The Developer Business Report

August 1995

AppleDirections

Inside This Issue

eWorld Is Coming!

If you're a member of Apple's developer programs, you'll soon receive your free copy of eWorld 1.1. Apple's transition from AppleLink to eWorld will take place between August and the end of the year, depending on your location. After that, you'll have the option of communicating with Apple via eWorld or the Internet. eWorld, Apple's official on-line business communications system, provides the benefits of a commercial on-line system as well as access to private content for program members. You'll want to install your software and start experiencing eWorld's electronic community as soon as you can. We'll provide all the details about the transition in next month's *Apple Directions.*

Apple News

Apple Ships New PowerPC 603–Based Systems for Consumer Market

75-MHz Internet-Ready Systems Part of Latest Macintosh Performa Release

Apple Computer, Inc., continued its transition to an all-PowerPC product line this month by introducing two series of PowerPC processor–based Macintosh Performa computers designed specifically for the home and home office—the Macintosh Performa 5200CD computer and Macintosh Performa 6200CD computer. Both computers employ the 75-MHz PowerPC 603 processor and include Internet access as well as quadruple-speed CD-ROM drives.

Additionally, Apple introduced the Macintosh Performa 631CD computer, the Macintosh Performa 640CD DOS Compatible computer, the Macintosh Performa 6116CD computer, and the MPEG Media Kit for Macintosh Performa 630, 5200, and 6200 systems.

All the new systems were shipped to meet increasing demand in the home market;

please turn to page 11

Strategy Mosaic

Strategy: 1999

By Gregg Williams, Apple Directions *staff*

Part Two: What You Should Do—Now

In last month's article, I talked about the evolution of the Mac OS platform through various key standard technologies. This month, I want to look (briefly) at how Apple technologies look to customers, then focus on which Apple technologies are relevant to you, the developer, and which you need to implement immediately.

Because the technology strategy of Apple Computer, Inc., contains a lot of pieces, this article—which concludes my two-part explanation of Apple's strategy—has been difficult to write. But just as I said last month, Apple's strategy becomes simpler when you concentrate on the parts that are relevant to you and your business.

Or, to put the previous paragraph in simpler terms: *There's a lot of stuff in this article, and if you don't read it all, you'll miss the part you need to hear.* However, I can summarize the overall thrust of this article as follows:

- Apple organizes its technologies into three layers: foundation (technologies that support the two layers on top of it), platform (the technologies necessary for implementing the Mac OS user experience), and domain (technologies useful to certain kinds of products).
- The Mac OS platform promises certain things to customers (which I'll get into later)

please turn to page 4

AppleDirections

Volume 3, Number 8

Apple Directions, the monthly developer newsletter of Apple Computer, Inc., communicates Apple's strategic, business, and technical directions to decision makers at development companies to help maximize their development dollar. It is published by the Apple Developer Periodicals group within Apple's Developer Press.

Editor
Paul Dreyfus (AppleLink: DREYFUS.P)

Technical Editor
Gregg Williams (GREGGW)

Business & Marketing Editor
Kris Newby (NEWBY.K)

Associate Editor
Anne Szabla (SZABLA)

Production Editor
Lisa Ferdinandsen (LISAFERD)

Contributors
Dave Curbow, Peter Bickford, Alex Dosher, Elizabeth Dykstra-Erickson, Kris Newby, Kurt Piersol

Manager, Developer Press
Dennis Matthews

Manager, Apple Developer Periodicals
Mark Bloomquist

Production Manager
Diane Wilcox

Prepress/Film
Aptos Post

Printer
Wolfer Printing Co., Inc., Los Angeles, CA

Editor's Note

Think Globally, Act Globally

Last month, IBM became the latest company to announce it will ship computers that run the Mac OS. The company said it would license the Mac OS to run on its Hardware Reference Platform (HRP) systems. HRP is the specification for PowerPC processor–based computers being defined by IBM, Apple Computer, Inc., and Motorola; HRP systems will run PowerPC native versions of Mac OS, AIX, Novell Netware, OS/2, Windows NT, and Solaris.

The news was enough to get Wall Street excited: Apple's stock jumped more than 10 percent in the days that immediately followed the announcement. We also think this is pretty terrific news. Although there are many details to be worked out, and the first IBM-produced Mac OS computers aren't supposed to ship until the second half of 1996, this announcement means that Apple's Mac OS licensing strategy is gaining considerable momentum.

Combined with other developments, we think this means that you can expect the market for your Macintosh products to grow—in global proportions—in the coming years. The key word in that last sentence is *global,* because a great deal of the growth is likely to come in locations outside the United States, currently the largest Macintosh market.

According to recently released data prepared by Dataquest, Asia will be the fastest growing market for personal computers between now and the year 2000. The market research firm projects that the Asian personal computer market will grow at an annual rate of just over 20 percent over the next five years, faster than Western Europe's 17 percent annual growth rate and double the 10 percent U.S. rate.

A recent report in the *Wall Street Journal* called "Computer Sales Sizzle As Developing Nations Try to Shrink PC Gap" implied that there are already considerable opportunities to market computers in less-developed nations. In Latin America alone, the personal computer market grew approximately 25 percent in 1994, according to International Data Corp.—faster than the growth of the U.S. computer market last year.

One of the nice things about trying to sell computers in less-developed nations, according to the *Journal* report, is that there are no legacy systems. Many, and in some areas most, business customers are first-time computer buyers, who don't have to think about preserving their investments in existing hardware and software. It seems that such customers would be more receptive than others to the virtues of the Macintosh computer, and Apple is going to considerable lengths to be sure that happens.

Take, for example, Apple's efforts in China. After launching a series of business initiatives in China over the past several years, including opening the Apple Publishing Center in Beijing, Apple hosted the China Market Forum in Beijing at the end of June. At the event, Apple representatives met with Chinese developers and others to give them an understanding of Apple technologies and market strategies so they can help build Apple's business among the vast Chinese population.

There are two straightforward messages here:

• Go native! Make sure your product joins the 750 current Macintosh applications that run in native PowerPC mode so you can market them to the expanding base of PowerPC processor–based Macintosh systems, which will soon be augmented by Mac OS–based computers from other manufacturers.

• Go global! Be sure your applications are WorldScript-savvy so they can be easily localized for new, expanding markets around the world. To make it even easier for you to develop international software, Apple is building support for Unicode into the Mac OS and providing other internationalization tools. We'll tell you all about them as soon as we're able.

By heeding these messages, you and your products will be poised to take advantage of tomorrow's Macintosh customer base, as it grows through the proliferation of Mac OS systems from licensees and the expansion of new geographic markets.

Paul Dreyfus
Editor

IndustryWatch: News & Perspective

Cool Developments

Prepared by the Apple Directions *staff*

Windows 95 Mixed Reviews—Opportunity for Apple

Reports about problems with Windows 95 continue to appear. Here are some of the bugs and incompatibilities that industry publications have noted:

• Windows 95 has compatibility problems with some installed hardware, such as video cards and modems; with hardware drivers; and with most virus checkers and disk utilities.

• Windows 95 preemptive multitasking works only with 32-bit applications; if only one 16-bit application is running under Windows 95, none of the applications—even those that are 32-bit clean—have access to preemptive multitasking. Since there are reported slow-downs in delivery of 32-bit Windows applications, it should be quite some time before the promise of Windows 95 preemptive multitasking can be delivered upon.

• Most people will need 8 MB of RAM or more to run Windows 95 at acceptable speed. However, most of the PC installed base has less than 8 MB, which will slow Windows 95 adoption.

Implications/Opinions: Although we still think Windows 95 will be a huge commercial success—and there will definitely be many good things to say about it—we think all the problems surrounding its initial release will tarnish its reputation, especially among customers who are upgrading from a previous version of Windows. This will give the Apple community an opportunity to reinforce the message that because of Macintosh hardware and software integration, Macintosh systems work more smoothly than their PC counterparts.

For the 1995 year-end holiday shopping crunch, the Macintosh platform could be in an even better situation competitively than it was at the end of last year. If you'll recall, Apple enjoyed record monthly sales during November and December of 1994, even though computer customers were anticipating the revolutionary new product from Microsoft (that is, Windows 95). During this year's holiday buying season, that new product will have shipped to very mixed reviews. That will be an ideal time for Apple and you to tell your customers that with Macintosh systems, users can spend more time working, learning, or playing instead of just figuring out how to get their computer to run.

September *Apple Directions* Online

The September issue of *Apple Directions* will be available on AppleLink, the Internet, and eWorld by August 15, at the following locations:

AppleLink: path—Developer Support:Developer Services:Periodicals:Apple Directions.

Internet: select Developer Services and Products at the location www.apple.com.

eWorld: in the Apple area of the Computer Center.

Microsoft Hedging Its Bets With Windows NT?

A report in the June 5, 1995, issue of *InfoWorld* magazine says that Microsoft sales representatives are gently urging some corporate IS managers to migrate to Windows NT instead of Windows 95. According to several large corporate IS managers interviewed for the story, Microsoft sales reps have offered a version of the Windows NT Workstation with the Windows 95 GUI at a similar price to what they had expected to pay to upgrade to Windows 95.

Implications/Opinions: Maybe Microsoft is hedging its bets in the face of the problems surrounding Windows 95. Although a Microsoft official interviewed for the story said that the sales reps' offers don't reflect any official Microsoft policy shift to offering Windows NT as an alternative to Windows 95, the market could force a de facto policy change. Said one IS manager quoted in the *InfoWorld* story, "For our specific needs, Windows 95 is a transitional operating system." Another said, "We've been testing both Windows 95 and Windows NT for more than two years now, and there's no question. For client/server computing, Windows NT is the better OS." Still another: "Windows 95's memory protection and multitasking model are poor in comparison with Windows NT's."

Windows NT also has problems of its own in gaining widespread corporate adoption. It requires at least 16 MB of memory and doesn't guarantee backward compatibility with existing Windows 3.X 16-bit applications. The situation could put many corporate IS managers, and the users they serve, between a virtual rock and a hard place. Again, this could provide Apple an opportunity—to take advantage of the uncertainty and make further inroads into the corporate IS market.

Just What Is OLE 3.0?

Microsoft outlined their technology plans for extending the OLE architecture for better support of enterprise applications. The next major release, OLE 3.0, will provide support for multipage and irregularly shaped OLE objects as well as for communications between objects distributed in a network environment (under Network OLE). It will also provide better support for transaction processing applications (under OLE Transactions).

Implications/Opinions: We're just going to be direct, here: We think this is a classic industry stratagem, which many refer to as "FUD" (spreading Fear, Uncertainty, and Doubt). Microsoft hopes to paralyze the market for component software in the face of Apple's superior product, OpenDoc, which will beat OLE 3.0 to market by as much as one year.

Microsoft's announcement is little more than a statement of direction. The specifications for OLE 3.0 won't even be finalized until late 1995, and products based on this technology won't be available until late 1996 or 1997. By late this year, Apple will already be shipping OpenDoc, and hundreds of OpenDoc components will be on the market long before OLE 3.0 is released.

Microsoft had to make a strategic statement of direction because its enterprise customers were demanding it (especially a statement of distributed object support for OLE). Microsoft is supporting these announcements with technology demonstrations, designed to make the technology seem closer than it really is, in an attempt to undermine the technical advantages of OpenDoc among key industry influencers.

Hard Disk Prices Dropping Sharply

The prices of hard disk drives have continued to plummet. The marginal cost per megabyte (the cost per additional megabyte) of drives in the 500 to 800 MB capacity range is less than $0.15, and it's expected to drop further.

Implications/Opinions: The difference in street price between a 540 MB drive and a 700 MB drive is already less than $25. You can expect the size of even entry-level computer customers' hard disks to increase dramatically in the coming months. For example, the smallest of Apple's new line of Performa systems (see news on page 1) will ship with a 500 MB hard drive, while the largest will ship with a 1 GB (gigabyte) hard drive. We would never advise you to come out with ever-larger versions of your products to fill your customers' hard disks. That said, if storage capacity has been a concern for you, the concern should soon lessen, substantially.

They're Just Getting Started, but "Watch Out, Bill"

We want to tell you about a unique (at least as far as we know) startup company that an *Apple Directions* reader was kind enough to alert us to. It's called Tenadar Software, and it specializes in games and learning software, two hot areas of the market. Its first product is a shareware HyperCard-based adventure game that's available from America Online and Tenadar's Web page.

What's unique about Tenadar is that it's run, with minimal adult intervention, by a group of ten-year-olds from the San Carlos Charter Learning Center in California. (We also like that Tenadar is first releasing Macintosh versions of its products, with Windows versions to follow. Those are some smart kids!)

Tenadar already has its own World Wide Web page, which the company's young principals designed and constructed themselves. After a basic introduction to the World Wide Web, they learned additional HTML programming skills by "Saving as Source" others' Web pages to examine how the features they needed, such as blinking text, hyperlinks, and sending e-mail to the author, had been constructed.

Most of their Web pages are still "under construction," and their best software is still under development. But we think you'll want to check out their home page on the Web at http://www.ipp.com/ipp/tenadar/tenadarhomepage.html. And if you download their first product, please pay the modest shareware fee; half the money goes to help their school's technology program.

Implications/Opinions: As a friend of ours says, sometimes you can spend too much effort trying to look for meaning in things. We're telling you about this just because we think it's cool. If you hear of other cool new developments, feel free to submit them to us for possible publication in this column; you can submit them to A.DIRECTIONS @applelink.apple.com.

. . . And Finally, From Our "Some People Have Too Much Time" Department

Someone surfing the 'Net sent us the following top ten anagrams for "Information Superhighway":

10. Enormous, hairy pig with fan
9. Hey, ignoramus—win profit? Ha!
8. Oh-oh, wiring snafu: empty air
7. When forming, utopia's hairy
6. A rough whimper of insanity
5. Oh, wormy infuriating phase
4. Inspire humanity, who go far
3. Waiting for any promise, huh?
2. Hi-ho! Yow! I'm surfing Arpanet!

And the number one anagram for "Information Superhighway" is:
1. New utopia? Horrifying sham

Implications/Opinions: 'Nuff said.

(Our apologies to whoever created this list; your name was not included in the posting.) ♣

Strategy Mosaic

1999

continued from page 1

and provides technologies for delivering them, but Apple depends on you to use those technologies in your products to deliver that promise.

• One of the most important things that the Mac OS platform promises is a rich, consistent human interface. Apple depends on you to implement certain basic technologies (which I'll also get into later). If you don't implement these technologies, the user experience is not consistent—and everybody loses.

• The changes that you must make to your applications do not require major programming efforts, and putting them off will make your product less attractive to Mac OS customers. You should implement these changes as soon as possible and roll them into the next incremental release of your application.

• The changes just mentioned belong to the Mac OS platform technologies layer. In addition, Apple has made lists of technologies that you may find useful if you have a product that fits into one of several domains ("learning," for example). Looking at these lists (discussed toward the end of this article) may give you ideas for improving your products.

