POWER POLITICS

At Stewart Alsop’s recent Agenda ‘90 conference, one peevish sentiment was clear: Developers don’t want to write for multiple standards, but they can’t agree which ones to drop. They are still arguing over OS/2 and various flavors of UNIX, and are reluctantly concluding that they must also support DOS with Windows and some kind of memory relief, plus the Macintosh.

But picking an operating system is almost simple compared to the problems of interoperability, and the prospective unification of UNIX may ease the burden further. Picking an operating system is a question of writing your application for a target environment, and forthcoming tools (so it is promised) will make it easy to port from one environment to another, or to generate code and user interfaces for a variety of environments simultaneously.

More than just connectivity -- navigating across a variety of communication systems -- interoperability involves communication with applications and environments you don’t control, implemented through remote procedure calls (RPCs), E-mail and database RPCs. In its most powerful, fine-grained form, it requires exposure and specification of programming interfaces for applications, so that programs can get at each other’s functionality and share structured data, rather than send flat files or thin streams of data and commands in the low-bandwidth keystroke-emulation mode common now.

Last month, Sun, Novell and Netwise proposed a standard platform for remote procedure calls (RPCs), the cross-network calls that underlie distributed applications. The impetus comes from Sun, but the company that will make money most directly is Netwise, a Boulder company that builds toolkits and compilers for remote procedure calls. This platform will compete with another, from H-P/Apollo, Digital, IBM, Microsoft and Transarc, for the OSF’s (and the world’s) blessing as its Distributed Computing Environment.

While distributed applications are a single (logical) appli-
cation -- one set of source code compiled into called or calling parts to execute in various places for efficiency of data access or processor use -- cooperative applications come from the other direction: independent applications working together. Technically, this fine distinction means that the modules in distributed applications appear to be linked statically at compile time, while those in cooperative applications are linked dynamically at execution time. In fact, the location or exact identity of the called RPC module may be determined at runtime, but the application logic assumes it's fixed. IBM's APPC communication protocol, integral to SAA (see Release 1.0, 88-3) allows either the distributed or cooperative model; object-oriented programming takes the message-based, dynamic-link paradigm to the extreme.

Note that E-mail itself is a distributed application, frequently built on top of RPCs, that provides facilities for store-and-forward messages and cooperative applications. Promoting the message approach, Action Technologies held its first Message Handling Service users' conference, promoting its mail transport system, even as most of the E-mail community was in Chicago at the Electronic Mail Association conference, genuflecting to the X.400 E-mail standard. Separately, two companies, Agility Systems and Beyond Inc., emerged to kick off the new field of mail-enabled applications, which let users and applications send messages to one another. H-P's NewWave (Release 1.0, 87-12, 88-6) has already implemented some of the same ideas, but with a different design center, focused on the individual user's desktop.

Sybase's notion of database RPCs, despite the name, tends toward the message, dynamic-link end of the spectrum. A single client/server application may work as a distributed application, but multiple applications can cooperate through a client/server architecture, and can use common stored procedure calls (which behave something like objects). As Sybase and vendors such as NetFRAME (see Release 1.0, 89-1) envision it, applications are divided into user and server applications. User applications call on server applications, which serve functionality -- application execution -- as well as data. Sybase expanded on some of these notions with the announcement of specs and tools for its database (Open Server) and client (Open Client) interfaces. With these, third parties can compete with Sybase's tools or its database -- but they will also be more likely to support them.

All these announcements move the focus of discussion and standards wars well beyond the operating system and user interface. (RPCs are just a small part of the "distributed computing environment" technology that the Open Software Foundation, for one, is about to evaluate.) It's common wisdom (if not practice) that you can build your code to execute on any given operating system, and you'll do both UNIX/Motif and OS/2/PM eventually, with tools to ease porting; the challenge is to communicate across environments.

We're early in the cycle: For the next year, vendors in these areas will be more concerned with marketing platforms and tools to each other than with marketing applications to customers. The assumption is that standards sell.

Please note that on the next few pages we are talking about concepts as well as implementations. All these ideas slide into one another; you can use different technologies to implement one gestalt, and vice versa. You can have objects without object-oriented languages, and you can store and forward messages on top of RPCs. Remember, you can't tell a Modernist from an Impressionist by examining the pixels.
OPEN RPC: THE NEW STANDARDS WARS

Sun, Novell and Netwise have proposed a standard for remote procedure calls across networks, based on Sun's RPC syntax, which would allow application developers to write code once and compile it for a variety of environments with the help of the appropriate tools. Unfortunately, they made the marketing blunder of not giving the standard itself a name -- proof that no one owns it, but a situation that makes it difficult to refer to. Call it the RPC platform... Each vendor will have its own implementation: Sun RPC/XDR (part of its widely licensed ONC/NFS), Netwise RPC TOOL, and NetWare RPC, a NetWare-specific repackaging of the Netwise compiler technology.

The group has offered its RPC platform to the Open Software Foundation. It will be competing with a proposal, based on the RPC in Apollo's Network Computing System, from H-P/Apollo, Transarc, IBM, Microsoft and Digital Equipment. This platform is currently licensed to about 50 resellers. As with UNIX and OS/2, the functionality will be similar, but the implementation details will be different. Welcome to the (new) standards wars!

The goal of both RPC platforms is to move one level higher in the magic layer cake of network standards, so that application builders can build network applications and concentrate on functionality without having to worry about which network or operating system a user might be using. The would-be standards themselves are simply a syntax for packaging and directing procedure calls across a network. The platform implementations generally come in two parts: An RPC compiler, to generate the RPC source code, and a "transport-independent" interface that achieves its independence with a library of network-specific code modules which can send and receive the RPCs over the underlying network and operating system. (You could do this by hand, of course, but that's what tools are for.)

The application developer can concentrate on the functionality because the RPC code handles tasks such as sequencing and addressing calls correctly, transforming data formats, and dealing with communication details. The result is not that a single given shrink-wrapped package can run anywhere, but that source code can be compiled to suit any particular set of environments and a shrink-wrapped package can be sold for that specific combination. Moreover, common RPCs let programs developed independently communicate with each other -- subject to higher-level conventions on data formats and semantics. Examples include Sun's network management system, which collects and responds to information about network activity and problems. Independent vendors can build their own agents/clients to communicate with it.

The Apollo and Sun/Novell/Netwise RPC platforms handle data and parameters somewhat differently, but the results are equivalent -- although unfortunately not compatible. Generally speaking, Apollo's contains more content -- which Sun believes should be handled at a higher level.

What is RPC?

Remote procedure calls are subroutine calls that happen to take place across a network. An application merely specifies that it is calling a procedure, and a pre-processor (commonly, if incorrectly, called a compiler) that knows which procedures/subroutines will be remote generates the appropriate RPC code (or "stubs") as necessary. That code is regular source code, and gets...
compiled with the rest of the application. The calling stubs transform data and parameters into a universal format that gets reinterpreted by server stubs at the other end.

The server stubs are generated in a similar way. The developer defines a variety of procedures or subroutines callable by the client part of the application, and a tool generates the appropriate code to interpret the commands and data generated in the RPC format for execution by the called procedure. The server stubs manage such things as scheduling requests from multiple clients or among multiple servers, forwarding calls, defining transactions to maintain integrity, registering themselves with the appropriate network directory service, and the like. The stubs use the local operating system functions; they can even provide operating-system independence on the server side, Netwise president David Andrus asserts, as long as you write clean, OS-independent code. (On the client side, they do not handle user interfaces.) The Netwise TOOL adds its own memory-management scheme in the DOS server implementation.

The user's own compiler generates the appropriate object code for each target environment, and the resulting platform-specific components can be sold as shrink-wrapped packages for a given kind of hardware and operating system.

At execution time, the stubs call on a library of interfaces included in the network software or operating system on each hardware platform. These interfaces handle the actual communication tasks, including finding the called procedure through a network directory, and hide the transport-level details from the applications.

But it's not magic. RPCs enable a single application to be split into separate remote components, or separate components to be joined; they don't provide interoperability between parts that would not otherwise work together. That's the job of the higher-level tools described on pages 5 and beyond.

RPCs in use

One example of RPC use is a server performing a database call at the request of a client query tool running on a PC. Another is a screen manager displaying a graphical object at the request of a client application running on a hardware server. Yet another is an E-mail package which beeps at Juan, acting as a server in response to a message (heralded by a procedure call) from the client on Alice's workstation. If Juan responds, he will use client code in his copy of the application to send a message back to Alice's machine, now acting as a server. It is the mail server that "stores and forwards," or reroutes a message, following procedures called by the client. The RPC itself is synchronous.

At the RPC level, server and client processes are generally present on both sides if you look closely at any cooperative (or "client-server") application. In each case, the requesting code is the client, and the responding code is the server. But both kinds of code modules can reside anywhere; client code can run on "server" hardware, and vice versa. Remote procedure calls enable them to get to each other even over a network without dealing directly with the network.

