Inside this issue

(see page xx)

APPLE HUMAN INTERFACE designer Tom Erickson writes about the early stages of HI design.
(see page xx)

BUSINESS & MARKETING

THIS MONTH'S MARKETING feature will help you decide if there's an opportunity to sell your existing products into the education market.
(see page xx)

SUPERMAC LAUNCHED ITS VideoSpigot card at the same time Apple introduced QuickTime last December. Read about how one company successfully comarketed its product with Apple.
(see page xx)

APPLE OFFERS developers many comarketing opportunities, from coverage of the product in *Apple Intro News* to major exposure to customers and the press.
(see page xx)

THREE NEW SALES AND distribution agreements were announced by Apple over the past several weeks.

(see page xx)

Cross-Platform Development Tool Announced

By Gregg Williams
Apple Direct Staff

Last month, Symantec Corp. and Apple Computer, Inc. announced a technology that will make it easier for developers to create Macintosh- and Windows-based programs. On June 23 at PC Expo in New York, the two companies announced a development and marketing agreement to provide Bedrock, a cross-platform framework for developing cross-platform applications. The companies made no product announcements, but both pledged to provide commercial Macintosh and Windows versions of this framework in the first half of 1993, with future versions supporting other desktop platforms such as IBM OS/2, UNIX®, and Microsoft NT. On the Macintosh side, Bedrock will support the production of both PowerPC- and Motorola 680x0-based applications.

Symantec will provide the cross-platform application framework—known as Bedrock—that it is currently using internally to develop applications for Macintosh computers and Microsoft Windows-based computers. Apple will contribute engineering resources and code from MacApp 3.0, Apple's framework for creating Macintosh applications. Apple and Symantec will use Bedrock technology internally and work together to support the developer community's transition to Bedrock.

"This agreement will alter the dynamics of the personal computer software industry by decreasing development complexity and increasing developer opportunities," said Roger Heinen, senior vice president and general manager of Apple's Macintosh Software Architecture division. "We believe that Symantec, with Apple's assistance, will deliver the best cross-platform framework to software developers."

According to Gordon Eubanks, president and CEO of Symantec, "Our customers face two major problems: A growing number of applications that need to be written, and a growing number of desktop computing platforms that need support. The Bedrock framework, developed from our internal technology and enhanced by Apple's experience in object technology, will enable corporate customers and independent software developers to begin solving these problems."

An application framework delivers a common set of building blocks or "objects" that provide the basic structure for an application. The framework enables developers to create applications more quickly and easily, while providing the freedom to innovate. MacApp, Apple's framework, has successfully provided this support for Macintosh developers.

By applying the framework approach to cross-platform development, Symantec's Bedrock framework will enable software developers to maintain a single version of source code for platform-specific versions of an application. In addition to shortening the development cycle, this cross-platform framework will allow developers to create more reliable applications that are easier to localize for the international market.

Why This? Why Now? For many, multiplatform development is no longer an option—it's a necessity. The personal computing market has matured to the point that many work environments use computers from more than one platform. Many users demand products that are available for both the Macintosh and the Windows platforms. From the developer's side, the high cost of developing competitive applications further encourages them to increase sales (and their financial stability) by offering their products on multiple platforms.

Recognizing these realities, Apple entered into the Bedrock agreement with Symantec as the quickest possible way to deliver a cross-platform development environment to developers. This strategy will strengthen the financial health of existing Macintosh developers (especially those who currently use MacApp), and it will also encourage Windows-only developers to create new Macintosh applications.

The common wisdom is that cross-platform development systems encourage developers to create weak, lowest-common-denominator applications. Apple believes, however, that Bedrock will motivate developers to create strong Macintosh (and Windows) programs. By automatically implementing all the user-interface details that developers currently have to implement manually, Bedrock will allow developers to focus their time and resources on those platform-specific features that will make the largest contribution to their application.

Bedrock will also address a longstanding concern of the MacApp community: support for platform-specific features. According to Steve Weyl, manager of the Developer Tools Engineering group, "Apple recognizes that MacApp was not

always able to keep up, but we have committed the resources and management is committed to keep Bedrock up with the latest system software."

TECHNICAL DETAILS

Bedrock is written in C++ and provides an object-oriented approach to creating an application. The framework as it exists today consists of more than 150 classes, many of which handle user-interface elements like windows, dialog boxes, and menus. The Bedrock framework uses these classes to implement those parts of an application that are usually (and that need to be) implemented in a standard way—for example, buttons that react in a set manner when the user clicks them.

The need to create applications on multiple platforms is one of the strongest reasons to use object-oriented programming. Bedrock will include code on both platforms that implements the object classes. That way, C++ source code that uses these objects will compile correctly on both the Macintosh and Windows platforms, creating applications that behave equivalently.

Bedrock can be described as comprising three separate parts: the Bedrock Class Library, Bedrock Resource Information, and the Bedrock Utility Managers. Here is some information on each of these parts.

The Bedrock Class Library. The Bedrock Class Library contains over 150 classes written in standard C++. These classes include libraries of components in six areas:

- *Collections* (includes arrays, matrices, and sets)
- *Draw objects* (rectangles, ellipses, polygons)
- *Graphics tools* (fonts, pens, wallpaper)
- *Controls* (checkboxes, list boxes, radio buttons)
- *Windows* (document and clipboard windows, dialog boxes)
- *Streams* (memory and text streams, files)

Because these components are platform independent, Bedrock developers won't have to remember the implementation details of these components on each platform. For example, an object from the SelectFileDialog class will implement a get-file dialog box for both a Macintosh application and a Windows application.

Bedrock Resource Information. This part of Bedrock provides a platform-independent description of the entire visual interface and other elements, including things like view positionings, dialog boxes, screen layouts, menus, and strings. Because the data specifying the visual interface is separated from the code, it's easier to localize applications—and they can be translated to a different human language without recompilation or access to the source code. Since the resource information is platform independent, developers don't have to maintain multiple sets of resource information.

The format of Bedrock Resource Information supports and extends Apple's Rez resource-definition language, including type definitions. The compiler for this information includes a full ANSI C preprocessor, which makes recursive definitions, macros, and conditional compilation possible.

Bedrock Utility Managers. These managers (sets of utility routines) promise the Holy Grail of multiplatform development—code for manipulating data and services (or at least some of them) in a platform-independent way. Here is a description of each manager and a little about what each one does:

•*Virtual Memory Manager:* Allocates and manages memory; provides memory compaction, relocation, and page-swapping capabilities.

•*File Manager:* Provides access to the file system, including network access and name-handling from partial file specifications.

•*International Manager:* Provides complete country and language information, including support for single-, double-, and multiple-byte character sets and formatting for numbers, currency, dates, and times.

•*String Manager:* Supplies standard string functions and conversion routines, and connects to the International Manager for country-specific string manipulation.

•*Validation Manager*: Provides string validation for numbers, dates, time, and keywords, as well as range and series validation for numbers.

•*Date/Time Manager:* Connects to the International Manager to provide country- and language-specific date and time manipulation support.

Bedrock and MacApp. Apple sees Bedrock as the successor to its current object-oriented framework, MacApp, and is working closely with Symantec to assure as smooth a transition to Bedrock as possible without artificially limiting Bedrock. Apple encourages current MacApp users to continue using it. When

Bedrock becomes available, Apple will provide MacApp developers with tools and services to help them make the transition.

BEDROCK IN MID-1993

The Bedrock framework promises to be an exciting new tool that will make it possible for many developers to develop strong products in both the Macintosh and Windows platforms. Bedrock will also give developers a head start on producing fast, groundbreaking native PowerPC programs for the new PowerPC Macintosh platform. *Apple Direct* will report on future details about Bedrock as they become available.

Apple To Develop Multimedia PDAs With Toshiba

By the middle of next year, Apple expects to have two distinct kinds of personal digital assistant (PDA) products: first Newton, the pen-based electronic notepad/organizer/communicator technologies unveiled with great hoopla in May 1992, and then a line of multimedia PDAs, which Apple announced more quietly, and with fewer details, last month.

The multimedia PDAs will be jointly developed with Toshiba Corp. of Japan, which will manufacture the products. Apple and Toshiba each anticipates releasing its own version of the new technology, which is expected to be based on core software licensed from Kaleida, the Apple/IBM multimedia joint venture. The first product in the new PDA line is expected to be released in mid-1993; pricing has yet to be determined.

The two lines of personal digital assistants will complement each other, yet be quite distinct. PDAs using Newton technology will accept users' freeform note-taking, drawing, calculating, scheduling, and other input to intelligently help them record, organize, and communicate ideas and information. Multimedia PDAs will enable users to view content from a variety of sources in a variety of situations. At the time of the multimedia PDA announcement, Time Warner, Inc., announced that it is exploring programs to provide a wide spectrum of multimedia content titles for consumers.

As is the case with Newton, there is little news about developer opportunities on this second line of PDAs, although both Apple and Toshiba are interested in working with select third parties on title and content development. Developers interested in discussing their product ideas for multimedia PDAs can contact Apple's Media Products Group (MPG) by AppleLink at MPG. DEV. Apple will respond to your ideas if they fit our strategic direction for multimedia products.

Apple released no specific information about the technologies to be used for the multimedia PDAs. As our strategy becomes more clear in the coming months and as more information is available on both lines of PDAs, we'll provide details in *Apple Direct.*

The PowerPC in Your Future

Here We Grow Again

By Gregg Williams
Apple Direct Staff

When the year 2000 rolls around, Apple will be selling exciting new Macintosh computers.

"What," you may ask, "kind of hype is this? The Macintosh is already eight years old—it'll be long dead by the year 2000!"

Well, not necessarily. It's true that today's top-of-the-line Macintosh Quadra 950 will be a quaint museum piece by the year 2000, just as the original Macintosh 128K computer is today. But Apple didn't get to 1992 by selling just one product—it switched processors (three times), added color, went modular, added MultiFinder, kept improving the system software, and did numerous other things to give the Macintosh room to grow. We're about to do it again—and in a way that will provide a smooth transition for you, your products, and your customers.

Looking at the future and what Apple wants to accomplish to outpace the competition, Apple engineers decided that tomorrow's CISC (complex instruction-set computing) processor technology probably won't give them the computing "horsepower" they will need to get the job done.

As it turns out, a solution rose from the black magic of last year's agreements between Apple and IBM: Apple, IBM, and Motorola agreed to work together on a new generation of RISC (reduced instruction-set computing) processors, a project that no one company could do by itself. This new chip architecture will provide all three companies with enough computing power to last well past the year 2000.

What makes RISC the answer to our future computing needs? Speed—a lot today, and more promised for tomorrow. Today, CISC chips perform in the range of 5 to 20 integer SPECmarks (a minicomputer benchmark of

performance), while RISC chips today can do 30 to 70 integer SPECmarks. (Engineers expect the first PowerPC chip to perform above the 30-integer-SPECmark range.) By the mid-1990s, RISC chips are expected to exceed well over 100 SPECmarks (with CISC chips not making equivalent gains without dramatically more expensive parts).

RISC chips deliver “more bang for the buck”—that is, more computing power for the money—than CISC chips do. And RISC also delivers more computing power than CISC per watt of power consumed. This last fact is perhaps as compelling a reason as any for RISC because Apple (and everyone else) is looking at new markets for portable computers and handheld electronic devices, both of which must run using batteries. (As an example of the suitability of RISC processors to handheld devices, Apple is using a RISC processor, the Advanced RISC Machines 610 chip, in the forthcoming line of Newton personal digital assistants. Similarly, the PowerPC processors will be highly suited for portable and handheld PowerPC Macintosh computers.)

So what do we want all this power for, anyway? Is having a faster Macintosh important enough to make us switch to an entirely new processor? (And is it even *possible* to have a Macintosh with a non-680x0-family processor? More on that later....) No, we need the extra computing power for more than just speed. We need it to make further improvements on the human interface (especially for products destined for the consumer market) and to power new technologies (like voice recognition and pen-based computing).

The PowerPC chips will be the next step in the Macintosh computer’s second decade. For years to come, Macintosh computers will come in two “families”—Motorola 680x0-based and PowerPC-based. Before the end of this decade, though, Apple will discontinue the 680x0 Macintosh family in favor of the PowerPC Macintosh family.

One important benefit of the Apple/IBM/Motorola joint venture is that, together, the three companies have the resources to create multiple implementations of the PowerPC architecture. In fact, engineers are working on four such chip designs: low-cost desktop, portable (low-power), mainstream desktop, and high-end desktop. The first preliminary silicon of the low-cost desktop chip will be available in late 1992, and the others will come out by mid-decade.

The POWER to Be Improved. No venture is a guaranteed success, but the PowerPC certainly has a lot going for it. It’s based on IBM’s successful RISC

processor, called the RS/6000 POWER processor (used in the IBM RS/6000 line of workstations). But that's just the starting point. One objective of this Apple/IBM/Motorola joint venture is to take this successful architecture and improve it for the personal-computer, portable, and handheld-device markets. Here are some of the improvements that they have made:

- The PowerPC processors will all be single-chip processors. (Today, only one low-end chip, the RSC POWER processor, is a single-chip implementation. The *full* implementations of the POWER processor architecture currently comprise either 7 or 9 chips.) Single-chip processors are, of course, necessary for personal computers, portables, and handheld devices.
- The PowerPC design simplifies the original POWER architecture. The PowerPC omits about 40 POWER instructions, including ones that are less likely to be used in personal-computing situations, as well as others that are too "expensive" to implement on a one-chip design.
- The PowerPC design improves the POWER architecture by adding instructions that make sense for the personal-computing market. For example, the POWER instruction set includes many double-precision floating-point instructions but no single-precision ones. The PowerPC adds these instructions because the personal computer market uses single-precision operations much more frequently.

64-Bit Elbow Room. The most visionary improvement to the PowerPC architecture is that it has built-in, backward-compatible expansion to 64-bit addressing. This means that PowerPC object code written for 32-bit PowerPC chips will continue to work correctly when, years later, 64-bit PowerPC chips become available.

The current PowerPC processor designs use 32-bit addresses, which can address 4 gigabytes—approximately 4,294 *megabytes* of memory. A 4-gigabyte address space may seem large enough today, but there will inevitably come a day when it's not. (I remember an early Motorola 6800 single-board computer with its ROM operating system "safely" located at address 16,384 because its designers couldn't imagine needing to expand past 16K of contiguous memory.)

The PowerPC currently *implements* 32-bit addresses, but its *design* includes a 64-bit addressing mode and allows any program to switch it between 32- and 64-bit addressing modes. (Many computations don't need more than 32-bit

address space or operand precision, and it would be naive and wasteful to *require* a processor to run in 64-bit mode all the time.)

More importantly—*listen up, veteran Macintosh programmers*—when the first PowerPC processors that implement 64-bit addressing become available, all existing (32-bit addressing) PowerPC object code will run, unchanged, on the new processors. This is in contrast to the Macintosh, where the transition from 24- to 32-bit addressing caused plenty of problems and much rewriting of code. "*64-bit cleanliness" won't be an issue on PowerPC Macintosh computers*.

