Inside this issue

TECHNOLOGY & INDUSTRY

EUROPEAN DEVELOPERS and distributors recently met to discuss solutions to piracy problems there.

(see NEWS folder)
***

"HEX, DRIVES, AND VIDEOtape," this month's CD, is packed with new and interesting goodies for developers, including a kit to help you fight piracy.

(see NEWS folder)
***

DINKER IS THE NEW dynamic linker now available for MacApp, to help developers add extensibility to their applications. It is included on several recent E.T.O. CDs.

(see NEWS folder)
***

WHY DEVELOP for the Macintosh? Our new brochure outlines many reasons that developing on this platform is good business.

(see NEWS folder)
***

APPLE IS ENTERING new distribution channels, and has recently signed an agreement with Sears to begin selling some Macintosh products in the consumer channel.

(see NEWS folder)
***

THROUGH THE It Shipped! program, you can announce new and revised third-party products. This article tells you what products recently shipped and how to participate in the program.

(see NEWS folder)
***

APPLE IS SPONSORING developers conferences in several countries during the upcoming months, including the Benelux area and France. These conferences will focus on a variety of development topics and issues.

(see NEWS folder)
*****

DAL TOOLKITS and servers, MacsBug, and other hot products were recently released by APDA, and have become available to developers withing the last several weeks.

(see NEWS folder)
*****

TOG SAYS that creating software that sells entails appealing to the Parent, Adult, and Child in all of us. If your view of the person who buys and pays for your software is purely one of an adult with a data processor for a mind, you need to expand your outlook.

(see TOG'S COLUMN folder)
*****

BUSINESS & MARKETING

APPLE WILL participate in various trade shows during the coming months. If there is a show you want other developers to know about, list it in GetNextEvent.

(see BUSINESS & MARKETING folder)
*****

HAVING THE RIGHT THING to say for the right occasion is a key to marketing your company through public speaking.

(see BUSINESS & MARKETING folder)
*****

LOTUS SAYS that moving your product to a new platform—the Mac way— is a very special process, indeed. Yuko Takagi of Lotus Development Corporation describes that company’s experience.

(see BUSINESS & MARKETING folder)
*****

MACINTOSH SOFTWARE sales out-performed the overall software market in 1991.

(see BUSINESS & MARKETING folder)
*****

Apple/DEC European Distribution Agreement

Apple Computer Europe, Inc. and Digital Equipment Corporation, Europe, have released the details of the distribution agreement that results from the letter of intent they announced in November last year. The agreement will enable Digital subsidiaries in Europe to provide Macintosh computers and peripherals to their large accounts with- in the context of integrated solu- tions involving Digital as well as Apple products. At the same time, Apple's dedicated reseller network in Europe (APPLECENTRES) will be able to offer Digital VAX and RISC-based servers as well as PATHWORKS interconnect products and a wide range of desk– top services.

The Europe-wide agreement will address the needs of customers who require large-scale, enterprise-wide integrated solutions. It is also the intention that some Digital "Complementary Solution Organizations," which are independent resellers or developers, will port their Digital client/server solutions to the Macintosh platform and license their products to APPLECENTRES. In the same way, Apple will encourage its developers to port their client/server applications to the Digital platform. The Digital Equipment Enterprise (DEE) sales entities, which focus on small- to medium-sized businesses, are not included in the agreement.

Japan Macworld Highlights Opportunities

Earlier this year, more than 80,000 people went to the largest Macworld Expo ever. Were you one of them? Probably not, unless you made the trek to Tokyo last February. Macworld Expo Tokyo outperformed last January's San Francisco Macworld by about 20,000 attendees.

This is more than just an impressive fact; it is a clear signal to developers worldwide about the level of interest—and opportunities—in Japan's Macintosh market (even more so when you consider that this was only the second Macworld ever held in Japan). The installed base of Macintosh computers in Japan is currently 300,000, a figure that Apple Japan expects to grow to 500,000 within a year. The record-breaking attendance at the Tokyo event makes Apple Japan's prediction look very plausible and emphasizes the growing importance of the Japanese market to developers worldwide.

Macworld Expo Tokyo, which was held February 20–22, 1992, came on the heels of Apple Japan's third annual Apple Developers Conference in January (see the "Japanese Developers Conference" news item in the March 1992 issue of *Apple Direct*). One strong theme of the show was its emphasis on new products.

Attendees showed great interest in QuickTime (which has not been very visible in Japan to-date). John Sculley demonstrated this product in his presentation, emphasized the importance of multi-media (especially in educational software), and spoke about hand- held Personal Digital Assistants (PDAs). Eikoh Harada, Apple Japan's director of marketing, demonstrated how easy it is to take an Apple television commercial in QuickTime movie format and re-edit it for a Japanese audience.

Another theme at the show was the continuing evolution of Japanese Macintosh system software. A Kanji TrueType upgrade has been announced, and Apple later plans to ship its first "world ready" release of system software, which will allow rapid localization of system software for international markets. The Japanese market will therefore have access to System 7, QuickTime, and any other system software technologies that are available to the U.S. and European markets.

Japanese developers and their products were much more visible than they have been in the past. According to Suzanne Forlenza, Apple's international evangelist, in Japan software is usually promoted by the company that distributes it; but at this expo, more developers personally showed their products. The selling of products on the show floor hasn't been done at Japanese trade shows, she said, but one company, MacLand, broke with tradition and was rewarded with long lines of attendees waiting to buy products. Overall, she saw "an incredible interest in software" at the show from attendees as well as presenters; she also feels that the show (and the Japanese market) is growing in professionalism, in terms of both the volume and the quality of business done there.

The expo was not without a display of experimental, cutting-edge technology. David Nagel and members of the Apple Advanced Technology Group (ATG) demonstrated several works in progress.

One demo showed a handwriting-recognition system integrated into a pen-based Macintosh. Another showed the "Casper" voice-recognition technology, which John Sculley showed on "Good Morning America" earlier this year. In this demo, the user opened a calendar and added new entries by using voice commands only.

Blueprint for the Macintosh

Details from the System Software Forum

by Gregg Williams,
Apple Direct Staff

Has it really been eight years since the world first saw the Macintosh 128K? Despite the troubles of the day, life was a lot simpler then—you had one box and one configuration, and you pretty much knew where you stood.

Not anymore! In eight years of struggle and growth, we've seen about a dozen Macintosh models available (many of which have hardware expansion of some sort), color and multiple-screen cap- abilities, EtherTalk, cache cards, MultiFinder, floating-point coprocessors, networking with computers from other companies—the list seems endless.

Certainly the Macintosh universe is growing, but is it growing well? Is it overextending itself? Will it be supplanted by younger, more-state-of-the-art platforms? Will it still be a competitive platform in 1995?

The answers to these questions are *yes, no, no,* and *you bet,* respectively. Late last year, we published an overview document called Blueprint for the Decade (Apple Partners and Associates received it in their December 1991 monthly mailing). In it, we described the mid- to long-range goals that have guided our past decisions and will continue to do so. On March 3 and 4, 1992, Apple hosted its first "System Software Forum," in which Apple management shared these same goals with about 80 key reporters and industry analysts.

This article attempts to highlight the most important points in the Blueprint document and give you additional details that we made public at the Forum. By doing so, we hope that you will be able to make better business decisions. If you are an Apple Partner or Associate, you should consider attending the 1992 Worldwide Developers Conference (May 11–15 in San Jose, CA), where we will share further details of our plans.

If You Don't Read Another Section of This Article…read this one. We want to make sure you get the following key messages:

•*"The Macintosh is our core business."* That statement, made by Roger Heinen, Apple vice president and general manager of the Macintosh Software Architecture Division, says it all. As you will see in the rest of this article, we have extensive plans for strengthening and extending the Macintosh architecture.

•*System 7 will be the foundation for future system-software growth*. Applications that are designed to the System 7 API (application program interfaces) will run on future versions of System 7, as well as on PowerOpen (the future descendant of A/UX) and Taligent.

•*The Macintosh will support multiple processor families,* including both the Motorola 680x0 family and a new generation of RISC (reduced-instruction-set computer) processors, and we will make the transition path smooth for users as well as developers.

•*Apple is committed to ensuring the success of both your business and your products.* This includes getting you better tools (including tools for development in the Windows platform) and finding new ways of getting your products to customers (through our new Software Access Initiative).

The rest of this article gives details of these and other important market and technical issues. You may want to browse through the topics and read the ones of most interest to you.

1991—A JOINT SUCCESS STORY

In 1990 and 1991, Apple gained market share by fundamentally changing the company, its products, and the way it does business. Apple's market share, worldwide, went from 9 percent in 1990 to somewhere between 15 and 20 percent in 1991. According to Dataquest, we reached an installed base of 7.3 million Macintosh computers, and we experienced more than 60-percent unit growth in 1991. The Macintosh is now regarded as one of the major personal computer platforms. (And Apple's partnerships with IBM haven't hurt things, either.)

We also have indications that our strategy has helped make you more successful. Third-party-software sales have grown significantly in the last year: 47 percent over last year for the Macintosh market, versus only 24 percent in the rest of the industry. (These figures are from the Software Publishers Association. For more details, see this issue's "Market Insight" column.)

MEETING TOMORROW'S MARKETS

The changing marketplace presents us with new challenges. Ease of use will become increasingly important as the personal-computing industry tries to reach a larger market that includes a wider variety of customers than it has dealt with so far. We at Apple feel that our experience in ease-of-use issues positions us to provide a much better computing experience than our competitors can.

We believe that the personal-computing industry is becoming a worldwide market, and our goal is for the Macintosh to become the preferred computer everywhere in the world, regardless of the (human) language and writing system used. We want people to have their Macintosh computers running in their native language, and we want Macintosh computers to handle multiple languages at the same time.

Product Marketing Strategy. We have an aggressive plan for continuing our industry leadership. We recognize the importance of software in the market, and we plan to give it the attention it deserves—by developing new ways of packaging, pricing, supporting, and distributing software. As Rick Spitz, senior director for Macintosh System Software, puts it, "We will concede no territory in which we are currently a leader." He adds that we have also identified several new areas where Apple will establish a leadership position.

Our system-software strategy will be more modular, starting with System 7 and continuing with "relentless frequent delivery" of add-on modules—such as QuickTime, the Open Collaboration Environment (O.C.E.), AppleScript, and other future technologies. Future "reference releases" of System 7 (see below) will bring all of these new technologies into one common base for platform consistency and user convenience.

Finally, we recognize that to be successful, Macintosh products must work in mixed computing environments—that is, with hardware and software in use today. Also, Macintosh developers must be able to create products for multiple platforms. We plan to give you the tools you need to do so.

System Software Directions. We will continue to extend the lead of Macintosh in ease of use, graphics, and networking with important technology introductions in each area.

For example, in the area of graphics, we are working on a new imaging model for Macintosh that will fundamentally advance the state of the art for dealing with color and the different writing systems used around the world. We also plan to make the Macintosh unequaled in the areas of collaboration and customization.

For the Macintosh to be successful around the globe, we realize that system software should be immediately available worldwide and that it should be as easy to use in, say, the Chinese language as it is in English.

Later this year, we will distribute the first Reference release of System 7. It will be one version of system software that will work on any Macintosh, in any (human) language that Apple supports. (Among its other features, this Reference release includes built-in support for 2-byte characters, which is needed for languages that need 2 bytes of storage for each character or glyph and for right-to-left writing systems.)

The Reference release of System 7 will be one set of code that will not change for 12 to 18 months (what a concept!). Users may drop system extension files into the System Folder to extend System 7 with new features or to adapt it to a new writing system (for example, French or Hebrew) or Macintosh model. (Apple might ship, say, an extension file that, when added to the System 7 Reference release, will allow System 7 to run on the Macintosh Quadra.) Following this new Reference release, we will support new Macintosh models by adding drop-in “CPU Support Extensions,” eliminating the need for a new version of System 7 every time we introduce a new model of the Macintosh.

We feel that collaboration will be an important part of personal computing in the years to come, and our Open Collaboration Environment, discussed in greater detail below, will give the Macintosh an advantage in this area. Users will be able to enhance their Macintosh computers through the customization that the Open Scripting Environment, also discussed below, will provide. The last two sections of this article will cover Apple’s future directions with the PowerPC (the processor being developed jointly by Apple, IBM, and Motorola) and the evolution of System 7.

If You’re Not a Marketing Type...you may want to start reading here. The sections that follow describe the new technologies that Apple plans to bring to market in the next few years.

O.C.E.—AN ARCHITECTURE FOR COLLABORATION

Before we get into Apple’s vision of the Open Collaboration Environment, or O.C.E., let’s first step back to what you might see in an O.C.E.-aware world:

While using your Macintosh, you double-click your “mailbox” icon and get a window filled with various items—electronic mail, voice mail, faxes, QuickTime movies, and documents with verifiable digital signatures. Regardless of the source, this is the one place you go to for all your incoming correspondence.

But there’s more beneath the surface. Three of the items in your mailbox are from a clipping service, a third-party company that knows your interests and automatically sends the news items that match your interest profile. On your Macintosh, you have an “agent” program that looks for these items, reads them, and stores them for you in a newspaper format. When you are reading your personalized newspaper, you can quickly and easily send copies of it to selected coworkers by using the O.C.E. mailer, which is in all O.C.E.-aware applications.

While you are out of the office, you can use your PowerBook to take orders from customers, write memos to coworkers, and send a purchase order to the accounting department, You can also approve a different purchase order and send it back with your digital signature. Your purchasing department can then process the order immediately, because it doesn’t need your physical signature on a physical piece of paper.

That evening, you can hook your PowerBook to an electronic-mail service such as MCI and let an O.C.E.-based program automatically send all your pending items. O.C.E. takes care of getting the items through MCI, onto your company’s network, and into the mailboxes of the right people.

O.C.E.—A Real-World Solution. Apple believes that O.C.E. will succeed because it was designed to fit into the real world, which usually includes partial solutions already in place and equipment and services from more than one vendor. And, as Dr. Gursharan Sidhu, Apple’s technical director of Collaborative Systems Development, puts it, “Apple is committed to making O.C.E. available on the leading computer platforms that our customers have and want to use.”

O.C.E. builds upon the foundation of System 7 and complements it (see figure 1). Together, they provide the tools that are needed to facilitate collaboration. Sidhu says that “the drive to work together in tasks that engage the intellect is very strong.” If collaboration is what people do to complete such tasks, personal computers can be of great help in helping people work together.

So far, computers can do several things to help collaboration, but they need to do more. Sidhu lists four barriers to collaboration:

•*Separation:* When people are not in the same physical place, such things as file sharing (via AppleTalk networks) and interapplication communication (IAC) help bridge the distance.

•*Simultaneity:* When two people are not present at the same time, you can use computers to store a letter now and deliver it when the recipient is at the Macintosh (this is called *store-and-forward messaging*).

•*Trust:* If you can’t be sure that documents are unaltered or that the senders are who they say they are, you’re not going to use compu- ters to do your collaboration. O.C.E. solves these problems, and its built-in encryption prevents un-authorized people who intercept your communication from reading them.

•*Comprehension:* A document does no good if it’s in the wrong format and the recipient can’t read it. O.C.E. provides standard formats for messages and letters, as well as a well-defined architecture for gateway-like software modules that translate the formats of messages and letters. These efforts will help ensure that users can comprehend the messages and letters they get.

Security Concerns. For the past few years, Apple has been working with RSA Data Security, Inc., to integrate its encryption technologies into the Macintosh framework. To ensure that an unauthorized party cannot read an intercepted communication, O.C.E. automatically encrypts all incoming and outgoing data on a net- work connection.

Because Apple has tightly integrated RSA’s technology into the Macintosh system, O.C.E. routines can encrypt and decrypt data at a rate of 1 million bits (128 K bytes) per second on 680x0 systems. (In this article, I use “680x0” to refer to Macintosh computers that are based on Motorola’s 68000 family of microprocessors.) Because of this, the encryption and decryption operations will be transparent to the user.

