NOT BY GENIUS ALONE

Why is software late? This is not a new topic, but one usually addressed as a consumer issue or as gossip. In fact, it reflects some fundamental problems and challenges in software development that the U.S. must address.

But first, much of the current commotion over late software is spurious: One reason that software seems "late" so often is that it is announced early under pressure from Wall Street and from competitors. Sometimes it's to keep a stock up; sometimes it's to keep at bay competitors who have announced but not shipped similar products; sometimes it's to stake out a pre-emptive position for a company or product.

There is only one legitimate reason for announcing early: To let both end-user customers and third-party users of a platform know enough to plan their own development strategies or to start building supporting tissue. If others will rely on the software, those others need to know about it early. Of course, many companies hope that others will come to rely on their products and that they are building platforms rather than spot offerings. While this is sometimes self-aggrandizing self-delusion, on occasion it is true -- and will increasingly be the case in a world of interdependent systems and object-oriented class libraries.

For now, adherence to this pre-announcement strategy is why operating software and tools in particular frequently look late: Everyone is tired of hearing about them by the time they come out, even when they're on schedule. Imagine the hue and cry had Microsoft shipped OS/2 without announcing it first...and the commotion that did ensue when third parties decided that AT&T and Sun weren't revealing enough about the next UNIX.

A second factor behind the perception of lateness is software's transformation into show business. It's no longer possible to work in obscurity, shielded from analysts and press. When Mitchell Kapor and Jonathan Sachs built 1-2-3, they could hardly get anyone's attention; now, The Wall Street Journal covers ON Technology before it has even announced a product.
The culprit is connectivity

Yet, late or not, software is taking longer and longer to build and seems to provide less value per unit, however measured. Why? The reason is the exponential growth of complexity as systems grow larger and heterogeneous, which leads to slower progress and diminishing returns per unit of development work. The bandwidth of communication both among people and among systems grows exponentially as projects get larger and need to fit in with existing systems. It used to be that you could write a single application to do a single task on a single machine or at least within a single session on a multi-tasking machine. (Of course we wrote software that way because that was all the hardware we had. A question: Is the 386/486 good enough to let us abandon our old hardware-aware, discrete applications? Or will it take the SPARC and N10 to support the kind of rich software we're now building?)

But now we need to worry about interprocess communication, multitudinous hardware environments and communication protocols, user interface specifications, data structures and integrity constraints for shared data...

The answer of course is modularity -- but that's different from insularity, and it raises both technical and sociological issues. Computer-aided software engineering, object-oriented programming (see page 7), "interoperability," environment-independent development tools, universal operating systems and standards in general are all solutions -- but that's like saying that money is the solution to poverty. How to achieve these worthy goals, not their essential desirability, is the question.

Modularity and object-orientation will enable discrete parts of a system to be developed separately yet work together -- thus theoretically enabling us to reduce the size of development teams and better manage the programs themselves. But those modules do have to work together in the end, and the teams do have to communicate to make them do so. So we have to recast the problems: How can we ensure that modules can work together? What are the standards and protocols the modules must embody? How do we get the teams (rather than the individuals) to communicate? What do we lose if we standardize? What happens to the geniuses in a world of conformity?

Yet we would argue that in the long run the graph plotting development effort against value of software (ROI, in a word) is best drawn with exponential axes in both directions: As the challenge grows greater and the work harder, programmers will nonetheless ultimately grow more productive as they benefit from cumulative investment in development not just of applications but also of tools -- tools to build tools, in short. But before we get there, we must address the challenge of building not large interdependent systems, but numerous reconfigurable, interoperable modules in efforts that must be coordinated rather than controlled centrally.

The goal of these technological attempts at covering up the plumbing is to create a world where creative geniuses rather than technicians once again can reign. But first, someone must connect the pipes and run mains from one water system to another. Beautiful houses without plumbing are still uninhabitable, and ease of use without power is useless. To use yet another analogy, we're trying to pad the steering wheels so that user can't tell whether they're square or round, but we must still give those users the ability to turn on a dime.

Release 1.0 10 April 1989
A new world of retrofit and Incrementalism

By and large, the issues now are no longer brilliance and discovery, but precision, efficiency, compatibility and adherence to standards and specifications. Pretty dull, eh? For a long time, the design of interoperable, integrated systems will not be susceptible to mass-production or automated design techniques, although database management for design data (see page 12), syntax-checkers and other tools will be of great help. Still, they will provide a workbench rather than an automated production line. In Toshiba’s Software Factory, for example, people are still much in evidence (unlike most Japanese automated facilities). The difference from other software development centers is that the people are well-coordinated, if not regimented.

A world view: Shifting the balance

Those of us in the software industry tend to look at its health as a measure of its economic impact: Are sales still growing at 40 percent per year? Are the vendors profitable? Are the stocks up? But software isn’t just a product to be sold; it’s a capital investment to be exploited.

The economic impact of the software industry itself pales beside the impact of its use by the rest of the economy to make its other resources more efficient and effective. Software didn’t give us just Lotus and Microsoft and Cullinet, with total capitalizations of a few billion dollars; it gave us Federal Express, American Express, American Airlines and Citicorp. It’s behind our standard of living and our productivity as a nation. It’s going to give us yield management everywhere (see Release 1.0, 89-2), increasing efficient use of assets in general.

But ineffective use of software is also giving us tax cheats and screwed-up billing, wasted water for lack of cost-based pricing, and publicized disasters such as the recent student-loan billing debacle and inventory problems at Carson Pirie Scott. The troubles at Lotus are nothing compared to the delays in efficient tax collection, the time people all over waste on paperwork and redundant data entry, the waste of inventory because of inefficient management.

Software is easy enough to buy, but it’s hard to install and hook up to the rest of what’s there. Much as we love creativity -- and we’re looking forward to the next neat metaphor along the lines of the spreadsheet, the outline, the PERT chart, the perceptual map (see Release 1.0, 89-2) -- nifty software packages are not going to give our country a competitive edge. In fact, companies in other nations can buy them almost as easily as we can. Software can be duplicated instantaneously, but it is put into use and its value realized only slowly.

Building great software packages gives you a healthy software industry; effectively using great software packages gives you a healthy country. Although we reject the xenophobia the following discussion implies, we are about to suffer a serious competitive disadvantage against the Japanese in the use (not the sale) of software. They are free to buy the results of our creativity -- OS/2, UNIX, Windows, the Macintosh, 1-2-3, SQL Server, Framework, Logic Gem, ThinkTank, CASE tools, expert system shells -- on the open market. For the moment, they aren’t terribly effective at using packages;

Release 1.0

10 April 1989
they build most of their applications from scratch, and don't yet have much use for personal productivity tools. Nor are they likely to be effective building software for the U.S.; application software in particular holds a great deal of implicit content and doesn't cross cultures easily.

But the Japanese are excellent at building large systems and working together in groups to do so. This capability will enable them to use software more effectively in all their industries and will provide them a competitive advantage not just in a narrowly defined software sector, but across the board. They have the people skills and attitudes that are ideal for building large, interconnected software systems. Toshiba's WorkBench software "factory" employs 2000 people. For now, the country is hampered because each system is built from scratch, and its close to a million programmers are developing a lot of redundant code. But there's very little redundant code within systems, where it's most prevalent in the U.S. Right now Japan also suffers from a terribly inefficient distribution system -- which is why the country has a lower standard of living than ours despite higher manufacturing productivity. (Other reasons include lack of physical space and agricultural protectionism.) But their manufacturing productivity has enabled them to compete effectively overseas, and the corps of programmers the country is building will become a far greater power when they start applying their efforts more effectively using reusable modules (which we may supply). In short, the trends favor them, with their capabilities for cooperation, and disfavor us, with our creativity and relative lack of discipline.