SOFTWARE: A SMOOTH TRANSITION PATH

"Well," you may say, "I'm all for progress, but I had enough problems moving my software from a 68000 to a 68030. Now you're going to replace the Motorola processors that have been in the Macintosh since Day One with something else, and you say *not to worry?*"

Don't worry. Really. Here's the bottom line on PowerPC:

• *Users will be able to run most existing, 680x0-based software at acceptable speeds on the PowerPC Macintosh computers*. They will not lose their current investment in Macintosh software.

• *There will still be a market for your existing 680x0-based software.* The PowerPC will mean new markets for you, but it won't take away existing ones. Because the PowerPC Macintosh models will run 680x0 software, each Macintosh sold will mean one more potential customer for your existing software.

• *By writing to the System 7 application programming interface, you will be able to create both 680x0 and PowerPC versions of your software.* If you follow the System 7 API and "play by the rules," you can use the same source code as the core for both the 680x0 and the PowerPC versions of your software.

• *There will be four different ways for you to create software that will run on the PowerPC.* Each path has its own advantages and disadvantages, and at least one of them will be right for you.

These four points are the only ones you really need to know. The sections that follow go into them in greater detail.

Emulating Success. How will the PowerPC Macintosh run existing 680x0 software? The shorter answer is the simplest: It will do so by emulating a

Motorola 68020 processor in software. The longer answer, however, is the more interesting one.

We believe that the PowerPC has enough horsepower to emulate the 68020 at acceptably fast (Macintosh II-class) levels. As proof of concept, Apple has demonstrated an experimental RISC computer (using a Motorola 88100 RISC chip—the PowerPC chips don't exist yet) running out-of-the-box System 7, Aldus PageMaker, Microsoft Excel, and scores of other programs at acceptably fast rates. This computer even ran the "Flying Toasters" screen saver from After Dark (not a small accomplishment by itself)!

Not content with merely "acceptable" solutions, Apple engineers came up with a way to greatly improve the simple 68020 emulation scheme. Upon examining the execution of various Macintosh programs, they found a very interesting fact: Many programs spend more time executing Macintosh system and Toolbox routines than they do executing their own code. (I'll simplify the text that follows by using the word *Toolbox* to include non-Toolbox routines that are also part of system software.) For example, when tested over a variety of tasks, PageMaker spent 88 percent of its total execution time in the Toolbox, while the numerically intensive Excel still spends about two-thirds (67 percent) of its time there.

Another interesting fact: Tests showed that, of the time spent in the Toolbox code (as described in the previous paragraph), most of it was exclusively spent in the same dozen Toolbox routines. PageMaker spent 92 percent of its Toolbox execution in these dozen routines, while Excel spent 95 percent of it there. (FYI, the top three routines—which account for over half the total time—are DrawText, EraseRgn, and Line.)

Although you might want to rewrite all the Toolbox routines in native PowerPC code, that's not practical. However, the data just given show that emulation can deliver *most* of the performance of native PowerPC code as you rewrite key Toolbox routines in native PowerPC code. Since these key routines will run blindingly fast compared to their emulated counterparts, we expect this two-level "smart" emulation scheme to be much faster than simple emulation of the 68020 instruction set.

Because of the "smart" emulation in the PowerPC Macintosh computer, your software will run acceptably on these machines without any extra work on your part. (OK, some programs—especially ones that don't follow Apple's programming guidelines—may need work, but you get the idea.) If that's true, why should you worry about PowerPC at all?

The answer is that, with some extra work, you can convert your software to run faster, have more features, and lengthen its “lifetime” on the PowerPC platform.

Start With System 7. System 7 is the software architecture that will take Macintosh software to the PowerPC platform. It will provide a single API for both 680x0 and PowerPC Macintosh models, and programs written for System 7 will be in good shape for conversion to the PowerPC platform. (More on this later.) Of course, you’ll have to do extra programming to use the new features that’ll be available on the PowerPC, but using System 7 now puts you in good shape for delivering the first PowerPC version of your product.

Four Development Paths to the PowerPC. Here are the four ways you can convert your software to the PowerPC platform:

•*“Smart” emulation.* As discussed earlier, “smart” emulation lets your existing, out-of-the-box 680x0-based programs run on a PowerPC Macintosh. However, such programs don’t take advantage of the speed and extra features that the PowerPC platform provides. That’s what the next three paths are for.

• *Binary-to-binary translation.* Believe it or not, the technology exists to take a Macintosh program in the form of a file of executable 680x0 code and translate that code, byte by byte, to a file of executable PowerPC code that does the same thing on a PowerPC Macintosh.

Echo Logic, Inc., a spinoff of AT&T Bell Laboratories, has a technology that makes binary-to-binary object-code translation possible. Their technology, called FlashPort, makes it possible to translate a working 680x0 Macintosh program to a working PowerPC program in a few weeks. (Note that this technology works on object code alone and does not require access to the program’s source code.) Echo Logic hopes that its technology will result in a final program that is, at worst, 10 percent slower (in execution time) and 10 percent larger (in file size) than a native PowerPC program.

One important aspect of binary-to-binary translation is that it allows any traditional Macintosh application to run on the PowerPC platform, regardless of the computer language(s) used to create the original application. This is especially important to companies that have applications written in Pascal, assembler, or any other non-C or non-C++ computer language. Because the first Apple-supplied compilers will be for C and C++, applications written in other languages will be at a great disadvantage when it comes to recompiling

or redesigning them as native PowerPC applications. However, with FlashPort, all applications are on equal footing.

Apple is working closely with Echo Logic to make the FlashPort technology available to developers. Look for that to happen in the second calendar quarter of 1993. We'll print more details in *Apple Direct* as they become available.

• *Recompilation.* This option is also called a *performance port* because, by recompiling your C or C++ program with a PowerPC compiler, you will get a PowerPC program that is faster and performs better, but that won't have any new features that are specific to the PowerPC platform. Also, since the resulting source code is for the PowerPC, your new program will have a relatively long "lifetime." Apple's experience with developers indicates that it will take 8 to 12 months to translate an existing Macintosh program to a "native" PowerPC equivalent.

• *Redesign.* By starting from scratch and writing a program specifically for the PowerPC platform, you will position yourself as a front runner in the new PowerPC Macintosh world. (Such a redesign effort is also called a *features port.*) Your program will be fast *and* it will have leading-edge features that will put it ahead of emulated, binary-to-binary-translated, and recompiled programs. A redesigned program will also have a very long "lifetime." Based on third-party experiences with System 7, Apple estimates that it will take 9 to 18 months to create a new PowerPC program from scratch or rework and existing one.

WHERE IN THE WORLD DO I START?

Before you wade into your first PowerPC development project, take a look at Figure 1. Traditionally, developers first port the current source code to the new platform (a performance port), then create a second version that adds new platform-specific benefits (a features port). The top two horizontal lines in Figure 1 show how this approach would work in moving a 680x0 program to the PowerPC platform.

However, the speed of doing a binary-to-binary translation changes the situation. Apple recommends that you start on a combined performance-plus-features port in early 1993, then do a binary-to-binary translation later, in the last half of 1993. Also, you will probably save some time and end up with a better-designed product if you do the performance-plus-features port in one step (rather than doing the two ports sequentially).

You probably won't finish the performance-plus-features port until sometime in mid-1994, but the binary-to-binary translation will be on the store shelves making money sometime in early 1994. (See the lower two horizontal lines of Figure 1.) Overall, this strategy gets better software into the market and gets it there earlier than the traditional approach does.

SKU Blues. It may occur to you that, once you have both 680x0 and PowerPC versions of your software, you also have a packaging problem—do you now have *two* products that will need two separate boxes?

Each such box is called a *stock keeping unit*, or *SKU*. Dealers don't like multiple SKUs because every SKU after the first one takes up space that the dealer would rather use to display and sell a completely different product. Distributors and inventory people don't like multiple SKUs because the extra SKUs have the overhead of a second product, but they don't provide any extra sales.

We are currently studying the issues associated with selling two versions of the same product. One alternative to multiple SKUs is to have one SKU and include both the 680x0 and PowerPC versions of your product in it. Normally, that means adding the cost of one or more floppy disks to each box—which also makes each unit you build more expensive. One solution that we are investigating is that of a new Installer program that would determine which Macintosh platform it's running on and add only those resources needed to run the program on that particular platform.

TOOLS AND SUPPORT

You can't create new PowerPC programs without good tools and good support, and we plan to see that you get both. Here are the details we have so far; we don't have delivery dates yet, but we will keep you informed through *Apple Direct* and other channels.

Development Systems. The first development system will be a 680x0 Macintosh running the Macintosh Programmers Workshop (MPW) under A/UX 3.0 (Apple's integration of System 7 with the UNIX® operating system). The system will run a new Apple C/C++ cross-compiler (that is, a compiler whose

output is PowerPC object code) based on compiler technology from Lucid (a well-known workstation compiler company).

The next development system will use working PowerPC Macintosh prototypes running an Apple Lucid C/C++ compiler running under MPW and PowerOpen (which is the PowerPC-based successor to A/UX 3.0, based on technology from both Apple and IBM).

Our eventual goal is to offer a lightning-fast, high-productivity development system that is totally native—that is, a system that can both compile and test code on a PowerPC Macintosh. This system will include a native Apple Lucid C/C++ compiler, and it will run under a powerful new development platform that will eventually replace MPW.

Tools, Tools, Tools. It takes more than just a compiler and a linker to make a good development platform. We plan to take the tools we use internally and develop them further to make them into powerful, reliable tools for commercial development. These tools include things like automatic-testing, code-coverage, and static-analysis programs—tools that help you build quality, performance, and reliability into your software.

One important tool is Bedrock, the development platform that Apple and Symantec will provide. (See “Cross-Platform Development Tool Announced” in the News folder) Bedrock will allow developers to work on the simultaneous creation of applications that run on the 680x0 Macintosh, PowerPC Macintosh, and Windows
platforms.

Developer Support. Support takes many forms, and we plan to provide the support you need to be productive. You need good documentation for both the development tools and the PowerPC Macintosh computers you’ll be using. We’re also planning various classes and developer events to get you up to speed on these new technologies. And you’ll continue to get useful technical and nontechnical information from *Apple Direct, develop* magazine, and the Developer CD Series.

SHARKS, RELATIONSHIPS, AND THE MACINTOSH

To modify Woody Allen’s observation about sharks (and relationships), computer platforms have to move forward constantly or they die. The orderly

evolution of the Macintosh computer to the PowerPC processor family is probably the biggest single step forward in the history of the Macintosh, but this transition also holds the most promise for the future. Sure, it's risky (no pun intended), but it's manageable—and the alternative is to stay where we are and get left behind.

Apple, IBM, and the rest of the computer industry have discovered that it's not practical to abandon a computer after, say, five years and start an entirely new computer design. People and companies have invested overwhelming amounts of time and money in the existing platform—so the attempt to prolong its life is worthwhile. Couple that with an attempt to make the platform *better,* and you have a good plan for a successful future. See you in the year 2000.

**********Getting Ready for PowerPC

Laying the groundwork for a new computer platform takes time, and though you can't sit down and write a native PowerPC program today, there *are* things that you should start doing to make the eventual conversion process manageable. Here are the most important ones:

• *Write in (or convert to) C and C++*. We recognize that many developers have written most or all of their source code in Pascal, but the universal availability of C and C++ make them the languages of choice, especially when the portability of source code to other hardware platforms is important. After much research and consultation with developers, we feel the need to focus Apple's attention on creating a development environment for the PowerPC platform based on C and C++. We are currently working on a next-generation C/C++ compiler (*not* a preprocessor) based on compiler technology from Lucid. We are also arranging for a third-party company to provide a procedural Pascal (that is, one that contains no support for MacApp or object-oriented programming) for the new platform. However, we strongly encourage you to use C or, better yet, C++ for new projects; this will ease your transition to PowerPC in 1993.

• *Align data structures.* Though the PowerPC chip will support 680x0-aligned data accesses, it often runs faster if data are stored on PowerPC word (32-bit)

and halfword (16-bit) boundaries. It's a small point, but the cumulative effect can be significant.

• *Don't depend on the 680x0 run-time model.* In general, *portability* is the name of the game. You need to minimize and isolate (or, better yet, eliminate) code that depends on things like the 680x0 stacks, the A5 world, and 680x0 code segmentation.

• *Isolate and minimize the use of low-memory globals and internal Toolbox structures.* As with the previous point, you should minimize the use of implementation details that are specific to the way that Macintosh programs work on 680x0 processors. In particular, we eventually plan to discontinue support for low-memory globals, and we will change (and hide) some of the internal data structures that the Toolbox currently uses. Remember, you can count on Toolbox and operating-system routines to act the same on the PowerPC platform, but things "underneath" those routines may change.

• *Don't intermix code and data.* As you've already discovered with both Motorola 68030- and 68040-based Macintosh computers, the size and type of caching can make programming practices that have never caused problems before, suddenly cause big problems. The PowerPC chips use even bigger caches than the 68040, so "Program defensively" is a good motto to follow.

We plan to write-protect code on the PowerPC, so embedding data in it won't work. Besides, all native PowerPC code—including interrupt handlers, callbacks, and patches—will be able to access all global and static data. In this way, you will be able to handle many of the situations that currently force you to embed data in code fragments.

• *Isolate or remove dependencies on 80-bit floating-point numbers.* You'll be better off if you can stick with double-precision (64-bit) floating-point numbers. One reason for this is that the PowerPC includes double-precision arithmetic instructions; because these are in the hardware itself, they will run faster than extended floating-point routines (which will be written in software).

A second (and even more compelling) reason for going with 64-bit arithmetic has to do with the use of "extended" (greater than 64-bit) floating-point numbers. It turns out that there are several floating-point number formats in use, with no single format being the commonly agreed-upon standard. This means that the extended floating-point format that the PowerPC Macintosh will use will probably be different from the 80-bit format that Macintosh programs currently use. On the other hand, there *is* a standard for 64-bit floating-point

numbers, so using (or converting to) them is probably the best way to avoid the problem of incompatible numeric data on the PowerPC.

• *Don't depend on 680x0 interrupt levels or supervisor mode.* PowerPC applications will run in user mode, and you won't have access to interrupt vectors. Of course, we will provide interfaces for interrupt handlers, but the era of undisciplined supervisor-mode execution will end with the PowerPC Macintosh.

Things will be different on the PowerPC, so plan for that now.

• *Be 32-bit clean.* Your program should already be 32-bit clean (that is, all addresses specified in your program must be full 32-bit addresses). The PowerPC chip also uses 32-bit addresses, and programs that assume that addresses are 24 bits wide will have serious difficulties. Since critical system resources will be located at addresses above 2 gigabytes, PowerPC system software will need *all* 32 bits of its addresses.