Authentication is a second security concern—in other words, how do I know that a document is really from Jane Doe? Within the O.C.E. framework, when you “sign” a document, O.C.E. routines generate a nonforgeable digital signature that goes with your document. When the recipient gets your document, the digital signature guarantees that the document was actually generated by you and that it hasn't been tampered with.

Once these three capabilities are in place and have been adopted by the general business community, we will be able to use electronic mail (and other services) where only signed and notarized pieces of paper are accepted today. Much of today's paperwork can be done with electronic equivalents that get the job done faster and with less effort and expense.

The Benefits of Standardization. O.C.E. delivers an open architecture for collaboration services. It provides a standard, "black box" interface for data going into it and coming out of it. By writing interface code to O.C.E.'s standard set of APIs, you can adapt your existing products and services to work with O.C.E. and take part in the benefits it delivers.

By using the O.C.E. APIs, you can create compelling and competitive products that exploit the full power of O.C.E. If, however, your source code must work on another hardware platform (or may someday need to), there is a second solution. Apple, Borland, Lotus, and Novell have developed and will support in their products an alternative API known as the Vendor-Independent Messaging Interface (VIM). By using the VIM interface, you can write code that works with O.C.E. and gives you the security and flexibility that only portable source code can give.

Directories. The O.C.E. concept of a directory is far broader than the conventional image of a telephone directory or the office directory in the lobby of a building. O.C.E. directories can contain a rich set of whatever data you desire, including not only text and numbers but also voice, video, executable programs, and any other data type you care to define.

The important thing to remember about directories is that you can extend the directory structure to include whatever data you want to provide. In fact, the same browser in a future O.C.E.-aware version of the Finder can be used to browse through any directory of information, including the directory of folders and files that the Macintosh's standard directory package provides.

A Foundation to Build On. The O.C.E. architecture radically extends the personal computer's sphere of influence. Before, you could collaborate with people on the same physical network. Using O.C.E., a large base of Macintosh computers can do business together, with a rich variety of services that are based on a shared, standardized foundation. (Remember that because Apple has designed and implemented both O.C.E. and all its services, you don’t have to.) Since we have provided the foundation, we look to you to build exciting new products and services on top of it.

PPC, IAC, AND APPLE EVENTS

System 7 introduced three very important components that gave the Macintosh platform powerful new ways of getting work done, ways that are not available on other platforms. Briefly, they are:

- *PPC (program-to-program communication),* which allows any two programs—either on one user’s Macintosh or on any two Macintosh computers on the same network—to exchange data.
- *Apple events,* standardized messages (sent over PPC communication links) that many different programs understand—like the Cut, Copy, and Paste commands, which all Macintosh applications know.
- *The Edition Manager,* which is built on the foundation of Apple events and automates the assembly of complex documents by updating subscriber sections whenever a publisher section changes.

The promise of Apple events is to end the era of ever-growing applications that try to do everything a user wants. (For example, the suggested partition size of Microsoft Word 5 is double that of Word 4; Word 5 includes such things as built-in thesaurus, grammar checker, and drawing modules). Apple envisions a future in which developers such as you create smaller, more modular programs that do one thing well and depend on other programs for needed services. Such products will be easier to design, implement, test, and maintain. You will be able to bring new products to market quicker, which will help your company be more financially stable.

Actually, to make these things happen, we have worked (along with interested developers) to create standardized suites of Apple events that we strongly recommend for integration into System 7 applications. A suite consists of events (for example, Get Data, Paste, and Save) and precisely defined objects (such as Document, Paragraph, and Rectangle).

Over the past 18 months, Apple has worked with developers and gotten their "buy-in" on Apple-event suites for text, QuickDraw graphics, and tables, as well as a Required Suite (mandatory for all System 7 applications) and a Core Suite (which contains Apple events and objects that most applications should implement).

Together, these suites comprise a document called the Apple Event Registry, the first version of which was released early this year. (It also has its own folder on this month's Developer CD.) Apple is working with developers to define more suites.

An Eventful Future. As the number of Apple event-ready applications grows, people will use Apple events more and more in their daily work. Their customers will want to buy more applications that respond to Apple events—and we hope that your applications are among the ones that do so.

But the story of Apple events doesn't stop here. Now we come to user scripting.

USER SCRIPTING

Apple events make interapplication cooperation possible, but user scripting will make it happen in the real world. (One definition of user scripting is "an approach to application customization and development for those who are not necessarily computer programmers." The HyperTalk language within Claris' HyperCard is the closest thing we have to user scripting today.)

The Golden Thread. User scripting is the golden thread running through the Macintosh of tomorrow and the years that follow. This new, fundamental line of control from the user to the Macintosh has several important characteristics:

- It will allow users to automate repetitive or complicated tasks.
- It will allow multiple applications to work together (most scripting languages today force you to stay inside one application).
- It will work with applications and data across a network.
- It is open-ended, and you can build on the basic framework to create fundamentally new scripting environments.

I believe that user scripting is the “sleeper” technology of the next few years. It promises to give the average Macintosh user capabilities that are unavailable on other platforms.

OK, So What Is It? In May 1991, Apple announced the Open Scripting Architecture as the foundation for add-on user-scripting systems that can be implemented by Apple or any developer. It is structured around three items: a new Scripting Manager (as part of system software), standardized suites of Apple events, and a mechanism by which an application can announce which suites it supports. Apple engineers designed the Open Scripting Architecture to provide standardized support for scripting without tying users to any one user interface for scripting.

Because scripts are ultimately equivalent to a list of Apple events and the objects upon which they work, scripts are not tied to any one human language (compare this, for example, with HyperCard, which *is* tied to an English syntax). Because it is the scripting system that translates a script to a human-readable form, you can take a script written in English and give it to someone using a Japanese-based scripting system. That person can immediately read that script in Japanese. (In fact, Apple demonstrated exactly this capability at the System Software Forum.)

By making the necessary foundations of a scripting system part of system software, Apple has opened the door for not just one kind of scripting but however many kinds developers want to create and users want to buy.

So How Will People Use It? That’s a good question, with several answers that we won’t know until somebody discovers them. But application-specific scripting systems have been in place for years (one of the earliest ones, still in use by many PC users, is the macro language inside Lotus 1-2-3). Listed below are some of the ways in which people have used such scripting systems.

The most obvious way to use scripting is to write scripts yourself—countless people have done so and continue to do so, even when the scripting system is hard to use. Alternatively, you may use scripts that someone else has written. You may also get them from coworkers, user groups, magazines, or companies that sell them, or the application you are using may include a supply of scripts.

In some cases, you can have the application “watch” what you’re doing and create a script from that. This is a more powerful way to generate scripts that most people can use, even if they have never written a line of script code themselves. (Once scriptable applications become common on the Macintosh, “recorder” programs that watch what you do and make a script from that will be very popular.) How else can users create their own scripts? You tell us.

So What’s the Big Deal? I submit that the script is a new and very important way of controlling the Macintosh, one that will generate unforeseeable benefits. Here’s my thinking:

Consider that a pre-Macintosh computer had exactly one source of user input, the keyboard. Enter the Macintosh, which adds the mouse as an input device. Great—now you have three ways of interacting with the computer: keyboard only (dull, but it can be done), mouse and keyboard (which is usual for most applications), and mouse only (which is particularly good for users who don’t like keyboards or for situations in which the computer doesn’t have a keyboard).

Now imagine a Macintosh (or a processor-based personal electronics product) with no keyboard and no mouse. Maybe it’s the size of your wallet (or smaller). Regardless of how you interact with it (and voice input is a natural here), the only way to influence it is to give it commands: in other words, to tell it to take object X and perform operation Y on it—which is very similar to the Macintosh way of selecting an object, then choosing an action to act upon it. So if your application is scriptable, it will be immediately usable with the hardware of the future—whereas nonscripting applications will not.

(It’s interesting to note that most people are comfortable using the mouse, which is “limited” to object/verb interactions. Many people are not at all comfortable with keyboards, which allow unlimited input and, with it, an unlimited number of invalid operations. Object/verb interactions do either something useful or nothing at all. So maybe the object/verb way of doing things isn’t so limited after all.)

The Scriptable Future. With System 7 and the Open Scripting Architecture, Apple has been building the foundation for user scripting for more than three years. The pieces are now in place. A few programs, such as Deneba’s Canvas 3, are ready to be scripted. Several scripting environments that conform to the Open Scripting Architecture, such as CE Software’s QuicKeys 2.1, Simple Software’s ControlTower, and UserLand’s Frontier, already exist.

For your application to enjoy the benefits of user scripting, you need to implement all the Apple-event suites that are appropriate, and you need to do it now. You will see some serious user scripting in 1992 and more in 1993. The future is scriptable.

IMAGING: THE NEXT GENERATION

In the last eight years, QuickDraw has been the graphical foundation of the Macintosh, and we wouldn't have a Macintosh without it. But customers always demand more, and what we need now is an imaging foundation for the '90s that is as powerful as QuickDraw was in the '80s.

Don't think for a minute, though, that QuickDraw is headed for the trash can—it has a long life and many enhancements ahead of it. But part of Apple's blueprint for the future includes creating a framework for imaging that will be powerful enough to handle the challenges not only of tomorrow's market but also those of the late '90s. This framework will provide state-of-the-art imaging power for applications that need it, while QuickDraw will continue to serve the needs of less demanding applications.

Imaging Today. When we talked to our customers and you, our developers, it became clear that we need to enhance several key elements of the Macintosh imaging architecture. First, the personal computer—for all of its considerable abilities—still isn't as easy to use in Chinese as it is in English. Writing applications for these different languages is even harder. At Apple, our goal is to change that: We want the Macintosh to be the undisputed first choice for personal computers in every nation (and language) of the world. To allow the Macintosh to display and print the written word as easily as we envision, we need to make fonts much easier to use and much more versatile.

Second, most people still don't use color because it is still too slow and too hard to use. With the new imaging architecture, we want to integrate people's use of color from scanner to screen to printer, with consistent, predic- table results.

We want to make color (in particular, color printing) as commonplace on the Macintosh as, say, fonts are today. We want you to be able to trust that you will get the color you want, regardless of what monitor or printer you are using.

Third, the Macintosh platform demands a richer imaging model—one that will be resolution-independent, support multiple color models, have extensive text and type services, and easily allow output to a great variety of output devices.

Currently, leading application developers are "pushing the envelope" ahead of the built-in color support that Apple provides. We're glad they're doing it, but we see a lot of redundant (and conflicting) technologies. We want to hand you as much color technology as possible so that you can build on top of it instead of inventing and reinventing the basic stuff.

Imaging Tomorrow. As I've just said, QuickDraw isn't going away. So how will Apple deliver an entirely new imaging model while maintaining compatibility with today's imaging model? Here's how:

•We will keep QuickDraw "healthy" by fixing bugs, making it work with the "world ready" Reference release of System 7 (including support for human languages that use 2-byte characters for text), improving its speed, and making strategic enhancements to it. One such enhancement is support for color management (which includes such things as ensuring that colors display and print equivalently).

•We will put the most useful new features into a completely new imaging architecture. This will be an optional extension that users can choose to include or exclude. Similarly, developers can use it a little, a lot, or not at all. It will be available if you want it, but we won't force you to use it.

•Once the basic imaging architecture is in place, we will continue to improve it by offering drop-in extensions. Future *themes* (as we will be calling them) will include support for 3-D (three-dimensional graphics), digital photography, and other services.

The Graphics Architecture. The new graphics architecture will bring higher-level, resolution-independent graphics to the Macintosh. It includes a rich set of operators that allow you to manipulate the basic graphic elements in many powerful and useful ways.

The most important feature of this architecture is that it deals with graphics at an object level. (These aren't objects in the object-oriented-programming sense, but they are objects in the sense that you create and manipulate lines and ovals, for example, rather than pixels.)

The graphics architecture gives you the following primitive (or elementary) object types:

•*Shape:* includes lines, curves, rectangles, and arbitrary polygons.

•*Path:* a collection of shapes.

•*Text:* treated as a first-class object; in other words, it can be transformed—rotated or skewed, for example—and still behave as editable text.

•*Bitmap:* can be resized or drawn at any angle.

•*Picture:* a container object that holds any combination of the above elements.

The imaging architecture includes transforms you can apply to objects: for example, matrix transforms (such as rotation, skewing, and scaling) and built-in clipping (that is, limiting drawing to be within a set parameter). The architecture also includes built-in object-collision detection—that is, you can easily tell if two objects touch or overlap.

Finally, this architecture contains considerable support for color—or, in its own terminology, *inks.* This includes built-in dithering and halftoning. It also knows how to do color separations and can work with more than a dozen color-specification families, including several versions of the RGB (red/green/blue) and CMYK (cyan, magenta, yellow, black) families.

The Open Printing Architecture. Everyone in the industry knows that it is often difficult to create a new printer driver. The new printing architecture makes this process easier in several ways, one of which is by building in common features that usually need to be included in a printer driver. Under this new architecture, you can create printer drivers in a matter of days or weeks, not months or half-years.

This new printing architecture is an open one; it includes “hooks” that allow you to add to or modify its behavior. It is device- inde-pendent and can output either QuickDraw commands or PostScript commands. It also supports a third type of device (the plotter) and can output the vector commands that a plotter expects to see.

The Open Printing Architecture will make things easier for users, as well. It will give them more control over some variables such as page size, but it will handle other details intelligently and automatically. It also provides more direct-manipulation actions (such as dragging a document icon over a printer icon to print the document) and gives users more feedback during printing.

The Line Layout Manager. We talked about this manager several years ago, but we delayed finishing it until we had an imaging architecture that could make full use of its power. That time has come, and the Line Layout Manager provides the key features needed to make the Macintosh as easy to use in, say, Chinese as it is in English.

The Line Layout Manager gives precise control of the layout of text on a line. Macintosh system software needs it to deal with what- ever script (writing system) the Macintosh is using. The Line Layout Manager, working hand-in-hand with the new "world ready" version of System 7, will make it possible for you to change the default script from French to Farsi, for example, by dropping the appropriate extension file into the System Folder.

Apple eventually wants to see the day when users can easily work in their native language and any other languages. Today, applications can work with multiple scripts, some reading left-to-right and others going in the opposite direction (even on the same line). In the future, the Macintosh will also be able to work in scripts that are written in columns, top-to-bottom.

The Line Layout Manager will also give you and other developers the layout control you need in order to deliver high-quality typography in all your applications; by doing so, you give your customers better-looking output.

Fonts. Tom Ryan, manager of the Macintosh Imaging group, says that one goal of the new imaging architecture is "better, typographic-quality text on every system in the United States, as well as around the world."

To make this happen, Apple engineers had to add certain features once considered "bells and whistles"—such things as contextual forms (glyphs whose shapes change based on the glyphs that come before or after them) and ligatures (two glyphs that merge into one when they occur next to each other—like "æ" and "œ"). Also, because the new imaging architecture treats text as just another kind of object, you will be able to do all sorts of graphic transformations on text strings and have them still behave as editable text.

Here's one feature that will be most welcome: The new imaging architecture will support both TrueType and PostScript Type 1 fonts equally, with no special restrictions or hidden glitches. Users will treat the two font types identically, and you as developers will deal with fonts through one API (application programming interface). In addition, Apple will be enhancing the TrueType font format to permit something called *algorithmic fonts.* These fonts allow one master font to generate a spectrum of related fonts that look similar but differ in one or more characteristics.

Upgrading Your Image. The new imaging architecture will give us higher-level and more extensive control over text and graphics on both the screen and the page. By providing more capabilities in the system software, we hope to “raise the bar” and provide Macintosh users with richer and more expressive products. We’re providing the tools, and we’re trusting you to make the products.

ADVANCES IN EASE OF USE

Ease of use is probably Apple’s greatest strength, so Apple wants not only to maintain it but also to increase this advantage over other computers. The sections below describe the directions that the Macintosh Look-and-Feel group, headed by Phac Le Tuan, find to be most promising.