(Most of our discussion of the Japanese software environment is derived from two studies, one by Ed Yourdon in his excellent newsletter, American Programmer, December 1988, (212) 769-9460; the other from ADAPSO, "Japanese Software: The next competitive challenge," January 1989, (703) 522-5055. We recommend both highly; we have touched on only a portion of their contents here.)

USERLAND: PLUMBING FOR THE MAC

As noted above, support for connectivity will be key in the systems of the future. Here's one example of such support...

Does the Mac support full-featured multi-tasking and interprocess communication (IPC)? No. Can you do multi-tasking and IPC on a Mac if you're willing to do all the work? Of course.

Part of the value of OS/2 and UNIX is the things they do: multi-tasking, memory management, interprocess communication. But another important value is given to them by the market: the implicit assurance that if you follow their rules, your software can interact with the software of other vendors who will do likewise. On the Mac, there are standards for multi-tasking as seen in MultiFinder and print-spooling (although there is less control of process scheduling than in, say, OS/2). But there are no general standards for IPC on the Mac, although the world eagerly awaits one, and Apple promises to announce one soon -- next month perhaps. (There are some vendors who do it within their own programs, such as TOPS, between Macs and PCs, and Lightspeed C, which communicates with the program you're debugging.)
A brief explanation of RPC

Remote procedure calls are commands (usually followed by a list of parameters) that can be executed remotely by another system. A number of vendors have established sets of "standard" RPC commands used to establish communications and perform common functions such as printing and opening files. If the commands are to be executed on other systems, there must also be a process that translates data into a common format and back, so that any system can read the data properly. Thus, "RPC" refers both to a set of environment-independent verbs and to the environment-specific systems that can interpret, translate and execute them within their particular environments. Precisely because the RPC system is "aware" of this distributed environment and can translate across it, it lets applications send "normal" procedure calls to other machines without knowing that they will be executed remotely (or at least by an independent system).

For now, there are a number of different RPC implementations. For example, Apollo offers a full-featured RPC protocol as part of its powerful Network Computing System environment, recently licensed by IBM, DEC and HP among others. NCS is competing for the endorsement of the Open Software Foundation against Sun's NFS, part of Sun's Open Network Computing, which is licensed to over 70 (mostly UNIX) vendors and uses Sun's RPC. In addition, Sun Microsystems' TOPS unit has SoftTalk, a proprietary RPC protocol set for communication between Mac and DOS systems; we assume Sun will merge the TOPS and ONC RPCs soon. OS/2 LAN Manager also has includes an RPC facility.

Unlike messages in the object-oriented world, RPC-dependent communications still need to know precisely what they will find at the other end -- which is why standards are so important. Theoretically, in an object-oriented world, you can just send messages assuming that the things on the other end will know what to do. In fact, object protocols are just higher-level abstractions that will probably be based on top of RPC.

An interoperable joke

This all reminds us of a software-industry joke. (Like all such jokes, it's interoperable with your choice of personalities.)

Alice, president of a high-flying communications software vendor, is sitting alone in her office when the phone rings. "Hello," she says into the phone. The other party hangs up.


The phone rings once more. "Who is this?" Alice shouts, by now quite annoyed. "It's Juan," says Juan, a disgruntled customer. "I just wanted you to know what it's like trying to establish a communications session with your product."
Stepping into the breach for now are Dave Winer and his new company UserLand Software, which he started after leaving Symantec in March 1988. Winer is a Mac programmer well-known enough (through his creation of ThinkTank, mostly) that he has some hopes of establishing a standard for IPC beyond what Apple is likely to announce. We assume that Winer is waiting for Apple’s announcement to refine the positioning of his own product. Certainly, no one’s going to do much with UserLand until Apple shows its own hand next month.

We also assume Apple has looked at UserLand IPC, currently shipping to developers, but for the moment Apple is working on its own approach. As we understand it, Apple’s IPC is a low-level platform that supports such a variety of protocols that it defines no real standard for communication. In this case, UserLand has a nice platform to extend. (One person familiar with both efforts calls Apple’s approach a "connection," in contrast to UserLand’s implementation of remote procedure call protocols.) Otherwise, users will go for the Apple approach regardless of technical merits.

UserLand IPC will give Mac programmers the ability to do IPC in a standard way, thereby communicating with other programs as well as within their own. Ultimately, UserLand’s goal is to build a standard RPC tool, akin to those in OS/2 and UNIX, for communication across environments rather than merely within the Mac environment.

UserLand is developing its IPC not merely from altruism, but in hopes of creating a market environment for its own new applications, all of which will be next-generation and assume the existence of IPC on the Mac. They could range from little utilities that communicate with IPC-aware applications, such as a slide-manager that could organize images or pages in any graphics or presentation program (not just PowerPoint) or an E-mail utility that could work from within any other IPC-aware package; to traditional applications (spreadsheet, word-processor) enhanced with IPC and other features; to a user shell/language that could use IPC directly to manipulate other programs and act like a UNIX shell or a batch/command language such as the DOS command language or REXX.

"New" ideas love company

Aside from solving a nagging problem for Apple, UserLand may also help the world (finally) to understand one of the major values of OS/2 -- its support for interprocess communication. While multi-tasking allows you to run several independent processes separately and simultaneously, IPC allows you to run them simultaneously and cooperatively. For example, a smart word-processor could send all capitalized words out to a database to see if they referred to customers listed in the database, or a calendar program could communicate easily with an E-mail program. (That happens now, of course, but rarely between separate products from different vendors!)

RELEASE 1.1

In our March issue, we said that the NeXT interface is synchronous, which is not true. In fact, it’s asynchronous, and works in client-server fashion as X does (only differently, because it sits on Mach). More on this later....
THE SUBJECT IS OBJECTS

At the recent Patty Seybold Technology Forum on object orientation, the un-stated and unanswered question was, Why? What is object-oriented program-ming going to give us, and why should we believe all these promises? We'd like to answer these questions with an analogy -- to the spreadsheet. As we will show, the parallel is not exact, but it provides some useful insights.

When spreadsheets first showed up, few people took them seriously. Users of financial modeling languages pointed out that spreadsheets offered little new, since you could build the same models as effectively and more efficiently using their languages. At the same time, the rest of the world had never built electronic financial models because they didn’t know how, and could not at first see the utility of spreadsheets. In the same way, traditional programmers can’t really see the point of object-oriented program-ming, since they can already build the systems they want using their current tools. And the people with the business models in their heads wouldn’t even try to construct them on a computer.

Fundamentally, the change from financial planning language to spreadsheet was trivial. It was just a different way of presenting and organizing the same logic. The underlying calculations and algorithms were the same. The difference comes only in the abstractions the human user deals with. Yet the ultimate impact of that tiny difference that informs spreadsheets has been huge, albeit among a small if growing class of people.

In the same way, object-oriented systems aren’t all that different under-neath, although they’re organized differently from the user’s point of view, and they may execute differently, just like an interpreted spreadsheet vs. a compiled financial model. The underlying algorithms are the same, as are the data values and the like. (However, in the end, there is usually more information in an object-oriented system than in a traditional one, as we will discuss below.)