****

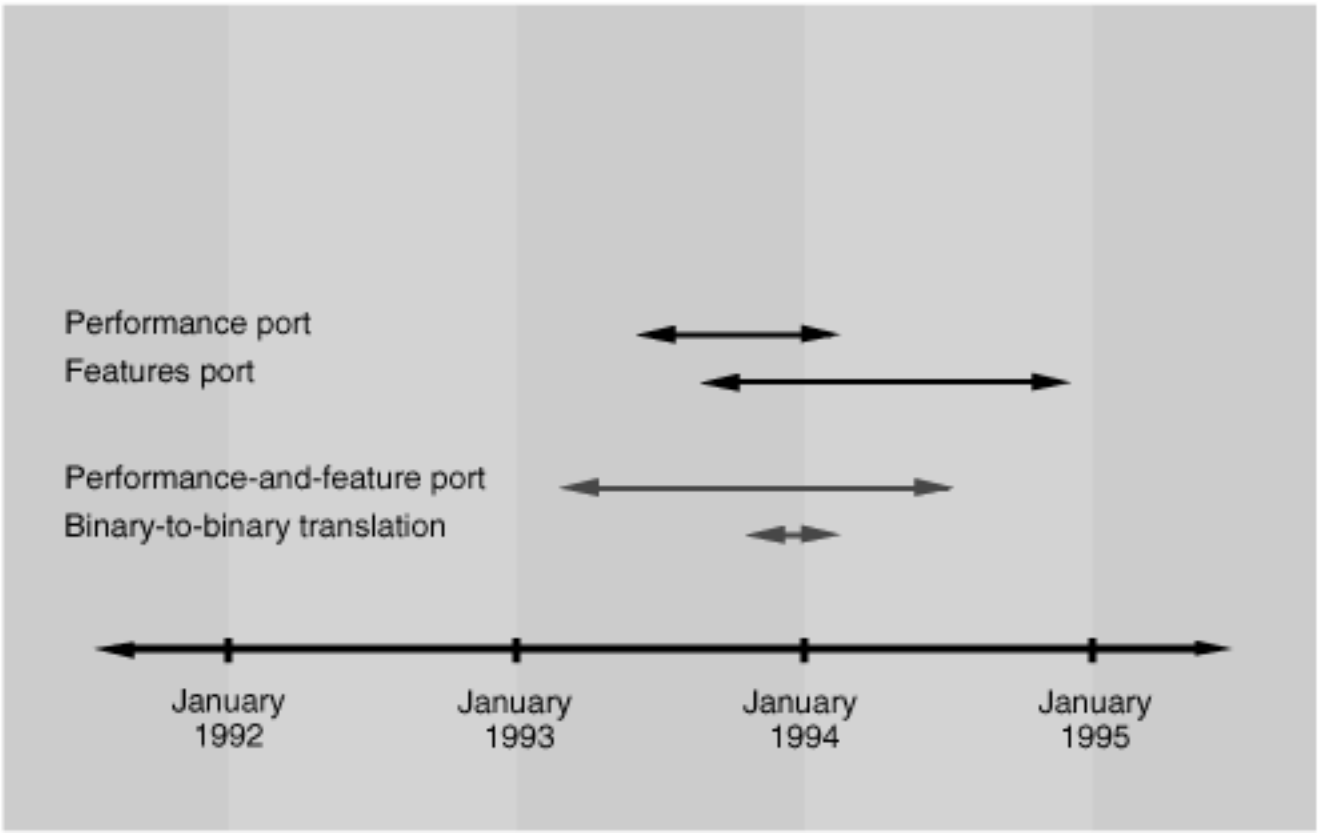


Figure 1: Making your first PowerPC applications. By doing a complete redesign (a simultaneous performance-plus-features port) and a binary-to-binary translation in parallel, you will come out with better products that are delivered earlier.

Apple's Vision for Enterprise Computing: VITAL

By Steve Rudman

When Apple announced the creation of its Enterprise Systems Division last year, it signaled Apple's commitment to providing networking solutions for large corporations. More recently, with the introduction of VITAL (Virtually Integrated Technical Architecture Lifecycle), Apple has unveiled a comprehensive framework for designing enterprise-wide information systems.

At its simplest, VITAL is a blueprint for building applications for vendor- and platform-independent information systems. Instead of a design in which host computers are the starting point and individuals the endpoint, the VITAL model prescribes systems that focus initially on the end-user. As Morris Taradalsky, the new vice president and general manager of Apple's Enterprise Systems Division, says, "VITAL is a framework for integrating the desktop into the enterprise environment."

Virtual integration refers to the fact that from the user's perspective, the enterprise system designed according to VITAL guidelines appears as a single, coherent entity, even though under the "skin" of the desktop there may be a multitude of applications, networks, and platforms. VITAL is a *technical architecture,* meaning that VITAL is a design guide for building vendor-independent information systems. The term *lifecycle* refers to VITAL's goal of providing not only a way to design new systems, but also a way of extending the life of existing systems.

Developer Opportunities. Instead of the inflexible, monolithic applications that have traditionally been used in information systems, VITAL-based systems incorporate reusable message-based modules, or *services.* Because VITAL itself does not actually "provide" the services (or applications based on them), but instead prescribes guidelines for the construction and interaction of these services, VITAL offers a wealth of opportunities for developers to build these services or their components. This article provides examples of such services and the role they play in enterprise networking.

The VITAL Model. The VITAL model consists of five interrelated logical environments. Each environment is based on a set of design principles and a set of prescribed services for building information systems.

•*Desktop Integration.* The Desktop Integration environment encompasses the human interface, personal application support, navigation, and interconnection throughout the enterprise network. This environment describes a series of reusable desktop- and server-resident computing objects and a higher-level manager that together are called *integration services*. Nearly all of the complexity of dealing with networks, mainframes, multiple user IDs, and complicated log-on procedures can be handled by these integration services instead of being "handcrafted" for each application. Examples of services within the Desktop Integration environment include database query submittal, transaction submittal, and an external data delivery service that is responsible for integrating reports and data from different sources and presenting the information in a format custom-designed for each user.

The Desktop Integration environment also provides a set of design templates that describe how to select, configure, and use the services to create enterprise systems applications.

•*Data Capture*. This environment contains the transaction-update services that create, maintain, and delete official data in operational databases. Examples of such services include a database access service, a database write service, and a transaction delivery service that provides both routing and acknowledgment of receipt. A transaction management service performs the function of a transaction monitor in a normal transaction-processing, multi-user operational system.

•*Data Access.* The Data Access environment maintains and distributes sharable corporate data for decision support, analysis, and reference purposes. The environment describes a shared data warehouse network, which consists primarily of data contained in the Data Capture operational databases, as well as a number of data access services. Major functions of these services include collecting, organizing, optimizing, delivering, and accounting for use of shared data.

There are several advantages to implementing a shared data warehouse network. Such a network defines the data access service for most users, as well as the service for transferring data from transaction-update systems to the Data Access environment. Application development is simplified and thus less

expensive and shorter. The standardized data retrieval and viewing environment prescribed by VITAL creates a stable environment within which many products can operate effectively.

Apple has already taken a major step toward providing easy access to data located in multiple, incompatible databases from desktop applications—Apple's Data Access Language product, introduced in 1990, lets Macintosh users access data from a variety of relational data sources. However, Apple intends to add to the set of database and operating-system environments accessible from a Macintosh by licensing Apple technology to appropriate developers.

•*Repository.* The Repository environment contains information about data and resources contained within the other VITAL environments. This environment consists in part of a data dictionary that defines such things as standard queries and stored procedures, standard validation criteria, authorization rules and profiles, business rules, and database structures. Repository services support the availability and administration of this information.

While Apple does not expect to develop its own repository product, we will work with developers to integrate VITAL services with key repository technologies found in IBM, Digital, Tandem, and UNIX® environments.

•*Systems Infrastructure.* The Systems Infrastructure environment of the VITAL model provides a framework that supports the other four environments. This environment supports the utility software, hardware, network, organization, security, and standards upon which the other environments are built. Representative services include an authorization service, which provides the operations needed to ensure that only authorized users can access or update data; a software license management service, which tracks the number of individuals using a particular application; a time-synchronization service, which provides the operations needed to synchronize clocks on remote computers; and a version control and synchronization service, which ensures that the activated desktop application or service is the correct version.

Learning More About VITAL. Apple has released a 50-page overview of VITAL, titled *Introduction to VITAL: Designing Information Systems for the 1990s.* You can order copies of the document from Apple by calling (800) 635-9550, extension 502. Each environment of the VITAL architecture will be described in more detail in documents to be released later this year. A separate document will provide vendor-specific information about products and

developer tools. In addition, computer-based customer training and corporate briefings will be part of the VITAL support provided by Apple.

Steve Rudman is an editor in Enterprise Systems Publications. He is the author of Introduction to VITAL: Designing Information Systems for the 1990s *and is currently editing several books about VITAL.*

1992 WWDC: What Apple Learned

If you went to this year's Worldwide Developers Conference (WWDC), you probably filled out a questionnaire to let us at Apple know how we did. We're still tabulating the results of the annual WWDC survey to report on them in one of the next issues of Apple Direct*. In the meantime,* Apple Direct *took its own informal poll of key Apple employees who participated in the WWDC to hear their thoughts about the conference and what they learned from it. This article is a sampling of the comments we received. You'll see from them that a few messages got through loud and clear, and that event planners are taking developers' feedback seriously as they plan future WWDCs.*

Michael Spindler, President and Chief Operating Officer

Apple events are great for building community and enthusiasm, and certainly the developer conference achieves this to a great extent. What we need is commitment to develop the best applications on Macintosh...first!

These people are one of Apple's most strategically valuable assets. We still have their support, but its getting tougher for them. They vote for us with their minds and hearts, but we need to help them with their wallets. Growing demand in Apple's core business, Macintosh, is Apple's most pressing issue today and we need their unwavering support to do this. They need to see great financial opportunities in writing Mac applications.

They have thousands of incredible ideas and skills, but none of it matters if they can't get attention in the marketplace. There is a lot of noise these days and its easy to get distracted. This industry will grow only as fast as we can improve our ability to solve specific problems. Apple needs to help those developers with the great ideas turn up the volume around solutions that make a real difference.

Once the market is exposed to a product, they determine what is of value and what is not. Listen to the customer.

Roger Heinen, Senior Vice President, Macintosh Software Architecture

Apple reached the next plateau at this year's WWDC. It was bigger and better than the 7.0 extravaganza last year. As always, I had a great time.

We learned several important things to help us formulate the event for next year. I agree with some developers that we should push the WWDC more toward the techno-junkie side...more technical in-depth presentations...limit ourselves to
a single strategy/marketing track each day.

I also agree that we could have handled the beta CD seeding issue differently, something better than “Roger, the ?*!@#, said we couldn’t,” or “my dog ate it.”

And I found it very interesting that we attracted so many non-U.S. developers interested in the Mac.

Thanks to all the people who worked to pull the event together. Thanks to all the developers who took their time to attend and help Apple make the Mac better. All in all, a great event!

Fred Forsyth, Senior Vice President,
Macintosh Hardware Division

I learned that many developers truly want Apple to succeed in continuing the innovation on the Macintosh platform so as to provide a better alternative than Windows.

They are also anxious to see our unit share continue to grow.

Kirk Loevner, Senior Director, Apple Developer Group

We have an incredibly dedicated and creative group of developers that continue to do great things with the Macintosh. They are demanding and always want more from us but I think they are pleased with the progress we’ve made over the past few years. There is no better forum than the WWDC for us to hear their issues, concerns, and desires. The WWDC helps us set our directions and priorities for the next year.

Rick Spitz, Senior Director, Macintosh System Software

Overall it was a very positive event, with good, open communication between Apple and our developers. I took a couple of key messages from theWWDC.

1) Developers seemed excited by Apple’s WorldScript and World-ready strategy and saw great potential for expanding markets outside the Roman world.

2) There are several areas that developers feel need more investment. The most common complaints were the Comm Toolbox and MacApp.

3) The decision not to provide a WWDC CD was not well received. Developers understood Apple's motivation for not generally releasing code before formal beta, but they still felt dissatisfied.

4) Some developers complained that too many sessions were high-level overviews or too marketing-oriented. They requested more detailed and "meaty" technical sessions.

Renee Rodrigue, Manager, Developer Events (and chief organizer of WWDC)

Due to popular demand, we're trying to move the WWDC away from Mother's Day for the future. Unfortunately, a few years ago some of us thought that avoiding other industry events in May was more important than avoiding Mother's Day and booked the conference to start just after MD for the next ten years.

We can't move the dates for next year, because no space is available. (The 1993 WWDC will be May 10–14 at the San Jose Convention Center). But we are aggressively researching dates for 1994 and beyond that will avoid Mother's Day.

A few other key points: I thought the agenda team did a great job organizing the content of the sessions and the flow of the sessions during the week—we have fewer complaints than ever regarding conflicting sessions. And we tried hard to increase the technical content. According to attendee feedback, we did keep it (mostly) at a pretty technical level, but we realize we need to keep moving in that direction (technology and code overload) in the future.

Also, it's interesting—and gratifying—that an even higher percentage of people who attended this year were repeat attendees. Almost 60 percent of the attendees had been to past WWDCs!

Shirley Stas, Manager, Developer Programs

Developers do want marketing and business information, in addition to technical information. We'll be analyzing the results of this year's WWDC marketing track to see how best to present this information in the future.

Many developers remarked on the difficulty of keeping up with the wealth of technologies coming from Apple, and in understanding how all our technologies fit together in the big picture. We in Developer Programs will work

on how to better communicate all this information to our developers over the next year.

Finally, one of the most common remarks I heard from developers at this conference, as at previous WWDCs, was that they really appreciated the chance to interact with Apple employees, and to hear about Apple's technologies and directions directly from Apple.

David Krathwohl, Manager, Developer Support Systems and Communications

If there's one point the WWDC drove home, it's the number of different developer workstyles out there. Our challenge is to provide information access methods to match these different styles. Developers want information fast, easy, and their way.

Dennis Matthews, Manager, Developer Technical Publications

We learned that developers are very eager to have *New Inside Macintosh.* We also learned that they like what we're doing with on-line documentation and want an on-line tool (display and retrieval engine) to use for their own documentation purposes. In other words, they would like us to make our product (Apple Doc Viewer) available to them. [Editor's note: *Apple will release the first three volumes of its completely rewritten* Inside Macintosh *this summer. The alpha version of Apple Doc Viewer is called Inside Macintosh Viewer.*]

Patrick Holleran, Manager, Human Interface, Collaborative Systems Development

We learned a lot from the WWDC because we tried something a little unusual. Our software, Open Collaboration Environment (O.C.E.), is in midalpha state right now, with a number of changes still anticipated. We set up several workstations and servers with our software on it in the Human Interface Lab at the convention center and let developers try things out and talk to us about the feature set, interface, etc. We ended up talking with hundreds of developers.

We are adding some features as a result of this experience, as well as considering a number of human interface modifications. So this was a really good experience.

Brian Harrison, Higher Ed Consulting Engineer, Charlotte

From an Apple field perspective, the WWDC is the time where Apple provides glimpses of the future to rekindle imaginations and provide direction for the next year (somewhat like being a water molecule in a proton magnetometer).

