AppleDirections

CONTENTS

Next month:
A close look at the Game Sprockets software development kit.

APPLE NEWS

IBM Licenses the Mac OS

New Mac OS Sublicensees Also Announced

Taking another large step forward in its expanding Mac OS licensing program, Apple Computer, Inc., recently licensed the Mac OS to IBM. As a result of the agreement, Apple and IBM expect to work together to expand Power PC microprocessor and Mac OS market share far beyond what it is today by offering customers additional sources and greater choices for Mac OS–based systems.

According to the agreement, IBM will be able to sublicense the Mac OS with IBM Power PC microprocessors to any manufacturer of boards or systems, marking the first time a Mac OS licensee has been granted such far-reaching sublicensing ability. The previously announced licensing agreement between Motorola and Apple gives Motorola the ability to sign anyone it wishes to sublicense the Mac OS, but Motorola must supply the logic boards, or complete hardware systems, used by the sublicensees. Both IBM and Motorola are able to sublicense the Mac OS without prior consent from Apple, although Apple will certify all systems sold with the Mac OS to ensure compatibility.

The agreement puts IBM's marketing and service prowess behind the Mac OS platform; that, and the efforts of IBM sublicensees of the Mac OS, is expected to further expand market opportunities for your products. IBM has already signed sublicensing agreements for the Mac OS with Datatech Enterprises Co. and Tatung Co., both manufacturers of a range of computer equipment.

IBM will be able to distribute the Mac OS on systems employing the current Apple

please turn to page 2

STRATEGY MOSAIC

Why Mac OS 8 Is Important

By Gregg Williams, Apple Directions *staff*

Part 1: Backward Compatibility and the Mac OS 8 Architecture

Mac OS 8 (formerly known by the code name *Copland*) is a big step in the ongoing evolution of the Mac OS, even bigger than the transition from System 6 to System 7. With Mac OS 8, Apple Computer, Inc., is stepping away from an operating system designed in the mid-1980s and moving toward a later version that will serve the Mac OS platform well past the year 2000. Eventually, Mac OS 8 will replace System 7.5 as the operating system that ships with every Mac-compatible computer, so you and your products need to be ready for it.

Apple Directions has written articles about various Mac OS 8 technologies, but with the upcoming widespread release of the Mac OS 8 Developer Release: Compatibility Edition, it's time for Apple to step back from the individual technologies and give you a "big picture" look at Mac OS 8 itself and why it is important to both your and Apple's continued success. That's what this two-part article is about.

Mac OS 8 is important to you, the developer, for four reasons:

- It provides a high level of backward compatibility with System 7, ensuring that, wherever possible, your current System 7 applications will continue to run under Mac OS 8.
- Many of its technologies provide the user benefits of increased performance, increased stability, and greater ease of use. These features are important to you because they provide an environment in which your

please turn to page 6

AppleDirections

Volume 4, Number 6

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/Graphic Design
Lisa Ferdinandsen (LISAFERD)

Contributors
Peter Bickford, Alex Dosher, Tony Francis, Cathy Hams, Caroline Rose

Manager, Developer Press
Dennis Matthews

Manager, Apple Developer Periodicals
Mark Bloomquist

Production Manager
Diane Wilcox

Prep and Print
Consolidated Publications, Inc., Sunnyvale, CA

Too Much News

There's so much news this month—and so little space for it—that we've decided this month to lend most of the space usually devoted to the Editor's Note to the news. Even by doing that, and by using a new design that accomodates more words per page, we *still* can't fit all the news—this month or any month. We'll report the items here that we think are most important for developers, but if you want to be sure you're getting all the news from Apple, you need to subscribe to Apple Directions Express, our biweekly email newsletter that digests the latest news and points to locations on the Internet and elsewhere for more information. Subscribe by sending e-mail—with the string <subscribe your real name> in the subject field—to adirections@thing1.info.apple.com. And, please, fill in the survey on the facing page—we need your input. The Editor's Note will return next month—space permitting.

Paul Dreyfus
Editor

APPLE NEWS

IBM Licenses the Mac OS

continued from page 1

proprietary Power Macintosh hardware design as well as on Common Hardware Reference Platform–based systems. Common Hardware Reference Platform is the specification devised by Apple, IBM, and Motorola for PowerPC processor–based computers that can run a variety of operating systems, including Mac OS, Windows NT, AIX, and Solaris.

"IBM has long been a strategic Apple partner, and has the global, technical, and marketing strength to propel the Macintosh platform to a new, expanded role and influence," said Dr. Gilbert Amelio, chairman and chief executive officer of Apple Computer. "This much-anticipated agreement instantly opens up business opportunities for computer manufacturers worldwide."

"The Mac OS is renowned for its ease of use and consumer appeal," said Michael J Atardo, general manager, IBM Microelectronics. "With IBM's worldwide reach and our relationships with systems manufacturers, we're committed to proliferating the Mac OS and building upon the success of the PowerPC platform. Apple has been our valued partner in defining the PowerPC chip and [Common Hardware Reference Platform], and today we join forces to enable the industry for Mac OS."

Since committing to licensing the Mac OS in September 1994, Apple has implemented a two-phase licensing strategy. In the first phase, Apple focused its efforts on licensing the Mac OS with Apple's proprietary hardware designs to a relatively small number of licensees. With the Motorola and IBM licensing agreements, Apple has moved into the second, far more open phase of its licensing strategy, in which licensees will design their own hardware based on the Common Hardware Reference Platform specification, independently of Apple. This is expected to bring greater innovation to the PowerPC/Mac OS platform, providing customers additional sources of Mac OS–based computers as well as a broader range of feature-set, price, support, and distribution choices.

The licensing agreement includes Macintosh System 7.5.x, access to the next major

please turn to page 12

July *Apple Directions* Online

July's *Apple Directions* will be available by June 15 at the following locations:

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

Internet—http://dev.info.apple.com/appledirections/adtoc.html

AppleDirections Survey

About this survey...

Dear *Apple Directions* Reader:

We'd like to find out about how you use *Apple Directions*, the paper newsletter in which you found this survey as well as its three online components—the Web site, the Apple Directions Express list server, and the Adobe™ Acrobat™ version on the monthly Developer CD. Please take a few moments to answer the following questions; your answers will help us give you information in the ways that are most useful to you.

Thanks,

The Editors

Attention Readers Outside the United States:
We need your input, too! When you've completed this survey, please affix appropriate postage and mail it to us, or fax your reply to 408-974-9423. Many thanks for your extra effort!

Start Here

General Questions About the Four Versions of *Apple Directions*

1) How often do you use the different versions of *Apple Directions*?

	Monthly	Weekly	Daily
Apple Directions paper newsletter*	❑	❑	❑
Apple Directions Web site	❑	❑	❑
Apple Directions Express list server	❑	❑	❑
Apple Directions in Adobe Acrobat on the Developer CD	❑	❑	❑

*The issue in which you found this survey

2) Which version of the complete *Apple Directions* do you look at first? (Note: Apple Directions Express is not included, because it's a digest of the news.)

❑ *Apple Directions* paper newsletter
❑ *Apple Directions* Web site
❑ *Apple Directions* in Acrobat on the Developer CD

3) Please rank the four versions of *Apple Directions*, placing a "1" by the version you most prefer, a "2" by your second choice, and so on:

___ Paper newsletter *Apple Directions*
___ Web site *Apple Directions*
___ List server Apple Directions Express
___ Acrobat *Apple Directions* on the Developer CD

About the paper newsletter

4) How much time do you spend reading each issue of the *Apple Directions* paper newsletter?

___ 0–15 minutes
___ 15–30 minutes
___ 30–60 minutes
___ 60 minutes or more

5) How many people read the copy of the *Apple Directions* paper newsletter sent to your company?

___ 1 ___ 1–5
___ 6–10 ___ 10 or more

6) Were you previously aware that additional subscriptions to the *Apple Directions* paper newsletter can be ordered for $49/year from the Apple Developer Catalog?

___ Yes
___ No

7) With this issue, we've updated the appearance of the *Apple Directions* paper newsletter. Please rate how well you like the new design in comparison to the previous design:

___ Much better
___ Slightly better
___ About the same
___ Slightly worse
___ Much worse

Please say why:_____

_____

8) Please rate the usefulness of the *Apple Directions* paper newsletter:

___ I don't need it at all.
___ I find it interesting but not useful.
___ I find it useful.
___ I find it very useful.
___ It's a vital tool for me.

About the Web Version

9) If you do not use the *Apple Directions* Web site, why not? (Check all that apply.)

___ I wasn't aware of its existence.
___ I can't get on the Web.
___ I don't like your Web site. (Please say why.)_____

_____

___ I don't like to use the Web.
___ Other. (Please explain.)

_____

_____

Please see other side

10) Please complete the following: I would be more likely to use the Apple Directions Web site if _____

_____

_____

About the Adobe Acrobat Version on the Developer CD

11) If you do not use *Apple Directions* in Acrobat on the Developer CD, why not? (Check all that apply.)

___ I wasn't aware of its existence.
___ I don't have a CD drive.
___ I don't like Acrobat.
___ I don't like to use online viewing methods.
___ Other. (Please list.)

_____

12) Please complete the following: I would be more likely to use *Apple Directions* in Acrobat on the CD if: _____

_____

_____

About Apple Directions Express

13) If you don't subscribe to Apple Directions Express, why not?

___ I wasn't aware of its existence.
___ I don't have an e-mail address.
___ I don't like Apple Directions Express. (Please say why.)

_____

_____

___ I don't like to use list servers
___ Other. (Please list.) _____

_____

14) I would be more likely to subscribe to Apple Directions Express if _____

_____

_____

About You

15) Please indicate the number of people who work at your company:

___ 1–5
___ 6–25
___ 26–100
___ 101–1,000
___ More than 1,000

16) In what country is your company located?

_____

17) Which platforms do you support? (Check all that apply.)

___ Mac OS
___ DOS/Windows
___ OS/2
___ UNIX®
___ Other. (Please specify.) _____

18) Please indicate the position you hold in your company:

___ Executive
___ Manager
___ Nonmanagement contributor
___ Sole propietor

19) In which area is your job focused?

___ Technical
___ Marketing
___ Both technical and marketing

Thank You!

Now that you're done, please fold this survey in thirds along the dotted lines, tape it closed, and drop it in the mail; if you're in the United States, postage is paid.

If you're not in the United States, either fax us the survey (408-974-9423) or affix appropriate postage. We thank you for your extra effort.

FOLD FOLD IN THIRDS AND TAPE SHUT FOLD

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS MAIL PERMIT NO 453 CUPERTINO, CA
POSTAGE WILL BE PAID BY ADDRESSEE

Apple Directions Survey
Apple Computer, Inc.
1 Infinite Loop, M/S 60-DP
Cupertino, CA 95014-9968

INDUSTRYWATCH

Putting SPA 1996 Software Sales Data Into Perspective

By now, you're probably familiar with Software Publishers Association (SPA) data indicating that revenues from Mac OS software decreased in North America last year. What you might not have read yet is that revenues from software for Wintel systems—that is, software that runs under DOS and Windows—also began to decline at the end of 1995, just after the release of Windows 95. Here's the SPA data, released in late March.

For the year, North American software revenues increased 12 percent in 1995 over sales the year before, climbing to $7.5 billion last year from $6.7 billion in 1994. Macintosh software sales decreased just under 14 percent in 1995, falling to $1.05 billion from $1.2 billion. Combined DOS/Windows software revenues increased 14 percent over the year before—to $6.4 billion from the 1994 total of $5.6 billion. (Revenues for Windows-based software increased 27 percent, while DOS software revenues fell nearly 37 percent.)

In the fourth quarter of 1995, or Q4 '95 (October through December), the Macintosh software picture started looking marginally better, while Wintel software revenues decreased sharply. Comparing revenues from Q4 '95 with those from Q4 '94, Macintosh software revenues fell 12.8 percent, while combined DOS/Windows revenues—including those for Windows 95 software—fell 8. 9 percent. Overall, the industry recorded an 8.8 percent downturn in revenues. (See the chart on this page for more details.)

Let's look only at the six largest-grossing categories tracked by SPA, which together account for more than 50 percent of sales, according to SPA data; the categories are entertainment, home education, word processing, spreadsheet, finance, and "other productivity." (The last of these actually shows the highest level of revenues.) In these categories combined, Macintosh products are performing better than their Windows counterparts: Q4 '95 Macintosh software revenues from products in the six largest categories combined decreased by 5.6 percent, while DOS/Windows sales fell more than twice that much—14.7 percent—from Q4 '94.

Implications/Opinions: It's hardly cause for jubilation to discover that it's not just Mac OS sales, but the entire industry, that took a turn for the worse last year. Digging still deeper into the SPA data, it appears there's yet a graver challenge facing all who depend on a robust, growing software market: Unit sales are increasing faster than revenues, meaning that software companies may be selling more products, but making less money.

Here's what Apple Developer Relations Vice President Heidi Roizen recently wrote about the situation in a letter to analysts and members of the press: "According to SPA estimates, Q4 ushered in a particularly disturbing trend: In North America, software for every platform suffered negative revenue growth, while simultaneously, unit sales soared 35 percent. This means the real problem is margin erosion—not only for Apple developers, but for the entire independent software vendor (ISV) community. And this occurred during Windows' most significant operating system upgrade cycle this century!"

SPA also tracks software sales outside North America, and their data suggests that a similar phenomenon took place last year in other geographies, as well. The following comes from SPA data for 1995:

- In Asian/Pacific markets (including Japan), unit software sales increased 179 percent, but revenues were up only 57 percent.
- European unit sales increased 41 percent while revenues were up 19 percent.
- In Latin America, unit sales climbed 52 percent although revenues increased only 11 percent.

There may be ways that software firms are maintaining their margins. For example, with the increase in units shipped, some of you may be enjoying the economies of scale associated with higher volume manufacturing. And, let's hope that this is requiring you to spend less money on a per-unit basis to publish your products. However, the differences between unit and revenue increases are so extreme, especially in North America, that it would take a huge reduction in manufacturing costs per unit to eliminate the apparent margin erosion.

There may be a silver lining here, at least for Mac OS developers: According to the same SPA data, and other research from PCData,

SPA North American Software Revenue, Q4 '94 to Q4 '95

(All figures in millions of dollars)

	Q4* '94 revenues	Q4 '95 revenues	Percent change
Mac OS software	$361.3	$306.4	–12.8 %
DOS software	$329.1	$212.7	–35.3 %
Windows-based software**	$1,578.8	$1,524.8	–3.4 %
DOS/Windows total	$1,907.9	$1,737.5	–8.9 %
Industry total	$2,265.5	$2,066.1	–8.8 %

*October to December
**Includes Windows 95

average revenues per unit remain higher overall for Mac OS software than for Windows applications, making the Mac OS market potentially more profitable than the Wintel market. Again, from Heidi Roizen: "The Macintosh software 'premium' holds true for the U.S. as well as international markets, and is reflected across all but three of the product categories tracked by the SPA. Furthermore, average wholesale revenues per unit are not falling as rapidly for Macintosh software as they are for Windows titles."

A further point to keep in mind about SPA data is that it doesn't encompass the entire industry. SPA collects data only from its members, and not every software company belongs to the association. Many smaller firms prefer not to pay SPA membership fees, which means that SPA data may well be biased toward the larger software publishers. If you want to help change this, you can join SPA; a registration form is available at the SPA Web site (**http://www.spa.org**), or you can call SPA at 202-452-1600 in Washington, D.C. or at 33-(1) 45-63-02-02 in Paris.

All this is meant to help put the SPA data about 1995 Mac OS software sales in an appropriate framework to help you evaluate it, but, as we said, it's certainly nothing to celebrate. If the entire industry is facing tough times, we have to work together—even with developers working on the Wintel and other platforms—to increase margins, whether by expanding the market, lowering costs, or some other step.

Apple is working hard to better the situation, especially for Apple platform developers. Apple Developer Relations (ADR) is currently being realigned to better meet the needs of the development community; among its primary goals are helping you reduce development costs and the time it takes for your products to reach the market, broadening sales and distribution channels for Mac OS software, and increasing customer awareness of Mac OS solutions. For more about ADR, see the news story on page 12. ♣

STRATEGY MOSAIC

Why Mac OS 8 Is Important

continued from page 1

application can perform better and give your customers a higher-quality experience.

• The Mac OS 8 programming model has been streamlined and enhanced to simplify the job of writing software for the Mac OS platform. In many cases, where System 7 provides several APIs (application programming interfaces) for getting a job done, Mac OS 8 standardizes—and supports—one.

• Other Mac OS 8 technologies enable you to add new features to your software—features that would be impossible or inconvenient for you to add under System 7. This will make it possible for you to create the next generation of leading-edge software that will be highly competitive in the market ahead.

This month, I'll cover the first two points, saving the last two points for next month's Strategy Mosaic column. In this month's column, I'll also be talking about how the Mac OS 8 program architecture—that is, the way the various parts of programs interact with the operating system itself—differs from that of System 7. (I need to do that before I can talk about what parts of Mac OS 8 deliver its main customer benefits.)

By explaining the differences between System 7 and Mac OS 8, I hope I can give you a clearer picture of how Mac OS 8 will change software development for you and how it will give you opportunities to create *better* software.

Compatibility With System 7

Before I can talk about the many reasons why Mac OS 8 is important to you, I need to state one important underlying fact: *Mac OS 8 is designed to run the vast majority of System 7 applications.*

Apple has learned from experience that backward compatibility is perhaps the most important factor driving customers' acceptance of new hardware or software. *Apple is committed to doing everything it can to preserve customers' current software and hardware investments and to make it as easy as possible for customers to make the transition to Mac OS 8.*

However, Apple cannot ensure backward compatibility by itself; the company needs your active participation. Apple will be making the Mac OS 8 Developer Release: Compatibility Edition widely available to developers in mid-1996. This will mark the beginning of the period in which Apple and developers must work together to ensure maximum backward compatibility for System 7 applications. Apple will do everything it can to work with you to achieve maximum compatibility, but your active participation is necessary to ensure the success of Mac OS 8.

Backward Compatibility

To save time, I'll omit the details and make a blanket statement: If a technology is needed to ensure that System 7 applications continue to run, Mac OS 8 includes it. So, for example, even though Mac OS 8 includes a simpler and more powerful file system, Mac OS 8 maintains the API of the System 7 File Manager. Why? Because if Mac OS 8 didn't include the older API, System 7 software wouldn't work! Mac OS 8 includes many other System 7 managers—the Event, Printing, and Memory Managers, for example—for this same reason.

However, please note that most of these backward-compatible technologies are not guaranteed to be in future releases of the Mac OS. (By "future releases of the Mac OS," I mean major revisions to the Mac OS past Mac OS 8.) Apple engineers have included them to ensure short- to medium-term compatibility with System 7 software, but Apple encourages you to use the more powerful Mac OS 8 equivalents, which Apple has designed to be the foundation for future releases of the Mac OS.