Half the fun is getting there

The end result is the same old applications, say the traditionalists. But they never get built, we would answer, just as most financial models never got built before the spreadsheet. Building an object-oriented application is a new process entirely. While a traditional programmer transforms the model in his head into procedural code in the computer, with the object-oriented approach the user transfers the model in his head directly into the computer, perhaps tries it out a few times and makes changes, and lets the computer transform it into executable code. Later on, someone else can come along, understand the model (but not the code) and modify it again.

Ultimately, object-oriented modeling will be like the spreadsheet in its im-pact among a certain class of people (large by software industry standards), those who model business processes rather than financial flows. The tiny differences between traditional systems and object-oriented ones will make all the difference in the world to people’s ability to build operational business systems, just as it did for their ability to build business models.

Of course, the parallel also extends to the objections that are lodged against the new approaches. The traditional modelers disliked spreadsheets
because they were tough to understand and audit. With a financial planning
language, you could look through the code and see what was going on, and
detect inconsistencies. Spreadsheets, because they involve direct manipu-
lation, let users manipulate formulas as they please, with no check on their
logic. Furthermore, because they seemed so easy to use, they were easy to
use wrongly. Careless or uninformed people could build cock-eyed models
that seemed to work fine.

Trust your objects!

In the same way, traditional programmers find object-oriented programming
scary because of the power it gives to people who may or may not know how to
use it right. There's no good way to audit an object-oriented program other
than to run it, and the flaws may not be apparent even then. The ease of
building object-oriented models and applications will attract people who
don't know what they're doing. Just as with spreadsheets, users must still
understand their problem to model it correctly.

Moreover, objects are not as simple as spreadsheets. (In fact, a spread-
sheet is a single class of object, with associated methods, or application
code. Each instance, or individual spreadsheet model, is executed by the
methods of its parent class.) Building object-oriented tools that can oper-
ate in a variety of hardware and software environments and communicate with
other systems is a lot closer to building 1-2-3 3.0 than to building Visi-
Calc or even the original 1-2-3.

In the end, object-oriented programming requires an entirely different mind-
set. Programmers who always know the flow of control of their programs (or
who think they do, or who can follow it by scanning the listings carefully)
have to learn to trust their objects, just as managers have to learn to
delagate to subordinates to get the work done.

The parallels diverge

But the parallel between spreadsheets and object-oriented environments also
diverges more fundamentally. Whereas most of the value in the spreadsheet
business is in modeling tools, and most user companies build their own
models, the divergence between builders and users is far sharper in the
object-oriented world. Indeed, it's as if the market were for pre-canned
spreadsheet models -- which users could modify and fill with their own data
-- rather than for the tool itself. (In fact, with few exceptions, vendors
who have tried to sell 1-2-3 models have failed.) But the object-oriented
world will be different. Much of the value will inhere in the models and
class libraries of pre-canned objects, not in the tools. This means that

- the market will be much more fragmented, in vertical segments;
- the models themselves will be far richer and more specific than most
  spreadsheet models -- and therefore more worth reusing;
- there will be a big business in class libraries, and a lesser one in
devlopment tools.

In the object-oriented world, there will be a lot of "knowledge" in models
and class libraries. It's as if spreadsheet vendors sold named ranges with
relationships rather than just empty sheets. The ability to build these models will still be beyond most end-users. But the ability to use them will be as widespread as the ability to use spreadsheets. Thus when people talk about the benefits of object-oriented programming, they are really talking about the benefits of class libraries and their susceptibility to user modification.

From model to execution

To steal and embellish (in time-honored object-oriented fashion) an explanation provided by Dave Taylor of Servio Logic, there's a hierarchy from modeling to simulation to reality. Modeling lets you describe what a system does and how its components behave, and can be a valuable exercise in itself. It lets you understand things, see relationships, make analogies.

Simulation tests the model. It lets you see the consequences of your assumptions, see if the results are as predicted, detect loopholes and inconsistencies.

Realization runs your system, as an "executable model." Rather than just model or simulate work, it accomplishes it. This process could send the report to Fred, or generate a paycheck? Then let it do so! The system the user builds to model or define his desired procedures, can actually enact them. This of course works only when the work product is in the system, or controllable by it, but that is happening more and more these days. (Groupware applications will be much easier to build this way; see below.) In the same way, a typical relational or network database stores information about real things somewhere else. An object-oriented database frequently contains the things themselves -- designs, text objects/documents, etc.

For example...

Now, let's make this all tangible. Suppose we're a small newsletter business. We buy a set of class libraries that includes subscribers, suppliers, and of course employees, with associated methods for paying employees and suppliers, signing up and billing subscribers, and generating mailing labels monthly. We can change a thing here and there, but basically it's pretty simple. And from our perspective, it's not much different from buying a canned application with a couple of menu-based customization facilities.

But now we expand into the conference business. We buy another class library, this one for conference attendees, with methods for hotel registration, billing, etc. Via multiple inheritance, we now treat our customers as both subscribers and potential conference attendees... We add a rule that requires attendees to be subscribers or to work at the same company at the same address as a subscriber (two per subscriber). It's much easier to add those restrictions, and to integrate object classes from different vendors, than it would be with traditional procedural applications.

Of course, the world still needs some conventions for objects, but they don't necessarily all need to be built in the same language as long as they can send intelligible messages to each other. (The worst that should happen, ideally, is that an object will respond "I don't understand." The actual worst is that an object will misunderstand, and do something wrong...) To modify an object, however, the user may have to know the appropriate language, although developers can put in easy-to-use modification tools.
GOOD DAY, SUNSHINE!

Achieving success in the computer business is tough. Maintaining a leading position is even tougher. The biggest challenge successful companies face is forgetting the ostensible reasons for their success and focusing on what really made them successful -- a change from the status quo. Now their success is the status quo, and they have to leave it behind. That's the scary move that Sun Microsystems is making this month as it announces new low-end hardware that brings it into the desktop market (home of both workstations and pcs). But it takes more than raw power to make a system successful; it takes a Unique Selling Proposition -- something the new system can do that no other one can (in perception at least). To our mind, Sun's hope is to trade on its slogan, "The network is the computer," and bring unique accessibility to it. In short, Sun's task is to do for the network what the Macintosh did for the pc itself. Sun's announcement was as wonderful as expected. Next, it must do something so wonderful it's unexpected.

While Sun's new SPARCstation line will interest technologists as the first mass-market incursion of RISC machines onto desktops, it will interest customers for the software that it will run. We see Sun's initial goal as rendering obsolete the pcs that sit on many engineers' desks: either next to a Sun, to run spreadsheets, write reports and manage tabular data, or instead of a Sun, because the user's firm can't (won't?) pay for the very best. While we don't expect to see many engineers throw away their pcs, they may give them to department secretaries or apprentices, and turn to Sun for new purchases for themselves.

That's all very well, but the big question is what Sun does next. How does it get from engineers' and bond traders' desks to Middle America's desks? The answer is not by selling through Businessland. (That would have been nice, but with luck Sun will be able to win better, non-exclusive terms from other retailers when its products become better established in the mass market.) Unlike NeXT, Sun already has distribution into the commercial world, through 600 direct salespeople and 150-plus VARs of varying capabilities. Like IBM when it first started selling the PC (but in a much-changed marketplace), Sun will sell the SPARCstation to the top of the market, targeting high-salaried users in companies with large computer budgets. The good news is that this is no longer virgin territory, where the concept of computers must be sold; however, the concepts of UNIX and workstations and power must still be sold.