This year's conference left me with a good feeling that Apple has its technological future well thought out and that Macintosh will indeed have a successful second decade. RISC is real, not too far away, and the transition to RISC for our developers and users will be smooth. OOP is the programming environment of choice, especially for cross-platform development.

The most memorable scene for me was Kai Fu Lee's team explaining that Apple was not yet ready to bring pen products to bear until the market matures a bit, then showing the audience demonstrations of our R & D efforts in this area, resulting in the entire audience's standing up and yelling "Ship It!"

Most importantly, I left with a renewed feeling that all things that can possibly be invented have not yet been, and that Apple customers have a lot to look forward to in the near future as developers perform the real magic in incorporating new Macintosh capabilities into even friendlier and more powerful applications and further enhance the best computing environment in the idustry.

I can't wait for next year....

A Note From Your Editor

It's been well over a year since Apple reorganized to pursue its new business directions. For a while, those new directions were a little sketchy; we'd talked in general terms about broadly expanding the Macintosh base, becoming a big player in both the business and consumer computing markets, entering into new strategic alliances, and forging new territory with multimedia.

By now, we've provided a great many more specifics, and I think our direction can be summed up in two words: cross-platform and multiplatform, as you'll see from the stories in this month's *Apple Direct.* Our alliance with IBM and Motorola means that the Macintosh computer of the future will be a RISC-based PowerPC, which will be both the current Macintosh platform (since PowerPC will support System 7 and existing Macintosh software) and, potentially, a new Macintosh platform. (See Gregg Williams's feature article for more details.)

Our newest alliance, with Symantec (News folder), will provide developers with a cross-platform application framework so that when you develop for Macintosh, you can also develop simultaneously for Windows—and vice versa. Later releases of the framework will also make it easier for you to develop versions of your Macintosh applications for other platforms, as well.

In addition to the new PDA "platform"—Newton—that we announced last month, another alliance, this time with Toshiba, will bring to market yet another "platform," a line of multimedia PDAs. The second kind of PDA is expected to be based on technology pioneered by Kaleida, the multimedia collaboration between Apple and IBM. The promise for both PDA technologies is great, although development opportunities for them are still limited.

Then, of course, there's VITAL and the VIM interface specification, not to mention A/UX and the PowerOpen operating system we're developing with IBM (which we touched on last month) and Taligent, our object-oriented operating system collaboration with IBM (which we'll cover when there's more to say).

All these alliances and announcements aren't easy to sort out. But you'll be glad to hear we're still clearly centered on the Macintosh computer: after all, PowerPC will be a Macintosh, we're doing a lot to sell the Macintosh in new ways to new customers (News folder), and for the first time Macintosh advertising is going head to head with Windows.

I think there's a clear message developers can take away from this increasingly intricate technical mosaic: While we continue to give strong

support for Macintosh development, we’re also doing our utmost to provide you new business opportunities. But let me know what you think—send questions and comments to APPLE.DIRECT.

A Glimpse of MacHack

By Neil Day

***Hack** (hak) n. 1: A clever, though not necessarily useful, feat of programming. 2: A sick, ugly, nasty affront to programmatic aesthetics.*

Leaving San Jose for the three-day event in Ann Arbor, I was expecting something along the lines of a mini-Worldwide Developer's Conference. In certain respects MacHack does bear a passing resemblance to the WWDC—there are great sessions on new technologies. Apple events, O.C.E., RISC, and QuickDraw GX were all covered from a heavily technical perspective. Also, the folks who attend number among the best of the Macintosh programming crowd.

That's pretty much where the similarities end. MacHack is basically a programmer's playland, where 300 enthusiasts get together to worship their favorite toy and push the outside of the Macintosh development envelope. Ideas for useful (and not-so-useful) new ways of using Macintosh technology abound. The atmosphere is purely technical; marketing is considered a perversion, and even an allusion to that art will earn you a dirty look.

The soul of the conference is the Hack Contest, which traditionally occurs on Thursday at midnight. Participants spend the majority of the 48-hour period preceding the contest polishing, and in some cases whipping up, a widget to enter. Everything about the event is designed to provide the best possible environment for the Macintosh hacker. To that end, there's 24-hour-a-day access to equipment and a wide variety of development software. In the interest of facilitating round-the-clock hacking there's an ample supply of caffeine-rich beverages. For those not familiar with what distinguishes a good hack from an evil one, a session entitled "The Essence of the Hack" goes into detail on hack design, including compatibility and user interface issues.

This year's Hack Contest ran from 1:00 A.M. to almost 5:00 A.M. Fifty-five hacks were submitted, all of which were shown. Among this year's winners were:

•"IR Man" by Mike Neill and Dave Falkenburg—An interesting combination of hardware and software that allows a standard remote control unit to control either a CD-ROM player's audio or a QuickTime movie.

•“NetMouse” by Jorg Brown—For those who use two Macintoshes adjacent to one another, NetMouse gives you the ability to access the second Macintosh in exactly the same way you would a second monitor.

•“DylanTalk” by Dean Yu and Fred Monroe—DylanTalk was an interesting MacinTalkesque hack that uses a dictionary of sound samples to produce hilarious-sounding continuous speech. The resultant speech sounds a bit like Bob Dylan belting out one of his less melodic pieces.

•“BellChoir” by Kathy Brade—When you feed BellChoir a score it plays a tune, but with a twist. Each Macintosh on the network plays an individual note, something like BellChoir’s analog namesake.

•“MovieFinder” by Alex Rosenberg and Leonard Rosenthal—I understand that MovieFinder replaced Finder icons with QuickTime movies (The demo of MovieFinder unfortunately coincided with a narcoleptic episode on the part of the author, so this is purely heresay).

While the main focus of MacHack is the Hack Contest, it is by no means the only interesting session. Another highlight is the annual Stump the Experts panel, where a panel of Apple engineers is pitted against the audience. This year’s Stump the Experts had a particularly humorous ending—after a close battle and much taunting, the audience walked out on the panel as soon as it regained the advantage, effectively ending the competition.

Another MacHack tradition is the Apple Feedback Session, also known as “Bash Apple.” This is a forum for the audience to vent its frustration with Apple’s products and policies to a table full of Apple employees consisting of directors, DTS and system software engineers, Developer Events folks, and so on. Unpopular responses to questions were met with projectiles from the audience—favorites were silly string, paper streamers, and explosive caps.

Participants expressed several areas of universal concern: The expense of developing software for the Macintosh was a particularly touchy area; the prices of development tools and training were deemed prohibitive to educational users and hobbyists and painful even for small software firms. Many expressed unhappiness with Apple technical documentation, saying they had difficulty locating useful information. Most of those present wanted access to prerelease technology earlier in the development cycle, so they could get a head start on developing cutting-edge products.

In general, MacHack is a great event if you’re into programming for the Macintosh—there’s no better place to meet other folks pushing the bounds of

the technology and to come away with ideas to fuel your own development efforts.

The next MacHack Conference will take place at the Holiday Inn in Ann Arbor, Michigan, June 16-18, 1993. You can obtain registration information by writing to MacHack ’93, 1264 Bedford Rd., Grosse Point Park, MI 48230-1116.

Neil Day is Product Manager for the Developer Support Center’s Tech Notes and Sample Code.

CD Highlights, July ’92

Over the next three months you’ll see several significant changes in the Developer CD-ROM. First we will gradually abandon our HyperCard-based Developer Info Assistant (DIA) and move this resource over to the Apple DocViewer. You have already seen the preview chapters of *New Inside Macintosh* formatted with DocViewer (formerly known as Inside Mac Viewer). In August you will receive our prototype issue of *develop* magazine formatted with this tool. Please tell us how you like it!

The new viewer is designed to improve speed, readability, and printability, and offers a flexible search mechanism. Also, the DocViewer allows you to open documents from the CD without using a particular word- processing application. (For the HyperCard devotees out there, we will be leaving the familiar HyperCard version on the CD until this transition is complete.)

In August, the Macintosh Technical Notes will be reorganized to match the subject categories for *New Inside Macintosh.* The organization of Tech Notes and, eventually, sample code will map precisely to corresponding chapters in this “bible” for Macintosh developers. (See this month’s article, “New Tech Notes to Be Released in August,” in this folder, for more information.)

Your feedback is important to us as we implement these changes. Members of Developer Support Systems and Communications, the group that brings you the Developer CD, *develop, Apple Direct,* Tech Notes, and Sample Code will be visiting developers around the country to get your input on how our support tools work for you. To tell us what you think about the Developer CD, write to us by AppleLink at DEV.CD.

This Month’s CD

Meanwhile, the July issue of the Developer CD Series, *Butch ASCII and the RunTime Code,* brings you some great new applications:

Sample Code: Thanks to the efforts of Neil Day, the new Tech Note and Sample Code Product Manager, and Dave Brubeck, a summer intern from Princeton, the sample code included on this disc has been completely updated and now works on the following platforms: MPW 3.2, MPW 3.2 C, MPW 3.2 C++, MPW Object Pascal, MPW 3.2 Pascal, MPW with MacApp 1.1.1. This is just the beginning! In the months to come there will be outstanding new

additions to the sample code collection each month. Take a look at it, try it and please don't forget to send your comments.

QuickTime: At last we are able to offer you QuickTime editing tools on our CD. Straight from our QuickTime development team here at Apple comes a collection of tools, samples, applications, and third-party digitizers to help you get your QuickTime movies up and running. In the QuickTime folder, you will also find MegaMovies, an application that demonstrates the use of standard movie controllers and how they manipulate movie files. This tool tests functions that integrate QuickTime into other components of the system software.

DAL & SNA•ps Tech Notes: Apple's Enterprise Systems Division has handed us a collection of thirty new Technical Notes covering Data Access Language and SNA•ps technologies. Check 'em out.

Networking Tools: Two important networking tools have been updated this month. Look in the Networking & Communications folder for the Network Software Installer 1.2.1, which has been updated for AppleTalk version 57.04. Also see the updated Network Management Specifications (SNMP). These specifications help you implement network management features for Apple products as they are standardized by the Internet Engineering Task Force.

Among our new networking applications is Snarfer, which directs a Macintosh computer on a network to examine each AppleTalk packet it is about to write to the network. If it meets operator-defined criteria, the packet can be altered and controlled in a variety of ways.

Testing & Debugging: New contributions here include LockVol, a cdev that can instantly lock and unlock volumes when testing your applications. Also see FindLurkers, which checks a project against your checkout directory and reports any discrepancies.

QuickDraw: Color Picker is an extension to the QuickDraw graphic model that enhances the current capabilities of the Apple Color Picker. ColorSync, another extension, will integrate a color-matching architecture into the Macintosh Operating System.

Apple Events: The AE Word Services Suite allows any application that performs text processing to use any word services application (spell checker, grammar checker, hyphenator, thesaurus) as if it were built in. Try it out for yourself. Also see the AE Suite Template, a form for submitting your suites for review. Using the template will help you work through event- and object-design issues.

XTND 1.3.5: Here is the developer's kit for the XTND file-translation system. It includes the XTND engine itself as well as sample source code showing how to write XTND translators and how to add XTND support to an application.

Sharon Flowers
Developer CD-ROM Product Manager

It Shipped!

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

The following products shipped in June 1992.

Publisher	Product (Version)
U.S.	
Amadeus Enterprise	GestaltLook! (1.0)
Baudville, Inc.	Badgemaker (46217)
Brooks/Cole Publishing Company	Maple V Academic Edition
Bungie Software Products Corp.	Minotaur, The Labyrinths of Crete (1.0)
Control Data Corp.	VistaCOM Mac (3.1.1)
	Desktop/VE (3.1.1)
Earthquest Inc.	Time Treks (1.0)
	Earthquest Explores Ecology (1.0)
Famous Engineer Brand Software	digiMatic (2.0)
Heizer Software	CompileIt! (2.1a)
	ConvertIt! (1.5)
	MasterScript (1.0)
	WindowScript (1.0)
Language Systems Corp.	SoftPolish (1.0)
MicroMath Scientiic Software	MicroMath Calc (alias MMCalc) (1.0.4)
Neon Software, Inc.	NetMinder Ethernet (2.1)

Salisbury Research Group	Presidential Election Project, (1.0)
SAS Institute Inc.	JMP Design (2.0)
SF/O	easyPrint (1.0)
Software Architects, Inc.	Laptop Liberator (1.0)
South Fla Wire Service	MacPress Plus (1.0)
Synergy Software	Versaterm (4.6)
SynOptics Comm., Inc.	LattisTalk Model 3394 Router/Repeater
Thursby Software Sys.,	TSSnet (2.4)
TerraNetics	SafeDeposit (1.0)
WorksWare	Reality Works (CW) (1.0)
	DataMate (1.2)

AUSTRALIA

Communication Research Institute of Australia (CRIA)	FormsDesigner (Version 1)

CANADA

Folkstone Design Inc.	Anatomist: A Human Anatomy	CD-ROM (2.1)
Motion Works	Motion Works PROmotion (1.0)	
	Motion Works PrimeTime CD	(Vol 1)
Netstrategy	ARA-Link QM (1.02)	

Developer Activities at MacWorld Boston

Developers can take advantage of a wide range of activities offered at Apple Developer Central during next month's MacWorld Expo, to be held in Boston's World Trade Center, August 4-7.

Apple Developer Central will provide developers with free introductory technical sessions on subjects ranging from cross-platform development and PowerPC to QuickTime and O.C.E. For more detailed information on the technical sessions, send your inquiries by AppleLink to NEAN.

Developers can also view a software tools exhibit, use demos of new self-paced training products (including Macintosh Programming Fundamentals and AppleTalk for Programmers), and obtain information about APDA and the newly formed Apple Events Developer Association. (You can also enter a drawing to win a PowerBook computer!) So if you're attending MacWorld next month, be sure to stop by Apple Developer Central, located on the upper level of the trade center.

Vendor-Independent Messaging Interface Specification Available

Last month Apple, along with the other trustees of the Vendor-Independent Messaging (VIM) interface specification, released Version 1.0 of the specification. This is the first release of the nonproprietary, industry-standard programming interface designed to enable software developers to create message- and mail-enabled applications for multiple computer systems and messaging platforms. The four trustee companies—Apple, Borland International, Inc., Lotus Development Corporation, and Novell, Inc.—announced on February 4, 1992, their intent to jointly develop and support the VIM interface standard.

The VIM interface is designed to alleviate the confusion created by disparate programming interfaces on multiple computer platforms and messaging systems. Historically, for an application to be messaging- or mail-enabled, the developer has had to write a different software module to communicate with each computer system and messaging platform that the application will run on. The sheer number of systems that the application would have to accommodate has prevented most developers from incorporating messaging and mail capabilities into applications. With the VIM interface available on multiple platforms, developers will have a standard interface to access many different messaging and mail systems.