Release 1.0 17 October 1989
The marketplace

Both camps have a large number of adherents. Some 75 vendors (out of 280 licensees) are currently delivering the Sun RPC platform syntax as part of Sun's Open Network Computing/NFS communications software, which will be a standard part of UNIX System V.4 and is already part of NetWare. It has support (not necessarily exclusive) from a large line-up including Interactive Systems (an UNIX vendor now owned by Kodak), Lotus, Oracle, RTI and Sybase -- and from 3Com and Banyan, which together with Novell provide potential access to 75 percent of the current installed base of networked pcs. (But that would require a mass upgrade.)

The Apollo RPC platform is part of Apollo's Network Computing System. The NCS camp has about 150 licensees (few of them shipping yet), but a powerful line-up of supporters, including many of the OSF heavy-hitters, as well as 3Com, which is supporting both efforts but is not yet selling either. Both DEC and IBM will include it in their operating systems.

The Apollo system is optimized for seamless, high-bandwidth connectivity, perhaps at the expense of breadth of platform coverage, but the capabilities (not syntax) of both systems are converging. The Netwise tools have made Sun RPC easier to use and less cumbersome, and Apollo and its licensees are adding support for IBM and DEC platforms, and for wide-area networks and OSI (which Sun already has). Ideally, the whole issue is invisible to users, except in that it makes it easier for them to find applications that work together, and those applications show up faster because vendors have to write for only one network substrate instead of several.

OPEN GROUPWARE: MAIL-ENABLED APPLICATIONS

Remote procedure calls, precisely because they are so low-level, don't make much difference to the applications users see. They are non-disruptive, helping developers distribute an application invisibly. More interesting is a whole new kind of application (which can be built on top of RPCs): cooperative applications. This approach lets users work ad hoc with a group of independent applications, and lets the applications work with each other.

One handy communication mechanism for cooperative applications (as long as they don't need real-time communication) is E-mail. E-mail enhanced with user programming tools and interfaces to common applications and information sources provides support for end-user development of groupware. (All this requires a very different kind of tool from an RPC compiler, something on the order of a spreadsheet as opposed to a financial planning language. Remote procedure calls may underlie a mail system, as financial language routines underlie spreadsheets, but they aren't visible to most users.)

E-mail does not provide the power and seamlessness of remote procedure calls, but it's more accessible and error-tolerant. There is less advance coordination required. Remote procedure calls are synchronous, like phone calls, whereas E-mail is patient and non-intrusive, and provides a store-and-forward facility which can hold messages until the recipient -- person or application -- is ready.

Release 1.0

17 October 1989
Mail models

There are two budding standards for large-scale E-mail: IBM's SNADS (for SNA Distribution Service, which is part of SAA), and the TCP/IP Simple Message Transfer Protocol, or SMTP, which is evolving toward the formal CCITT X.400 standard. They correspond almost perfectly to SAA and UNIX -- in proprietoriness, culture and consistency of implementation. (DEC's VAXmail is also widely used, just like VAXes.)

At the other end of the spectrum is Action Technologies' Message Handling Service, an open standard (you can use the protocol without buying the tools, but your users will need an MHS server from Novell, Action or a licensee). It performs similar functions, but isn't directly comparable:
Differences in scale translate into fundamentally different systems. MHS is local, and works over local-area networks and asynchronous auto-dial. X.400 is global, and manages more complex data translation and a worldwide range of addresses...but it doesn't scale down well and it requires fairly sophisticated network underpinnings (an OSI protocol stack) to operate at all.
"X.400 is to an army tank," says Soft•Switch chairman Mike Zisman, "as MHS is to a Volkswagen. They're both useful, but for different tasks." (Presumably, then, SNADS is a private tank.) All this means good business for Zisman, whose company provides gateways among local-area networks, the various proprietary enterprise-wide systems such as VAXmail and IBM's PROFS, and the X.400 standard.

MHS has been around for a long time as part of Action Technologies' The Coordinator, and was built as a quick, tidy elegant way for The Coordinator to handle messages. Novell encouraged Action to generalize it, and now offers it (via a free coupon) as part of every NetWare system sold. You can also buy it directly from Action Technologies (see Release 1.0, 86-10, 89-6). Until this year it hasn't had much of a following, but 100 people turned out for the first-ever MHS developers' conference, including ComputerLand, which uses MHS to send messages between warehouses to manage inventories and other data. Like all these communication systems, MHS consists of protocols and a message-handling (server) component. The primary protocol is an 18-line ASCII file that contains parameters describing where the message should go, whether acknowledgement is required, etc. At either end, each MHS application needs to know how to implement "seal" and "unseal" routines, creating and interpreting the 18-line header.

Mail-enabled applications

These standards are a start. Now we need interfaces so that they can carry messages to and from applications, instead of just transmitting files and messages between people.

What would such an E-mail-based system look like? As it happens, it would look a lot like the MIT Information Lens and Object Lens systems, designed with grants from Xerox, Wang and DEC among others and running on Xerox D-machines with a UNIX revision underway (see Release 1.0, 86-10 and 88-9).

Two start-ups are working on just such a system, with object-oriented user tools, but using more widespread pc platforms. Both companies have worked with the MIT project leader, Professor Thomas Malone, and he is co-founder and director of, and consultant to, one of them, Agility Systems. Malone also runs MIT's new Center for Coordination Science.

Release 1.0

17 October 1989
Both these systems fill a gap that struck us hard at OOPSLA: Lots of object-oriented programming tools help you design application user interfaces -- with windows, dialogue boxes, radio buttons, scroll bars -- but none of them seems to know much about performing tasks, such as composing and distributing a memo, handling a customer request for a special service or planning a boss's itinerary. Nor do they know about programming interfaces to existing applications -- so that you could, say, get data from Dow Jones or a 1-2-3 file through a nice, user-friendly form, or request an advance through a nasty, scary form that asks the name of your first-born.

The trick is not to sell object-oriented programming, but to sell an application tool. We need class libraries that include people, resources, tasks, groups and interfaces to applications, as well as screen objects and attached procedures. E-mail can provide the transport for all this, so that no one needs to write complex remote procedure calls but can (invisibly) use those of the mail system, at least for non-realtime applications.

All this is very nice, but it requires a fair amount of plumbing-level work. Each client and each server application needs to know about message syntax and be prepared to look for messages on the mail server each time it is loaded. (With the RPC approach, by contrast, the server procedure had better be there or the client will simply get an error message.) The breadth of each vendor's application-interface library will be an important determinant of success, helped by vendor efforts such as Lotus' DataLens.

Compare and contrast: Beyond and Agility

Beyond was founded last year by Chuck Digate, with start-up funds from Lotus stock he acquired as senior vp of analytic software products. MIT also has an equity stake in return for a non-exclusive license to its technology. The Beyond development team includes people from ON and Ontologic (see Release 1.0, 89-9); its board includes Zisman, who sits on no other outside boards. He is excited by Beyond's potential to complement the plumbing systems Soft•Switch provides, and the chance to "cooperate instead of compete."

Agility was founded this month by John Landry, former evp of development at Cullinet, with start-up funds from the sale of his Cullinet stock and a settlement he received in Cullinet's recent acquisition by Computer Associates. Landry promises that full specs of Liaison will be released by year-end, with shipments to follow shortly. Agility will be using some as yet unidentified third-party technology to get a kick-start.

Neither system is implemented yet, so these descriptions are more than usually subject to change. Now read on....

Agility's system will be Mac- and PC/Windows-based, while Beyond's will start on the PC, with the Mac to follow. The Beyond system seems to be focused on group information-sharing, while Agility's Liaison will be more concerned with application integration, but that's a fine distinction. Both companies will sit on top of a wide range of mail systems and adhere to standards, they say, but are a little vague about precisely which ones.

Beyond will attack three intertwined information-oriented problems, says Digate: information overload (with filters, although he won't use the term in marketing), the restricted reach of information (it's hard to find and to

Release 1.0
17 October 1989
send), and people’s reluctance to communicate (why risk it unless someone asks?). In short, Digate sees a world where there’s too much information; information is not packaged right to reach its targets; and people are reluctant to add to the mess. The Beyond product will include an E-mail front-end (user agent, in E-mail parlance), intelligent agents (i.e. tools to build them), an engine that can process messages and trigger agents, and a lower-level interface for corporate and independent software developers to integrate their applications with the system. Beyond itself will provide interfaces for a variety of popular applications and formats.

While it’s hard to compare smoke with mirrors (which is which?), we get the sense that Agility’s Liaison, is more focused on the outside world (in keeping with Landry’s mainframe background). It will include the same general capabilities listed above, but Landry tends to talk about external information services -- Dow Jones, Lexis and Nexis -- while Digate talks about person-to-person communication and pc applications. Agility will provide support for a variety of input styles -- menu selection, phone buttons, voice, even fax -- to attract non-typing executives. Landry is excited about the interconnection of previously incompatible public and private mail systems -- MCI, Internet, VAXmail, PROFS, etc. -- while Digate seems to be focused initially on the closer-to-home issues of effective communication and application integration within a workgroup.

