMS

Guide to Unix and Other Open Systems

Vol. 8, No. 9 • ISSN: 1058-4161 • September 1993

Apple Computer and Open Systems

The Other Desktop Standard

By Michael A. Goulde

IN BRIEF: Since its introduction in 1984, Apple's Macintosh has been regarded as the standard against which other graphical user interfaces are measured. However, because of Apple's refusal to license its technology, the Macintosh has played a secondary role to the de facto desktop standards set by Intel-based PCs and Microsoft Windows. Recognizing that its future may be jeopardized by continuing to pursue a proprietary direction, Apple has begun several initiatives that represent the emergence of a competitive open systems strategy. New Macintoshes based on the new PowerPC RISC microprocessor are expected as early as Spring 1994, and Apple has plans to license technology that will provide a Macintosh interface and allow Macintosh applications to run under Unix on other vendors' platforms. The company is also working with several partners to license key Macintosh technologies and standards to compete against those offered by Microsoft. Perhaps more importantly, it has defined an architecture and methodology, called VITAL, that provides a blueprint for customers who want to build vendor-independent information systems for the distributed enterprise. VITAL will help establish credibility in open systems, complement Macintosh's technologies, and position the company to offer real competition to Microsoft. *Report begins on page 3.*

Commercial Benchmarks

So Much Ink, So Little Value

HARDWARE AND DATABASE vendors make extensive use of benchmarks to position the performance of their systems and database engines against the competition in an effort to convince customers of the superiority of their products. Especially as we face the annual barrage of fall system announcements from Compaq, IBM, Digital, Sun, and HP, the avalanche of benchmark numbers overwhelms and under-informs. The confusion generated by benchmarks is particularly frustrating when it comes to commercial applications, and, when distributed, client/server applications are considered, the problem is even worse.

First of all, many aspects of technology contribute to a system's performance. These include the processor itself, compiler design, system architecture, and subsystem performance. Processor variables include characteristics like clock speed, cache size and architecture, instruction set, pipelining, and number of processing units. Optimizing compilers can make a tremendous difference in performance. System architecture variables include multiprocessor architecture, processor-to-memory bandwidth, bus speeds, I/O capacity and bandwidth, and more. Subsystems include storage and networking characteristics. Benchmarks have to be designed to measure performance independent of any design characteristic so that the results can be compared from one system to another. And then the comparisons have to be made among like systems, either in terms of configuration or cost, or some combination. Apples-to-grapes comparisons only yield fruit salad.

What is relevant to measure depends on the nature of the workload. Scientific and technical applications rely more on CPU speed, floating point performance, and memory size and throughput. Commercial applications rely more on broader system design issues. Benchmarks for scientific and technical applications, such as Linpacks, Specmarks, Floating Point Operations, and various measures of graphics performance, seem well-accepted in their segments. Commercial benchmarks, on the other hand, are a moving target.

Database transactions have received the most widespread attention, and now we have what is at least the third generation of transaction processing (TP) benchmarks, the Transaction Processing Performance Council C Benchmark (TPC-C). This is designed to more accurately reflect a complex OLTP application containing five transaction types that model a real business application—order entry.

However, commercial applications are not all transaction-oriented. For example, decision support involves query and data manipulation. How do we measure that? Even more critical is measuring the performance of a broad range of client/server applications in which both the network and the client machine enter the equation.

Recently, Patricia Seybold published a commentary in *Computerworld* pointing out users' difficulties in getting acceptable performance from client/server applications. Her E-mailbox was flooded with tales that far exceeded the gloomy picture she portrayed. How do we measure client/server performance and define what is acceptable? In host-terminal architectures, we have acceptable measures of response time distribution. Are those same measures applicable in client/server applications? And against what workload?

It is time for the industry to step up to this problem and form a Client/Server Performance Council, comprising vendors and users, to develop acceptable benchmarks. If you think this is a good idea, send me an E-mail note at mgoulde@mcimail.com. I will forward your comments to the vendors that have been hyping client/server architectures as (1) their strategies for the '90s, and (2) the answer to users' prayers.

OPEN INFORMATION SYSTEMS

Editor-in-Chief
Michael A. Goulde

MCI:
MGoulde
Internet:
mgoulde@mcimail.com

Publisher
PATRICIA B. SEYBOLD
Analysts and Editors
JUDITH R. DAVIS
STANLEY H. DOLBERG
MITCHELL I. KRAMER
DAVID S. MARSHAK
RONNI T. MARSHAK
JOHN R. RYMER
ANDREW D. WOLFE, JR.
Copy Editors
ALBERT C. D'AMATO
MIRIAM F. D'AMATO
Art Director
LAURINDA P. O'CONNOR
Sales Director
PHYLLIS GUILIANO
Circulation Manager
DEBORAH A. HAY
Customer Relations Manager
DONALD K. BAILLARGEON

Patricia Seybold Group
148 State Street, 7th Floor,
Boston, Massachusetts 02109
Telephone: (617) 742-5200 or
(800) 826-2424
Fax: (617) 742-1028
MCI: PSOCG
Internet: psocg@mcimail.com
TELEX: 6503122583

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

POSTMASTER: Send address changes to *Open Information Systems*, 148 State Street, 7th Floor, Boston, MA 02109.

Apple Computer and Open Systems

The Other Desktop Standard

Apple Computer Incorporated (Cupertino, California) is in the midst of perhaps the most significant transition in its history. The company that built the computer "For the Rest of Us" appears to many to be losing momentum and competitive advantage at an alarming rate. Once admired for the elegant integration of hardware and easy-to-use software, the Macintosh has become a confusing collection of point products. Only die-hard Macintosh aficionados know the difference between the Classic, Centris, Quadra, Quadra AV, Workgroup Server, LC, LCII, LCIII, Performa, Duo, and PowerBook. Some of these products are the same machine with different names attached for different markets. The first instantiation of the company's personal digital assistant strategy, the Newton, has had a rough start, requiring several upgrades to the operating system in its first few months on the market. Over the past few years, products have been late to market and are sometimes delivered before they are really ready. New strategies appear overnight but disappear just as quickly. Apple used to be led by visionaries like Steve Jobs and Jean-Louis Gassée, who inspired customers and employees alike. Today's visionaries seem to have difficulty just articulating their vision.

Stalled Growth and Profitability

Although Apple's sales continue to grow, they are growing no faster than general PC industry growth, leaving Apple's market share flat. Unit sales grew 40 percent in 1991 following sharp price-cutting but only 18 percent in 1992. Price pressure from the Intel PC market has forced continuing price reductions, hampering revenue gains and causing gross margins to plummet to what are—for Apple—historic lows. The largest layoffs in the company's history and cutbacks in R&D spending are part of an ongoing process to slim down the company's cost structure so that it can compete with commodity PC vendors. But at what price? And Chairman John Sculley's dream of a Personal Interactive Electronics revolution entails too much risk to warrant betting the entire future of an $8 billion company on it.

A New Direction

However, Apple's new directions and its strategy for renewed growth are beginning to become clear. Its three-pronged strategy includes workgroup and networking solutions for business, desktop and notebook systems and hand-held appliances for individual business users and consumers. On the hardware side, Apple is migrating from the Motorola 680x0 microprocessor to the PowerPC RISC microprocessor, which Apple engineers designed in conjunction with Motorola and IBM. Its Open Systems Software group is charged with creating portable, client/server solutions for business, government, and education customers based on key Apple technologies and de facto and formal standards. Apple wants to play a major role in distributed computing and is articulating some of the best ideas about how to implement multi-tier client/server architectures. It is in a good competitive position to do so. The Macintosh is the only technology with a large enough installed base—over eleven million worldwide, according to *The Hartsook Letter*—and a large enough volume of annual shipments—around 3 million units—to seriously challenge Microsoft Corporation (Redmond, Washington) at the desktop. And multimedia will play as large a role in Apple's future as it will in that of any other company in the industry.

The company has been developing an open systems strategy that is a marked departure from past Apple strategies. Although it has no plans to license the MacOS to all comers, it does want to Mac-enable other vendors' platforms. While the Common Open Software Environment (COSE) Common Desktop Environment (as currently defined) doesn't hold a

lot of interest for Apple, many COSE participants are key allies in Apple's future plans. It has begun to view standards and consortia differently than it has in the past. Servers have begun to play a more significant role in its business model. As the countdown to PowerPC continues, open systems are playing an increasingly important, and visible, role for Apple.

The Real Windows NT Challenge

According to various market researchers, the Macintosh has between 13 and 17 percent of the installed base of desktop computers. The problem Apple faces is that its share of mind among customers in the marketplace is also only about 17 percent. In the big Windows NT versus Unix debate, why is it that no one argues about the impact Windows NT will have on the Macintosh? All of the attention is on a much smaller market for Unix desktops. And why haven't the Unix vendors taken a more favorable view of Apple's open initiatives as a competitive weapon in the war against Microsoft? Apple needs to make the Macintosh a viable—and valuable—desktop component of enterprise open systems architectures, not a wart that MIS has to suffer to integrate and not an aside that the open systems community dismisses.

Nature of the Challenge

As an integrated hardware and operating system platform, the Macintosh meets the classical definition of a proprietary system. The Macintosh must compete simultaneously against other hardware platforms and other operating systems. The rest of the industry has adopted the view that a company can be in the hardware business or in the system software business, but not both. NetWare and Unix are products of a software company. Windows comes from a software company. In spite of the trend in the industry to separate hardware and software suppliers, Apple has persisted until now in resisting the proposition that wide availability of system software on many vendors' platforms makes customers feel safe and protected.

Finding a Viable Model

The model of supplying both hardware and operating system software is becoming less viable, particularly in a high-volume, commodity market such as desktop computers. Apple has finally begun to lay the groundwork for unbundling its operating system, interface, and services and making them available on other vendors' hardware platforms. However, the success of this strategy will depend on how many hardware companies will license system software from a competitor and actively promote it. Apple can't change its business model the way NeXT and Wang did and become a software company. With an installed base many times that of Unix workstations, such a move wouldn't make sense. Apple needs to continue selling hardware, but it also needs to shed the proprietary label and become identified with open systems.

Finding Market Share

Apple will have to finesse selling hardware and software in competition with the licensees of its software products. In order for Apple to do that, licensees must be shown the opportunities that might stem from a market expanded by multiple sources of Macintosh-capable systems. That expanded market can come from only two sources: market growth and expanded market share. Market growth might be fueled by wider availability of easy-to-use systems. Gaining market share primarily means capturing business from Microsoft. In order to do that, Apple needs a strategy that extends the reach of its products beyond its own customer base and beyond its own channels. That kind of a strategy must be based on open systems.

"Open systems" has many meanings, and they are all in play at Apple. The company has gone from a closed hardware architecture and proprietary operating system with open interfaces to an open hardware architecture and a more open operating system strategy. This report will examine the key components of Apple's approach to open systems: Apple Services for Open Systems, including OpenDoc and Apple Open Collaboration Environment (AOCE); the Power products (PowerPC, PowerOpen, and Macintosh Application Services); and Virtually Integrated Technical Architecture Lifecycle (VITAL). Apple has the foundation to compete successfully, but it cannot do it alone.

Apple Services for Open Systems

Apple recognizes that it has core technology which can provide the foundation for advanced cross-platform capabilities that other vendors can capitalize on in order to fulfill the objectives of open systems. The umbrella term for these technologies is Apple Services for Open Systems (AppleSOS). In many ways, AppleSOS is analogous to Microsoft's Windows Open Services Architecture (WOSA) and is designed to expand Apple's scope and take the company into different markets. The strategy is to use the AppleSOS architecture to define the technology components that will be built to be both cross-platform and cross-operating system capable. The services encompassed include everything from running Macintosh applications (Macintosh Application Services) on other platforms to data access (Data Access Language, or DAL), to document interchange (OpenDoc), and others. Illustration 1 shows the structure of the unbundled components of Apple SOS.