What can Sun's machines do that most users can't already do on a simple pc? Power increments are not enough to overcome the value-subtracted of switching to a new operating system and new applications, or of looking for applications that aren't there. The ability to run existing applications removes a problem but doesn't offer an incentive. There has to be even more. Sun already has a broad range of nicely scalable machines and software but no single, outstanding Unique Selling Proposition.

The network is the computer

We believe Sun's potential Unique Selling Proposition, as its slogan implies, is connectivity...if it can effectively make its benefits tangible. Sun did this with the 386i, making networking integral and easy to use, but only among 386is. The new systems need not just connectivity among them-
selves and between desktop and printer, but among workstations and servers, encompassing a variety of heterogeneous machines, even non-UNIX ones.

Sun's stock in trade should be to provide access to all of these for mere mortals, as Apple did for single-user applications on the Macintosh. But while Apple could limit its attentions to its own proprietary environment, Sun's efforts, if they are to be meaningful at all, must provide seemingly transparent access to a variety of environments. In other words, Sun must atone for others' sins as well as for its own. Part of this effort will come from a broadening and simplification (for users) of the capabilities of Sun's RPC technology; see page 5.

As for applications to exploit this connectivity, we expect to see Sun focus on incipient groupware markets such as electronic publishing and text base management, as well as traditional design and engineering applications and database servers. Sun has the opportunity to do for document management -- not just layout, but sharing and manipulation of content, from e-mail filtering to customized document generation -- what it has already done for engineering. (Interestingly, the graphics co-processors Sun has announced not only provide stunning graphics performance, but also speed up scrolling and display of text.) So far Sun has kept its software efforts under wraps, but Sun software evangelist Dave Cardinal is about to get more aggressive and visible, now that he has built up the staff to support such an effort.

At the high, end, Sun is not just trying to keep DEC out of the workstation market, but going after some of DEC's server business with the SPARCserver line. We suspect that anyone with enough imagination to leave DEC will probably want to take a look at NetFrame as well as the Sun line, but Sun's machines are available now.

Sun is going after the world at large, venturing from a market it owns (albeit one it has fought for hard) into a new market where it's an interloper. Yes, Sun is taking a huge risk. But it would be a bigger risk to stay put.

NeXT wins Businessland

While Sun is pulling out all the stops on its hardware and scaring off competition with user-friendly pricing, NeXT is taking the high road, selling through a single chain (for now) that has committed to support the product. Sun is attempting to create an infrastructure to support its product that may include third parties, clones, customers and the entire UNIX community, while NeXT wants to build a tighter, more controllable edifice with support only from IBM (if that company can get its act together). For NeXT, Businessland is a big win; it has no other way of reaching commercial customers. By contrast, it's a small loss for Sun, and one for which we guess it didn't fight too hard, given Businessland's penchant for special terms.

While NeXT has already signed up Lotus, Aldus and Ashton-Tate (as well as Sybase), its real edge in the Nineties will be its object-oriented development environment and the new applications and class libraries third parties will build with it, as described on pages 7 to 9. Meanwhile, Sun's approach is right for now, and if it can truly make communications transparent it will (properly) flourish over the years ahead. With time and third-party support, Sun will be able to offer the basic benefits of the NextStep environment -- maybe even while adhering to standards. Yes, NeXT has it now -- but perhaps before the world is even ready for it.
Reverse engineering is the key to exploiting and enhancing existing investments in software rather than building new applications from scratch. However, it's not easy. In fact, says one pundit, it's like trying to derive the photograph of a subject from a Picasso painting. The transformations are many, and may reflect the painter as much as the subject. In fact, functional software is likely to be a little more intelligible than works of art (by definition functionless), if only because software does something. But reverse-engineering remains an art, not a science. It's best performed by clever people using powerful tools....

ViaSoft has long offered one of the most powerful reverse-engineering tools around, Via/Insight (see Release 1.0, 87-10). It's a batch system that decipher COBOL code, then presents it to the user in an interactive fashion, allowing him to walk through it with ease, jumping through, say, all uses of a certain data element or showing all the locations where it is defined -- either inconsistently or redundantly. ViaSoft has sold 215 copies of Via/Insight since 1986.

But Via/Insight suffers from its genesis in the mainframe world: The system doesn't help the user edit the code on-line; it's not an incremental compiler such as Saber-C (see Release 1.0, 88-8). In addition, it's a single application that can't easily share its knowledge of program structure with any other application -- a tester, for example, or a database construction kit to build new applications around existing data, or any other software engineering tool. Moreover, it would be nice to be able to combine information about several programs for use with a single tool to detect possible redundancies, inconsistencies or perhaps even synergies.

But that's more a problem of implementation than of capability. Via/Insight's underlying parsing capabilities could serve a variety of tools. Rather than build the parser into such a set of tools, ViaSoft has separated it out in proper client-server fashion. The parser now constitutes a powerful back-end, the "Analytical Engine." The engine decompiles the programs (source and object code) and builds a knowledge base about them -- modules, how control flows from module to module, data declarations, and how and where the data elements are used. In essence, the Analytical Engine builds the sort of knowledge base that the forthcoming repositories from IBM and Pansophic, among others, are designed to hold. (It's been designed to be compatible with those -- with luck!)

Via/Insight has now been repackaged as Via/Center, including the Analytical Engine as a discrete but bundled component, at the same price range ($60,000 to $87,500). In conjunction with the announcement of Via/Center, ViaSoft has also released Via/SmartTest, an interactive tester-debugger that also includes and relies on the Analytical Engine, for $39,500 to $59,500. (You can buy both Via/Center and SmartTest for $99,000.) SmartTest has already won 44 customers since December 1, about half of them satisfied Via/Insight customers, and the rest new.

There's no reason long-run that the Analytical Engine couldn't manage similar information about C programs, using data generated by a Saber-C-based product, or about dBASE programs, using data generated by a version of Clear Software's Clear (Release 1.0, 87-10). The knowledge each of these genera-
tes is language-independent -- and could be transferred from language to language more effectively through the Analytical Engine than through some sort of point-to-point translation which might not optimize the programs for the target language.

But that's all speculation. For the moment, Via/Center remains the kind of product whose very limitations -- mainframe-based, COBOL-oriented -- will endear it to its potential customers.

Cultural divergence

As COBOL is to C, so is Via/Center to Codan (for Code Analyzer), a tool from Implements, a new company founded by Norton Greenfeld, formerly head of software development at Applied Expert Systems. Taking the same approach as Via/Center from a different vantage, Codan builds a repository for C-based applications. Its database lacks the richness of Via/Center's and has no associated tools yet, but it costs only $198. Codan parses C programs and stores the results in a C-tree database that lists who calls whom, where variables are defined, and so on. (See illustration below.) The output for now comprises twelve predefined reports, plus whatever else you can build with the simple report writer included. But they provide the data that a programmer can waste a lot of time finding, and allow him to use his intelligence to far better purpose.

What makes both these systems interesting is that they're not limited to assessing a single program but can manage and cross-reference an arbitrarily large collection -- a company's entire software base, in fact. These tools can be valuable both for reverse engineering and maintenance, and simply for finding out what's there so you don't have to write it fresh but can instead edit it. (Call this a first step towards object-oriented development.)