Advances in Software. For most of us, the Finder *is* the Macintosh user interface. When we’re not running an application, the Finder makes visible the visual metaphors by which we direct the Macintosh to do operating-system tasks (for example, dragging a document into the Trash to delete a file). Even when we are running an application, the Finder is still available for immediate use (in System 6 when using MultiFinder and in System 7).

More importantly, most developers use the Finder as a model when they design their program’s user interface. So it should come as no surprise that any software improvement of the Macintosh computer’s ease of use will somehow touch or be embodied in the Finder.

System 7 will be the foundation for any software improvements in the ease of use of Macintosh computers. The following are four areas in which Apple engineers plan to make improvements:

•*Managing complexity:* System 7 began the process with Balloon Help, which answers the question “What is this?” We need to give users new tools that answer other questions such as “How do I do this?” and “Why is it so?”

We can do other things to help users visually manage complexity. We plan to use color, shades of gray, and sound to give the Finder a new, more informative look. We are also examining animation as a way to help users remember what user-interface elements do.

•*Managing large amounts of information:* The disarray of most office desks (well, mine, for sure) attests to the nearly overwhelming amount of information most of us regularly deal with. We want to expand the Finder interface to give users new ways of structuring information.

One promising new metaphor is (look at your desk) "piles" of information. Most of us file items by content—we read the item and file it based on its content. Piles represent an earlier attempt at organization: You probably put an item in a pile (without having read it) based on how you intend to act upon it—such as *read these first* or *have these filed* or *give these to Mikey.* On the desktop, a pile has its own icon (as a folder does), but you will be able to "riffle through" the pile and see summary information on each item—without having to open windows or individual documents.

•*Enriching current interface elements:* System 7 includes several extensions to pre-System 7 behavior. Alias files, for example, made the file/folder/window metaphor richer. Drag-and-drop execution extended the metaphor of dropping a file into a folder to include opening a file by dropping it onto an application that can open it.

One possible extension to the current user interface is that of drawers. When you move the mouse pointer to a certain area, a drawer "opens," revealing new icons to access; move the pointer away and the drawer slides out of view. Another possible extension makes it easier for the user to drop items into a folder nested several levels deep.

•*Extending the environment:* By carefully adding new functions to the Finder, we can make the Macintosh environment easier to use. We plan to make more use of "magic routing" (which currently puts extensions and control panels into the right folder when you drop them into the System Folder).

Of course, AppleScript and other scripting environments will allow users to customize the behavior of their Macintosh. Remember how System 7's System file did away with the Font/DA Mover utility? We are also looking at integrating the functions of the Installer utility into the Finder itself.

Advances in Hardware. Although the PowerPC line of Macintosh computers is discussed in greater depth below, it is important to mention it here for two reasons. First, we should note that we will need more computing power to deliver more-powerful user-interface elements—such things as animation and speech input.

Second, we should also realize that as the range of computing power across Macintosh models widens, the user interface must be scaled to fit the computing power available. This will probably mean that high-end Macintosh models will have computation-intensive user-interface elements in addition to the elements available across the entire Macintosh line.

New Technologies. New “flavors” of personal computing are on the horizon, and we already know that they contain user-interface challenges and opportunities:

•*Pen-based computing* has already received a lot of attention at Apple. The availability of or reliance on a pen-based interface brings up many fundamental questions. If, for example, you completely replace a keyboard with a pen and a screen, the number of pen gestures needed becomes important. If your interface needs the user to know, say, five gestures, that’s one thing. But if it needs 25 gestures, several questions arise: Is this too many gestures? Can users remember them all? How much help can we give users without its becoming intrusive?

We believe that the pen must be highly integrated into the architecture of the computer, the application, and the document. If it is not, the pen will feel clumsy and “tacked on”—and that’s not good enough for the Macintosh. Apple already has a strong handwriting-recognition architecture in place, and we feel that we will be able to offer a more natural, less intrusive form of pen-based computing than our competitors can.

•*Speech* is another important future technology. According to Le Tuan, “Speech is definitely the technology that is most likely to reduce size and cost.” Both size and cost considerations enter heavily into two attributes to which Apple is committed: portability and affordability in both the Macintosh computer and the new type of product that John Sculley calls “personal electronics.”

Speech technology takes two different directions, *speech input* and *text-to-speech output.* In these related fields, says Le Tuan, “We believe that we have one of the best technologies in the world.” Once this technology is in place, the keyboard and display screen will no longer be the limiting factors in reducing the size of a computer or personal-electronics product. Speech technology will make an entirely new class of product possible.

(At the System Software Forum, Le Tuan demonstrated the experimental speaker-independent “Casper” Macintosh, which listens for keywords and executes commands. This is the same experiment that John Sculley publicly demonstrated earlier this year.)

•*The consumer electronics market* will present Apple with new challenges in ease of use. For one thing, Apple will be selling its products to a much wider audience. “The challenge for us in terms of ease of use,” says Le Tuan, “is that we will encounter users with all levels of education. It is very important to have a system that is immediately operable as soon as you open the box.” Since ease of use has always been Apple’s strongest point, we believe that we can successfully apply our experience to the consumer market.

Other Advances. The Macintosh is expanding past a “one size fits all” model of computing. We see four distinct markets: *education, homes, small businesses,* and *enterprise systems* (corporations). This will lead us to devise different user-interface solutions based on the users and the type of computer they are likely to have. We also feel that doing user testing in “neutral” environments is not enough; we plan to do more testing “in the field,” wherever users may be working.

We also plan to do more user testing outside the United States. Apple is committed to creating a “world ready” computer that can easily be used by anybody, in any language, anywhere in the world.

QUICKTIME: MULTIMEDIA FOR THE MASSES

QuickTime, the first of several extensions to the System 7 software architecture, presents an open-ended architecture for dealing with time-based data. Right now, that means digitized audio and video, but in the future Apple will add MIDI music data, analog video, and more. (QuickTime also includes graphics-compression/expansion routines that are useful even if you’re not into QuickTime movies.) There’s a lot to QuickTime, as the following sections will demonstrate.

A Success in QuickTime. As promised, Apple shipped the first commercial release of QuickTime in December 1991. There are more than 3 million QuickTime- ready Macintosh computers (this includes any model with color and a 68020 processor or better). Apple has shipped more than 12,000 QuickTime developer kits, and developers are using them. As of early March 1992, there were more than 60 QuickTime products on the shelf; more than 220 products are shipping or announced.

For users, Apple will soon be shipping the QuickTime Starter Kit (due by June 1992 at a suggested retail price of less than $200 in the U.S.). This kit will give users what they need in order to start using QuickTime creatively. One of the most interesting items is the QuickClips CD-ROM, which contains more than 400 megabytes of QuickTime movies. It will also include utilities such as a simple player/editor, recorder, and movie-conversion program and an “electronic application catalog,” which will familiarize users with the variety of available QuickTime products.

QuickTime Does Windows. We recognize that you live in a world where you must be successful on more than one compu- ter platform (usually Microsoft Windows or IBM-compatible DOS besides the Macintosh), and we want to give you tools that will help you do so.

QuickTime will not be limited to the Macintosh platform. According to Doug Camplejohn, Apple’s QuickTime product manager, “We’ve been approached by every major player, software and hardware...about bringing the capabilities of QuickTime to other platforms.” Since January 1992, APDA has been selling the QuickTime Movie Exchange Toolkit, which contains tools that will help you take data from other platforms and play them back on a Macintosh as a QuickTime movie.

We are currently working on a QuickTime Player that will allow you to play QuickTime movies under Microsoft Windows 3.0. We will license this technology to you for inclusion in Windows products, and we will ship a developer’s kit for this technology by the end of 1992. (To show that this technology is real, Camplejohn showed a QuickTime movie playing on a Dell IBM-compatible computer running Windows 3.0.)

The Apple Philosophy. QuickTime is yet another example of Apple’s approach to helping you develop better software: We put the foundation routines into system software, which gives you the freedom to concentrate on the “guts” of what you really want to do instead of spending all your time building the foundation from scratch.

There are other nice things about QuickTime that reflect Apple’s way of doing things. QuickTime is optional for both you and users. It adds to what you can do, but it doesn’t break what you’ve already done. It’s easy to install—just drop the QuickTime extension into the System Folder, and reboot. (To make an IBM clone ready for multimedia, a user has to buy an expensive kit of add-on hardware and software. The procedure for installing the kit takes—I’m not making this up—more than 65 nontrivial steps. My favorite step is, “If wire pinouts don’t match, refer to the wiring diagrams to build your own.” You get the idea.)

QuickTime is opening up entirely new uses for Macintosh (and, later, Windows) computers. I know that this may sound like hype, but it’s not. With QuickTime, it is as easy to imbed short video-and-audio movies into a document as it is to paste a drawing in or add text in another font. (I can’t wait until the word processor I use adds QuickTime capabilities.)

If it makes sense for your application to add movies to its documents or for it to create movie content for use elsewhere, you should consider adding QuickTime support. I don’t want to make you feel insecure or anything, but hundreds of developers are already doing so.

NETWORKING

When AppleTalk first came out, it was revolutionary, because it connected you with the rest of your workgroup—that is, with all the people around you in your office. As computers have become more sophisticated, however, so have the demands we place on them. Now our “workgroup” may include one person on another floor, another in the next building over, and a third in another city (or state, or country). It may also include (and often does) someone who uses a different kind of computer and network. To sum it up, you need a solid network that can easily work in a multivendor world.

Apple has two goals in networking. According to Jim Groff, the senior director of marketing, Enterprise Systems Division, our first goal is to “ensure that the Macintosh is a first-class citizen in every leading local-area-network environment.” The second goal is that, whatever area of networking we are active in, “We want to do it better than the way it is done on other platforms.” This means that, for example, not only do we want users to connect easily to both IBM and DEC networks but also to take for granted that they can copy and paste data from both the IBM and the DEC windows to any other window.

The key to Apple’s current and future success is the tight integration of all the layers of networking—hardware, networking software, network services, user services, and networked applications—each of which builds upon the pieces underneath. The foundation of Apple’s networking is AppleTalk, and by licensing the AppleTalk technology to other platforms, Apple is building an ever-widening circle of compatibility in the multivendor world of networks.

Apple’s partnership with DEC is its oldest—it goes back to agreements signed in early 1988. DEC has been shipping Pathworks for Macintosh since early 1991. Pathworks provides a wide-area network and network services for a mixed VAX/ Macintosh environment.

A new product called Desktop ACMS will help integrate Apple and DEC computers doing on-line transaction processing. In Europe, DEC is selling and servicing Macintosh computers along with its VAX computers.

Networking is the area in which Apple’s joint efforts with IBM are most advanced. In 1991, Apple shipped a token-ring card that allows Macintosh computers to connect to an IBM token-ring network. IBM has shipped a router that can handle AppleTalk, and it also sells a document, *Apple/IBM Guide to SNA Networking*, that tells how to use Macintosh computers with OS/2, MVS, and AS/400.

In the “standards” environments—that is, TCP/IP, OSI, and UNIX—Apple already has the supporting products MacTCP, MacX (for X Window support), and A/UX (Apple’s combination of the Macintosh user interface with a standard UNIX environment). Apple is also beginning to provide support for connection to ISDN.

Apple is also working with other companies that provide network support—Novell, Banyan, and Microsoft, among others. To summarize, Apple is committed to providing network connectivity wherever it is needed, while maintaining the power and ease of use Macintosh users have come to expect.

THE POWERPC PLATFORM

There are several business and technical reasons Apple entered into a partnership with IBM and Motorola to make PowerPC computers. However, one reason stands out in relation to Apple's product strategy: To make the Macintosh platform processor-independent while preserving developers' and users' investments in current Macintosh technology.

What does this mean to you as a developer? It means that the same application binary file that you are shipping today will run on a PowerPC Macintosh. It means that you can create programs in the future that will run, as is, on the 680x0 as well as on the PowerPC models of the Macintosh. (For details about how PowerPCs will be able to run existing 680x0 applications, see the article after this one entitled, "How I Learned to Stop Worrying and Love Emulation.")

The sections that follow will tell you more about the PowerPC chips and how Apple intends to use them.

What Is PowerPC? The PowerPC project is a joint effort of Apple, IBM, and Motorola. The PowerPC processor design will be an adaptation of IBM's RS6000 RISC technology, adapted to meet the future needs of the personal-computing market. PowerPC processors will fit on a single chip (unlike the multichip RS6000 processors).

Eventually, there will be a family of PowerPC processors, each adapted to the needs of a particular market segment. The PowerPC instruction set will include some new instructions and remove unneeded instructions from the RS6000 instruction set—all done to enhance the PowerPC's performance in personal-computing situations. The first chips will appear in late 1993, followed by other designs in 1994 and later. Currently, there are plans for four design centers (or types) of PowerPC: Low-Cost Desktop, Mainstream Desktop, High-End Desktop, and Low-Power.

One interesting fact about the PowerPC architecture is that its current design can be extended to include 64-bit addresses. Not only that, but applications designed to run on a PowerPC system with 32-bit addresses will run on a 64-bit-address PowerPC without modification. (Contrast that to the amount of pain that happened—and is still happening—in the conversion from 24-bit Macintosh applications to 32-bit ones.)

Although 32-bit or even 48-bit address spaces will certainly be sufficient for the foreseeable future, Apple, IBM, and Motorola want to make sure that the market doesn’t outgrow the PowerPC architecture anytime soon. The intent is to design a processor that will be sufficient for the personal-computer and related markets through the end of this decade.

Why a RISC? Like many other companies, Apple believes that as this decade progresses, RISC processors will far outstrip today’s conventional CISC (complex-instruction-set computer) designs. However, Apple has another reason for favoring RISC chips: Compared with CISC chips, RISC chips provide far more computing power per unit of power consumed. This factor will be important in the new personal-electronics products. Such devices will need to run for long periods of time on the relatively small amounts of power that batteries can deliver.

Current CISC technology can perform at 10–20 integer Specmarks (a benchmark measurement of processing power), whereas RISC can perform at 50–70 integer Specmarks. By 1995, PowerPC performance should be well over 100 integer Specmarks, a figure that Philip Koch, Apple’s manager of RISC Technology Development, describes as “nipping at the heels of scalar supercomputer performance.”

The PowerPC chips will easily outperform CISC designs. The first PowerPC chip will have a frequency of 50 megahertz and will contain more than a million transistors. By the middle of the decade, this should increase to a 100-MHz frequency and more than 10 million transistors. Compare this with processors such as the Motorola 68040 and Intel 80486, which commonly run at frequencies under 50 MHz and have about 1.2 million transistors.

What will all these transistors be used for? Koch says that PowerPC chips will contain very large on-chip caches and “multiple independent functional units.” The caches will make the chips faster by minimizing the time they spend pulling data from sources outside the processor itself.

The multiple functional units will help implement *superscalar* behavior—the ability for the chip to execute more than one instruction per cycle. (In contrast, *pipelining* enables a chip to execute more than one instruction at once but finish slightly less than one instruction per cycle.)

SYSTEM SOFTWARE STRATEGY

According to Phil Koch, “The OS strategy for Macintosh is really very simple: We’re going to support the same software architecture on all platforms, and that software architecture is System 7.”

The More Things Change... Future plans for Macintosh system software include the addition of a microkernel between System 7 code and the hardware on which it is running (see figure 2). Phil Koch describes it as “essentially a very small operating system within the operating system....[that] encapsulates the hardware and certain services such as address-space management and task management.”

The microkernel performs the critical function of providing a standard set of services to the system software, regardless of the details of the hardware itself. In essence, the microkernel will make it possible for the System 7 we’re using today to work on both 680x0 and PowerPC processors.

Apple will create separate but equivalent—that is, same API—microkernels for 680x0 as well as PowerPC Macintosh computers, thus making the same advanced services available to any program, regardless of which processor is in the Macintosh being used. These include such advanced features as

•Large 32-bit demand-page addressing.

•True preemptive multitasking.

•Virtual memory.

•A component architecture (called the agent architecture) based on objectlike interfaces.

•High-performance message-based intertask communication subsystem.

•Support of dynamic linked libraries.

•Synchronization between tasks through messages and semaphores.