Apple Services for Open Systems

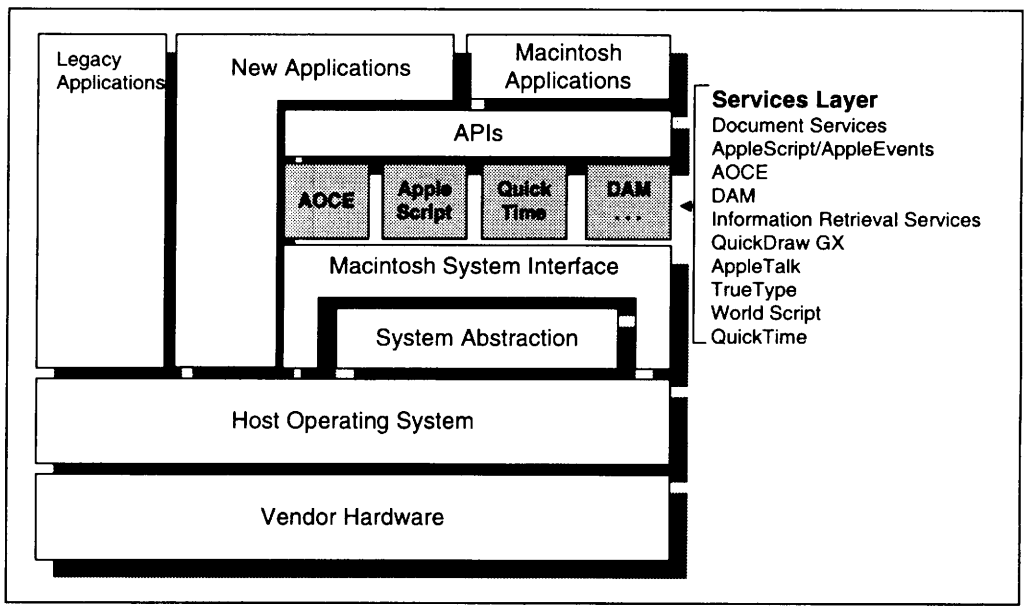

Illustration 1. The services layer brings key Apple technologies and services to other platforms, coexisting with existing applications while supporting or interoperating with Macintosh applications. APIs are cross-platform, and the Macintosh System Interface provides System 7 services.

One of the key differences between AppleSOS and Microsoft's WOSA is that WOSA is an integral part of its Windows products, while AppleSOS services can be implemented on any vendor's platform. WOSA is only available on Windows, and, although Apple will implement its services as an integrated part of the Macintosh operating system, the architecture allows those services to be implemented independently on other operating systems as well. By unbundling services and making them available separately, Apple will give other vendors a choice of technologies to incorporate in their systems. In fact, in several areas, like the Open Collaborative Environment (OCE) and DAL, Microsoft and Apple are working to bring AppleSOS to Windows. Apple has already begun lining up other vendors to support its strategy, including IBM (Armonk, New York), Sun Microsystems Computer Corporation (Mountain View, California), Hewlett-Packard Company (Palo Alto, California), WordPerfect Corporation (Orem, Utah), Borland International Corporation (Scotts Valley, California), and others. Apple has recently announced an integration center that will, in fact, concentrate on delivering interoperable implementations of its services on other platforms.

Unix vendors in particular stand to gain from adopting Apple's technologies. Unix has never been known as a user-friendly operating system. Even efforts such as the Common Desktop Environment do not bring Unix usability close to the level of the Macintosh or, for that matter, even close to Microsoft Windows, in terms of ease of use. Proliferation of Macintosh capabilities by Unix vendors on their platforms would help them sell Unix systems and also help Apple continue to drive the Macintosh as a viable alternative in corporate accounts. Making certain Macintosh services available on Unix would also increase the scalability of those services beyond what Apple itself can provide with the Macintosh.

Apple's VITAL: Blueprint for Enterprise Client/Server

As one of the few remaining proponents of a proprietary desktop computer system, Apple would be the last company one would expect to define a completely open, distributed information architecture. However, because the company was faced with an internal information management challenge the equal of any other $8 billion company, it was forced to define just such an architecture, which it has subsequently packaged as Virtually Integrated Technical Architecture Lifecycle (VITAL). This architecture is both a model for client/server computing and a framework for integrating the desktop into a global enterprise environment.

VITAL Is Vendor Neutral

Uncharacteristic of most vendors' architectures, VITAL does not specify any particular implementation. The initial motive behind the definition of the architecture was based on the multiplatform, multivendor environment of Apple's existing information systems. The company shared the same requirements and concerns of any large, global company. It needed to provide timely access to accurate information to users throughout the world in a rapidly moving and fast-changing industry. And it had to leverage legacy systems wherever possible.

Five Environments

VITAL is a proposed architecture for enterprise data management and data access in a multiplatform, multiprotocol environment. The VITAL model defines five conceptual computing environments shown in Illustration 2: Desktop Integration, Data Capture, Data Access, Repository, and System Infrastructure. Its data management model uses a combination of operational databases and data warehouses which are both distributed and multi-tiered. Global warehouses feed mid-tier warehouses, which, in turn, work with local access servers to provide local clients with data services. VITAL is based on replication and propagation of sharable data from operational databases based on authorization and the economics of data distribution. It offloads the reporting and decision-support functions that make system configuration and tuning of operational systems so difficult.

Repository Environment

A VITAL repository manages the metadata and has the software services and databases needed to supply the access control, routing, and synchronization required for distributing information from Shared Data Warehouses to information consumers. It provides for consistent business rules, data definitions, system usage rules, and resources. The repository is designed around the three-schema approach, maintaining an internal schema representing the data in a DBMS format, a logical or conceptual schema reflecting the business concepts, and the external schema, which is how users view the data. This separates the physical structure of the data from the way the data are used. The logical view is an aggregate of the information contained in all the individual databases in the Shared Data Warehouse environment. The Repository defines all globally accessible applications, services, tasks, data, and other resources on an enterprise network. It controls both the content and dissemination of information about data dictionaries, catalogues, code libraries, and directories. Also stored in the Repository is configuration-management information which allows different database versions to be synchronized.

REPOSITORY GUIDE. The Repository Guide describes many different directories, each of which performs a specific function. These include shared data directories, which define stored procedures; a shared services directory, which contains usage profiles; security and user

directories, which provide access control information; and the "where-used" directory, which contains information necessary to locate and notify clients of changes in shared resources.

VITAL's Five Environments

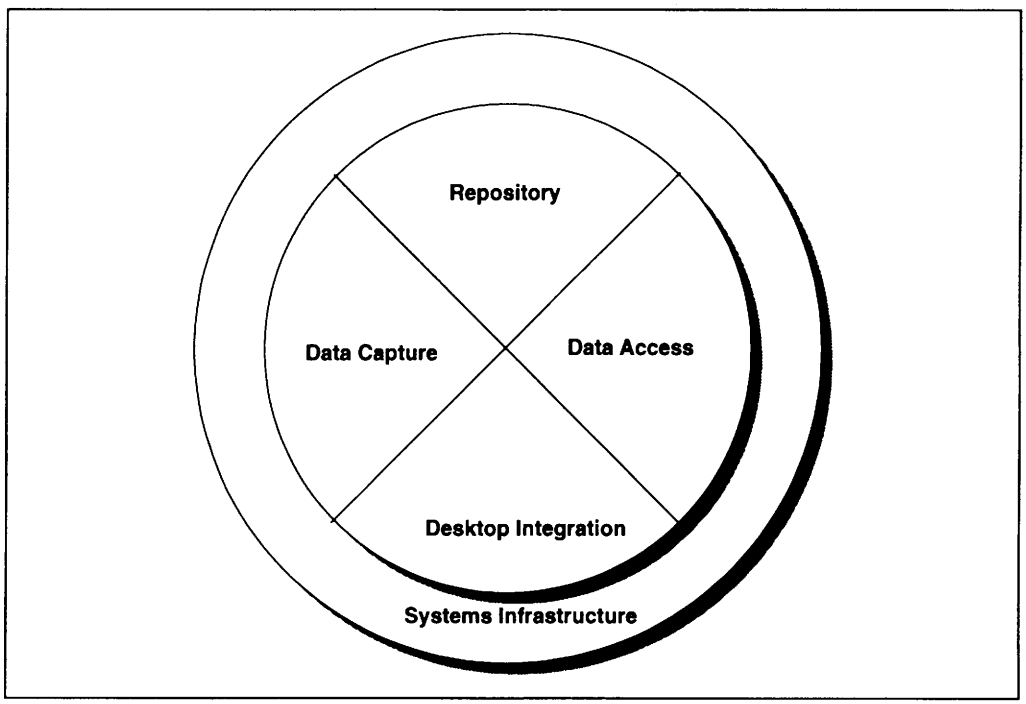

Illustration 2. VITAL consists of documents describing the characteristics and requirements for five areas of functionality in wide area, client/server application environments.

NAVIGATION SERVICES. Two navigation services help client applications locate the optimal instance of a service. All service-offering functions register with the Sharable Services Directory. An information consumer, usually an application, calls the Dynamic Selection Service to locate the best-suited provider of the service. No specific method for identifying the best-suited provider has been defined yet, but that principle is defined.

There are no products on the market today that implement the entire Repository environment, although certain functions, such as data replication, update contention management, and transaction monitoring, are performed by available products. It is possible that AOCE could pick up some of the functions. In the meantime, Apple encourages the use of existing products, such as IBM's AD/Cycle or Digital Equipment Corporation's (Maynard, Massachuetts) CDD/Repository, integrating where possible, but recognizing that today's capability for integration is limited. Customers would be well-advised to consider international repository standards in designing their own VITAL implementation.

Data Capture

The Data Capture environment is mostly concerned with creating and maintaining valid data in operational systems. It is primarily transaction-oriented: receiving, validating, and routing transaction requests; handling commit and rollback processing; managing the scheduling of transactions; and propagating snapshots of central databases when certain events, such as database updates or transaction rollback, have occurred. It also provides indirect write access to an enterprise's data warehouse for all "external" sources of information, i.e., data sources not integrated into the local VITAL Repository.

An important goal of the Data Capture environment is to validate new data from external sources without placing demands on the source database, particularly if that database is on a

remote host. This requires having an application that creates a bridge between the external source and the VITAL information environment. Incoming information is converted to standard schema and uploaded in the form of a master copy, separate from the originating database. Master copies are contained in designated data warehouses, which are the sole sources of information about particular topics or functions and are fed data by distributed "operational" databases. Desktop systems that are a part of the Data Capture environment support normal data entry and validate new data entering the VITAL environment. Updates from capture systems are queued for later processing asynchronously. The importance of messaging comes through clearly in Apple's plan.

Data Access

Data Access uncouples the distribution from the collection of information and distributes already processed information from data warehouses to information requesters. Key to providing data access are network data warehouses, which improve access to shared data and provide a distribution system that is independent of the transaction-oriented databases which collect the information.

Information that has been validated and is available for sharing is shipped from the appropriate operational databases in the Data Capture environment to the network of Shared Data Warehouses (SDWs) in the Data Access environment. Data warehouses may periodically produce summaries of information for use locally in Local Data Warehouses. In order to reduce overhead, infrequently used information can be transferred on demand to limit resource utilization. The SDW network manages and distributes information on behalf of the owners of a source database. Data owners, which are part of the Data Capture environment, are responsible for the accuracy and validity of the information provided to the warehouses.