The Programmer

It's the opening day of the trade show and someone found a serious bug when one particular function is called. The function itself checks out clean, so you ask your codan system to report all calls to that function. The report shows that the function is called for value many times, but one call is for effect, ignoring the critical returned value. **Bug Spotted!**

The Analyst

Customer reaction at the trade show was unanimous. They want to run modules in parallel using side-by-side windows. Until then, you don't have a marketable product. To make this "little" change, you have to find all the global variables and the routines that use them. Your codan system finds and reports complete information for a quick redesign. **A Quick Start!**

---

Release 1.0

10 April 1989
RELEASE 0.5: HARDWARE CAN BE FUN

We recently agreed to see a prototype of the yet-to-be-announced Poget computer, mostly as a favor to an old friend. But our condescension was unwarranted: It's the first one we'd really use ourselves. Roughly the size of a Sharp Wizard -- yes, you need a capacious pocket, but it's definitely smaller than most leather diaries -- it runs true MS-DOS, with a full 24-by-80 screen. We aren't allowed to say exactly how it does this, or to discuss the battery technology, but suffice it to say that among its techniques is power management. That is, the system shuts off its cpu between user keystrokes and system activity, saving power and thereby reducing the weight of the power supply. It would have been a nifty thing to take to the Soviet Union, but there must be some regulation that would forbid it anyway.

RELEASE 0.5: IDEAFISHER -- SOFTWARE THAT'S ALL CONTENT

"We supply the facts; you supply the smarts." Just as a non-intelligent database can provide the perfect complement to an intelligent programmer (page 13), so can a text base of associations provide the memories and nuggets of ideas to be exploited by an intelligent, creative thinker.

We recently saw a quick demo of IdeaFisher, a unique product 12 years and $3.5 million in the making. In a sense, it's an idiot savant: It remembers everything, and understands nothing. It's a giant cross-referenced text base or thesaurus of words and phrases representing concepts and images, enhanced by a series of questions. It contains 25 megabytes of data (compressed down to 14), including 675,000 direct links among its 60,000 elements. Company founder Marsh Fisher, co-founder of the Century 21 real estate chain, hired 200 industry specialists to help build the database. There's a small hierarchy of categories (such as people/things, activities, varieties) but otherwise no regular structure. IdeaFisher also includes QBank, 3000-plus common-sense questions designed to elicit creative thinking or idea extension: "What if you reverse the roles? What about an older version...?" etc.

IdeaFisher simply supplies a thinker with associations he knows and understands, but may have forgotten. In other words, it reminds you of what you already know, but doesn't teach you anything. It says, "If you're thinking about Russia, did you think about Gorbachev, and Red Square, and piroshki, and Communism, and..." That's our list; IdeaFisher probably has hundreds of entries, and further links from them to others.

We can see the tremendous potential utility of IdeaFisher as a memory jogger for creative people. However, we can also see how easy it will be to misposition. It's not smart or intelligent; to the contrary, it's dumb, a giant cross-referenced list of everything its developers could think of. In essence, it's a publishing medium for a set of concepts that would be impossible to publish or use any other way. It's perhaps the cleanest example of a hypertext base we've seen, and the most content-rich and logic-free software package we've ever seen. Users can add their own associations; if IdeaFisher is successful we expect to see a profusion of topic-specific add-in packages that will add concepts and associations specific to particular areas of interest. The base product costs $1795 and comes on six to 20 diskettes -- a perfect use for CD-ROM if we ever saw one.

Release 1.0 10 April 1989
Electronic distribution of software has been a promise (and a sinkhole for venture funding) for years. It's clear most business software won't be shipped around in trucks by the year 2000; it will be delivered over wires or fibers, or at the very least manufactured at the point of demand. But very few vendors making and selling software in the quantities needed to be a market leader have been interested in leading the way, because they understand correctly that these systems will generally reduce costs and inefficiencies rather than increase sales. The smaller vendors who might benefit from wider distribution have lacked the market power to get such systems established. Besides, costs have been high and reliability questionable.

But forget all that. Assume that with CD-ROM a point-of-sale software manufacturing system now costs a retailer just $1995 to install, with no monthly charges other than supplies (blank disks, documentation, etc.). Because the system simply transfers an image from a master, it works with most disk formats.) And assume a vendor with popular products signs up....

This is now the case. Symantec has signed up to sell through on-site manufacturer On Demand Systems -- without great enthusiasm, but never mind. The important thing is to flow enough product through for the system to gain critical mass. "It's a reasonable effort to gain us some additional distribution through ComputerLand," says Symantec ceo Gordon Eubanks. "Overall, the costs are somewhat less, but the dollars we get are the same when the product sells through." Symantec saves because it doesn't have to press disks even though it does supply packaging and documentation to go with software manufactured on site. Symantec gets its cash later, only when the software sells, but avoids the risks of stock-balancing -- a polite term for taking returns. (Incidentally, widespread adoption of such systems should change the current practice of channel-stuffing at quarter's end.) Symantec will also get faster sales reports from the 183 ComputerLands now signed up.

Meanwhile, other large vendors may see the appeal of electronic distribution, if only for their lesser titles. For every dBASE, there are multiple skus (stock-keeping units, or discrete titles) of MultiMate and RapidFile. For every 1-2-3, there's Manuscript, and Symphony, and Freelance.

Resellers, of course, have always liked the idea because of cash flow considerations, but wanted brand-name software to make it interesting. The store never runs out of stock (although it may lack manuals); more importantly, the store takes no inventory risk, and is better able to carry slow sellers or new items. In fact, the greatest beneficiaries are likely to be off-mainstream UNIX products and expensive OS/2 products -- especially if UNIX continues to come in many varieties. The year 2000 may arrive early.

Nonetheless, nothing will overcome the basic fallacy behind the traditional pitch for electronic distribution, that it will help expand a software vendor's market: There is not a distribution bottleneck, but a demand bottleneck. Simply making a given package available won't ensure that it sells through. Wider distribution may help sales, but not much. (And retailers will gladly stock most products for which there's widespread demand.) There is still no substitute for the knowledgeable salesperson who says, "I use it myself..." or even discuss a product's features intelligently. The shallowness of the one-liners vendors use to prepare salespeople is readily apparent whenever the customer is interested enough to ask a second question.

Release 1.0

10 April 1989
RELEASE 1.2: IMPLEMENTATION SUPPORT

Last summer we wrote about implementation support, the use of "presentation" tools in interactive group sessions to foster communication and group understanding. Decisions are relatively easy to make, we argued, but hard to get a group to carry out. Implementation support sets the group marching in the right direction....

Bernie De Koven, a premier implementation-support practitioner, gave us a call the other day with some thoughts about the practical realities of using any system that involves personal interaction rather than simple data analysis or manipulation. When does implementation support work best? When people really want implementation, he answers. De Koven says he’s discovered that 80 percent of meetings are basically political; their purpose is neither two-way clarification of ideas nor consensus. Instead, they’re focused on information dissemination rather than information generation, on posturing and on public grooming of the organization chart.

The software he uses isn’t remarkable; what distinguishes his work is the way he uses software to foster communication and clear presentation of ideas. In fact, he notes, MacDraw II works better than a formal presentation product because it highlights content over form. (We suppose that’s why those academics who draw their charts live with markers on an overhead are so convincing.) In the same vein, De Koven has found that LCD screens displayed through overhead projectors are effective precisely because they are more familiar and less intimidating than formal slide presentations. The overhead looks clunky and low-tech, he says, although underneath it’s high-tech. The use of a live computer rather than slides allows participants to work together to construct their own model rather than comment on finished, unmalleable slides.