These features will give you the power you will need to write state-of-the-art applications in the 1990s. For example, one of the first uses of the microkernel will be a new I/O subsystem designed for concurrency and high real-time performance. This new I/O subsystem will support DMA (direct-memory access) and asynchronous SCSI operations. Some of the microkernel’s features—such as preemptive multitasking—will take a while to be available at the application level, although they will be immediately available to low-level code such as the new I/O subsystem.

...The More They Stay the Same. Even though the microkernel will make new things possible, the machine will still be the same Macintosh you know today. The transition to the PowerPC will be effortless for both users and developers. Users will still see the same user interface and use the same peripherals and floppy disks. According to Phil Koch, developers will see "one source [code] base to maintain, written to a single API that's available on 680x0- as well as PowerPC-based Macintosh computers, as well as a set of tools to help developers move their applications to PowerPC."

All in all, the PowerPC architecture is exciting in that it represents a smooth transition to a new level of Macintosh performance and function. It also demonstrates Apple's long-term commitment to the Macintosh. Developers and users alike can look forward to the PowerPC because it offers new possibilities without making people walk away from a significant investment in an incompatible architecture.

CONCLUSIONS? YOU WANT CONCLUSIONS?

We've covered a lot of ground, and I hope that you can step back and see how comprehensive a vision this is. This article and the System Software Forum both covered technologies, not products. Through *Apple Direct* and other channels, we will continue giving you details of the products as they emerge. I hope that this article has gotten you excited about the future of the Macintosh and that you'll be better able to make plans for the years ahead.

****

Figure 1: O.C.E. and System 7. Together, these two can overcome the four major obstacles to collaboration: separation, simultaneity, trust, and comprehension. The shaded area at the bottom represents the underlying foundation that all the functions above it share.

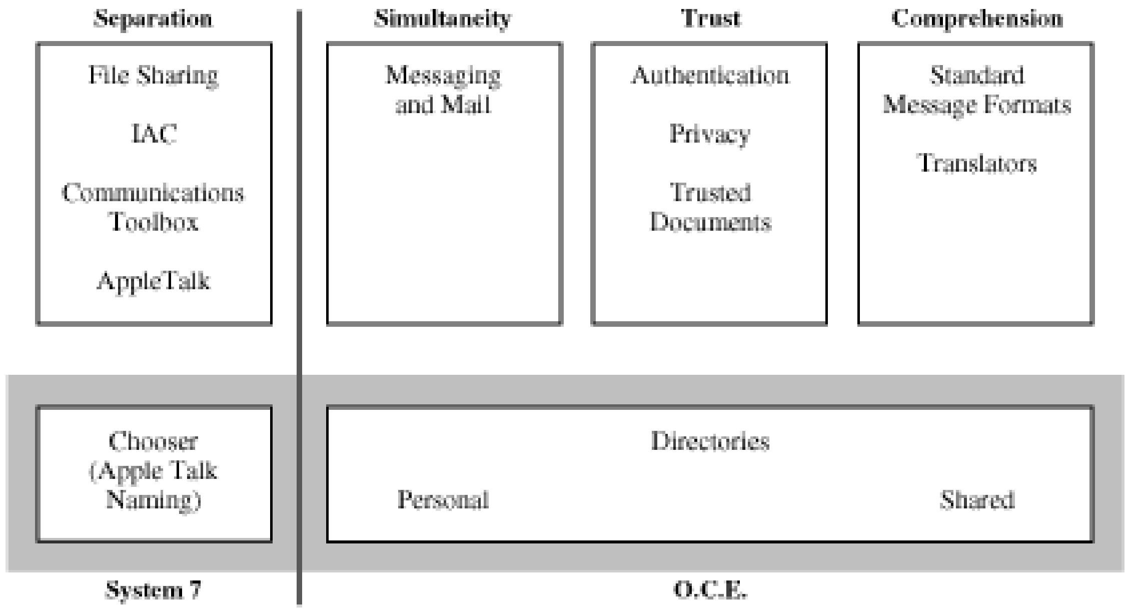
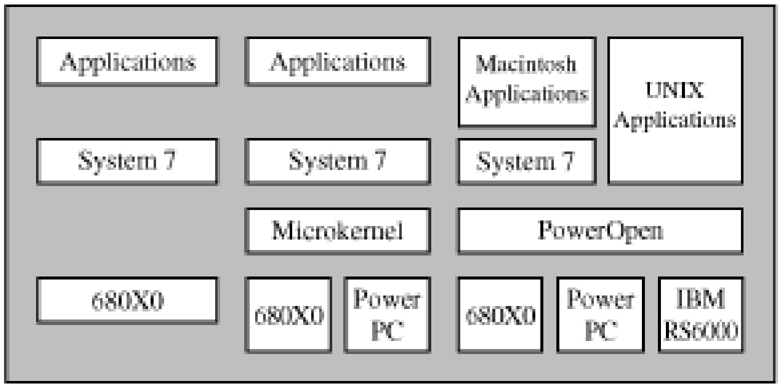


Figure 2: The future of the Macintosh architecture. Each column represents one future kind of Macintosh computer. Applications will continue to run on Motorola 680x0 (left column), but the microkernel will allow the same programs to run on PowerPC Macintosh computers as well (middle column). The PowerOpen architecture will allow Macintosh and UNIX programs to run on 680x0, PowerPC, and RS6000 processors (right column).



****

Developer Support Directions

Developers are important to us, and one tangible expression of that is the $50 million that Apple spends annually in support of the developer community. The following sections summarize several new directions of support.

Cross-Platform Support. We are working to give you tools you need to develop applications for both the Macintosh and the Windows 3.0 environments. In the future, we will consider supporting additional platforms if it makes sense for us to do so.

According to Kirk Loevner, director of the Apple Developer Group, "MacApp will form the foundation of Apple's thrust into the cross-platform area. You will see us expand MacApp to support cross-platform development." MacApp is a good product for many reasons, and this statement of direction is just one more. We also plan to create a QuickTime-movie player for the Windows 3.0 platform.

Enhancing MPW. We understand how critical a good software-development environment is, so we are enhancing the Macintosh Programmer's Workshop (MPW) development environment to meet your needs. In the works are incremental linking, a better user interface, and the ability to use dynamic linked libraries.

Software Access Initiative. You face several major problems in getting your products sold. One problem is shelf space—too many programs competing for a limited amount of display space in stores. Another is your lack of visibility to users. If potential customers don't know that your product exists, they won't be buying it, no matter how good it is.

A new program, the Software Access Initiative, will give you new ways to reach customers. In late 1992, we will start shipping new models of the Macintosh that will contain a CD-ROM drive. As part of our Software Access Initiative, these systems will include a CD-ROM that contains demo programs for numerous Macintosh products. Purchasers can browse the CD-ROM at their leisure and thus make better buying decisions.

We are also looking into several other new channels of distribution: catalogs, software distribution through AppleLink or some other electronic information service, and software kiosks that can demonstrate and sell literally hundreds of titles in a compact space.

Antipiracy Efforts. You've been reading about Apple's antipiracy initiatives in previous issues of *Apple Direct.* (See "The Ways and Means to Fight Piracy" article, in the March 1992 issue, and "Apple Launches Antipiracy

Plan," in the October 1991 issue. Also, be sure to check the new "Antipiracy Toolkit" folder in "Hex, Drives, and Videotape," the April 1992 Developer CD.) Apple is continuing the fight on several fronts, and Apple Direct will keep you informed on such efforts in future issues.

Delivering to Developers. New technology will make little difference if we don't have an enthusiastic base of developers supporting it before it is publicly announced. Over the past few years, we have evolved a system that involves a few nondisclosed developers early in a project and adds more developers and support as the project nears completion.

When the software reaches the beta stage, we seed the software, documentation, and sample code to all our developers. This process ensures that when we release the software to the public, we can list anywhere from several dozen to more than a hundred products (either shipping or announced) that use the new software technology.

(We managed to do this successfully with QuickTime: At last January's Macworld Expo, one month after QuickTime was publicly announced, third-party developers demonstrated more than 100 QuickTime-savvy products.)
****

The System 7 Foundation

The key to understanding Apple's operating system strategy is that the System 7 API (application programming interface) is the foundation for building your Macintosh programs, both now and in the years to come. Applications built for System 7 will run on future versions of System 7, PowerOpen (the future descendant of AU/X), and the Taligent operating environment.

System 7 is Apple's mainstream personal-computing operating system for desktop and notebook computers. Its birthright is the user-centered design that has led the industry for eight years. We'll continue extending this computing paradigm through the '90s—adding voice and pen input, improving the platform's graphics and communications, and restructuring the operating-system kernel to bring collaboration and new media to desktop personal computing.

A/UX 3.0 is Apple's combination of an industry-standard implementation of the popular UNIX operating with the easy-to-use Macintosh environment.

A/UX provides the same user interface and runs the same applications as System 7, but with a UNIX foundation. The next major release, A/UX 4.0, will bring the benefits of A/UX to the PowerPC-based PowerOpen platform. It combines the best technology from IBM’s highly successful RS/6000 AIX family and Apple’s A/UX.

Taligent, an independent company jointly owned by Apple and IBM, is developing a new kind of operating system for the ’90s. By building the entire system on object technology, the Taligent system will dramatically increase the productivity of commercial as well as custom-application developers and enable software innovation to keep up with the pace of hardware development.

All three platforms will support the System 7 API. That means that applications you write today for Macintosh can be used by buyers of these three platforms in the future.

****

How I Learned to Stop Worrying and Love Emulation

Elsewhere in this article, I made what might sound to you like a totally impossible claim, that PowerPCs will be able to run existing applications (containing nothing but 680x0 object code). How can Apple possibly accomplish this? The answer is *emulation.*

“Oh no,” you groan, “You’re going to try the old emulate-the-processor-in-software trick? That’s always slow! No one will bother running PageMaker on a PowerPC Macintosh if it runs as if it were on a Macintosh Classic.” But Apple engineers have come up with an approach that works.

They analyzed the execution of various Macintosh programs and found a very interesting fact: Every program spends more time executing Macintosh System and Toolbox calls than it does executing the code the programmer has written. Equally importantly, the Macintosh probably spends most of that time executing only a handful of Toolbox calls. For example, PageMaker spends 88 percent of its time in the Toolbox and Excel spends 67 percent of its time there.

Both programs use DrawText the most, followed in decreasing order by EraseRgn, Line, and GetFontInfo. Excel spends about 35 percent of its time in OS and Toolbox calls executing one call—DrawText (PageMaker spends about 20 percent). Past that, Excel spends no more than 11 percent of the

OS/Toolbox time executing EraseRgn, 10 percent executing Line, and less time for all other Toolbox calls. (PageMaker spends 16 percent of the OS/Toolbox time executing EraseRgn, followed by 15 percent executing Line, and so on.)

Apple’s solution for running 680x0 software on the PowerPC has two parts. The first is the obvious (although not trivial) one: to write PowerPC software to emulate the 680x0 processor and emulate your program, instruction by instruction. The second part is, I think, pretty clever: to rewrite key OS and Toolbox commands in “native” PowerPC code and run them instead. The native code executes about ten times as fast as 680x0 code can be emulated. Since every program spends a large amount of time executing a handful of known, fixed routines, the overall performance of this hybrid executed/emulated system will approach that of a system executing native code alone.

Can this approach work? At the System Software Forum, Apple demonstrated, as proof of concept, an experimental computer that used the Motorola 88000 RISC processor. It ran out-of-the-box System 7, PageMaker, and Excel at what looked like acceptable levels—using 680x0 emulation only. As ultimate proof, the machine also ran the famous After Dark “Flying Toasters” screen-saver module. Obviously, the same computer would have run much faster had OS/Toolbox calls been rewritten in native processor code.

Developer Options. This hybrid emulation/execution scheme will give you several development options. Among them are (in increasing order of effort and reward):

•*Emulation:* You can continue developing applications as you do today. These will have the fastest time to market, and they will run on 680x0 as well as PowerPC Macintosh computers.

•*Binary-to-binary translation:* We are testing the feasibility of tools that might take a 680x0 binary (executable) file and translate it to an equivalent PowerPC binary file. Our efforts with prototypes of such tools are encouraging, which means that binary-to-binary translation may be another way to move existing Macintosh programs to the PowerPC platform.

•*Recompilation*: If you recompile the program you’ve written (probably in C or C++) on a PowerPC development system, the resulting PowerPC program will be even faster and have access to more PowerPC features.

•*Redesign:* Although this option may take more time, the result is a program that makes the PowerPC (and your company’s reputation) shine. You will be able to add features that no 680x0 Macintosh can equal, and your program will have a long life.

Introducing the Developer Support Center

Increased commitment, faster answers from a single source

Providing continually improved support to developers is a vital part of Apple’s business plan for the 1990s. In the Apple Developer Group, supporting developers is not just an afterthought or an add-on—it’s what we do.

The developer support team is proud to announce the results of ongoing efforts to improve how information requests are handled—through the creation of the Developer Support Center.

The Developer Support Center evolved in response to developers’ concerns about getting development support from Apple. You need to clearly understand where to go for what types of information, so that you don’t get passed from person to person in search of the right answer. You also need answers quickly; waiting up to ten working days for an answer isn’t acceptable. When you call Apple for development support, you shouldn’t have to make multiple phone calls. You deserve to get an accurate, courteous, and thorough response as quickly and efficiently as possible, from one central source.

Organizing the right combination of resources to meet these needs has been a challenge; however, we feel we’ve made some major strides during the past year. By improving the process through which we handle information requests and by implementing a new, tiered approach to support, we’ve drastically reduced response time. Most questions are answered in fewer than three business days, and many developers also tell us that the quality of the responses has significantly improved.

The result is a new resource for handling inquiries—the Developer Support Center (DSC).

The DSC is the gateway to a wide variety of resources available to answer development-related questions. Put simply, if you’re developing on an Apple platform and are unsure of where to turn with a development question, the DSC will help you locate the best resources to meet your needs.

The DSC has a new AppleLink address, DEVSUPPORT. AppleLink messages that previously were sent to MACDTS, A//DTS, and DEVHOTLINE should now be sent to DEVSUPPORT. For more information about how to contact the Developer Support Center, see the sidebar below, “How to Contact the Developer Support Center.”

No matter how you submit your question (E-mail, telephone, or fax), you can expect it to be forwarded to the right person, an expert on that subject. And the DSC monitors all requests for information from the time they come in the door until you've received a satisfactory answer.

Getting Answers. If you've called Apple for development support before, you've probably come in contact with the Developer Support Hotline. Although the phone number is still the same (408/974-4897), when you call now, we will greet you as the DSC and the first contact you make will be with one of the DSC Specialists.

When you contact the DSC by telephone, the DSC Specialist will often be able to answer your questions immediately. In cases where further research is required, the Specialist will draw on a variety of resources to get the requested information. The DSC will alert you if we'll be unable to provide a full answer within three business days.

The entire DSC, composed of several groups, including Developer Technical Support (DTS), also handles requests sent by E-mail (AppleLink, Internet) or fax. All E-mail messages are collected hourly by DSC Specialists, who are responsible for making sure that questions are either answered immediately or routed to the person or resource best qualified to give an answer (thus the "tiered" approach).

No matter who ultimately responds to the inquiry, you will receive an answer or information on your inquiry's status within three business days. Furthermore, if your question will be better addressed by a department other than the DSC, the DSC Specialist will direct you to the best resource.

Developer Experiences. Developers who haven't used our support services recently will be pleasantly surprised by the impact these changes are making. Here are the experiences of several developers who have contacted the DSC since the addition of the tiered support structure.

Jeff Bronez of Bobbing Software, an Apple Associate, says, "I have gradually phoned with wider- and wider-ranging questions and have received quick and complete answers to all of them. Exactly what questions won't you answer? I'm going to keep on calling until I find one!"

Bary Pollack of General Parametrics had this to say about his experiences working with DTS: "Such immediate and personal response from DTS is one of the reasons that working with Apple and developing in the Macintosh environment is so rewarding. You may not be aware of the difference between DTS support and support from other vendors such as IBM and Microsoft. It’s even more different than night and day!”