What Apple Promises to Customers

Any discussion of what you, as a developer, should do must start here, with Apple's promises to customers. Why? Because Apple relies on you—through the products you create—to use Apple technologies to deliver to customers the promise of the Mac OS platform.

Since the introduction of the term *Mac OS* in early 1994, Apple has given a consistent message to customers—Mac OS computers "stand out" (that is, offer a superior computing experience) and "fit in" (are able to work with existing non–Mac OS standards that the customer might already be following).

The figure "The Mac OS advantage for customers" illustrates this message in more detail. The center column of jigsaw pieces illustrates the "stand out" part of the strategy, while the two large pieces on either side show how the Mac OS "fits in" with the rest of the world.

The most relevant pieces to this discussion are those in the center, "stand out" column. There are two pieces to the foundation of Apple's "stand out" strategy:

- RISC hardware, in the form of the PowerPC processor
- component software, in the form of OpenDoc components

These two pieces form the foundation for three things that (according to Apple) the Mac OS delivers better than competing platforms:

- the easiest user interface
- the best environment for media-rich documents and applications
- a superior level of integration of communications and collaboration into the user experience

Keep these three points in mind, because in the customer's view, the Mac OS platform delivers them largely to the degree to which you, collectively, supply them in your products.

A Technology Framework for Developers

As you saw in last month's Strategy Mosaic, Apple has a seemingly overwhelming number of technologies that it wants you to support. But is it equally vital for you to adopt each of these technologies?

The answer is *no.* And to help you understand the relative importance of these technologies, Apple has separated them into three layers of technologies, each of which depends on the ones below it. The layers are, from bottom to top, foundation technologies, platform technologies, and domain technologies.

The foundation technologies are the (dare I say it?) bedrock on which the Mac OS platform depends: the PowerPC processor and OpenDoc component software. *Apple Directions* has printed numerous articles on the importance of both these technologies, so I won't dwell on them here. But I will make two brief points:

- OpenDoc will be available for the Mac OS platform this fall, with a Windows version (from Novell) following in 1996.
- Already, Apple has said that all of the new computer models it announces will use the PowerPC processor. Furthermore, Apple has said (at last May's Worldwide Developers Conference) that, by the end of this year, 95 percent of all the computers it manufactures will use the PowerPC processor. The message? If you haven't already, you need to take your software "native" (that is, convert it to run on the PowerPC processor).

The middle layer is the Mac OS platform technologies layer. Here, you find those core technologies that define what makes the Mac OS unique. I'll talk about this layer next.

The top layer is the domain technologies layer. This includes technologies that are useful for certain types of products. I'll talk about this layer and about what technologies are most relevant to each domain toward the end of this article.

Platform Technologies (Required Reading)

If you're skimming this article, stop and read the following sections (at least up to "Domain Technologies") carefully. This is the heart of the matter, where the rubber meets the road, the inside scoop, the Real Deal, Ground Zero.

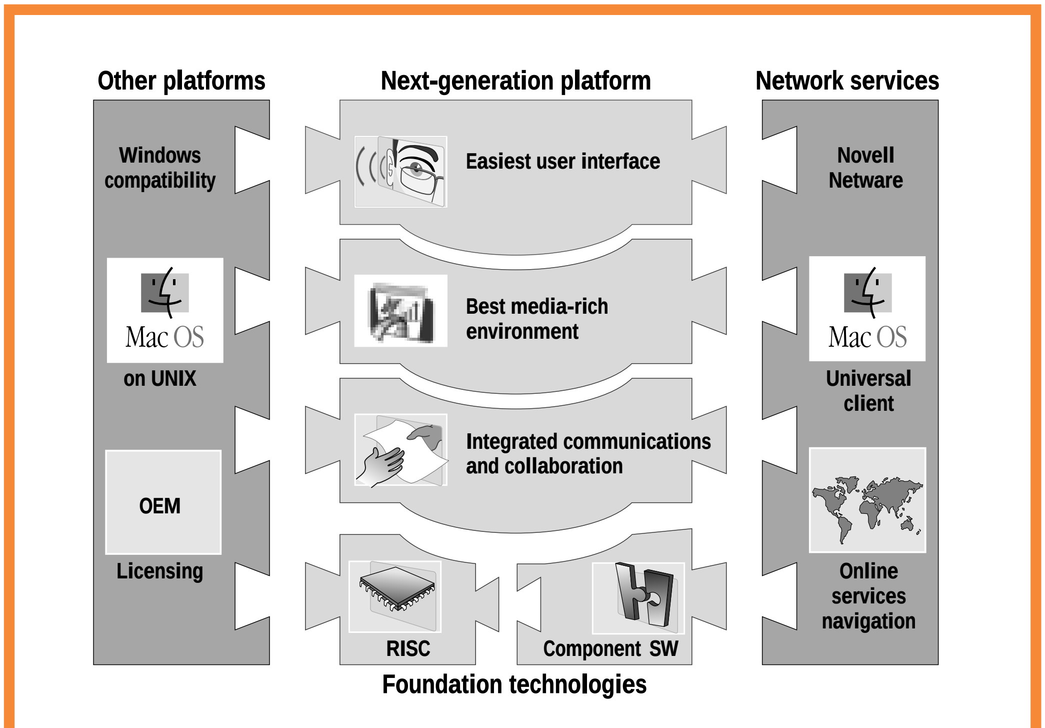
Oh, and did I mention that this is the important part?

As I said at the beginning of this article, the quality of the human interface of the Mac OS is *very* important. Customers must perceive the Mac OS human interface as both rich and consistent:

- *Rich* means that the interface offers features that make customers more productive and improves their experience of using a computer.
- *Consistent* means that a feature is present wherever it makes sense, and customers don't have to remember where the feature is and isn't present.

If customers do not believe both these points, their perception of both your product and the Mac OS platform goes down, and everybody loses.

Let's take the case of drag-and-drop objects. Everybody loves the feature, but nobody uses it. I love the feature, but *I* don't use it. Why? Because not enough programs use it. Because I have to



The Mac OS advantage for customers. The jigsaw pieces in the center column show how the Mac OS "stands out," while the pieces on each end show how it "fits in" with the world at large.

The Apple Technology Framework

Learning

DOMAIN

- QuickTime 2.x
- QuickTime VR
- QuickTime Conferencing
- QuickDraw 3D
- Internet solutions
- Sound Manager
- Speech Manager
- HyperCard
- Apple Media Tool

PLATFORM

- AppleScript
- Apple Guide
- PowerTalk mailer
- Thread Manager
- Mac OS human interface (Toolbox, guidelines)
- Drag and Drop
- Open Transport
- QuickDraw GX printing
- Standard data types

FOUNDATION

- PowerPC 603 (K–8)
  Power PC 601/604 (higher education)
- OpenDoc (higher education)

Graphics and publishing

DOMAIN

- QuickTime 2.x (for electronic publishing)
- QuickDraw GX (text and graphics)
- QuickTime 2.x (for electronic publishing)
- Multiprocessing API
- ColorSync 2.0

PLATFORM

- AppleScript
- Apple Guide
- PowerTalk mailer
- Thread Manager
- Mac OS human interface (Toolbox, guidelines)
- Drag and Drop
- Open Transport
- QuickDraw GX printing
- Standard data types

FOUNDATION

- PowerPC 604
- OpenDoc

Games and entertainment

DOMAIN

- QuickTime 2.x
- QuickTime VR
- QuickDraw 3D
- Sound Manager
- Speech Manager
- Video hardware acceleration

PLATFORM

- AppleScript
- Apple Guide
- PowerTalk mailer
- Thread Manager
- Mac OS human interface (Toolbox, guidelines)
- Drag and Drop
- Open Transport
- QuickDraw GX printing
- Standard data types

FOUNDATION

- PowerPC 603

Productivity

DOMAIN

PLATFORM

- AppleScript
- Apple Guide
- PowerTalk mailer
- Thread Manager
- Mac OS human interface (Toolbox, guidelines)
- Drag and Drop
- Open Transport
- QuickDraw GX printing
- Standard data types

FOUNDATION

- PowerPC 601
- OpenDoc

Text in orange denotes a technology that is not critical to this domain; see main text for details.

Communications and collaboration

DOMAIN	• PowerTalk • PowerShare • Internet solutions	• QuickTime Conferencing • Telephone Manager
PLATFORM	• AppleScript • Apple Guide • PowerTalk mailer • Thread Manager • Mac OS human interface (Toolbox, guidelines)	• Drag and Drop • Open Transport • QuickDraw GX printing • Standard data types
FOUNDATION	• PowerPC 603/620 • OpenDoc	

remember, "Well, I'm in Word-Perfect now, and it supports drag-and-drop, but Microsoft Word *doesn't* support it, so I can't drag this WordPerfect text into that Word document." Rather than ask myself that question every time I want to move data, I just sigh and use the old tried-and-true Clipboard instead. (And why is that a no-brainer decision? Because *every program implements the Clipboard.*)

In the past, Apple has said, "Adopt technology X, because customers will want it when they see it, and you will need to use it to keep your customers happy and your competitors at bay."

Here's what Apple's saying now: "If your software does W, you need to adopt technology X and roll it into your next release. If potential customers don't get the benefits of X whenever they use their Mac OS computer, they won't buy Mac OS computers, and you won't have a market for your software." It's a matter of "raising the bar," and Apple can't do it on its own—it depends on you doing your part as well.

Apple's bottom-line message to you is, "Here is a list of technologies that, for the sake of the Mac OS platform, you need to implement in your software as soon as possible. We think this is a reasonable list—it's as short as we can make it, and the technologies do not require major efforts to implement."

So, with that said, here are Apple's recommendations for platform technologies. (For more information about many of these technologies, see the "Resources" box at the end of this article.)

To-Do List for Application Developers

• *Every piece of software should support the Mac OS*

Technical and scientific

DOMAIN	• QuickTime 2.x • QuickDraw GX • QuickDraw 3D	• QuickTime VR • Multiprocessing API (text and graphics) • Internet solutions
PLATFORM	• AppleScript • Apple Guide • PowerTalk mailer • Thread Manager • Mac OS human interface (Toolbox, guidelines)	• Drag and Drop • Open Transport • QuickDraw GX printing • Standard data types
FOUNDATION	• PowerPC 604 • OpenDoc	

Custom solutions

DOMAIN	• PowerTalk • PowerShare • HyperCard	• Denali • Apple Media Tool • GuideMaker
PLATFORM	• AppleScript • Apple Guide • PowerTalk mailer • Thread Manager • Mac OS human interface (Toolbox, guidelines)	• Drag and Drop • Open Transport • QuickDraw GX printing • Standard data types
FOUNDATION	• PowerPC • OpenDoc	

run-time environment. It should be PowerPC native and 32-bit clean, and it should support cooperative multitasking. (These days, the last two items are universally observed.)

• *If your software supports the Clipboard, it should support Macintosh Drag and Drop.* Drag-and-drop behavior must be both consistent and available everywhere to be useful, and the Drag Manager was designed to make it as simple as possible for every program to implement a consistent, compatible form of drag-and-drop behavior.

• *If your software supports online help, it should guide users through commonly performed operations using Apple Guide.* The great thing about Apple Guide help is that you can add it to existing products, even products you've already delivered to your customers. Apple Guide files are easy to create and can be done by someone with no programming skill. If you can't support Apple Guide, then make sure to put access to your custom help system in the Help menu.

• *If your software supports printing, it should support QuickDraw GX printing* (which is a small, easily implementable subset of the full QuickDraw GX architecture). It gives your customers a better human interface for and more control over printing. In addition, your customers will get GX printing extensions and portable digital documents "for free"—all they have to do is use QuickDraw GX.

Also, you should keep in mind that QuickDraw GX will be an integral part of the Copland operating system and will no longer be an optional installation. By implementing QuickDraw GX printing now, you give much-appreciated features to current QuickDraw GX users and begin the process of becoming Copland-compatible.

The Apple evangelist for QuickDraw GX reports that developers have implemented QuickDraw GX printing in as little as one day "but no more than a week, tops."

• *If your software supports documents, it should support PowerTalk mailers.* Mailers, in case you've forgotten, make it very easy for customers to mail documents without leaving the application that created them. Customers using PowerTalk also get many of its benefits—including PowerTalk's catalogs, universal in-box, and key chain—without any additional work on your part.

As with QuickDraw GX printing, PowerTalk mailers are easy to implement, and you don't have to learn all of the PowerTalk API (application programming interface) to add PowerTalk mailers to your documents. Also, PowerTalk becomes standard with Copland, so implementing PowerTalk mailers now will decrease the amount of work you will have to do later to support Copland.

The PowerTalk evangelist reports that developers have implemented PowerTalk mailers in "around a week." See the "Resources" box for the reference to a *develop* article on how to implement PowerTalk mailers.

• *If your software supports documents, it should also be able to select, cut, copy, and paste all the basic data types* (pictures, text, sound, movies, and 3D objects) in your documents. Apple believes that the only way to fulfill the promise of a rich, consistent human interface is to make sure that if a program creates documents that hold user-created content, such documents should be able to contain the above five basic data types.

Of course, if your software can manipulate these data types (through the Clipboard), it should also support Drag and Drop. Software that can do both will be making an important contribution toward fulfilling the promise of the Mac OS platform.

When customers can put pictures, text, sound, movies, and 3D objects in their documents and drag and drop any data they can select to any document, they will perceive the Mac OS platform as offering a superior computing experience—one that offers data-rich documents and no-exceptions ease of use.

• *If your software is network-centered (for example, a network administration program or a network-based calendaring or e-mail program), you should use Open Transport to implement its network connectivity.* Look for an article on Open Transport in next month's issue of *Apple Directions.* For now, let it suffice to say that Open Transport makes it easier for you to write network-centered software and gives you a larger audience to sell to.

• *Finally, you should prepare your software for the international markets by providing, at the very least, minimal Text Services Manager (TSM) support.* For you, being TSM-aware means adding a few lines of code to allow your software to recognize TSMTE (Text Services Manager TextEdit) events. For customers, this means that if they are using a 2-byte script (Japanese, for example), they will be able to enter text in a 2-byte script everywhere TextEdit is used—filenames and dialog text-input fields, for example. (See the "Resources" box at the end of this article for pointers to relevant documents.)

Making your software TSM-aware is a complete solution only for software that has no text input or that uses TextEdit for all its text input. All other programs will require more work (see "'Recommended' List for Application Developers," later in this article), but becoming TSM-aware is a necessary precondition to that work.

To-Do List for Hardware Developers

If you design and sell hardware products, here is the list of hardware technologies Apple wants you to support:

• *PCI (Peripheral Component Interconnect).* This is, of course, the new bus that Apple is using for peripheral-card expansion in the new Power Macintosh 9500 and all future Mac OS computers. (It replaces the NuBus™ interface.) If you want to sell add-in cards to the Mac OS market, PCI is the way to do it. (See also "PCI Cards for the Second Generation of Power Macintosh," on page 13 of the August 1994 issue of *Apple Directions.*)

• *FireWire.* Officially known as IEEE P1394, this is a new public standard, championed by Apple, for a high-speed (100 or more megabits per second) serial bus. FireWire is an inexpensive bus for computer peripherals and consumer electronics and is suitable for the real-time delivery of data (including on-demand audio and video). You will see it used in everything from test instruments to video cameras to high-capacity mass storage devices, and you may want to investigate using it yourself.