Productivity: From persons to groups

Elsewhere, products that began as personal tools are acquiring a more explicit group orientation (but will nonetheless require application of human skills such as De Koven’s). Among them are IZE and Verity’s TOPIC, with network versions forthcoming. But there’s more to group support than network access. Another product, for PCs rather than the Macintoshes De Koven favors, is Decision Pad from Apian Software (see Release 1.0, 88-8), one of those from-nowhere start-ups that hangs in there with a good product and lives to produce a second release. Decision Pad’s second release is more explicitly focused on group decision-making, whereas the first version was a personal decision-analysis (as opposed to data-analysis) tool that could be shared. DP’s new version allows for "voting" and reconciles individual preferences within a group.

This is the kind of task at which a product such as Neuron Data’s Nextra (see Release 1.0, 89-2) should also flourish. It allows people to define items along a set of axes, and then maps the result into a two-dimensional map. It calculates fast enough that new information can be added on the fly to redraw the map and clarify perceptions. Moreover, different people’s different perceptions can be clarified and resolved, if appropriate. Or at least people can come to an understanding of their disagreements.

In short, to paraphrase Eliza Dolittle, what makes groupware isn’t what it does, but how it is treated.

Release 1.0

10 April 1989
THE ED\textsc{wards}

presented by John Dvorak

The prize: An EDdy bear to each winner!

The VaporWare ED\textsc{ward} (presented by Stewart Alsop):
Winner: Lotus Development Corporation, accepted by Frank King.

The best-name ED\textsc{ward}:
Contenders: Adobe for Streamline; Channel Computing for Forest & Trees; Funk Software for All\textsc{ways}; Apple Computer for WildCard (withdrawn); Network General for The Sniffer.
Winner: Aldus for Persuasion, Paul Brainerd.

The Forestry ED\textsc{ward} for paper consumption:
Contenders: Apple; 3Com; IBM; Novell; Sun Microsystems.
Winner: Microsoft, accepted by Pam Edstrom of the Waggener Group.

The Ester Dysan of Adventure Moldings ED\textsc{ward}:
Contenders: Dan Flystra of VisiCorp and Frontline Systems; Dan Brickland; Steve Jobes of NeXT; Vadim Who? of Clear Software; Enzo Toressi; Safi Quereshey of AST Research; Cunningham Communications; Vinod Koshla; Avery Moore of CompuCom (and anyone whose name we misspell -- oops!).
Winner: Irene Greif (not Ilene Grief) of Lotus.

The Go-Back-to-Cleve\textsc{land} ED\textsc{ward}:
Contenders: IntelliCorp; NeXT; dBASE/Dbase; Powerbase/Power Base; ComputerLand; ED\textsc{venture} Holdings/RE\textsc{lease} 1.0 (withdrawn).
Winner: Interbase/InterBase, Jim Starkey.

The That's-My-Best-Friend? ED\textsc{ward}:
Contenders: Phil Estridge (the original); Art Einstein of Lord Einstein; Joe Canion of Compaq; Louis J. Doerr; James Manzi; Samuel J. Kaplan of GO Corp.; Willard E. Peterson of WordPerfect; Katharine Branscomb of IntelliCorp; Catharine Carhart of Orion.
Winner: Linwood "Chip" Lacy of Ingram Micro D.

The Mike Tyson Pugilistic Enthusiasm ED\textsc{ward}:
Contenders: Ed Esber of Ashton-Tate; Philippe Kahn; Larry Ellison; Dave Norman; Michael Shane; Open Software Foundation.
Winner: Bill Joy of Sun Microsystems.

The Mike Markkula/Gordon Bell ED\textsc{ward} for judicious mobility:
Contenders: Spencer Leyton; Ron Posner; David Blumstein; Graham Beachum.
Winner: Vern Raburn of Microsoft, Lotus and Symantec, among others.

The ED\textsc{ward} for sartorial splendor:
Contenders: Ben Rosen; Dan Bricklin.
Winner: Mitchell Kapor of ON Technology.

Release 1.0 10 April 1989
The EDward for physical splendor:
Contenders: [censored]

The EDward for IBM alumni:
Contenders: Frank King; Graham Beachum; Gideon Gartner; Don Casey; Morris Taradalsky; Art Goldberg; Glenn Henry; Mike Maples; Sam Albert. Winner: Andy Heller, now associated with Kleiner Perkins.

The EDward for Apple/Sun footsoldiers:
Contenders: Joe Roebuck; Jim Bean; Jim Davis; Chuck Berger; Chuck Boesenberg (MIPS). Winner: Wayne Rosing.

The EDward for achievement by a white male:
Contenders: Vadim Yasinovsky; Fred Gibbons; Peter Norton; Dan Bricklin. Winner: Arthur Einstein of Lord Einstein O'Neill.

The Mysteryware EDward:
Contenders: ON, GO, Saros, Calypso. Winner: Asymetrix, Paul Allen.

The EDward for best ROI on a consulting contract:
Contenders: Jim Manzi; Rich Shapero; Ron Posner; K.C. Branscomb; John Merson; Gordon Bell; Jon Lazarus. Winner: Ed Esber of Ashton-Tate.

The Pax Machina EDward for cross-cultural understanding:
Contenders: Bill Lohse; Marty Mazner; John Sculley; Victor Alhadeff; Tandy for its acquisition of GRID; Microsoft for its investment in SCO; Wil Jones of Mesa View Ranches (and founder of Multimate). Winner: Bruce Katz of Rosewood Stone Group, founder of The Rockport (shoe) Company.
RESOURCES & PHONE NUMBERS

Bernie De Koven, Himself, (415) 857-1757
Bill Ray, Apian Software, (415) 851-8496
Vadim Yasinovsky, Clear Software, (617) 232-9788
Marsh Fisher, Fisher Idea Systems, (714) 474-8111
Norton Greenfeld, Implements, (508) 358-5858
Dan'l Lewin, Conall Ryan, NeXT, (415) 424-0200
Randy Rand, Paul Salzinger, On Demand Systems, (415) 434-2122
Stav Prodromou, Gerry Purdy, Poget Computer, (408) 737-8100
Sesha Pratap, Saber Software, (617) 876-7636
Dave Taylor, Servio Logic, (415) 748-6200
Patty Seybold, (Seybold's) Office Computing Group, (617) 742-5200
Wayne Rosing, Dave Cardinal, Sun Microsystems, (415) 960-1300
Gordon Eubanks, Symantec, (408) 253-9600
Dave Winer, UserLand, (415) 854-0662
Kent Petzold, ViaSoft, (602) 952-0050

GONE FISHING

Thanks to everyone who has written or called us with feedback on the Forum. I’m sorry I haven’t thanked you personally, but I’m rushing off to Moscow with no time to spare (in search of further excitement). I will be exploring the use of computers in the Soviet Union with Gary Chapman of Computer Professionals for Social Responsibility and CPSR director Sherry Turkle (author of The second self).

The wonderful EDventure staff will still be here to take any further feedback on the Forum, answer questions about Release 1.0, etc. See you in May!