Databases in the VITAL environment can be of just about any type—relational, object-oriented, CODASYL, or even flat files—providing they support SQL queries from the Data Access environment. While SQL is the standard data access language, VITAL does not require a specific method. System 7's Data Access Manager (DAM) can route queries either through DAL, Oracle's SQL*Net, the Sybase Connection tool, or Microsoft Open Database Connectivity (ODBC) drivers. Ideally, the warehouse database should be run on the most appropriate platform and physically installed where it makes the most sense.

Desktop Integration Services

The desktop represents the client side of the architecture and can be either the source of newly-captured information or a consumer of information. The Desktop Integration environment is connected to the Data Capture, Data Access, and Repository environments by service modules, relying upon Data Access for read and Data Capture for write access. The concept behind the Desktop Integration environment is that the user should be shielded from the details of configuration, access methods, and location. The Desktop Integration environment consists of 31 independent services, including electronic mail, authentication, transaction and query submittals, and software license management, all of which are available to client programs. An Integration Services Manager is the primary link between client applications and the VITAL services.

System Infrastructure

Apple's own internal information architecture is a prototype of the environments for which VITAL is intended. It consists of IBM mainframes, Digital VAX/OpenVMS systems, Unix workstations, and, of course, Macintoshes. The only assumption made about the infrastructure is that it provides all of the required services.

Remote access and file synchronization are central to Apple's vision of future computing. VITAL aims at providing a complete, albeit highly abstracted, model of enterprise computing. The level of abstraction is intended to allow VITAL to apply equally to all forms of business.

The VITAL model does not prescribe a fixed enterprise computing system. The hardware changes, the software services change, available applications change, and data change. Under VITAL, all of these changes propagate through the enterprise network, rather than being imposed all at once throughout.

From Concept to Implementation

Apple is in the midst of bringing its internal information systems into alignment with the VITAL model. It has developed an extensive set of procedures and methodologies for doing so which are set out in the VITAL documentation. The company appears to have a sound understanding of the technological, information, business, and organizational issues that are involved in re-architecturing an enterprise information infrastructure. The challenge it faces is how to leverage VITAL in its product activities.

Apple is qualifying third parties to provide training and consulting using VITAL methodology. By May 1993, Bear River Associates (Berkeley, California); KMPG*ExIS (Palo Alto, California); Martin Marietta's Electronics, Information and Missile Group (Englewood, Colorado); and RWD Technologies (Columbia, Maryland) had signed up.

VITAL is unique in the industry in that it defines a product-independent architecture and framework for a broad range of different types of client/server implementations. We expect Apple to sign agreements with several key companies before the end of the year in which VITAL will become a focal point for a broad range of consulting services and product implementation.

Data Access: DAM, DAL, and ODBC

The Data Access Manager (DAM)

Apple's current implementation of the Data Access architecture is built around DAM which is incorporated within the Macintosh operating system and provides an application programming interface (API) for data access. One of the protocols that DAM supports is the Data Access Language (DAL) implemented in the DAL client software. Until System 7.1 was released, DAL was an integral part of the Macintosh operating system. Now it is supplied by database access application vendors or through the Apple Professional Developer's Association.

With the introduction of System 7 in 1990, Apple added DAM as a high-level data access abstraction. DAM sits above specific data access protocols, such as DAL, and provides two APIs. A high-level interface, which is accessed using just five programming calls, is used by applications to invoke prepared SQL routines stored in special query documents that may be stored on the client or on the server. A low-level API is made up of 13 programming calls, which support a more dynamic and interactive interchange between client and server.

DAM is not tied to DAL. Database vendors with their own client-side APIs can work inside DAM so that client applications can access either DAL or other proprietary database interfaces through the same basic code. Optimal APIs can be chosen and tailored for a particular database alongside other general calls that work with any DAL-supported database. Developers can make the speed-versus-generality decision with each application developed.

How DAL Operates

DAL was originally developed as CL/1 by Network Innovations, a company Apple acquired in 1988. Digital developed VAX-based DAL server software to provide the programmatic interfaces between DAL and the SQL language requirements of databases from Oracle Corporation (Redwood Shores, California), Sybase Incorporated (Emeryville, California), Ingres Products Division of ASK Corporation (Alameda, California), and Digital's own Rdb. Apple's more recent association with IBM has resulted in similar DAL servers for IBM mainframes and AS/400 systems. In addition, Blyth Software Incorporated (Foster City, California) developed a DAL server for mainstream databases running under Unix, and

Data Access: DAM, DAL, and ODBC

Novell Corporation's (Provo, Utah) NetWare SQL supports DAL as well. Microsoft is doing an implementation of DAL for Windows as a part of its ODBC-DAL agreement with Apple.

DAL is available for over 17 data sources on 14 platforms, and it consists of a client component for the Macintosh and a server component that is platform-data source specific. The client component of DAL is a system software extension along with associated files that provide a desktop application written to the DAL API with transparent access to data stored on any supported host database. DAL applications make calls to the Macintosh DAL client, which reads a DAL Preferences file to determine the route to the data source. Calls then go through the appropriate DAL network transport and to the DAL host server. The DAL Server receives requests from the client application which are in the ANSI-based DAL SQL dialect, and carries out the requests in the dialect of the target database. DAL allows desktop and mobile client applications to access and merge data concurrently from multiple sources. The DAL client includes support for many applications, such as Lotus 1-2-3 and Microsoft Excel, in addition to API support.

ODBC Support

Apple entered into an agreement with Microsoft to support ODBC through DAM and DAL. (See Illustration 3.) The Mac-based DAM will be extended to support the complete 51-function API for ODBC in addition to the existing low- and high-level DAM routines. All ODBC functions will be implemented through DAL functions, so any existing DAL server can be accessed by a Macintosh client application using either the DAM or ODBC APIs.

As part of the agreement, Microsoft is supporting a DAL client in Windows for access to DAL servers. Developers can also write to the ODBC API and access an ODBC driver that is integrated with the Macintosh DAL client, thus providing access to all DAL-supported databases or to other databases for which an ODBC Macintosh driver exists.

Advantages for Developers

Providing both ODBC and DAM functionality on the Macintosh represents an effort on Apple's part to attract more third-party database vendors to its platform. In addition, including ODBC in DAM provides several benefits for developers. A Windows developer who uses ODBC calls can expect Windows-based client applications to work unmodified in a Mac version. Developers also benefit by having a DAL driver on Windows PCs. By using ODBC, a developer gets access to the DAL-supported host databases simply by loading a single DAL driver on the client PC. And Mac developers who use the ODBC API in DAM will be able to port their applications to Windows more easily.

PowerPC and Macintosh

Perhaps the most difficult but also the smoothest transition for Apple will be from the Motorola 680x0 processor to PowerPC. This move was necessary for Apple to keep pace with the power curve set by Intel and by RISC architectures. While not explicitly an open systems move, it has stimulated many of Apple's efforts in dealing with other vendors and in promulgating its technology throughout the industry.

Apple is expected to announce its first systems based on the PowerPC RISC microprocessor in the first quarter of 1994. These systems will be based on the 601 version of the processor running at approximately 66 MHz, and they may provide performance that exceeds that of the Intel Pentium processor due to the superscalar design of the PowerPC processor (multiple instructions executed during each clock cycle). The 601 chip was largely designed by IBM, unlike the jointly designed follow-on processors, the 603, 604, and 620. It was intended to be a low-cost, high-performance chip. (Unlike the Power and Power2 processors used by IBM in the RS/6000 systems, the PowerPC is a single-chip design.) We expect Apple's PowerPC systems' pricing to begin under $3,000, which is very competitive in today's market. The systems will have all of the familiar hardware features of a Macintosh, and they include features like built-in SCSI, AppleTalk, Apple Desktop Bus, and the Macintosh ROMs.

Although it will be using the same processor as the recently announced IBM PowerPC RS/6000, the Apple machines will clearly be Macintoshes.

DAL and ODBC Architecture

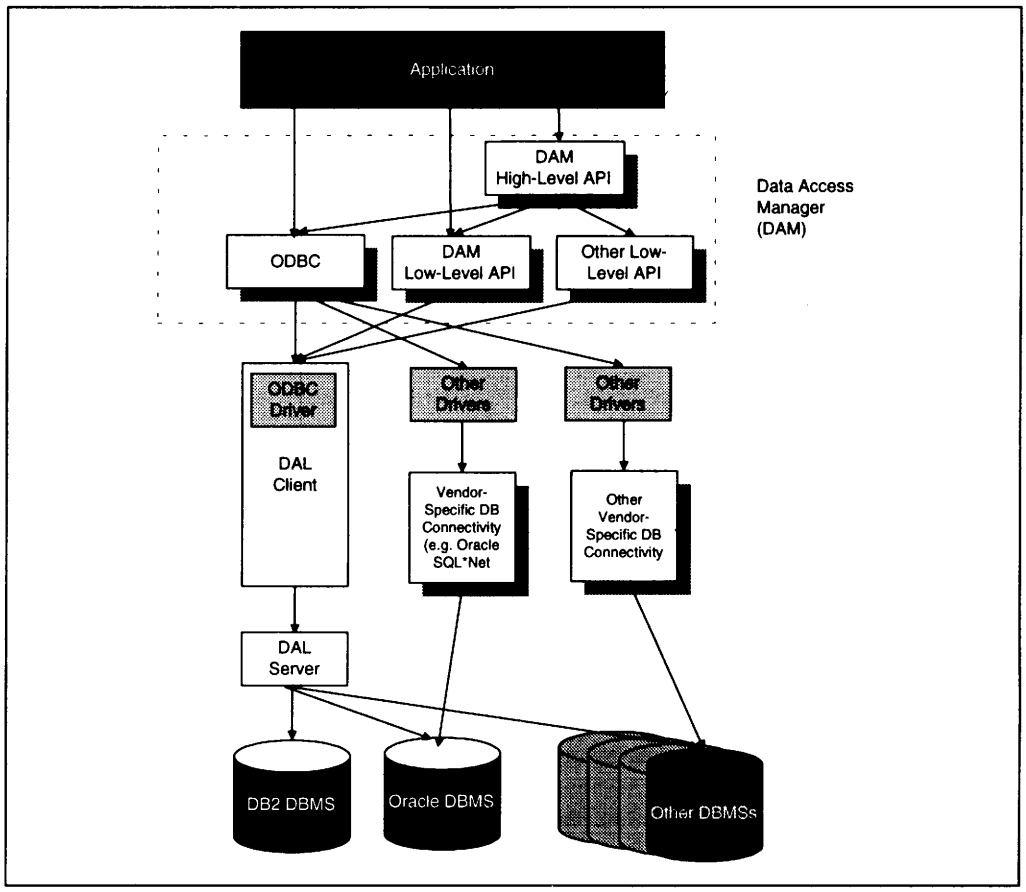

Illustration 3. DAL and ODBC combined provide Macintosh application developers with a reasonably complete data access solution that uses a client/server architecture. It also allows Windows applications to use the same DBMS ODBC drivers on the host database.

Follow-on Processors Bring Added Power

The 603 chip is a smaller version of the 601 and is due later in 1994. It is designed to consume much less power and to be suitable for portable computers and other "green" designs. It will have approximately the same performance as the 601 processor. The next generation of PowerPC Macintoshes will be based on the 604 chip. This processor will offer at least twice the processing power of the 601 and is expected to be available in volume in 1995. Things get really interesting in 1996, when the 620 PowerPC will be available, offering up to six times the speed of the 601. This processor will support some very interesting capabilities in software, including continuous, speaker-independent speech recognition and cursive-handwriting recognition.

The new processors will allow Apple to remain competitive in performance and pricing with Intel-based systems while potentially offering better price/performance than MIPS-based and Alpha AXP-based Windows NT systems.

Macintosh Operating System Strategy

Apple will offer two operating system choices on the PowerPC Macs. System 7.x, the current Macintosh OS, is being ported natively to the new systems. It will be a full implementation of the current system, and applications need only be recompiled to run on the new processor.