• *GeoPort.* As reported in the March 1995 issue of *Apple Directions* (page 6), the GeoPort wired-connection and communications architecture has been adopted as a standard by Versit. (Versit is a global initiative formed last November by Apple, AT&T, IBM, and Siemens Rolm Communications, to promote interoperability of computer and communications equipment.)

Versit recently announced the availability of the specification of the Versit GeoPort standard.

(Further information is available on the Internet at http://www.versit.com.) Versit hopes that the dropping of all licensing fees and royalties connected with the commercial use of GeoPort will encourage manufacturers of computers, telephones, and other communications and information hardware to adopt GeoPort.

• *HRP (Hardware Reference Platform).* If you happen to be interested in making computers capable of running the Mac OS, you will probably be interested in the HRP.

The HRP specification, which is being drafted by Apple, IBM, and Motorola, defines a PowerPC processor–based computer that can be built from industry-standard components. HRP-compliant computers will be able to run versions of the Mac OS, Windows NT, AIX, OS/2, and perhaps other operating systems.

To-Do List for Driver Developers

The to-do list for people who write drivers is, unfortunately, not well defined in many cases. As you may know, the next major revision of the Mac OS, code-named *Copland,* will change the way you must write drivers—that is, code that controls a physical device or a system service. (For more information, read Chapter 5, "About the Copland I/O Architecture," of the *Copland Technical Overview* document. You can get this document in several ways; see the "Resources" box at the end of this article.)

If you're developing a driver for a PCI networking or display video card, you're in luck. By following certain Apple guidelines (see the "Resources" box for details), you can write your PCI card driver in a way that will be compatible with the driver model that Copland will use so that the card will continue to run correctly under Copland.

If you're developing other kinds of PCI cards (for example, PCI-to-SCSI interfaces, real-time control cards, and networking cards), Apple has not finalized the System 7–to–Copland transition story; Apple will let you know when it has the details figured out. For now, study the *Copland Technical Overview* and try to factor out core code in your driver from activation code; this will minimize the work you will have to do to interface to Copland.

To-Do List for Developers of Extensions

You have probably heard that, to permit future growth in the Mac OS, the Copland architecture is incompatible with today's extensions. For now, you should read whatever Copland materials Apple supplies. (Chapters 3 and 4 of the *Copland Technical Overview* are particularly relevant to extensions.)

As with drivers, you should think about factoring the core code of your extension from code that ties your core code to today's extension model. Any steps you take in this direction will help you when you make the transition to Copland later.

"Recommended" List for Application Developers

It's true that all the items I've put in the above to-do lists don't take that much time to implement. But I have a few things left over that, in all honesty, I can't put in the to-do list for application developers, because they require substantial efforts to implement.

So here they are, as a list of two application modifications that Apple "highly recommends" you make.

• The first one is simple: Support WorldScript in your application. Once you do that, it will be easy to *internationalize* your product—that is, to convert all text to another script (from English to Japanese, for example). To make your application fully suited to sell in other countries, you need to *localize* it, which includes internationalization but may require other changes that will make your software acceptable to another culture. (This includes changes necessitated by different business practices or social customs.)

Companies (including Apple) are finding an increasing fraction of their revenues coming from outside their home markets; Japan, in particular, is a market that no developer should ignore. If you build WorldScript into your application now, you'll be able to enter international markets more quickly whenever you decide to do so.

• The second recommendation is more daunting but still very important: *Make your program scriptable and recordable.* This involves factoring your application so that other programs can use it to get work done by sending it Apple events.

For applications that are likely to be used interactively with other programs—particularly applications that can be used to modify text, graphics, or other user data—Apple events and AppleScript are central to automating complex user tasks. In addition to enabling workflow solutions, these technologies can be used in combination with Apple speech recognition, Apple Guide, and OpenDoc to create a more powerful user experience and greater overall functionality.

Apple will introduce various forms of "active assistance" into Copland and later releases to the Mac OS, and applications that are "deaf, dumb, and blind" to Apple events will be less useful in those future environments. Also, Apple events are a part of OpenDoc, and supporting them now will help you make the transition to OpenDoc.

Domain Technologies

So far, I've talked about foundation technologies (on which the Mac OS platform is built) and platform technologies (technologies that allow the Mac OS to provide a superior computing experience). The foundation technologies are a given; the platform technologies should be present wherever they make sense. Now it's time to look at domain technologies.

Unlike the technologies in the first two layers, the ones in this layer are optional, but Apple believes that there is an affinity between certain technologies and certain domains.

Domains describe categories of products in which the Mac OS platform naturally "stands out," and the technologies matched to each domain are ones that you will probably find useful for creating products that "stand out" themselves. Apple has identified seven product domains:

• learning
• games and entertainment
• productivity
• graphics and publishing
• technical and scientific
• communication and collaboration
• custom solutions

The figures on pages 6 and 7 show expanded versions of the Apple technology framework, one for each of the seven product domains. These expanded versions show the Apple technologies that are the most relevant to each domain.

I want to draw attention to one characteristic of these seven diagrams. Though the technologies in the domain (top) layer of each diagram differ from domain to domain, the technologies in the platform (middle) layer are always a subset of the same nine technologies. (The platform layer of the "Productivity" figure shows all nine technologies.) To make it easier to see which of these technologies are present and absent in the platform layer, technologies that are not critical to that domain are listed in gray and should be considered to be "absent" from the platform layer for that domain.

Here are a few notes about the domain technologies figures:

• Note that each domain lists the standard data types and the Mac OS human interface (both the guidelines that define the human interface and the Toolbox routines that you use to implement them) as part of the platform layer.

• The games and entertainment domain is interesting because it omits the largest number of so-called essential technologies. Apple is being honest here—it has omitted the technologies that aren't critical to this domain. The Thread Manager is critical, because you can use it to increase the responsiveness of a game; Open Transport is critical to network games, but not to stand-alone games and entertainment titles.

• It's interesting to note that, with the exception of the games and entertainment domain, Drag and Drop, AppleScript, and Apple Guide are listed in the platform layer of each domain.

• Granted, it looks odd that the "Productivity" figure has nothing in its domain layer. But productivity applications are, by definition, "horizontal"—that is, they are useful by many people and for many purposes. This explains why all nine platform technologies are in the platform layer. In a way, "productivity" is a domain that is not domain-specific, and this helps justify the domain layer being empty. This is not to say that none of the technologies listed in the domain layer of other figures are not useful for productivity applications—just that they are not critical.

• "Custom solutions" is the domain of consultants, in-house developers, value-added resellers (VARs), information-systems vendors (ISVs), and "power users." Here, the emphasis is on the quick creation of specialized software to provide solutions for small groups of people.

Often, custom solutions are created by using preexisting "chunks" of technology that would otherwise be prohibitively expensive to build from scratch—PowerTalk, PowerShare, and Apple Guide/GuideMaker fit this category. You can use AppleScript to create custom solutions by tying together applications that are Apple event–aware. Today, you can create stand-alone solutions using HyperCard and the Apple Media Tool. In the future, you will be able to use Denali to create custom solutions for the Mac OS, Windows, and other platforms, in a way that is very similar to Microsoft's Visual Basic.

Resources

Apple Guide

See the article "How Do I Use Apple Guide?" on page 14 of the September 1994 issue of *Apple Directions* for details on this technology.

Copland

You can get the *Copland Technical Overview* document from Apple's ftp server, ftp.info.apple.com, using the path Apple Support Area:Developer Services:Technical Documentation:Tools, Demos, & Tech Overviews:Copland WWDC Materials.sit.hqx.

If you are using a Web browser, you can also get to Apple's ftp server by going to location http://www.info.apple.com/dev, then clicking on the highlighted phrase "ftp site."

FireWire

For more information on adopting FireWire, send e-mail to AppleLink address FIREWIRE or Internet address firewire@applelink.apple.com. On the Internet, you can get the current FireWire draft at ftp://ftp.apple.com/pub/standards/p1394.

OpenDoc

Apple Directions has run news or articles about OpenDoc in almost every issue since last November. In particular, see "OpenDoc *Is* Cross-Platform" (November 1994), "OpenDoc Your Mind" and "OpenDoc and Your Business" (December 1994), and "OpenDoc—One Architecture Fits All" (June 1995).

PCI Drivers

The documents "New Device Driver Model" and *Designing PCI Cards and Drivers* are available on the Developer CD; use the path Dev.CD Jun 95:Technical Documentation:PCI Information.

PowerTalk Mailers

A *develop* article, "Building PowerTalk-Savvy Applications" (*develop* Issue 16, page 39), tells you all you need to know to add a PowerTalk mailer to your application. Also, check out the associated sample code program, CollaboDraw, which you can find on any *develop* Bookmark CD. You can also find it on the May 1995 Developer CD, pathname Dev.CD May 95:Sample Code:AOCE Sample Code:Standard Mail.

Text Services Manager

For information on TSMTE, see New Technical Note TE 27, "Inline Input for TextEdit with TSMTE." (On the June 1995 Developer CD, you can find this file using the path Dev.CD Jun 95:Technical Documentation:Mac Tech Notes (DocViewer):TE • Text.)

For information on the Text Services Manager and related issues, see *Inside Macintosh: Text.* You can find this book on the June 1995 Developer CD, using the path Dev.CD Jun 95:Technical Documentation: Inside Macintosh.

The Future Is Up to You

So there you have it: between last month's article and this one here, Apple's technology strategy for now and the future—almost to the next decade (to say nothing of the next millennium!). Last month's article showed the overall evolution of Apple's technologies over time, and this article tells you which Apple technologies you should be implementing now.

But the most important thing for you to remember from this article is that Apple is depending on you to implement key technologies that will make the Mac OS platform as good as Apple designed it to be. And I can't stress enough that every developer must do his or her part to ensure our collective success.

Every application that uses the Clipboard must implement Drag and Drop or customers cannot use it, nor will they recognize drag-and-drop data manipulation as one of the features that makes them want to use Mac OS computers—and your products.

Drag and Drop, by itself, is not going to make or break the Mac OS platform. But when you combine

Drag and Drop with AppleScript and scriptable applications, the PowerTalk mailer, QuickDraw GX printing, Apple Guide, Open Transport, and full use of the five major data types (pictures, text, sound, movies, and 3D objects) and the Mac OS human interface, you have an extremely rich environment for computing—one that distinguishes both you and Apple from the competition.

These technologies do not require major programming efforts on your part; in fact, some of them can be done in under a week. You should add support for these technologies to your software and include it on your next minor release to customers.

Apple feels that the changes it's asking for are both necessary and reasonable and requests your help in making the superior design of the Mac OS a reality in customers' hands and minds. In doing so, you'll be creating superior products that will make you successful as well. ♣

The editors wish to thank Deb Dennis and Ruth Anderson for preparing the illustrations, "The Apple Technology Mosaic," that accompanied last month's Strategy Mosaic.

Apple News

Apple Ships New Computers

continued from page 1

personal computer sales to home customers have now surpassed business computer sales. Also, recognizing that home customers buy computers primarily to gain access to home learning products and that multimedia is a must for these customers, Apple has made sure that all the new systems are equipped to run multimedia software. Given Apple's efforts to meeting these customer needs, those of you with home learning and multimedia products can expect to be able to reach an ever-broadening base of Macintosh users in the home.

Macintosh Performa 5200CD and 6200CD Product Details

The Macintosh Performa 5200CD and 6200CD systems ship with eWorld, Apple's online service, which will provide customers easy access to the Internet and its increasingly popular World Wide Web. In addition to being Internet-ready, both the Macintosh Performa 5200CD and 6200CD series feature an internal 14.4-Kbaud modem that provides a full-duplex speakerphone, answering machine, and fax send/receive capability.

The speakerphone provides automatic dialing from the built-in address book or from the keyboard. It also includes advanced features such as the direct translation of alphabetical strings into phone numbers. The digital answering machine can store virtually unlimited messages and play them in any order. The system also allows monitoring of calls while recording.

The principal difference between the Macintosh Performa 5200CD and 6200CD series of computers is that the Macintosh Performa 5200CD series ships in a new, expandable all-in-one form factor, including a tilt-and-swivel base and an integrated 15" monitor. Macintosh Performa 6200CD computers employ a modular design, which keeps the monitor separate from the computer itself. Both systems ship with System 7.5, including Apple Guide, AppleScript, and Macintosh PC Exchange, as well as an array of pre-installed application software and CD-ROM titles.

In addition to their 75-MHz PowerPC 603 processor and the other features already mentioned, Macintosh Performa 5200CD computers ship with 8 MB of memory, expandable to 64 MB; either an 800 MB or a 1 GB (gigabyte) hard drive; front-mounted speakers; and front panel controls, including infrared remote control. Macintosh Performa 5200CD computers have a U.S. Apple price of $1,999 to $2,299, depending on the configuration.

Macintosh Performa 6200CD systems come with either 8 MB or 16 MB of memory, expandable to 64 MB; a 1 GB hard drive; an Apple Multiple Scan 15" display; and front panel controls, including infrared remote control. They have a U.S. Apple price of $2,399 to $2,999, depending on the configuration.

Macintosh Performa 631CD Product Details

The new Macintosh Performa 631CD computers provide 68LC040 performance with a built-in double-speed CD-ROM drive. Additionally, they provide expandability through an 030-compatible LC PDS slot—which can accomodate cards you've developed for Macintosh LC II, LC III, LC 475, LC 550, or LC 575 systems—and a communications slot. They can also be expanded to allow for video input, video output, and television viewing.

The system is based on the 66/33-MHz 68LC040 microprocessor and comes with 8 MB of memory, expandable to 52 MB; a 500 MB hard drive; a 14" Macintosh Performa Plus display; an internal double-speed CD-ROM drive; and an external 14.4-Kbaud data fax send/receive modem. The Macintosh Performa 631CD system has a U.S. Apple price of $1,599 to $1,649, depending on how it's bundled.

Macintosh Performa 6116CD Product Details

Another new entry in Apple's line of PowerPC processor–based computers, the Macintosh Performa 6116CD computer, is based on the 60-MHz PowerPC 601 microprocessor. It comes with 8 MB of memory, expandable to 72 MB; a 700 MB hard disk; a double-speed CD-ROM drive; an Apple Multiple Scan 15" display; and an external 14.4-Kbaud data fax send/receive modem. The Macintosh Performa 6116CD system has a U.S. Apple price of $1,899 to $1,949, again depending on the bundle.