One limitation of Apple's intent to provide backward compatibility is that Apple is not committed to supporting code that goes "underneath" the official API of a supported technology—to do so would prevent Apple engineers from making significant innovations in Mac OS 8. So if your code manipulates a technology's private data structures or global variables, it probably won't work under Mac OS 8.

Backward compatibility is, however, a double-edged sword. If Apple supplies complete backward compatibility, Apple supplies how much it can innovate, it can't fix some things that really need fixing. If the level of backward compatibility provided is too low, too many existing applications don't work, and customers are reluctant to move to the new version of the operating system.

The cost of some innovations is just too high. Apple engineers certainly would have liked, for example, to deliver a new operating system that provides a separate, protected memory space for each application, but for technical reasons, they couldn't deliver that *and* keep backward compatibility with the vast majority of existing software.

Because of this, Apple decided to innovate in steps—which allows Apple to evolve the Mac OS and still keep both itself and you financially healthy. Mac OS 8 is the first such step, and it is an exciting one that will provide many benefits to both you and your customers. *In the next major evolutionary step of the Mac OS, an application written to the Mac OS 8 programming model will—without modification—receive the full benefits of preemptive multitasking, while running in its own separate address space.*

Some Exceptions

Apple engineers spent a lot of time deciding what parts of System 7 *not* to support. Here are the major things that Mac OS 8 doesn't support, and why:

• *Device drivers.* Under System 7, device drivers are too hard to write, and they sometimes fall prey to stability problems. The Mac OS 8 driver architecture is more powerful and simpler to implement; in addition, Mac OS 8 drivers are designed to run faster and be more stable.

For those of you who write device drivers, it is very important that you work with Apple to create Mac OS 8 versions by the time Mac OS 8 ships. Without new drivers, your driver-based hardware products will suddenly cease to work when your customer installs Mac OS 8—which is something neither you nor Apple want to see happen. Apple *is* tracking the status of over 1,500 hardware products to ensure that customers get the drivers they need simultaneous with the release of Mac OS 8.

• *INIT-based extensions.* A good number of Mac OS products use INIT-based extensions, and you will have to redesign them to work under System 7. (To give you sufficient time to make the transition, Apple told you about this change in 1995, and some of you have already changed your products. Good work!) Many INIT-based extensions make global patches during system startup; Mac OS 8 has no global 680x0-style trap dispatch table, so you will have to redesign extensions that depend on this kind of patching.

Apple engineers did not make the decision to discontinue support for INIT-based extensions lightly, but they felt that Apple would be compromising Mac OS 8 severely if they allowed them to continue working. INIT-based extensions are a major source of system instability, and they complicate the programming model. Mac OS 8 includes new methods of extending the system that are stabler and easier to extend and support in the future.

Performance, stability, and ease of use are important to your customers, and you can deliver these benefits if you make sure your software runs well under Mac OS 8.

• *Desk accessories.* They're dead, D-E-A-D. They won't work under Mac OS 8. Why? Because there's no need for them anymore, and Apple wants to move to a simpler programming model. Desk accessories were designed to provide application-like functionality in a Macintosh computer that had only 128K of memory, but you had to work around various limitations to create one.

Today, Mac-compatible computers capable of running Mac OS 8 have at least 8 MB of memory, and Mac OS 8 virtual memory ensures that simple applications use only a small amount of memory. Apple talked about eliminating desk accessories under System 7.0 and didn't—but now it's time to do so. (However, if you want to create a "desk-accessory runner" utility that will allow customers to run them under Mac OS 8, there'll probably be a market for it.)

Mac OS 8 User Benefits and You

To users, Mac OS 8 delivers numerous benefits, all of which belong to one of three broad categories:

• *Increased performance.* Users will find that their Mac OS 8–based computers will be faster and more responsive. Existing applications will, in general, show modest performance improvements, but applications that take advantage of new Mac OS 8 features will show the greatest performance increases.

This increase in performance comes from two separate sources. The first is the fact that Mac OS 8 is written almost entirely in PowerPC processor code (unlike Power Macintosh computers today, where the operating system—System 7—contains a significant amount of 680x0 code, which runs slower under emulation).

Of course, the final portions of the Mac OS 8 that are now written in native code run faster. In addition, Mac OS 8 also gains speed because the slowdown from "context switching" (which happens anytime PowerPC processor code calls 680x0 code, or vice versa) occurs much less frequently.

The second source of increased performance comes from the fact that the new technologies in Mac OS 8 include improved algorithms that run more efficiently. To get the increased performance, however, applications must make use of these new technologies.

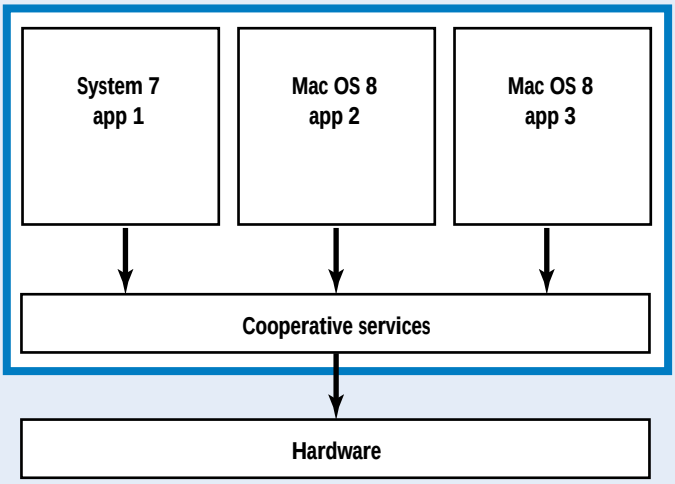
• *Increased stability.* Computers running Mac OS 8 will be significantly more stable than today's System 7–based computers.

Mac OS 8 achieves much greater stability by being the first version of the Mac OS to make use of the processor's supervisor and user modes. (These are modes of instruction execution for the PowerPC processor.) Under System 7, applications and the operating system run in the same mode (supervisor mode), thus making it possible for any code to change any memory location. (Because of this, an application can inadvertently change data belonging to the operating system, possibly causing a system-wide malfunction.)

Under Mac OS 8, critical parts of the operating system—including the microkernel, I/O drivers, the networking subsystem, and important parts of the file system—execute in supervisor mode, while all non–operating-system software executes in user mode. Since user-level code is prevented from changing data belonging to code executing in supervisor mode in Mac OS 8, applications and other user software cannot corrupt the data associated with critical operating-system components. For example, if an application running under Mac OS 8 crashes, it cannot harm the file system or its data.

Other parts of Mac OS 8 that execute in user mode—for example, the Mac OS 8 open font architecture—have their data protected

System 7 Program Architecture



Under System 7, all applications and the operating system itself run in the same address space, thus increasing the likelihood that one malfunctioning application can cause another application or the entire computer to malfunction.

from corruption by other programs. Mac OS 8 does this by implementing these parts as *server programs,* a type of program that I'll discuss later in more detail.

Finally, through a mechanism called *guard pages* (also discussed later), Mac OS 8 provides some greater stability for existing applications by preventing them from damaging other applications' data if they attempt to write outside the boundaries of their own stacks and heaps.

• *Ease of use.* Mac OS 8 includes numerous features that make it easier to use. Some features come "for free"—for example, every application running under Mac OS 8 automatically adopts the improved browser portion of the Mac OS 8 open/save dialog box. (However, if you modify your application to use routines from the new Mac OS 8 Navigation Services API, your application will get the full benefit of the new open/save dialog box.)

Other ease-of-use features, such as the Mac OS 8 Tips window (which offers users tips on how to use their computer more efficiently), require that you modify your application to take advantage of a given feature. Still others—for example, the find-by-content feature of the Mac OS 8 Finder—are connected to operating-system features, not individual applications. (For more details, see "Looking Forward to the Copland User Experience," in the April 1996 issue of *Apple Directions.* You can also find this article at **http://dev.info.apple.com/appledirections/apr96/stratmos.html** on the World Wide Web.)

Many Mac OS 8 features deliver improvements that are important to the user, but what does that mean to you, the developer? These features, even when they do not affect you directly, are important to you because they allow you to create software that delivers these user benefits to your customers. In other words, *performance, stability, and ease of use are important to your customers, and you can deliver these benefits if you make sure your software runs well under Mac OS 8.* In addition, you can deliver even larger measures of these benefits if you write additional code that taps into Mac OS 8 features.

With that introduction in mind, the next step is to get an overview of how Mac OS 8 program architecture differs from that of System 7. Then we can take a look at several Mac OS 8 technologies and how they benefit users. (One note is in order here: My explanation of Mac OS 8 multitasking shares some material with Tony Francis's two-part article on Mac OS 8 multitasking in last month's and this month's issue. The material in my article is at a slightly higher level and offers a different perspective of Mac OS 8 multitasking. I think you will find it useful.)

The Program Architectures of System 7 and Mac OS 8

As shown in the figure on this page, all System 7 applications and the cooperative services they use run in the same address space. (*Cooperative services* is the Mac OS 8 term for code that has been written in such a way that applications must cooperate with each other to synchronize their access to these services. In System 7, all system code is cooperative. In Mac OS 8, cooperative services include portions of the Human Interface Toolbox and all the code required for backward compatibility with System 7.)

The stability of System 7 can be threatened by any code inside the single address space. Since both applications and system software run in the same mode (supervisor mode), *any* code that malfunctions has the potential to corrupt any code or data in the computer. (If System 7 virtual memory is turned on on a Power Macintosh computer, however, all code stored in memory is read-only.)

The largest improvement that Mac OS 8 makes to overall system stability is the use of supervisor mode to protect critical data. Because applications and other non–operating-system code run in user mode, such code cannot change any of the critical operating-system data it accesses.

Mac OS 8 adds two new kinds of code. (See the figure on page 9.) The first is the *server program,* which is code that performs its work offscreen, generally to provide a service to one or more other programs. A server program cannot have any code that users interact with or that uses the Mac OS 8 cooperative services; instead, it usually contains calculation- or I/O-intensive code.

Server programs represent another source of improved stability. *In Mac OS 8, all code in memory has read-only status, so corruption of code is impossible.* However, the code in a given address space can (in general) change data in the same address space. Code in other address spaces cannot change the data associated with a given server program; this contributes to overall system stability. Mac OS 8 implements some of its services as server programs, and you can create your own server programs as well.

The second new kind of code that Mac OS 8 adds is the *reentrant service,* which can be used simultaneously by multiple pieces of code. When an application uses a reentrant service, its typically gets its own copy of the static data associated with the reentrant service, which means that pieces of code can call the same reentrant service and still execute correctly. Reentrant code is extremely important to Mac OS 8 because, without it, the operating system could not do preemptive multitasking.

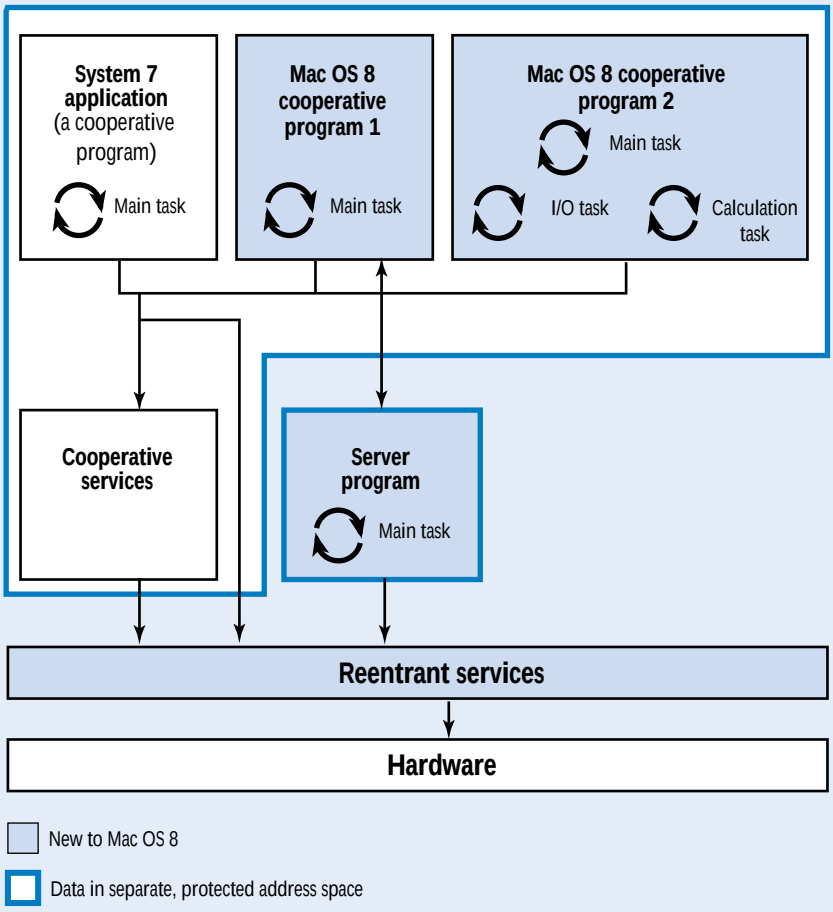
Mac OS 8 also adds a new term, *cooperative program*, for a kind of code that exists in both System 7 and Mac OS 8. A cooperative program typically handles the human interface and interaction with the user, as well as coordinating its work with other cooperative programs to access cooperative services. By definition, every System 7 application is a cooperative program. As you will see, at least part of every Mac OS 8 interactive application (that is, an application that has a human interface and interacts with the user) *must* be a cooperative program.

Note the following things about the Mac OS 8 program architecture:

- Cooperative programs can use cooperative services (but only the main task of a cooperative program—which might be the only task the program has—that can use cooperative services).
- Cooperative programs and server programs can communicate with each other.
- All programs have at least one task, called its *main task.* Any program can have additional tasks associated with it, and these tasks can communicate with each other.
- A cooperative program can use multiple server programs. In addition, a server program can serve multiple clients and can continue to run even when no cooperative program is currently communicating with it.
- Cooperative programs, cooperative services, and server programs can all use reentrant services.
- Server programs can call reentrant services but not cooperative services.

Mac OS 8 Program Architecture

Cooperative program address space



This figure has been simplified to show one occurrence of each of the three types of applications that can run under Mac OS 8. In actuality, multiple occurrences of each type of application can be running at the same time.

Multitasking in System 7 and Mac OS 8

Under System 7 cooperative multitasking, individual applications continue executing until they "decide" to release control, thus allowing the background process of another application to begin executing. Even though this results in a usable form of multitasking, the operating system itself does not control the processor's scheduling. Even under the best of circumstances, an individual System 7 application (which has no way of knowing what other applications are running or whether they have a greater "need" to execute) makes inefficient use of the processor, which often results in the processor idling when it could be used for productive work.

Mac OS 8 differs from System 7 in that the Mac OS 8 operating system itself retains control of which body of code (or, to be more precise, which *task*) executes, and for how long.

Here is a very important fact about Mac OS 8: *Mac OS 8 completely makes the transition to preemptive multitasking—there is no cooperative multitasking.* However, the Mac OS 8 Process Manager includes a cooperative scheduling algorithm that simulates the behavior of System 7 cooperative multitasking. (The next section explains this process in more detail.) The Process Manager's cooperative scheduling algorithm affects applications only—that is, a cooperative program that doesn't relinquish control as it should will not halt the execution of tasks outside the cooperative environment—for example, the microkernel and server programs.

Mac OS 8 divides the processor's time between the various tasks, with eligible tasks at the higher priority levels running first. When a task is no longer eligible for execution, then the task at the next highest level runs. (Apple will supply a list of priority levels and what kinds of code should run at that level. When you create a task, you should assign it an appropriate priority level.)

Because of their need of uninterrupted access to the processor, the highest-priority tasks—for example, real-time and operating-system tasks—run until they are blocked, even if other tasks at the same level are waiting. For all the other priority levels, though, the following happens: When multiple tasks at the same priority level are eligible to run, the microker-

nel gives each of the tasks a "slice" of the processor's time, then moves on to the next eligible task with the same priority. If a task of higher priority becomes eligible, the microkernel suspends the lower-priority task and begins executing the higher-priority task.

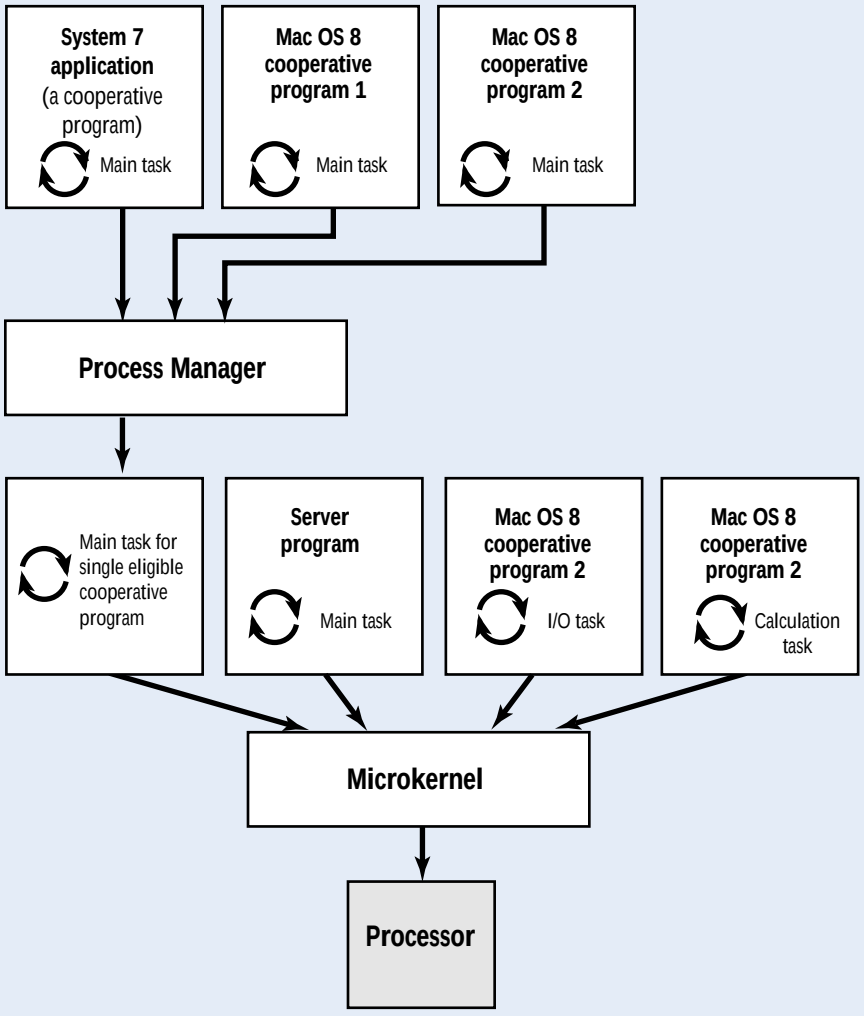
Inside Mac OS 8 Multitasking

The figure on this page shows how the operating system selects which task the processor will execute next. Since the System 7 application and the two Mac OS 8 cooperative programs all run in the cooperative environment, only one of them gets a "slot" in the pool of tasks eligible for execution. (As I mentioned earlier, the Process Manager works with the cooperative programs to handle this switching.)