“I have to admit that my impression of your support group be- fore was that I submitted prob-lems to what seemed like a black hole. It’s refreshing that it’s no longer like that,” says Jeffrey Siegel of Evergreen Technologies, an Apple Partner.

Or, as Scott Marcy from First Class Systems, another Apple Partner, says, “I really appreciate your prompt and efficient service. Every time I’ve called, I’ve received wonderful support, and I’ve had some unusual problems! My problems have always been solved quickly and to my full satisfaction.”

We encourage all developers to use the DSC for fast, efficient handling of development questions. Be sure to check the new edition of the *Developer Handbook* for valuable information about how to best utilize the DSC. Partners and Associates will receive it in the May monthly mailing.

Also, for those who are attending the Apple Worldwide Developers Conference, Jim Abrams, manager of developer support, will lead a presentation called “One-on-One with the Developer Support Center.” Check your agenda for details.

Your feedback is especially important to us. As Abrams puts it, “At Apple, support is not just another ‘pretty’ message—it’s a key deliverable.” To help us continue improving our services, send us your remarks via AppleLink to DEVFEEDBACK.

****

HOW TO CONTACT THE DEVELOPER SUPPORT CENTER

Specialists at the Developer Support Center are standing by to answer questions and hear your ideas for future service improvements. Apple Partners and Associates get unlimited access to the DSC. Non-members can contact us for limited support or to request an application for the Partners and Associates program. Here’s how to contact the DSC:

By telephone:	(408) 974-4897

Telephone hours:	Monday–Thursday from 8 A.M. to 5 P.M. Pacific time; Friday from 8 A.M. to 4 P.M. Closed daily from noon to 1 P.M. (voice mail operates 24 hours a day)
By AppleLink:	DEVSUPPORT for inquiries requiring a response; DEVFEEDBACK to remark on our services (no responses made)
By Internet:	DEVSUPPORT@applelink. apple.com (electronic mail messages can be sent any hour, day or night—the DSC will respond within 72 hours, starting the next business day)
By fax:	(408) 862-7602
By mail:	Apple Computer, Developer Support Center, 20525 Mariani Avenue, M/S 75-3T, Cupertino, CA 95014

European Developers Discuss Piracy Issues

As part of the company's worldwide antipiracy campaign, Apple Computer Europe recently hosted an antipiracy summit. At this meeting, 30 participants representing European developers, distributors, and industry associations shared a range of ideas about fighting software piracy.

On the whole, the Europeans agreed with their American counterparts about the importance of educating users about software copyright laws and the need to reinforce them with visible piracy litigation. Participants also discussed problems unique to Europe. These discussions centered on how to develop consistent policies and license agreements across vastly different countries and legal systems.

Some issues discussed by the group included:

•*User education.* Participants voiced support for Apple's efforts to educate users about software-copyright laws. Apple's most recent activity includes a mailing of piracy-education kits to user groups, consultants, resellers, and dealers. Apple also plans to include antipiracy information in its hardware manuals and in the Guided Tour HyperCard stack.

Participants felt, however, that increased awareness alone won't guarantee compliance by all users; for some, the fear of getting caught is the only useful incentive. They stressed the importance of the piracy litigation campaigns of the Business Software Alliance (BSA) and local antipiracy associations. And to increase the effectiveness of these campaigns, the developers talked about focusing activities on specific market segments.

•*Accessible demo software.* Software is often pirated by users whose initial intent was merely to "test drive" a software package. It was recommended that developers make it a regular practice to create and freely distribute demonstration software.

Developers can further reduce users' temptation to make illegal copies by placing demos on product disks. This would give users the option of giving friends or colleagues a crippled—but adequate for evaluation—version rather than the actual application.

•*Network protection*. The summit attendees favored user-counting server applications over other copy-protection schemes. This type of application assures that there are never more people using a given application on a network than are legally licensed. It was suggested that Apple could strengthen acceptance of this approach if it set standards for or built this ability into its software.

• *Need for worldwide consistency.* The group emphasized that software-protection policies should be adopted worldwide in order to avoid gray-market issues. The creation of a uniform standard software license, on the other hand, was rejected for several reasons.

First, radically different country-specific copyright laws make it difficult to create a consistent license agreement. Furthermore, some software developers don’t want uniform licensing, since they use site licensing as a competitive advantage.

Apple will use the results of this meeting to strengthen its European antipiracy campaign. In addition, the company is currently translating two educational brochures on piracy, one from Apple and the other from the BSA, into several different languages.

Developers who would like to explore more ideas for fighting piracy or to discuss their own ideas with other developers should participate in the antipiracy discussion on the AppleLink network (path–Developer Support:Developer Talk:Antipiracy Discussion.) For more information, see the antipiracy cover story in the March, 1992 issue of *Apple Direct*).

Dynamic Linking for MacApp

Many application domains naturally possess an unbounded set of desired features. A signal-processing application, for example, needs an unlimited number of filters. A graphics package, an unlimited number of sketching tools. Of course, no real application can present an unlimited number of anything to any given user. The problem for developers is that the market does appear to want an unlimited number of features. What's a developer to do?

Some developers have met this challenge by building some kind of extensibility into their applications. Using this extensibility feature, a user can choose, for example, which graphics tools to use.

Until now, MacApp, Apple's premier development tool, has provided no support to the developers who want to add extensibility to their applications. Dinker, the result of some recent work by Apple's Advanced Technology Group, is a dynamic-linking mechanism for the Macintosh OS. It enables you to write applications whose functionality can be extended at launch time by the addition of new Object Pascal, Object Modula, or C++ classes. The classes can be added automatically at launch time or, under user control, while the application is running. Once "dinked" in, a new class is virtually indistinguishable from a statically linked class.

A great deal of work has gone into making Dinker fast. Loading a new class into an application on a Macintosh IIfx, for example, takes less than 0.5 seconds. We believe that current dinking speeds will be acceptable to ordinary users.

In addition, for those who wish to encourage a third-party market surrounding their applications, Dinker provides a way for other MacApp developers to leverage off your application by developing extensions for it. To do this, these other developers will not need access to the source code of your application.

Dinker is included on several recent E.T.O. (Essentials, Tools, Objects) CDs (discs #6, #7, and #8) as a currently unsupported MacApp experiment.

CD Highlights,April ’92

This column is your guide to the latest Developer Series CD, telling you what’s new and notable. To quickly access everything listed below, see the “What’s New on This CD?” folder (located at Dev. CD April ‘92: What’s New on This CD?). Here you’ll find aliases to every new and updated package on the CD, including all those highlighted below. To suggest something for upcoming Developer CDs, send us the survey located in the Start Here folder of the current CD; we’ll do our best to include suggested items on subsequent discs.

The April CD, “Hex, Drives and Videotape,” includes the following highlights:

System Software: Included in our new international system software are System 7 Tune-Ups for German, Hebrew, Italian, Portuguese, and Turkish systems. You’ll also find the latest versions of Greek, Icelandic, Polish, Thai, and Turkish system software.

TrueType Font Strategy: This “white paper” is an overview of Apple’s font strategy. It includes background about the development of TrueType and its capabilities, as well as information about how TrueType fits into future strategies for Macintosh and other technologies.

N&C TMON Pro Macros/Templates: This package provides a TMON Pro 3.0 User Area for AppleTalk. The templates and macros simplify trapping on AppleTalk calls and provide templates for viewing the parameter blocks and associated data structures used by them.

World Database Stack: Use this HyperCard stack for making maps from a 50-megabyte database. It includes source code and file formats for accessing the data from Pascal.

Apple Events: Included this month are two revised tools for Apple events. The *Apple Event Registry* has been completely redesigned to be more useful and readable. Use it as a reference when implementing Apple events. This month you’ll also find the AEObject-Edition 1.0.2 Sample from DTS. This sample code demonstrates how to use the Edition Manager and the Apple

Event Manager, as well as some features of the Apple Event object model and the Apple Event object support library.

Disinfectant 2.6: This is the latest version of Disinfectant, a free antivirus utility for the Macintosh that detects and removes known Mac viruses. A protection INIT (extension) and comprehensive on-line manual are included.

More Technical Docs: This month's disc has Developer Notes for the new Macintosh LC II. It also includes the AppleTalk Release Notes 1.0 for AppleTalk and five new and updated Macintosh Technical Notes.

Antipiracy Toolkit: This contains information and tools to help you fight piracy in your organization. It includes the self-audit SPAudit software, ideas for combating piracy, information about software-copyright laws, an antipiracy awareness ad, and a guide to where to look for more information.

Brochure Focuses on Mac Development Advantages

Why develop for the Macintosh? You may know about the benefits of the Mac’s integrated hardware and software, the Toolbox, and our object-oriented development tools. But do you know that the worldwide installed base of Macintosh systems exceeds seven million units?

To help more people understand the advantages of developing on the Macintosh, the Apple Developer Group has produced the “Macintosh Development” brochure. It outlines many reasons that Macintosh development is good business, including:

•Macintosh has a market for your products: an installed base exceeding 7.3 million units; 60 percent growth in worldwide unit sales of Macintosh CPUs in 1991, versus 10 percent for the industry as a whole; 47 percent increase in worldwide unit sales of Macintosh applications in 1991, versus 24 percent for the industry as a whole (according to various industry analysts).

•Macintosh offers the richest platform: integrated hardware and software architecture; built-in system “toolbox;’ object-oriented development tools; future technologies including System 7 extensions, RISC/PowerPC, Taligent, and more.

•Apple supports your success with technical information and resources, technical support, localization and marketing support.

This brochure describes many aspects of Macintosh development that have been described in *Apple Direct* articles, the brochure “Blueprint for the Decade,” and developer services announcements on AppleLink. The new brochure will be distributed at industry events such as the Software Publishers Association Conference and Macworld Expo, and through focused developer-recruitment efforts worldwide.

To get more information or to receive a copy of the brochure, call the Developer Support Center at (408) 974-4897, or write to: Developer Support Center, Apple Computer, Inc,. 20525 Mariani Avenue, M/S 75-3T, Cupertino, CA 95014, USA; AppleLink: DEVSUPPORT.

Apple Begins New Retail Merchandising Program

As the breadth of its products and markets changes, Apple is entering new channels of distribution. Its stated plans for consumer-specific Macintosh products and other personal electronic devices include eventually entering into agreements with a variety of retailers such as mass merchandisers, consumer electronics stores, and office product superstores.

As part of a program in the consumer electronics channel, Apple recently signed an agreement with Sears. That company will sell selected Macintosh computers through its Office Centers. The program with Sears will initially run for six months and currently includes the Macintosh Classic II, LC II, IIsi, and PowerBook 140 models. These will initially be available in 70 Sears Office Centers.

Sears is one of a growing number of resellers that routinely hard-bundles third-party software as part of the hardware offerings. After considering numerous applications, Sears chose to bundle ClarisWorks with each of the CPUs sold through its Office Centers. To help developers who may wish to foster their own relationships with these new partners, Apple will keep you informed about its new channel relationships as they emerge.

Developer Resources

Get answers to many of your development questions, expand your knowledge, or find help for your programming projects by tapping into these resources.

Developer Associations. These groups can provide programming resources such as informational meetings, bulletin board services, technical tips and techniques, and contact with other developers to help with your development projects.

• *MacApp Developers Association.* Dedicated to supporting users of MacApp. For information, call (206) 252-6946.

• *SPLAsh Developers Association.* Dedicated to supporting users of Symantec programming languages. For information, call (415) 527-0122.

• *Macintosh Scientific and Technical Users Association.* Provides members with a forum for the exchange of information on Macintosh computer use in scientific and engineering applications. For information, call (508) 755-5242; AppleLink: CONS.LAB.MFG.

• *United Kingdom Developer Council.* Dedicated to helping support any Apple UK developer. For information, contact Paul Smith; telephone: Henley (0491)574295; AppleLink: PGSMITH.

• *Austrian Macintosh and Developer Association.* For information, contact Klaus Matzka, Ungargasse 59 A-1030, Vienna, Austria; AppleLink: AMDA.

• *The Netherlands - VAMP (Vereniging Actieve Macintosh Programmeurs - Association for Active Macintosh Programmers).* For information, conact VAMP, P.O. Box 11029, 2301 EA Leiden, The Netherlands.

• *Argentina - Get Info, Association for Argentinian Macintosh Developers.* For information, contact Jorge Sagasti, Zapiola 1625-1249, Buenos Aires, Argentina; telephone: (541) 551-4990.

Macintosh Development Services Directory. This directory of third-party development-related service providers can help supplement your development efforts. Experts are available, covering the entire development spectrum including technical trainers, contract programmers, product localizers, product testers, and technical documentation writers. To order, contact APDA.

Important Apple Phone Numbers

• *Ordering APDA products.* (800) 282-2732 (U.S.); (800) 637-0029 (Canada); (408) 562-3971 (other countries); AppleLink: APDA.

• *Apple Software Licensing.* For information on licensing Apple software for site, internal, or commercial distribution, call (408) 974-4667.

• *Developer Support.* U.S. and Canadian providers of commercial products and services can find out about Apple's developer support programs by calling (408) 974-4897. For non-U.S. support programs, see a complete listing in the latest *APDA Tools Catalog.*

• *Developer Training.* Apple Developer University offers course and self-paced training products for aspiring and advanced programmers. To obtain a course catalog or register, call (408) 974-6215.

• *Apple Consultant Relations Program.* Write us and find out how you can build solid relationships with the professional computer systems consulting community throughout the U.S. Address: Apple Computer, Inc., Apple Consultant Relations, 20525 Mariani Avenue, M/S 36-AJ, Cupertino, California 95014, USA.

• *U.S. Apple Authorized Dealers.* To locate a U.S. dealer near you, call (800) 538-9696.

Authorized Third-Party Developer University Training Sites. Selected Developer University courseware is available through regularly scheduled classes and, in some cases, on an on-site basis at the following training locations. Please contact these offices directly for information about classes, locations and fees.

• *EDS*, 5228 Tennyson Parkway, Plano, TX 75024;(214) 403-5261.

• *University of Maryland,* Computer Science Center, College Park, MD 20742-2411; (301) 405-2956.

• *Cornell University,* Cornell Information Technologies, 220 CCC Garden Avenue, Ithaca, New York 14853; (607) 255-4983.

• *University of Oregon,* Portland Center, 720 S.W. Second Avenue, Portland, Oregon 97204; (503) 725-3055.

Developers Conferences Outside the U.S.

Apple will sponsor developers conferences in seveveral countries during the upcoming months. Here are two of them.

Benelux

Apple will sponsor a developers conference in Antwerp, Belgium, on June 18, 1992. This event, targeted at developers, system integrators, and Macintosh consultants, will focus on Apple events, scripting, and marketing for small-to-medium-sized businesses.

For more information, contact Bernadette Mergaerts at Apple Computer NV SA, Rue Colonel Bourg Straat 103, 1040 Brussels, Belgium; phone: 00 32 2 741 23 61; fax: 00 32 2 735 76 19; AppleLink: MERGAERTS.B.

France

June 22–24, 1992, Apple will host Journées Du Développement (JDD) 1992, the France developers' conference, at the Casino de Deauville, in Deauville, France. Approximately 40 hours of sessions will focus on a variety of development topics and issues.

For more information, contact Louk Janssen, 12, Avenue d'Océanie, Z.A. de Courtabœuf, 91956 Les Ulis Cédex, France; phone: 00 33 1 69 86 34 74; fax: 00 33 69 28 74 32; AppleLink: JANSSEN.LOUK.

QuickTime CD Correction

Last month’s developer mailing contained a CD that included the QuickTime *Electronic Apple Intro News* and the "CD-ROM Titles Sampler." The QuickTime movies that were to be part of the QuickTime Apple Intro News Stack were mistakenly omitted from the CD.

The stack will launch, but if you double-click on the QuickTime "badge" within the movie frame, you will receive a "no such window" error message.