Macintosh Performa 640CD DOS Compatible Product Details

Apple's latest market research data shows that its DOS-compatible Macintosh computers, which run both the Mac OS and DOS/Windows systems, are successfully attracting previous IBM PC–compatible computers to the Macintosh platform. (See Market Research Monthly on page 22 for details.) Apple's latest cross-platform Macintosh system, and the first to be made part of its consumer-oriented Performa line, is the Macintosh Performa 640CD DOS Compatible computer. Powered by both a 66/33-MHz 68LC040 processor and a 66-MHz 486 DX2 processor, the Macintosh Performa 640 DOS Compatible computer lets users run both Macintosh and DOS/Windows software.

It comes with 12 MB of memory (8 MB dedicated to the Mac OS, 4 MB dedicated to DOS/Windows), expandable to 52 MB; a 500 MB hard drive; an Apple

Macintosh Multiple Scan 15" display; a double-speed CD-ROM drive; and an internal 14.4-Kbaud data fax send/receive modem. The Macintosh Performa 640CD DOS Compatible has a U.S. Apple price of 2,299 to $2,349, depending on the specific bundle.

MPEG Media System Details

The newly released Apple MPEG Media System provides full-motion, full-screen video at an affordable price. The system works with the Macintosh Performa 630, 5200, and 6200 systems and provides a dual-buffer architecture with 16-bit CD-quality sound. The Apple MPEG Media System, which includes an MPEG video card and five MPEG CD-ROM titles, has a U.S. Apple price of $299.

New Newton Connectivity

If you're thinking of using a Newton device as part of a commercial or vertical-market product, you will be interested in knowing about a new set of integration libraries—that is, code libraries that make the process of direct data exchange between a Newton device and either a Mac OS–based or Windows-based application much easier to implement.

Until now, users have relied on the Newton Connection Kit (NCK) to export and import data between a Newton device and the files created by applications that support NCK data exchange. However, users have asked for a direct link between their Newton device and their applications. In addition, some application developers have created their own direct links to Newton devices, but such solutions have been platform-specific and difficult to implement.

To address the needs of these users and developers, Newton engineers have developed the Desktop Integration Libraries. Developers of Mac OS and Windows-compatible applications can use these integration libraries to connect directly to a Newton device and exchange and synchronize data with Newton applications. Also, Newton developers can use these libraries to implement more reliable connectivity to Mac OS and Windows-compatible applications, resulting in higher-quality products that can be delivered to market quicker and with much less effort.

These libraries offer Mac OS or Windows-compatible application developers a simple cross-platform, platform-independent API (application programming interface) for Newton application communications, with the following advantages:

• You need make only one development effort for both platforms. On the Mac OS platform, the libraries work in all the major C and C++ development environments. On the Windows side, they are implemented as dynamic linked libraries (DLLs).

• These libraries currently support the following connection types: MNP/serial (on both platforms) and AppleTalk/ADSP (Mac OS only). The Newton group is currently investigating a solution for infrared connections.

• Because of the libraries' modular design, software you write now can easily "plug into" any additional connection types that the Newton platform might support in the future.

• The libraries automatically handle data conversions that include byte swapping (important for data exchange to Windows-compatible applications) and ASCII to Unicode (Newton devices use Unicode, while desktop computers use ASCII). In addition, the libraries include hooks that allow encryption of the data being transmitted.

• Newton applications use rich, free-form data structures called *frames,* which may contain data that can't be anticipated by the fixed data structures used in C and C++ programs. The libraries have been written to accommodate Newton frames and the free-form data structures they provide.

Library Availability

Apple has created these libraries to encourage developers to create compelling Newton/desktop solutions. The libraries are currently in beta form and are available to any developer who signs a beta software agreement. For information, send e-mail to Internet address directconnect@newton.apple.com. ♣

Technology

CD Highlights

Tool Chest Edition, August 1995

The first thing you might notice about this month's disc is that its volume name ends with the suffix *TC.* Starting with this disc, we'll be using a slightly different naming scheme: Tool Chest Edition CDs will use the suffix *TC,* Reference Library CDs will use *RL,* and System Software, *SW.* We're doing this so you'll be able to pick a CD by edition from a server without having to remember which edition corresponds to what month.

Adobe™ Acrobat Exchange LE (described in detail below) makes its debut on this month's disc. Be sure to check out the indexed search capability, and the prototype Contents Catalog in the About This CD folder.

Also, as of this month we are discontinuing the On Location indexes, because the application is no longer sold or supported. Similar functionality is provided by the new Adobe Acrobat index for text in documents and by the System 7 Find File application for filenames, modification dates, and other file-related information. If you find On Location useful, you can still generate your own indexes.

So, in addition to updates to the Macintosh Easy Open and Open Transport SDKs, here is this month's new and revised stuff.

Adobe Acrobat Exchange LE 2.0.1

Adobe's Acrobat Exchange LE enables users to view, navigate through, and print any PDF files provided on the Developer CD. Exchange LE differs from Acrobat Reader in that it ships with the Verity search engine; users can now search across multiple documents.

Acrobat software is based on the PostScript™ page-description language, so documents converted to Acrobat retain their original look and feel. Acrobat also provides a variety of quality printing and viewing features. With it, you can do the following:

• copy text and graphics
• make use of any bookmarks or hyperlinks that have been created by CD content contributors
• "custom zoom" documents so you can read them more easily

Adobe Acrobat requires System 7.0 or later and at least 2048 KB of memory (although 3100 KB is preferable); it works with Macintosh computers with at least a 68020, or PowerPC, processor.

Here are a few issues surrounding Acrobat that you'll need to be aware of:

• Acrobat Exchange LE 2.0.1 does not support Kanji.
• Acrobat Exchange LE 2.0.1 installs substitution fonts appropriate to your system configuration at the time of installation. Later, if you either enable or disable QuickDraw GX, you will need to reinstall Acrobat Exchange LE 2.0.1 to ensure that the correct version of the required fonts are installed in their proper locations.
• To take advantage of the search capabilities of Adobe Acrobat, please add the ".pdx" files to your list of indexes to search. (These files are in the Acrobat Indexes folder on the top level of the CD.)

ASLM SDK 2.0GM

Apple Shared Library Manager (ASLM) 2.0 allows you to create and use dynamically linkable and loadable shared libraries. It supports System 6.0.5 and later, and it runs under the single Finder. It also supports native PowerPC shared libraries and clients.

ASLM 2.0 contains two parts. The first is simply a continuation of ASLM 1.1.2 for 680x0, which contains bug fixes and some minor feature enhancements. It is binary compatible with ASLM 1.1 clients and shared libraries; please see the Change History document for a list of changes and bug fixes. The second part of ASLM 2.0 is made up of tools and run-time support for "native" PowerPC shared libraries. The run-time support is provided by an extension called Shared Library Manager PPC.

See the release notes for more information.

Business Opp's—Germany

This brochure serves as a foundation for doing business in the German computer market. It provides important information about the market, as well as information pertaining to sales and marketing practices in Germany. You'll also find an address list and a reading list to assist you in researching more specific questions easily.

ColorSync 2.0 SDK

ColorSync 2.0 is an updated version of ColorSync, providing prepress-quality color matching and separations. ColorSync 2.0 offers substantial benefits for developers, including a standard architecture and profile format for color matching on the desktop that delivers the solution that users have asked for.

Developer Notes Update 7/95

This developer note describes the differences between Workgroup Server 9150 systems and Power Macintosh 8100 series computers. Use this note in conjuction with the Macinotsh Developer Notes *Enhanced Power Macintosh Computers* and *Power Macintosh 8100/110 Computer.*

HideMenubarEtc

This code demonstrates how to hide the menu bar, the desktop, or both. You hide the desk-

please turn to page 20

Inside This Section

Human Interface

House Hunting in the Information Age

By Peter Bickford

I'm writing this article while sitting in the back of my realtor's car, a place where it seems I've spent most of the last several weeks. As I type away on my beleaguered PowerBook (adjusting the backlighting so as to drain the battery at record speeds) the perky person in the front chats on about the many diverse areas we're passing. In the past hour, these have included a "surprisingly affordable" neighborhood where the locals apparently save water on the yards by planting old car hulks instead. Then there were the "executive communities" of overpriced houses placed so close together that you could reenact the neighborhood scenes from Rocky I in them. (I picture myself opening the window of my new $450,000 shoebox and yelling to my neighbor, "Yo Phil! Youze got an EtherNet transceiver that youze could loan me?" "Shaddup ovah thaa!!!" screams the "executive" in the house to my left.)

I've only been house hunting for three weeks now and I already feel my sanity slipping away. My head is packed with thousands of Important Facts about houses scattered over a 320-square-mile area. My wife and I have begun plotting house sightings with colored pins on a large street map like generals in a war room. And experience has given us cold insight into the hidden meanings of newspaper listings such as "4BR/2BA CHARMER, GRT. LOC!". (Read: "Four-bedroom, two-bathroom ruin, perched conveniently between a highway overpass and the gates of hell.").

As anyone who's been through it knows, house hunting has a way of consuming you totally—mentally, emotionally, and financially. It's a truism that you never find exactly the house you want, but I'm starting to believe that a lot of us just hunt to the point of exhaustion, then jump on whatever house is around that is anywhere near passable. The past few decades have seen the introduction of listing services and networked databases that give buyers access to potentially hundreds of thousands of house listings, but somehow all this data only seems to add to the confusion. The problem is that, as in many fields, house hunting is a matter of having way too much data available, and far too little real information.

Data Versus Information

Although there are more rigorous definitions, you can think of information as data that you care about. A weather satellite, for example, may transmit millions of numbers regarding cloud patterns back to earth as data, but it becomes information only if I can use the numbers to determine whether I should bring an umbrella with me tomorrow. Most of the facts and figures in our databases and spreadsheets are just data—they become information when we can use them to draw specific conclusions.

Turning data into information is like mining for gold. We first locate a source of data that seems to contain flecks of information buried deep within it. We then go through a process of searching and refining to separate the useful information from the mountains of debris. If we do our job perfectly, we're left with a handful of pure, relevant information without accidentally leaving anything valuable in the discard pile.

Searching for a home by going through newspaper listings is like heading off to the Yukon to pan for gold. It's time consuming, it's ineffective, and you simply can't work your way through enough raw material to stand a good chance of striking it rich. Realtors, on the other hand, can use their MLS database ("Multiple Listing Service") to rip through mountains of data about any home on the market—everything from its list price to the dimensions of the master bedroom. Their problem is that the tools they use for searching and sifting through it all are incredibly crude. Without a smart realtor to compensate, you'll miss out on seeing houses that might have been perfect for you, and you'll waste weeks looking at houses that are flat-out wrong. As a result, you'll probably pay too much for a house that wasn't really the best one available.

In this information age, we should be able to do better than this. In fact, if we can solve the same informational problems that make house hunting such a chore, we'll be well on our way to building better systems for doing everything from project accounting to product forecasting. Standing in our way, however, are problems of information arbitrage, data corruption, inflexible searching, and poor information visualization.

Sins of Omission: Information Arbitrage

In the financial markets, arbitrageurs are people whose entire living is based on noticing minute changes in trading values, then buying or selling before the rest of the world catches up to the same information. By law, all the figures are publicly available to everyone at the same time, but the successful arbitrageur is the one who finds the key information and acts on it.

As the saying goes, knowledge is power. When business is being conducted, what you know or don't know can make the difference between getting a good deal and getting stiffed. A big part of why real estate brokers are useful is that they have access to information you can get, but can't get easily. Sure, you could spend all day and night driving around the neighborhoods looking for "For Sale" signs, then tabulating the list prices, square footage, and so on. Real estate brokers, however, can get all this and more just by checking their MLS systems. This information is priceless—and it's precisely why MLS systems are kept closed to people who aren't members of the real estate profession.

Even when you do have access to the database, however, there are still arbitrage games being played. There is a "client" printout that gives you various statistics about the house, but intentionally omits key figures such as the loan amount (how much the person selling owes on the mortgage—and effectively the lowest price they can sell for). Other informational systems play similar games with key figures such as supply chain lead times, contract contingencies, or the real numbers on the balance sheet. Many times, these systems don't actually omit the relevant statistics, but instead bury them so that only the experienced eye can pick them out.

Whether this is ethical or not depends on the situation. Keep in mind, however, that when designing or dealing with informational systems, there is sometimes a financial motive for making some information easier to find than other information.

Data Corruption—When What You See Is Not What You Get

The second enemy of useful information is data corruption. I suspect that if H.A.L. in *2001: A Space Odyssey* was tasked with collating real estate listings (his mission was to "record and process information without distortion or error") he'd get through about two "3BR/1BA CHARMERS" before he blew the realtor entering them through the pod bay doors.

Often, you simply can't believe the data in the database—either because it was vague, mistyped, misclassified, omitted, or an outright lie. Searchers learn by experience which data is likely to be unreliable, and will avoid searching for it. For instance, a good realtor would avoid searching for the description "roomy," but might well use the more reliable "1800+" as a criterion in the Total Square Footage field.

If a database field is to be truly useful as a search index, it has to be (a) required and (b) well defined. This is a good reason to use pop-up menus, list boxes, and other restricted elements whenever possible for setting such fields. When the user must type a value into a searchable field, make sure you check its reasonableness and format before accepting it. Without some checks or filters, data naturally tends to degrade whenever humans are involved. Watch out also for fields that are filled in with random junk, because the person entering the value has no idea what to put there. If you really want to be able to search on that field in the future, make it easy for the user to enter the data correctly.

Flexible Searching: Getting the Information the User Really Wants

Most database searches work on strict Boolean logic to get their results. I might say, for instance, "Show me all the houses that cost less than $250,000 and have at least 1600 square feet, a fireplace, and either a pool or a spa." Although these sorts of searches are easy for computers to process, they're often not really what is needed. For starters, most users have trouble forming Boolean expressions that involve different sorts of terms—the "and" and the "or" in this case. It also doesn't help that in casual use, the English "and" ("Give me all the black ones and all the red ones") is really the Boolean "or."

The bigger problem, however, is that Boolean searches cut with razor precision, whereas the user's needs are more roughly defined. In the house example, for instance, would I be interested in a house priced at exactly $250,000 (instead of the "below" I specified)? How about a real beauty at $189,000 that had a spa and pool, but no fireplace? And if it had a great yard, could I put up with only 1580 square feet inside?

Good realtors spend the first several hours getting to know the clients' needs, and trying to distinguish the values they place on different aspects of a home. Often, they'll show the buyer around to several houses just so they can gauge the reaction to different features. Eventually, many home buyers ask their brokers questions like "Can you show me houses like that one we saw on Thistle Court, only without the crack house next door and for about $65,000 less?"

A good searching system would let users specify their criteria, then bring back not only the exact matches, but the close matches as well (ranked in order of closeness). A really good searching mechanism would combine this with the ability to do the "find things like this one" searches that pull up the key features of the home or other item in question and do proximity matches.