In the end, it’s a question of focus. Both systems will ultimately address workgroup, enterprise and wide-area communications. Both will have a user programming language and an interface for applications (unfortunately, different ones). You’ll be able to do more or less the same things with either, but initial positioning will depend on what object libraries and interfaces each vendor starts with.

Fantasy: I’ll have my agent call your agent

What can you do with either tool? Well, say Digate and Landry, you can build personal, organization or inter-enterprise applications, or integrate all three. The following scenario is our own, and is an example of what you might build, not what you would get, with either product:

Alice might be calling on a new customer, Malleable Metals Corp. She broadcasts a message over the company network asking if anyone has any information on Malleable Metals. She can’t just query the corporate database because (let’s be realistic) this kind of information is not yet kept in corporate databases. On the other hand, this is the kind of message people would normally throw away. So she sends it in the form of a message that says, "If you think you know anyone at MM, please press this button to execute a query against your personal database." All the receiver has to do is press the button, and the query is run against a personal database that would normally be off-limits to Alice. As it happens, Juan thinks he used to know someone at MM, so he presses the button. A few lines of data, plus a comment about why Jeremy the purchasing manager didn’t buy, are automatically sent back to Alice with no further work on Juan’s part.

Now Alice is set. She composes a nice letter to Jeremy with an apologetic reference to the product deficiency, now fixed, that provoked his scorn last year. She incorporates some text and diagrams from a corporate database of product information. She sends a copy of the letter to Juan personally,
and to the entire sales department. Fred, new in the company, reads it avidly, but Henry, an old hand, has programmed his system to ignore any mail addressed to the sales department -- unless it comes from the sales manager. (Henry may miss a couple of interesting tidbits, and he knows all about MM and would have enjoyed telling Alice so, but that's his problem.)

After Alice takes Jeremy to dinner at the Quilted Giraffe, she files her expense account, using a form developed by the accounting department for company-wide use. The system takes one look at the amount ($300 for dinner for two), compares it to her spending limit -- and sends the report to her manager for review. He puts it on hold, but releases it to the accounting department for payment (by pushing a button) after a brief chat with Alice.

Later on, when Alice wins an order from Jeremy, the order form automatically notifies her boss as well as the shipping and invoicing groups. She offers to buy Jeremy a copy of her mail-enabled toolkit and send him her company's order interface so that he can place follow-up orders automatically. If he wants, she'll even offer to query his inventory system periodically, so that he'll never run out, but it may take a while for her to earn his trust. For now, he's content to place orders when he thinks it's time. (This is an implementation of EDI, or electronic data interchange, for which a whole set of higher-level standards -- purchasing, ordering and billing transactions -- is being developed by ANSI and EDIFACT, a UN-sponsored group.)

Behind the scenes

All this can be done fairly easily -- if you have the right tools. Note the use of messages, forms, filtering and external communications (using some sort of gateway). End-users can send their own messages, and people such as those in accounting can create expense-account forms that flow information directly into the general ledger and a special tax-accounting system. The pc support group may get involved in building an EDI ordering system that's integrated with the production and shipping departments. Alice's word-processor has a couple of extensions that enable her to download properly formatted boilerplate from a corporate text base, and to send copies of her letters by selecting from a menu integrated into her word-processor. The only time she used the Beyond or Liaison E-mail system directly was when she broadcast her message searching for information on Malleable Metals. To build that request, she selected "fuzzy search" from a menu, added her criterion (Malleable Metals), and typed an explanation of her request in a short cover note (an unstructured text field, for those who care). For each user who executed the query at her request, the system translated it into the proper format for that user's data files.

Now, a clever programmer could set this all up today. However, he or she would have to do most of the work by hand. Tools such as Agility's and Beyond's will provide a variety of application-specific interfaces to manipulate existing applications and their own interface for user-developed applications, and will use existing transport mechanisms. Initial applications will no doubt include automatic calendaring, and a "capture" function that could take almost any data and turn it into an object. Eventually, with a more integrated system, virtually all the data in a company will have become objects. In other words, you could accomplish work through the system, which "knows" what's going on. Jeremy, for example, will be registered as a potential customer; when his order comes through, the system gives him some new characteristics as an active customer.
NewWave and Together

It's hardly fair to talk about these two products, whose promises haven't
been tested yet, without mentioning Hewlett-Packard's two-year-old NewWave,
which is real if not yet complete. NewWave has approached the problem of
communicating applications at the single-workstation level first, with more
concern for the individual user's operating environment and office automa-
tion than information management. (It could be a handy environment for
Beyond or Agility.) Specifically, the most noticeable thing about NewWave
is its linking of data files and applications into "objects," so that users
needn't deal with them separately. Beyond that, it provides linking facili-
ties so that, for example, text, graphical and spreadsheet or database data
can be incorporated in a single compound document without losing its charac-
ter (i.e., each component is an object with its own methods). NewWave has a
number of standard messages, such as display and print, which are inter-
preted appropriately by each object. But the system's agent facility is
still limited: It's either keystroke capture and replay, with limited sup-
port for conditional statements, or a language targeted at MIS types.

Yet another approach is that of Together from Coordination Technology (see
Release 1.0, 88-6), an OS/2-PM-based product due out next year. It focuses
on collaborative task automation and workgroups and has a less ad-hoc, indi-
vidualistic feel than we expect from Agility and Beyond.

An inside view

A major contribution of Agility and Beyond (following the original Informa-
tion Lens) is semi-structured messages -- forms with both human-oriented
text and with data that the system can use for filtering, routing decisions
or to generate the appropriate messages for each application supported.
Some of these application messages will be at the keystroke level, while
others will be direct application-to-application communication; it all de-
pends on how easy it is to get inside the third-party applications. With
the exception of some databases, most pc applications at present have user
interfaces, but not much of a programming interface.

That's why, as far as we can tell, Agility and Beyond are focusing their in-
itial efforts on information-intensive applications -- collecting and fil-
tering information, sending transaction messages to databases, and routing
messages among people -- rather than the more-complex, inside-the-applica-
tion efforts that Hewlett-Packard is taking on with NewWave.

The NewWave team's goal is to get more and more application vendors to pro-
vide application programming interfaces so that applications can deal with
each other directly rather than pretend to be users, passing identified com-
mands, data sets and parameters rather than keystrokes (i.e. calling spe-
cific application functions that it "knows" about). It has spent much of
the last two years rounding up and fostering support from third-party appli-
cation vendors, who are tailoring their applications to work with NewWave.

The promise of Agility and Beyond is to work from the outside and accomplish
effective integration more quickly through end-user tools for communication
with and manipulation of a variety of more-open applications -- applications
that are targeted to answering queries, executing transactions, and return-
ing large volumes of information.

Release 1.0

17 October 1989
OPEN SERVER: SYBASE OPENS UP

Joining the forces of openness, Sybase has announced Open Server and Open Client (nee DB-Lib). In the context of RPC standards and the application-manipulation tools discussed above, the Sybase protocols and tools are designed for application vendors who want to build either client or server applications to work with a wide variety of other servers or clients, as the case may be. Sybase is using its own well-received SQL Server as an anchor in trying to establish its protocols as standards, while the mail-enabled applications and NewWave will be selling the power and ease-of-use of their user interfaces and end-user programming tools.

Sybase has taken some criticism for its proprietary language, Transact-SQL, a programming rather than a query language (as in Structured Query Language) used for writing server-based stored procedures. But its approach is coming into its own. The traditional notion is that databases/servers store data, and applications/clients manipulate data. The user/client describes data for the server (using a "data description language") and queries it, and writes applications to do things with it. The server maintains data integrity mostly by protecting it from unauthorized changes, and transactions are defined by the applications.

In the Sybase model, the server has applications too: They are shared procedures that can be assembled and sequenced by client applications to create ...call them "composite" applications...that comprise a set of allowable actions as determined and built by the dp department or other central group.

This model maintains data integrity at a higher level than the traditional data-centered approach: SQL Server enforces a hierarchy of permissions. Some users and applications can use only stored procedures, which (presumably) follow or enforce business rules. Some tables can be changed only through stored procedures. Except for the lack of explicit inheritance and other trappings, this model in effect turns the data in the database into objects, accessible only through defined procedures (methods).