Apple, Borland, Lotus, Novell and IBM have all announced their intentions to support the VIM interface standard in future products. Apple will support the standard in future versions of System 7 and the Open Collaboration Environment. Borland will take advantage of the new interface in its Object Exchange (OBEX) technology to let its customers access a variety of mail and messaging environments. Lotus will support the VIM interface in future versions of its cc:Mail and Lotus Notes products. Novell products supporting the specification will be NetWare Global Messaging and NetWare MHS (Message Handling Service). IBM's office-product strategy will support the interface.

The specification is available at no charge. You can obtain the VIM interface specification version 1.0 by calling Apple at (408) 974-2375. Apple provides technical support for developers implementing the VIM interface specification. Instructions for obtaining technical support accompany the specification.

Announcing ADG-Japan

As part of its efforts to assist developers outside the U.S., the Apple Developer Group (ADG) has announced plans to launch ADG-Japan. Located in Tokyo, the new unit will report directly to ADG Senior Director Kirk Loevner although it will be a partnership between ADG and Apple Japan. Its goal will be to facilitate the success of the Japanese developer community in producing and selling locally developed software.

"Japan has been the fastest growing market for Apple over the past few years," Loevner said in announcing ADG-Japan. "Although Apple has enjoyed tremendous sales success in Japan, the growth in locally developed applications has not kept pace. Most of the Macintosh software products sold in Japan today are localized versions of U.S. developed software. Although this has served the local Japanese market well up to now, the lack of locally developed software will ultimately stall the future growth of the Japanese market for Apple."

The search for the ADG-Japan manager has already begun; the ADG-Japan team is expected to be made up of both Far East nationals and U.S. citizens.

Now Available From Apple

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

Apple Products

Apple Events/AppleScript Programming Tutorial
R0224LL/A
$150.00
(This Developer University tutorial will be sold at a special introductory price of $120.00.)

A/ROSE Software Kit, v. 1.1.6
M0794LL/C
$150

MacApp 3.0.1 Bundle, Disk Version
B0618LL/A
$395.00

MacApp 3.0.1 Bundle, CD ROM Version
B0619LL/A
$350.00

MacApp 3.0.1 Update, Disk Version
B0861LL/A
$250.00

MacApp 3.0.1 Update, CD ROM Version
B0862LL/A

$200.00

Macintosh Common Lisp v. 2.0
B0752LL/A
$495.00

Macintosh Common Lisp v. 2.0 Update Bundle
B0753LL/A
$200.00

MPW Development Environment v. 3.2.3
M0019LL/E
$150.00

MPW Toolbox Interfaces & Libraries v.7.0.1
M0615LL/D
$40.00

MPW Command Reference
R0253LL/A
$30.00

MPW Assembler v.3.2.3
B0236LL/E
$150.00

MPW C v. 3.2.3
B0237LL/E
$150.00

MPW Object Pascal v. 3.2.3
B0235LL/E
$150.00

MPW C Bundle v.3.2.3 (Disks)
B0031LL/I

$425.00

Macintosh Programmer's C Workshop v.3.2.3
B0502LL/C
$425.00

MPW Object Pascal Bundle v. 3.2.3 (Disks)
B0030LL/I
$425.00

MPW C & Object Pascal Bundle 3.2.3 (Disks)
B0032LL/I
$550.00
Macintosh Programmer's C & Object Pascal Workshop v.3.2.3
B0508LL/C
$550.00

MPW Development Environment Update v. 3.2.3
M0027LL/E
$30.00

MPW Assembler Update v.3.2.3
M0320LL/E
$30.00

MPW C Update v.3.2.3
M0325LL/E
$30.00

MPW Object Pascal Update v.3.2.3

M0321LL/E
$30.00

MPW C Bundle Update v.3.2.3 (Disks)

B0034LL/I
$60.00

Macintosh Programmer's C Workshop Update v.3.2.3
B0531LL/C
$60.00

MPW Object Pascal Bundle Update v. 3.2.3 (Disks)
B0033LL/I
$60.00

MPW C & Object Pascal Bundle Update 3.2.3 (Disks)
B0035LL/I
$60.00

Macintosh Programmer's C & Object Pascal Workshop Update v.3.2.3
B0231LL/I
$60.00

MPW C++ v.3.2
M0346LL/F
$175.00

MPW C/C++ Bundle v.3.2.3
B0878LL/A
$575.00

Macintosh Programmer's C/C++ Workshop v.3.2.3
B0879LL/A
$575.00

MPW C++ v.3.2 & MPW C v.3.2.3 Update

B0927LL/A
$75.00

MPW C/C++ Bundle Update v.3.2.3
B0928LL/A
$100.00

Macintosh Programmer's C/C++ Workshop Update v.3.2.3
B0929LL/A
$100.00

MacTCP v.1.1 Token Ring Extension
R0248LL/A
Free update to all who own MacTCP Developer Kit or have licensed MacTCP v.1.1.
(NOTE: This part will not be a stand-alone product; will be soft-bundled with MacTCP v.1.1.)

Personal LaserWriter NTR Printer Developer Note
R0258LL/A
$20.00

SADE v.1.3.1
M0026LL/F
$150.00

SADE v.1.3.1 Update
M0614LL/E
$30.00

XTND Developer's Kit v.1.3.5
R0096LL/B
$30.00

Third-Party Products

McCLint C Code Syntax Checker v.2.20
T0210LL/E
$149.95

McCLint C Code Syntax Checker v.2.20
McCPrint C
Code Beautifier v. 2.20
T0092LL/E
$99.95

Object Logo v.2.6
T0426LL/B
$195

(See “APDA Top-Ten Sellers” for ordering information).

APDA Hot Product July ’92

Special Introductory Offer

Order either product by August 31, 1992, and receive a free copy of Beyond the Required Suite*, the Apple events guidebook.*

Apple Events/AppleScript Programming Tutorial

Get ready for AppleScript! Learn the basics of Apple’s new scripting architecture with this self-paced disk and workbook tutorial. Using core modules from Apple Developer University’s classroom course, you will learn how to make your applications script-ready and recordable and be prepared to take advantage of the release of the AppleScript software. Includes free *Beyond the Required Suite* Apple events guidebook.
B0939LL/A Special price $120. Save $30.

Apple Events/AppleScript Special Bundle

This bundle includes the Apple Events/AppleScript Programming Tutorial, Apple Event Registry: Standard Suites, and free *Beyond the Required Suite* Apple events guidebook. Apple events provide the standard language for applications to communicate with one another.The Apple Event Registry is the dictionary of currently defined standard Apple event constructs, formatted for readability and support with summary descriptions, expanded definitions, and thorough indexing for easy look-up.
B0940LL/A Special bundle price $195. Save $40.

APDA Top-Ten Sellers

1. E.T.O. Starter Kit (includes subscription)
2. MPW C v. 3.2 bundle
3. E.T.O. Subscription/Renewal
4. Macintosh Programming Fundamentals v.1.0.1
5. QuickTime Developer's Kit v. 1.0
6. MPW C & Object Pascal bundle
7. APDA Technical Info Mailing
8. MPW C++ v. 3.1
9. MacApp 2.0.1 bundle
10. MPW Development Environment v. 3.2

Ordering Information

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

Software Development Councils Can Enhance Your Business

As an Apple developer, you want to ensure your success in the marketplace. You need answers to the “What do I need to know?”, ”Who do I contact?”, and “How can I get help?” marketing, business, and technical questions that face software development companies
trying to make better business decisions.

Now you can join a software development council in your area and get answers to your questions in a professional atmosphere for information exchange. There are a number of these key trade associations for software companies throughout the U.S. whose mission is to attract capital, customers, and employees to enhance their local area software industry. These software associations provide a wide range of programs, publications, services, and contacts to local area software companies to enable them to operate more profitably and compete more effectively in the marketplace.

The councils also provide you a setting for professional networking; getting to know other software leaders gives you the chance to exchange information and ideas and to learn how others are tackling the same tough problems you now face.

Software councils are typically open to all presidents, CEOs, and senior managers of businesses involved in the design, development, or distribution of computer software products and services. Members include large and small companies, established and start-up, and represent all major platforms. Dues vary by company size and type of membership.

Member benefits include industry directories, white papers on the industry, annual salary surveys, lobbying efforts, special interest groups, and important networking with other members. In addition, seminars of topical interest are offered, featuring executives from top hardware and software companies. Topics include development partnering, venture capital funding, worldwide marketing opportunities, antipiracy information, financial strategies, marketing information, and technical support.

Once you’re on board, you’ll have an opportunity to band together with other companies to create a unified voice for influence and change in your local software industry. And what’s more, you can act as a great influence for other developers interested in Apple development by demonstrating your own

success and by helping to start Macintosh special-interest groups where Apple information can be more widely disseminated.

The following list provides a state-by-state list of key U.S. software councils. (Apple provides this list as an information resource only and does not endorse any of these groups.)

Arizona

Arizona Software Association
Michael Cerruti,
Executive Director
7418 East Helm Drive
Scottsdale, AZ 85260
(602) 443-2780

California

Software Council of Southern California
Dawn Neja,
Executive Director
c/o Oaktree Consulting Group
21041 S. Western Avenue, #160
Torrance, CA 90501
(310) 938-7462

Software Entrepreneurs' Forum
Barbara Cass,
Executive Director
P.O. Box 61031
Palo Alto, CA 94306
(415) 854-7219

Georgia

Southeastern Software Association
Harriet Hollis, Executive Director
P.O. Box 190056
Atlanta, GA 30319

(312) 938-5052

Illinois

Chicago High Tech Association
Ken Boyce,
Executive Director
53 West Jackson, Suite 1634
Chicago, IL 60604
(312) 939-5355

Illinois Software Association
Richard A. Reck,
President
KPMG Peat Marwick
303 E. Wacker Drive
Chicago, IL 60601
(312) 938-5052

Indiana

Indiana Software Entrepreneurs Forum
Jane Barto, Vice President
One North Capital
Suite 1275
Indianapolis, IN 46204
(317) 264-2820

Massachusetts

Massachusetts Computer Software Council
Joyce Plotkin,
Executive Director
581 Boylston Street
Boston, MS 02116
(617) 437-0600

Minnesota

Minnesota Software Association
Tom Kolbo, President
P.O. Box 5521
Hopkins, MN 55343
(612) 941-9337

New Mexico

Software Entrepreneurs and Developers Association
Karl Smith, President
1009 Bradbury Court, SE
Alburquerque, NM 87106
(505) 272-7522

New York

Capital Region Technology Development Council
New York State Software Conference
Raymond Piascik,
Executive Director
P.O. Box 1907
Albany, NY 12201
(518) 445-2230

Oregon

Software Association of Oregon
Ken Maddox,
Executive Director
19500 Gibbs Drive, Suite 110
Beaverton, OR 97006
(503) 690-1395

Pennsylvania

Delaware Valley Application Development Center

Alice Crowell,
Executive Director
University City Science Center
3624 Market Street
Philadelphia, PA 19104
(215) 387-2255

Pittsburg High Technology Council
Timothy Parks,
Executive Director
4516 Henry Street
Pittsburg, PA 15213
(412) 687-2700

Texas

Houston Software Association
Byron Zaner
c/o Power Line
11140 Westheimer
Suite 300

Houston, TX 77042
(713) 780-2826

Washington

Washington Software Association
Julie Schaeffer,
Executive Director
18804 North Creek Parkway,
Suite 112
Bothell, WA 98011
(206) 483-3323

West Virginia

West Virginia Software Valley Institute, Inc.
Ritchie Ireland,
Chairman, Board of Government
President, Bell Atlantic Knowledge Systems, Inc.
1500 MacCorkie Avenue, S.E.
Charleston, WV 25314
(304) 344-6961

New Tech Notes to Be Released in August

Apple Tech Notes will employ a new organizational scheme starting in August. The new format, which combines Tech Notes with the previously separate Q&As, will make it easier for you to find the technical information you need.

The new format eliminates the numbering scheme and organizes the notes according to the kind of technology and subject matter they document. The new format will not alter note contents, but simply make them easier to find and access.

The notes will be arranged according to several general technical categories, such as Macintosh, Apple II, Enterprise Systems, RISC, and Consumer Products, so that Tech Notes about a particular Apple technology can be accessed more easily.

The existing 300-plus Macintosh Technical Notes have been reorganized into the same categories used by the forthcoming *New Inside Macintosh*, the complete rewrite of *Inside Macintosh*, *Volumes I-VI*, to be released starting this summer. The categories are Platforms & Tools (PT), Communications (CM), Networking (NW), Macintosh Toolbox (TB), QuickTime (QT), Interapplication Communications (IC), Files (FL), Memory (ME), Processes (PS), Operating System Utilities (OS), Hardware (HW), Overview (OV), Devices (DV), Imaging (QD), and Text (TX). (The August *Apple Direct* will contain details about the release of *New Inside Macintosh.*)

Instead of being designated by a number followed by the Tech Note name, each of the newly formatted notes has a title comprising a name and a technical designation (as shown in Figure 1 below). The technical designation follows the convention *T.D.C.ItemName*, where *T* is the general technical category (such as Macintosh, Apple II, or RISC), *DC* is the documentation category (the *New Inside Macintosh* categories listed earlier), and *ItemName* is a very short description of the note's contents.

For example, under the new naming scheme, former "Macintosh Tech Note #55: Drawing Icons" becomes, simply, "Drawing Icons" and receives the designation "M.QD.IconDrawing" (where *M* means *Macintosh* and *QD* means *Imaging*). "Tech Note #226: Moving Your Cat" is retitled "Moving Your Cat" and carries the designation "M.FL.PBCatMove" (where *M* means *Macintosh* and *FL* means *Files*).

Tech Notes will also now include Q&As that previously were available separately on the Developer CD and AppleLink. The Q&As will be ordered according to the general technical categories and *New Inside Macintosh* categories and will be updated monthly. The same organizational scheme will also be used for DTS Sample Code and Snippets.

The new release of Tech Notes introduces another improvement: You'll be able to print the new Tech Notes on both the 8 1/2 x 11–inch paper used in the U.S. and the A4 paper that is standard elsewhere.

Material from many previously released Macintosh Tech Notes will be included in the *New Inside Macintosh*. As *New Inside Macintosh* volumes are made available, Tech Notes containing material included in the new volumes will be eliminated so that each piece of technical information is covered by only one document.