Finally, a truly great search engine would use knowledge of my relative priorities, as well as the topic in question, to find its answers. For instance, I could tell it that my "must haves" are a two-story house no more than 20 minutes from my work, for less than $250,000—oh yes, and I'd like a tile roof if I could get it. My dream search engine would then

- start by retrieving a list of all two-story houses
- check these against the houses with asking prices of 6–10 percent higher (more if they'd been on the market a long time), but whose loan amounts were less than $250,000
- cross-reference each house location with a trip planner program that considered average traffic speeds between the house and my work location—not just how far away it was in absolute distance
- put the houses with a tile roof nearer to the top of the list than similarly priced ones without

As you see, giving customers what they want and need can be very different from merely giving them what they ask for.

Visualizing the Results

The final step in turning data into information is giving the user some way of making sense of the results. This means more than just dumping out the database records. Think about the story that the information tells and try to present it in the clearest possible form. In a text-based environment, this means at minimum that you put the important fields first and try to differentiate them from the rest of the data for easy scanning. Spend some time asking users what fields are important for them to see, and make those easy to find. Similarly, avoid cluttering the output with useless text—remember that anything you display that doesn't help you is hurting you, so get rid of those redundant legal notices, time stamps, and code legends. Finally, organize and sort the output in an appropriate way, rather than just using the physical order of the database.

Sometimes, as is the case with the MLS system, the primary way you present information shouldn't be textual, but graphical. Good visualization systems give you a way to see the information in a way that brings insights ("Uh-Oh! That nice-sounding house on Center street is right on a busy intersection in a bad area of town!"). The master of this subject is Edward Tufte, who wrote the brilliant *Envisioning Information.* Close behind, however, are the creators of the game Sim City. I can't tell you how much I'd pay to be able to find a series of possible houses, see them displayed on a map, then overlay them with maps of relative crime rates, traffic density, and other displays that this $39.00 video game does so well. My living room recreation of London's Cabinet War Room with all its colored push pins is a feeble imitation of this. Still, it's worlds better than the record dumps that MLS provides.

For the first few decades of the information age we spent our time collecting and integrating vast stores of data. Now it's time we put the tools in place to make sense of it all. Who knows? It may even save future generations from spending big parts of their lives in the back of a realtor's car.

Till next time,
Doc
AppleLink: THE.DOKTOR

Peter Bickford, itinerant human interface engineer, was overjoyed to learn on last night's news that the "quality suburb" he'd been looking at most closely in his house-hunting quest is the scene of a new craze where fun-loving neighborhood kids throw Molotov cocktails through living-room windows. Stay tuned.

Introducing the Apple Color LaserWriter 12/600 PS

How Apple Is Delivering Top-Quality Printing at One-Tenth the (Previous) Price

When Apple's engineers set out to design the Apple Color LaserWriter 12/600 PS printer, they decided to make a device that could deliver top-quality output—both near-photographic color graphics and crisp text—and to deliver it at a price many times cheaper than other such devices. They succeeded in both their technical and price goals. On top of that, they developed new technology that caused Apple Computer, Inc., to file for 42 separate patents (!).

The new printer's technology provides the level of color printing quality usually associated with output devices that are ten times as expensive. We wish we could show it to you, because you really do have to see output from the printer to believe just how good it is. Until you can use one yourself, you'll just have to take our word for the fact that the Color LaserWriter prints photographs that look almost as vibrant as the originals, as well as clear, crisp *black* text, a combination we've never seen delivered by anything even close to its price.

The Color LaserWriter 12/600 PS has a U.S. Apple price of under $7,000, making it possible for a broad range of customers to own a high-quality printing device. Small and large design shops, desktop publishers, copy services, businesses of all sizes, and many educational institutions will now find it within their means to do their own high-quality printing.

What does this mean for you? Both a lot and a little. A lot, because your products that make heavy use of color just became much more useful; your customers who buy or use a Color LaserWriter 12/600 PS will be able to enjoy top-quality output at their desktops. A little, because you don't have to do anything for your product to take advantage of this latest LaserWriter printer; Apple engineers have done most of the work to ensure that what your customers see on their monitors is also what prints on the Color LaserWriter.

Those of you interested in assuring the quality of color output from your applications will want to look closely at ColorSync 2.0, which ships with the Color LaserWriter. ColorSync 2.0 is the latest version of Apple's technology that matches color between imaging devices, mapping from one device's profile to another to create accurate color output. ColorSync 2.0 includes the following enhancements:

- improved color-matching architecture and color-matching method
- international Color Consortium (ICC) profile format support
- increased support from application developers
- improved performance

ColorSync 2.0 is included on this month's Developer CD (path: What's New:ColorSync 2.0 SDK), along with a variety of technical documentation.

Even though the Color LaserWriter 12/600 PS doesn't require any work on the part of the vast majority of developers, we wanted to tell you a little about its abilities and the technical wizardry behind them—if for no other reason than to mark what we think is an important step in making high-quality printing available to "the rest of us."

What It Can Do

The Color LaserWriter 12/600 PS provides the following features:

- true 600-dpi (dots per inch) printing
- near-photographic quality for color images

• crisp, laser-quality monochrome printing for text and lines through unique compression technology

Additionally, the Color LaserWriter 12/600 PS prints at the rate of twelve pages per minute (ppm) for black-only print jobs and three ppm for color, both at 600 dpi. Further, its compression technology, which is primarily responsible for its low price, allows the Color LaserWriter 12/600 PS to work with approximately one-tenth the memory of comparable devices.

True 600-dpi Printing

As its name implies, the Color LaserWriter 12/600 PS delivers 600-dpi printing. It's said to be "true" because it meets the following six specific criteria for 600-dpi printing:

• The controller is able to represent data as a 600-dpi structure.

• The laser can scan at a frequency that forms 600-pixels-per-inch in a horizontal direction.

• For vertical resolution, the laser must be able to pulse at 600 dpi and the printer must perform all development processes—such as developing and paper transfer—at 600-dpi increments.

• It matches 600-dpi resolution both horizontally and vertically for clean edges.

• The diameter of the laser beam as well as the printing drum must both support the small dots of between 60 and 80 microns used to produce 600-dpi images.

• Toner particle sizes are extremely small—between 4 and 12 microns in diameter.

Near-Photographic Color Imaging

The Color LaserWriter 12/600 PS provides near-photographic color image quality equivalent to the output of a 2250-dpi bilevel printer. One of the reasons it can do this is that it employs a multilevel rendering process, allowing it to provide intermediate levels of gray or color for each pixel instead of just the two levels offered by bilevel printers.

Another reason that it delivers such great color is that it ships with Apple Color PhotoGrade technology, which introduces color printing innovations in five main areas. First, Color PhotoGrade makes up for the fact that color laser print engines by themselves can't render 256 color or gray shades per pixel—in the same way that the original Apple PhotoGrade compensated for the inability of monochrome print engines to render 256 grays per pixel. In doing so, Color PhotoGrade assures uniform and consistent print quality through the full spectrum of colors and shades of gray.

Second, Apple Color PhotoGrade supports a unique halftone screening method combining the best of both line-screen and traditional clustered-dot–screen half-toning methods with the ability to modulate each pixel. With a combination of these methods, Apple PhotoGrade treats light regions differently from dark regions and minimizes the overlap of black pixels with color pixels to deliver richer, more saturated color.

Third, Apple Color PhotoGrade uses unique scaling and filtering methods before halftoning an image. It does this to maintain the best quality when printing images with resolution lower than 600 dpi. Without this special scaling and filtering, other Adobe PostScript devices have difficulty doing a good job reproducing low-resolution images.

Fourth, Apple Color PhotoGrade is designed to work optimally with the Color LaserWriter 12/600 PS compression technique, called Apple Contone Compression Technology, to deliver better performance and quality. (See the next section for a description of this compression technology.)

Finally, Color PhotoGrade delivers finely tuned output of the print colors—cyan, magenta, yellow, and black—to produce bright, saturated graphics and text. Close examination of an image printed by the Color LaserWriter 12/600 PS shows that it provides a much more saturated, less blotchy look than other comparably priced printers.

Sharp Text With Top-Quality Images, Affordable Memory Requirement

One of the major technical leaps forward that makes the Color LaserWriter 12/600 PS able to deliver high-quality output at such a low price is the compression technology it uses, both for the frame buffer and I/O. Printers that deliver the kind of quality provided by the Color LaserWriter 12/600 PS can require over 120 MB of memory, making them very expensive. Because of the proprietary compression algorithm developed by Apple just for the new printer—Apple Contone Compression Technology—the Color LaserWriter requires only 12 MB of memory.

Most memory in existing PostScript printers is used up by the frame buffer, which stores an entire rendered page before it's printed. Using a "normal" frame buffer model, full-color images waiting in the frame buffer to be printed by the Color LaserWriter 12/600 PS would require 122 MB of memory. Apple Contone Compression Technology uses a compression ratio of 15:1 to reduce the frame buffer's memory requirements to just over 8 MB. The operating system uses another 3–4 MB of memory, depending on the size of the page, which is why the Color LaserWriter 12/600 PS printer ships with 12 MB of memory, expandable to 40 MB.

To preserve overall print quality for text, lines, graphics, and images, Apple Contone Compression Technology takes advantage of several observations Apple engineers made during testing of related color output technologies. They noticed that the human eye treats edges and interiors differently: With edges, the eye is sensitive to position, smoothness, and resolution; with interiors and images, the eye responds more to changes in color and tone.

Consistent with this, Apple Contone Compression Technology separates interiors and edges and then compresses them differently, retaining interior color depth at the expense of sharpness and the sharpness of edges at the expense of color depth. This results in sharp text and lines and rich color for images and filled areas.

Apple named the new compression technology because it's key in delivering near-continuous tone—or contone—quality. Imaging standards define contone quality as the ability to deliver 16.7 million different colors per pixel. A 24-bit monitor, for example, is able to deliver true contone quality. Few printers can accomplish this, although Apple Contone Compression Technology—and many of its other features—help the new Color LaserWriter come close to providing contone quality.

While we're on the subject of compression, the Color LaserWriter 12/600 PS also uses I/O compression to speed the transmission of data from users' systems to the printer. The Macintosh LaserWriter driver sends JPEG-compressed data over the network directly to the printer, which decompresses the data before sending it through the standard PostScript interpretation process. File transmission over the network can present a major performance hurdle, especially when printing large files; the Apple Color LaserWriter 12/600 PS printer's JPEG compression speeds this part of the process.

Performance Boost Through Special ASICs

The new Color LaserWriter also employs three specially designed ASICs, along with related coprocessors, to augment the performance of its AMD29030 RISC processor. One of these ASICs controls a video coprocessor that drives the print engine, taking care of most decompression as well as managing the halftone process and applying Color PhotoGrade laser control. Another controls a special compression/decompression coprocessor that aids the printing of complex pages. A third runs a color-matching accelerator, which incorporates the most computation-intensive aspects of the ColorSync 2.0 color-matching process so it doesn't diminish performance. The color-matching accelerator enables ColorSync 2.0 to transform a 5.5 MB image in just two seconds.

Other Features

The Color LaserWriter incorporates a variety of other features too numerous to list here. Among them is a technology called *Imaging Device Protocol,* which allows PostScript printers to print while taking care of other operations, such as handling status queries and receiving other print jobs.

Also, the Color LaserWriter makes the Desktop Printer extension, an element of the QuickDraw GX print architecture, available to non–QuickDraw GX users. The extension makes it possible for users to change printers without going through the Chooser, drag print jobs from one printer to another or from the Desktop to a printer, and send print jobs to multiple printers at the same time. The Color LaserWriter also supports Pantone, an industry-standard color-matching system.

In addition to working with the complete line of Macintosh, Power Macintosh, and Mac-compatible computers, the Color LaserWriter 12/600 PS is compatible with Windows-based systems. The printer comes standard with "plug and play" networking support, including LocalTalk, Ethernet, TCP/IP, and IEEE 1284 bidirectional parallel connections. A SCSI interface is included for storing extra fonts on optional internal and external hard disk drives.

See for Yourself

So far, this description of the Color LaserWriter adds up to about 2,000 words. Unfortunately, we can't ship you a copy of *Apple Directions* (or any complex graphics) printed on the printer; if we could, those images would be worth at least as much as the 2,000 words in convincing you that this is one hot product.

We think you'll want to see for yourself. If you're in the United States, you might want to wander down to the nearest Kinko's copy shop. Kinko's is making the Color LaserWriter 12/600 PS (along with the QuickTake 150 digital camera) available for customer use in over 750 Kinko's locations. If you're in another location, contact your Apple dealer. We think the printer is worth a look.

After you see it, we think you'll agree that Apple has just taken a huge step in opening up a new market with the printer and, at the same time, made widespread, inexpensive, quality printing a distinct possibility for the very near future. ♣

OpenDoc Human Interface FAQs

Embedding, Viewers, Part Factoring, and Cross-Part Compatibility

By Dave Curbow, Elizabeth Dykstra-Erickson, and Kurt Piersol

Here's the latest list of frequently asked design questions about OpenDoc software, and their answers, from the Apple Computer, Inc., OpenDoc human interface team. If you're not doing OpenDoc programming yet (heaven forbid!), this column will help you get an idea of what's at stake when you design OpenDoc components.

Q: I still have some conceptual questions about the boundaries of what should and should not be done with OpenDoc containers/parts. In particular, I am envisioning a data-flow application. Imagine that a set of parts, such as blenders, filters, and amplifiers, are "connected" together. There are also two types of containers that contain parts connected to each other in specific ways, one where these parts are "user-wirable" such that the positions cannot be changed and parts can not be added or removed (but the embedded parts can be edited); and the other where the connections between the parts are "hard-wired" and only specific types of parts can be added and removed (those that are compatible with the data-flow application). The parts can provide a user interface: A filter can have a pop-up menu for the amount of filtering, and a blender can have some GUI controls to control how much each source should be blended for the final output.

How should the parts be connected? Can I use OpenDoc to control the topology of how one part is positioned, oriented, and connected with respect to another? Or is this an abuse? Further, what kind of restrictions does OpenDoc place on how parts can be connected to each other? The underlying implication is that a part's behavior is dependent on other parts to which it is connected. Is this against the philosophy of OpenDoc?

A: The content kind suggests the appropriate functionality your editor should provide. For example, an editor for a drawing typically provides alignment and layering operations, while editors for text or spreadsheets do not. Certainly, your editor can control how it allows parts to be connected—you are the one who decides what is appropriate.

Generally, we encourage developers to allow *any* kind of part to be embedded. However, in some situations, it's appropriate for a container to only allow certain parts to be embedded. For your application, for example, it may not make sense to allow the user to embed a sound part in your container. Or, it may make sense to allow other parts to be embedded, but not allow them to be connected to the other parts.

For example, a sound part could be embedded and used to store a message about the purpose of the container.