-- Esther Dyson

Future topics:

- Computers in the USSR (also known as "Our spring vacation");
- Visual programming languages: Modeling and simulation.
- A composition engine.
- Active and passive objects.
- And much more... (If you know of any good examples of the categories listed above, please let us know.)
# RELEASE 1.0 CALENDAR

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Location</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 17-19</td>
<td>Macintosh business conference &amp; exposition - New York City.</td>
<td>New York City</td>
<td>Sponsored by Cambridge Marketing. Keynote by John Sculley; also MacUser’s Marty Mazner and Fred Davis and the MacUser Eddy Awards (not to be confused with our EDDy bears!). Call Brice Hereford, (800) 262-3378 or (617) 860-7100.</td>
</tr>
<tr>
<td>April 18</td>
<td>Oracle at the analysts'</td>
<td>New York City</td>
<td>Sponsored by New York Society of Security Analysts. Larry Ellison speaks. Contact: Lourdes Figueroa at (212) 344-8450.</td>
</tr>
<tr>
<td>April 18-20</td>
<td>Connect '89 - Boston.</td>
<td>Boston</td>
<td>Sponsored by Cahners Exposition Group with Larry DeBoever and Dale Kutnick. For MIS managers mostly. Contact: Dave Sell at (203) 964-0000.</td>
</tr>
<tr>
<td>April 20-21</td>
<td>COMPUTER ACCESS AND USE FOR DISABLED PERSONS - Newington, CT.</td>
<td>Newington, CT</td>
<td>A topic of increasing social and business impact. Sponsored by Trace R&amp;D Center (see Release 1.0, 87-6). Contact: Gregg Vanderheiden, (608) 262-6966.</td>
</tr>
<tr>
<td>April 30-May 4</td>
<td>CHI '89: Conference on human factors in computing systems</td>
<td>Austin</td>
<td>Sponsored by ACM/SIGCHI and a host of other groups. Contact: Claudia Raun, MCC, (512) 338-3798.</td>
</tr>
<tr>
<td>May 1-4</td>
<td>Computer viruses '89 - Chicago.</td>
<td>Chicago</td>
<td>Against them, of course. Sponsored by Computer Security Institute. Contact: June Kolikoff at (508) 393-2600.</td>
</tr>
<tr>
<td>May 2-4</td>
<td>Speech Tech - New York City.</td>
<td>New York City</td>
<td>Produced by Media Dimensions, publisher of Speech Technology magazine. Everything from voice mail and audiotex to neural networks and speech recognition. Call Stan Goldstein, (212) 533-7481 or -3943.</td>
</tr>
</tbody>
</table>

*Release 1.0* 10 April 1989
<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Location</th>
<th>Description</th>
<th>Contact Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 7-9</td>
<td>Culpepper CEO Forum - Atlanta. Meet fellow software ceos and share experiences, focusing on sales and sales management.</td>
<td>Atlanta</td>
<td>Contact: Warren Culpepper, (404) 668-0616</td>
<td></td>
</tr>
<tr>
<td>May 7-10</td>
<td>Software Maintenance: Share the vision - Atlanta. Annual meeting and conference of the Software Maintenance Association.</td>
<td>Atlanta</td>
<td>Contact: Judith Golub, (415) 941-5623.</td>
<td></td>
</tr>
<tr>
<td>May 7-10</td>
<td>*IBM computer services and consultants executive conference - Phoenix.</td>
<td>Phoenix</td>
<td>Contact Norm Urquhart, (914) 749-3154.</td>
<td></td>
</tr>
<tr>
<td>May 9-11</td>
<td>OpticalStorage89 - San Jose. Back for the fourth time by popular demand.</td>
<td>San Jose</td>
<td>Sponsored by DISK/TREND and Freeman Associates. Call Darlene Plamondon, (408) 554-6644.</td>
<td></td>
</tr>
<tr>
<td>May 9-11</td>
<td>Second international conference on computer-assisted learning - Dallas.</td>
<td>Dallas</td>
<td>Sponsored by U of Texas’ Computer Learning Research Center. Contact: Janet Harris, (214) 690-2204.</td>
<td></td>
</tr>
<tr>
<td>May 11</td>
<td>Massachusetts Software Council spring membership meeting - Newton, MA.</td>
<td>Newton, MA</td>
<td>Sentry’s Software Market survey, new Board of Regents chairman Paul Tsongas on higher education, demos, etc. Contact: Joyce Plotkin at (617) 437-0600.</td>
<td></td>
</tr>
</tbody>
</table>

Release 1.0  10 April 1989
see Release 1.0, 89-2), Henry Ancona of DEC, Bruce Hasenyager of Merrill Lynch, Bill Murphy of H-P. Contact: Pam Davis, (617) 354-5555.

May 15-18

May 15-18

May 16-18
CEPS '89 - Chicago. Corporate electronic publishing systems. Sponsored by Cahners Exposition Group. With Joe Gugilemi, IBM; Yuri Rubinsky, SoftQuad; Raymond Kammer, NIST; Dan Cheifetz, Odesta; Jim McNaul & Art Coles, Xerox; others. Contact: Kim Ray, (203) 964-0000.

May 16-18
Quality Week - San Francisco. Quality assurance (and all it implies) for software. Sponsored by Software Research. Contact: Emily Chais, (415) 957-1441 or (800) 942-SOFT.

May 19-20

May 21
Warning! Annual Bay to Breakers Race - San Francisco. Sponsored by the San Francisco Examiner.

May 21-24

May 21-24
Expert systems and the leading edge in productions and operations management - Hilton Head Island, SC. Sponsored by AAAI, the Operations Management Association, others. Contact: Libby Shropshier, (803) 777-2231.

May 21-24
Artificial intelligence and intelligent tutoring systems - Orono (near Bangor), ME. Sponsored by AAI and the University of Maine. With Donald Michie of Edinburgh, Alen Shapiro of Action Software (yield management), others. Call Jim Toner or Danny Kopec, (207) 581-4092 or 581-3522.

May 23-25

May 29-June 2
Ninth international workshop on expert systems and their applications - Avignon, France. Contact: Jean-Claude Rault, 011 331 4780-7000.

Release 1.0
10 April 1989

May 31-June 2 *SIGMOD - Portland, OR. User interfaces to dbms systems and some more esoteric management of data topics. Sponsored by ACM/SIGMOD. Keynote: Esther Dyson! Contact John Bruno, (503) 629-8383.

June 5 Toshiba Micro Industry Golf & Tennis Celebrity Tournament - Industry Hills/Los Angeles. With Tom Selleck, others; funds raised go to the Starlight Foundation for terminally ill children. Contact: Dave Freeman, (714) 558-8813.

June 5-8 *AIIM show & conference - San Francisco. "Discover the power of imaging." Keynote: astronaut Wally Schirra. Sponsored by the Association for Information and Image Management. Contact: James Breuer, (301) 587-8202.

June 6-8 International expert systems conference and exhibition - London. Sponsored by Learned Information. Contact: Jean Mulligan, (011) 44 865 730275; fax, 865 736354.


June 12-16 Summer USENIX technical conference - Baltimore. Tutorials on networking, environments, development tools, window systems, etc. Contact USENIX at (213) 592-1381 or 592-3243, or John Donnelly, (303) 499-2600 (exhibits), or Neil Ground Water, (703) 883-1221 (papers).

June 13 Effective marketing for a small computer and software services company - Chicago. Sponsored by ADAPSO. Contact: Kelly Bailey, (703) 522-5055.