The new Tech Notes will be available beginning in August on the Developer CD, on AppleLink (path—Developer Support:Developer Services:Technical Documentation), at the Internet address ftp.apple.com., and through APDA. (For information on ordering APDA products, see "APDA Top-Ten Sellers")

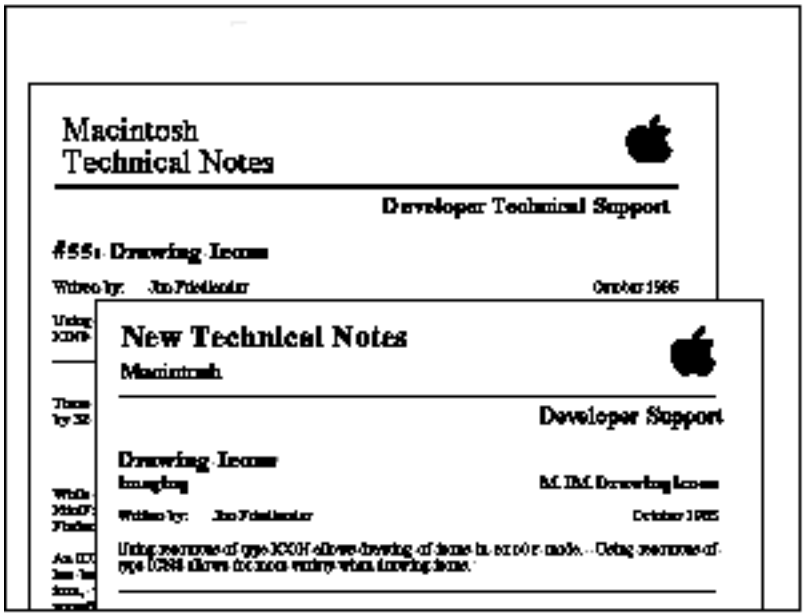


Figure 1: The old Tech Note format (behind) and the new format (on top). Note the new naming scheme.

Stories & Vignettes as Design Tools:
How I Approach The Early Stages of Design

By Tom Erickson

Every year I serve as a mentor for a couple of undergraduate courses in interface design. Typically, students choose some real-world domain, interview people to discover how they do things and what problems they face, and then design an interface to address those problems. In addition to the fun of being exposed to the wide variety of design problems that serve as grist for the classes, I also find that interacting with students gives me new insights on how I approach design problems.

Last week, a student sent me the following message:

> Subject: Individual Fieldwork Analysis
>
> I have completed my first interview and am
> attempting to categorize and analyze my data
> before my follow-up which is later
> this afternoon.
>
> My problem is that I still don't understand
> what "categories" are and what kind of
> analysis to run (or even what analysis means
> in this context).
>
> What is my next step? What kind of categories
> am I looking for? What does analysis mean???
>
> —xxx

This message evoked immediate sympathy in me. "The poor guy doesn't have a chance," I thought. "He's just spent an hour talking with someone who has a rich store of anecdotes, experiences, and opinions, and he thinks he has to

analyze it and break it apart into nice neat categories. He'll probably end up throwing out almost everything of value. Why on earth is he approaching interface design like that?"

The reason he's approaching it that way is because lots of interface designers—me included—have told him that's how they do it. Certainly, whenever I talk about how to do interface design, I describe user studies and task analyses and prototypes and user testing, and present them as steps in a fairly well-structured process, with analyses and diagrams and categories.

The truth of the matter is that all of this is just a story I've made up, after the fact, to explain things in a respectable fashion to people who don't do interface design. Unfortunately, this isn't the best story to tell people who are actually going to go out and try to do it. It leaves out a lot of the fuzzier, informal methods that never make it onto the overheads with their bulleted lists and tidy flow diagrams.

I'd like to tell you how I actually approach a design problem during the earliest stages of the process: what goes on inside my head, what makes my ears prick up during an interview, what clues me into the fact that I'm headed in a profitable direction, what makes me say "aha!"

A few warnings: First, writing this down is likely to make the process seem more structured than it really is. Interface design is inherently fuzzy stuff, which is a strength: It is a mistake to be too formal in the early stages of design. Second, my approach may appear idiosyncratic; it's intimately entwined with my particular inclinations. I hope that revealing my own nonformal methods will help others discover approaches that are suited to them. Finally, this should not be taken as a rejection of more formal methods: controlled observation, user testing, and laboratory studies all play important roles in interface design.

COLLECTING PERSONAL STORIES

Good design proceeds from understanding people's deep desires. One of my methods for doing this is collecting stories and generating vignettes.

I almost always begin by talking to users. My principle goal is to collect stories that people like to tell. I believe that the stories people tell about what they do and how they do it contain information vital to designing good interfaces. Stories reveal what people like about their work, what they hate about it, what works well, what sorts of things are real problems. People's stories capture a little bit of truth about their lives, their coworkers, and their culture.

For example, there's the near-classic story that emerged from one developer's effort to develop a Japanese version of a very successful spreadsheet package. It so happens that in Japan the current year is the number of years since the beginning of the emperor's reign. (It's as if the calendar begins anew with the coronation of each new emperor.)

Since the spreadsheet development effort was occuring during the reign of the then-elderly Emperor Hirohito, the developer—always considering the users' needs—decided that it would be a good idea to add a way to reset the base year. After all, with the emperor getting on in years, it would be necessary to start numbering years from the beginning of the new emperor's reign.

Fortunately, the developer tried the product out on users before shipping. It turns out that, according to long tradition, the emperor is considered immortal, and even though everyone presumably knows differently, only an ignorant *gaijin* would be so rude as to directly refer to the emperor's anticipated demise. Needless to say, the date-resetting feature was left out of the software.

I like this story because it captures a bit of the alien nature of a foreign culture and it emphasizes the importance of trying things out on users even when a product has been thoughtfully designed.

I judge the "goodness" of a story by telling it to other people, and seeing how much they nod or laugh as they listen. When they hear a "good" story, often listeners will say, "Oh yeah, something like that happened to me…" and will tell me their own version of that story to add to my collection. As a consequence of retelling the Hirohito story, I now have a collection of stories from Great Britain, France, Germany, and the Middle East that illustrate surprising cultural differences.

GENERATING VIGNETTES

Next, I use these personal stories to make up my own, mostly fictional stories. Essentially, after listening to lots of personal stories, I begin to recognize common themes and events, and gradually formulate one or more "metastories," which state what the personal stories I've heard have in common with each other. I'll call this type of story a *vignette* to distinguish it from the personal story. Vignettes focus on a particular circumstance or situation, and try to capture some of the recurring characteristics found across stories.

One of my favorites is the “Guilt Pile” vignette, which I put together after doing a series of interviews with office workers about how they organize information in their work environments.

The Guilt Pile Vignette: Almost everyone can identify with this:

You receive something that you feel you ought to read but, because it’s not vitally important, you don’t want to read it immediately.

Instead, you toss it on a pile of documents that probably sits on a work surface in a corner of your office. You feel good. Things are under control, and with a minimal investment of time!

The guilt pile gets higher and higher. It begins to look like there’s literally a mountain of stuff you ought to read. You begin to feel pretty uncomfortable.

Provoked by the pile’s height, you sort through it, discarding articles that no longer seem interesting, perhaps selecting one or two to read. The winnowed pile—now of a much more manageable size—is put back in its place. You feel good. Things are under control again.

I like this vignette because it not only explains how people use piles, but why they use piles. It captured something that struck me: almost all the people I interviewed were embarrassed at the state of their offices and at how badly they managed information. People felt good at the times when they had managed to sort through everything and organize it, no matter how casually.

I think the wish to feel good and on top of things rather than feeling bad and out of control is a “deep desire.” Most people can empathize with these feelings, and anyone who can come up with a product that enables folks to do a better job of satisfying such deep desires is going to make a mint.

From Stories and Vignettes to Scenarios and Prototypes

The ultimate goal is for the features of both personal stories and vignettes to help you create scenarios that drive the design of interface prototypes. Scenarios are essentially scripts that describe how users will perform some task by using the functionality and interface being designed. Designers typically create scenarios to ensure that their prototypes provide enough functionality to allow a user to accomplish some meaningful task.

Next month I’ll describe the next phases of the design process, staying with the example of guilt piles. I’ll describe a piles scenario and some prototypes of

electronic piles developed by a few human interface people at Apple's Advanced Technology Group.

THE MORAL AND A CAUTION

Stories and vignettes can be powerful design tools because they are an excellent way of explaining the reasons for a design feature to others; they can be engaging, compelling, and memorable. However, stories—even those directly from users—and the vignettes I generate have a large component of fiction. They're like pearls, initially generated by an irritating grain of truth. Our memories embroider, distort, and otherwise modify our stories so they are more entertaining and more in line with our beliefs. Thus, a designer must supplement the use of stories with other techniques, such as careful observation, interviews, and expert informants' input. Nevertheless, there is no approach to collecting knowledge in the early stages of design that is more productive, more thought provoking, or more fun.

Tom Erickson is an interface analyst and designer in Apple's Advanced Technology Human Interface Group, where he designs and prototypes interfaces for future technologies and applications. Among his accomplishments at Apple was designing the System 7 Publish and Subscribe metaphor.

GetNextEvent

The " **" indicates the trade shows/events at which Apple Computer, Inc. is scheduled to exhibit as of press time. This list may be incomplete. If you have information about a show that you want listed here, contact Developer Technical Communications, 20525 Mariani Avenue, Mail Stop 75-3B, Cupertino, CA 95014. For further information, check the Events folder on AppleLink (path—3rd Party Connection:Events).

July 14 through 16
****Mactivity '92**
Santa Clara, CA
Contact: Winehouse Computer Company
AppleLink: Mactivity
(408) 354-2500

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

July 15 through 17
****Apple Enterprise Computing Conference**
Anaheim, CA
Contact: Rhonda Rekieta
AppleLink: REKIETA.R
(408) 974-0103

July 21 through 23
****Object World**
San Francisco, CA
Contact: Tom Chavez
AppleLink: TOM.CHAVEZ
(408) 974-5718

July 26 through 31
****SIGGRAPH**
Chicago, IL
Contact: ACM
(212) 869-7440

July 28 through 30
****Apple Enterprise Computing Conference**
Dallas, TX
Contact: Rhonda Rekieta
AppleLink: REKIETA.R
(408) 974-0103

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

August 6 through 9
****American Bar Association**
San Francisco, CA
Contact: ABA
(312) 988-5880

August 12 through 14
****Apple Enterprise Computing Conference**
Boston, MA
Contact: Rhonda Rekieta
AppleLink: REKIETA.R
(408) 974-0103

September 8 through 12
****Orbit,** Basel
Contact: Kathrin Mäder
Switzerland 1 832 81 11

Fax: Switzerland 1 830 63 06

September 13 through 17
****IOUW - Int'l ORACLE Users Week Conf.**
San Francisco, CA
Contact: Oracle Corporation
(415) 506-7000

September 22 through 24
****UNIX Expo**, New York, NY
Contact: Bruno Blenheim, Inc.
(201) 346-1400

September 22 through 25
****Seybold**
San Francisco, CA
Contact: Seybold Seminars
(213) 457-5850

September 23 through 30
Bureau 92, Brussels, Belgium
Contact: BKK
02 7627183
Fax: 02 7629434

September 29 through October 1
CAD CAM Show
De Hallen, Kortrijk, Belgium

Contact: De Hallen
056 204000
Fax: 056 217930

September 30 through October 3
****SPA,** Washington, D.C.
Contact: The Conf. Dept. of SPA

(202) 452-1600

October 19 through 21
National AEC Expo - Arch., Eng., Const.
San Francisco, CA
Contact: Expoconsul Int'l
(609) 987-9400

October 21 through 23
****Winworld**, Zurich Window Exhibition
Contact: Kathrin Mäder
Switzerland 1 832 81 11
Fax: Switzerland 1 830 63 06

November 8 through 11
****MacIS USA**
New Orleans, LA
Contact: Jerry Starr
AppleLink: JERRY.STARR
(408) 974-3836

Cross Marketing Educational Products

Emerging Learning Market Creates Opportunities

by Janine Firpo,
Apple Computer, Inc.

There are growing opportunities for developers to sell existing products—those originally meant for other markets—into education. Likewise, education developers can cross-market to consumers or small businesses. And it's not necessarily as difficult and expensive as it might seem.

With the right kinds of products, positioning, pricing, packaging, and other considerations, you may be able to meet the education needs of several markets with a minimum of development fuss or other product changes.

To help you decide if there is a cross-marketing opportunity for your products, this article describes the size and current needs of four merging markets: K-12 (kindergarten through twelfth grade), higher education (post high-school), consumer, and small business. Next month's marketing feature will focus on the steps you can take to move your product from one market to another.

THE "LEARNING" MARKET

Have you heard this one? A time traveller from 200 years ago was amazed by all the changes he saw in the world today. The only place he felt at home was in educational institutions—because they had changed so little in the last two centuries. This is a popular joke in education circles because it makes a point that, until recently, has not been very far from the truth.

However, this may not be true in the future. Pressures from society, businesses, and industry are changing the way we view education—what it should teach and how. These pressures also have an impact on traditional markets because each has an increasingly important educational component. The result: K-12, higher education, small business, and consumer markets are changing—and merging into what we call the "learning market."

Everywhere in the popular media, from books written by leading visionaries to articles appearing in local newspapers, we are hearing about the dawn of the information age, the knowledge revolution. Success will depend on having reliable, rapid access to overwhelming amounts of information. Because

people will have to extract meaningful information from quickly changing events, they will need the ability to assimilate and analyze discrete pieces of data. Businesses that see this trend realize that they must, in many circumstances, retrain current employees who came from the “old school” of learning. And for their new-hire positions, they are seeking highly skilled students who have information analysis skills.

Because businesses are demanding a more highly-educated workforce, they are putting pressure on schools to produce students who have the ability to make decisions quickly and adapt to rapid business change. Furthermore, many parents want to quick-start their children’s education by giving them learning advantages at an early age, as well as during their formal schooling. And adults must continue learning to meet the changing demands of the workplace and to enrich their lives. This means that educational needs may become increasingly pervasive throughout our lives, prompting the evolution of a “lifelong learning” market.

At the same time, the movement toward digital technologies is bringing together the formerly distinct worlds of computers, media, content, and communications. The “information industry” is therefore characterized by mergers, acquisitions, and diversification. How these new enterprises develop and sell products is in turn changing the nature of markets and distribution channels.

Learning-Market Opportunities. This blending of previously distinct segments is creating enormous opportunities for developers to cross-market products. Consider again the four traditional segments: K-12, higher education, consumer, and small business. Figure 1 shows the relative size of these market segments.

Apple’s share is increasing across all of them. Obviously, this is very important to developers, because more installed Macintosh hardware means more software sales.

A recent report from the Software Publishers Association indicated that total Macintosh software sales are up significantly. The news is particularly good in education markets, where Macintosh software sales were up 151 percent in 1991.

But what is the individual situation of each market? To answer the question of whether these markets warrant your consideration, let’s look at the opportunities

based on market size, Apple's position, and specific software needs of each segment.

K-12: A HUGE GROUP OF KIDS

K-12 is a huge market, comprising 47 million children and 2.7 million teachers at more than 100,000 schools. Although it is difficult to pinpoint an exact figure, an average estimate for the total number of personal computers used in K-12 is about 6.4 million. The vast majority of these systems are installed in the schools, where they are used by students in the classroom, lab, or media center. (See Figure 2 below.)

Apple is the leader in the K-12 market, outdistancing IBM by more than a two-to-one ratio in both penetration and market share. More than 80 percent of K-12 schools have at least one Apple personal computer, either Macintosh or Apple II.

The Apple II/Macintosh balance is shifting, and last year more Macintosh than Apple II computers were sold to the K-12 market. Schools are demanding the Macintosh for their long-term needs and have made the Macintosh LC the best selling personal computer in K-12. We expect the Macintosh to continue leading the way.