To answer your second question, you can indeed implement two types of containers to do what you describe. However, you can achieve the same results by providing a "lock-like" function in your container. The "lock" is considered a setting on the container. For further detail, see page 58 in the *OpenDoc Human Interface Guidelines.*

Parts were designed to permit communication with other parts by means of the extension mechanism. Check out the "Extending OpenDoc" chapter in the *OpenDoc Programmer's Guide.*

Q: Can my container draw a graphical representation of the interconnections of its embedded parts?
A: Of course. You may want to have additional presentations as well. For example, you may also want to show a list view (ordered by name of component, perhaps?), or an outline view so that some groups of parts can be collapsed—whatever you think would be useful to your users. The View menu is the ideal vehicle for letting the user choose how to display parts; for further detail on using the View menu, see page 63 of the *OpenDoc Human Interface Guidelines.*

Q: I have a question about read-only documents. We have a requirement for allowing local developers to customize our maintenance console for particular installations. The end-user, however, should not be able to **accidentally** *rearrange the maintenance console. I envision a local developer creating a "maintenance console" document, and providing that along with our document shell to the user. Can we "lock" this document so that the user can't move stuff around or damage it in any way? The user would not alter the content of our document, but would just view it. Does this make sense? Should I allow the user to do anything to the document?*
A: There's no standard mechanism for what you describe, but it makes sense for some containers to supply a "lock" setting. You must use your discretion. Generally we think you should allow the user to edit the document, but the changes can only be saved through the Save A Copy command to a new document.

Q: I'm writing a viewer for my kind of content because I want users to always be able to see my kind of part. In what ways should my viewer differ from an editor?
A: A viewer is extremely handy for doing exactly what you want: making sure that users can always read and print content created by your editor. There are several significant differences between editors and viewers. First, your viewer should disable commands that allow the user to modify the contents of any part that the viewer is handling. That is, the Cut, Paste, Clear, and Insert menu commands should be dimmed. The Delete key should also be disabled. If your editor takes keyboard input (for example, if your editor is a text part), then your corresponding viewer should ignore the keystrokes.

Furthermore, because the user needs some feedback indicating that the system saw the keystrokes (text input or deletion) and decided to ignore them, you should play the system beep. If the user keeps pressing keys, you may want to display a dialog box saying something like "This content cannot be modified by typing. Keystrokes will be ignored."

In addition, your viewer should not accept a drop (the end of a drag-and-drop) operation. You should also remove menu commands such as Font Size. It doesn't make sense to simply disable these operations, because the user can't make them available.

Editors should display splash screens as infrequently as possible—subject to legal requirements. However, we encourage you to let your viewer display its splash screen occasionally. Viewers are free—and you're providing a service to users by letting them look at content they might not otherwise be able to view. So, feel free to have your viewer display a splash screen occasionally to remind users that you also have a full-featured product in which they might be interested.

And finally, make sure that the icon for your viewer looks like a viewer icon and not an editor icon. To see the differences, see the examples in the *OpenDoc Human Interface Guidelines.*

Q: I'm breaking my application up into OpenDoc parts, and I'm having some doubts about how to divide up the functionality among the parts. Do you have any advice?
A: The underlying issue is granularity, one of the most difficult parts of designing for OpenDoc. The reason for this difficulty is that you are trying to anticipate what users want. Choosing where to put part boundaries is as much art as science, and you should take the time to do it right, because it will make a big difference in the success of your product.

One of the best approaches is to "make up stories" about what the user will do. As an example, here are two stories, one for someone using a database report, and one for someone creating that report in the first place.

The first story goes something like this: A user is reading a document and sees a button labeled with the name of some database report mentioned in the document. He presses the button, and a report appears in a separate window.

The second story might go like this: A power user is creating a document, and she wants to include a report from a database that's available within the company. She creates a special "report button" that will display the report. The report itself has a bit of text and a chart. The user creates a form-like document with a little spreadsheet to calculate the chart values and links the spreadsheet to the database using a database-query part. She then creates a chart, links it to the spreadsheet, adds an explanatory paragraph, and then hides the spreadsheet and database part from the reader. The resulting form is dropped into the "report button" where it is ready for later examination by a reader.

Now the next trick is to look for the "natural" breaks in the parts at hand, based on the way you phrase the stories. Try to phrase the stories without using jargon; if you can easily phrase them without inventing new terms, it's likely that ordinary people will understand where the boundaries are between the parts. If you do use jargon, make sure it's commonly used by your target customers. (Reading magazines is a pretty good way to find out the common jargon, since they specialize in being understandable.)

The first story involves using a button to display a "hidden" report. You want to let people include reports as an annotation, of sorts. Is a button the best way to do it? That depends on your user, but since a lot of people understand pressing buttons to get more information, it's a pretty good device.

The second story involves a power user or programmer putting together a "canned" report. The story mentions several "natural" elements—query, button, chart, and spreadsheet. Implicit in the scenario is a container—probably something like a forms container.

It would probably be best to make each of these items a separate part, and then use linking and scripting to hook them up.

To be specific, you might make a scriptable database query part that can be the source of a link, and use a spreadsheet part as the destination of the link to display the results. The spreadsheet part would have the usual features for formulas and such. The chart part would have formatting features for fonts, 3D bars, and the like.

You might create a scriptable forms part that can serve as a layout tool and as a source of input for the query if needed. The forms part would also be able to hide embedded elements, so that users would only see the forms elements and table. This hiding feature is really just a formatting tool, not a required feature, but any editor can provide this feature if it makes sense. A word processor or page layout editor might provide this capability.

Finally, I'd use a button part to open the forms part and cause it to update with query results. This "annotation button" part would remember the forms part (or any embedded part) as well as run a script when the button is pressed. The button would then be embedded wherever the user wanted to put the canned report.

The advantage of this approach is that you have a powerful set of general-purpose tools to sell, rather than a single, specific part. Each of the parts can be reused in a different solution later: The forms tool might be used to create simple entry pads, instead of being a database display mechanism; the button might display other kinds of annotation, not just canned reports; and the spreadsheet part might be used to display the results of non-database links or just tables entered "by hand." (The database query part alone is a relatively single-purpose part.)

The fact that the annotation button, the forms package, and the spreadsheet are such generically useful tools also suggests something else: that other developers may be building them and that you could license these tools for inclusion in your own product, rather than building them all yourself.

Q: What will happen when users try to integrate parts that were created by different part developers, but that were not tested together for compatibility? Who does the user call if the parts are not compatible with one another?

A: This question has two answers. The first answer speaks to you, the developer. The second answer speaks (as well as the situation permits) to the user's predicament.

To the developer: Component Integration Laboratories (CI Labs) will provide a validation suite that developers can use to validate their code to ensure that their editors meet a minimum set of integration criteria. If you wish, CI Labs will also validate your editor with many other editors, for a fee. Editors that receive the CI Labs sticker are guaranteed to meet the integration criteria; however, this does not guarantee that all of the parts a user wants to use will work well with each other. Of course, vendors are ultimately responsible for making sure the editors work; CI Labs can help to make sure they work with each other.

As a developer, if you discover an integration problem, you should inform your platform vendor (for example, Apple, IBM, or Novell) and CI Labs so that the validation process can be improved.

To the user: You can find out who to call either by making the part active and looking at the "About" box, or by selecting the part and choosing the Part Info command from the Edit menu. Then you will have to work with the vendors of both parts to determine which part is in error (or whether some other circumstance is causing the error).

We know that's not the solution you want to hear, but it's the only one the situation permits. In fact, it's no different than what you have to do when you get an error that you suspect may be caused by a third-party extension. ♣

CD Highlights

continued from page 13

top simply by creating a large background window and filling it with the appropriate pattern. You hide the menu bar by setting the Menu bar height to 0, updating the gray region, and then forcing all windows to redraw. You can also remove the rounded corners on every screen device by using a method similar to that used with the menu bar.

Introduction to Copland

This document discusses the direction of future versions of the Mac OS and how the next version, Copland, helps implement them. It also discusses the specific goals of Copland and its major benefits to customers and developers.

LabelMenu

This code demonstrates a program with a Finder-like label menu. Each label menu item has a 12 x 16 pixel 'cicn'. The color and name of all the items are updated if the user changes anything in the Labels control panel. This also demonstrates how to change a menu tile to an icon.

Macintosh Developers Guide

The *Macintosh Developers Guide* is a showcase for development tool products, vendors, and services. From application development tools to solution tools to client/server tools, this guide provides you with information to help you make informed decisions on tool purchases, which will ultimately help make your software development efforts more effective and efficient.

MacODBC SDK 2.0b1

MacODBC 2.0b1 is the first beta distribution of the next version of the MacODBC Driver Manager, Configuration Manager, and developer APIs. This package is for developers who wish to create applications or database drivers for ODBC. It replaces MacODBC 1.0.1.

This version includes the following features:

- support for native Power Macintosh applications and drivers
- compatibility with the ODBC 2.1 specification
- a new Cursor Library, which eases indexing through large data sets

CD Highlights

• a 680x0 verision of ODBC 2.0 that's compatible with ODBC 1.0.1 drivers and applications

MoreFiles 1.3.1

MoreFiles is a collection of high-level routines written over the last couple of years to answer File Manager questions developers have sent to Apple Developer Technical Support (DTS) engineers. The routines have been tested (but not stress-tested), documented, and code-reviewed by DTS. This release adds new routines and fixes several bugs.

MoreFiles provides high-level and FSSpec-style routines for parameter-block–only File Manager calls; useful utility routines that perform many common File Manager–related operations; a robust file copy routine; a recursive directory copy routine; catalog searching routines; high-level and FSSpec-style routines for Desktop Manager calls; and routines for dealing with pathnames. See the file !MoreFiles Read Me for a description of fixes and improvements in version 1.3.1.

PlainTalk Speech Technologies

This package contains version 1.4 of English Text-to-Speech and Mexican Spanish Text-to-Speech, and a lot of developer information about using them, including Speech Manager documentation and an AppleScript scripting-addition called Say, which makes it easy to have AppleScripts talk.

QuickDraw 3D 1.0

QuickDraw 3D is a cross-platform 3D graphics library developed by Apple Computer, Inc. This folder contains the 1.0 release of QuickDraw 3D for the Mac OS.

QuickDraw 3D goes beyond providing 3D graphics. It provides an integrated solution for both users and developers. QuickDraw 3D encompasses a standard file format (3DMF), acceleration layer, input architecture, high-level geometries, and extensibility.

QuickDraw 3D is a shared library for PowerPC processor–based computers running the Mac OS. The API is written in C, with support for development using C++. QuickDraw 3D allows for immediate mode and retained rendering. The API is object based and provides a large number of geometry types. The file format accomodates both text and binary modes, with encoding for "endianness" so that files can be transported to other platforms.

Please read the developer notes in the Development folder for vital information you'll need during the development cycle.

QuickDraw GX 1.1.2

This folder contains the latest release of QuickDraw GX software. It provides all the code you need to use QuickDraw GX, plus documentation, development tools, sample code, and human interface guidelines.

ROM Build/Download 3.2.4

This is a series of tools (with source code) that allow developers to build a declaration ROM that is suitable for downloading from the assembled and linked image.

Server Remote Control 1.1

The two applications contained in this folder, Remote Control and Server Controller, make it possible to start or stop a file server running on a Macintosh computer from a second Macintosh computer at another location on the network. They provide an example of applications using asynchronous PPC Toolbox code for communications, and of a faceless background-only application used as an agent to control the remote system.

Source code, written in Metrowerks Pascal, is provided for both programs. The applications were built using the Universal Interfaces Version 2.0. Note that the functionality provided by these applications can be easily duplicated with the scriptable Finder and AppleScript.

ShowInitIcon

Use this code to display an icon when starting an extension. This is a replacement for the many other ShowIcon-style utilites. It's written entirely in C, and works with several different C compilers. A Pascal header is included as well.

Solutions & MM Developers Guide

This folder includes the *Macintosh Solutions and Multimedia Developers Guide.* To view a listing for a company or a product in the guide, click the company name, product name, or page number in the table of contents. Alternatively, use the bookmarks on the left-hand side of the screen to select an individual listing. Bookmarks are listed by product name and company name and are also divided by category. The introduction contains detailed descriptions of the categories.

Standard File Samples

This folder provides the following three samples involving standard file dialog boxes:

• CustomGetFolder demonstrates a CustomGetDialog for selecting a folder or volume. This sample is based on Steve Falkenburg's sample of a few years back, the sample code in *Inside Macintosh: Files,* and the *Human Interface Guidelines.* It also has Balloon Help strings for the Select button.

• CustomPutAppend shows a CustomPutDialog with an Append button that does not ask the user if he or she wants to replace the existing file. It's also based on the sample code in *Inside Macintosh: Files,* and the *Human Interface Guidelines.* It also has Balloon Help strings for the Append button.

• CustomPutSuffix demonstrates a CustomPutDialog with a Save button that checks to see if several files (with the name *filename + suffix*) already exist. A dialog hook procedure is responsible for presenting separate Replace dialog boxes.

Toolbox Assistant Update #2

This folder contains a new *Macintosh Programmer's Toolbox Assistant* database for QuickDraw 3D.

UnmountIt 1.2

UnmountIt lets you easily unmount and eject shared volumes when Macintosh file sharing is in use. UnmountIt demonstrates how to enable user server control calls to control AppleShare/file sharing, how to drop volumes on an application icon, and how to unmount and eject a disk. All Metrowerks Pascal source code is included.

Coming Next Month

Next month's CD will include all of the technical documentation you could possibly want, in one big, happy, searchable Acrobat index.

Alex Dosher
Developer CD Leader

Business & Marketing

Market Research Monthly

Inside This Section

Power Macintosh 6100 Attracting Customers From DOS/Windows Platform

Apple's strategy to attract DOS/Windows and new users to the Macintosh platform with its DOS-compatible Macintosh systems is working, according to a proprietary customer study just completed by Apple Computer, Inc.

The research shows that the Power Macintosh 6100 DOS Compatible computer is successfully attracting its target customers—that is, customers choosing between Macintosh and IBM PC–compatible systems (so-called fence-sitters) and customers who currently use PC-compatible systems.

You can expect your efforts to develop "native" PowerPC applications to pay off as Apple builds market share with its cross-platform strategy, especially from among the ranks of fence-sitters and PC-compatible customers, who, according to the data, purchase more native PowerPC products than traditional Macintosh customers. The data also points to the higher education market as a possible target for new PowerPC software, since the DOS-compatible Power Macintosh 6100 appears to be more popular than other Macintosh models among users in that segment.

We'll present the data in a moment, but first some background.

A couple of years ago, market researchers at Apple discovered that the great majority of IBM PC–compatible users simply wouldn't consider purchasing Macintosh computers because they couldn't run DOS and Windows applications. The same was true of many first-time buyers, who needed to be sure that their first computers could run the widest variety of software available.

To build market share, Apple faced the challenge of having customers from the PC/DOS/Windows world, and first-time buyers, at least consider buying the Macintosh computer. Out of that situation were born the first DOS-compatible Macintosh systems—computers with both Motorola 68040 and Intel microprocessors that could run both Macintosh and DOS/Windows software. The Power Macintosh 6100 DOS Compatible computer, one of the latest members of Apple's DOS-compatible family, employs both a 66-MHz PowerPC 601 chip and a 66-MHz 486DX2 chip.