June 13-16 Second international conference on artificial intelligence and law - Vancouver, British Columbia. Sponsored by ACM. Contact Edwina Rissland, (413) 545-0332 or Michael Mills, (212) 554-3180.

June 18-22 International joint conference on neural networks - Washington, DC. Sponsored by both IEEE and the International Neural Network Society, in an amicable compromise. Pat-

Release 1.0 10 April 1989
terned after previous events; you'll recognize all the right speakers, including Kosko, Sejnowski, Widrow, Grossberg, Carpenter, Hecht-Nielsen, Amari. Contact Nomi Feldman, (619) 453-6222.

June 19-21  Videotex Industry Association annual conference - San Francisco. E-mail, software, services, etc. Call Robert Smith, (703) 522-0883.

June 20-22  *PC Expo & National CASEcon - New York. Sponsored by PC Expo. Keynote by Dave House of Intel. Call Steven Faher, (800) 444-EXPO or (201) 569-8542. To be followed by....

June 24  *Intergalactic user group conference - New York City. For user group officials and other involved people. A follow-up to PC Expo. Call Joe Rigo, (212) 249-6418.

June 21-23  *Programming language design and implementation - Portland, OR. Sponsored by ACM SIGPLAN. Contact: Bruce Knobe, (508) 879-2960 x3376 or knobe@S56.Prime.com.

June 25-28  Twelfth international conference on R&D in information retrieval - Cambridge, MA. Sponsored by ACM SIGIR and other groups. Contact: Bruce Croft, (413) 545-0463.

June 25-28  *Twenty-sixth Design Automation Conference - Las Vegas. The big event for designers and vendors to designers. Sponsored by ACM. Contact: Pat Pistilli, (303) 530-4333.


July 17-21  *CASE 89 - London. Sponsored by Index Technology and a host of academic groups, including London's Imperial College. Contact: Elliot Chikofsky, (617) 494-8200, x 1989.

July 23-25  *Sun Expo '89 - Santa Clara. Sponsored by Sun Observer magazine for Sun users and resellers. If there is a hot UNIX box, this is it. Come see for yourself and meet the growing Sun community. Contact: Clayton Peters, (408) 296-7111 or (800) 828-EXPO.


Release 1.0  10 April 1989
July 31-August 4 *SIGGRAPH '89 - Boston. Sponsored by the Association For Computing Machinery. The annual festival for visual, graphical thinkers. Contact Cindy Stark, (312) 644-6610.

August 1-3 Comdex Asia/Pacific - Sydney, Australia. Sponsored by Interface Group. Contact: Cheryl Delgreco, (617) 449-6600.

August 6-9 DB2 Users Group annual meeting - Chicago. Sponsored by International DB2 Users Group. Contact: Samantha Sipowicz, (312) 644-6610.

August 9-10 *Macworld Expo - Boston. Contact: Peggy Kilburn, (617) 326-9955.

August 9-11 *Conference on object-oriented dbms applications - Santa Clara, CA. Sponsored by Santa Clara University. Contact: Mohammed Ketabchi, (408) 554-2731 or mketabchi@scu.bitnet.


August 22-26 *IJCAI-89 - Detroit. The international version of AAAI. Sponsored by the American Association for Artificial Intelligence. Contact: Claudia Mazzetti, (415) 328-3123.

August 23-25 TechDoc '89 - San Jose. "Publishing in the '90s...the art of publishing and the science of information management." Technical documentation and all its ramifications, including hypertext. Sponsored by Graphic Communications Association. Contact: Patti Hill, (703) 841-8160.

August 24-September 1 *Eleventh World Computer Congress - San Francisco. With a focus on tools and application software this year; in the U.S. for the first time in 24 years. Sponsored by 46 IFIP member societies. Call Nancy Dana, (303) 696-6100.

September 6-9 Breakaway '89 - Orlando. Sponsored by ABCD, the microcomputer industry association (mostly dealers). Contact: Deborah Keating, (312) 240-1818.

September 7-10 Comtec '89 - Singapore. Regional micro exhibition. Sponsored Microcomputer Trade Association of Singapore. Contact: Yong Mee Hiong, Singapore 2913238; fax 2965384.

September 12-14 NetWorld - Dallas. Managed by H.A. Bruno. Contact: Adam Torres at (201) 569-8542 or (800) 444-EXPO.

September 13-15 *Conference on Computer-Supported Cooperative Work - London (Gatwick). Inspired by the successful U.S. events, but likely to focus even more on social issues. Sponsorship unclear, but we're trying to find out! Contact: Lorna Meek, 011 44 (753) 73232.

Release 1.0 10 April 1989
<table>
<thead>
<tr>
<th>Date</th>
<th>Event Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 18-20</td>
<td>DataStorage - San Jose. Sponsored by DISK/TREND and Freeman Associates. Call Darlene Plamondon, (408) 554-6644.</td>
<td></td>
</tr>
<tr>
<td>September 20-23</td>
<td>*Seybold computer publishing conference - San Francisco. Sponsored by Seybold Publications. The usual extravaganza, expanded from desktop publishing to include all electronic publishing. Contact: Kevin Howard, (213) 457-5850.</td>
<td></td>
</tr>
<tr>
<td>October 1-4</td>
<td>*ADAPSO Management Conference - Orlando. Mingle with your peers (and Disneyworld’s nearby just in case). Contact: Sheila Wakefield, (703) 522-5055.</td>
<td></td>
</tr>
<tr>
<td>October 2-6</td>
<td>*OOPSLA - New Orleans. Sponsored by ACM/SIGPLAN. Come meet your fellow objects and share procedures. Send a message to Carole Mann, (407) 628-3602.</td>
<td></td>
</tr>
<tr>
<td>October 2-6</td>
<td>CD-ROM Expo - Washington, DC. Sponsored by IDG Conference Group. Contact: Mitch Hall, (617) 329-8090.</td>
<td></td>
</tr>
<tr>
<td>October 2-6</td>
<td>Interop 89 - San Jose. Interoperability made tangible, with tutorials, discussions, product demos and pitches. Sponsored by Advanced Computing Environments. Contact: Mark Belinsky, (415) 941-3399.</td>
<td></td>
</tr>
<tr>
<td>October 3-5</td>
<td>PC Expo - Chicago. Sponsored by PC Expo. Contact: Steven Faher, (800) 444-EXPO or (201) 569-8542.</td>
<td></td>
</tr>
</tbody>
</table>

*Release 1.0* 10 April 1989

November 5-10  *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; Mike Dolhi or Adam Young at Scribe Systems, (412) 281-5959 for SIGDOC 89. (How about a joint committee on standards for the use of apostrophes?)

November 13-17  *Comdex - Las Vegas. Also including MACdex. Contact: Jane Wemyss at (617) 449-6600 or (800) 325-3330.

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.


January 28-31  **EDventure Holdings PC (Platforms for Computing) Forum - Tucson, AZ. Sponsored by us! Note that it's earlier this year. Contact: Daphne Kis, (212) 758-3434.

February 20-22  Computer science conference - Washington, DC. "Cooperation" is the theme, among processing units, technologies, disciplines. Sponsored by ACM. Contact: Barbara Kyriakakis, (703) 323-2318.

March 5-9  *Seybold Seminars '90 - Boston. ...moves east. Call Kevin Howard, (213) 457-5850.

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 10 April 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. 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? ________________________________

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