If you supplied QuickTime movies for the CD, be assured that the pressing which was distributed to customers and resellers was done correctly. Developers will receive a corrected CD in this month's mailing. In the correct version, the words "The Movie" follow the "Apple Intro News" title on the CD label.

Our apologies for this error.

It Shipped!

Through the “It Shipped!” program, you can announce new and revised third-party products in *Apple Direct.* It Shipped! listings are also made available on the 3rd Party Connection AppleLink bulletin board. You can obtain an It Shipped! application by downloading it from the AppleLink network (AppleLink path—Developer Support:Developer Services:Apple Information Resources:Developer Program Information:It Shipped! Program). Or contact Todd Luchette at (408) 974-1241 (voice) or (408) 974-3770 (fax). Once you’ve completed the application, send it to Engineering Support, Apple Computer, Inc., 20525 Mariani Ave., M/S 42-ES, Cupertino, CA 95014; Attn: It Shipped! Program. Or send it via AppleLink to IT.SHIPPED.

Optionally, you may wish to send us a copy of your product to be placed in the Engineering Support Library, where it may be checked out by Apple’s testing groups for compatibility testing, or by research and development employees for evaluation. If you would like your product(s) to be included in the Engineering Support Library, send them to the address above. The following products shipped in February 1992:

Publisher	**Product (Version)**
U.S.A.	
Bliss Interactive Technologies	Resource Library Vol. I Public Domain Color Pictures & QuickTime Movies-CD (1.0)
Bowers Development Corp.	AppMaker (1.5)
CE Software	Tiles (1.0.1)
Comstat DataComm Corp.	StatLoader Mac (1.03)
Coyne Company	MacHam Radio Advanced (1.0)
	MacHam Radio Extra (1.0)
Dycam, Inc.	Dycam Model 1 (2.1)
Intelligence At Large	Mac Programming 101: Apple Events (1.0.1)

	Mac Programming 101: Publish & Subscribe (1.0)
	Macintosh C Programming Primer Disk—Volume I (2.0)
	Macintosh Pascal Programming Primer Disk—Volume I (1.0)
JK Blanchard Enterprises	The BlackJack Tutor & System Checker (2.0)
Kent Marsh Limited	NightWatch II (2.01)
Microlytics, Inc.	Word Finder Plus (4.0)
MicroMaps Software	KidMaps (1.0)
Neon Software, Inc.	NetMinder LocalTalk (1.1)
Oracle Corp.	ORACLE for Macintosh (2.0)
Procom Technology, Inc.	GigaTower II (NA)
Psybron Systems, Inc.	calenDAr (1.1.3)
	Contact! (1.0)
Sunrise Software	Zap Phone Spud version (1.1)
Type Solutions, Inc.	The Incubator! (1.00)
Videomedia, Inc.	Auto-PICT QT (2.0)
Working Software, Inc.	Findswell (2.2)
	Lookup (2.2)

Now available from Apple

The following list shows which APDA products have become available to developers within the last several weeks. To get a full listing of all APDA products, check the current *APDA Tools Catalog.* For new-product announcements and the most up-to-date price lists, check AppleLink (path—Developer Support:Developer Services:Apple Information Resources:APDA—Tools for Developers).

If you're interested in the latest version numbers of all Apple system software products, check "Latest Rev" in the Information Resources folder on the current Developer CD. Latest Rev also tells you where to obtain those system-software products. In addition, the "CD Highlights" section on page 3 of this issue tells you which new system software releases appear on the current CD.

DAL Developer's Toolkit Bundle v.1.3.5
B0582LL/B
$695.00

DAL Server for MVS/TSO Bundle v.1.3.5
B0579LL/B
$20,000.00 (contact Apple Software Licensing)

DAL Server for VM/CMS Bundle v.1.3.5
B0578LL/B
$15,000.00 (contact Apple Software Licensing)

DAL Server for MVS/VTAM Bundle v.1.3.5
B0581LL/B
$30,000.00 (contact Apple Software Licensing)

International DAL Server for MVS/TSO Bundle v.1.3.5
B0667LL/B
$20,000.00

International DAL Server for VM/CMS Bundle v.1.3.5
B0669LL/B

$15,000.00

International DAL Server for MVS/VTAM Bundle v.1.3.5
B0668LL/A
$30,000.00

LaserWriter IIf and LaserWriter IIg Printers Developer Note
R0230LL/A
$20.00

MacsBug v.6.2.2 (disk only)
R0065LL/C
$15.00

MacsBug v.6.2.2 (manual and disk)
R0064LL/B
$34.95

SourceBug v.1.0.1
R0228LL/A
$100.00

APDATOPTEN SELLERS*

1. E.T.O. Starter Kit & Subscription
2. QuickTime Developer's Kit v. 1.0
3. MPW C v. 3.2 bundle
4. Macintosh Common Lisp v. 2.0b1
5. DAL v. 1.3 VM/CMS Server
6. Macintosh Programming Fundamentals v.1.0.1
7. MPW C++ v. 3.1
8. MPW C & Object Pascal bundle
9. MacTCP v. 1.1 Developer's Kit
10. Inside Macintosh, Vol. I-VI + X-Reference

*as of 3/18/92

To place an APDA order from within the U.S., contact APDA at (800) 282-2732; in Canada, call (800) 637-0029. For those who need to call the U.S. APDA office from abroad, the number is (408) 562-3910. You may also send an AppleLink message to: APDA. If you’re outside the U.S., you may prefer to work with your local APDA contact. For a list of non-U.S. APDA contacts, see the “International APDA Programs” page in the *APDA Tools Catalog.*

Creating Software That Sells, Part 2

In the mid-1970s, television manufacturers began introducing the first all-electronic tuners, replacing the clunky mechanical tuners of old. They supplied them initially on their TVs with remote control, because the remote controls could then also be purely electronic, suddenly making remote-control TVs with electronic tuners practical. Gone were the heavy relays and motors necessary to mechanically turn the tuner knobs.

A consumer electronics store in San Francisco, Village Electronics, was selling remote control TVs to the industry-average 20 percent of their customers at this time—but it was an uphill battle, for the first mention of remote always produced the exact same "tape recorded" message from the prospective buyer: "I'm not so lazy that I can't get up off the couch to change the channel." The swift retort "Oh yes, you are!" rarely seemed to produce a useful result.

The owner of the store, unhappy with the weak profit margins on the manual TVs, decided to work up a new sales technique for "moving" remote-control TVs, one based on an emerging branch of psychology called Transactional Analysis. Within two weeks, the store's product mix had gone from 20-percent remote control to 80-percent remote control, along with an overall increase in sales.

TRANSACTIONAL ANALYSIS

Transactional Analysis, or TA, was developed as a discipline by Eric Berne beginning in the late 1940s. It started gaining popularity in 1964 with the publication of Berne's book *The Games People Play,* followed quickly by Thomas Harris' book *I'm OK, You're OK.* By the mid-1970s, it was quickly becoming a central tenet of motivational psychology, that wicked step-sibling of clinical psychology dedicated to getting people to buy things they had no intention of buying.

Transactional Analysis is a 20-dollar word for a very simple idea on Berne's part: Since very few clinical psychologists are mind readers, why not stop trying to peer into people's subconscious minds and instead concentrate on what they say and how they say it. If you say "hello," and I say "hello" back, we have just conducted a transaction. Eric Berne came up with such a highfalutin' term solely because he was a Freudian psychoanalyst and he needed long, long words for simple ideas in order to get his colleagues to read his papers.

I cover some of Berne's early research into human intuition in chapter 15 of my book, *Tog on Interface.* It is his fundamental model of the human ego that the San Francisco storekeeper made use of in increasing the profitability of his TV sales; it is this fundamental model that you can make equally good use of to increase your own product's sales.

THE THREE EGO STATES

Berne observed that people have three states of mind among which they move frequently and—in the case of normal, healthy people—fluidly. He named these states Parent, Adult, and Child.

The Parent ego state is the source of our tape-recorded messages: "Don't throw that [any object smaller than four feet in diameter], Billy, you could put an eye out." "Sara, how many times have I told you not to..." "Why? Because I said so!" The Parent ego state can be either nurturing or critical. As Berne describes it, *Parent* means being "in the same state of mind as one of your parents (or a parental substitute) used to be, and you are responding as he would, with the same posture, gestures, vocabulary, feelings, etc."

The Adult ego state is pure logic: "Just the facts, Ma'am." It is the state of mind we are most likely to design our products for if we assume that our user population is adult. The Adult ego state responds well to spec sheets and dispassionate sales pitches. In Berne's words: *Adult* means "You have just made an autonomous, objective appraisal of the situation and are stating these thought-processes, or the problems you perceive, or the conclusions you have come to, in a non-prejudicial manner."

The Child ego state is the site of both our playfulness and our tendency to pout. It is, paradoxically, the oldest of our ego states. Being in the Child state, according to Berne, means, "The manner and intent of your reaction is the same as it would have been when you were a very little boy or girl." The Child ego state is the one in which adults are most likely to deny spending any time.

Adult males, in particular, often go to extreme lengths to hide their childlike thoughts and actions by limiting their play to such socially acceptable outlets as golf, tennis, and drinking. They invent elaborate pretenses and nonsense to create the illusion of always being in the Adult state. Believing such nonsense can cost developers a lot of sales. Read on.

HOW TO SELL REMOTE CONTROLS

To the San Francisco storekeeper, the consistency with which patrons announced the exact words “I’m not so lazy that I can’t get up off the couch to change the channel” was a dead giveaway that they were speaking not from their Adult ego state but from their Parent ego state. Therefore, the storekeeper reframed the sales problem as this: The Child ego state will want the remote; the Parent will do his best to block any spending on such frivolity; and the Adult ego state can, if convinced, swing the deal.

(Keep in mind that even though this discussion may sound as though we are dealing with three separate individuals locked up in a single head, we are really talking about three different states—frames of mind—of a single, integrated individual. TA, like most newly encountered theories, can seem a bit strange at first. One does grow quite used to it with time, and it is remarkably powerful. Bear with me.)

The key, as the storekeeper saw it, was to keep the Parent out of the discussion until he had won over the Adult. Since the Parent was constitutionally incapable of keeping his opinion to himself once the remote had been revealed, the store owner would keep the remote control in his back pocket, with the customer’s Parent ego state blissfully unaware of its presence.

Then he would deliver a ten-minute sales pitch on the value not of having a remote control but of having an all-electronic tuner. No more biannual tuner cleanings at $35 a pop. No more trying to jiggle the control just right so the station comes in. No more having to fuss with a second, separate tuner for UHF, with the strange human interface it used to entail.

He then extolled the virtues of an electronic volume control, which would not produce horrifying static noises after a few years of use, and all the rest of the controls—color, hue, brightness, and contrast—that would no longer randomly change settings after a few years of aging.

By the end of his pitch, he had built more than the $100 price differential into the electronic control—the savings in tuner maintenance alone would amount to $175 over the life of the set. He had the Adult, if not sold on buying the more expensive set, at least convinced that it ultimately represented the better value.

And then it was time to play his trump card: He pointed to a tiny disk on the front of the set and said, “And now that they have replaced all the mechanical contraptions of old with pure electronics, they can put this little thirty-cent photocell in the front of the TV so they can give you this,” at which point he would slap the remote control into the customer’s hand.

The waiting Child ego state would go wild and start pressing all the buttons on the remote control while the store owner now explained, to the Parent, that the value in the remote control lay not in changing channels every half hour but in muting the 180,000 commercials to which the prospective owner and any children would otherwise be subjected over the next ten years.

The only time his pitch ever backfired was when he sold a set after inadvertently failing to mention the remote control at all! The customer paid for the TV and was quite happy until she saw someone inserting batteries into a strange plastic wand that came with her new set. When she found out the wand was a remote, she exclaimed—and I quote—“I’m not so lazy that I can’t get up off the couch to change the channel!” It took another 15 minutes to resell her on the value of the electronic tuner.

HOW TO SELL SOFTWARE

When the storekeeper a couple years later became an Apple dealer, he discovered that he had the same three-in-one customers coming in to buy computers. Their initial line was always the same: “I don’t want to play any games. I want this for business.”

He would always agree with his customers and then watch in amusement as they fought an unsuccessful battle not to be attracted to any of the store computers at which neighborhood kids were engaged in a battle with intergalactic warriors. The customers would almost invariably walk out with at least one game among their purchases.

Your software is bought and used by people who have a childlike side to their personalities. I don’t care if you sell statistical-analysis packages to aging government accountants. Inside those gray-suited and bow-tied bodies are eight-year-old kids just dying to get out. If your view of the person who buys and pays for your software is that of an adult with a data processor for a mind, you need to expand your outlook.

The storekeeper retired to Apple in 1978 and soon teamed up with J. D. Eisenberg to write “Apple Presents...Apple,” the keyboard introduction to the Apple II (see *Tog on Interface,* chapter 14). They had in it a small game in which the user helped a rabbit find his carrot. The marketing people said that such a babyish character wouldn’t work for business types. The storekeeper added an alternative character with an alternative task: help a Swiss banker find his gold. He then presented the users with the following introductory, branching question:

“Would you rather come to the aid of a rabbit or a major financial institution?” He later tested that program on more than 20 CEOs of large companies. Every last one of them chose the rabbit.

Last month, I talked about the features of Painter that make people go wild, whether they are snatching it up in a store or writing a glowing review. Most of those initial-reaction features appeal not to the Adult but to the Child. If you want to design and package software that sells, you will consider that inquisitive, fun-loving Child who is standing just inside the stuffy guy in the suit. Just look at the success of a product such as After Dark, created ostensibly to fulfill an adult (mainly parental) concern but selling in such high numbers because of its appeal to the child within us all.

Writing to appeal to the Child does not entail upsetting the Parent. The greatest fun of all to children is success. Design software that produces immediate, if small, successes, and the world will beat a path to your door. Play with Painter, play with Vellum, and see how quickly you gain a sense of power and control. Then capture that same quality in your own products.

Until next month, I remain—TOG: Retired San Francisco storekeeper.

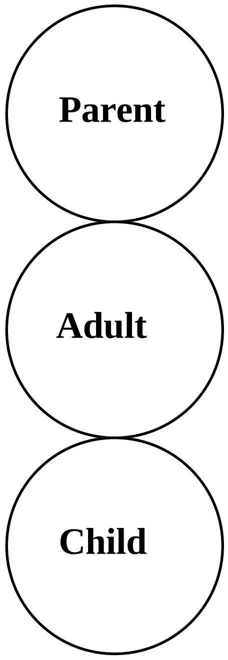


Figure 1: The three ego states

The ** indicates the trade shows/events at which Apple Computer, Inc., is scheduled to exhibit as of press time. This list may be incomplete. If you have information about a show you want listed here, write to Apple Direct Event Calendar, 20525 Mariani Avenue, Mail Stop 75-2B, Cupertino, CA 95014. For further information, check the Events folder on AppleLink (path—Developer Support:Developer Services:AppleInformation Resources: Developer Events).