This month's data comes from Apple's study of a representative sampling of early customers of the DOS-compatible Power Macintosh 6100 computer. During its quarterly *Recent Buyers Study,* Apple interviewed several hundred customers who purchased DOS-compatible Power Macintosh 6100 systems, as well as purchasers of regular Power Macintosh 6100 systems, between January and March of this year. The study shows that the DOS-compatible Macintosh systems are successfully attracting former DOS/Windows users and new customers, at a greater rate than Power Macintosh systems that use only a PowerPC microprocessor.

Key findings of the study include the following:

✔ More than 50 percent of buyers of DOS-compatible Power Macintosh 6100 computers are either buying their first system, replacing an IBM PC–compatible machine, or acquiring a Macintosh system to use with an existing PC.

✔ DOS-compatible Power Macintosh 6100 systems are replacing IBM PC–compatible systems at three times the rate of regular Power Macintosh 6100 systems.

✔ Nearly half of customers of DOS-compatible Power Macintosh 6100 computers also use a PC-compatible machine; when they use their Macintosh systems, they spend about three-fourths of their time in the Mac OS environment.

✔ About three-quarters of DOS-compatible Power Macintosh 6100 customers have used an IBM PC–compatible system, much higher than buyers of other Macintosh models.

✔ Users of the DOS-compatible Power Macintosh 6100 purchase more native PowerPC applications than users of the regular Power Macintosh 6100 model.

✔ Buyers from the higher education sector constitute a greater percentage—fully one-fourth—of the DOS-compatible Power Macintosh market than buyers of other Macintosh models; by comparison, only about half as many regular Power Macintosh 6100 computer customers are in higher education. In the other market sectors (such as home, K–12, business, and so on), there's little difference between adoption rates for DOS-compatible Power Macintosh 6100 systems and regular Power Macintosh systems.

What does this mean for you? As we've said many times, keep developing native PowerPC applications. Apple is doing its part to attract new customers for those applications—and, according to this month's data, at least—is doing it successfully. ♣

Developer Outlook

Commercial Developers on the Benefits of AppleScript

By Kris Newby

When the first printing press was introduced to Europe in 1458, politicians and the Church denounced it as an instrument of the devil, one that would be used to spread subversive ideas. They were quick to recognize that *language is power.* But try as they might to hold on to this power, written language became widely accessible, literacy spread like wildfire, and the Renaissance began.

Some 500 years later, *scripting languages* are making computers more accessible to mainstream users. And while hardcore programmers may dismiss these simple languages as a necessary evil, the accessibility that they provide may well lead to a renaissance in computer use in the coming decade.

Many commercial developers are beginning to realize that AppleScript doesn't just benefit end-users and solutions providers. Once they've built AppleScript support into their products, they find that it helps them win more customers, it reduces support costs, and it optimizes internal development processes. In this article, six commercial Macintosh developers discuss why they decided to support AppleScript and how this technology has improved their business's bottom line. (For information on how AppleScript fits into Apple's overall technical strategy, see page 6 of the article "Strategy: 1999.")

AppleScript Ain't Latin

Before the European Renaissance, Latin was the common language of scholars and the elite. But once the printing press made its way across Europe, books published in easy-to-grasp native languages were the "bestsellers." In a way, C++ is the Latin of the Macintosh programming elite, and AppleScript is the lay person's easy-to-grasp computer language.

Macintosh users appreciate AppleScript because it enables them to easily

- write scripts to automate redundant operations, thereby avoiding boredom and errors
- automate complex operations that are difficult to remember and repeat
- create links between off-the-shelf programs, enabling the development of workflow solutions that intelligently carry out complex series of operations
- customize Macintosh systems to reflect individual or organizational preferences

From a programmer's perspective, AppleScript is an object-oriented programming language, and as such, it has its own API that you have to support if your users are to take full advantage of Apple's Open Scripting Architecture (OSA). There are three levels of support that your applications can provide: scriptability, attachability, and recordability. And of course, since AppleScript is an integral part of System 7.5, many capabilities built into AppleScript are always available to you and your users, no matter what level of scriptability your application supports. These built-in features include object-oriented programming with inheritance, powerful list and record operations, and persistent data, to name a few.

AppleScript Benefits at a Glance

It's been almost three years since AppleScript first shipped, and many developers who've adopted this technology have begun reaping the rewards of their efforts. Most of the developers we spoke with all shared a common belief—that it's becoming extremely important for individual software applications to be able to work together, and AppleScript will be the glue that facilitates this inter-application communication. What's more, these developers appreciate the fact that AppleScript frees them from the burden of anticipating every end-user need. Because of the ease with which customers can use AppleScript to customize their applications, developers are able to focus on their core competencies.

Beyond the advantage of inter-application communication, the developers I interviewed appreciated the following AppleScript benefits.

- *Enhanced sales of commercial products.* By making your commercial product scriptable, you increase its ability to be used in specialized markets. VARs and ISVs can more easily build your product into custom software solutions, and this, in turn, will provide your product with a competitive advantage and enhanced sales.
- *Extended product life cycles.* Within your company, technical people can create AppleScript scripts to enhance an application's functionality without revising core code. The speed with which you add new features, scripted extensions, and patches helps extend the life of your product between major revisions.
- *Reduced programming redundancy.* Because AppleScript provides standard functionality across the Macintosh platform, it eliminates the need for you to write application-specific scripting tools or macros for each product you develop.
- *Simplified product localization.* AppleScript scripts are compiled in a dialect-independent format, so text within those scripts is automatically displayed in the default language of a given system. For example, a script written and compiled in English, then opened on a machine running the KanjiTalk operating system will display AppleScript scripts in Japanese. This enables you to support native-language scripting with minimal, incremental AppleScript work.
- *Positioning for future market opportunities.* Apple Computer, Inc., will increasingly use AppleScript as a means of delivering new technologies to the Macintosh platform. More specifically, OpenDoc, Apple Guide, and many system software extensions will interface with the Mac OS through AppleScript and Apple events, so that seamless updates will depend upon your product's scriptability. And by supporting AppleScript, you'll be able to take better advantage of market opportunities that arise from these developments.

In the following six vignettes, you'll be able to see how a wide variety of commercial developers have reaped these benefits of AppleScript technology.

Nyhthawk Productions: Using AppleScript to Build Workflow Solutions

Sal Soghoian is a custom software developer and the creator of Sal's AppleScript Snippets, a set of 70 editable scripts designed to extend the functionality of QuarkXPress, a high-end publishing application.

Soghoian creates AppleScript–based workflow solutions for corporate clients across the United States. Some of his more interesting projects include the following:

- *An ad layout tool.* Soghoian used AppleScript, QuarkXPress, and Filemaker Pro to create an automated ad layout tool for a well-known law publisher. To set up a new ad, a person chooses from among five standard ad formats. The user then responds to a series of menu-based queries, adding graphic elements such as "New" or "Improved" icons with a click of the mouse. Using this tool, the publisher has reduced page layout time from 45 minutes to 3 minutes.
- *A catalog builder.* A Texas-based company that sells products to grocery stores hired Soghoian to create a script-based solution for building food product catalogs. Layout of the catalogs is automated with AppleScript, QuarkXPress, and Canto's Cumulus image database. Now the layout process, which used to take five people one week, takes only one day.
- *Online information delivery.* Soghoian created a script that automatically extracts text from a financial company's daily investment reports, then places it in a text stream for automated delivery to online customers.

Soghoian relies heavily on commercial products to build his custom software solutions, and scriptability is an essential criterion for selecting commercial "building block" products. He sees other businesses making similar choices. For example, he recently witnessed a major book publisher switch from a high-priced photo manipulation application to Apple's $99 PhotoFlash application, because PhotoFlash's scriptability and recordability enabled the publisher to extract and crop Photo CD images in batch mode.

"You can no longer afford to be a stand-alone, do-all software product in this market," says Soghoian. "The next wave of growth in the software market will be in workflow solutions, because mainstream users don't want to have to depend solely on C programmers to automate processes. By making your commercial products scriptable, your company can capitalize on this growth.

"I see scriptability becoming incredibly important in two areas—publishing and communication," adds Soghoian. "Both of these industries involve time-consuming tasks where scripting can deliver significant productivity gains."

Soghoian considers AppleScript to be the best scripting language available today, and through scripting classes he's taught at publishing conferences, he's recently seen tremendous growth in support for this technology. (Besides working on an interactive AppleScript/QuarkXPress CD, Soghoian will be teaching several scripting classes this year, including one at the QuarkXPress Conference, Thunder Lizard Productions, September 21–22 in Orlando, Florida, and the QuarkXPress Scripting Crash Course at the Seybold Publishing Seminar, September 27–29 in San Francisco, California.)

CambridgeSoft: Improving the Chemistry Between Customers

CambridgeSoft Corporation is a software company that specializes in helping chemists and other scientists draw, model, analyze, and manage chemical information. Its ChemOffice product suite includes ChemDraw, a tool for drawing 2D chemical structures; Chem3D, a modeling and analysis tool for 3D chemical structures; and ChemFinder, a chemical information management system.

Joel Wolff, Ph.D., CambridgeSoft's customer service manager, uses AppleScript to quickly deliver new product features to customers who would otherwise have to wait on the schedules of busy programmers.

"Because our products are so specialized," says Wolff, "we often depend on AppleScript to help us interact with other products. For example, we recently developed a dihedral driver that rotates a molecule, analyzes its bonds, then exports bond energy levels to the charting function in Microsoft Excel. With AppleScript, we were able to add this feature in a few days, without the help of our programmers.

"Customers often call our technical support department asking for advice on how to add custom features, and in many cases, we can supply customers with a script that meets their needs.

"We're also using AppleScript functionality on our World Wide Web site. This service enables our customers to search for and download specific chemical structures from our ChemFinder library. Though we don't charge anything for this service, we feel it provides us with enhanced customer loyalty and sales.

"ChemOffice is one of the most widely used chemistry-related products available on the Macintosh today, and I think this product, coupled with flexibility that scriptability provides, is why so many chemists use the Macintosh."

Quark, Inc.: Using AppleScript to Localize for Languages

Serving the diverse needs of prepress and publishing companies is a gargantuan task, and Quark, Inc., has used AppleScript technology to help meet these needs. Its high-end publishing product, QuarkXPress, is fully scriptable, and the company aggressively markets this competitive advantage through ads, press releases, and online script libraries, as well as through value-added scripts included with its CD-ROM version of QuarkXPress.

Quark employees also use AppleScript to automate an important aspect of their product development—the localization of documentation into seven different languages.

Garry Dufford, a software localizer at Quark, talks about the scripts he developed for Quark's localization department: "When we lay out our user documentation with QuarkXPress, we have to flow chapters, captions, and sidebars as separate blocks of text. Before we implemented AppleScript, translations required someone to cut and paste all these text blocks into one file, send them to the translators, then cut and paste them back into the original document.

"To automate this tedious process, I wrote three scripts. The first script, 'StorySucker,' pulls all the text out of a Quark document, merges it into a single text file, then inserts flags that tell the script where to put each block of translated text later on. Before we send the English text to the translator, the 'Replacer' script checks the documentation for common interface commands and replaces them with appropriate translations, which are extracted from a Filemaker Pro database. Finally, the 'StoryPourer' script flows translated text back into the correct spots in the Quark document. To further expedite this process, we can run the scripts in batch mode overnight.

"What I like best about AppleScript is that I can create these types of solutions quickly, without having to be a full-fledged programmer. Computers love redundant tasks and humans hate them, so why not let scripts perform these tasks faster and more accurately than we can?"

Canto Software GmbH: Streamlining Picture-Perfect Businesses

Canto Software, a Germany-based image and media database company and the creator of the Eddy-winning Cumulus image database, was initially a reluctant supporter of AppleScript. But soon after building scriptability, recordability, and attachability into their products, they began to see tangible benefits. Now with the zeal of the converted, they're even encouraging other developers to support it.

"We realized we needed a technology like AppleScript at the point where we became overwhelmed with special customer requests," says Jennifer Neumann, president. "For example, many of our customers needed a facility for scanning multiple images in a batch mode, but each required a unique file-naming convention: Some wanted date-based filenames, others wanted custom alphanumeric formats, and so on. Though this sounds like a simple request, we found that our programmers were spending weeks on simple user interface changes such as this. Ultimately, we decided that using AppleScript was the best way to address these varied customer needs."

AppleScript capabilities have even enabled Canto to compete with high-end, host-based prepress software. What used to require proprietary software and an expensive host system can now be done inexpensively with a Power Macintosh, QuarkXPress, Cumulus, and AppleScript.

Neumann talks about how the publisher of *Time* magazine uses its Cumulus image database and AppleScript to process the 500 images or so that they gather each day from news services such as API and Reuters: "With this system, *Time* downloads images, converts images into a format suitable for its publication, then catalogs them in an easy-to-find manner. They also use AppleScript to automatically discard images that have been stored on its computers longer than 23 days."

Realizing the importance of solutions marketing, Canto is in the process of developing an organized procedure for responding to customer requests, pairing these customers with value-added resellers and AppleScript programmers.

Aladdin Systems, Inc.: Communicating the Advantages of Scriptability

"As a developer of communication products," says Leonard Rosenthol, director of advanced technology at Aladdin Systems, "we realized that automating repetitive communication processes, such as logging on to online services, was important to our users. Then, rather than creating our own scripting language, we saved ourselves and our customers time by standardizing on AppleScript technology."

Rosenthol described typical end-user applications that take advantage of Aladdin's StuffIt file-compression utility and AppleScript. One script helps employees in a large company compress their daily reports and then automatically e-mail each to specific managers with one keystroke. Another prepress script extracts images from a Quark document, processes each in Adobe Photoshop, then compresses and e-mails them to a local print shop.

"I think recordability is the most important AppleScript service to provide for customers," says Rosenthol. "Very few users actually want to write scripts—they want to use them. With recordability, you can let the computer worry about translating a complex set of tasks into the proper AppleScript syntax."

Aladdin publicizes its AppleScript support through packaging copy, press releases, brochures, and Apple solutions guides. "Users who need this kind of capability look for it," says Rosenthol. "In fact, I think one of the reasons that our SIT.com compression product sold so well in the last year is because it's the only scriptable product of its class."

Novell, Inc.: Aligning Its Products to the OpenDoc Model

"In many ways, I think the greatest benefits of AppleScript are yet to come," says Dave Harding, the product manager of WordPerfect marketing. "As more developers support this technology, we'll reach a critical mass of compatible products, enabling users to more easily custom-tailor software solutions. AppleScript is really the precursor to the OpenDoc model, proving that individual software companies can collaborate to provide better customer solutions."

Novell is currently practicing what it preaches, by collaborating with DeltaGraph to provide in-place "live" chart creation within WordPerfect documents.

Harding adds, "Our customers will more clearly see the advantages to this approach when we release our AppWare visual application tool. With its Apple events/AppleScript backbone, users will be able to build custom applications, called *QuickTasks*, quickly and easily. For example, a financial company could launch a QuickTask that would enable them to download money market rates from an online service, save them in an Oracle database, then publish this information in a WordPerfect document. This kind of capability will not only provide end-users with one-button, intelligent capabilities, but it will get IS managers excited about scripting

Publicizing Your AppleScript Support

Here are several communication vehicles for letting computer users, value-added resellers, and independent software vendors know about your scripts or about your product's AppleScript compatibility.