I'll have my procedure call your procedure

A traditional transaction -- debit must match credit -- is a simple, data-oriented example of integrity constraints. A high-level example would be reflected in the following order application assembled from stored procedures and governed by rules and triggers on the server: A user application's request, "Process this order," causes quite a commotion at the server. First, it checks the customer's credit. Then, if the response to that query is positive, it sends a message to the shipping department, to create a pick list and a shipping label, and to the receivables group, to create an invoice and a credit entry. If one product is out of stock, the shipping department (executing a trigger procedure) notifies the billing department, which might note the fact on the customer's invoice. Company policy (reflected in yet another stored procedure) determines whether the customer gets billed anyway, or is billed only when the missing product is shipped.

This could all be handled as several client applications, of course, but it's easier to keep it on the server, both for integrity and efficiency. For starters, there would then be no way to ship a product without generating a bill for it. (Now we know why we get review copies of software with bills for $0.00.) Second, suppose this company now decides to go into
telemarketing. The new telemarketing application runs on a Macintosh, so as to be accessible to the part-timers who handle the phone lines. They fill in the data on the Mac, which sends the data along with a database RPC that calls that same application. The Mac front-end calls Open Client library modules to transmit the RPC and data.

Two other changes are made, by adding conditional statements to the database RPCs that check credit and generate invoices: Instead of searching company files, the credit-check routine queries a public database of credit-card risks using the credit-card data provided by the telemarketers. And the invoice RPC generates a credit-card charge that is transmitted to the credit company. All this is much simpler than generating a new application.

Thus the Sybase approach is a way of defining, building and segregating libraries of reusable code modules along with their data. The particular procedures SQL Server has supported with Transact-SQL have been traditional database programming, but Sybase is now attempting to generalize the calling interface -- with Open Client and Open Server -- so that other kinds of functions can be called and shared in the same way. A user application ends up being a series of calls to the shared application code in the servers. And, like traditional objects, the servers now have the capability of sending database RPCs to each other, as of Sybase's Release 4 of SQL Server.

The Open Interface

Sybase is trying to establish a standard way of communicating between clients and servers. The primary models are Sybase's own front-end tools and SQL Server. Open Server and Open Client contain toolkits and intermediary interfaces which handle communications, etc., so that users needn't be concerned with system underpinnings (once someone has built and installed the server and client components using the toolkits). It also provides protocols for calling "database RPCs," which can manipulate whole tables and other data types as well as the atomic data typically handled by RPCs, and which can call other database RPCs.

While SQL Server handles relational data structures (including fields that may hold text or image data), the Open Server interface offers a way to make other kinds of server applications accessible to Open Client users. For example, stored procedures on non-SQL servers could filter stock prices from a real-time feed, find letters matching keyword criteria, or generate results from calculations. Open Client contains tools and reusable code libraries for generating requests from a server and handling the responses; Open Server contains tools and source code for receiving and sequencing requests and delivering the results. (Servers have to use Open Client to send calls to other servers.) The application content is still up to the developer.

In this high-level context, the client is always the client, although underneath the covers the server may send a remote procedure call to the client or to another server. The presumption here is that the client and server may come from independent vendors, and there will probably be a published set of services available. In the RPC model, both client and server come from the same vendor, even though they execute on different machines, and the degree of mutual knowledge is much greater. RPCs make it easier to distribute a single application; database RPCs make it easier to share applications and tools. This will change in time, however, and will raise interesting intellectual property issues. See the next few pages.
RELEASE 1.2: INTELLECTUAL PROPERTY REPRISE

We recently attended a two-day workshop on intellectual property composed mostly of lawyers, with a sprinkling of industry types, one economist and a few others. We don't actually want to retract anything we said in our August issue, but we learned a lot and gained some valuable perspective, which we now want to share. Specifically, we favor (without great fervor) a shorter time limit on protection, and (as before) a specific statement by vendors of what they claim to own. Meanwhile, as applications take on more of the flavor of languages as inter-application communication increases (see the previous 12 pages), proper definition of intellectual property increases in importance. We strongly believe that changes in the software marketplace in the future will require clear, straightforward protection of the genuinely new work to enable that market to flourish.

Overall, while we still favor strong protection of intellectual property, the long 17- or 50-year limit of patents and copyrights respectively makes us uneasy. Suppose there really were some technology, the withholding of which could hurt the public interest or perhaps give the withholder some egregious monopoly position that could not be overcome by antitrust law? This is a fast-moving business, and the effects of any protection policy are hard to predict, since there may be clever ways to use or circumvent it.

For those reasons, it might make sense to reduce the period of protection to five years -- long enough for a developer to realize a profit on a worthy effort, but not so long as to cause any irrecoverable harm or permanent monopoly. We don't see anything particularly terrible (except for a lot of bureaucratic wrangling to make it happen) about a 5-year limit on copyright and patent protection for software -- although it might make people eager to embody their software in hardware, and might create other market dislocations we can't predict. If a shorter time limit is the price of getting strong, clear protection for software, so be it.

In the end, it is not the role of the intellectual property laws to guard against unfair (use of) market power. They should encourage the market, and antitrust laws should discourage the abuse of market-given power. As a business decision it is frequently in a vendor's interest to put all or some of its creative work in the public domain, as Adobe is doing, since widespread use of any of that work will increase the value of complementary products in which it might have a time advantage, and will certainly increase the value of its implementation work. (It will also forestall competitive inroads.) We believe the availability of strong protection appropriately leaves it to the owner to decide which components to make freely available and which to protect.

Patents vs. copyrights

Secondly, we tended to dismiss the differences between patent protection (17 years) and copyright protection (50 years or more), since anything longer than 10 years in our industry extends before recorded history or beyond the predictable future. The fundamental difference is the specificity of patent protection: You have to assert upfront what it is that you are protecting, and it is those claims -- not something asserted ex post facto -- that govern the case. Then those claims are compared both with what the supposed infringer is doing, and (if the infringer can find any) with prior art that might limit those claims.

Release 1.0 17 October 1989
We’re beginning to feel that something closer to the patent system might be more appropriate to what we would call the "functionality" of a product, or its design, as opposed to its implementation in code (both source and object and any automatic translations) and its screen displays and audio components if any, which are properly protected by copyright. Since patents protect useful inventions, they may be more appropriate than copyright for everything other than displays and code. Patents adhere to the "define-yourself" rule, laid out below and in Release 1.0, 89-8. The purpose of such self-definition, as we see it, is not just reward for effort, but avoidance of redundant effort. However, the patent system doesn’t work well yet for software; it needs to be streamlined and turned into a market rather than a repository (or black hole).

Define yourself

For intellectual property, the medium is not the issue. The intellectual property system is predicated upon providing incentives both for good solid engineering work (implementations), and for creativity, which is best represented and protected as specifications.1 "Protection" is accorded to the results of intellectual effort in order to foster the effort by allocating rewards to it. Those results include applications’ structure, sequence and organization (to the extent that they represent original work and are not a mirror of common real-world procedures), interfaces, protocols, etc. Of course, effort isn’t rewarded directly: What may have been tough for one person could be easy for another, and a brilliant idea is worthless if someone else implemented it before you (even if you could have thought of it yourself). This year’s hot product is next year’s left-over.

Thus the most important issue is a proper definition of the stuff -- not of software in general, which is like trying to define art, but of the particular "family jewels" being protected in each case. These are analogous to the particular arrangement of paint on a canvas (not the frame, and not the canvas or the paint), or to the particular sequences of notes and of words, separately, in a song (not the instruments used, or the medium, which are akin to the language used in a program, and not the particular recording, which is akin to the implementation, or work product, that embodies the specification represented by the intellectual effort).

---

1 Exact correspondence between value and tangible unit is always illusory, but many modes of pricing will work, and will make a market. For example: We live in New York, where the subway costs $1, regardless of distance traveled and discomfort suffered, and taxis charge by miles, plus minutes of waiting time. In Washington, where the workshop was held, the subway is priced more or less by distance, and taxis charge by zones. This minimizes taxi drivers’ incentive to take the long, costly way and customers’ incentive to take short but congested routes that waste taxis’ scarcest resource -- time. Additionally, a one-dollar rush-hour surcharge reflects taxis’ greater value (because of demand) during rush hour. The precise units of utility and incentives vary, but they correspond roughly to the benefits received, and overhead is minimized because the system is clear.
The well-lighted field

Our basic assertion (as laid out in Release 1.0, 89-8) is that the playing field should be well-lighted -- not necessarily even. Vendors themselves could provide the guidelines, rather than rely on uncertain interpretation of possibly inapplicable copyright laws. They would be required to define their property promptly and clearly, and others would be able to determine whether or not they risk infringing. (There would still be room for challenges on broad claims, but at least claims would be disclosed beforehand.)

Why is this so important? First, it puts the infringer on notice of what is considered proprietary and not to be copied. (We assume everyone knows you may not copy the implementation outright.) Second, it helps the market work by letting potential users know what has already been invented. A good description will also serve as an advertisement. The market works: It will serve to foster innovation and its benefits, but it can do so only when it has economic objects with which to work, and that is what proper definition of intellectual property will provide. If you can’t define it, how can you provide incentives for it?