PowerOpen, the Unix environment based on IBM AIX, will also be offered on Apple's servers, if not on desktops as well.

Macintosh Apple is making available to IBM and other PowerPC licensees technology that allows Macintosh applications to run on top of the PowerOpen Environment. (The PowerOpen Environment, shown in Illustration 4, consists of an Application Binary Interface, or ABI, based on IBM's port of its AIX Unix operating system, Version 3.2.5, to the PowerPC architecture. PowerOpen is essentially AIX; however, other vendors would not want to call their versions by IBM's brand name; hence, the PowerOpen designation.) Apple's software, shown in Illustration 4, will be called Macintosh Application Services, and it will enable Macintosh applications developed for the Motorola architecture as well as applications written for the PowerPC Macintosh to run on PowerOpen Environment systems. Applications written for 680x0 Macintoshes will be supported by a 68040 emulator, which interprets instructions for the PowerOpen platform. The Macintosh toolbox is being ported directly to the PowerPC, and, since Macintosh applications spend up to 90 percent of their time making toolbox calls, the performance penalty incurred for the 10 percent or so of the time that emulation has to be used should be minimal.

PowerOpen Environment and Macintosh Application Services

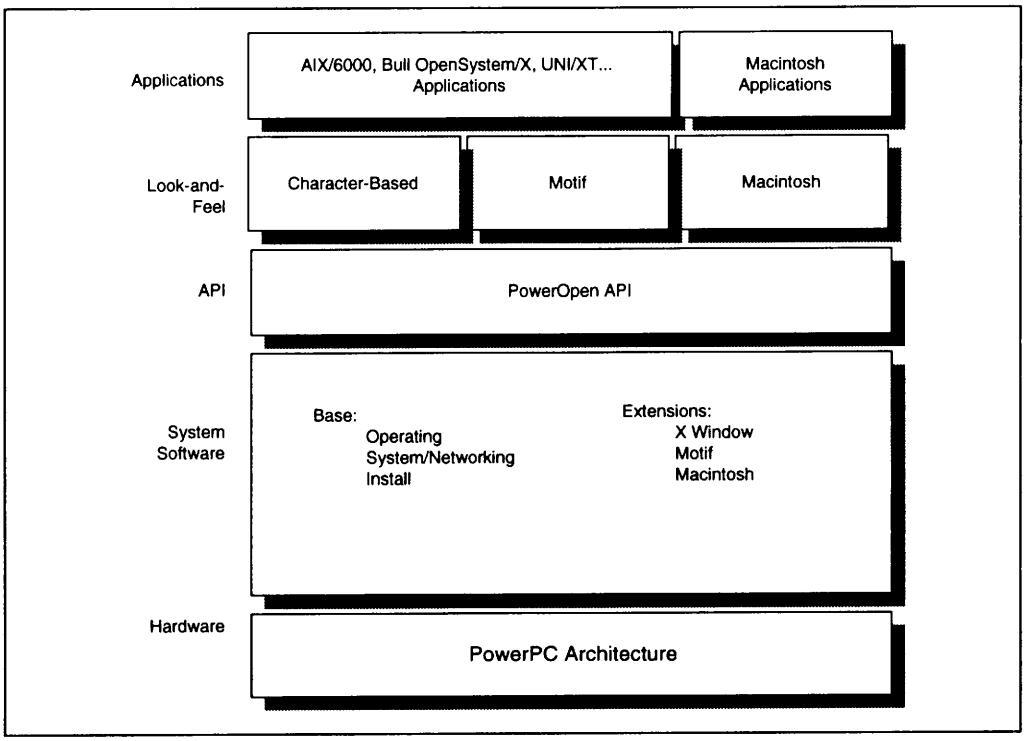

Illustration 4. PowerOpen Environment builds the application environment for both Unix and Macintosh applications on top of base-level Unix services.

Macintosh Application Services, shown in Illustration 5, will run as a single session within an X Window. The interface consists of a System 7 Finder, and all operations and interface widgets are the usual Macintosh ones. Cut-and-paste within the Macintosh window works normally, and cut-and-paste between the Macintosh and X Window applications is supported within the limitations of the X Window clipboard.

Apple is expected to bring Macintosh Application Services to other Unix platforms, such as HP UX and Solaris, in addition to PowerOpen.

Developer Options for System 7

Under the ported version of the System 7 Macintosh operating system, Macintosh developers have three options for running their applications: emulation, translation, or porting. Macintosh application binaries will run in emulation mode on the PowerPC Macintosh. The environment on PowerPC includes a 68040 software emulator. On the 601 PowerPC processor, this will allow applications to run at speeds comparable to a 68040-based Macintosh. The second option is for developers to perform a binary translation of their Motorola 68K code to PowerPC code. (See *Open Information Systems*, Vol. 7, No. 11 for more information on Echo Logic's FlashPort translation technology.) In addition, third-party tools vendors are providing assembly-language translators as well as translators to convert Pascal and ObjectPascal to C and C++. (Pascal and ObjectPascal are not supported on PowerPC.) This option provides native PowerPC performance. Of course, developers have the option of doing a native port of the application, in which case they get additional performance benefits as well as access to new capabilities enabled by the additional performance of PowerPC, such as speech recognition, text-to-speech, sound, telephony, video, 3-D rendering and animation, and complex modeling and analysis.

Application Services Architecture

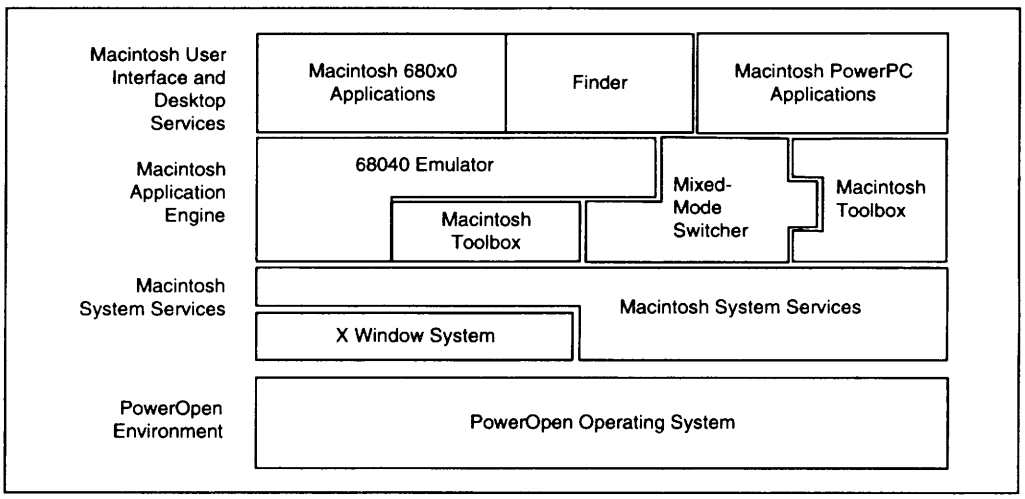

Illustration 5. Macintosh Application Services Architecture provides a complete Macintosh environment within an X Window on PowerOpen-based systems. Macintosh 680x0 applications are serviced by the 68040 emulator and the Macintosh Toolbox. Macintosh PowerPC applications are serviced directly by a Macintosh Toolbox ported to the PowerOpen operating system.

The Newton Model for Licensing

Licensing of technology to other vendors is rapidly becoming an accepted practice at Apple, making it more likely that the company may ultimately license the Macintosh OS to other vendors, not just the Mac Application Services. The Newton Model of licensing will be key for Apple to compete successfully against Microsoft. Apple licenses the Newton operating system to Sharp Electronics, Motorola, Cirrus Logic, Siemens Private Communications Systems Group, and Kyushu Matsushita Electric. Apple should be able to take the same approach with the Macintosh OS that it has with the Newton to attract other hardware platform vendors and greatly expand the market for Macintosh applications.

OpenDoc—Supporting Object Interoperability

OpenDoc is Apple's open, cross-platform architecture for compound documents that include text, graphics, QuickTime (video), and other kinds of data. The architecture is open, and the source code is available to allow vendors to implement in their own products. It is Apple's answer to Microsoft's Object Linking and Embedding (OLE) 2.0, replacing large single-

purpose applications with collections of objects, called "parts," all held in a container document. It is also Apple's intention to have OpenDoc ported to Windows and interoperate with OLE 2.0. WordPerfect Corporation (Orem, Utah), Novell, IBM, and Borland have endorsed OpenDoc, and they plan to deliver implementations for Windows in mid-1994. IBM will incorporate OpenDoc in future releases of OS/2 and will use the IBM System Object Model (SOM) to package and execute parts of a compound document. Novell will support OpenDoc in AppWare, and both Novell and WordPerfect will deliver OpenDoc capabilities on Windows. Borland will make its applications into OpenDoc containers and is considering adapting its development tools to product OpenDoc part editors. Apple is also working with Taligent (Cupertino, California) to ensure interoperability between the Taligent environment and OpenDoc.

OpenDoc supports key compound document capabilities:

- Creation of compound documents: Current and future data types can be placed into an OpenDoc document using cut-and-paste or drag-and-drop methods.
- Editing in place: Any type of content is edited in place in a document without having to start the creating application in a separate window.
- Document management: OpenDoc tracks document revision history and maintains basic version control.
- Cross-platform support: Interoperability among platforms is supported, allowing users to share and interact with complex documents, no matter which platform the user is working on or on which platform a document or part of a document was created.
- Preferred editor: The user specifies a preferred editor for each different data type and uses that editor, no matter what type of document is being edited. A preferred word processor can edit text in a spreadsheet, for example.
- Access to object services: OpenDoc will be compliant with Common Object Request Broker (CORBA) specifications, allowing users to access a range of services across CORBA-compliant object systems.

OpenDoc Architecture

DOCUMENTS. Documents in OpenDoc aren't tied to a creating application. A document is no longer a single block of content but, instead, is comprised of smaller blocks of content, or parts.

PARTS. The basic building blocks of documents in OpenDoc are *parts*. Every part has a type and contains data: A text part contains characters, a spreadsheet part contains cells, a video part contains digitized video. The type of data that each part contains is defined by the developer and is known as the part's intrinsic content.

A part may contain other parts. Documents build their own hierarchical structure, and each document has one root part into which all other parts are embedded. The ability to contain other parts is determined by the developer; however, if a part can contain at least one other type of part, it can contain all types of parts.

PART EDITORS. Individual programs that manipulate and display a specific type of content are called *part editors*. They may be components used to build an entire solution or solely to build documents. Part editors can best be thought of as large-grained objects that will allow developers to create new applications by assembling part editors.

FRAMES. Boundaries between parts are called *frames*. Each part of a document has its own content model of objects and operations that is presented to the user. The frame demarcates changes between parts within its model. Frames are areas of the display that represent a part.

PART HANDLERS. When a part is displayed or edited, a *part handler* performs those tasks. Part handlers display the part for viewing, editing, and printing; edit the part by accepting events and changing the state of the part; and manage persistent and runtime storage of the part and reading from persistent storage into memory. Part handlers may be full function or may only provide viewing capabilities. Part handlers are dynamically linked to a document at runtime.

STORAGE. Multipart documents require specialized persistent storage capabilities. The mechanism in OpenDoc is based on Apple's own Bento standard. It assumes that the storage system gives each part its own data stream and that reliable references can be made from one stream to another, an assumption that allows parts to be integrated into a single document. (Bento is a multipart, hierarchical storage format, with a Bento Services index. It is analogous to the Microsoft OLE format. Apple licenses the source code for Bento to third parties to include in other operating systems.) Since OpenDoc is designed to support cross-platform capabilities, it must also support the ability to write multiple representations of a given part.