• *The Macintosh Solutions & Multimedia Developers Guide.* To list your scriptable product in this guide, contact JointSolutions Marketing, 408-338-6471, jsolver@aol.com. The closing date for the next issue is September 5, 1995.

• *Scripting SIGs.* Most of the larger Macintosh user groups run scripting special interest groups (SIGs) that provide you with a focused forum for getting the word out about your product's scriptability. To obtain contact information on Macintosh user groups, contact Sam Decker, UGCSam@eworld.com, 408-461-5700 ext. 202.

• *Online bulletin boards.* Several of the major online services run scripting bulletin boards, including America Online (Computing Forums:Special Interest Groups:AppleScript) and eWorld (Macintosh Development Forum:Languages/Tools:Scripting:AppleScript). You can also send announcements to users interested in Macintosh scripting across the Internet by using the "Macscripting" mailing list. To subscribe or find out more, send an e-mail to listserv@dartmouth.edu.

tools and their inherent power to create custom solutions."

Scripting Languages and Computer Literacy

Just as no one fully understood the change that printed language would bring to civilization, developers and users are just beginning to realize the benefits of scripting languages such as AppleScript. Someday computer "programming" as we know it will be as simple as telling your computer, "Get Apple stock quotes and plot them for the last month." Scripting languages are the first step on the road to seamless natural-language programming. And by supporting AppleScript, you're also taking the first step toward cross-platform OpenDoc compatibility. And, who knows, your efforts may contribute to widespread computer literacy and a renaissance in computer use in our own time. ♣

Marketing Feature

Ideas for Saving Packaging Costs (and the Environment)

By Kris Newby

"I speak for the trees."—The Lorax

There's a tongue-in-cheek saying in the packaging design business that goes, "If you can't design a *good* package, then make it big and red." The software industry, unfortunately, appears to have taken this advice literally. All too often, efforts to attract software shoppers has led to bigger rather than better software package designs. And while the motivation behind this trend is understandable—after all, packaging is often your best advertising vehicle—the trend is detrimental to the software industry as a whole.

First, the growing size and proliferation of packaging form factors is angering software retailers, who would like to squeeze more software titles onto store shelves. (There are also quite a few developers who wouldn't mind more shelf space.) Second, larger software boxes cost your company more to produce and ship. And third, many consumers and environmentalists are frustrated by "airware"—the wasteful encyclopedia-sized software boxes used to hold one or two small diskettes.

In this article, I discuss software packaging from an environmental perspective. We explore how other industries have dealt with similar packaging issues, and what the software industry can learn from these experiences. Then, we share some ideas on ways you can cut down on packaging waste, while also reducing the cost and environmental impact of your packaging.

The Role of Packaging

For the record, it's important to say that ideas on reducing packaging waste aren't worth the paper they're printed on if they prevent your package from meeting these objectives:

- protecting the product
- attracting and informing customers
- adapting to retail requirements

But frequently, software publishers select a package design based on what everyone else is doing, never questioning whether there's a better way to achieve the same goals.

Eva Anderson, the editor of *Communication Arts: ECO Newsletter* and a designer of ecologically friendly packaging, talks about what happens when you look at packaging with an eye to environmental impact: "The biggest surprise that clients have when I review their existing packaging is how much money can be saved. And with good design, environmentally friendly packaging doesn't have to impact sales."

Another reason to take a hard look at packaging waste is the recent rise in bulk paper costs. Not only have prices gone up every quarter for the last four quarters, but Mark Diverio, a paper industry analyst at UBS Securities in New York City, thinks that prices are unlikely to drop anytime in the next few years. He bases this prediction on high material costs, lower manufacturing capacity, and the efforts of paper mills to recover from a four-year downturn.

In Search of Standards

It doesn't take a Ph.D. in material science to figure out the environmental impact of setting a standard that reduces the size of software boxes. This move would also save software publishers material and shipping costs, and make retailers and customers happier.

But when Dan Van Hammond, K-Mart's merchandising director, proposed (on behalf of key mass merchants) a smaller software packaging standard at the 1994 Software Publishers Association Fall Conference, he was vehemently booed by software developers. (K-Mart's proposed 8.5 x 5.75 x 1-inch CD package would effectively double the number of products that K-Mart could display and allow the retailer to market everything face-out.)

K-Mart's seemingly helpful proposal was met with outrage, then largely ignored by software publishers, because of these issues:

- *Boxes are preview mechanisms.* Other than the recent development of CD-ROM samplers, the software industry has no product preview mechanism in place. And since so many consumers make buying decisions at points of purchase, publishers use package surface area to communicate "why buy" messages. Scaling down box size would reduce the "billboard" space for these product messages.
- *Box envy.* There's a fear that if your competitor puts a bigger box next to yours, the consumer will grab their box.
- *Mixed channel requirements.* Software stores, mass merchants, bookstores, and entertainment stores have different packaging form-factor needs. Bookstores want book-sized software boxes, music stores want music-CD–sized software packages, and so on.
- *The cost of retooling without a standard.* Developers and retailers are afraid to commit to a new box or display size without a standard in place. And because

the industry is in the middle of the transition from floppy-based software delivery to CD-ROM–based delivery, any standards decisions are all the more difficult.

In the absence of standards, many industry people feel that software packages will continue to grow in size. Jim Oppenheimer of AGI, a packaging manufacturer, says, "You wouldn't believe how many times I've seen software boxes arbitrarily increase in size, as the result of 10-second conversations around a conference table."

John Schikora of Ivy Hill Packaging comments: "I believe that if the software industry doesn't move soon, mass merchants will drive the issue of packaging size. The Wal-Mart and Sears class of distributors have already tried to set up standardization deadlines for the software industry, and eventually they'll put some teeth behind their threats."

The SPA's Packaging SIG

The Software Publishers Association (SPA) has taken the first step to addressing important packaging issues with the formation of a Software Packaging Special Interest Group (SIG). At the SPA's Boston Conference on September 29, 1995, the SIG will distribute its recommended packaging standards.

An SPA spokesperson, Mandy Braun Strum, talks about this effort: "The SPA recognizes the need for uniformity in packaging that doesn't stifle innovation, and we're taking proactive measures to ensure a smooth transition for the software industry to set packaging standards. The SPA strongly encourages an industry-wide move toward the adoption of the Packaging SIG's standards. These standards should facilitate consistent placement of important retailer and consumer information, while promoting creativity and concern for environmental issues."

Lessons Learned From Other Industries

If the software industry is in search of insights on establishing packaging standards, two good places to look are the cosmetic and music industries.

The cosmetic industry is an example of an industry that couldn't settle on its own packaging guidelines, so in the interest of consumers, the government stepped in. Packaging legislation was driven by consumers, who felt that many cosmetic packages—designed with big lids, hollow bottoms, and large air spaces—misrepresented the amount of product being delivered. Because the cosmetic manufacturers couldn't decide upon their own voluntary guidelines, the government established strict regulations that specified design parameters such as cap size versus bottle size, and air volume versus product volume.

Another situation closer to the software industry's plight occurred when the music industry began shipping music CDs. Initially, record publishers shipped CDs in the "CD-longbox" package, so that retailers could display CDs in racks used for traditional records. Many artists and publishers disliked this format, which was twice the size of a plastic jewel box, because of the unnecessary materials it consumed. High-profile recording artists, such as Sting, David Bowie, and Yanni, entered the fray for environmental reasons, using their clout to lobby for reduced package waste. On the other side of the battle were retailers, who were reluctant to spend money on new display systems.

This conflict was finally resolved because of the efforts of a few key industry people. Six of the top record publishers sat around a conference table with key distributors to work out an equitable solution—retailers agreed to a smaller jewel case packaging standard in exchange for rebates that would offset display refixturing costs. Record companies essentially funded these rebates through savings realized from reduced packaging costs.

What can the software industry learn from these examples? First, if developers don't take a leadership role in creating packaging standards soon, retailers or the government may force a less-than-optimal solution on everyone. Second, the music industry controversy proved that a few key individuals can make a difference in creating responsible packaging standards.

Individuals can also have a big impact on company-internal packaging decisions. Here are examples of two companies that have taken a leadership role in environmentally friendly packaging—The Voyager Company and Apple Computer, Inc.

Voyager Thinks Small

The Voyager Company was one of the first large software publishers to move to a smaller format box. The company also created an environmentally friendly box—an unbleached brown kraft box—for its not-for-profit *Amnesty International* CD-ROM title. And though Voyager's primary objective in using this box was to reduce cost, its distinctive appearance helps it stand out on retail shelves.

When asked whether the smaller box format has cost them sales, Bob Stein at Voyager replied, "We've probably lost some sales in traditional software stores, but we felt that moving to a smaller, book-sized box was fundamentally the right thing to do. It's an ecologically responsible size, and bookstores and customers love it."

Since Voyager began using this box, CNN, Scientific American, and Ion have also opted for this form factor. In addition, Compton's NewMedia and Interplay have standardized on smaller-than-average box sizes.

Apple's Environmentally Friendly Approach

In 1991, Apple Computer decided to integrate environmental factors into its packaging design criteria, and its first big step was to change hardware boxes from fully bleached kraft corrugated cardboard to uncolored, recycled brown corrugated cardboard. They also reduced overall material usage in each box, including polystyrene inserts that aren't biodegradable. These changes not only resulted in a more environmentally responsible use of materials, but they also reduced packaging costs and shipping blemishes.

"Today we specify a minimum of 10 percent post-consumer waste and 50 percent total recycled content for our paperboard packaging materials, and we try to find 100 percent recycled material if it's available," says Peggy Jensen, a packaging production manager at Apple. "Though our marketing people insist on using high-visibility white boxes for our software cartons, we selected an unbleached, recycled, clay-coated sheet that meets the marketing requirements. (Clay-coating is a natural coating that makes it easier to print on recycled paperboard). I think the biggest thing that Apple has learned over the last few years is that you can still design great packaging with environmentally friendly materials."

Ten Ways to Save Packaging Costs (and the Planet)

Here are ten more ideas on how your company can cut packaging waste and design environmentally friendly packaging, while at the same time saving your company money.

1. Reduce material use. By decreasing package size and eliminating unnecessary trays or inserts, you save on up-front material costs and shipping. From an environmental perspective, you use fewer natural resources to create a package and consume less landfill space when the package is thrown out. What's more, with a good design and the use of flaps and die-cut windows, smaller packages can be just as effective in attracting customers as larger ones.

2. Specify recycled and recyclable materials. The cost and quality of recycled materials has improved so dramatically over the last few years, that specifying them for your packaging and collateral has very little impact on packaging objectives. Besides that, manufacturing recycled paper requires 27 to 44 percent less energy than manufacturing virgin paper (source: *Communication Arts: ECO Newsletter*).

3. Use less ink coverage, and avoid toxic inks and bleaching. Eva Anderson, *ECO* editor, recommends that you use less ink coverage and avoid toxic inks (such as metallic inks) in your packaging designs. And, if possible, don't use bleached paper, since bleaching is one of most toxic processes in paper production. "This reduces environmental impact on both ends of the process. At the beginning, fewer toxins are released into the environment during ink manufacture and the printing process. On the back end, less sludge is created in recycling the paper components of your package."

4. Make it easy to separate recycled materials. The use of some nonrecyclable materials in software packaging is unavoidable, but try to design your package so that purchasers can easily separate recyclable parts for reuse. It's also helpful if you add box copy that reminds your customers to recycle your packaging.

5. Consider using a jewel box alternative. Many of the major packaging vendors now offer recyclable alternatives to the clear plastic jewel boxes used by many CD-ROM manufacturers. (If your product is used by young children, you may actually improve customer satisfaction by eliminating hard-to-open jewel cases.) Of course, one of the big selling points of jewel cases is that most CD mastering facilities own machines that automatically stuff and assemble these packages. But there are alternatives, such as Ivy Hill Packaging's ECO-PAK FLP (a paperboard cover/plastic tray combination) that can be machine assembled and loaded. (For an example of this type of package, take a look at any Apple Developer CD published since April 1993.)

6. Specify a dual-use package. Design a package that won't end up in a trash bin as soon as the customer unpacks your product. Design a package that can be used as a "keeper" for storing CD discs or manuals.

7. Ship mail orders without boxes. If you ship software directly to customers, mail these orders in a minimalist package. After all, these paid-in-advance customers don't need fancy retail boxes with persuasive copy.

8. Create electronic manuals. With the help of technologies such as Apple Guide and Adobe Acrobat, developers can, for the first time, create effective all-electronic user documentation. (For example, Electronic Arts used Adobe Acrobat to create all-electronic documentation for their Top Ten Mac Pak product.) This approach reduces material, printing, and shipping costs. It also has the potential to provide users with an intelligent help system more useful than hard-copy manuals.

9. Deliver software updates through electronic services. Though electronic updates won't truly be practical for a few more years, faster modems and technologies, such as OpenDoc and intelligent agents, will make this delivery vehicle an efficient material-free way of updating software. One company that's already distributing software updates electronically is Working Software in Santa Cruz, California (creators of SpellSwell, FindSwell and Last Resort utilities). If you're a registered Working Software user and you provide them with an e-mail address, they'll send you intermediate software updates for free.

10. Champion a smaller packaging standard. Perhaps the biggest environmental impact that the software industry can bring about today is to establish a smaller standard software package. The music industry has proven that just a few individuals can make a difference in breaking a standards gridlock. If you're interested in packaging issues, join the SPA's packaging standards meeting this fall (session information will be posted on the SPA Web site, www.spa.org), or organize a standards group within your own software market segment.

On Designing "Green" Packaging

"Green" packaging design isn't just good for the environment—it's good for the software industry. By periodically reviewing your packaging to eliminate waste, you can reduce the unit price of your products, thus improving the profitability of your company. Environmentally friendly packaging can improve customer satisfaction, and, in some cases, can also be used as a competitive advantage. (For example, The Body Shop, an all-natural cosmetics retailer, and Ben and Jerry's, the Vermont-based ice cream company, have successfully built memorable corporate identities around environmental friendliness.) And finally, by doing your part to create a smaller software package standard, you'll be helping to save another type of nonrenewable resource—retail shelf space. ♣

Kris Newby is a technical communications consultant and freelance writer based in Palo Alto, California. In one of her past lives, she managed packaging projects for Convergent Technologies, Inc., and Apple Computer, Inc.

APDA Ordering Information To place an APDA order from within the United States, 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 716-871-6555. You can also reach us by AppleLink at APDA or by e-mail at APDA@applelink.apple.com. More detailed APDA ordering information is available at the following locations:

- Internet: http://www.info.apple.com/dev/apda.html
- AppleLink: Developer Support:Developer Services:APDA
- eWorld: in the Developer Corner of the Computer Center