In addition to the chosen main task from one cooperative program, all other tasks join this task to form a pool of tasks that can participate in preemptive multitasking. (Some tasks may be temporarily blocked; these will not be preemptively scheduled until they become unblocked.)

Note that all the tasks *except the main tasks of the cooperative programs* automatically go into the pool of tasks that can participate in preemptive multitasking. In this figure, one of the tasks in the pool is the main task of a server program and two are secondary tasks (I/O and calculation tasks) of a cooperative program in the cooperative environment. As I stated earlier, the microkernel coordinates the running of tasks based on their respective priority levels.

Mac OS 8 Preemptive Multitasking



This figure shows how cooperative multitasking occurs for the main tasks of the three cooperative applications shown in the figure on page 9. The Process Manager determines which of the main tasks (at top) becomes eligible for preemptively scheduled execution. From the pool of tasks eligible for preemptive scheduling (middle), the microkernel decides, based on the tasks' priority levels, which task the processor will execute next.

Multitasking Alternatives in Mac OS 8

In Mac OS 8, there are two ways you can add preemptively scheduled tasks to your application:

- The first way, which is the easier to implement, is to have your Mac OS 8 application (which is a cooperative program running in the cooperative environment) run a separate task that does not have any user interface or call any of the Mac OS 8 cooperative services. These additional tasks are preemptively scheduled and can run at a set priority level.

By assigning tasks the appropriate priority level, you can make your application run faster and appear more responsive to users. For example, you should give higher-than-average priority to a task calculating a moving 3D object for display in real time. You should give lower-than-average priority to a task that collects small amounts of data from a low-priority laboratory instrument.

- The second way to add preemptively scheduled tasks to your Mac OS 8 application is to create a server program, which gains both preemptive scheduling and memory protection for its data. Because of the increased complexity of coordinating communication between the cooperative and server programs, you are more likely to use this method only when you are adapting an existing client/server solution to Mac OS 8 or when you need to draw upon the other advantages of server programs. These advantages include memory protection for the program's data, the ability for the server program to be running whenever the computer is on, and the ability for the server program to be shared by other tasks. Remember, however, that server programs cannot have any user interface or call any of the Mac OS 8 cooperative services.

Preemptively scheduled tasks offer one way to increase your application's performance. Another, different way of doing so has been brought forward from System 7.5, which used the Thread Manager to create multiple paths of execution within a System 7.5 application. Mac OS 8 supports this same mechanism using the Cooperative Thread Manager and refers to these multiple paths of execution (which can be a part of any task) as *cooperatively scheduled threads.* The Cooperative Thread Manager allows System 7 applications that use the System 7.5 Thread Manager to continue to work under Mac OS 8.

Memory Protection

Mac OS 8 offers five kinds of memory protection, all of which contribute to the increased system stability of Mac OS 8.

First, Mac OS 8 offers protection for all code. Code is always mapped into read-only memory areas, so there is no possibility that it can be corrupted by any malfunctioning code elsewhere in the system.

Second, the data associated with code executing in supervisor mode—including the microkernel, I/O drivers, the networking subsystem, and important parts of the file system—is read-only to cooperative programs and other user-level software. This greatly decreases the ability of applications to cause a system-wide crash.

Third, the separate address spaces for data in Mac OS 8 also offers protection. As I pointed out earlier, if code in a given address space malfunctions, it cannot corrupt the data in a different address space. This means that, for example, if a server program crashes, it will not crash the Mac OS 8 file system.

Fourth, by assigning access permission levels, where appropriate, to memory areas that your application uses, you can reduce the possibility that other software will be able to corrupt the data that belongs to your application. This feature is most useful in enhancing the stability of System 7 applications and the cooperative-program part of Mac OS 8 applications.

Under Mac OS 8, you can assign a *memory area* (which is loosely defined as a contiguous range of memory locations within an address space) one of three levels of access permission. One level, read/write, allows other software in the same address space to read from or write to memory locations. The second, read-only, allows other software to read from but not write to such memory locations. The third, the "excluded" level of access permission, forbids other software to either read from or write to a memory location.

Fifth, another kind of memory protection, *guard pages,* enhances system stability by limiting the amount of damage that software can do if it attempts to read or write outside the memory area it's entitled to access. When software creates a memory area, it can specify that extra memory at both ends of the memory area are to be designated as guard pages. If any code—from either the user or supervisor mode—attempts to read from or write to guard pages, the processor generates an exception.

Under Mac OS 8, launching an application automatically causes its application stack and heap to be surrounded by guard pages. This means that, for example, if an application does something that causes its stack to overflow, the attempted access to a guard page causes an exception and results in the safe termination of the application. This is an improvement over what often happens in this situation under System 7—namely, that the application overwrites memory belonging to another application, resulting in one or more applications crashing.

Virtual Memory

You probably know what virtual memory is—it's a way of making the computer seem to have more physical memory than it actually has. Code that needs to be executed is loaded into physical memory and stored elsewhere—usually on a hard disk—when it is no longer needed. If an application needs to execute code that's not in physical memory, the operating system—automatically, and invisibly to the application—loads the required code into physical memory and ensures that it executes correctly.

In System 7, virtual memory is optional; when it's on, it relies on a user-determined amount of disk space. In contrast, Mac OS 8 virtual memory is always on, and it allocates its disk space dynamically; it is also a more sophisticated, more efficient virtual memory implementation than is used by System 7.

Virtual memory is another one of the automatic benefits of having your applications run under Mac OS 8. Whether your application takes specific advantage of Mac OS 8 features or not, virtual memory works better than it does under System 7 and allows users to make the best possible use of computer memory and run more software simultaneously than would otherwise be possible.

Mac OS 8 File Manager

The Mac OS 8 File Manager delivers a number of file-management improvements:

- *Increased performance.* The Mac OS 8 File Manager runs faster because it is written in PowerPC–processor code.
- *Support for larger volumes and files.* We live in a world where the maximum volume size for mass-storage devices under System 7, 4 gigabytes, is no longer enough. Mac OS 8 improves the HFS (Hierarchical File System) volume format by supporting volumes of up to 2 terabytes.
- *Support for multiple volume formats.* The System 7 File Manager implementation is tied to the HFS volume format, which makes it more difficult for the operating system to support other volume formats. Mac OS 8 corrects this with an extensible architecture that allows support for arbitrary volume formats to be added. Initially, the Mac OS 8 File Manager will support the HFS, DOS, AppleTalk Filing Protocol (for access to AppleShare and Personal FileShare volumes), and CD-ROM (High Sierra, ISO 9960, audio CD, and Photo CD) formats, with support for new volume formats as they become available. Because of this new architecture, an application that uses the Mac OS 8 File Manager will be able to access whatever mass-storage volumes are connected to the computer—even volume formats that haven't been invented yet—without your having to rewrite the application.
- *Improved Standard File dialog boxes.* All applications will automatically display an enhanced browser in their open/save dialog boxes. However, applications that use the new Mac OS 8 Navigation Services API will have open/save dialog boxes that offers users more information about files and offers better ways to find a desired file.

Improved User Experience

Mac OS 8 does many things to improve the user's experience, many of which are centered around what the desktop looks like and how it behaves. (For a more detailed description of the user-experience elements of Mac OS 8, see "Looking Forward to the Copland User Experience," in the April 1996 issue of *Apple Directions.*) These features will give users an immediate benefit from using Mac OS 8, and existing applications that "play by the rules" with the human interface will automatically inherit as much of the new appearance of the Mac OS 8 desktop as possible.

In addition, Mac OS 8 includes some user-assistance technologies that you can add to your application to make it more useful to your customers. These include help (Balloon Help, Apple Guide, and tips), delegation (triggers, notifiers, and scheduled tasks), and "experts." Again, see the *Apple Directions* article mentioned earlier for details.

Integration of Key Technologies

As the Mac OS evolved, Apple innovated, and added those technologies to the Mac OS through system extensions. Extensions represent a potential weak link in the stability of the operating system, so one of the ways Apple is increasing system stability in the Mac OS 8 is to integrate key Apple technologies into the operating system itself. Mac OS 8 does this by integrating selected key technologies—including OpenDoc, QuickDraw GX, WorldScript, and Open Transport—into itself.

Next Month

The Mac OS 8 story is too big to tell in the space available for this month's column. Next month, I'll go into two major areas that continue the list of Mac OS 8 advantages: I'll explain how Mac OS 8 simplifies the programming model you have to live with every day, and I'll describe the major Mac OS 8 technologies that allow you to do things that are difficult or impossible under System 7. ♣

Author's note: I'd like to express my sincere thanks to Tony Francis, author of the upcoming book Mac OS 8 Revealed *from Addison-Wesley Longman. Despite his own deadlines, Tony freely contributed his time to reviewing this article. My understanding of Mac OS 8 comes almost entirely from reading draft chapters of his book.*

APPLE NEWS

IBM Licenses the Mac OS

continued from page 2

release of the Mac OS, and access to the 16 local language versions of the Mac OS that Apple has so far approved for licensing (U.S. English, U.K. English, Italian, German, French, Spanish, Portuguese, Swedish, Norwegian, Danish, Finnish, Dutch, Traditional Chinese, Simplified Chinese and Japanese.)

New Apple Developer Relations Charter, Organization

Apple Developer Relations (ADR) recently reorganized and expanded its operations to better provide the comprehensive range of technical and business resources required by developers. The newly defined organization, which will now report to Apple Chief Administrative Officer George Scalise instead of to Research and Development, includes not only the traditional evangelism and technical support functions, but also adds a new group to provide business support.

Under the leadership of Vice President Heidi Roizen, ADR will consolidate developer activities from all over Apple into one organization to coherently and effectively meet the needs of the Apple development community. To do so, ADR will be organized into the following five primary units:

- Evangelism will conduct day-to-day management of developer accounts and lead efforts to evangelize Apple technologies and products among you and your developer colleagues.
- Developer Marketing will define and promote Apple's developer proposition. In this role, it will initiate comarketing and other programs to broaden the "shelf-space" allocated to solutions for Apple platforms, thereby helping developers reach a larger base of customers. Developer Marketing will also administer Apple's member-based developer programs, including the Partners, Associates Plus, and Associates programs for both Newton and Macintosh developers, as well as the Apple Developer Catalog and Apple's developer communications and periodicals, including *Apple Directions.*
- Developer Technology Services will be responsible for meeting developers' technical needs with training, technical services, and specialized support programs. Included in this group will be Developer Technical Support (DTS), the groups that provide hardware and software documentation, Developer University and other technical training services, compatibility/testing labs, developer bug reporting, resolution and management, and specialized engineering support.
- The new Developer Business Development group will chiefly identify strategic partnership opportunities in the developer community. It will help strategic developers work with Apple, driving the process for licensing technology to and from Apple. This group will also be responsible for recognizing and communicating the business implications of new Apple technologies, as well as for developing relevant business data to share with the Apple development community.
- International Developer Relations will adapt and deliver ADR programs and opportunities for developers worldwide to assure that Apple provides consistent levels of support for developers outside the United States.

"Developers are the lifeblood of the Macintosh," said Dr. Gilbert Amelio, Apple's chairman and chief executive officer. "Apple must deeply integrate the development community into its global mission, and this new reporting structure is a logical step in that direction. We want to foster an environment where Apple Developer Relations can more efficiently and effectively work across all functional areas, for the mutual benefit of Apple and the vendors who support our products."

The group's expanded responsibilities are a direct result of developer requests to provide a fuller complement of marketing, business development, and relationship management support. "In recent months, we've seen a visible improvement in the way Apple regards the developer community," said Duane Schulz, president of Now Software. "Apple has actively sought our input, and these organizational changes show the company is serious about integrating developer issues into its business strategy."

Developers in key growth segments have endorsed the new structure. "We are very pleased to see Apple increase its focus on developers' business and comarketing needs on a worldwide basis," said Alan Lefkof, president and CEO of Farallon. "I expect this will

help Farallon and Apple do great work together on a variety of compelling Internet and intranet applications."

"Our developers face many challenges and opportunities in today's volatile technology marketplace," said Heidi Roizen, ADR vice president. "Their support of the Macintosh platform is dependent upon our ability to provide not only a rich technical opportunity, but also a healthy business proposition. We aim to deliver on both counts."

We'll include more specifics about the newly defined ADR organization when they're made available.

Apple Multimedia Program Becomes Apple *Media* Program

To reflect its broader charter, the Apple Multimedia Program has changed its name to the Apple Media Program. AMP (yes, it's still AMP) will be enhancing its Internet presence to deliver more information, more often, and to provide significant opportunities for members to participate on-line.

AMP member services will now be delivered primarily over via the Web. The AMP Web site will be updated regularly with information and tools for developers working with new media—including demos, Survival Guides, white papers, market research reports, multimedia guidebooks, success stories, discounts, and on-line chats. The latest information will be available in a members-only area. AMP will keep new media developers current on Apple's offerings for authoring and playback whether it's for CD-ROM, Internet, or other delivery mechanisms.

The Web site also includes the AMP Member Showcase—an area on the AMP Web Site that provides information on member products and services as well as a link to their own Web site. Other member benefits include co-marketing programs, discounts on a limited amount of Apple hardware, special new media developer events, and discounts on third-party products and services.

Additionally, members will receive the quarterly Apple Multimedia Information Mailing, consisting of a newsletter and a CD containing the information that's posted to the Web site as well as other content. The Apple Multimedia Information Mailing is also available separately from the Apple Developer Catalog. (For Apple Developer Catalog ordering information, see page 36.)

If you'd like more information about the Apple Media Program, check out the new AMP Web site at http://www.amp.apple.com.

New Release Schedule for Mac OS 8

Apple Developer Relations Vice President Heidi Roizen sent an e-mail message to Apple developers on April 29, 1996, announcing that the first developer release of Mac OS 8 won't be available until mid-year, and confirming that the customer release will ship in mid-1997. (Mac OS 8 is the official name of the next major release of the Mac OS; it was formerly referred to by its code name, *Copland*.)

Following are excerpts of Heidi Roizen's memo, in case you haven't seen it yet. We'll provide further details about the Mac OS 8 release schedule in *Apple Directions* and Apple Directions Express as soon as they're available.

Dear developers,

There is some news that I need to share with you now, as an expression of Apple's commitment to open communication: We've stated in the past that it has been our goal to deliver the first widespread developer release of Copland—what we are calling the Compatibility Edition—to all attendees at the Worldwide Developers Conference (WWDC, May 13–17, 1996). After thorough analysis, it has become apparent to the engineering team and me that we will not be able to achieve this goal.

The Copland developer release strategy is designed to get the operating system into your hands as early as possible. Driver and extension developers need an early start and application developers need to ensure that current shipping products can be tested for compatibility.

The goals of the Compatibility Edition have been specifically designed to meet these needs. Unfortunately, Apple will not be able to deliver a release that is of high enough quality to achieve these goals by WWDC. Rather than provide a WWDC-specific release that would not allow you to make any progress (and further delay the delivery of a release that would meet your needs), we are focusing all resources on delivering the Compatibility Edition as quickly as we can by mid-year.

We are scrutinizing our plans as you are reading this, so we can provide more specific release information at the conference together with more information about commercial availability, which we expect to now take place in mid-1997. Copland remains a high priority for Apple Computer. We are taking steps now to ensure that the product will be aligned with Apple's corporate strategy.

I realize that you were expecting to receive the developer release during the conference time frame. I sincerely apologize that we are not able to get you this release as soon as we had hoped. However, in the interest of fulfilling my personal number-one commitment to you, that Apple deals with the developer community "straight up", I believe that this is the right answer for us all.

We remain committed to your success as our partners in making the Macintosh platform an ongoing success.

Best regards,
Heidi

New QuickTime VR 1.0 Tools Made Available as Apple Plans Next QuickTime VR Release

Last month Apple Computer, Inc., released several new, beta-quality tools for creating QuickTime VR objects and panoramas and began to gather developer feedback for the next release of the virtual reality tool. The new tools—Make QTVR Object and Make QTVR Panorama—let creators of Web site, users of computer graphics program, and photographers more easily create QuickTime VR virtual reality content. The tools are available on the QuickTime VR Web site (**http://qtvr.quicktime.apple.com/**). To help assure that QuickTime VR 1.1, the next release of the technology, best suits developer needs, Apple also released a

preliminary version of the specification for the new release.

Make QTVR Object is used to create a QuickTime VR object movie from photographs or computer-generated images of an object, such as a piece of jewelry, a human figure modeling clothing, or a car. It converts a QuickTime movie whose frames consist of images of the object into a single QuickTime VR object. The resulting QuickTime VR object can be rotated by the user to see all sides of the object. Object files generated from a single row of photographs around an object are typically about 500K in size.

Make QTVR Panorama is used to create a QuickTime VR panorama from a panoramic photograph or a computer-generated image of a scene. Make QTVR Panorama converts a panoramic PICT file from a computer graphics program or a scan of a panoramic photograph into a QuickTime VR panorama. The user can pan and zoom within the resulting QuickTime VR panorama. Panoramas generated from photographs can be as small as 150K.

Both of these tools create content that can be made part of a Web site, a CD-ROM title, or another program, such as a kiosk presentation. This content can be used with the QuickTime VR Authoring Tools Suite or authoring programs such as Apple Media Tool, Macromedia Director, or mFactory from mTropolis.

In addition to providing the new tools for QuickTime VR 1.0, Apple is beginning the process of creating the next release of QuickTime VR. Apple has shipped to key developers and partners the QuickTime VR 1.1 Application programming interface (API) specification for review and comment. Apple will gather feedback on this specification and intends to announce the features of, and schedule for, QuickTime VR 1.1 at the 1996 Worldwide Developers Conference, held May 13–17 in San Jose. The announcement is expected to include seeding and final delivery dates for the next release of QuickTime VR along with details of its feature set and of the seeding program.

If you sign a nondisclosure agreement with Apple, you can receive the QuickTime VR 1.1 API specification. For more information, registered Apple developers can contact Developer Seeding at dss.software@applelink.apple.com. Please provide your name, company name, mailing address, and phone number. To become a registered Apple developer, call the Apple Developer Hotline at 408-974-4897.

You can continue to use the QuickTime VR Authoring Tools Suite 1.0 for creating photographic panoramas from multiple photographs ("stitching"), for adding clickable "hot spots," and for creating complete virtual reality experiences. The QuickTime VR Authoring Tools Suite 1.0 is available from the Apple Developer Catalog. (See page 36 for ordering information.)

Apple Licenses Sun's Java

Apple Computer, Inc., recently announced that it has licensed Sun's Java programming environment. Apple plans to use the license to make Java an integral part of its operating systems, including the Mac OS, Pippin, and Newton.