The existence of protection for intellectual property encourages not just its production (which might happen anyway) but its distribution and use. Goods are produced to be used or sold, and "contain" their own value. The value of intellectual property lies in its use. But the value to the originator goes down if it is broadly used -- unless he can own it and get compensation for its use (see "Standards," page 17, for some caveats). The absence of intellectual property protection would ultimately encourage secrecy: There would be a disincentive to spreading information about software, since this would lead to loss of value and competitive advantage.

A brief history of software as a business

How have we come so far, then, with our relatively loose notions of intellectual property? The reason is this: In the beginning, software was part of a relationship. You bought your machine and your software from the same vendor, and there was a direct contract, which might be measured by the machine it was licensed for use on, or the number of users, or some other measure of economic utility.

More recently, software has come in discrete packages, easy to define and price by the piece, if hard to protect from literal copying. There is no signed contract, and the software works by itself -- a fine model for traditional copyright protection. Developers are working hard to make software even easier to use, with on-line "help" -- press a special combination of keys, and the software itself will flash up a message telling you how to use it. This is great for customers -- and it’s pure intellectual property.

But in the future, the market will shift again. Independently developed and sold software packages will be expected to work together without benefit of cler -- er, heavy-duty systems integration efforts. Software will increasingly be sold in function-sized components that are expected to interoperate over networks. The software "package" as a unit of functionality, and of the "value" that intellectual property law is designed to encourage, no longer holds. As described on pages 1 to 12, applications are no longer discrete packages of functionality, but are rather collections of functions
addressable individually. In other words, applications are becoming increasingly like languages: What's the difference between a procedure call and a verb in a language? Intellectual property protection will encourage the publication and specification of the interfaces that will make such interoperability possible.

Reuse through interoperability and through modification of existing code will be of benefit not just to developers, but to users. From the present world in which separate companies develop software as a business, and are willing to provide support and marketing as part of that business, we will move to a world in which the users develop the software -- as a byproduct of running a business. That is, restaurants will be developing their own recipes. What are these recipes? Things like purchasing procedures, expense-account management, financial products, production-planning techniques...any business policy or procedure that can be routinized and automated.

Their choice will be to keep them secret for competitive advantage, or, if there is an appropriate market with defined property rights, sell them for financial gain (even as they work on improvements which will keep them ahead of their competitors).

How can these businesses be encouraged to share their recipes rather than keep them secret? How can we encourage a secondary market in application components? The prospect of financial gain from selling their intellectual property will give them the incentive to publicize it, market it and teach others to use it -- activities they would otherwise avoid. (Whether they do so themselves or contract with others to do so is immaterial.)

If the rights could not be assigned and the "property" were free, what would happen then? Perhaps there would be big business opportunities in training, software identification and support -- but the atmosphere would encourage secrecy, since knowledge of what software existed to solve a particular problem would be at a premium, and it would be in the interest of the originators to keep their work a secret -- available only (say) to consulting clients under contract. Thus knowledge about the software would be kept from the market, since the software itself could not be protected.

How it could work

The hope, of course, is that software itself will be able to help us manage this bureaucracy -- to define software, classify it, maintain on-line databases, and help developers discover if they're inventing a new wheel, or if they can reuse one already invented by someone else. That, in essence, is the basis of our argument for intellectual property protection -- to encourage not just the creation but the specification, definition, classification, distribution and use of intellectual property. As software-development techniques improve, we are approaching a point where the clear statement of a problem implies its solution. The welfare of the world in general should be improved by a system that encourages people to share their expertise, even for a fee, rather than keep it secret.

Standards

As we are seeing with user interfaces, there's a value-subtracted that comes with the kind of diversity that copyright is designed to encourage -- the
The cost of learning to use a new interface. Add to that the cost of re-engineering software components to talk to other software components if we can't agree on (pricing for) standards. Those are real costs that customers should be willing to pay money to avoid. (If they perceive those costs as too high -- or if other vendors see an opportunity -- alternatives will be proposed, just as Microsoft and Apple are now doing for PostScript.) Moreover, intellectual property protection provides the incentive for vendors to build products worthy of standardhood. Later on, the market frequently works to make the standards more available. Regardless, we don't believe it makes moral or practical sense to legislate that standards should be free. There's a difference between "open" and "free."

Conclusion

The goal is not fairness, which is probably impossible to achieve anyway. People may be rewarded more or less than they really deserve, and some products become standards for reasons other than sheer quality (such as speed to market, superior marketing or simply expectations, as when they come from a market leader). The market doesn't always work each time; it works statistically. There are points in the cycle where vendors seem to have too much power, or market or profit share, but things usually even out in the long run. In the continuing cycling between standards and chaos, clever players foster the standards that benefit users, and are then forced either to innovate further, or cede their growing power to others who are jealous of it -- as the Adobe case illustrates.

In the midst of all this movement generated by self-interest, technological progress is a by-product. The mere availability of protection will foster faster spread of innovative ideas and techniques, and the establishment of standards. Standards can spread widely under license or with permission from the owners, who have an incentive to foster their use. We certainly hope to avoid a ponderous bureaucracy like the Patent Office; it's our hope that encouraging vendors to "declare" their property would encourage other vendors to provide accessible information services about it.

A POSTSCRIPT ON ADOBE SYSTEMS

When a company acquires (or is believed to have acquired) too much marketplace power because of its exclusive ownership of a standard, the market often produces a counterforce. A case in point: Microsoft and Apple last month challenged Adobe's hegemony in font standards, disrupting what looked like a stable, clearly lighted (if not level) playing field, and, says John Warnock, "made us look like an underdog." In response, Adobe promised to open up (i.e., publish the specs of) its font technology. This saga provides a perfect illustration of the role of intellectual property in the software business.

It may well be true that Microsoft and Apple would have gotten together even had Adobe released its technology at last spring's Seybold Conference instead of this fall's, and it may be true that Adobe would have released its technology anyway, but Adobe would have won more sympathy by moving earlier.

It's a delicate game. In the end, Microsoft and Apple will have a tough time countering Adobe's "web" of licenses and matching its implementation.
expertise, but much of the outcome still depends on IBM's position, which is unclear (as usual). Adobe played the game well, but screwed up the timing.

What for Adobe now? It has hardly given everything away. It still retains almost everything it has; it has merely opened up potential competition that may take a couple of years to show up -- at which point Adobe should be well along on extended versions of what it's now selling.

The whole episode strengthens our belief that the market usually takes care of things. The only question is, Would we have had PostScript at all if Adobe had no rights to it? Probably. Do we really need intellectual property protection if market forces make companies give it away in the end anyway? For now the answer would be yes, but a shift in the market towards components and user development (see "Open Groupware," above) makes our answer no. Without intellectual property protection we might not get the next thing after PostScript in an increasingly fluid world.

For further reading:

"Standardization and Intellectual Property," by Joseph Farrell, Berkeley and Hoover, July 1989. This is an edited version of a paper given at the CES-LaST (Arizona State's CEnter for the Study of Law, Science and Technology) conference in Phoenix last February. It's a lucid examination of the complex interplay between standards and intellectual property rights. Conclu- sion: "Stronger protection of intellectual property may retard or may advance de facto or informal standardization; it plausibly does retard de jure or formal standardization, because it tends to increase vested interest... [This suggests that] certain forms of intellectual work in network [interoperability-dependent] industries should be less protected than in other industries." As one who prefers de facto standards, we disagree, but we find Farrell's reasoning and exposition compelling. For copies, call his office at (415) 642-9854.

"Policy and Pragmatism: The Broader Issues in the Debate on Software Patents," by Brian Kahin, Esq., draft, September 1989. Again, we disagree with what we perceive as the conclusions, that patents should be avoided, but this paper clearly explains the problems with the system, and decries the bureaucracy and paperwork it entails. So let's fix the system, not abandon it. For copies, call (617) 864-6606.

Release 1.0 is published 12 times a year by EDventure Holdings, 375 Park Ave., New York, NY 10152; (212) 758-3434. It covers the pc, software, CASE, groupware, text management and connectivity markets, artificial intelligence and legal issues. Editor & publisher: Esther Dyson; associate publisher: Daphne Kis; circulation & fulfillment manager: Lori Mariani; executive secretary: Denise DuBois; editorial consultant and copy chief: William M. Kutik. Copyright 1989 by EDventure Holdings Inc. All rights reserved. No material in this publication may be reproduced without written permission; however, we gladly arrange for reprints or bulk purchases. Subscriptions cost $395 per year, $475 overseas.
MAKE MINE PERSONAL

What will computers mean for mass media? There are two clearly visible results: personalization and interactivity -- which are simply two forms of the medium's knowledge of the user, one static and one dynamic. Personalization, easy to achieve now, raises social questions; interactivity still faces technical challenges.

Just as technology is giving us the promise of a "global community," it is also giving us the means to fragment it into tiny interest and market groups.