Application Needs. The missing links in today's K-12 software offerings are comprehensive math and science applications for K-6 students. There are also big opportunities for foreign language products, vocational education applications, and English as a second language (ESL) aids. Vocational products for computer literacy and business simulation should be particularly successful, because ESL products could be marketed not only in the United States, but also internationally. They could also be helpful to adults who must learn English.

The bottom line: When you think about new market opportunities, think expansively. Opportunities often hide in unsuspected places.

HIGHER ED: A SELF-REGENERATING MARKET

Higher education, another large market, is quickly self-regenerating. That is, it contains more than 14 million students who graduate (turn over, to put it in marketing terms) every four years. The market also includes 750,000 faculty members at approximately 3,500 institutions. Estimates indicate that 8 million

computers are being used by students, faculty, and institutions of higher learning.

As in the K-12 market, students own the vast majority of these systems. Higher education faculty are also avid users of personal computers. In fact, more than 80 percent of higher ed faculty members own computers. (See Figure 3 below.)

Apple maintains a lead in the higher education market overall, and the highest percentage of students and faculty use Apple products. Even better, our share continues to grow in this market. The potential market size, turnover rate, and number of computers owned by students makes higher ed a very attractive cross-marketing opportunity for developers.

Faculty Is Key. In essence, if you can win the hearts and minds of the faculty, you can win over both the students and the institutions. Here's why: More than half of the 80 percent of faculty who own computers use software for instructional purposes. The faculty understand the value of computing technology, and it has an impact on student and institutional software purchase decisions more than 75 percent of the time.

This influence is very important because students are not apt to buy software with the discretionary income that they could use to buy T-shirts or concert tickets.

To overcome this sales obstacle, developers should consider creating software solutions that can be integrated into the curriculum. That way, you can sell products as supplemental or even required tools for courses. Indeed, a significant portion of Apple's work with higher education software concentrates on integrating solutions into the curriculum.

By developing such software, you will get the technology into students' hands and it will have a more significant impact on the way they learn. So in higher education, develop for the students and sell to the faculty.

Opportunities. The real winners in higher education are comprehensive solutions for an entire semester or year within any discipline. Many vertical, niche solutions exist today that students can use for a few days or a week of course work. However, very little software exists that can be used for an entire quarter or semester. This is particularly true in introductory science, humanities, and business courses.

Such products that are integrated into the curriculum and that are sold through faculty recommendations require new thinking about distribution. One successful approach is establishing alliances with textbook publishers who already have a marketing/sales infrastructure to directly reach your primary customers, the discipline-specific faculty. Working with publishers, you can produce new products that range from textbook/software combinations to completely electronic textbooks.

THE CONSUMER MARKET: 7 MILLION-PLUS OPPORTUNITIES

The U.S. population hovers around 250 million, and is growing at a rate of 0.85 percent per year. These people live in more than 94 mil-lion households, approximately 18 percent of which own personal computers.

Not surprisingly, this number is expected to grow significantly throughout the decade. In fact, a Dallas-based marketing group predicts that there will be more computers than children in the average U.S. household before the turn of the century.

Interestingly, Apple's research shows that, in the home, computers are used most often for entertainment and education purposes. At least 36 percent of households with computers have at least one household member that uses the computer for schoolwork. That is, 7 percent of U.S. households today (almost 7 million) use computers for education-related purposes. What that number will be in the year 2000 is unclear, but there is no doubt that it is likely to grow dramatically. Thus, there is an incredible opportunity for developers to cross-market education products to the home.

Apple has a tremendous amount of name-recognition in the overall consumer market. With an increased focus on this market, new relationships with retailers, such as Sears and CompUSA, and a new marketing/ channel organization, we are working aggressively toward becoming a consumer market leader.

In home education, Apple is already in the forefront. At home, K-12 teachers use Apple products more than those of any other company. We also have a high percentage of the K-12 student, higher education student, and higher education faculty home-use markets.

Software Needs. Consumers use their home computers for several purposes: managing a home business, doing work brought home from the office, education, and so on. (See Figure 4.)

The consumer opportunity can be broken into four application segments: home business, personal finance, education, and entertainment. Some of the most striking needs among these segments are low-end productivity tools for the home-business buyer; financial planning tools; preschool, simulation, and "infotainment" products for the home-education buyer; and graphic adventure products for game-lovers.

SMALL BUSINESS

Compared with the markets already described, the small-business computer market is highly segmented; it is replete with vertical markets. Several of these markets correlate with the higher education curriculum in such disciplines as architecture, engineering, law, graphic design, finance, marketing, and other aspects of business.

If you define a small business as a company having fewer than 500 employees, more than 15 million can be counted in the U.S. alone; that number increases by more than 600,000 every year.

The 12 million personal computers installed in small- to medium-sized businesses are typically used by well-educated people who are likely to be familiar with technology and to recognize its value in their success. These users continue to buy technology. In fact, small-business owners spent about $7 billion on hardware alone last year.

Apple's overall share of this market is on the rise. However, in the context of education product opportunities, it isn't useful to talk about the small-business market in general, because Apple's (or any vendor's) market share fluctuates widely from one vertical segment to another. Apple makes one of its strongest showings among small businesses related to education, in areas such as architecture, engineering, and advertising. Each of these vertical segments is well-suited for cross-marketing education products.

Software Opportunities. Small businesses have very specific software needs. They require targeted products that provide high-level capabilities and professional-quality output. Business people want tools that will help them do their jobs more effectively, increase productivity, track their businesses, and increase the quality of output they provide to customers.

Several products have already moved across education and business boundaries very successfully. High Performance Systems' iThink, a business modeling package, is an example of a higher education product that migrated into business markets. In the other direction, ADAM Software, an anatomy and dissection product developed for medical professionals, is gaining popularity in medical schools, high schools, and the home.

SHOULD YOU CONSIDER CROSS-MARKETING?

In short, the opportunity in all of these markets is extremely large. But is your product a good candidate for the education market? Can your education product be marketed to home users or small businesses? What steps are needed to make the move?

Next month's marketing feature will discuss these issues, and the "Developer Outlook" will spotlight the experience of a developer who successfully made the transition across markets.

Janine Firpo is the higher education evangelist at Apple Computer, Inc. She works closely with colleagues who specialize in the K-12, consumer, and small- business markets.

****

Figure 1: 1991 U.S. personal computer market

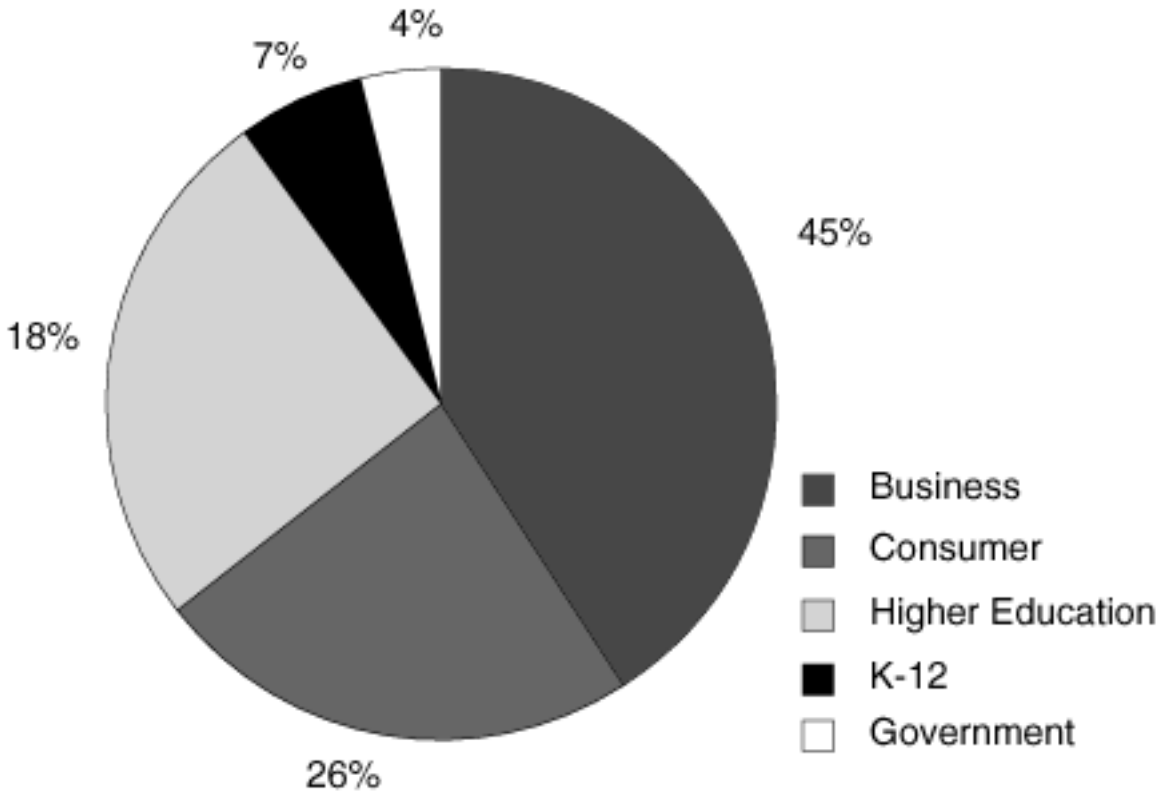


Figure 2: U.S. K-12 personal computer use

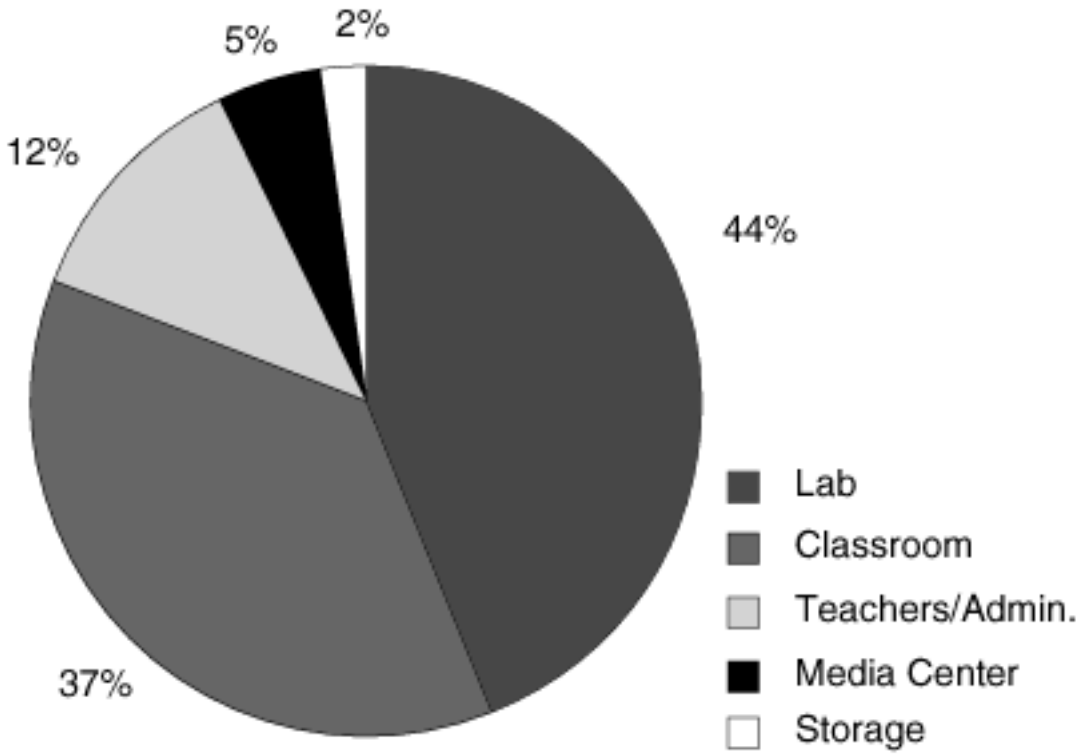


Figure 3: Distribution of personal computers in U.S. higher education

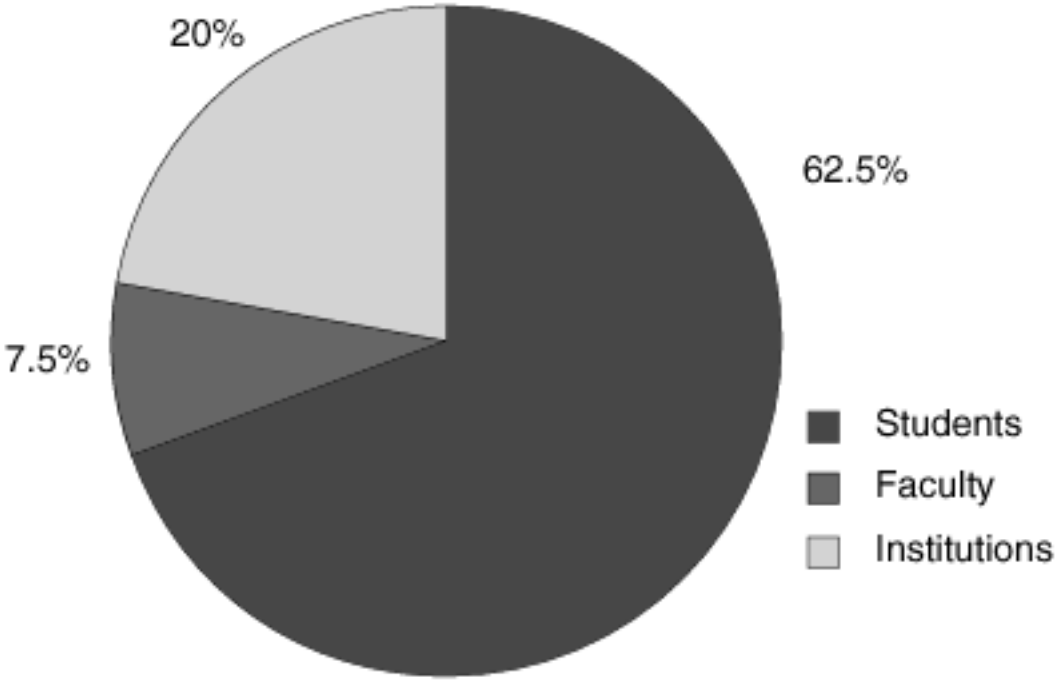
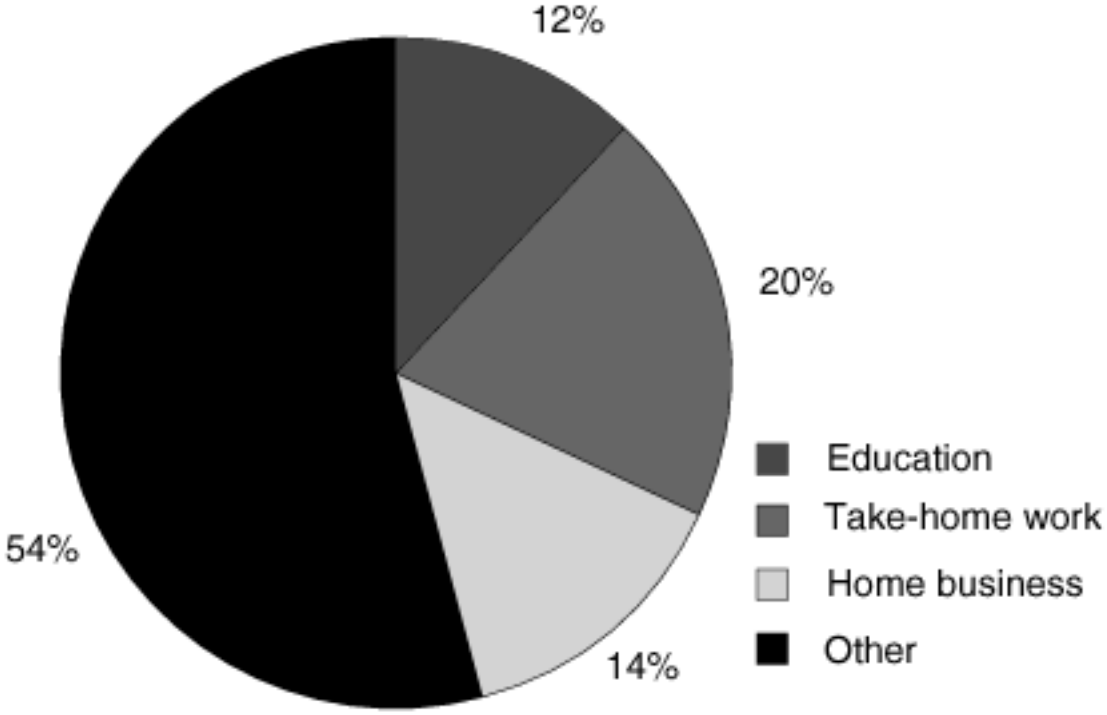


Figure 4: Uses of home computers in U.S.