You'll be able to create applications for Apple platforms that take advantage of the built-in Java technology. A Java-capable word processor, for example, will enable a student writing a term paper to embed a Java applet in the electronic document that demonstrates a scientific theory through animation. With a Java-enabled spreadsheet, a financial analyst could download a graphing applet from the Internet, plot data from a ledger, and upload the resulting graph to a network server.

The combination of Java with the Mac OS should further expand the Macintosh computer's technical edge for Web graphics and other new Internet media development. According to the Chicago-based consultant Mirai, 40.9 percent of respondents polled in 1995 already created the graphics for Web sites on a Macintosh computer.

Apple expects to incorporate Java into its media authoring technologies, Internet servers, and client software and in Cyberdog, its OpenDoc-based Internet suite. With Java, media authors will be able to use one language for a multitude of solutions that address the needs of individuals as well as organizations.

"This is an important step in Apple's plan to launch cutting-edge Internet products and services," said Larry Tesler, Apple's vice president of Internet platforms. "Licensing Java from Sun will enable Apple to make this key Internet standard widely available in its products."

"With the addition of Apple to the growing list of Java licensees, Java truly becomes the standard for the creation of platform-independent applications," said Alan Baratz, president of JavaSoft, Sun Microsystems. "As one of the leading personal computer suppliers, and a household name in consumer-oriented, easy-to-use technology, Apple's license brings Java to a whole new category of users."

If you want to begin working with Java on the Macintosh, a variety of Macintosh Java development environments have already been announced, including the following:

- Roaster from Natural Intelligence; for more information, go to the Natural Intelligence Web site (**http://www.natural.com/pages/products/roaster/index.html**).
- Caffeine from Symantec, Inc.; for more information, go to the Symantec Web site (**http://www.symantec.com/lit/dev/mactools.html**).
- Discover Programming with Java from Metrowerks; for more information, go to the Metrowerks Web site (**http://www.metrowerks.com/products/discover/java.html**).

Technical Support Now Available to All Developer Program Members

If you're a member of the Macintosh Associate Program, the Newton Associate Program, or the Apple Media Program (formerly the Apple Multimedia Program), you now can take advantage of programming-level support from Apple's Developer Technical Support (DTS) engineers. A new support option, just introduced by Apple Developer Relations allows developers to purchase support on a per-question basis. Members of other programs—Macintosh and Newton Partners, and Macintosh and Newton Associates Plus—can continue to make use of DTS support as they have in the past.

If you're a member of one of these programs anywhere in the world, you can begin to take advantage of DTS support immediately. Submit your code-level programming questions by e-mail to devsupport@applelink.apple.com. DTS will respond to all questions within three business days. The charge per question is $50 (U.S.). If your question reveals

a bug in Apple's software or documention, DTS will not charge you (in fact, they'll thank you!).

Please note that DTS is equipped to handle code-level programming questions about Apple's application programming interfaces. Any comments, questions, or feedback about this and other Apple developer programs should be sent to devfeedback@applelink.apple.com.

Increased Power PC Compatibility, All-in-one Design Featured in New Apple Hardware

In April, Apple Computer, Inc., introduced a variety of new hardware products, including the following:

- four faster Power Macintosh models: the Power Macintosh 7200/120 computer, the Power Macintosh 7600/120 computer, the Power Macintosh 8500/150 and∆8500/132 computers, and the Power Macintosh 9500/150 computer. All the new Power Macintosh systems can be upgraded to run at 200 MHz, once faster processors are available; additionally, Apple announced several PowerPC upgrade options
- the Power Macintosh 7200/120 PC Compatible computer, which includes a 120 MHz Power PC 601 chip to run the Mac OS and one of two PC Compatibility cards—one with a 100 MHz 586 processor, the other with a Pentium 100 MHz processor—to run Windows 3.1, Windows 95, and DOS 6.22. Both PC Compatibility cards can also be used with any existing Peripheral Component Interconnect (PCI)-based Power Macintosh system
- two all-in-one Power Macintosh systems for educational customers, the Power Macintosh 5260/100 computer and the Power Macintosh 5400/120 computer

Details about the new products follow.

New Power Macintosh Models

The Power Macintosh 7200/120 computer, driven by a 120 MHz PowerPC 601 processor, provides clock-speed improvements over the current Power Macintosh 7200/90 and Power Macintosh 7200/75 systems. With optional level-2 cache, the new computer provides up to a 50-percent performance improvement over the Power Macintosh 7200/90 system. The Power Macintosh 7200/120 system with 8 MB of memory, a 1.2 GB hard drive, and quadruple-speed CD-ROM drive has a U.S. Apple price of $1,899.

The Power Macintosh 7600/120 computer includes a 120-MHz PowerPC 604 microprocessor, a 256K level-2 cache, and clock speed improvements to provide up to a 50-percent performance improvement over the Power Macintosh 7500/100 system. A Power Macintosh 7600/120 computer with 16 MB of memory, a 1.2 GB hard drive, a quadruple-speed CD-ROM drive, and 256K level-2 cache has a U.S. Apple price of $2,999.

The PowerPC 604-based Power Macintosh 9500/150, Power Macintosh 8500/150, and Power Macintosh 8500/132 systems are higher-clock-speed versions of the current Power Macintosh 9500 and Power Macintosh 8500 models. The new computers both provide up to a 25 percent performance improvement over current models. U.S. Apple prices for these systems range from $3,899 for a Power Macintosh 8500/132 with 16 MB of memory, a 1.2 GB hard drive, a quadruple speed CD-ROM drive, and 256K level-2 cache to $4,799 for a Power Macintosh 9500/150 with the same configuration in addition to on-board video.

For those wishing to upgrade existing systems, Apple introduced the Power Macintosh Processor Upgrade Card 132 MHz (U.S. $899) and Power Macintosh Processor Upgrade Card 120 MHz (U.S. $599); both can be used with Power Macintosh 7500/100 systems, which include a processor upgrade slot. Additionally, the new PowerPC 604 processor-drive 8500 Logic Board Upgrade (U.S. $1,799) allows Macintosh Quadra 800, Macintosh Quadra 840av, and Power Macintosh 8100 computers to be upgraded to the performance of a Power Macintosh 8500 computer. A 7600 Logic Board Upgrade (U.S, $1,299) is also available for upgrading a Power Macintosh 7200 to the processing power of the PowerPC 604 chip and level-2 cache capabilities of the new Power Macintosh 7600 computer.

PC-Compatible Solutions

Apple's next-generation PC-compatible solutions—the Power Macintosh 7200/120 PC Compatible computer (U.S. $2,599 to $2,799, depending on configuration) and a PCI-based Power Macintosh using one of the two new PC Compatibility cards (U.S. $799 for 586 version, U.S. $1,049 for more expandable Pentium version with more level-2 cache)—run virtually all Windows, DOS, and Mac OS software, giving customers the greatest level of compatibility of any mainstream personal computer. Improving on former cross-platform systems, the computer and cards can be connected to a variety of networks, including Novell Network SPX/IPX, TCP/IP and NETBEUI protocols in Windows and DOS environments via a built-in Ethernet connector and ODI and NDIS 2.0 drivers. Due to robust multinode support, the systems, with the appropriate network access, can support two Internet connections simultaneously—from the Mac OS and Windows environments.

These cross-platform products are expected to draw new users to the Mac OS platform from among the ranks of existing Wintel customers. Apple customer research shows that the previous cross-platform solution, the Power Macintosh 6100/66 DOS Compatible computer, attracted x86-based PC owners to the Macintosh platform at almost 3 times the rate of other Power Macintosh systems. In fact, the majority of Power Macintosh 6100/66 DOS Compatible customers were not replacing another Macintosh system; instead, they were either acquiring their first personal computer, replacing a x86-based PC, or adding to their installed base of PCs.

Educational Systems

The Power PC 603e-based Power Macintosh 5260/100 computer (U.S. $1,699) includes a quad-speed CD-ROM drive, 16-bit CD-quality stereo speakers, an integrated 3.5 inch 800 MB IDE hard drive, built-in 14-inch (viewable area 12.3 inches) color monitor, mouse, and AppleDesign keyboard, and 16 MB of memory (expandable to 64MB). In addition, customers can add a video input card, a video-out connector, or a TV tuner.

The more powerful PCI-based Power Macintosh 5400/120 system (U.S. $2,299) also uses the Power PC 603e chip. It features 16MB of DRAM (expandable to 136MB), an internal 1.6 GB hard drive, and built-in Ethernet. The system comes equipped with a video input card and video-out connector, an expansion bay for an optional TV tuner, as well as a quad-speed CD-ROM drive, stereo speakers, a microphone, built-in 15 inch (viewable area 12.8 inches) color monitor, mouse, and AppleDesign keyboard. ♣

Technology

develop Issue 26: Mac OS 8, QTC, and SOM

Issue 26 of *develop,* Apple's award-winning technical journal, has more great content than ever. In this, our biggest-ever issue, you'll find articles on Mac OS 8, QuickTime Conferencing, SOM™, and more:

• "Planning for Mac OS 8 Compatibility"—Mac OS 8 brings changes that may affect your code. This article discusses the compatibility ramifications of Mac OS 8 and gives some sound advice for how to get your code ready now.

• "Connecting Users With QuickTime Conferencing"—QuickTime Conferencing allows users to share time-based data such as video and sound. Here are the basics.

• "OpenDoc Parts and SOM Dynamic Inheritance"—Although you don't need to know much about SOM to write OpenDoc parts, with a little knowledge about this underlying technology you can do some very useful things.

• "Adding Custom Data to QuickDraw 3D Objects"—By defining your own attribute and element types, you can attach custom data to QuickDraw 3D objects. This flexibility opens up a world of new possibilities, a few of which are explored in this article.

• "64-Bit Integer Math on 680x0 Machines"—There's a 64-bit library built into the Toolbox on the Power Macintosh, and there's also one built into QuickDraw GX. Finally, here's a library that will work on any Macintosh, using built-in routines if available.

You'll also learn about living copacetically with the Display Manager, automating complex editing tasks in MPW, integrating scripts into your application, and avoiding the (revised) top ten printing crimes. We hope you'll judge this to be the best-ever issue of

please turn to page 24

CD HIGHLIGHTS

Reference Library Edition, June 1996

The new and improved version of Apple HTML Local Search wasn't quite ready for this month's disc, so we'll deliver it, along with another big pile of technical documentation in HTML format, on the July System Software/SDK edition. Those of you who tried the March version and haven't sent in your survey yet—please do! I don't want to see any gripes about Acrobat on the Semper.fi list server from people who haven't sent us their feedback about this possible new direction. . . .

In addition to updates to the Developer University schedule, Macintosh Technical Notes, Macintosh Technical Q&As, and Toolbox Assistant, here's this month's new and revised stuff.

develop Issue 26

This is the electronic version of *develop,* the Apple Technical Journal, Issue 26. For details, see the article to the left.

MoreFiles 1.4.2

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). The routines have been tested (but not stress-tested), documented, and code-reviewed by Apple 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

• routines for dealing with pathnames

See the file !MoreFiles Read Me for a description of fixes and improvements in version 1.4.2.

Network Server Developer's Guide

The *Developer's Reference Guide for the Apple Network Server* is the source for information about developing client/server applications and writing device drivers for the Apple Network Server. The guide documents differences between AIX for the Apple Network Server and IBM's AIX.

Reference Library Edition

This guide covers hardware and software requirements to be considered when developing AIX for Apple Network Server device drivers or server applications and a Mac OS client. Part I covers the development of client/server applications, including both Mac OS and AIX considerations. Part II covers the

please turn to page 24

Multitasking Under Mac OS 8

Part 2: Multithreading

By Tony Francis

In last month's issue, Part 1 of this article provided a detailed look at multitasking under Copland—or Mac OS 8, as it's now known. It described how Mac OS 8 provides separate address spaces for data, a cooperative address space in which most applications execute, and system-wide preemptive multitasking that simulates the cooperative multitasking needed by System 7 applications.

To conclude our look at Mac OS 8 multitasking, Part 2 of this article describes Mac OS 8 multithreading capabilities, features that will help your applications use the computer's processor more efficiently. This article is based on a chapter from the forthcoming Apple Press/Addison-Wesley Longman book, Mac OS 8 Revealed *by Tony Francis, to be published in the third quarter of this year.*

• • •

Just as the operating system makes the most efficient use of computer resources through its multitasking capabilities, a program can make the most efficient use of computer resources by incorporating multithreading capabilities. Whereas multitasking efficiently interleaves the execution of multiple programs on a single CPU, multithreading efficiently interleaves multiple paths of execution within a single program or set of programs. For example, one thread of execution in a program might handle user interactions, another might perform calculations, and a third might perform file I/O.

Multithreading makes an application highly responsive to the user while increasing overall system performance. On multiprocessor computers, where tasks execute simultaneously on multiple processors, multithreading offers significant performance gains to applications.

When you develop for Mac OS 8, you can thread products using one or a combination of the following three approaches:

- You can divide operations so that more than one task is performed in a single process; for example, you can incorporate a main task and one or more additional tasks in a cooperative program.
- You can divide operations so that they are performed by tasks in more than one process; for example, you can incorporate a user interface task in a cooperative program and background processing tasks in a separate server program. (Under Mac OS 8, a server program is one that manipulates data within its own protected address space. Cooperative programs run in the same address space and must share access to a library of code called the Mac OS 8 cooperative services. See the first part of this article for more details.)
- You can arrange operations so that more than one cooperatively scheduled thread is performed within a task.

Key Terms and Concepts

Before going any further, it will be helpful for you to understand the following key terms, which will be used frequently in this discussion:

A *thread* is a path of execution for an application. To thread an application is to give it more than one path of execution.

A *task* is the basic unit of program execution. Preemptively scheduled and assigned a priority by the Mac OS 8 microkernel, every task has its own stack and set of registers. The microkernel uses processes to track the resources required by tasks, so that every process is associated with at least one task, and several tasks can be associated with a single process.

A *cooperatively scheduled thread* is one of multiple paths of execution in a task. Within a task, these threads cooperate by yielding execution control to one another. Cooperatively scheduled threads—which are the same as System 7.5 threads—can be scheduled for execution only when the task containing them is running. Although the microkernel preemptively schedules all eligible tasks for execution, programs have execution control over the cooperatively scheduled threads they create. From the main task of a cooperative program, any cooperatively scheduled thread can call the Mac OS 8 cooperative services.

On a *multiprocessor computer,* a multithreaded program can send its tasks to separate processors for simultaneous execution.

Programs use *interprocess communication* to exchange information among tasks within processes or between tasks in different processes.

Note that in Mac OS 8 terminology, the term *task* describes the entity referred to as a *thread* in some other operating systems, such as UNIX® and Windows NT. With Mac OS 8, the term *thread* is more abstract because, as mentioned above, there are three ways to implement multiple paths of execution in Mac OS 8.

Whereas some operating systems allow you to create multiple threads within a process in the way that you can create multiple tasks within a Mac OS 8 process, Mac OS 8 supports another level of threading—the creation of cooperatively scheduled threads within a task. This level of threading was introduced with the System 7.5 Thread Manager. To avoid confusion, Mac OS 8 terminology refers to the Mac OS 8 version of the System 7.5 Thread Manager as the *Cooperative Thread Manager.* The term *cooperatively scheduled threads* refers to threads created through the Cooperative Thread Manager. The term *task* refers to a preemptively scheduled path of execution in Mac OS 8.

Major Points of Interest

You can thread your software to increase system efficiency and user responsiveness. A multithreaded program is structured into parallel operations, each of which has access to the CPU. This feature allows a user to continue working within an application without waiting for the application to complete lengthy operations. For example, a scientific simulation application could create one thread that handles user interaction and another thread that performs intensive statistical calculations in the background. The user can continue to interact with the program even while it's performing statistical calculations.

To the user, it appears that multithreaded operations take place simultaneously. However, the microkernel interleaves the execution of these operations on a single CPU so quickly that it looks as if they're happening simultaneously. The microkernel also supports multiprocessor computers, where parallel operations happen simultaneously. At the initial release of Mac OS 8, the operating system supports the System 7.5 programming interface defined by DayStar Digital and Apple Computer for multiprocessor computers; Apple intends to provide full multiprocessor support in subsequent releases of Mac OS 8.

please turn to page 20

Addictive Interfaces

Building Interfaces Your Users Can't Stop Using

By Peter Bickford

To the outside world it was 11 o'clock in the morning, and the sun was shining brightly. I, however, was in a dark, cavern-like building watching a kid locked in a deadly karate battle against a many-armed monstrosity. To the right of me, a child no older than twelve was dropping bombs on a terrorist hideout, while behind me a woman sat mesmerized as she turned falling blocks of various shapes so that they would form orderly rows as they landed.

I was standing in a video arcade, the third I'd visited that morning. I was on a case, sent by Mark Gavini, Apple's games evangelist, to discover the secrets of building successful game interfaces. As I looked around the dimly lit interior, I saw row upon row of machines, some with throngs of people waiting to use them, others looking lost and lonely as they tried in vain to pull in an audience with their gaudy self-running demos.

All this time, I was busy asking myself one question: "Why?" What makes *Mortal Kombat* draw in the crowds while *Street Fighter* stands deserted? What spell does *Tetris* cast that makes people seem not to be able to stop playing it? And why had someone done great violence to the *Mad Dog McCree* game? I wasn't leaving until either I found some answers, or I ran out of quarters.

And I had brought an awful lot of quarters. . . .

Secrets of Successful Games

After some time watching and playing the games, I was starting to notice some patterns. I wanted to double-check my information, however, so I knocked on the door marked "private" and asked to speak to the person running the place. As it turned out, the people inside were surprisingly eager to cooperate. They showed me sales figures, told me which games were hot and which ones had bombed, as well as giving me their own theories about why. Together, we pieced together the puzzle of what it takes to make a successful game. As it turned out, the answer was very simple.

To make a great game, you need to accomplish three things.

1. Hook your customers by getting them to lose themselves in The Game.
2. Keep them "in play" as long and as deeply as possible.
3. Keep 'em hooked by giving them greater challenges and greater rewards.

Step #1: Getting Your Customers Hooked

Getting users hooked begins with enticing them into leaving their everyday lives behind for a moment and entering that unique reality known as The Game.

• *Make the game as alluring as possible.* The game that grabs the most attention will be the one that offers players the most awesome sensory and mental experience. This is where all your technical skills come in, giving your players the flashiest and most realistic graphics: stereophonic (or more) sound and music; movies; animations; virtual reality. Use every weapon you can to catch users' attention and invite them into The Game.

Keep in mind that the mental component of game playing is every bit as important as the sensory part. Even the most sense-shattering shoot-em-up becomes boring unless it somehow captures a user's imagination. Great games offer the user a sense of challenge, wonder, or accomplishment. Some games, such as *Tetris,* have just enough sound and animation to get by. What really makes it special, however, is its sense of absolute mental involvement.

• *Make it easy to start playing.* We've all sat down with friends to play a new board game, only to have the evening come to a crushing halt as someone was forced to read out loud the game's six pages of rules and regulations. This is usually followed by 20 minutes of clarifications and confused muttering. Compare this to the classic game of "Othello," whose rules can be explained in a few sentences, but whose strategies can take years to master.