Calling between objects in the OpenDoc architecture will be done using IBM's System Object Model (SOM) as its object calling mechanism. SOM provides a binary standard for object interfaces that conforms to the CORBA standard for distributed object messaging. It also allows developers to work in a wide range of languages and lets parts call each other with no additional programming effort. SOM provides access to distributed services through its CORBA-compliant API.

Apple's objective is to make OpenDoc the accepted standard for compound document integration. The company is working with groups like X/Open, the Open Software Foundation (OSF), and the Object Management Group (OMG) to try to help OpenDoc become adopted as an open standard. It does not incorporate support for standards such as SGML, ODA, or ODIF directly, although parts can support these standards. Apple's format conversion technology could manage interchange between OpenDoc and standard formats, and nothing prevents standards from being supported by parts editors. For example, an SGML parts editor could be supplied by a third party. Although Apple is making source code available and is soliciting input from other vendors, it is not yet garnering enough support for OpenDoc to gain as much momentum as Microsoft's OLE 2.0. Apple and IBM hope that, by working closely together, they can change this situation. OpenDoc would seem ideally suited for inclusion in the COSE Common Desktop Environment specification, but there has been no indication whether or not this will happen.

Apple's Open Collaborative Environment (AOCE)

The Open Collaborative Environment (OCE), first announced in March 1992, provides a set of operating system-level facilities intended to support the creation of mail-enabled, collaborative, user-focused applications. Within AOCE, mail becomes a common transport for data of all types, including documents, forms, and video. AOCE is open in a more granular sense, enabling interoperability among applications on a single platform, and between an application on one platform and an application on another. It is initially being introduced only on the Macintosh, but it will find its way eventually to other platforms, such as Windows.

AOCE Clients and Servers

AOCE client software called PowerTalk modifies the Finder with a control panel and a number of extensions and templates. A desktop mailbox is added which contains all incoming and outgoing files. Also added to the Finder is a Catalog icon, which contains a hierarchical network browser and shared lists of network services. Also included in the environment are

encryption and digital signatures to provide data security and authentication. Encryption is managed by the AppleTalk Secure Data Stream Protocol (ASDSP), although actual data encryption uses the Data Encryption Standard (DES), and authentication is performed by the Authentication Manager. ASDSP is similar in architecture to Kerberos, but it is not Kerberos based.

Although AOCE is designed to operate in a peer-to-peer environment, there is also an AOCE server called the PowerShare Collaboration Server, which provides store-and-forward messaging, remote logon, and user authentication. The server also hosts various common communication services, e.g., gateways, that can be reached from users' mailboxes.

Network Directory Services

AOCE creates and manages network-wide directories. The Catalog icon contains these directories, and they can be shared by many network applications. In the past, each application had to maintain its own directory. However, Catalogs contain more than directory information. They provide access to a hierarchical directory of basically every entity that is significant in the network environment: users, shared disks, printers, and more. Moreover, Apple has published the network interfaces of its directory service, allowing PC LAN vendors and others to integrate their directory capabilities into OCE; at least one LAN vendor has a working prototype running already. To help third-party developers register their services and data with the OCE directory service, Apple has also implemented a Standard Directory toolbox which, like Standard Mail, can be easily linked into a Macintosh application.

The first implementation of AOCE is designed for Mac-only networks. However, Apple has indicated that it will port AOCE and related technologies to Windows. Apple has published the OCE mail APIs and is working with E-mail vendors to have gateways ready when OCE is delivered. In the meantime, AOCE-enabled applications will have to communicate with the rest of the world through back-end gateways, including gateways to X.400 mail systems, X.500 directories, and the Internet.

E-Mail Gateways in Progress

Although OCE's compound-document mail facility clearly benefits the Macintosh environment, its benefits for the mixed environment are not so clear. Apple recognizes that isolated, text-only mail systems have outlived their welcome. An enterprise can't even begin to be fully automated if it can't mail compound documents among all its systems. Thus, Apple is trying to foster E-mail interoperability with other systems, particularly with PCs. Indeed, a significant theme of the OCE initiative is interoperability.

Conclusions

Apple is pursuing a hybrid strategy of delivering proprietary systems on the desktop while embracing open systems as the basis for enterprise information systems. Its desktop strategy of providing proprietary Macintosh software on a variety of hardware platforms is not unlike Microsoft's strategy. The key differentiator is that Microsoft does not sell hardware and does not compete with the original equipment manufacturers (OEMs) that support its system software offerings. Getting those same OEMs to support Macintosh either natively or on PowerPC, knowing that Apple will be competing with them for each machine they sell, is going to be a negative incentive. However, there are a number of those OEMs who will take the position that each system sold that does not run a Microsoft operating system, whether theirs or Apple's, strikes a blow for freedom of choice. The only problem with this argument is that it represents a choice between two proprietary operating systems.

In a broader sense, Apple is laying the foundation for an effective strategy to compete with Microsoft on its own terms. It is providing basic, system-level services that are available to all application developers. Apple's value has been in making the operating system the focal point of where the hardware meets the software, allowing application developers to leverage the work of the hardware developers. In addition, it is making those same services available

Conclusions

throughout the industry to other vendors to include on their platforms in an interoperable fashion. It is on this point that Apple's approach dramatically diverges from both Microsoft's and Novell's. Microsoft's WOSA assumes a Windows client. Novell's AppWare assumes a NetWare or UnixWare application server. Apple makes no such assumptions, and it may, therefore, become the most open of all three vendors.

The success or failure of the PowerPC-based Macintosh is less dependent on the hardware that Apple delivers than it is on the success of its software strategy. Supporting existing applications on the ported version of the Macintosh operating system is expected. Making an environment available on other platforms for those same applications may not directly help sales of Apple's hardware, but it will make customers feel safer about choosing Macintosh applications and feel as if they have more choice. More importantly, it will expand the number of seats which any independent software vendor (ISV) counts when it makes decisions about porting priorities. Apple's strategy will help the company keep its place in the porting queue.

Finally, VITAL has the potential for establishing Apple's credibility far beyond the desktop. However, in order for that to happen, it must find the right strategic partners. Considering how little most vendors have to offer in the way of insights about how to implement large-scale client/server architectures, customers should be beating down Apple's door to get at VITAL. We will see who comes knocking and whom Apple lets in. ◐

Next month's *Open Information Systems* will address
Progress Version 7

For reprint information on articles appearing in this issue,
please contact Donald Baillargeon at (617) 742-5200, extension 117.

Open Systems: Analysis, Issues, & Opinions

FOCUS: VENDOR STRATEGIES

AppWare: Retooling the Distributed Application Marketplace?

Distributed applications haven't overwhelmed the marketplace—yet. The latest diagnosis and treatment plan for the problem can be seen in the recent AppWare strategy announcement from Novell Corporation (Provo, Utah). On the surface, the AppWare announcement signals Novell's move into the market for application development tools. A deeper look reveals that this move is in the context of Novell's commitment to expanding the network computing market, positioning NetWare for growth in that market, and stalling Microsoft Corporation's (Redmond, Washington) NT- and WOSA-centered push from the desktop to the server. Why is Novell pursuing application development tools? Simply, because growth in the low-end network computing services market has slowed, making increased adoption of network-based applications in the corporate market the key to growth for Novell. Thus, helping break the paralysis in distributed application development goes right to the heart of long-term growth for Novell.

Historically, the core value to users of NetWare has been in user-accessible network services, such as printing and filing. Novell wants to leverage its dominance in the low-end network services market into participation in the market for network-based, mission-critical applications for the corporate environment. Witness NetWare 4.0 and its global directory and authentication services, which directly target the corporate IS requirements for large-scale applications. However, directory and authentication services, unlike print and file services, are used by applications, not by users. All this leads to a three-part rub for Novell:

- Distributed applications that require the distributed services of NetWare have not arrived in numbers.
- Tools to build distributed applications are scarce.
- Novell's traditional users and channels can't address this problem without massive assistance.

The sluggish take-off of NetWare 4.0 testifies to this conundrum. With the AppWare announcement, Novell has committed to building the infrastructure required to populate the marketplace with distributed applications that will use NetWare application services. This undertaking is monumental in scope, and we believe it is unlikely to succeed in the near term.

Evolving Novell Strategy

It is important to understand that Novell views the networked services market in evolutionary terms, framed by the current and future needs of its customer base. Novell stays in close touch with its customer base and has engaged in formalized market research through a variety of means, including focus groups. In the past year, Novell's major-account focus groups provided four key insights that have guided the AppWare strategy: These customers want to build network-based applications using NetWare, they suffer a painful backlog of application development projects, they are pursuing multiplatform strategies, and they see the multiple platforms and the paucity of current development tools as an impediment to developing and deploying network-based applications.

Novell's Buying Spree. In the past year, Novell has made a series of moves that appear to signal a shift away from the narrowly focused "NetWare everywhere" strategy. Novell's solid position at the top of the network operating system (NOS) world in recent years has spun off the cash to fund numerous investments and strategic relationships, including the acquisitions of Digital Research, Unix System Laboratories (Summit, New Jersey), Serius Corporation (Salt Lake City, Utah), and Software Transformations Incorporated (STI), and taking a 20 percent stake in HyperDesk (Westboro, Massachusetts). As obscure as it may seem, the common thread running through these events continues to be "NetWare everywhere." The difference is that NetWare is no longer just print and file services. It is evolving from an optimized print and file server into a high-performance, general purpose services engine for large-scale networked environments. Novell has outlined plans to offer NetWare Loadable Modules (NLMs), which would provide imaging services, telephony services, video services, software distribution services, network

management services, and more. This movement to higher-value services makes sense, but it will not be easy because the demand for such services remains underdeveloped.

Although Novell is outlining an expanded role for NetWare, the company still plans to focus on the systems services side of the business. Unlike Microsoft, Novell is exceedingly clear about remaining focused on providing not applications, but application *services*. For example, Novell wants to provide imaging services for application developers to use in their imaging applications but not end-user imaging applications. In this respect, Novell is addressing the applications market in the same way the traditional system vendors have, while Microsoft has defied conventional wisdom and, with some notable exceptions, competed successfully in the market for end-user applications while simultaneously providing system services for application developers.

Infrastructure, Tools, and Heterogeneity

Novell views the dearth of distributed applications as symptomatic of three major problems: Lack of infrastructure of standardized system services (including services that might generally be considered middleware services), lack of tools that simplify distributed application development, and the complexity of developing and implementing applications across diverse platforms. (See Illustration 1.) Novell cites Lotus Notes as an example of a development effort where as much as 60 percent was devoted to building system-level infrastructure (such as distributed global directory services and distributed database services). Therefore, by enhancing the breadth of infrastructure services offered by NetWare, additional network applications should be drawn into the market. This strategy goes straight at Microsoft and the Windows Open Services Architecture (WOSA), and it just might frame the contention between the two companies over time.

Incorporating Middleware into the NOS. Why couldn't the infrastructure problem be solved with a combination of third-party middleware products and Novell offerings? After all, STI, Serius, and HyperDesk were already addressing these issues. Novell believes that, even if middleware could solve a set of point problems, it wouldn't solve the overall problem of removing technical complexity and reducing business risk for developers. Application developers would still have to integrate pieces of middleware and fill in the missing pieces of distributed application infrastructure. Developers would also have to take a series of business risks on small, undercapitalized companies. And even if those issues could be addressed, the resulting distributed applications wouldn't necessarily be targeted at NetWare. Therefore, integrating middleware such as the STI Universal Component System (UCS), HyperDesk Distributed Object Management System (HD-DOMS), and the Serius programming environment into the NetWare environment should represent a key benefit to developers—and to Novell. The simplicity of the Microsoft alternative and the fact that the forces of the marketplace are unmistakably tilting toward Microsoft have motivated Novell to actively address the sticking points in the distributed application development process and to offer a framework based around its key asset—NetWare.