Previously people might have had different experiences from the same book, movie or TV show; now even the tangible content of those things can vary according to who experiences them.

What will that do to our ability to share experiences? What happens to shared cultural artifacts such as the Johnny Carson Show or Roseanne Barr?

We're already seeing some fragmenting of the media -- the mass-market magazines and TV shows have lost their sway, replaced by cable television, videos, business magazines targeted variously at entrepreneurs, chief information officers and computer users, women's magazines aimed at specific age or lifestyle groups, etc. They form our filters onto common experiences -- which movies to see, which books to buy, etc. Today's movies reach blockbusterhood by attracting people who see the same film three, four or more times, rather than by reaching vast numbers of different individuals.

Filters? Or blinders?

Personalization involves filtering -- finding and selecting items that meet the user's criteria. It's a way of reducing the huge amounts of stuff out there to something of the kind and quantity that an individual can handle.

It's easy to accomplish, but is it always appropriate? As time passes, what will happen when everyone is reading and seeing tailored information? Old salt Henry, on page 9, is losing touch because he filters out new information. It's a little worrisome, because people can learn more from experiences they didn't seek than from reinforcing the familiar. (Imagine an infinite supply of Nancy Drew or Hardy Boys books. Some kids might never graduate to anything else.) If Alice likes Dave Dinkins for mayor of New York and screens out all the negative news on him and all the favorable news on his opponent Rudolph Giuliani, can she be a responsible voter?

Of course, people do that all the time, but computers can make it easier and more airtight. If corporations had effectively screened out all Mac ads and Mac salesmen five years ago, we might still be working on character-based pcs. (As it happens, we still do, but at least we know what we're missing.)

Nearer to home and closer to now, consider the magazines you read. Do you read every article and ad? Or do you generally skip the sections on money-market instruments? Do you invest in stocks or gold? There are some advertisers out there who want to talk to you -- and only you. With a combination of techniques -- database management, marketing feedback and selective binding -- they'll be able to, in a cost-effective way that hasn't been

Release 1.0

17 October 1989
possible until recently. Selective binding means the assembly of tailored versions of magazines. Other methods include selective distribution of coupons, and, of course, direct mail itself.

**Kaleidoscopes and magic mirrors**

Such personalization by third parties is...call it efficient, or call it sinister. Giuliani may decide Alice is not worth advertising to; he'd rather spend his scarce campaign resources on Juan. Those who believe in advertising believe in it partly because of its educational value -- but it is persuasion rather than education that advertisers are paying for, and if they can separate the two they probably will. Preaching to the dubious is expensive!

With luck, the need for common experience may overwhelm the abilities of computers to give us uniqueness and targeted marketing. Just watch how people dress, how mesquite smoking swept the eating-out community, how people line up to play the same arcade games... Part of it is the impact of mass advertising, which may disappear, but part of it is a universal desire for safety in numbers and shared experience. Despite market fragmentation, people talk to each other about the videos they see, and have friends over to watch with them. Somehow a movie isn't as much fun if you can't share the experience, either during or after. People will still use personalized media to select mass media, and vice versa. People still listen to the radio even though tapes are just as handy -- partly for news, partly for breadth of selection, and partly because they need to know what's hot.

(Note that not only information will be personalized. Right now, cars come in a variety of colors and configurations; it's just that you still have to pick from the selection that your dealer has available. In the distant future, with production carried out by reprogrammable robots, customized industrial and consumer goods will be commonplace. Just as in the old days, everything will be made individually -- but volumes will be orders of magnitude greater because the process will be automated.)

**Interactivity: A million tiny variations**

Enough of social commentary. While automatic personalization and targeting is possible, both at the user's behest (see Desktop Data, page 21) or by advertiser selection, interactivity poses a whole different set of technical problems. Someone's got to make the stuff.

When you're talking about an interactive game, or an exercise tape that says "Very good, Alice!" even when she's a foot away from touching the floor with her fingers, that's easy, but a rich-textured interactive novel or video is something else again. The first hypertext novel, "David's Sling," for example, took author Marc Stiegler almost five months beyond the original three years to restructure for hypertext -- even though he had written even the linear version with a hypertexty tool. (All this was in his spare time, in case you were wondering. Will sheer scale make hypertext become like movies instead of books, produced by a team of specialists directed by one person with a vision?) But this is still only user selection, or "dynamic personalization," within the confines of a large block of multi-media content. Hypertext enables you to create personal paths through mountains of text or video material, but it's not interactive in the sense of adapting to a user.
Basically, it's pretty easy to create variations on digital information: You can create a large number of examples of Boolean-search exercises or spreadsheet problems by varying only a couple of parameters, and tailor a training exercise for a user's particular deficiencies. But you can't do the same easily for a movie with multiple possible plot lines, and tailor it to a viewer's mood or political leanings. The actors would have to do their thing over and over, slightly differently, when it's difficult enough to get them to do it right once.

Son of Tin Toy

Content-based interactive experiences will have to be big sellers just to cover their costs. We may end up in a world of simulated acting, an extension of the achievements of John Lasseter and the Pixar team in Tin Toy, an approaching-lifelike animated video of a live tin toy and its encounters with a live but inhuman baby. Will people endure ersatz characters for the sake of interactivity? We doubt it. Although the distinction is fine, interactivity will initially mean mostly selection among a variety of canned or generated responses, rather than what we would call true interactivity, where a system responds to the user rather than lets the user select.

It's this fine distinction that will maintain the line between people and interactive machines. You can program machines to be random (or irrational), to be sure, but only a person can achieve consistent irrationality. (We hope so!)

ONE MAN'S MULTIMEDIA IS ANOTHER MAN'S HYPERMEDIA, OR ANIMATION, OR AUDIO...

We've always been a little skeptical about the appeal of multimedia, quite apart from our general hostility to a term that is used mostly to mean "next-generation neat stuff" with whatever particular features excite the speaker. But we were more impressed than we expected to be with Apple's demonstrations of videodisks front-ended by HyperCard. As it happened, we had just talked to television producer Dick Hubert, who is trying to make a business out of the inevitable arrival of video advertising. Everywhere you turn, vendors are offering VCR cassettes to promote their products -- Apple, IBM, Pyramid, Lotus, others. But they have to do all this by direct mail. Why not create a medium -- a video magazine -- on which they can piggyback?

The idea intrigued us, but we can't imagine sitting still through an hour, or even half an hour, of video. This is the time-pressed generation, after all. We flip through magazines, zap commercials, chat on the phone in our cars, read mail during meetings, and when we talk to only one person we give it a special name -- "quality time." Hubert's solution -- that you can fast-forward -- just doesn't cut it in an age of remote-controlled tvs and, yes, hypertext.

Apple's combination of HyperCard and videodisk is the medium that will be able to carry all those messages. The immediate problem, of course, is cost, which translates into a small installed base to receive those messages, and a high cost to reach them. For a year or two down the road, Hubert's plan makes perfect sense. Advertisers will pay a lot to reach the readership of a controlled circulation publication aimed at heavy-dollar computer purchasers. (Let's start with a high-tech audience likely to have
The sticking points are two. One is getting quality editorial, which costs a lot -- especially if you're spreading the cost over hundreds or thousands of viewers instead of the millions you can reach with regular tv. (Of course, advertisers will pay a lot more for such a targeted audience; this is the volume/quality tradeoff of cable tv extended one notch over.) The second is that not enough people yet have the systems to watch this on. You need some kind of front-end, HyperCard on a Mac or the equivalent on a PC, and a video system attached. The vendor, of course, has either to provide different versions for each set-up, or go with just one.

DESKTOP DATA: ALL THE NUTRITION, NONE OF THE CALORIES

Yet another vendor has come out with a filtered-news service. This one, NewsEDGE from Data Resources and Lotus veteran Don McLagan, is delivered right on your pc, in real-time, instead of on fax like the two other personlized computer-oriented news services we covered last summer (Release 1.0, 89-7). For better or worse, NewsEDGE is not industry-specific -- but it can be tailored to be. Yet it still lacks the industry-specific details and insight of the trade press abstracted in Computer Focus.

NewsEDGE addresses the higher end of a broader market, costing anywhere from $7500 annually for a starter kit of PR NewsWire and McGraw-Hill News, up to $40,000 for the complete service which will include Reuters and Dow-Jones -- as soon as Desktop Data can get the contracts signed. (By comparison, Reuters on a teleprinter is $7200 per year, and Dow Jones is about $150 per hour plus connection charges if you get it on-line over a public network.) NewsEDGE lacks the take-it-to-the-washroom convenience of the fax news services we covered earlier, but it delivers more information, and in electronic form so that you can manipulate it yourself.

Desktop Data does this through an FM receiver that feeds the on-line services onto your pc (in background), along with a filtering tool that lets you select criteria that will trigger alerts and store relevant articles in a separate clippings file. It's up to you how much hard disk to devote to the live news database, which will flush itself automatically. Otherwise, it could grow at three megabytes a day if you took the full service. The clipping file, which stores only stories you select, is managed separately.