If you want people to lose themselves in your game, it's important that they be able to start playing as soon as possible. Keep the rules simple, and make sure the overall goal is clear. Instead of making the users read instructions, let them start playing immediately, and then have the game teach them how to play. *Sewer Shark,* for instance, uses a gruff flight instructor to lead the "rookie" (the player) through the rules of engagement before taking off. Of course, experienced players can skip these instructions by pressing a button.

Step #2: Keep 'em in Play

Once you've got your users playing, your biggest task is to let them enjoy the ride. You want to keep them "in play" as long as possible without annoying distractions that wrench them back into reality.

• *Consider hiding the rest of the computer.* This may be the only time you hear a human interface person give this advice, but in some cases it's perfectly OK to hide nongame windows, the desktop, and even the menu bar itself. Gaming is not generally a multiapplication task. ("I'll just copy these cells from my Excel worksheet, then paste them into my laser rifle's energy setting . . ." Yeah, right.) When you're chasing your office mates around the alien complex in *Marathon II,* you don't want to be worrying about that unfinished report lurking in the background window.

So let users focus on playing, but give them an easy way to get back to work when the time comes. If you've hidden the menu bar, make sure there's an easy way to show it again. Also consider showing it if the user moves the mouse to the top of the screen, types lots of wrong keys, or otherwise indicates confusion.

• *Don't have them steer the game—put users in the game.* Using commands like "Go North" in computer games was OK when the games were being played on teletypes. To really lose themselves in the game, however, users need to be able to play without consciously thinking about the controls. Joystick control and point-of-view perspectives such as those used in *Wolfenstein 3-D* are ideal for building a sense of involvement. If you have to use keyboard navigation, base the controls more on hand position than on the first

letters of the commands (for instance, use the arrow keys or I-J-K-L for directionals, rather than N-S-E-W). Make sure you also give users an option to choose the keys that work best for their special needs. In any case, make sure that you haven't designed in so many key commands that your users are unable to use them without stumbling around.

• *Minimize the user's memory requirements.* Low-end, game-playing computers usually have at least a megabyte or two of free RAM for running your game. Their users, however, usually only have about 7 (±2) bytes of free memory while they're busy blasting away aliens. Respect this limitation and don't expect your users to remember things while playing your game. Design any sort of program help so that it can be summoned when needed (and forgotten when no longer needed). Avoid memory-based puzzles ("Now, remembering what the little elf told you in scene 1, you should be able to figure out the answer to this final riddle!"). Users will remember strategy, but forcing them to remember anything else is a sure road to player frustration.

• *User control.* Many events in a computer game are random, but one of the core principles of any sort of successful gaming is that users are in control of their own actions. Frustration sets in the moment users feel they are struggling with the computer in order to control their own movements. For users to feel like they're the ones in control, you should start by making sure the computer responds to any user action within .2 seconds. If you can respond in 200 milliseconds or less, users will feel like they're controlling the action onscreen. After 200 milliseconds, the computer is seen to be merely echoing their actions.

You should also try to cut users some slack in responding to their actions. If users are walking along the ground and want to climb a tree, don't make them start the climb only at the precise pixel position where the tree trunk begins—allow them to start climbing if they're within, say, 20 pixels of the tree trunk.

• *Challenge, don't frustrate.* Players love new challenges, but they despise pointless frustration. Realizing the difference is what separates games whose coin inserts get a workout from those whose front glass has been shattered by furious users. For example, I am convinced that there is a special place in Hades reserved for game designers who let players continue for any length of time after making a move that ensures that they can not possibly win the game. Forcing users to back up and start over is about as enjoyable as making them retype a document they lost when their computer's power failed.

Step #3: Keep 'em Hooked

With most software, it's enough that customers can use it enjoyably, productively, and without frustration. If you want to build a really great game, you want more: to get the users so hooked that they can't help but come back for more. You don't just want user-friendly games, you want user-addictive games.

For inspiration on how to design these, you can look to the great behavioral psychologist B.F. Skinner. Skinner was studying "conditioning," which in his case amounted to teaching rats to press little levers in their cages to get food. Initially, Skinner would establish the lever-pressing behavior by giving the rats a food pellet every time the lever was pressed. This seemed like great fun to the rats for a while, but eventually, Skinner was left with a cage full of obese rats who only got around to pushing the lever for more food when there was nothing for them to watch on TV.

If Skinner had wanted to raise fat rats, he could have just given them a feed bag and dispensed with the levers altogether. But no, Skinner wanted to turn his rats into lean, mean, lever-pushing machines. So, he started varying how many times the rats would need to press the lever in order to receive a food pellet. If he made them press the lever too many times before giving them food, the rats eventually gave up trying. On the other hand, if he paid off their lever-pressing too frequently, they got bored. As it turned out, the way to maximize their lever-pressing was to give them a fairly high payoff rate at first, then lower the rate so that the food pellets only dropped occasionally. The rats, having seen the potential for reward, would keep pushing the lever again and again, periodically receiving reassurance that their efforts were not totally in vain.

The parallels between Skinner's rats and, say, contestants in lottery scratcher games are obvious. Computer gamers, however, are playing on a slightly higher level.

True, you do need to start by giving your gamers good "first quarter" play. Make sure that the very first challenges or opponents they face are not insurmountable. Have mercy on players who are just starting out—some pinball games are even gracious enough to give an extra ball if the player's score is *below* a certain point.

After that, start making the game more difficult by stages so that the player has to keep working harder and acquiring new skills in order to advance. So as not to bore more accomplished players, you may also want to give users a way to skip over the easier portions of the game—or more devilishly—make the early opponents adjust in difficulty to match the playing abilities of the last gamer.

Unlike Skinner's rats, however, gamers are capable of seeing the big picture. To convince them to keep playing, you need to give them more than just the same old payoffs. You need to couple their new challenges with new experiences, new possibilities, and new rewards. In doing so, you are satisfying players' mental needs, not just their desire for cool-looking explosions. Players may get a physical rush out of playing your game, but it's the way you fulfill their mental needs that addicts them and keeps them coming back for more.

That's all the time I have for now. Until next time, happy gaming!

—Doc

Peter Bickford is a member of Apple Computer's Human Interface Design Center. You can reach him on AppleLink at THE.DOKTOR or on the Internet at the.doktor@applelink.apple.com.

Mac OS 8 Multitasking

continued from page 17

Threading

In the same way that multitasking makes the system more efficient, multithreading makes an application more efficient. A multithreaded application has more than one path of execution. For instance, one thread in an application might handle user interactions, another thread might perform I/O, and yet a third might perform background calculations. Even when this application waits for user interaction or the completion of I/O operations, it can perform background calculations.

Not all programs benefit from being multithreaded. For instance, an application that handles user interactions and little else would not benefit greatly by having more than one task. However, if your application presents a user interface while performing time-consuming I/O or processing-intensive operations in the background, you can provide real user benefit by threading the application.

Under Mac OS 8, you can thread your products by using one or a combination of three different approaches. You can divide your program's operations so that they are performed

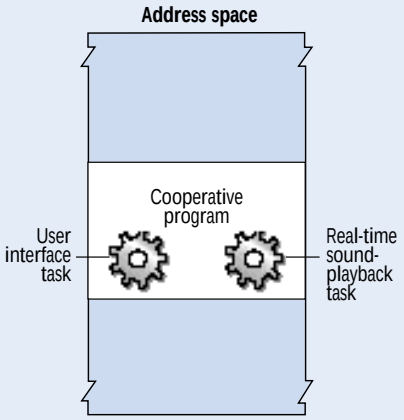
- by tasks in more than one process
- by more than one task in a single process
- by more than one cooperatively scheduled thread within a single task

Tasks in Different Programs

As discussed in Part 1 of this article, under Mac OS 8 there are two different types of programs. The first is called a *cooperative program,* which executes in the cooperative Toolbox environment. The second is referred to as a *server program,* which manipulates data within its own protected address space. For example, an application may consist of a cooperative program that presents a graphical user interface and a server program that performs I/O or calculation-intensive operations in the background. Thus, a World Wide Web application may consist of a cooperative program with which the user creates and maintains a local Web site and a server program that makes Web pages available to remote users. These two paths of execution make the application multithreaded. The main task of the cooperative program manages the user interface for the portion of the application that manages the Web site. The main task of the server program manages network I/O for the Web-page server portion of the application.

It is often necessary for these two tasks to communicate. For example, the server program's task may send network activity information to the cooperative program's task, allowing the cooperative program to display this information to the user. To share this data safely, these tasks must synchronize their access to it. For example, the server program can place this data in a read-only area of shared memory, thereby allowing the cooperative program to read but not change it. As you'll see later in this article, Mac OS 8 not only provides interprocess communication mechanisms for exchanging data between tasks, but also offers synchronization mechanisms for protecting the data exchanged between tasks.

Two tasks within a cooperative program



You might choose to thread an application by using separate cooperative and server programs whenever you need one of several tasks to

- have its data protected within its own address space
- be available whenever the computer is on
- extend its services to multiple client programs

You can use a server program to create a task characterized by address space protection, availability, and client extensibility. You can then use a cooperative program to create a task that manages user interaction on behalf of the server program.

Address Space Protection. You might implement a thread of execution in a server program to increase the reliability of that portion of the application. The operating system builds a separate, protected address space for the data of a server program; therefore, errors in other programs can't corrupt the data of that server program. For example, a database server that provides critical information could be implemented as a server program. Thus, even if other programs on the computer were to crash, the server program could continue serving data. You could then implement another thread of execution in a cooperative program, allowing the user to enter, change, and analyze data maintained by the server program.

Availability. You might implement a thread of execution in a server program whenever a task should run the entire time a computer is on. Whereas users typically launch and quit cooperative programs, server programs are usually launched automatically when the user starts a computer, and they're usually run until the user shuts the computer off. For example, a task that receives electronic mail messages whenever the computer is on should run as part of a server program. You could then implement another thread of execution in a cooperative program, allowing the user to read and respond to incoming messages.

Client Extensibility. When a single thread can be used by more than one other program, that thread can be implemented as a task in a server program. For instance, a server program that monitors network activity might be useful to a suite of network-management applications, such as an e-mail gateway program, a program for managing Web sites, and a program that backs up remote disk drives.

Other Indirect Interaction Between a Server Program and the User. A task within a server program can also interact with the user indirectly by calling the Notification Manager, a reentrant service, to send a user notification. (A reentrant sevice is code that can be used concurrently by several pieces of code.) A user notification is an audible or visible indication to the user that a program requires the user's attention. User notifications can take such forms as sounds, icons that blink at the top of the screen, and on-screen alert boxes containing short messages. For example, an e-mail server program can use the Notification Manager to play a sound and display a blinking icon in the menu bar to notify the user of incoming mail. This alerts the user of the need to open a cooperative program to read and respond to incoming mail.

Multiple Tasks in the Same Process

Although threading an application with separate cooperative and server programs has its advantages, the simplest and most straightforward way to thread an application is to create additional tasks in a cooperative program. From its main task, a cooperative program can start additional tasks to off-load work that doesn't involve the user interface. By letting the main task manage the program's user interface and using other tasks to perform time-consuming data processing or I/O operations in the background, an application can perform more efficiently and offer greater productivity to the user; by placing multimedia operations in a separate real-time task, an application can perform time-critical multimedia operations without interruption. For example, a multimedia authoring program can use the main task to interact with the user and another task to capture and save video data in real time or to perform real-time sound playback. The figure on page 20 illustrates a cooperative program that incorporates two tasks.

Whereas the main task is cooperatively scheduled against the main tasks of all other cooperative programs in the system, all additional tasks created within a cooperative program are preemptively scheduled with all other eligible tasks in the system. For this reason, time-intensive I/O, computationally intensive data processing, and critical real-time operations make the best use of the CPU when they're placed in tasks other than the main task.

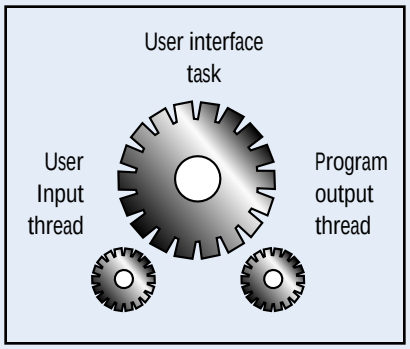
Because the part editors within an OpenDoc document handle user interaction, they use cooperative services. For an OpenDoc document, the main task incorporates the document's constituent part editors, and this main task is cooperatively scheduled like the main task for any other cooperative program. However, you can also create additional tasks from an OpenDoc part and, like any additional tasks created for a cooperative program, these tasks must use only the reentrant services.

It is often necessary for tasks within the same process to share information. To ensure the integrity of this information, tasks should synchronize their access to it. Compared to data sharing between tasks in separate programs, data sharing between tasks in the same program requires less overhead. (Mac OS 8 interprocess communication and synchronization mechanisms are discussed later in this article.)

Cooperatively Scheduled Threads in the Same Task

Cooperatively scheduled threads are useful when you'd like to provide multiple paths of execution within a single task. Cooperatively scheduled threads, created with the programming interface defined by the Cooperative Thread Manager, are invoked from tasks. Cooperatively scheduled threads can be scheduled for execution only when the task that created them is running. These types of threads are said to be cooperative because they yield control to one another at programmatically defined times. This prevents one cooperatively scheduled thread from being preempted by any other thread within the same task.

A task with cooperatively scheduled threads



The main task of a cooperative program can call the Mac OS 8 cooperative services from any of its cooperatively scheduled threads. As illustrated in the figure on this page, a cooperative program—such as a laboratory control application—can use one cooperatively scheduled thread to present a window for user input and another thread to present a window for program output. A user of this program could then instantly view data returned from laboratory instruments in one window while controlling the instruments from another window.

Cooperatively scheduled threads can call cooperative services only from the main task of a cooperative program. Even when implemented in background tasks, cooperatively scheduled threads are useful if

- you want to gain explicit control over the time when threads yield execution control to one another
- you want shared data to have more simplified synchronization
- you want your software to gain the switching efficiency delivered by cooperatively scheduled threads

Program Control Over Thread Execution. Unlike preemptively scheduled tasks, which the microkernel can interrupt at any time and cause to execute in any order, cooperatively scheduled threads are executed under the control of the program using them. Within a program, in other words, one cooperatively scheduled thread cannot interrupt another thread. Instead, cooperatively scheduled threads explicitly yield control to each another at points defined by you.

Simplified Synchronization. Cooperatively scheduled threads simplify data synchronization. Whereas multiple tasks sharing data must use synchronization mechanisms to access that data safely, you don't need to employ these mechanisms when using cooperatively scheduled threads. Because a task's cooperatively scheduled threads cannot preempt one another, access to the data they share is automatically serialized. You simply need to ensure that each cooperatively scheduled thread yields control only after making changes to data shared with another cooperatively scheduled thread.

Efficient Switching. Because cooperatively scheduled threads are called from a task, the microkernel doesn't perform a context switch whenever a task switches between them. A task is therefore slightly more efficient at switching between cooperatively scheduled threads than the microkernel is at switching between preemptively scheduled tasks.

Multiprocessor Support

Mac OS 8 also provides support for multiprocessor computers. A multiprocessor computer has more than one processor to execute instructions. For example, the Genesis MP computer from DayStar Digital is a Mac OS–compatible computer that includes four PowerPC processors for simultaneously running parallel threads of execution. The Genesis MP platform is an example of an asymmetric multiprocessor (AMP) system. On an AMP system, one processor—the master processor—executes all operating-system–related operations, such as making scheduling decisions and performing I/O. All other processors—called *slave processors*—perform operations allocated to them by the master processor. To take advantage of the Genesis

MP platform, a program must explicitly call its programming interface to schedule threads for execution on its slave processors.

(For the Genesis MP platform, DayStar Digital and Apple collaborated to define a System 7.5 programming interface allowing you to thread your programs. Applications using this programming interface remain compatible with Mac OS 8.)

The Mac OS 8 microkernel is designed to provide symmetric multiprocessor (SMP) support as well as AMP support. On an SMP system, each processor executes its own copy of the operating system and communicates with the other processors as needed. Although the first release of Mac OS 8 doesn't support an SMP implementation, Apple intends for future versions of the operating system to provide this support. You won't need to perform any special programming to take advantage of future SMP systems; instead, the operating system will automatically schedule multiple tasks to execute simultaneously on all available processors.

Interprocess Communication and Data Synchronization

In a multitasking environment, it is necessary for tasks to communicate. Tasks cannot move between processes. However, tasks using the interprocess communication mechanisms provided by Mac OS 8 can pass information to each other, even across different address spaces. A calendar application, for example, might consist of a cooperative program and a server program that share data. The cooperative program might allow the user to view and maintain a personal datebook. The server program might handle meeting proposals submitted by colleagues at network-connected computers. When it receives a meeting proposal, the server program passes this information to the cooperative program for display in the user's datebook.

Tasks sharing the same set of data must synchronize changes to the data. When the user of the networked calendar application schedules an appointment, for example, a task for the cooperative program must indicate that it's updating the user's calendar. If a colleague on the network simultaneously schedules an appointment with the user, a task for the server program must check for this indication and, upon finding it, block its own execution. To resume execution, the blocked task needs to know when the other task is finished updating the user's calendar. The microkernel provides various services that allow tasks to synchronize their operations in this way.

Apple Events

Apple events are the most pervasive form of interprocess communication in Mac OS 8. An Apple event is a data structure used chiefly for sending information or instructions to a task. An Apple event contains a flexible hierarchy of additional data structures. This flexible hierarchy allows data to be shared between tasks at various levels of detail. For example, a meeting proposal can be sent across the network in an Apple event targeting a day, hour, or range of minutes within a user's calendar.

Apple events were introduced in System 7 so that applications could share services and information with each other. In System 7, only applications could use Apple events; other types of software, such as device drivers, could not. In Mac OS 8, all types of software can use Apple events. Communication can take place between tasks in the same process or in different processes, in the same address space or in different address spaces, on one computer or on connected computers. An OpenDoc part can also use Apple events to communicate with other parts and with tasks in any address space. An Apple event contains the identification of its destination, and the operating system delivers the Apple event to that destination.

Some of these uses of Apple events in the Mac OS 8 platform include the following:

- *Event notification.* Apple events are the chief means by which the operating system informs programs about user and system activity. For example, when the user chooses a command from an application's menu, the operating system sends an Apple event to the application; this Apple event contains all the information the application needs to begin responding to the user action.
- *Scriptable automation.* Programs that respond to Apple events can be controlled and automated by means of scripts created with scripting languages such as AppleScript. Using the assistance services of Mac OS 8, for example, a scriptable program can automate complex or seldom-used operations for the user.
- *Data sharing between programs and within a program.* A home finance application might use an Apple event to request that a communications program obtain current stock market information from an online service provider. The communications program, in turn, could return this information to the home finance application in an Apple event. Different tasks for the same program can share information in this manner.
- *Synchronization between tasks.* A task performing network I/O, for example, might send an Apple event to another task informing it that a file has been successfully sent across a network.