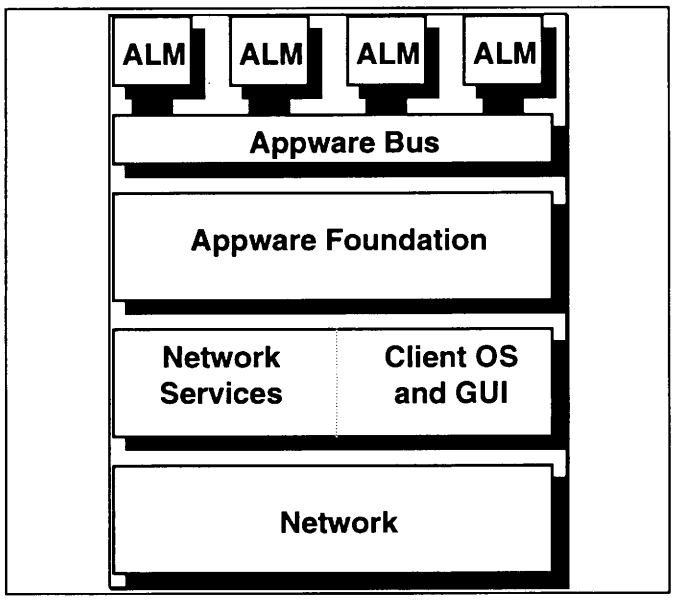

Illustration 1. The AppWare architecture insulates the developer from the uniqueness of platforms, networks, and graphical user interfaces.

The AppWare Product Set

AppWare consists of five major elements: the AppWare Foundation, the AppWare Bus, AppWare Loadable Modules (ALMs), the ALM Construction Kit, and the Visual AppBuilder. The AppWare Foundation is derived from the Universal Component System from STI, while the rest of the pieces are what were the Serius Programmer product from Serius Corporation. Novell is porting all of the pieces sourced from Serius to the AppWare Foundation set in order to have a consistent cross-platform capability. (See Illustration 2.)

AppWare Foundation. The AppWare Foundation provides a consistent application developer API to operating system services across the supported platforms in order to enable portability of ALMs and ALM tools. While strategic developers were interested in the technology, UCS was considered incomplete and STI was too small before Novell acquired it to merit a strategic commitment. UCS originally shipped Macintosh and Windows libraries, and it recently ported to Unix. The AppWare Foundation currently supports the following Unix versions: HP-UX, SunOS, Solaris, Ultrix, and UnixWare, with AIX and OS/2 on the drawing board for 1994. (See Illustration 3.)

AppWare Bus. The AppWare Bus provides process-level interapplication communication between the ALMs through a protocol called the Object Interaction Protocol. The AppWare Bus interface is implemented consistently across ALMs, so, theoretically, ALMs can interoperate to form a virtual application regardless of the source of the ALM. Novell sees the AppWare software bus becoming comparable to Hewlett-Packard Company's (HP, Palo Alto, California) SoftBench or Sun Microsystems Computer Corporation's (Mountain View, California) ToolTalk interapplication mechanisms.

AppWare Loadable Modules (ALMs). ALMs are large-grained objects that can be combined to form a user application. For example, a database-access ALM or a document-indexing ALM might be combined with other special purpose ALMs to form a usable application. Novell compares ALMs to C++ classes to illustrate that many more C++ classes than ALMs would be required to build a full application. However, ALMs could be developed in C or C++. Serius currently provides database access, communications, and graphical interface ALMs. Novell plans to implement NetWare file, print, and message-handling services as ALMs.

ALM Construction Kit. Development of ALMs is done with the third-generation language tools and utilities provided in the ALM Construction Kit. Novell indicates that efforts will be put in motion to eventually provide robust third-generation interactive development environments through relationships with companies such as Borland International Corporation (Scotts Valley, California) or CenterLine Software (Cambridge, Massachusetts).

Visual AppBuilder. The Visual AppBuilder is a visual programming environment where ALMs are linked "without coding" to create user applications. In fact, some of the work can get done by "connecting the dots," but a considerable amount of "handwork" is still required to build a production-quality application.

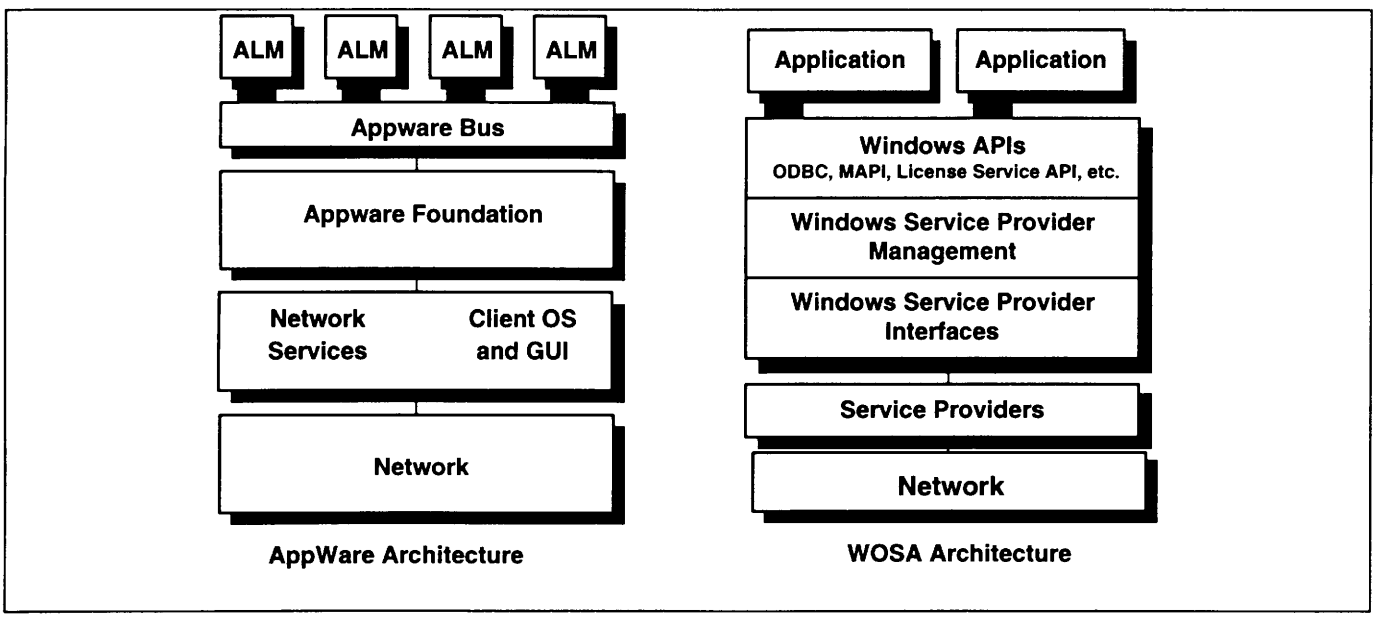

Illustration 2. This illustration contrasts the AppWare Architecture with the WOSA Architecture. AppWare consists of technology that was acquired from Software Transformations and Serius and that is undergoing integration and porting over the coming year. It consists of a set of services and APIs supporting the development of applications. WOSA service providers are generally from third parties, while Microsoft supplies only service provider management and service provider and application interfaces. AppWare is oriented toward integrating application modules; WOSA is oriented toward providing access to back-end services.

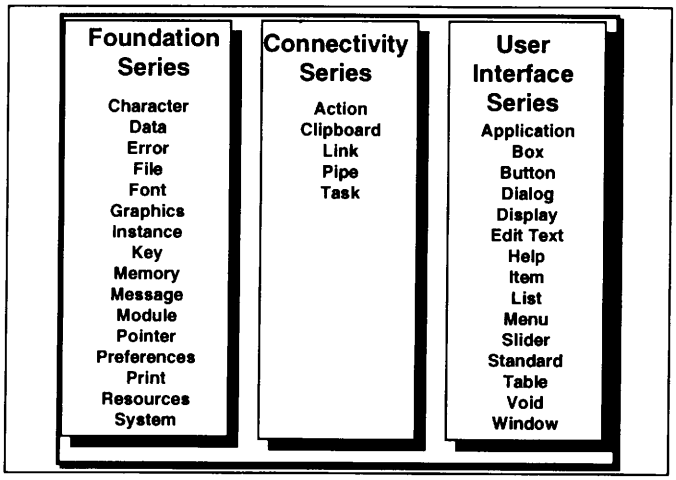

Illustration 3. The AppWare Foundation consists of object classes that are segmented into functions associated with operating systems, networks, and GUIs to provide consistent systems function across platforms.

HyperDesk HD-DOMS Role

The HyperDesk HD-DOMS technology has an indirect relationship to AppWare. Novell plans to port HD-DOMS to the AppWare Foundation to achieve platform independence. Separately, Novell plans to implement NetWare system services through the HD-DOMS in order to provide a platform-independent means for applications to access system services network-wide as objects, through the services of the HD-DOMS Object Request Broker (ORB). Services will be implemented with a registration interface developed with the Interface Definition Language (IDL). Over time, the STI development shop will surface the HD-DOMS service registration API through the AppWare Foundation set, and will manage CORBA compliance through the ORB abstraction in the AppWare Foundation. However, questions remain as to just how firm these plans are.

AppWare and Unix

AppWare is planned to be ported to Unix because Novell continues to treat UnixWare as the future of both Unix and NetWare. UnixWare is being positioned as a vertical application engine that uses NetWare services for many system functions such as file services, print services, and future services as discussed above. UnixWare appears to be last on the list of platforms to which the full AppWare environment will be ported.

The AppWare Distribution Dilemma

In its current state of development, AppWare does not yet have much synergy with the Novell reseller channel. Like the AppWare product plan, the AppWare distribution plan has an extended implementation schedule. Following the opening salvo of the AppWare announcement, Novell is actively working with the Novell "developer channel," including strategic vendors such as WordPerfect and Lotus Development, to obtain the vital bellwether endorsements and proofs-of-concept. Novell points to as many as 10,000 independent software vendors (ISVs) representing 40,000 developer seats in this developer channel.

Novell has much work ahead to get AppWare off the ground. The company must focus its near-term efforts almost entirely in its developer channel to build acceptance of the AppWare strategy. Then Novell will have to insinuate ALM projects into the developers' project priorities and onto their product calendars and marketing plans. When and if developers respond and begin developing ALMs, Novell will still have to apply significant force to populate the market with ALMs. For example, Novell will have to shoulder much of the burden of verifying the interchangeability of the ALMs through an AppWare equivalent to the NetWare "Yes, it runs with NetWare" certification and testing program. Finally, at the other end of a long process, a new business opportunity could materialize for the hungry Novell reseller channel. The theory is that the resellers will train their Certified NetWare Engineers (CNEs) to become "connect the blocks" AppWare application builders (Certified AppWare Engineers?), custom-crafting business applications from catalogues of standard ALMs with the Visual AppBuilder. However, this distribution model will require Novell to execute a large-scale make-over of the network integrator channel, which currently has low application development skills. Such a make-over is an unlikely eventuality.

Implications of AppWare

Novell believes that NetWare users want to be able to construct applications from standard application components and run the applications on NetWare. In an attempt to meet this need, Novell has crafted the AppWare strategy, which boils down to nothing less than an attempt to drive sweeping change through the ISV community and the Novell channel. The potential impact of AppWare on ISVs goes beyond a new chip architecture, system platform, operating system, or interapplication mechanism. Although ISVs have modularized application structures and isolated platform dependencies, the ALM concept necessitates a far greater response than simple adaptation or porting of existing applications. And, however great the impact on the technical side, there are the equally important business and marketing implications.