April 21 through 23
EUC Conference (European University Consortium)
Bruges, Belgium
Contact: Veerle Vandereecken
32-2-741 22 11
Fax: 32-2-741 22 42

April 23 through 25
****The Mac Show,** Tampa, FL
Contact: Libby Barland
(215) 540-9111
Fax: (215) 628-0882

April 23 through 26
****Macintosh Consultants Conference,** Boston, MA
Contact: Mac Consultants Network (209) 545-0569

April 25 through 27
NSBA - Nat’l School Boards Assn., Orlando, FL
Contact: Teresa Dumouchelle
(703) 838-6722

April 28 through May 1
****Computer 92,** Lausanne
Contact: Kathrin Mäder
Phone: Switzerland 1 832 81 11
Fax: Switzerland 1 830 63 06

May 3 through 8
****IRA - Int’l Reading Assoc.**
Orlando, FL
Contact: IRA (302) 731-1600

May 5 through 8
****Worlddidac**, Basel
Contact: Kathrin Mäder
Phone: Switzerland 1 832 81 11
Fax: Switzerland 1 830 63 06

May 11 through 15
****Worldwide Developers Conference**
San Jose, CA
Apple Computer, Inc.
(408) 974-4897

May 12 through 16
****IFABO,** Vienna A-1020
AppleLink: A.MARCOM or BONENGL.L

May 13 through 15
****GTC West,**
Sacramento, CA
Contact: Dave Draxler
(916) 452-4902

May 18 through 22
****INTEROP**
Washington, DC
Contact: Debbie Moessinger
(415) 941-3399
May 28 through 29
****QuickTime—The Conference & Film Festival**
San Francisco, CA
Contact: Macworld
Communications
(415) 267-1745
Fax: (415) 422-0766

May 31 through June 3
****SPA European Conference**
Cannes, France
Contact: Software Publishers
Association
(202) 452-1600

June 1 through 5
****Object Expo New York,** NY
Contact: G.G. Schafran
(212) 274-0640

June 8 through 11
****AEC Systems,** Dallas, TX
Contact: Kelly Baxter, AEC
Systems
(215) 444-9690

June 8 through 12
DAC—Design Automation

Anaheim, CA
Contact: Marie Pistilli
(303) 530-4562

June 10 through 13
****Logic Zurich, Apple Expo**
Contact: Kathrin Mäder
Phone: Switzerland 1 832 81 11
Fax: Switzerland 1 830 63 06

June 15 through 17
NECC—Nat'l Educ. Computing Conference

Dallas, TX
Contact: Sharon Finke
(503) 346-3537

June 23 through 25
PC Expo, New York, NY
Contact: H.A. Bruno, Inc.
(201) 569-8542

July 14 through 17
Object Expo, London
Contact: G.G. Schafran
(212) 274-0640

July 26 through 31
SIGGRAPH
Chicago, IL
Contact: Barbara Voss
(212) 752-0911

August 4 through 7
****Macworld Expo**, Boston, MA
Contact: Mitch Hall & Assoc.
(617) 361-8000

Finding Your Voice
A Public- Speaking Primer for Developers

by Ramond Nasr, Apple Computer, Inc.

You've spent months preparing your product for market. All the pieces are in place—the distribution and marketing plan, production schedules, the PR and advertising strategies, your opinion leaders campaign—and now you're getting requests to speak at various meetings and functions. How persuasively you deliver your messages can quite easily dictate the future course of your organization.

Regardless of the kind of forum, speaking publicly is not only a powerful marketing tool, but it's also an excellent PR opportunity for your company. It's an occasion to convey that you and your company are knowledgeable, interesting, and worth listening to—and gives the impression that you are in demand. (Consider how important this can be for a startup!) Putting your best foot forward on these occasions is critical to advancing your company's image; you'll not want to take any chances.

Woody Allen once said, "Eighty-five percent of life is just showing up." This is also true of public appearances, but you'll want to make sure that you're doing the right thing during the other 15 percent of the time. Whether you write your own speeches or work with a speechwriter, it's important that you find "the right voice" for each occasion. This means tailoring your words and developing your messages to meet the audience's expectations.

Delivering powerful speeches takes a lot of practice, both in preparing and in presenting them. Even veteran speakers often return to the basics to make sure they are on target. To help you "find your voice," I'd like to offer a few basics, or rules of thumb.

GENERAL RULES OF THUMB

There are several important, yet fundamental, things that will help make your speaking engagement a success. First, a speech is designed to be heard, not read. Keep in mind that you are writing for the ear, not for the eye. Accordingly, like a piece of music, speeches should make full use of rhythm, cadence, and punctuation, and have the same kinds of interludes, such as pauses, crescendos, and tonal shifts. Also, it helps to use simple, graphic, and concrete language that lands easily on the ear.

Another simple rule: Make sure the objective of the speech is clear and relevant to the audience; tailor every speech to the audience and occasion. A "boilerplate" speech is seldom successful. Every audience has different expectations and levels of familiarity with the speak- er and the subject. Although the main messages may remain con-sistent from speech to speech, the ways they are expressed will vary with each audience. In every case, knowing an audience's ex-pectations is the single factor that puts you into a position to exceed those expectations.

Those rules of thumb apply to every kind of speech you can make. However, every occasion also demands a special touch. What follows are some tips to help you find the right voice for each speech you'll give.

THE SPEECH OF INTRODUCTION

When you are introducing another speaker, the speech contains three main points: the other person's name, title, and the subject of his or her speech. Of course, no one ever wants to stop here, and it would be pretty boring if they did. Audiences want to hear about the personality of the speaker you're introducing—and in small doses this frequently makes the speaker feel more welcome and gives the audience some helpful background. Here are a few things to keep in mind when preparing a speaker introduction:

•Keep it brief. "Brief" here means 30-45 seconds. As a courtesy to the audience and to the speaker you are introducing, the less time spent introducing the keynote speaker, the more time there is for the rest of the program. This is especially true of evening or banquet-style events, which should never run later than 10:00 p.m. This is your chance to keep the program on schedule.

•Remember who they came to hear. The audience came to hear the keynote speaker, not your introductory remarks. Make sure that whatever you say is low-key, understated, and not overly dramatic; it's not your show.

•Try to avoid clichés. By using such phrases as "...without further ado..." or "...the next speaker needs no introduction...," you risk coming across as being insincere or too lazy to prepare something thoughtful to say. Instead, try to diplomatically convey how delighted you are that the speaker has accepted the

invitation to speak. You can do this by beginning with a comment like, “I know we’re all interested in hearing what our guest has to say, so please join with me in welcoming...”

•*Never preempt the speaker’s subject.* It’s rude to do this to your guest. And never read the speak-er’s résumé; it’s okay to include a few highlights from the bio- graphy—in fact, it’s almost required—but do so briefly, in only the most narrative of ways.

CEREMONIAL SPEECHES

When giving this kind of speech (such as for dedications, promotions, site expansions, farewells, and so forth), try not to burden the audience with troublesome—or taxing—ideas. This is an important event, and you don’t want to run the risk of bursting the ceremonial bubble by talking about an indiscreet subject (that is, stay away from politics, religion, and money). A few guidelines work for all types of ceremonial speeches:

•*Pour on the praise.* Don’t hold back. For example, if the event is to commemorate Henry’s 25 years of service to the organization, then go ahead and comment on Henry’s courage, endurance, and commitment to quality. “In my profes- sional life, I’ve met very few people who are as conscientious as Henry...”

•*Back it up with specifics.* Speaking concretely about Henry’s accomplishments demonstrates that you actually know something about him.

•*Personalize.* Offer anecdotal comments that throw light on the person and what he means to you.

Of course, this is difficult to do if you don’t actually know the guest of honor. But here’s an easy solution: “I’ve not had the privilege of knowing Henry until this evening, but his supervisor said...” Saying something like that shows that you have taken the time to get to know a few things about Henry.

PRODUCT INTRODUCTIONS

Ideally, months before the new-product launch, the marketing department will have “fleshed out” the key marketing messages for your new products. Once these messages are carved into stone, here are four key ideas for your product-introduction speech:

•*Keep it snappy and inspired.* The most important concern is to transmit your enthusiasm not only for the specific products but also for the organization's long-term vision.

•*Don't make the statesman a "demo dog."* As Guy Kawasaki comments in *The Macintosh Way,* "Have the person who knows the product the best, not the highest-ranking, do the demo. Just because a president or vice-president is at the event doesn't mean he has to do the demo."

•*Invite a host of luminaries to comment on how great the product is.* It is always better to hear the words "This is a spectacular product" from a credible industry commentator than from a person representing the company who made it.

•*The audience should "glow and tingle."* The single most important objective is to make audience members feel as if they cannot live without the product. When the speech is over, they should want to do nothing but get their hands on it and feel the magic themselves. (If you'd like more information about this, see Paul Sherlock's book *Rethinking Business-to-Business Marketing.*)

ANALYST MEETINGS/ REMARKS TO SHAREHOLDERS

There's a formula for both of these kinds of speeches: The chairman delivers the "big picture" overview presentation, the president gives a report on the operations side of the business, and then the chief financial officer delivers the numbers. This is what the audience expects, so try your best to meet those expectations; of all the audiences in the world, the investment community is not one to play games with.

The first two speakers—the chairman and the president—should be honest, optimistic, and confident about the future. The audience expects the CFO to give the numbers, and to illustrate his or her remarks with lots of charts.

Also, make sure you are up to speed on the Security Exchange Commission rules that constrain these types of presentations. As one would expect, there's not much room for creativity. The speeches may sound boring, but remember, the audience wants to hear about the return on their investment; they didn't come to be entertained.

You can make these presentations a bit spicier by adding a product demonstration to the speech, but again, the challenge is to create a positive feeling about the company's competitive position without sounding overly optimistic about the future. The analysts want consistency, and the press wants the latest scoop. That's a tough combination to deliver in a single speech.

THE VIDEO PRESENTATION

The videotape presentation—in lieu of an actual speaking appearance—has become commonplace on the executive presentations circuit. It saves you the hassle of travelling to Salt Lake City to address the National Association of State Directors of Transportation Districts—and it still gives your company a presence there. (The secret is, of course, knowing when you should be there and when a video will suffice.)

Again, meeting the audience's expectations is paramount. People seated in front of video monitors expect a Peter Jennings-style of address, not a 20 minute keynote-style speech. Also, as Edward R. Murrow once commented, when television is used effectively, it can educate, enlighten, and even inspire. When used unwisely, it is nothing but "a box of lights and wires." So here are a few tips:

•*Brevity is the viewer's friend.* The longest you ever see Peter Jennings on the screen at one time is about a minute. No one wants to hear more than one minute of a talking head, primarily because they're not used to it.

•*Use the tricks of the TV trade.* A good director, a tightly-written script, and a telegenic speaker can make a three-minute spot a blockbuster. Changes of camera angle, charts or visuals, and a video "field report" effectively break up the talking-head portions of the video and give the production a professional look.

•*Conduct teleprompter-reading practice sessions.* Many speakers are not familiar with the teleprompter. If you're not given several opportunities to practice, you can appear awkward, stilted, and uncomfortable on the screen. If you feel ill at ease, then it is very likely that the audience will, too.

THE KEYNOTE ADDRESS

A great political philosopher once said that, "Talent is hitting the target that everyone knew was there. Genius is hitting the target that no one knew existed." If you want to deliver a presentation that is "a work of genius" (even at the most basic level), there are few helpful suggestions that will help you structure your keynote address.

First, keep it brief. Blaise Pascal once said, “I apologize for writing such a long letter, but I didn’t have time to write a short one.” The same holds true for speeches. Speeches measured by the hour die by the hour. The Gettysburg Address ran for 3 minutes and 10 seconds, and yet it advanced some of the most powerful ideas in the history of humanity. Generally, if you can’t reduce your remarks to 20 minutes, people will perceive that you don’t know what you’re talking about. Keep it simple, succinct, and entertaining.

Next, every keynote speech should have these three elements: an introduction (including the thesis), the body, and a conclusion.

The Introduction. This part of your talk should be witty and brief and should draw the audience into the speech. This is a perfect opportunity to win audience respect and trust. However, breaking the ice is often the most difficult hurdle for a speaker. For tips about how to do this, see the article below, “How To Break The Ice.”

The introduction should also include your thesis—the single most important idea the audience should take away from the speech. State it simply and with conviction. Here’s an example of how John Sculley presented his thesis in a 1989 speech to school-district superintendents: “The world is undergoing dramatic change, where the economy is going to be, in many ways, very different from anything we have experienced in the past. *Today, I’d like to give you a perspective of this changing world from the vantage point of a CEO running a corporation, that may add to your perspective as leaders and CEOs of your school districts...”*

The Body. The body of the speech should begin where the thesis statement left off. It should contain:

•*The main points.* These serve primarily to corroborate the thesis (no more than three main points, please).

•*Amplification of the main points.* This is basically an expansion or special treatment of the main points.

•*Materials that support (or prove) the main points.* Just as a lawyer in a courtroom offers a proposition and supports it with concrete evidence, you should define your thesis and follow it with specific supportive ideas. Support can come in many forms, such as examples, illustrations, metaphors, anecdotes, statistics, and quotations.

Although I won’t elaborate on each of these, I’d like to say just a few words about numbers and quotations. Round off numbers, always voice them to your advantage, and use them frugally. For example, instead of saying, “four-point-seven percent,” it is better to say, “five percent.” Or instead of saying, “Twenty percent of the U.S. population...,” say, “One in five Americans...” Also, be careful with numbers. Most people come to speakers’ forums for entertainment, not number crunching. It’s easy to lose an audience in the numbers.

Using quotations and expert opinions, especially when they’re not expected, adds a refreshing dimension to public presentations. A few rules for using quotations: Never use more than three per 20-minute speech; they are to be savored and delivered with reverence; and a good quote is usually pithy—the audience should be able to absorb it in one earful.

•*Transitions.* Transitions between main points (such as “in addition, in contrast, on the other hand, yet, additionally,” and so forth) let the audience know that you are moving from one idea to another. They define the beginning and end of the “modules of thought,” indicate the relationships between the main ideas, and remind you that you are moving from one idea to the next.

The Conclusion. Once you’ve said, “In conclusion” or “To summarize,” the audience will stay with you for about another minute. This is the perfect opportunity to help the audience recall your main points. There are several ways to end the speech in a short, punctual, and entertaining way. You can restate the core idea, challenge the audience to action, offer a closing quote or story, or use any of several techniques. I also recommend that you end with “Thank you very much.” Although some people feel this isn’t necessary, I always use it. If nothing else, it lets the audience know when to clap.

Whether you’re a seasoned veteran on the speaking circuit or are just starting to find your public voice, applying some speechwriting basics and really paying attention to the needs of your audiences will make the difference between a memorable speech and a humdrum one.

A final word: If you can wield any influence over choosing the menu for the banquet, make sure they’re serving something good, because, “A well-fed audience is always less critical.”

Raymond Nasr is an executive speechwriter in Apple Computer's Corporate Communications division. This article was adapted from his presentation at Stanford University's Graduate School of Business, where he teaches a workshop on corporate speechwriting.

****

How to Break the Ice

Mikhail Gorbachev once said that the most difficult part of perestroika was knowing where to start. Similarly, launching into a speech is often the most difficult challenge a speaker faces. Here are a few tips that will help you break the ice and begin to "soften" the audience:

•**The honest compliment.** Finding something that the speaker genuinely appreciates about the audience can be a powerful force. For example, John Sculley delivered a speech in Washington state a few years ago, in which he commented on how delighted he was to be in the apple capital of the world. "Although we could never compete with the number of apples you ship out of your state each year, we're giving it our best shot..."

However, be careful how you use this tactic. After all, as one speaker once pointed out, "A pat on the back is only a few vertebrae higher than a kick in the butt..."

•**The common bond.** This is an easy way to establish empathy with the audience, provided the common bond is genuine. The speaker can say, "I happen to have a similar enthusiasm and respect for the Future Farmers of America. I grew up on a farm in Iowa and still have several close relatives living there..."