Mac OS 8 fully supports the programming interfaces defined by the System 7 Apple Event Manager. Mac OS 8 supports System 7 high-level events, but only the main tasks of cooperative programs can send high-level events other than Apple events. Main tasks for cooperative programs can also use the PPCBrowser mechanism and the PPC Toolbox functions, but all other tasks can use only the PPC Toolbox functions. Apple events are faster and more flexible than other high-level events or the PPC Toolbox services, which Mac OS 8 supplies only for System 7 application compatibility.

Low-Level Interprocess Communication

In addition to Apple events, you can use shared data, shared memory areas, and the Microkernel Messaging Service (which transports data between tasks) for interprocess communication. Apple events are sort of the lingua franca of Mac OS 8, permitting the operating system, programs, and scripts to communicate with each other locally and across networks according to a well-established messaging protocol. By comparison, shared data, shared memory areas, and the Microkernel Messaging Service require you to establish and follow your own conventions for using these low-level forms of interprocess communication.

Shared Data. Shared data is available to multiple tasks in the same process. As explained in Part 1 of this article, different tasks for the same program share the same memory areas for dynamic storage allocation. Tasks can store shared data in these memory areas. Two tasks in the same process, for example, can share a single set of global variables for communication and synchronization purposes.

Shared Memory Areas. A task can create a shared memory area to share its data with a task in another address space. If programs in different address spaces share a large amount of data, especially when that data is continually updated, a shared memory area is likely to be a more efficient mechanism than Apple events for distribut-

ing that data among tasks. A shared memory area can begin at the same address in various address spaces (which is useful if the tasks sharing the data refer to it by pointers), or it can reside at different addresses. A shared memory area can have different access permissions in different address spaces; for example, a program might write data into a shared memory area in its own address space but make the data read-only to programs in other address spaces, preventing other programs from corrupting the data.

Microkernel Messaging Service. The Microkernel Messaging Service provides the underlying structure for Apple events. If for some reason you can't (or prefer not to) use Apple events, the Microkernel Messaging Service is available for transporting data from one task to another, in the same process or in different processes, in the same address space or in different address spaces. The messaging service allows bidirectional data transfer so that data can be part of a message, and additional data can be returned in the reply. It's up to you to establish your own conventions for interpreting the information exchanged with this service.

Data Synchronization Among Tasks

Two or more tasks sharing information must synchronize their access to that data. Otherwise, two tasks independently making changes to the same data might corrupt its integrity. Mac OS 8 offers synchronization mechanisms to protect the data shared among tasks.

Remember that cooperatively scheduled threads don't need to use these synchronization mechanisms, because one cooperatively scheduled thread can't be preempted by another thread within the same task. You simply need to ensure that each cooperatively scheduled thread yields control only after it makes changes to any data shared by other threads.

For these synchronization mechanisms to safely protect program data, a program must observe synchronization conventions. For example, if one task holds a lock to particular data, another task must not modify the data until it has acquired the lock. (Locks and other synchronization mechanisms are described below.)

Locks. A lock is a data structure used to synchronize access to a shared resource such as the contents of memory locations. Only the task holding a lock is allowed to modify the data associated with the lock. A simple lock prevents other tasks from acquiring the lock until the task holding it has released it. A read/write lock allows one or more tasks to acquire the lock for the purpose of simultaneously reading data, but this type of lock allows no more than one task to modify the data at a time.

Microkernel Queues. A microkernel queue can be used as a synchronization mechanism or a very simple interprocess communication mechanism. One or more tasks use a microkernel queue to notify another task of some occurrence—for instance, the completion of an asynchronous operation. The task being notified examines the microkernel queue for the notification; this task may, for example, block until the notification appears. A microkernel queue can hold multiple notifications.

Communication with this mechanism takes place in one direction only; that is, the tasks writing to a microkernel queue don't receive replies from the task reading the queue.

System Notification Service. The System Notification Service allows one task to broadcast information about a change in the state of the system. Any number of other tasks can subscribe to its notifications. For example, the device driver for a display screen can use the System Notification Service to announce that the user has changed screen resolutions or bit depth. Programs relying on the resolution or color capabilities of the device can then take action based on the notification.

Atomic Operations. An atomic operation is a simple routine—such as one that increments or decrements a value, tests and sets a value, or compares and swaps values—that executes to completion; it cannot be interrupted. In Mac OS 8, these operations are implemented using instructions provided by the CPU.

Event Groups. An event group consists of bits that can be set individually or in different combinations and used to signal client tasks. You can use atomic operations and event groups to implement your own sychronization mechanisms. The microkernel, for example, uses event groups to implement locks.

Summary

A multithreaded application helps the user be more productive. For example, by performing user interface operations in one thread of execution and time-consuming network I/O operations in another thread of execution, a multithreaded application enables the user to continue working with the application without waiting for lengthy network operations to complete.

Multithreaded applications use interprocess communication to share information. For example, one task for a statistical simulation program could perform a time-consuming calculation in the background. When finished with this calculation, the task could send its result in an Apple event to the main task of the program. ♣

Tony Francis has been a technical writer in Apple's Developer Press group for ten years. He has contributed to numerous books in the Inside Macintosh *series and was lead writer for* Inside Macintosh: Imaging With QuickDraw, Inside Macintosh: Devices, *and* Advanced Color Imaging on the Mac OS. *The book* Mac OS 8 Revealed, *from which this article was adapted, will be published by Apple Press and Addison-Wesley Longman in the third quarter of 1996 (352 pages and CD-ROM: ISBN 0-201-47955-9; $34.95). You'll be able to find it in most good technical bookstores, or you can reserve your copy from Addison-Wesley by calling 800-822-6339 (from the United States) or by sending a fax to 617-942-2829 (from other locations).* Mac OS 8 Revealed *© 1996 Apple Computer, Inc. Reprinted by permission of Addison-Wesley.*

Planning a Product for Mac OS 8

You can take the following steps now to prepare your software for multithreading:

1. Consider whether your program consumes very much processing time when it's not interacting with the user. If it does, separate your code into components that perform user interface tasks, computations, and I/O operations. You can then implement these components more easily as separate threads of execution in a multithreaded program.
2. Make your existing application AppleScript-scriptable. This will prepare your application to use Apple events for interprocess communication.

Issue 26

continued from page 16

develop. And speaking of judging, we're pleased to announce that we've won top honors in the 1995 Northern California Technical Communications competition held by the Society for Technical Communication. In the Magazines category, *develop* won not only the highest-level award, Distinguished Technical Communication, but also Best of Category. It then went on to win a Merit award in the STC's 1995–96 International Technical Publications Competition.

You can check out *develop* on this month's Developer CD, at the *develop* Web site (**http//dev.info.apple.com/develop.html**), or in print if you've subscribed to *develop* through the Apple Developer Catalog. Please let us know what you think at develop@apple.com or AppleLink address DEVELOP; you're the most important judges of all. ♣

Caroline Rose
Editor, develop

CD HIGHLIGHTS

Reference Library Edition

continued from page 16

development of device drivers, including Open Firmware requirements. Part III documents the AppleTalk API. Part IV provides the manual pages ("man pages") for Apple-specific AIX commands. There is an appendix noting the differences between various keyboard layouts.

This guide is to be used in conjunction with AIX developer guides from IBM. You can find information on how to obtain the IBM AIX developer guides and tools on IBM's Web site (**http://www.ibm.com**) or by contacting an IBM representative. For specific information on relevant IBM AIX guides, see the beginning of each section of the *Developer's Reference Guide for the Apple Network Server.*

Note: This document is to be used as a reference when you develop products for AIX for the Apple Network Server. It only describes the differences between AIX for the Apple Network Server and IBM's AIX; it is not intended to fully document AIX.

QDGX Spooler Patch 1.0.2

The QuickDraw GX Print Spooler Patch fixes a problem that occurs when you try to print to AppleShare and Novell NetWare print spoolers. This patch makes changes to the QuickDraw GX extension and the LaserWriter GX printer driver. You should work with copies of the GX extension and GX driver, and then move the modified files into your system's Extensions folder.

QDGX Spooler Patch 1.0.2 replaces previous spooler patches and fixes a problem that prevented users from patching certain localized versions of QuickDraw GX.

QuickTime VR Tools 1.0b2

These QuickTime VR tools help you create QuickTime VR content. They are beta-level and unsupported by Apple DTS. Source code for the tools is available in the folder QTVR Beta-Level Tools Source.

The application Make QTVR Object takes a QuickTime movie whose frames are photographs or views of an object, and combines them into a single QuickTime VR object. The application Make QTVR Panorama takes a PICT file generated by a graphics program or a scanned wide-angle or panoramic photograph, and converts the PICT into a QuickTime VR panorama.

The QuickTime VR tools make it easy for users of computer graphics programs to create QuickTime VR content. The tools also help photographers convert photographs of an object into a QuickTime VR object, and to convert wide-angle or panoramic photos into a QuickTime VR panorama.

However, the tools do not support stitching of photos, the creation of "hot spots" on panoramas, or the combination of panoramas, objects, or both into multiple-node movies. Use the QTVR 1.0 Authoring Tools Suite (available in the Apple Developer Catalog) for these purposes.

Speech Recognition Manager

This folder contains version 1.5 of Apple's new Speech Recognition Manager. Version 1.5 of the Speech Recognition extension embodies the first officially supported developer release of the Speech Recognition Manager.

The documentation included with this release is the draft *Speech Recognition Manager,* an Adobe™ Acrobat™ file found in the SR 1.5 Documentation folder. The documentation describes version 1.5 of the Speech Recognition extension and the API and Toolbox it embodies. Input from early adopters of Apple's speech recognition technology has greatly improved the Speech Recognition Manager API, and version 1.5 contains many features that were not available in version 1.4.1 or earlier versions of the extension.

If you find bugs in the extension or problems in the documentation, please use the Apple Bug Reporter stack. Include the version information in your problem description and send it to AppleLink address APPLE.BUGS or Internet address apple.bugs@applelink.apple.com.

WASTE 1.2a6

WASTE is a text-editing library for Macintosh programmers. Designed to be a viable replacement for TextEdit, it offers several enhancements, including the ability to handle text files larger than 32K. Like TextEdit, WASTE handles multistyled text and relies on Script Manager services for drawing, measuring, and hit-testing the text, and for finding word and line breaks. The built-in support for inline input makes WASTE fully compliant with WorldScript II; however, WASTE does not support bidirectional scripts, such as Arabic.

Note: This is *not* an Apple product. It is provided on an "as is" basis. Apple Computer, Inc., is not responsible for any problems you may encounter in its use.

Alex Dosher
Developer CD Leader

Business

Feature: Former Microsoft marketing manager Cathy Harris tells you about product strategies she learned during her years working on Microsoft's PowerPoint product. Learn some useful techniques from the market leader.

Microsoft Under the Microscope—Learning From the Market Leader

By Cathy Harris, President,
A Bit Better Corporation

When people think of Microsoft, many things come to mind—power, Bill, slaves, and arrogance—and of course there are those comparisons to the Evil Empire and Star Trek's Borg. (It's with sadness that I note the lack of creativity in such comparisons; a recent refreshing exception compared Microsoft to a refrigerator's crisper drawer: It promises crispy goodies but delivers mushy goo.) But after a cathartic round of Microsoft bashing, saner folks usually come around to admitting that there are a *few* aspects of the company that are downright admirable. After all, Microsoft's profitability, market share, marketing prowess, and management team are among the best in the world.

I've noticed that many software developers are quick to point out how they're different from Microsoft. Some even use those differences to fortify enmity against Microsoft. But as the industry matures and consolidates, it's critical for developers to put Microsoft under the microscope and ask, "What can we learn from these guys?"

In this article I share some product strategies that I used while working on the PowerPoint presentation software team at Microsoft. I wouldn't go so far as to say it's the "gospel according to Microsoft," but it's some general advice that can be effectively used by any software company to create better, more profitable products.

In Praise of Competition

With Microsoft in its current top-dog position, it's easy to forget that the game they play the best is "catch-up." Ask those who've seen Microsoft in their rear view mirror—ask Philippe Kahn from Borland, or Ray Norda from Novell, or Jim Manzi, the ex-CEO of Lotus. One of the ways Microsoft got to the top was by systematically going after each competitor in their path, then beating them.

But before you buy a Bill Gates dart board, consider this: This period was not only good for Microsoft—it was good for customers too. Users of Microsoft Word, Excel, and PowerPoint all benefited enormously from those products being honed against extremely fine competitors. More recently, Microsoft has found themselves bereft of competition in several key categories, and many would argue that their recent releases are not the better for it.

The bottom line is that competition is good for everyone. It brings out the best in development teams, it challenges us to take more risks, it expands markets, and, most important, it benefits our users. Because of this competition, our users get better products, lower prices, and a wider variety of products to choose from. The key is to use your competition as a motivator to help your team create better products and marketing plans.

Like it or not, Apple and many Macintosh developers are playing catch-up today. And since nobody plays this game better than Microsoft, let's check out a few strategies in their play book to see what we can learn.

Lesson #1: Follow the Data

There's an old political adage that says the only way to fully understand an opponent's affiliations and motivation, is to "follow the money." Applying this wisdom to software development, in order to fully understand the motive behind a potential customer's purchase decision, you need to "follow the data."

Some Product Strategies That Work

Here are some product strategies that helped make Microsoft PowerPoint the best-selling graphics product in the world.

- *Follow the data.* Before you create a product specification, gather hard market data on customer demographics, purchasing patterns, usage, and all the factors that drive purchase decisions. Don't discount the importance of Information Systems (IS) groups and external opinions in these decisions.
- *Adopt the dual-platform religion.* Having consistent product versions for both the Macintosh and Windows platform is becoming a more important purchasing criterion for medium to large businesses. It's no coincidence that software companies with the highest degree of compatibility also bring in the best profits.
- *Dissect the competition.* Don't skimp on competitive evaluation. Get everyone on your product team involved with hands-on analysis of your competitors.
- *Sacrifice for market share.* Make sure your company has a consistent metric for measuring your product's success with respect to its competitors. In this maturing software market, increasing your market share is key, so do anything you can—including sacrificing short-term profits—to gain additional share points.

Classic Microsoft marketing involves a good amount of customer research, both formal and informal. Microsoft product marketing people know that they have to understand not only customer demographics, purchasing patterns, and usage, but also the factors that drive purchase decisions. By assuming you understand a user's purchase criteria without backing it up with real data, you may lead your company down the wrong path, creating products with major deficiencies.

Start by thoroughly analyzing your users' behavior. Let's say you're working on a stenography product that will help court reporters and legal secretaries work more efficiently. But what does knowing your users' titles tell you about the purchase decision? Not very much. To effectively plan your strategy, you need to fully immerse yourself in the flow of data in and out of your users' lives. Here are some essential questions that should be answered:

- What other products do your potential customers use?
- What types of people do they work with?
- How do your customers interact with others?
- Do they share files?
- What happens to the files that a person creates?
- Who else uses them?
- What sources of information do these customers draw from?

One useful technique is to create a flow chart that details your target customers' work environments and diagrams the flow of data through their work community. This diagram provides a key starting place for identifying product compatibility issues.

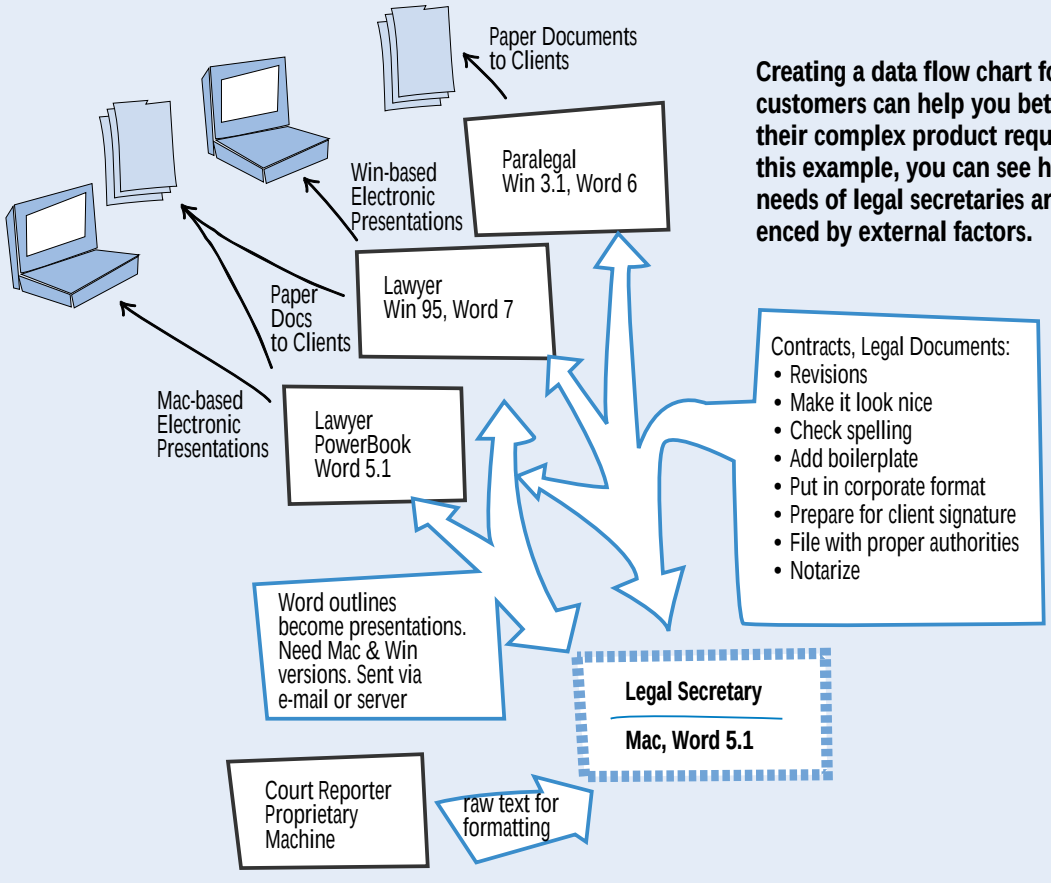
The next phase is to clearly understand who makes the purchase decisions and why. Most users will *say* that they made the purchase decision, while in fact, what they really made was the decision to spend the money at a given time. It's likely that part of the decision of *which* product to buy was made by someone or something else.

For the sake of this argument, suppose I decide to buy a home finance product. I may have heard of several different packages, but chances are, I won't buy anything until I've talked to a few key people that are affected by my decision. Ultimately, I'll probably choose Quicken because that's the product my accountant uses. If I buy Quicken, I can give electronic files to my accountant, who can then do my taxes more efficiently. Or maybe I'll choose it because that's the same product that my ad hoc technical support friend uses. The point is, more often than your customers consciously realize, they choose products based on factors beyond a feature list.

In large corporations, products are often purchased through IS or other corporate purchasing groups. While you may not like this model, it's foolish to ignore the power and sales potential of these groups.