It is true that distributed application development suffers from all of the problems cited by Novell in explaining the AppWare strategy. However, the fundamental strategic question that will be answered over the coming months and years is: How far will Novell go to fund and sustain the AppWare effort to change the way applications are conceived, built, marketed, and delivered to the user? In the current context of declining margins and increased competition, ISVs have to put resources to work on the projects that will yield maximum license sales. We are skeptical that AppWare will make the priority cut.

Competing with Microsoft

The bottom line with respect to archrival Microsoft is that, if Novell sits back and lets the Visual Basic/Visual C++/Access/SQL Server/OLE/Windows NT application development machine operate unchecked, two predictable results will occur: The network-based applications that emerge in the market will not be designed to exploit NetWare services, and Novell's crown jewel will become a cash cow. Novell anticipates that ISVs will perceive AppWare as yet another API, and in fact, they do. In order to counteract that effect, Novell will position AppWare against the Microsoft offerings as the way for developers to avoid getting locked into a Windows-centric view. For example, Novell plans to provide Object Linking and Embedding (OLE) interoperability as part of the AppWare Foundation, along with other object-linking and interapplication mechanisms. Novell hopes to create a cross-platform alternative to Microsoft WOSA that will address the reluctance of ISVs to commit to Microsoft. This positioning will help. However, Novell is being optimistic in even targeting the ISV priority-two slot (after the Win32 API). Again, while ISVs will be interested in a business proposition related to an installed base as large as Novell's, this strategy greatly underestimates the pragmatism afoot in the ISV community. Novell is approaching the ISV community with a new paradigm that is likely to receive only lip service, particularly until some concrete results are achieved by Novell.

Conclusions

The most striking aspect of AppWare is the scope of the undertaking and the degree of substantive change required by the industry and by users if the effort is to truly succeed. The most directly competitive product to AppWare, the Galaxy Application Environment from Visix Software (Reston, Virginia), has been delivering for over a year what AppWare is targeting for mid-1994, and Galaxy Release 2.0 is expected imminently. Galaxy users, both in-house corporate developers and ISVs, have already completed the first round of cross-platform applications. Developers are embracing Galaxy because it represents a fundamental technology leap while retaining continuity with the way applications are developed and brought to market, and, unlike AppWare, it does not presuppose a preference for a particular network. While AppWare is definitely something to watch, developers and users should hold Novell to the task of investing in all dimensions of the AppWare proposal, including broad market development. Today, AppWare offers nothing more than the sum of its individual parts, so developers that could meet their objectives with the Serius Programmer or the Universal Component System could work productively with AppWare now. Those who need something more will have to wait until spring '94 for the next major installment of AppWare deliverables.

We think most developers should stay on the sidelines regarding AppWare to see if Novell has the commitment to effectively integrate all of the pieces and then to market AppWare both to ISVs and to end-users. If the slow uptake on NetWare 4.0 is any indication, the chances of immediate success are slim. We think Novell has an outside chance of making AppWare successful.

— S. Dolberg

FOCUS: INDUSTRY

Unix Vendors Agree on Common Kernel Specification

The Unix system industry, finding common ground in a mutual threat—Microsoft Corporation (Redmond, Washington) and Windows NT—has announced that it is well on the way to developing a common kernel level and a Base application Programming interface (API) for Unix-based operating systems. Ironically, the announcement in New York on September 1 was made in the same auditorium where the formation of the Open Software Foundation (OSF) was announced in May 1988, an event that marked the beginning of five years of divisiveness in the Unix industry. The issues of 1988 have been replaced by today's reality: the challenge to Unix posed by Windows NT. The industry has given up its battles over which kernel is the best or true Unix and which APIs are the correct APIs. It has been forced to accept the need for one common API to meet the challenge posed by Microsoft.

Only by unifying around one API, the Unix Industry believes, can it provide Unix with the consistency

necessary to counter customers' concerns that buying one vendor's Unix locks them in just as much as if they choose to buy Windows NT from Microsoft. Achieving this objective outweighs any advantage gained by individual Unix vendors in maintaining unique kernel APIs.

Proposed Specification a Starting Point

The announcement in New York was of the first public draft of a proposed specification containing over 1,100 kernel-level and Base APIs for Unix. These APIs complement the Common Desktop Environment announced by the Common Open Software Environment (COSE) group last March. (See *Open Information Systems,* Vol. 8, No. 3, March 1993.) The specification contains file system calls, math library routines, signals, sockets, TCP/IP, I/O, curses (a "standard" terminal I/O library), internationalization, and kernel libraries. In addition, the specification includes headers and commands.

The Unix API specification has been made available to the industry for comment, and it will be submitted to X/Open toward the end of 1993 for consideration through X/Open's Fast Track process to become an extension or follow-on to XPG4. The X/Open Fast Track process could result in a new XPG specification as early as the end of 1994.

It is important to emphasize that what was announced was not an agreement to standardize on a single Unix or on a single kernel. Even after the specification is accepted, there will be many different technological implementations from different software suppliers and system vendors which all support the specification. There are likely to be as many versions of Unix as there are Unix vendors. But determining the extent to which a product is "standard" will now become much easier: The criterion will be whether or not it supports the full API. The accuracy of a product's claims to be standard Unix will be determined by conformance-testing, not genealogy.

Deriving the Specification

The foundation for the Common API specification is the XPG4 base-level specification, which itself contains many standard APIs, such as POSIX 1003.1. The common Unix specification is a superset of the XPG4 specification and extends it in areas where widely supported specifications exist. The Common API also contains the System V Interface Definition 3 Level 1 (SVID 3) and the Open Software Foundation's Application Environment Specification (AES). Because AES conforms to SVID 2, there is a lot of overlap between the two. Additional APIs were included based on an analysis of APIs used by 50 market-leading Unix applications. SunSoft Company (Mountain View, California) analyzed even more applications and found that the 50 used were statistically representative of a sample of 4,000. Binaries from market-leading Unix applications were tested for API usage on leading hardware platforms.

The proposed specification covers nearly 90 percent of API usage by the applications tested. Usage analysis contributed a large number of APIs outside the XPG, SVID, and AES, even though only those APIs with significant usage were included. Code investigation revealed that the areas of most significant API usage included additional math routines, BSD 4.3 memory routines and Reno sockets, TCP/IP, and file system symbolic links capabilities. The proposed specification contains 926 system interfaces, 70 header files, and 174 commands. Just over half the APIs come from the XPG4, as do all of the 174 commands. SVID 3 contributed over 30 percent of the APIs, mostly for improvements to curses routines for color and internationalization. (See table.)

The result of the analysis and preliminary API specification was then validated against the API usage of 10 leading Unix applications. The specification provided an average of 98 percent of the base API coverage of those applications. This represents a significant improvement over the 60 to 80 percent coverage provided by the XPG4. Clearly, this API, particularly when combined with the Common Desktop Environment, gives developers writing for Unix a far more complete set of APIs for writing portable applications than they have ever had in the past. Many of the areas that are not defined are those where specific machine dependencies are involved. APIs dealing with those layers probably should be collected into one group and labeled something like the Hardware Abstraction Layer (HAL) in Windows NT to make clear where the source of the difference lies.

Several areas are not included in the specification, including threads, system management, and security. The reason for the exclusion is primarily that standardization work is currently underway within the IEEE and POSIX. When those efforts are completed, their standards will be incorporated into the specification.

Vendor Response

Over 75 Unix vendors expressed support for the specification at the announcement, and others have been falling into line since. On the one hand, since the specification is headed toward X/Open-standard status, any open system vendor essentially has to commit to it or risk being labeled a renegade. On the other hand, the majority of operating systems will require relatively little work in order to achieve compliance with the specification, since most already support the vast majority of the APIs. For example, Santa Cruz Operations (SCO, Santa Cruz, California) expects work on 30 APIs to take about six person-months. SunSoft expects that even less work will be required for Solaris to conform. For users, this represents a particularly frustrating aspect about the announcement. Most are wondering, "If complying with this specification is so trivial, why has it taken so long?"

The answer is that supporting any given set of APIs has always been relatively easy. Getting the various interested parties within the Unix industry to agree on which ones should be standard has been difficult. What has changed today is that many Unix application developers are anticipating higher volumes for Windows NT than for any single version of Unix and perhaps than for all the different versions combined. As a result, traditional Unix independent software vendors (ISVs) are beginning to consider making the Microsoft operating system their top priority for development. Individual Unix vendors are now threatened with falling further back on ISV porting priority lists, and Unix in general is threatened with falling behind Windows NT in application availability. In order to prevent this, developers have to be shown a strong business case for keeping Unix as a high development priority. Reducing the cost of developing for a multitude of vendors' platforms is one benefit that the specification will provide. Simplifying support, training, and documentation are others. Reducing the cost of code maintenance is also a factor.

Sources of API Specifications for Common Unix Specification*

Functional Category	Spec Total	XPG4	SVID-3	AES	Use-Based
Memory	19	11	12	12	17
Curses	324	114	324	0	27
clib	114	92	98	81	65
Math	64	43	52	43	24
Internationalization	62	60	12	11	9
proc	61	32	43	37	36
File System	44	28	40	35	35
Standard I/O	41	40	41	41	33
Signals	25	12	16	12	16
dev	20	12	18	13	6
Sockets	19	0	0	0	15
All Other	113	22	60	15	16
Subtotal	926	479	736	322	299
Headers	70				
Commands & Utilities	174				
Total	1170				

* Rows do not total because some APIs are shared across different sources.

Vendors will continue to support all their existing APIs on their platforms so that today's applications will have no problems running in the future. Support for APIs in the specification which are not part of their current platform will be added either to future releases or simply provided as maintenance releases. In addition, as long as there are missing pieces in the specification, such as threads or security, vendors will still be offering unique APIs that ISVs will have to support. There is nothing to prevent vendors from offering alternatives to standard APIs which provide extended functionality. A vendor might also have an API to support a special feature of its platform of which an ISV might want to take advantage. To the extent an application uses those additional or different APIs, functionality is gained at the price of portability.

Do ISVs Really Benefit?

ISVs can choose to leave their applications untouched or to modify their applications to support the new specification. If they support the new specification, their code will become more portable, and the cost of maintaining code for different platforms will decrease. However, certain applications may need to take advantage of certain platform-specific extensions beyond the specification for performance, functionality, or compatibility reasons, so universal portability will not arise out of this effort.

To the extent that required functionality is provided by the API specification, ISVs will choose to stay within it. It is most likely that ISVs that have to wring the last bit of performance out of a platform will use platform-specific APIs. Applications which, like database management, often use a platform in nonstandard ways are good candidates for a sacrifice of portability.

Users Gain, but Face Decisions

The industry has been after users to make purchase decisions based on standards and high level interfaces for a number of years. The truth of the matter is that users realized that the existing formal and de facto standards were inadequate for really freeing them from making product-based decisions. Each vendor's Unix varied from the others', making portability an issue and limiting the availability of software on certain platforms. Each vendor's Unix differed enough in system administration and management to lock customers into one variant, almost as much as if they were buying a proprietary operating system.

The combination of the Open Desktop Environment and now the Common Unix API gives customers a much firmer base of standardization to work from in selecting specific products. (See illustration.) Software availability should improve as ISVs take advantage of the portability offered by the new specification.