While the fax services are perfect for now, and are also a good potential vehicle for advertising, Desktop Data's NewsEDGE is closer to the model of the future. It gives the user direct control over his criteria, and gives him the news in real-time, whereas Computer Focus is an intermediary, abstracting the news and providing a once-a-day feed.

Desktop Data's filtering works in a number of ways, all of them automatic. It provides the usual keyword matching, date and time criteria, and allows you to work with whatever categorization the news vendors themselves provide. McGraw-Hill, for example, attaches one or more of 120 categories to each story it transmits. More interesting, Desktop Data (like Individual Software and the Dow Jones DowQuest Connection Machine-based service) pro-
vides relevancy feedback. In other words, if you select one story from a list, you can say (in effect), "Get more like this one." Using the original story as a model, it can then generate a query, or profile, to find similar articles. That query can be saved or edited, and given a name. Thus you can generate multi-word queries for concepts. (See Release 1.0, 88-1, for a full discussion of these techniques.)

Compared to the news service vendors themselves, NewsEDGE offers the same news in a single, fixed-fee package, with powerful filtering and manipulation techniques and a usable interface. At this point, Desktop Data's contract limits reuse of the news feed to a limited number of print-outs for internal distribution. As we noted in August (see Release 1.0, 89-8), a major part of the information distributor's job involves packaging and pricing the product. Thus the news is all there, but we expect to see substantial evolution in the product as it gets installed and users start to understand the power of information in electronic form. It's too valuable just to read.

"Today's the day NewsEDGE earns its keep," says Don McLagan. It's the Monday after the Dow fell 190 points, and anything could happen. We however, are sitting here in blissful (or dangerous) ignorance, without NewsEDGE.

Computer Focus: a midterm assessment

Computer Focus, after a number of weeks, is growing on us. Last week, for example, it put all the object-oriented news from OOPSLA on our front page -- while for most computer folk, Sybase, Oracle and Lotus were probably the front-page news. However, we find its coverage irritatingly brief, and we don't feel we can call you for "further details" on 20 stories a day. (For starters, our fax machine couldn't hack it.)

On the other hand, we're delighted at the new policy of putting phone numbers at the end of some items. Many of them are reworked press releases (a useful service) and we like to be able to call the companies involved.

(P.S. to Computer Focus -- We hope you're building a file of all those names and numbers; you could sell those too for a handy sum.)

QUESTIONNAIRE RESPONSE

The responses to our questionnaire about videotaping the Forum were interesting (sorry about the teeny envelope!). The overwhelming sentiment was: Keep the transcripts; we don't care that much about videos. And the reasons were as noted above -- who wants to sit through hours of linear experience when you can flip through a book in the bathroom? So, the market has spoken.
The theme this year is "Into the seismic decade: Standards and earthquakes." We will explore the delicate interplay between standards -- temporary rigidities or fault lines in the market landscape -- and the huge upheavals that occur when the forces of change and innovation take over. Are standards a fool's paradise, or an appropriate compromise between practicality and disruptive progress? Can large companies innovate? Can small companies succeed (without big backers)? Do the standard-setters have too much power? Is the traditional pc community losing power to a new world of networked systems, MIS departments and corporate standards? Should it care, or should it embrace the new world and change it from within?

More important: Are we addressing the right set of standards? The problem of multiple operating systems will eventually be ameliorated not by the actions of a single company but by cross-OS tools, clearer market segmentation and a gradual shift to new standards. But the standards wars won't be over; we'll be able to fight about interoperability and communications protocols (see pages 1 to 12), database standards, distribution and pricing practices.

Addressing these issues will be the speakers and panelists listed across, and a few more to be added. We're limiting the number of speakers and panelists in order to leave extra time for "virtual panels" with you and your peers in the audience -- customers, resellers, competitors, suppliers, investors. Speakers and panelists will discuss how standards are set and superseded in a variety of contexts, including: "Why the operating system is obsolete" (Epstein), dynamic data exchange, RPCs, E-mail and messages as approaches to interoperability (Gates, Metcalf, Reinstein and Zisman), end-user application tools (Tesler and Landry), "The care and feeding of intellectual property" (Warnock), "The evolving box" (Canion), "The software design manifesto" (Kapor), "Microsoft isn't the enemy" (Eubanks), application servers (Torresi), "Behind the pretty (inter)face" (Liddle), "Evolution and revolution in microprocessor technology" (Slater), the laptop market as the last bastion of truly personal computing, and other topics.

Thinking Machines founder Danny Hillis will give our "token user talk" this year. He will discuss "Simulating evolution," showing how he uses Connection Machines to model evolving populations...and evolving standards. There are some intriguing parallels.

But that's not all. An opening reception, Monday night's dinner honoring Ben Rosen, and other meals and events will also give you ample opportunity for a key part of the Forum, the chance to mingle with your peers away from office pressures and among the cactuses. In keeping with the conference's seismic theme, Colleen Barton of Stanford will lead a geology field trip for spouses and others (adults only) in the Tucson countryside. The Forum hotel, the Westin La Paloma, sits on a desert mountainside. It's thirty minutes from the airport, which has good service through Phoenix from the West and Dallas from the East. There's shopping, swimming, tennis, golf...and day-through-evening childcare, so bring your family.
Speakers and panelists

Rod Canion	 Compaq
Bob Epstein	 Sybase
Judy Estrin	 Network Computing Devices
Bill Gates	 Microsoft
Danny Hillis	 Thinking Machines
Steve Jobs	 NeXT
Bill Joy	 Sun Microsystems
Jerry Kaplan	 GO Corporation
Mitchell Kapor	 ON Technology
Bob Kavner	 AT&T Data Systems
John Landry	 Agility Systems
Dave Liddle	 Metaphor
Jim Manzi	 Lotus Development
Dennis McEvoy	 Cooperative Solutions
Bob Metcalfe	 3Com
Harry Reinstein	 Aion
Ben Rosen	 Sevin Rosen
Michael Slater	 Microprocessor Report
Larry Tesler	 Apple Computer
Enzo Torresi	 NetFRAME
David Tory	 Open Software Foundation
John Warnock	 Adobe Systems
Ann Winblad	 Hummer-Winblad
Mike Zisman	 Soft-Switch

Company presentations will include Agility Systems (Liaison), Aion Corporation (Application Shells), Answer Computer (Apriori), Apple (multi-media), Desktop Data (NewsEDGE), Interactive Images (Easel), Lotus (Notes), Lysis (Support Information System), MacroMind (Director), Netwise (RPC TOOL), Pinpoint (Computer Focus), Saros Corporation (FileShare), Silicon Graphics, V.I. Corporation (DataViews), Xanadu (the Xanadu server) and others to be announced.

Release 1.0	 17 October 1989
RESOURCES & PHONE NUMBERS

Tom White, Action Technologies, (508) 358-5259
John Warnock, Adobe, (415) 961-4400
John Landry, Agility Systems, (508) 358-5259
Andy Klein, Saul Marcus, Paul Leach, Apollo, (508) 256-6600
Chuck Digate, Beyond Inc., (617) 621-7123
Roger Moody, Coordination Technology, (203) 268-4045
Don McLagan, Desktop Data, (617) 890-0042
Bill Crow, Webb McKinney, Hewlett-Packard, (408) 773-6289
Dave Andrus, Mark Hatch, Netwise, (303) 442-8280
Darrell Miller, Novell, (801) 379-5900 or (800) 453-1267
Dave Tory, Open Software Foundation, (617) 621-8700
Harvey Golomb, Pinpoint Information Corp., (800) 767-3636 or (703) 263-0407
Mike Zisman, Soft Switch, (215) 640-7448
Larry Garlick, Sun Microsystems, (415) 960-1300
Bob Epstein, Stewart Schuster, Sybase, (415) 596-3500
Phil Lehman, Transarc, (412) 338-4406
Dick Hubert, Videoware, (212) 399-3811

For further reading:

"MHS Technology: A Briefing," by Action Technologies (above), 1989. This paper discusses not only MHS, but all the other "standards" MHS will communicate with, and puts it all in perspective.

COMING SOON


- CompuServe, Prodigy, MCI Mail, USENET, Internet, et al. The only way we know to get around to getting online is to commit ourselves to writing about them...

- Network navigation.

- Application servers.

- And much more... (If you know of any good examples of the categories listed above, please let us know.)

Release 1.0

17 October 1989
RELEASE 1.0 CALENDAR

October 20-21 A delicate balance: Technics, culture and consequences - Los Angeles. Some thought-provoking topics, but will the right people be listening? Call Chantal Toporow, (213) 813-6194 or Nik Warren, (213) 392-6595.