Try creating another flow chart, this time charting your users' work environment and information flow from the perspective of an IS person. Include factors such as user product training, support, file compatibility, previous file migration/translation issues, and so on.

Data Flow Diagram for Legal Secretary



Creating a data flow chart for your target customers can help you better understand their complex product requirements. In this example, you can see how the product needs of legal secretaries are heavily influenced by external factors.

By analyzing these two flow charts, you'll learn a tremendous amount about what types of compatibility issues will become key purchase factors. These are the *silent* factors that don't show up on anyone's list of "features we need to have in your product." Many of us have provided software that meets all the feature check-off points, only to be knocked out of contention by a factor that wasn't on the list at all. The factors mentioned above start to form the most important underlying criteria behind purchase decisions. These are the factors your user community thinks are so obvious that they shouldn't even have explain them to you. Unfortunately, these are also factors that often go unnoticed, because product managers "haven't heard them come up as issues."

Lesson #2: Adopt the Dual-Platform Religion

The goal of the previous data flow analysis is to understand where purchase dollars will come from, and why. For Macintosh developers, the answers to these questions will most likely lead you to an increased understanding of the role that Windows-based products have in your users' lives. You may even find, as many companies (including Microsoft) have, that the best thing you can do for your Macintosh release is to create a good Windows version.

It's not a coincidence that the best-selling Macintosh software products also have good Windows versions and excellent cross-platform compatibility. It's a natural outgrowth of the fact that every Macintosh product's main competitor is a Windows-based product. People evaluating software decisions don't have the luxury of supporting two incompatible product standards. If Macintosh products remain islands of incompatibility, they'll be forced into extinction. But if there are good dual-platform solutions available, then Macintosh communities will be allowed to survive. Then, as IS groups realize the cost-effectiveness of Macintosh administration, repair, and training, Macintosh installations will expand.

Today, companies like Adobe, Claris, and Intuit are the ones scoring the highest compatibility marks and, not coincidentally, reaping greater profits. These companies wisely took a good look at Microsoft's approach, which at the time was using cross-platform compatibility as the key competitive hammer against Lotus, WordPerfect, and Software Publishing. Microsoft didn't invent the concept of cross-platform compatibility, but they nearly perfected this approach with their Excel and PowerPoint products.

Good dual platform development requires an almost religious drive for consistency and compatibility. Consistent interface design is the cornerstone: Macintosh and Windows versions should look and work the same, with the only differences being the standard user interface elements of each platform. And when you're faced with those laborious discussions on which keyboard shortcut to use, go for consistency across platforms using the standard of your largest platform market. This makes for hard choices and long emotional discussions, but it leads to better sales and more economical development.

Commitment to cross-platform consistency is very difficult, but it's mandatory for making world-class software.

Here's a case in point: Compare the histories of Microsoft PowerPoint and Aldus Persuasion. Persuasion, a more full-featured product than PowerPoint, developed a Windows-based version with a totally different menu structure, different keystrokes, and radically different file formats than its Macintosh peer. The only similarities between these versions were the name and the feature set.

PowerPoint, on the other hand, was a product with fewer features, but there were more similarities between the two versions: identical menus, file formats, keyboard shortcuts, fonts, and so on. Macintosh PowerPoint market share grew rapidly, despite Persuasion's feature superiority, primarily because PowerPoint was the best cross-platform solution in town. The IS folks—the people who control the presentation software purse strings—placed a high value on having a consistent cross-platform presentation solution for their respective companies. After all, employees often need to share slides, even if they own different types of computers. PowerPoint fulfilled this need. Meanwhile, the Macintosh version of PowerPoint became the Windows-based version's best weapon against PC market leaders, Harvard Graphics and Lotus Freelance, since they had no Macintosh versions.

Commitment to cross-platform consistency is very difficult, but it's mandatory for making world-class software. A high percentage of shared code is a necessary starting point, but dual-platform consistency has to extend to every part of the product team: The two versions need to have *one* product specification, *one* schedule, *one* development team, *one* testing team, and *one* marketing team. All team members should be fluent in both products—team members should have both machines and versions on their desks, and everyone should know both products equally well.

If you think this means a heck of a lot of training for people, you're right. But the only way that developers will be able to come up with good cross-platform solutions is to understand the opportunities and limitations of each approach. We all know that developers don't write good Macintosh applications until they've come to understand and embrace Macintosh technology. Well, the same goes for writing, developing, testing, and marketing good dual-platform products. These are methods that Microsoft used to ship excellent dual-platform versions of Excel and PowerPoint.

Another important requirement of good dual-platform development is "sweating the details," and there sure are lots of them. Fonts, for example, can cause a world of trouble. What happens to your nicely formatted document that uses the "Penguin Bold" font when you translate it to a Penguin-free environment? Does the system make reasonable substitutions? Is it easy to globally replace fonts? Do you ship a compatible set of fonts across both platforms? Fonts, color specifications, picture translations, file translations, and version compatibility are the stuff that customer's nightmares are made of. Make sure that you have a few extremely detail-oriented people ferreting out these issues and providing you with good solutions to these hard problems.

Lesson #3: Dissect the Competition

By the time you've identified the forces that influence product purchases, you probably have a pretty good idea of what your competition is. Obvious competitors are those that do similar things on similar platforms—Microsoft Works versus ClarisWorks, for example. It's usually pretty easy to identify these products, although sometimes you can fall into the trap of denying that you have any competition at all, because your product represents "a new category" of products. Even if that's true, people were using something to solve this problem before, and that something is your competition.

It's important to look for less obvious competitors. There are typically products or services that solve the same user problem as your product. Remember the classic mistake made by railroad companies, who believed that their only competitors were other railroads; they forgot that *transportation* was their real business, and they were subsequently clobbered by the trucking industry. These products may provide customers with partial, impractical, or bad solutions, but take a good, hard look at them nonetheless. The Microsoft Office team, for example, should take a good look at ClarisWorks.

Where should *you* look? First look at products on other platforms. Windows is an obvious one, but also check out UNIX tools, particularly if you're a producer of high-end graphics tools. Then look to completely different paradigms. For example, will phone company services eventually replace your nifty new networking product?

Last but not least, it's important to remember that if you have any installed base at all, *you* are your own competition. This means that your new product has to provide "more user value" than your existing product. Last year the Macintosh community reminded the Microsoft Word team that they do indeed have competition in the form of Word 5.1. And, as software categories reach maturity (word processing is an excellent example), it becomes increasingly difficult to get your user base to upgrade.

Once you've identified competitors, you need to sort them out. Divide them into three groups: serious competitors, minor competitors, and insignificant competitors. Now you're going to dive into some serious competitive analysis.

First, spend at least 30 to 60 minutes trying out each insignificant competitor. Record installation time, how much memory the product consumes, and system requirements. Then spend the rest of your time actually using the product to solve a real problem, all the while watching for good ideas. Every place where the product has taken a different approach from that of your product, ask yourself, "Why was it done this way?" Does the product make different assumptions about the users than you do? What is valuable about this way of thinking? Jot down only the things that you think are valuable. Don't spend time writing competitive bullet points just yet.

Next, spend two to three hours with the minor competitors, again, looking for good ideas and differences of opinion. Remember that these products all represent the efforts of a group of intelligent people who thought about the same problems you did, and—for whatever reasons—came up with different answers. There are very valuable things to learn from their hard work, and it's extremely rare that a product is so bad that it has nothing to recommend it. For each product, write down three advantages and disadvantages to the product approach.

Then, spend at least two weeks using each serious competition product for everything. If it's a drawing product, use it to do drawings, as well as status reports, schedules, memos, and anything else you can think of. While these documents may not all represent the types of things the product was designed to do, the mere exercise of using the product will expose you to its features and capabilities in a way that a structured analysis wouldn't. As you use this product, learn it as you would if you'd just spent $600 out of your own pocket. Learn the shortcuts. Go to the competitor's Web site to find their tips and tricks. Find out all the cool things you can about it. Learn to love it.

After a few weeks, you should have a thorough understanding of the differences between your product and your competitors', so you can create a thorough feature comparison matrix and advantages/disadvantages document. Now you're ready to sort the good from the bad ideas. Then think about how you can improve upon them. Get the entire product team together to demonstrate all the cool competitor features; this tactic gets your team's creative *and* competitive juices flowing.

While many companies feel that this type of competitive analysis should be done exclusively by the marketing department, you'll design a much better product if the entire development team is involved. Don't be stingy in buying copies of your competitor's software for your programmers, testers, or program managers. Competition is everyone's business. Programmers will learn things in their investigation that the marketing folks won't, and vice versa. The important thing is to understand why your competition chose to be different, and what that difference in thinking can mean to you and, most important, to your customers.

Lesson #4: Sacrifice for Market Share

A common weakness that I see in many software companies is a lack of consistency in quantifying their status with respect to their competition. People determine market leadership in different ways, but Microsoft doesn't. According to Microsoft, leadership is defined by market share, and market share is more valuable than anything else, because it's the definitive voice of the customers.

Business decisions become simpler with this market share leadership clearly articulated as the objective. At Microsoft, product success is measured by market share, and anything you can do to increase your share is a *good idea.* If you can sacrifice short term profits for market share, then do it. If you have to cannibalize one of your own products to increase market share, then do it.

Thorough user research, good dual platform execution, and a targeted analysis of your competitors will place you in an excellent market position, even before your first product goes out the door. And by following this advice, you'll be well on your way to giving your competitors a run for their market share. Next, you'll need a specific plan of whose market share you want, and how you're going to get it, but that's a topic for another day. ♣

Cathy Harris (cathyh@bitbetter.com) is president of A Bit Better Corporation, makers of Screen Beans clip art characters (http://www.bitbetter.com). Previously, Harris worked at Microsoft for six years as the manager of product planning for Microsoft's graphics business unit in California.

Listings

IT SHIPPED!

The following 140 Macintosh products were entered into the It Shipped! database between March 15 and April 15, 1996. Congratulations to all of you with new shipping products! Approximately 340 have been entered into the database since Apple reinstated the program in late 1995. Products developed first for the Mac OS platform are designated with "MAC FIRST" on the far right; products available only for Mac OS systems receive the designation "MAC ONLY."

The It Shipped! database is used by Apple employees when they prepare advertising, collateral, and white papers and when they help customers find computing solutions; it's also broadcast to key industry publications. For more information about the It Shipped! program visit the It Shipped Web Page at (**http://dev.info.apple.com/itshipped.html**).

To enter your Macintosh product in the database, you can obtain a form from the site (**http://dev.info.apple.com/thirdparty/submission.html**).

You must also send a copy of the product to Apple at this address:

Apple Computer, 1 Infinite Loop, M/S: 301-1ES, Cupertino, CA 95014, USA.

Company	Product	
4-Sight plc	4-Sight iSDN Manager	MAC FIRST
4-Sight plc	4-Sight iSDN Manager for Networks	MAC FIRST
ACA Architecture & Computers Aids	ARC+ Software for Architecture	
Active Imaging	Snapper Series of Image Capture Boards	
Adaptec Corp.	PowerDomain 2940UW	MAC ONLY
Adaptec Corp.	PowerDomain 3940UW	MAC ONLY
AG Group	NetMeter	MAC ONLY
AKTIV Software Corp.	Duppies	
Attain Corporation	IN CONTROL	MAC FIRST
AWOL Software Productions	AWOL Utilities	
BDW Software	Digital Ditto	MAC ONLY
BDW Software	Remote Control for Macs	MAC ONLY
C4SI	C4SI SD	MAC FIRST
Capilano Computing Systems	DesignWorks	MAC FIRST
Celestin Company	Apprentice	
Chancery Software	CSL Curriculum Orchestrator	MAC ONLY
CheckMark Software	CheckMark Payroll	MAC FIRST
CheckMark Software	MultiLedger	MAC ONLY
Chris W. Johnson	Chris' Puzzle	
Color Partnership	Color Synergy 1.2	
Corporate Solutions	FileMaker Pro 3.0, Part 1	MAC FIRST
Corporate Solutions	Internet Essentials	MAC FIRST
Corporate Solutions	MacManagement, System 7.5	
Cross Culture Limited	Qickworks	MAC FIRST
Cross Culture Limited	Stock Keeper	MAC FIRST
Dantz Development Corp.	DiskFit Direct	MAC ONLY
Dantz Development Corp.	DiskFit Pro	
Deneba Software	Deneba artWORKS	MAC ONLY
Deneba Software	The BigThesaurus	MAC ONLY
Deneba Software	UltraPaint	MAC ONLY
Deneba Software	Canvas	MAC FIRST
Deneba Software	Spelling Coach Professional	MAC FIRST
Dunaway Products	Signalize! RIP	MAC FIRST
Dunaway Products	SmartPort	MAC FIRST
E-magine	ProView	
Eagle Data Protection	MACIlock PRO	MAC FIRST
EduPress Publishing	Ultimate ClarisWorks Solutions 4.0	MAC FIRST
Enhance Cable Technology	SoundJack Sr. and SoundJack Jr.	MAC FIRST
Farallon Computing	Timbuktu Pro for Macintosh 2.0	MAC FIRST
FCR Software	FCRppp for Mac OS	MAC FIRST
Gallery Software	DATStudio Pro	
Griffin Technology	60 Hz Adapter	
Griffin Technology	II Series Adapter	
Griffin Technology	Mac Res Adapter	
Griffin Technology	Mac Sync Adapter	
Griffin Technology	Mac/PC Adapter	
Harmonix Limited	Primary Rate ISDN PCI Card	MAC FIRST
Haywood & Sullivan	Sullivan's Scanning Tips & Techniques on CD-ROM	
Highware	DiskGuard	
Highware	DiskGuard Remote	
Highware	DiskGuard Remote	
Highware	FileGuard	
Highware	MultiHome	
Highware	Personal Backup	
Highware	PopupFolder	
Intelligent Technologies	IntelliBots	MAC FIRST
Interstudio	Domus.Cad 9.0	MAC ONLY
Iverson Software	Collection of 99 XCMDs and XFCNs	
James Renken Software	Useless	
Jim Henson Products and Sunburst	Muppet Math	MAC ONLY
Jim Henson Products and Sunburst	The Lost Treasures of Zabidonia	MAC ONLY
Jones Digital Century	Charlton Heston's Voyage through the Bible - New	MAC FIRST

Company	Product	
Jones Digital Century	Charlton Heston's Voyage through the Bible - Old	MAC FIRST
Jones Digital Century	Jones Telecommunications and Multimedia Encyclopedia	MAC FIRST
Kachi!Soft Corp.	KACHI!Project Manager	
Lari Software	LightningDraw GX	MAC FIRST
LifeLong Software,	LifeLong Universe	MAC FIRST
Literate Software Systems	Across Lite	
LizardTech	Planet Color 1.5.8	
Logitech	Cordless MouseMan for Macintosh	
Logitech	TrackMan Live! for Macintosh	
Logitech	TrackMan Marble for Macintosh	
MacinStuff	The online MacinStuff Times	
Main Event	Scripter, the AppleScript Construction Set	MAC FIRST
Mark/Space Softworks	Communicate Lite	
Mark/Space Softworks	Mark/Space PC-ANSI Tool	
Mark/Space Softworks	Mark/Space Videotex Tool	
Mark/Space Softworks	Mark/Space ZMODEM Tool	
Mark/Space Softworks	PageNOW!	
Maui Software	TimeTracer	
Maui Software	Yank	
MegaWolf	Fenris Multiport PCI Serial Expansion	
Micro Dynamics	MARS/NT	MAC FIRST
FormMaker Software		
Microspot USA, Inc.	3D World	
Misty City Software	Grade Machine	MAC FIRST
MVP Solutions	Retrieve It! 2.0	
Natural Intelligence	Roaster	MAC ONLY
NetLOCK	NetLOCK for Macintosh	
NetManage	XoftWare for Mac OS	
NetManage	Chameleon for Mac OS with PacerTerm	
NetManage	WinSock SDK for Mac OS	
NetWings	NetWings Gateway System (NGS)	
Nisseb softwares	Dr. Jekel	
Nisseb softwares	SCDialFix XCMD	
Norman Franke	SoundApp	
OceanAtlas Software	Power OceanAtlas	MAC FIRST
Optima System	PageSpinner	
Oracle Corp	Oracle Media Objects	MAC FIRST
Pilgrim New Media	Her Heritage	

Company	Product	
Plastic Thought	3d-Active.com Vol. 1	MAC FIRST
Ra Data as	InfoPress	
Ray Sauers Associates	DragInstall 2.0	
Seapine Software	TestTrack	
Smith Analytics	Safety Engineering Assistant	
Softron Media Services	FrameDisplay	
Softron Media Services	GPICommander	
Softron Media Services	OnTheAir Studio	MAC ONLY
Softron Media Services	TVScheduler	
Sonic Desktop Software	SmartSound For Multimedia	MAC FIRST
SPARROW Corp.	Kmax	
Specular International	Specular BackBurner	
Spider Island Software	TeleFinder v5	MAC FIRST
STAZ Software	Classroom Publisher	
STAZ Software	FutureBASIC II	
Stone House Systems	GoChart	
Sunburst	Mr. Murphy's Chowder	MAC ONLY
Sunburst	Tiger's Tales	MAC ONLY
Symantec Corporation	Suitcase	MAC ONLY
SYNEX	Bar Code Pro	MAC FIRST
SYNEX	Label Press	MAC ONLY
SYNEX	MacEnvelope	MAC ONLY
System Clinic	DTP603	
TeleType	Digital Gourmet Deluxe	MAC FIRST
TeleType	Digital Gourmet for Newton	
TeleType	T-Script	
TeleType	TeleType GPS	
Tenon Intersystems	XTen	
The Henry Starr Co.	Font-o-rama!	
Toothpick Software	1000Words	
Toothpick Software	ExtraPlugs	
Tri-millennium Technology	GYPSY-2000 SCSI-to Versatec Plotter Adapter	
Water's Edge Software	SuperCDEFs	
Water's Edge Software	Tools Plus	
White Pine Software	eXodus 6.0	MAC FIRST
Wisdom Quest Multimedia	eLibrary 96	
Working Software	Spellswell 7 version 2.0	
WriteWare	Stealth Stylus Product Line	
Xerox Corp.	TextBridge Professional Edition	
Yamaha	CDR 100	
Zedd Software	Infinity	

DEVELOPER UNIVERSITY SCHEDULE

Developer University (DU) offers a broad range of Mac OS and Newton programming instruction through hands-on classes and self-paced training products. Classes are offered in Cupertino, California and through selected third-party trainers.