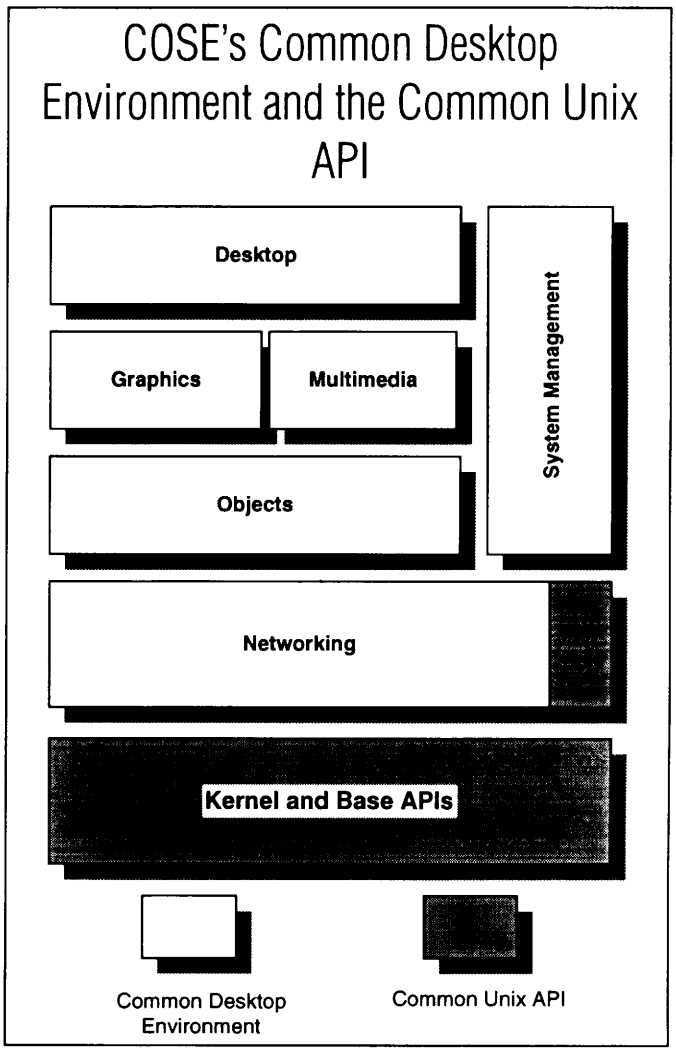

Illustration. The Common Unix API specification combined with the COSE Open Desktop Environment specification yields a complete API for developers of all types of applications, from end-user to system management applications.

This will not happen overnight. It will take over a year for the X/Open specification to be complete. Conformance tests will emerge shortly thereafter, and platforms will have to be brought into compliance, tested, and certified. Then ISVs will have to revise their codes to conform. It will easily take two years before users see tangible benefits; however, intangible benefits in the form of reduced confusion and uncertainty are already accruing.

Impact on Windows NT

The concept that one specification can have many implementations is hard for Microsoft to swallow. In the past, it has been hard for the Unix industry to swallow as well. But now Microsoft will have competition from a large segment of the industry based on a single specification with many different implementations. The Microsoft approach to consistency and portability is to have one technology both define a specification and implement it. The Win32 API for Windows NT is a specification and a technology. It is not designed to have multiple implementations. The Win16 API for Windows 3.0 was the same, but many clever programmers developed alternative implementations in products ranging from Wabi from SunSelect Company (Billerica, Massachusetts) to Wind/U from Bristol Technology (Ridgefield, Connecticut).

What is shaping up is not necessarily a battle between Win32 and its future extensions versus the Common Unix API and its future extensions. It is a battle between products implementing APIs. It may mean that some vendor supports Win32 and the Common Unix API, or that Microsoft or a partner supports the Common Unix API on Windows NT. With the competitive positioning of Windows NT against Unix, this is only likely to occur if compliance with the new XPG standard becomes a requirement for government procurement. We shall have to see what rolls out over the next few years.

How Do Consortia Roles Change?

Now that USL and OSF are committed to implementing a common specification, the role of OSF in the industry becomes clearer, while the role of Unix International (UI) becomes more obscure. Like USL, OSF will be a provider of operating system technology that implements the common specification for those companies that choose to buy rather than to make their kernel and that prefer the technology direction and licensing of OSF/1 over the direction and licensing of SVR4. Companies delivering OSF/1-based operating systems, such as Digital Equipment Corporation (Maynard, Massachusetts), Hitachi (Brisbane, California), Kendall Square Research (Waltham, Massachusetts), and others, now look less like risk-takers because the source of their underlying technology will be a non-issue. USL will bring SVR4 code into specification compliance rather easily. Vendors, such as Data General Corporation (Westboro, Massachusetts), that deliver SVR4-compatible operating systems using mostly their own kernel codes will bring their systems into compliance themselves.

UI becomes even more of an SVR4 user group, even though most of its members are original equipment manufacturers (OEMs). It becomes a vehicle for a unified voice to address concerns and requirements to Novell. However, those concerns are now issues between supplier and customer and have less of an industry-wide focus.

Changing the Value of the SVR4 Brand

When the specification is complete and finalized as an X/Open specification, it will be the X/Open specification that has significance in the marketplace, not the Unix trademark. Unix and the SVR4 brand that Novell paid so much for now become worth less than they had been. Users will not make purchase decisions based on the source of a vendor's kernel technology. They will evaluate operating systems based on functionality, reliability, availability, serviceability, and all the other criteria they see as important. Whether or not the operating system is based on SVR4 will decline in importance.

Of even greater importance to Novell and USL is how the future direction of Unix will be controlled. To a large extent, the responsibility for shepherding the standard that defines "true" Unix is now in the hands of X/Open. Future evolution, at least at the level of interface specifications, will be a process managed by that consortium. Others, such as Unix International or the OSF, may make recommendations, but control now rests squarely in the hands of X/Open, not USL or Novell. This is good news for the consortium, giving it renewed credibility, and, with branding revenues, potentially more cash. Users will most likely feel that the future of Unix is in the hands of a more neutral party than it has been in the past. However, operating systems are only a part of open systems, and X/Open's charter extends to much broader and much more difficult issues, not the least of which are system management, object-oriented standards, and certification-testing. — *M. Goulde*

An

Invitation

to

The Seybold Executive Forum℠

October 24th through October 28th
Hancock, Massachusetts

Bringing Enterprise Systems into the '90s (& Beyond)

Redesigning Legacy Systems and Manual Processes
into Distributed Client/Server Applications
and Streamlined Workflows

Learn by Doing

Using:

- Rapid Applications Prototyping
- Imaging & Workflow Technologies
- Business Process Redesign Methodologies
- Object Modelling & Rules-Based Design
- Electronic Data Interchange (EDI)
- Tools for End-User Information Access & Retrieval
- Information Architecture Planning Methodologies

To Register or for a Complete Program Brochure,
Call (617) 742-5200, or (800) 826-2424

Patricia Seybold's Computer Industry Reports Order Form

Please start my subscription to:

		U.S.A.	Canada	Foreign
☐ *Workgroup Computing Report*	12 issues per year	$385	$397	$409
☐ *Open Information Systems*	12 issues per year	$495	$507	$519
☐ *Distributed Computing Monitor*	12 issues per year	$495	$507	$519

Please send me a sample of: ☐ *Distributed Computing Monitor* ☐ *Open Information Systems* ☐ *Workgroup Computing Report* ☐ *Paradigm Shift—Case Studies in Distributed Computing*

Please send me information on: ☐ Consulting ☐ In-Depth Research Reports ☐ Conferences

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

Name: _____ Title: _____

Company Name: _____ Dept.: _____

Address: _____

City, State, Zip code, Country: _____

Fax No.: _____ Bus. Tel. No.: _____

Card #: _____

Exp. Date: _____

Signature: _____

Checks from Canada and elsewhere outside the United States should be made payable in U.S. dollars. You may transfer funds directly to our bank: Shawmut Bank of Boston, State Street Branch, Boston, MA 02109, into the account of Patricia Seybold Group, account number 20-093-118-6. Please be sure to identify the name of the subscriber and nature of the order if funds are transferred bank-to-bank. **IO-993**

Send to: Patricia Seybold Group: 148 State Street, 7th Floor, Boston MA 02109; FAX: (617) 742-1028; Phone: (617) 742-5200; MCI Mail: PSOCG

1992 and 1993 Feature Reports from Patricia Seybold Group
To Order these Back Issues, Fax (617) 742-1028, or Call (617) 742-5200.

Office Computing Report

Volume 15 issues—$40

- ❒ 11/92 Visual Programming—Application Design for End Users and Professional Developers
- ❒ 12/92 The Notes Phenomenon—The Industry Reacts to Lotus Notes

Volume 16 issues—$50

- ❒ 1/93 Microsoft Access—"Cirrus" Database Project Gets down to Earth

Workgroup Computing Report

- ❒ 2/93 WordPerfect Information Systems Environment—WordPerfect Reveals Its Blueprint for Workgroup Support
- ❒ 3/93 Lotus Notes Release 3—Extending the Notes Paradigm
- ❒ 4/93 Can Windows NT Meet the Challenge?—Microsoft's Next Generation Operating System Stirs the Industry
- ❒ 5/93 Action Technologies' Workflow Products—Coordinating the Activities of People as They Work Together
- ❒ 6/93 Implementing Groupware—Groupware May Be Hazardous to Your Organization's Status Quo!
- ❒ 7/93 Issues in Remote Computing—Notebook PC Boom Raises Question: How to Handle Detachable Computing
- ❒ 8/93 Scheduling-Based Workflow—IET's W-6 Attacks the Complex Area of Time-Dependent Processes

Back Issue Total $_____

Open Information Systems

Volume 7 issues—$50

- ❒ 7/92 Integrating Applications in the Real World—Evolution, Not Revolution
- ❒ 8/92 Windows NT 3.1—Microsoft's Bid for Desktop Dominance
- ❒ 9/92 Oracle's Version 7—Can It Leapfrog the Competition?
- ❒ 10/92 Galaxy from Visix—Application Portability Breakthrough?
- ❒ 11/92 European Open Systems Architectures—Europe's Vendors Strike out for Open Distributed Systems
- ❒ 12/92 The Unix Data Center—Fact or Fiction?

Volume 8 issues—$50

- ❒ 1/93 X/Open in the 1990s—Making Open Systems Safe for Users
- ❒ 2/93 Highly Available Open Systems—Expanding Today's Definition
- ❒ 3/93 Sybase System 10—Can It Manage Enterprise Data?
- ❒ 4/93 Unisys ASD Framework—Meeting the Challenges of Software Development in the 1990s
- ❒ 5/93 Unix and PC Interoperability—Toward the Utility Era of Computing
- ❒ 6/93 Open Electronic Mail—Interoperability through Standards
- ❒ 7/93 SAP's R/3 Software Suite—Architecturing Open, Integrated Software for Distributed Enterprises
- ❒ 8/93 Tomorrow's Microkernel-Based Unix Operating Systems

Back Issue Total $_____

Distributed Computing Monitor

Volume 7 issues—$50

- ❒ 11/92 Transarc Encina—Will Distributed OLTP Systems Overrun the Mainframe's Last Stronghold?
- ❒ 12/92 UI-Atlas Distributed Management—Object-Oriented, Distributed Management for the Unix System V World

Volume 8 issues—$50

- ❒ 1/93 Component Software—A Market Perspective on the Coming Revolution on Solutions Development
- ❒ 2/93 Switched Internets—The Coming Gigabit Revolution in Enterprise Networking
- ❒ 3/93 IBM's System Object Model—Cornerstone of an Open Distributed Object Computing Environment
- ❒ 4/93 Encapsulating Databases—Practical Uses of Object Technology to Inprove the Value of Relational Data
- ❒ 5/93 OMG's CORBA 2.0—Industrial Grade Standard for Distributed Object Computing?
- ❒ 6/93 Enterprise System Management—The Quest for Industrial Strength Management for Distributed Systems
- ❒ 7/93 Detachable Computing—Vendors and Users Scramble to Support Occasionally Connected Computers
- ❒ 8/93 Modeling Network Behavior Using Objects—Horizon Strategies' AROs Increase Simplicity and Flexibility of Distributed Applications

Back Issue Total $_____