October 25-27 SGML '89 - Atlanta, GA. Chaired by SoftQuad's Yuri Rubin-sky; sponsored by Graphic Communications Association. SGML will be to text what SQL is to data; here's a chance to discover it early. Call Norm Scharpf, (703) 841-8160.

October 26 The R&D gold mine - Ann Arbor. Sponsored by Regis McKenna. "Improving your effectiveness in using nonprofit research organizations as a source of new products and technologies." Contact: Elizabeth Batson at (415) 857-9388.


October 29-November 1 3Com network systems forum - San Jose. Sponsored by 3Com. With 3Comers, plus Microsofties Steve Ballmer and Mike Murray, Lantean Marke Tebbe, H-Pite Doug Chance. Contact: Cheryl Soderberg, (800) NET-3Com or (408) 562-6400.

October 30-November 1 Seventh annual Seybold Executive Forum - Boston. "Preparing for the global information age," with Mike Zisman (page 5), Bob Kayner, Ellen Hancock, Sam Fuller, McGraw-Hill's Thanos Triant, Dow Jones' Bill Dunn, MIT Media Lab's Muriel Cooper, others. Sponsored by Fatty Seybold's Office Computing Group. Call Deborah Hay, (617) 742-5200 or (800) 826-2424.

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>November 1-3</td>
<td>*UNIX expo - New York City. Keynote by noted UNIX fan Ken Olsen of DEC; also speaking: Bill Joy, Bertrand Meyer, John White. Managed by National Expositions Co. Contact: Roger Halligan or Heidi Dethloff, (312) 332-4650 or (212) 391-9111.</td>
<td></td>
</tr>
<tr>
<td>November 5-10</td>
<td>Hypertext '89/SIGDOC 89 - Pittsburgh, PA. Much larger, for better or worse, than the first, wonderful hypertext conference in the fall of '87. Hypertext covers the first three days; SIGDOC the last three. Sponsored by ACM. Contact: Elise Yoder at (412) 327-8181 for Hypertext '89; Nina Wishbow at (412) 323-2600. (How about a joint committee on standards for the use of apostrophes?)</td>
<td></td>
</tr>
<tr>
<td>November 6-8</td>
<td>C++ at Work - Tyngsboro (outside Boston), MA. Making religion work: Product demos and useful tutorials (one called &quot;Black holes in C++&quot;). Speakers include C++ author Bjarne Stroustrup (keynote) and representatives from Borland, Apple, Ontologic, others. Sponsored by BU's Wang Institute. Contact: Carol Canning or Lisa Gennell, (508) 649-9731.</td>
<td></td>
</tr>
<tr>
<td>November 13-15</td>
<td>UIST - Williamsburg, VA. Symposium on user interface software and technology, sponsored by ACM SIGGRAPH and SIGCHI. Contact: John Sibert, (202) 994-4953.</td>
<td></td>
</tr>
<tr>
<td>November 13-17</td>
<td>*Comdex - Las Vegas. Also including MACdex. The original: No oat brasn, objects or other additives. Contact: Jim Collins at (617) 449-6600.</td>
<td></td>
</tr>
<tr>
<td>November 13-17</td>
<td>Supercomputing '89 - Reno, NV. Conveniently located near Comdex, if you tire of small computers and big hoopla. Keynote by Cray Research chairman John Rollwagen. Contact: Ron Bailey, (415) 694-4500.</td>
<td></td>
</tr>
<tr>
<td>November 15-16</td>
<td>The repository conference - Orlando, FL. Once IBM has announced, could such a conference be far behind? With Ken Orr, Burt Rubenstein (recently moved from Index to Bull), IBM's Hayward Schwartz, others. Contact: Cynthia Burt, (508) 470-3880.</td>
<td></td>
</tr>
<tr>
<td>November 28-29</td>
<td>Forrester technology management forum - Cambridge, MA. &quot;Client/server computing: The vision becomes reality.&quot; With the usual suspects: John Sculley, George Conrades, Ken Olsen, Bill Gates, Enzo Torresi, Ray Noorda, Bob Epstein, John Young, Frank King, Ed Esber, rising star Wayne Carpenter of Saros, others. (Try putting these egos in order!) Call George Colony or John McCarthy, (617) 497-7090.</td>
<td></td>
</tr>
</tbody>
</table>

Release 1.0 17 October 1989

November 29-30  Emerging software technologies conference - Innisbrook (near Tampa), FL. Sponsored by ADAPSO's Software Industry Section. For developers, commercial or in-house. With interactive sessions on object-oriented databases, mail-enabled applications, IBM's repository manager, viruses, and other current topics. Discount for ADAPSO members; call John Gracza, (703) 522-5055.


December 4-6  *First international conference on object-oriented and deductive databases - Kyoto. Sponsored by IEEE, MCC, many others. Contact: Professor Kiyoshi Agusa, 011 81 (75) 256-1677, or Won Kim at MCC, (512) 338-3439.


1990

January 15  Massachusetts Computer Software Council annual meeting - Newton, MA. Contact: Joyce Plotkin, (617) 437-0600.


January 18-19  Second annual conference on software support - San Francisco. How to turn a problem into a competitive advantage. Sponsored by the Institute for International Research. With Barbara Brizdle, Software Strategies; Deborah Fain, Lysis Corp.; Tom Evans, Answer Computing (see Release 1.0, 89-7). Contact: Georgette Asherman, (212) 883-1770 or (800) 345-8016.


Release 1.0  17 October 1989
<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Location</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>January 28-31</td>
<td><strong>EDventure Holdings PC (Platforms for Computing) Forum</strong> - Tucson, AZ.</td>
<td>Tucson, AZ.</td>
<td>Sponsored by us! New speakers will include Danny Hillis, Thinking Machines; Mike Slater, Microprocessor Report; Rod Canion is returning. See pages 24 to 25. Contact: Daphne Kis, (212) 758-3434.</td>
</tr>
<tr>
<td>February 6-8</td>
<td>Goldman Sachs technology investment symposium - New York City.</td>
<td>New York City.</td>
<td>Analysts' outlooks and companies' promises. Call Andrew Krawitt, (212) 902-7771.</td>
</tr>
<tr>
<td>March 5-9</td>
<td>*Seybold Seminars '90 - Boston.</td>
<td>Boston.</td>
<td>...moves east. Call Kevin Howard, (213) 457-5850.</td>
</tr>
<tr>
<td>April 23-26</td>
<td>*First international conference on systems integration - Morristown, NJ.</td>
<td>Morristown, NJ.</td>
<td>Sponsored by ACM and IEEE groups. Call Peter Ng, (201) 596-3387.</td>
</tr>
</tbody>
</table>

*Release 1.0 17 October 1989*

April 25-27  *Conference on office automation systems - Cambridge, MA. Sponsored by ACM and IEEE groups. Call Joan Staunton, (212) 869-7440, or Robert Allen, (201) 829-4315.

April 27  *Computer Bowl - Boston. Sponsored by the Computer Museum. Call Kate Jose, (617) 426-2800.

May 1-3  Second annual conference on innovative applications of artificial intelligence - Washington, DC. Sponsored by the American Association for Artificial Intelligence. Contact: Claudia Mazzetti, (415) 328-3123.

June 3-6  Spring Comdex - (back in) Atlanta. Sponsored by the Interface Group. Call Elizabeth Moody at (617) 449-6600.

June 3-6  *ADAPSO Conference - Washington, DC. Call Frank Ianacone at (703) 522-5055.


June 20-29  Design Automation Conference - Orlando, FL. Sponsored by IEEE and ACM groups. Call P.O. Pistilli, (303) 530-4333.

July 31-August 3  *AAAI-90 - Boston, in the heart of AI-land East. Sponsored by the American Association for Artificial Intelligence. Contact: Claudia Mazzetti, (415) 328-3123.


October 21-25  *ECOOP-OOPSLA 90 - Ottawa, Canada. Joint conference on object-oriented programming: Systems, languages and applications. Sponsored by ACM. Contact:

Please let us know about any other events we should include.
-- Denise DuBois

*The asterisks indicate events we plan to attend. Lack of an asterisk is no indication of lack of merit.

Release 1.0  17 October 1989
SUBSCRIPTION FORM

Please enter my subscription to **Release 1.0** at the rate of $395 per year in the U.S. and Canada. Overseas subscriptions are $475, airmail postage included until December 31, 1989.* Payment must be enclosed. Multiple-copy rates on request. Satisfaction guaranteed or your money back.

Name

Title

Company

Address

City State Zip

Telephone

How did you hear about Release 1.0?

89-10

*Effective January 1, 1990 the subscription rate will be $495 in the U.S. and Canada, and $575 overseas.

Please fill in the information above and send with your check payable to: EDventure Holdings Inc.

375 Park Avenue, Suite 2503

New York, NY 10152

If you have any questions, please call us at (212) 758-3434.

Daphne Kis

Associate Publisher

**Release 1.0**

17 October 1989