Advanced C++
5 Days/$1000
Classroom
July 22–26, Cupertino

Apple Events/AppleScript Programming
5 days/$1,000
Classroom, Self-Paced
August 12–16, Cupertino

Creating OpenDoc Parts
5 days/$1,500
Classroom
June 24–28, Cupertino
July 15–19, Cupertino
August 12–16, Cupertino
September 16–20, Cupertino

Intermediate Programming: 7.5 Topics
Self Paced

Introduction to PowerPC
Online

Introduction to PowerTalk
Online

Introduction to RISC Technology
Online

Macintosh Debugging Strategies & Techniques
3 days/$900
Classroom
July 8–10, Cupertino

Multimedia Development with QuickTime VR
3 days/$900
Classroom
May 21–23, Cupertino
June 18–20, Cupertino
July 16–18, Cupertino
August 13–15, Cupertino
September 17–19, Cupertino

Newton Programming: Essentials 2.0
5 days/$1,500
Classroom
May 20–24, Cupertino
June 17–21, Cupertino
July 22–26, Cupertino
September 9–13, Cupertino

Programmer's Introduction to PowerPC
Online

Programmer's Introduction to RISC and PowerPC
Self-Paced, Online

Programming with MacApp
On demand—call DU Registrar for more information

Programming with QuickDraw GX
4 days/$1200
On demand—call DU Registrar for more information

Programming with QuickDraw 3D
3 days/$900
Classroom
May 20–22, Cupertino
July 22–24, Cupertino
September 9–11, Cupertino

QuickStart Mac OS Programming
5 days/$1,500
Classroom
June 3–7, Cupertino
July 8–12, Cupertino
August 19–23, Cupertino
September 30 – October 4, Cupertino

Scripting with AppleScript
2 days/$600
Classroom
June 10–11, Cupertino
July 22–23, Cupertino
September 9–10, Cupertino

To register for a class or to get a complete course description by fax, call the Developer University Registrar at 408-974-4897.

Course descriptions can also be found electronically at the following locations:

AppleLink—Developer Support:Developer Services:Apple Developer Services:Developer Information:Developer University

Internet—**http://dev.info.apple.com/du.html**

America Online—Computing:Computing Forums:Development:Mac Development Q&A:Developer University.

APPLE INTERNET PAGE

This feature is devoted to informing you about where you can go on the Internet for online information about Apple Computer, Inc.; its products, technologies, and programs; Mac OS and Newton programming; and other subjects that pertain to the business of computer product development. It includes Internet resources from Apple Computer, as well as from other companies and people. Apple sites are designated by an Ⓐ; we can't guarantee the information in the non-Apple resources, but we think you'll find them useful. The list is alphabetized according to the name of each Web site or other type of resource; new resources are indicated with this mark: NEW

You'll find this feature particularly helpful when you view it at the Apple Directions Web page (located at http://dev.info.apple.com/appledirections/adtoc.html). There, all the names of the locations listed in this article are linked to the sites themselves; clicking the names will take you directly to the relevant Internet locations. We'll update this feature every month, based both on what Apple is doing on the Internet and on your feedback.

Know of a particularly useful Internet resource for Apple platform developers? Whether it's a Web page, a list server, an FTP site, or a newsgroup, let us know about it and we'll consider adding it to this feature next month. Send your suggestions to a.directions@applelink.apple.com.

Always Apple Ⓐ
http://always.apple.com/
Developed by a group of Apple employees in their copious free time to give loyal Apple customers a place to congregate on the Web. The site focuses on candid customer input, live chats, and good news about Apple.

Ambrosia Cafe
http://www.ambrosiasw.com/cafe.html
Another "everything you want to know about the Macintosh" page.

Apple Competitive Information Ⓐ
http://support.info.apple.com/competitive/competitive.html
Information comparing Mac OS computers favorably with (mostly) Windows 3.1–based machines.

Apple Computer Ⓐ
http://www.apple.com/
The Apple Computer home page.

Apple Developer Catalog Ⓐ
http://www.devcatalog.apple.com/
The Apple Developer Catalog is now online; check it out! It's a great way to view and order development tools, technical resources, training products and information if you're developing applications and solutions for Apple platforms.

Apple Developer Services and Products Ⓐ
http://dev.info.apple.com/
The main page for Apple Computer's developer services, including back issues of *Apple Directions.*

Apple Developers Listing
http://www.amsys.co.uk/applelinks.html
Links to the Web pages of hundreds of Apple platform developers.

Apple Directions Express List Server Ⓐ
http://dev.info.apple.com/appledirections/adexpresscurrent.html
Apple Directions Express is our biweekly e-mail digest of business news and information from Apple, sent to you over the Internet and posted at this Web site. It includes pointers—live links at our Web site—to other sources for more detailed information. Subscribe by sending e-mail to adirections@ thing1.info.apple.com. In the subject field, type the string "subscribe <your real name>".

Apple Education Ⓐ
http://www.info.apple.com/education
Use online forms located at this site to request product specifications, information about the Apple Education Series (bundled products), and technical support from Apple engineers.

Apple Europe Ⓐ
http://www.euro.apple.com/
The front door for information about Apple activities—including developer services—in Europe.

Apple Forever Ⓐ
http://www2.apple.com/appleforever/
Regular updates about the company and special communications from Apple executives.

Apple FTP Sites Ⓐ
http://dev.info.apple.com/ftpmain.html
ftp://ftp.info.euro.apple.com/Apple.Support.Area/Developer_Services
Go to these sites to download Apple software and documentation; the second site is a mirror site of the main location, maintained specifically for European developers.

Apple International Developer Services and Products Ⓐ
http://dev.info.apple.com/intl.html
Contains the current list of international Apple Developer Services locations and contacts.

Apple Internet Servers Ⓐ
http://www.apple.com/documents/otherappleservers.html
Includes lists of other Apple Web sites as well as Gopher and FTP sites.

Apple List Servers Ⓐ
The following are several pertinent Apple Internet mailing lists, and the addresses you can send messages to if you'd like to subscribe. For each one, type "subscribe <your real name>" in the body of the message you send.

Apple Information Alley Ⓐ
infoalley@lists1.info.apple.com

All Press Releases Ⓐ
pressrel@thing2.info.apple.com

Apple List Servers (continued)

Software Updates A
swupdates@thing1.info.apple.com

New Hardware A
newhdw@thing2.info.apple.com

Newton Press Releases A
newtonpr@thing1.info.apple.com

What's New on Apple Developer Web Pages A
devnew@thing1.info.apple.com

Apple Media Program A
http://www.amp.apple.com
Includes information about Apple's multimedia technologies as well as a searchable database of multimedia developers.

Apple Pacific A
http://www.info.apple.com/pacific/
Contains information about Apple offices and developer support in the Pacific region, including Japan, Australia, and Latin America.

Apple Software Licensing A
http://dev.info.apple.com/swl/swl.html
Official information on whether you need a license from Apple and how to obtain one.

Apple Solution Professionals Network (ASPN) A
http://support.info.apple.com/aboutapple/aspn.html
Download the latest directory of consultants who specialize in Macintosh solutions.

Apple Tech Info Library A
http://til.info.apple.com/til/til.html
Apple's official technical support database—updated daily—with over 12,000 articles on all aspects of Apple products, past and present.

"Ask Apple" Tech Support FAQs A
http://support.info.apple.com/askapple.faqs/askapplehome.html
Frequently asked questions about Apple systems, and their answers.

Brad's WebSTAR/MacHTTP
http://www.ape.com/webstar/
A database of all the Macintosh computer-based Web sites that the owner of this site can find—so far nearly 1,000 entries strong.

CI Labs
http://www.cilabs.org/
Provides a great deal of OpenDoc content.

Complete Conflict Compendium
http://www.islandnet.com/~quill/c3data.html
A listing of all Macintosh computer software conflicts and cures known to the site's owners.

Cult of Macintosh
http://www.utu.fi/~jsirkia/mac/
Another "everything Macintosh" compendium of information for Macintosh lovers.

DayStar Digital
http://www.daystar.com/DayStarHome2.html
Contains information about DayStar's PowerPC upgrade cards and their newly released Mac OS–compatible Genesis MP media-publishing workstation.

Development Tools A
http://devtools.apple.com/
Listings of Apple and third-party development tools and a variety of technical documentation and white papers.

Digitool (Macintosh Common Lisp)
http://www.digitool.com/
Contains information on the Macintosh Common Lisp (MCL) product line.

Dr. Gilbert F. Amelio
http://www.natsemi.com:80/profit/gil.html
The Web page of Apple's new CEO and chairman.

NEW **Electronic Publishing Risks**
http://www.poulton.com/eo-why.htm#why
Free information about the legal and financial risks of online communication, provided by Poulton Associates, which provides insurance and risk management services for U.S. clients.

Gradient—DCE for the Macintosh
http://www.gradient.com/
Contains information about Gradient's Mac-DCE product, an implementation of OSF DCE Secure Core functionality for Macintosh clients.

guideWorks
http://www.guideworks.com/
The Apple Guide home-away-from-home page.

Guy Kawasaki's EvangeList List Server A
All the good news about Apple platforms that Guy can find. For information on how to join, send an e-mail message to macway-request@solutions .apple.com for an automatic reply. (Any message will work.)

Guy Kawasaki's Semper.fi List Server A
A two-way list server that encourages communication between developers and Apple. We suggest you subscribe to the digest version; to do so, send a message to listproc@solutions.apple.com. In the body of the message, type "set semper.fi mail digest".

NEW **Happy Puppy's Macintosh Games Page**
http://happypuppy.com/games/mac/
Go to this site to find something to do with all that free time you have (or to divert you from the work you're trying to do).

Hartsook Letter
http://www.hartsook.com
Excerpts from The Hartsook Letter, written by long-time Macintosh market analyst Pieter Hartsook.

It Shipped! Ⓐ
http://dev.info.apple.com/itshipped.html
The home page for the It Shipped! program.

Key Apple Developer Relations Contacts Ⓐ
http://dev.info.apple.com/adrcontacts.html
Intended mostly for use when standard Apple feedback mechanisms aren't working.

Mac*Chat Newsletter
http://www.cts.com/browse/xxltony
An online newsletter directed primarily at Macintosh customers to help them make the best use of their Macintosh systems. To subscribe, send e-mail with the string "SUBSCRIBE MACCHAT" in the body of the message to listserv@vm.temple.edu.

MacHack
http://www.machack.com/
Find out about the annual MacHack hackers' conference.

Macintosh Advantage Ⓐ
http://www.apple.com/whymac/
Dedicated to showing why Mac OS–based systems are better than PCs running Windows 95. Go to this site for details about Apple's contest for the top Macintosh-hosted Web sites.

Macintosh Application Environment Ⓐ
http://www.mae.apple.com
Contains a sample of the Macintosh Application Environment (MAE), software that lets UNIX workstations run Macintosh applications.

Macintosh Help Wanted
http://www.memphisweb.com/mathew/default.html
http://www.memphisweb.com/nammac/default.html
Need to find programmers and others to work on developing Macintosh products? Go to these locations for help.

Macintosh PowerBook and Mobile Computing Ⓐ
http://www.info.apple.com/gomobile/
Complete information about PowerBook computers.

NEW **Macintosh Prices, France and United States**
http://www.ie2.u-psud.fr/~peirano/mac_us.html
Contains lists of French and U.S. Macintosh prices, as well as a brief history of the Macintosh computer, information on potential future Macintosh and Mac OS–compatible systems, and performance data about most Macintosh models.

Macintosh Programming Tools
http://www.astro.nwu.edu/lentz/mac/programming/tools.html
A terrific source for Apple and non-Apple Macintosh programming tools.

Macintosh Speech Recognition
http://www.vannevar.com/Mac_SR
As described by its owner, "a showcase for Mac speech recognition tips, tricks, and software."

Macintosh Vendor Directory
http://rever.nmsu.edu/elharo/faq/vendor.html
A directory of companies with products for the Macintosh computer.

MacintoshOS.com
http://www.MacintoshOS.com
A non-Apple site intended primarily for users, with news, shareware, online discussions, and a particularly useful history of Macintosh computers all the way back to the original Macintosh 128K system.

Mac OS Ⓐ
http://www.info.apple.com/macos/
Go here for the latest information on the Mac OS.

NEW **Mac OS 8**
http://www.macos.apple.com/macos8
Apple Computer's official source for technical and business information about Mac OS 8 (formerly code-named *Copland*), the next major release of the Macintosh operating system.

Mac QC Links
http://www.seapine.com/qclinks.html
Information about Macintosh quality-control software and services with links to sites where you can find tools for testing and debugging Macintosh software. To submit your product/service for listing or to request a link to your page, send e-mail to macqc@seapine.com.

MacTech Magazine
http://www.mactech.com
Contains MacTech Magazine's list of Internet locations for Mac OS developers.

Mac vs. UNIX Web Server Performance
http://www.netdreams.com/net.dreams/papers/theTest.html
Contains a server performance comparison between a Power Macintosh computer and a Sun SPARC workstation, both being used as Web servers. Guess which system wins?

MacSciTech
http://www.macscitech.org/
The home page of MacSciTech, the association for scientific/engineering/technical Macintosh users.

Metrowerks
http://www.metrowerks.com/
Find out about Metrowerks' CodeWarrior PowerPC development environment.

Nathan's Everything Macintosh Page
http://www.cs.brandeis.edu/~xray/mac.html
A virtual treasure trove of Macintosh information, as its name implies.

Natural Intelligence
http://www.natural.com/
Information about Natural Intelligence's tools and solutions, including Roaster and Roaster Professional.

Newton Ⓐ
http://dev.info.apple.com/newton
Includes information about Newton 2.0 and Newton Toolkit 1.6.

Nisus Software
http://www.nisus-soft.com/~nisus/
Click on the Trash icon at this site and see where you end up!

OpenDoc Ⓐ
http://www.opendoc.apple.com/mainpage.html
The place to go for the OpenDoc 1.0 SDK and OpenDoc sample parts.

OpenDoc Part Ideas
http://www.eng.uci.edu/~sroussey/NetVision/software/od_parts/
A repository of OpenDoc software, as well as ideas for software that could be implemented as OpenDoc components.

NEW **Part Merchant Home Page**
http://www.partmerchant.com/
An OpenDoc part storefront on the Internet, sponsored by Kantara Development, where you can post your new OpenDoc parts and make them available for downloading by customers.

PC Fairy Tales
http://www.icsi.net/~crfrank/newpcTales2.toc.html
Information to help debunk common Macintosh myths.

Pictorius
http://www.pictorius.com
Contains information about Pictorius Prograph CPX and Peregrine, its visual application client/server database programming environment.

Pippin Ⓐ
http://support.info.apple.com/pippin/
Contains technical information about designing products that will work with Apple's PowerPC processor–based, low-cost CD playback device.

Polymorphic E-zine
http://www.webcom.com/icog/polymorphic/index.html
A new online magazine about Macintosh programming, focusing in particular on Pascal and C/C++.

Power Macintosh Ⓐ
http://www.info.apple.com/powermac/powermac.html
http://www.info.apple.com/ppc/ppchome.html
Two useful sites for information about Power Macintosh computers.

PowerTalk Ⓐ
http://dev.info.apple.com/evangelism/powertalk/
Resources for PowerTalk programmers.

QuickDraw 3D Ⓐ
http://www.info.apple.com/qd3d/
Everything you need to know about QuickDraw 3D.

QuickDraw GX Ⓐ
http://www.info.apple.com/gx/gx.html
Look here for information on QuickDraw GX as well as links to other non-Apple sites.

QuickDraw GX Fan Club
http://www.ixmedia.com/quickgx/quickgx.html
http://www.ixmedia.com/quickgx/subscribe.html
The first site includes information designed to encourage the use and development of QuickDraw GX; the second provides addresses of two e-mail lists for receiving updates about QuickDraw GX.

QuickTime Ⓐ
http://quicktime.apple.com
News and technical and marketing information about QuickTime.

QuickTime Live! Ⓐ
http://live.apple.com
Apple's site for showing multimedia broadcasts of live entertainment, including images, videos, sound, and QuickTime VR.

QuickTime VR Ⓐ
http://qtvr.quicktime.apple.com
You can find samples of QuickTime VR products here, as well as information on how Apple's virtual reality technology works and how you can incorporate it into your multimedia products.

QuickTime VR–based Web Pages
http://www.bmwusa.com/ultimate/roadster/z3downloads.html
Take the new BMW Z3 roadster for a QuickTime VR test drive.

http://pathfinder.com/time/special/baseball
Shows the 1995 World Series in QuickTime VR.

http://sfasian.apple.com
View a QuickTime VR version of an exhibit of Mongolian art.

http://www.honda.com/cars/odyssey/
See QuickTime VR interiors of Honda Motor Company's new car models.

http://www.interart.net/
Real estate tours of actual, for-sale properties.

Quinn's Human Interface Subtleties
http://redback.cs.uwa.edu.au/Quinn/WWW/HumanInterfaceSubtleties.html
Lists the many human interface subtleties that continue to make the Macintosh user experience richer and easier.

NEW **Shareware.com**
http://www.shareware.com
As its name implies, a site devoted to providing shareware.

NEW **Software Unboxed**
http://www.unboxed.com
Broadcast Software's online distribution site for commercial software; you can post a special "locked" version of your software, using technology provided for free by Broadcast Software. Customers then contact you to purchase a "key" to "unlock" and download the software.

Technotes Ⓐ
http://dev.info.apple.com/technotes/Main.html
Contains all Technotes—new and old—as well as author's guidelines for contributing your own technical notes.

Third-Party Products Ⓐ
http://dev.info.apple.com/thirdparty/
Fill out the form located at this site to add your products to this list.

Ultimate Macintosh Page
http://www.freepress.com/myee/ultimate_mac.html
Contains more Mac OS information and software than you could possibly imagine exists.

User Group Connection
http://www.ugconnection.org/vendors/vendors.html
Resources and services for marketing your products to Apple's most influential and enthusiastic users: user groups.

NEW **Web Promotion Services**
http://www.computer.net/~owlseye/
An inexpensive service for promoting your Web site to a broad variety of Web indexes and search engines.

Yahoo
http://www.yahoo.com
The well-known Internet directory and search engine. Yahoo is currently trying to strengthen its listing of Macintosh-related Web pages; if you run one, or know of one, go to this site and search to see if Yahoo already includes the page you'd like to list. If not, click the Yahoo category to which your page belongs; then click the "add URL" icon at the top of the screen, and follow the instructions that appear.

Internet Resources for This Issue

News

- SPA site—http://www.spa.org
- April 1996 Strategy Mosaic—http://dev.info.apple.com/appledirections/apr96/stratmos.html
- Apple Media Program site—http://www.amp.apple.com
- QuickTime VR site—http://qtvr.quicktime.apple.com/
- Natural Intelligence site—http://www.natural.com/pages/products/roaster/index.html
- Symantec site—http://www.symantec.com/lit/dev/mactools.html
- Metrowerks site—http://www.metrowerks.com/products/discover/java.html

Technology

- *develop* site—http://dev.info.apple.com/develop.html
- IBM site—http://www.ibm.com

Business

- A Bit Better Corporation site—http://www.bitbetter.com

Apple Developer Catalog Order Information

To place an Apple Developer Catalog order from within the United States, contact Apple Developer Catalog at 800-282-2732; in Canada, call 800-637-0029. For those who need to call the U.S. 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. The Apple Developer Catalog is also available online on the Web at (**http://www.devcatalog.apple.com/**).