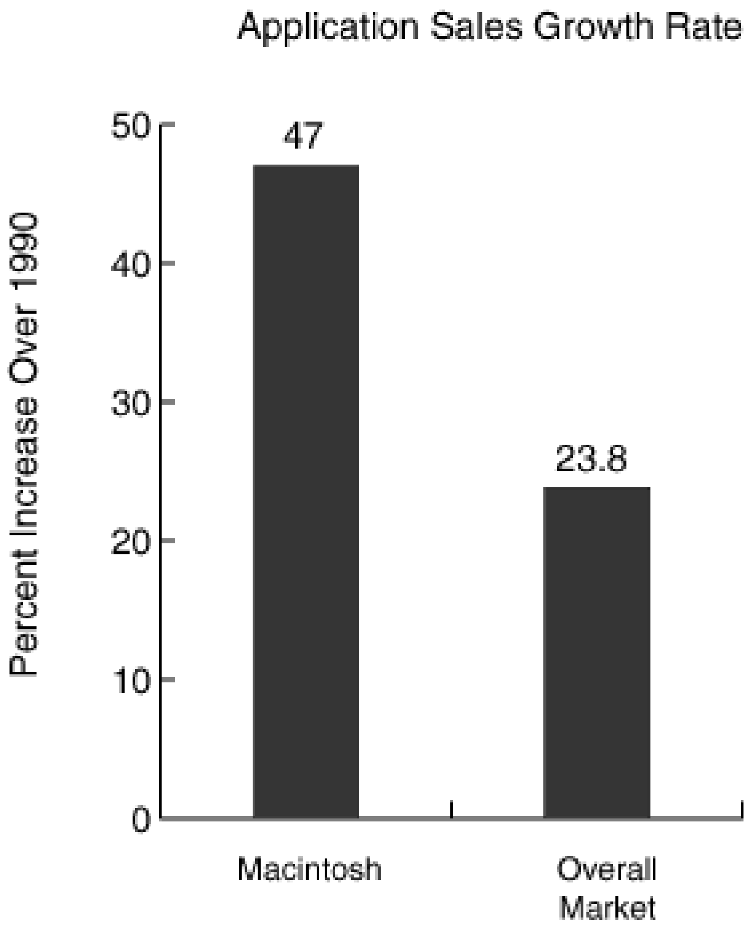
•**Humor.** Effective use of humor can put an audience at ease. Here is an example from David Rockefeller, then chairman of the Chase Manhattan Bank, speaking at the Commonwealth Club of California: "This is the second time I have had the good fortune to address the Commonwealth Club of California which, if little else, certainly testifies to your indomitable courage!"

•**Surprise openings.** You can use a device, prop, or something unexpected to grab the attention of the audience. It's risky, though, because if it's not "just right," it can fall flat.

Before using something, try it out on a few people to see if it works.

- **Illustration.** Starting with an attention-grabbing story builds a “curiosity” factor in to the speech. It’s also fun to weave a story in and out of the speech as it progresses.

Macintosh Software Sales Increase 47 Percent



In the January through November period of 1991,* Macintosh application-software worldwide unit shipments increased more than 47 percent over 1990, more than twice the rate of the overall market-growth rate of 23.8 percent.

Also, sales for Macintosh international software applications were at their highest levels in November 1991. Unit shipments were 54.4 percent higher in 1991 than in 1990. Approximately one-quarter of 1991 Macintosh software revenues came from international sales.

Source: Software Publishers Association

*December data not available at press time.

Taking Your Application to a New Platform
Do It for the Market–The Mac Way

by Yuko Takagi, Lotus Development Corporation

When you move a product from one platform to another, you'll be faced with a variety of choices and challenges, as we were when creating Lotus 1-2-3 for Macintosh. Questions arise from day 1 and continue cropping up every step of the way: How do you get started? How can you preserve cross-platform compatibility (always a primary concern)? How should you create and then validate the effectiveness of an interface? As you can imagine, we had plenty of ups and downs while trying to meet these challenges.

But in retrospect, some things clearly stood out as the Right Thing To Do. To help you along your way, here is a very informal set of guidelines, based on our recent experience, for your cross-plat-form projects.

This isn't intended to be comprehensive coverage of the topic; just look at it as pointers from a group of people who have been down that road. Also, note that even though our experience was DOS-to-Mac (and this article focuses on that particular process), many of these suggestions can apply to developers moving in either direction.

DO IT FOR THE MARKET

One of the first things you should examine is the target audience for the product. For many developers, the impetus for creating an alternative version of their product comes from the needs of their installed base. For example, some large customers for your DOS product may also have a lot of Macintosh computers. Your initial impulse may be to create a Macintosh version primarily to give those customers maximum use of your product.

But you should think more broadly than that. If you're going to the time and expense of re-creating your product on another platform, you ought to scrutinize whom you really should do it for.

Our suggestion: Look beyond your current users, and do it for the broader market. A good application that is moved from one platform to another—and that preserves all the qualities that made it successful in the first place—should and will have a much wider appeal than to only your installed base. Done properly, the new version can provide your company with opportunities to expand into totally new markets. The result is a significantly greater reward—an increased return on investment from your platform-crossing effort.

ADJUST YOUR MIND-SET

Once you've decided who your target market is, you may be eager to launch right into the design work. *Fight that impulse.* Before beginning any design work, it is absolutely critical that you make some mental preparations.

No matter what computer(s) are on your desk, begin immersing yourself in the ways and means of your new platform. Start getting into the mind-set of potential users. It's not just the technical details you need to absorb, but also the "spirit" of the platform. Get it into your blood.

As part of that experience, try learning a new Macintosh application. Pick an application type that you are unaccustomed to using and that is different from what you plan to develop. The point is to put yourself in the shoes of someone who is using a new application for the first time. Take note of where you get stuck or where things go very smoothly. You'll learn a lot by observing your own behavior.

Drawing and painting programs are particularly good candidates for this exercise, since they make full use of Macintosh graphic-interface conventions. But whatever you choose, make sure it's a leading application that has a good reputation in the Mac community. Some titles we found useful were MacWrite and MacPaint from Claris, and SuperPaint from Silicon Beach.

Another important step is to do your reading homework. Three indispensable books are *Apple Human Interface Guidelines: The Apple Desktop Interface* (Addison-Wesley Publishing: 1987); *The Art of Human-Computer Interface Design,* edited by Brenda Laurel (Addison-Wesley Publishing: 1990); and *Tog on Interface* (Addison-Wesley Publishing: 1991). The first is Apple's user-interface bible for developers, and the second is a collection of papers about interface design written by Apple and non-Apple experts. Both are available through APDA.

The third is a collection of columns from *Apple Direct* on human-interface design. Written by Bruce Tognazzini, Apple's former UI evangelist, these columns discuss various points of interest in interface design, such as how pop-up menus should work or how to design alert boxes. They contain a wealth of information.

You should also keep pace with what's happening in the market. Keep a constant watch on product reviews, both positive and negative, and for all application types. You'll learn a lot by following what reviewers like and don't like, and you may find these reviews to be a source of fresh ideas.

However, make sure you read them with a critical eye, and don't take everything reviewers say at face value. Remember that *MacWeek* and *InfoWorld* reviewers, for example, may approach the same product with two different biases:

One publication is Mac-oriented, and the other is PC-oriented. Although neither is right nor wrong, you must understand each viewpoint so that you can put it into perspective and use the information, as appropriate, to tailor your product to the target market.

After you've gone through these steps, you'll begin to have a feel for what the "Macintosh way" is all about. This approach has such a strong following among users that some people go so far as to call it a "religion." But whatever you call it, the important point is that you remain constantly aware that Macintosh users have strongly rooted expectations of how an application should and shouldn't work.

Although understanding those expectations may take some work on your part, living up to them will go a long way toward helping your product gain acceptance in the wider Macintosh market. Use those expectations to your advantage.

LET THE DESIGNING BEGIN

Once you've made sufficient mental preparations, you're ready to tackle the real design work. This next point is very, very important: You may be tempted to use your PC product as the basis of the new Macintosh one. *Don't do it.* Start from scratch, and assume that you will use none of the old interface. In the end, you may use some of it, but if you do so, make sure it's a well thought-out choice.

A major design consideration is that DOS programs are strictly application-centered, whereas Macintosh applications are more document-centered. What exactly does this mean? When Mac users want to use their computers, they don't have to think, "Now I'm going to launch my word processor and my spreadsheet." Instead, they simply open their Project Report and their Budget Analysis documents, without regard for what applications are needed to run them.

In keeping with that approach, there is no dedicated application window on a Mac, and certainly not one that must precede the document window in opening order. The only screen "real estate" (apart from a document) a Macintosh application should permanently occupy is the menu bar across the top of the screen.

There are similar basic principles for menu design. Wherever it's possible to take commands out from under a menu tree and replace them with direct manipulation, do so. Also, don't give users the opportunity to make mistakes. If certain commands or tools are not relevant or shouldn't be used at a particular point, they should be grayed-out.

For example, don't just leave a tool out in the open and have it repeatedly beep when the user erroneously tries to use it. Many PC programs handle this kind of situation by displaying completely different menus for different areas of the application.

In Macintosh design, you should not change contexts on a user. (But if you feel that you must do so, first read Chapter 33, "Making an Interface Articulate," in *Tog on Interface.*) Rather than changing menus and tools, the preferred approach is to gray out individual items when they are invalid.

Remember that much of good Macintosh interface design is just plain good interface design—principles that should apply to the design of *any* interface. For example, dialog boxes should be kept small *not* because many Macintosh computers have small screens but because dialog boxes shouldn't obscure the user's work on the screen.

Similarly, box layouts should be kept consistent. Screens and dialog boxes should use plenty of white space to be more pleasing to the eye and easy to read. Also, avoid multiple cascading menus, because they are cumbersome to use.

But be aware of times when you *should* violate some good design principles to comply with de facto Macintosh standards. Take a look at the File Close dialog box in any Mac application. In response to the "Save changes before closing?" prompt, you'll probably find that the default "Yes" button is at the left or upper left in relation to all the other buttons.

Good design principles suggest that the default button should be on the right or the lower right for faster access, but the File Close dialog box layout is a de facto standard and shouldn't be tampered with.

All this talk about "design principles" might make you feel as if you are bound by too many rules and constraints. But don't think of them negatively, as "rules." These guidelines have been around for quite some time, and for good reason: They have stood up to lots of user testing. When something feels uncomfortable to you, examine it more closely. You'll probably discover that there really is a sound, logical reason behind it.

And again, remember that these are guidelines, not straitjackets. There is plenty of room for creativity within them. The floating palettes in Lotus 1-2-3 for Macintosh are an example of that kind of creativity. These palettes contain, among other things, draw and style tools that can be moved anywhere on the screen or be closed if the user doesn't want to see them. They do not permanently occupy any space, yet they are always readily available.

PROTOTYPES AND TESTING

Once you've developed a pretty solid design, you're ready to start creating prototypes. Create not one prototype, but two. Make the first one almost a pure port of the original application or at least a recognizable derivative. Retain the interface conventions of the original DOS platform, such as having the Copy command at the top level of the menu.

Then make a second prototype— your best first attempt at a Macintosh interface. To allow for quick changes and iterations, don't use production code to create your prototypes. Some tools that we've found especially well suited to the task include SuperCard from Silicon Beach; Prototyper from Now Software; and AppMaker from Bowers Development.

Now you're ready to test your prototypes on real users. When selecting testers, don't go to your tried-and-true customers. Instead, select some dyed-in-the-wool Macintosh users. (User groups might be a good place to find them.)

Sit these people in front of your two prototypes, and videotape the entire test. Why? Because that videotape will become an indispensable political tool later on when you need to "sell" your interface to your company's upper management.

Conduct lots of testing sessions. Let your testers tell you what is or isn't Mac-like. Do you have Copy, Move, Quit, and other common commands in the right places? Do things behave as expected? Is the application, overall, intuitive to learn and use?

The key points are to have two prototypes, select testers who are Mac users, and videotape the testing sessions.

PRODUCTION MODE: STAY ON TRACK

Once you've tested your design and are ready to go into full production mode, are you home free? Not exactly. There are still many places where you can steer off course.

Take a look at your development team. You're likely to have a group of people who have little or no Macintosh experience. And no matter how good your initial design may be, the small, seemingly subtle decisions that each team member makes can affect the look and feel of the product. What can you do to motivate your team properly?

First, make sure you have *some* Mac people on your team. Don't worry if they're in the minority. Any time you put Mac enthusiasts into a group of willing "converts" from the PC world, they will happily play the role of Macintosh evangelists. Those people will be an invaluable resource, not only for product and technical information but also for showing others the Macintosh way of doing things.

Also, have your developers set up their own Macs, install the system software, connect to the network, and so forth. (They'll be amazed at how easy it is.)

Next, expose them to all the "neat" things available on the Mac. Get them lots of INITs and cdevs (since so many are inexpensive shareware or public-domain software, it'll be easy on your budget), and let them discover what's fun about the Mac. Don't discourage games. Many of them have terrific graphics.

Next, introduce your team to examples of good Mac applications, such as the ones you tried at the start of your learning process. Make sure they understand why these are good applications. Once you've gotten this process going, it shouldn't take long before your team is really excited.

Another area in which things can go awry is in writing your documentation. It is very important that the writers get their terminology straight. For example, the Macintosh world has documents instead of files, folders instead of subdirectories, and pointers instead of cursors. And some terms, such as *insertion point,* don't even exist on the PC. Make sure that your writers are sensitive to these differences. And don't forget to include these people in your motivational efforts.

Throughout the development cycle, there will be strong forces trying to pull you away from doing The Right Thing. Knowing that you'll run into these difficulties, make sure you're prepared by doing your homework in advance. The videotapes you made during user testing will probably come in very handy. Also, you may want to keep some key product reviews on file.

WORKING WITH APPLE

As a serious Macintosh developer, one thing you must do is to sign up for the Apple developer program (or use whatever resources are available in your country). If you are reading this article, then you may already be a member and know the benefits, so I won't go into detail here. Just be aware that Apple can be helpful to you in many ways.

Get some back issues of *Apple Direct,* and read (or reread) Bruce Tognazzini's Human Interface column. Get his book, *Tog on Interface.* Also, get a subscription to *develop,* the Apple technical journal. And if you need to, sign up for Apple Developer University courses. The long and short of it is to learn everything you can and take advantage of all the tools Apple makes available.

YOU CAN DO IT

At times, you may encounter a particularly knotty problem and want to throw up your hands. It just can't be done, you'll say. But the truth is, preserving cross-platform compatibility is doable without sacrificing the Macintosh way. Yes, it can be done; it just takes more work.

The best evidence of that is the graphing feature of Lotus 1-2-3 for Macintosh. Graphing has been hailed as among the best and more "Mac-like" aspects of the product. And yet it is compatible with graphs created in the DOS version of 1-2-3. On the Macintosh side, graphs are objects on a draw layer, with individual components capable of direct manipulation. On the DOS side, information about graphs is stored as named graph "settings" in a worksheet file. You can store multiple graphs with a file by going to the menu and saving different settings under different names.

Clearly, each approach is done in the characteristic style of its platform. Note, too, that the interfaces are very different between the two versions, and yet compatibility has been maintained.

Don't be afraid of completely altering the interface on your new Mac version. Your existing users won't mind change, as long as you offer some clear benefits in exchange for the pain. Also, remember that there will be far more people moving to your Mac application from some other Mac application than moving from the DOS to the Mac version of your product. So what appears to you to be a radical change may never be an issue to most users.

ASK YOURSELF: WILL PEOPLE LOVE THIS?

As you forge ahead, don't forget to stop every once in a while and reflect. Mussie Shore, the UI designer for Lotus 1-2-3 for Macintosh, points out, "People work with their PCs, but people love their Macintosh computers." His rule is to constantly ask himself, "Will people love this design, or will they just work with it?" To create a true Macintosh application, you can never lose that thought.

Yuko Takagi is Lotus Development Corporation's associate product manager for Lotus 1-2-3 for Macintosh. Lotus Development Corporation is based in Cambridge, Massachusetts.

APPLE DIRECT

Apple Direct is Apple's monthly developer newspaper, covering business and technical issues for decision makers at develop-
ment companies. It is published by Apple Computer, Inc.'s Developer Support Systems and Communications (DSSC) group.

Publication team:

EDITOR-IN-CHIEF:
Dee Kiamy (AppleLink:KIAMY)

TECHNICAL WRITER/EDITOR:
Gregg Williams (GREGGW)

PUBLICATIONS AREA MANAGER:
Hartley G. Lesser (H.LESSER)

FORMATTER:
Lisa Ferdinandsen

CONTRIBUTORS:
Jessa Vartanian, Suzanne Dills, Sharon Flowers, Monica Meffert, Suzanne Forlenza, Kris Newby, Cindy Cain, Tammy Gniffke, Katherine Parsons

MANAGER, DSSC:
David A. Krathwohl

CONTENT GROUP MANAGER:
Greg Joswiak

FILM:
Aptos Post, Aptos, CA

PREPRESS:
Prepress Assembly
San Francisco, CA

PRINTER:
Wolfer Printing Co., Inc.
Los Angeles, CA