Creating Your Own Revolution

Make a Bigger Splash Through Comarketing

by Steve Blank, SuperMac Technology, Inc.

When Apple introduced QuickTime last December, SuperMac was ready with the VideoSpigot digital video capture card to help lead the mad dash to success.

To the outside world, it may have appeared that SuperMac was extremely fortunate to have introduced a product for the digital video revolution at precisely the right time. But as you might surmise from your own experience, SuperMac had been working closely with Apple and other developers on the product and its launch for more than a year.

The fact that this was a cooperative marketing effort was critical to our success. While timing was very important—having the right product ready at the right time—the real coup was being able to harness the resources of multiple key players to launch and market the technology. We realized early on that going it alone would be a big mistake, considering the scope of the task. Joining forces with other developers—getting other players onto the QuickTime bandwagon—helped prevent the revolution from becoming an evolution.

Why was this a revolution? This was not just the launch of a new group of products; it was the birth of a new technology, a totally new way of doing things.

The ramifications: Before, video manipulation belonged almost solely in the domain of professional production companies and post-production houses. But with this new technology, everyday people could experiment with their camcorders and Macintosh computers and produce extraordinary video results.

But would users understand the potential of this new technology? Would they be willing to use it? Would it seem approachable?

THE COMARKETING APPROACH

It became apparent that a team approach to marketing this new technology would most effectively address these questions. To get a new technology or product adopted by the mainstream market, you must offer not a lone technology, but an entire solution, a package that helps users grasp the proposition your product or technology makes and achieve satisfaction from it.

(For a more detailed discussion of the "whole product" concept, see "Crossing the Chasm" by Geoff Moore in the May 1992 issue of *Apple Direct*.)

By joining forces with Apple and other developers, not only did we collectively offer users more of a "whole product" solution, but we were also able to augment our resources by leveraging other companies' efforts.

Forming the Team. We began discussions with Adobe Systems, WordPerfect Corporation, and other developers to get them committed to team-marketing the QuickTime technology breakthrough. Here's how it happened.

SuperMac had developed a video-editing application to test VideoSpigot. We considered marketing it ourselves, but decided against it after taking a hard look at the situation. We saw that SuperMac would be better off and farther ahead in the long run if we sold the program to another software developer, and then worked together to build the new market.

We felt we could make more of an impact and sell a lot more VideoSpigots if another credible company marketed the software and continued its development.

SuperMac is a hardware company, and from a previous foray into software publishing we know that it takes a dedicated, concerted focus to evangelize a new category of software—and a lot of marketing money and muscle. Being a hardware company, our focus and the resulting allocation of resources isn't conducive to the development effort needed to market and subsequently revise a software product.

After discussions with several companies, we sold the editing product to Adobe Systems, who named it Premiere. We chose Adobe for several reasons, most importantly because of its track record of evangelizing PostScript®, completely new system software that ultimately changed the face of the computer industry. We then struck a deal with Adobe to bundle our product with Premiere, and in the first three months 20,000 copies of the bundle—VideoSpigot or VideoSpigot Pro, and Premiere—were sold.

Since then, we have renewed our agreement with Adobe to indefinitely bundle Premiere with VideoSpigot and VideoSpigot Pro. With the bundle, users receive a more complete solution than they would with the individual products.

Furthermore, we felt we had to work with existing application-category leaders to effectively help end-users grasp the implications of QuickTime technology and our product. We contacted leading providers of word-processing,

multimedia, and other software. We enlisted such companies as WordPerfect, MacroMind•Paracomp (now known as MacroMedia), and several others.

A key WordPerfect person worked closely with us so that at the QuickTime introduction they could show a QuickTime movie—captured with VideoSpigot and edited with Premiere—in a WordPerfect document. MacroMedia showed Director and Action! presentations that incorporated QuickTime movies. Other companies used the technology in their products and demonstrations, as well.

Seize the Opportunity. The point I'm trying to make is that whenever a technology breakthrough is around the corner, seize the opportunity. If you can lock onto a product idea that supports the new technology concept—go for it.

And resist the temptation to keep everything close to the vest and go it alone. Especially if your company is smaller and more resource-limited, find like-minded companies and do something together. Ultimately, you'll make a bigger splash and offer users a more complete package than you could offer by yourself.

PREPARING THE MARKETPLACE

The comarketing effort didn't end with persuading other developers to use the technology in their products. Because of the scope and magnitude of the QuickTime concept and the introduction activities, we also had to create very special promotional materials.

The first step was creating a white paper explaining why analog-based video had failed to become a mainstream computer operation, and why digital video was the answer. During the nine months before the official QuickTime launch, the paper was widely distributed to industry influencers, the press, video enthusiasts, and anyone who showed an interest in the technology.

Next, we needed user testimonials. We instituted an aggressive beta program with industry analysts, end users, and other developers. From this group we culled several compelling examples and created both printed and video testimonials that highlighted real-world uses of Adobe Premiere, WordPerfect, and VideoSpigot.

The print testimonials were easy to distribute and very exciting. (It's great to see pioneers become emotional about a technological breakthrough.) The video samples created by these testimonial subjects were compelling and highly professional in content. They demonstrated the wildly diverse uses of

QuickTime/ VideoSpigot that were possible with today's equipment (everything from a corporate sales presentation, to a commentary on the starkness of New York City, to an educational video on eye surgery).

We also created a "Spigot CD" that included all of SuperMac's digital video marketing collateral, testimonials, sample movies, and other QuickTime-related materials. We even modified our VideoSpigot advertisement to include quotations from these testimonials.

We used the testimonials as widely as possible, and included them in trade show handouts, placed them in pertinent publications, and sent them to customers who inquired about our products.

It was critical to properly educate reviewers in the computer, business, and video press who had never experienced anything like QuickTime. It was also important to make sure they understood the limitations of this fledgling technology—as well as its potential.

To get the information across, we created a nuts-and-bolts reviewers kit and guide for our product. The kit included performance benchmarks from an independent laboratory, comparisons with competitive products, educational materials about digital video, and facts about the product. We also worked closely with our partners to make sure that every reviewer had products from and contacts inside our partners' companies (including Apple). In short, we all pulled together to make the review effort stronger and more effective.

THE INTRODUCTION AND AFTERMATH

To achieve the most dramatic, high-profile launch, we introduced VideoSpigot in SuperMac's and Apple's MacWorld booths in January. In addition to showing a variety of software running with VideoSpigot, we also demonstrated planned enhancements to our current system-level software, including compression and CD-ROM technology, at Apple's QuickTime room off the show floor.

This demo met the objective of showing people not only the current product, but also a glimpse of the future. Our partners also talked about and demonstrated our product, along with their own, in the many MacWorld panel discussions in which they participated.

Next, we had to demonstrate how this new technology would work in real-life situations. Demo '92 and TED3 were excellent venues.

At Demo ’92, we equipped a kiosk with a Macintosh Quadra, a large-screen SuperMac monitor and graphics card, HyperCard, VideoSpigot, and Premiere. We used those products to show attendees some highlights of the conference and other events by means of QuickTime movies—edited and produced only hours after the events happened. A camera operator shot footage, and an editing room equipped with Macintosh Quadras, videotape decks, VideoSpigot, and Premiere created the movies on-site.

At TED3, the semiannual industry event for creative leaders in education, entertainment, technology, and design, we created an interactive “electronic newspaper.” Conference guests could view articles and movies in this “video newspaper,” which highlighted the previous day’s events.

PUBLIC EDUCATION

These events were excellent for attracting the interest of industry insiders, but a real challenge was inciting people to buy products from our partners and us. To do that, we needed to educate the public directly. With our partners, we decided to offer seminars and hands-on classes throughout the U.S. and Europe.

One of the easiest solutions was to fit ourselves into existing classes. For QuickTime and our products, the American Film Institute (AFI) was a natural. AFI had a ready-made cadre of interested professional users as students. We sent product managers to train these classes, in a room fully equipped to make QuickTime movies.

The results were spectacular. The students ate up the technology, and their regular teachers were sold to the point that they became “evangelists” for the products. Now AFI was a partner in the process, too.

Combining the resources of Adobe, MacroMedia, Apple, and others, we travelled across the country holding a series of one-hour classes to teach people how to make QuickTime movies. To get the maximum impact from this investment, we also invited regional press people to not only cover the story, but also to learn how to make movies themselves. We also captured the mindshare of dealers by including them in these seminars with groups of potential buyers.

This three-month road show was an effective public relations/evangelism/education/sales/marketing tool. But it was by working with our partners that we were able to keep interest and enthusiasm high, in two ways: By directly touching our customers, we got the best market impact

possible; and by positively influencing the editorial community, we achieved favorable product reviews and technology trend articles.

CREATE YOUR OWN REVOLUTION

Yes, this is a wonderful, heartwarming success story, and I'll bet at this point you're muttering, "That's great for SuperMac, but we're just a small company. How could we do all that?"

Based on our recent experience, here's what I think can make the difference for any company. (Think of Aldus at the birth of desktop publishing and MacroMind before multimedia became in vogue.)

• *Keep your ear to the ground.* Watch the industry and other developers—and Apple—carefully for signs of the next big innovation. Apple has described its future technology plans for system software; somewhere in there lies a tremendous opportunity. Kaleida and Taligent will also provide major opportunities for innovation and fresh approaches to problem-solving.

•*When you've got something hot: focus, focus, focus.* Don't try to dilute your marketing and development efforts by working on several things at once. If your idea is hot, you'll know it. So focus on that idea/product and you've got half the job done.

• *Start working on your solution—your products—as early as possible.* Before a new technology is fully introduced and exploited, this may seem risky. However, the payoff can be really big if you get in on the ground floor with an innovative, exciting product idea.

•*Scrap the not-invented-here syndrome.* Find other companies whose products can take advantage of the new technology and your products. Work together to create a more complete package to offer users when the technology is unveiled. You'll be able to combine the ideas, resources, and manpower of several companies, and together you'll make a bigger splash than any one company could make alone. Also, take advantage of the various comarketing activities that Apple offers. (For more information, call the Developer Support Center at (408) 974-4897. Also, see "How to Comarket New Products With Apple" in this folder.)

• *Put your efforts where you can win.* Try to put your efforts and resources into a venture in which you can become a significant player, preferably Number

One. If you can't, then consider not playing or finding another venture. Put your efforts into products/markets where you can win.

•*Show 'em you've got sex appeal.* Entice the media, analysts, other developers, and potential customers with well-planned and executed promotions, demonstrations, and training. Shoot for maximum impact (which doesn't always have to mean maximum investment).

• *Keep mining the vein.* Product introductions are only the beginning. As for SuperMac, we are not about to stop mining the QuickTime vein. We will continue to develop partnerships with innovative software companies, and are looking for more opportunities to leverage the Apple effort. And it goes without saying—we are continuing to make our video products as innovative and desirable as possible.

See you at the movies!

Steve Blank is the vice-president of marketing for SuperMac Technology, Inc.

How To Comarket New Products With Apple

When Apple introduces a major new product, we offer developers the chance to join us and market your new software and hardware products that showcase our new technology.

These comarketing opportunities include coverage of your product in *Apple Intro News* and participation in Apple press briefings and introduction events. They provide your company and product with significant exposure to the press, customers, Apple field sales representatives, and the Apple distributor and reseller community. During the past 18 months, as we've introduced a significant technology, dozens of developers have worked with us to promote hundreds of products. Table 1 shows how many developers and products participated in comarketing activities in 1991 and 1992.

To be considered for participation in Apple comarketing programs, your product must be more than merely compatible with the new technology; it must be designed to take specific advantage of it. For example, an animation product that had been modified to allow users to save files in the QuickTime 'moov' format qualified for comarketing when QuickTime was introduced. Your product must ship within a designated time—usually 60 days—after the introduction.

We'll keep you posted about upcoming comarketing opportunities on AppleLink approximately three to four months in advance (path—Developer Support:Developer Services:Marketing Information & Services:Marketing Services:Apple Services:Intro Co-Marketing Opportunities). *Apple Direct* also publicizes Apple introductory activities and other comarketing events.

If you think your product qualifies, fill out a Product Description form (found at the AppleLink location just mentioned) and return it, following the instructions on the form. If your product is selected, we'll notify you. (Although we'd like to respond to every application, we can only reply to those companies selected for participation.)

If your product is selected, you'll be asked to provide a product description and a screen shot to be published in *Apple Intro News,* which features new third-party products that take specific advantage of a new Apple product or technology. Approximately 400,000 paper copies are distributed to Apple resellers and to the Apple field sales force, who then pass the publication on to their customers.

Additionally, we distribute about 45,000 copies of a CD-ROM containing an electronic version of *Apple Intro News* to all Apple sales representatives and resellers to use as a sales tool or to give to customers. In electronic form, *Apple Intro News* is a HyperCard stack that includes not only product descriptions but also product demonstrations in the form of QuickTime movies that you will be asked to provide.

Apple offers other comarketing opportunities on a more limited basis. If your product is selected, we may ask you for an announcement to be included in press materials that are distributed several months before a product introduction and another announcement for the press event held during the introduction. A small handful of third-party products may be demonstrated at the press events. You may also be able to demonstrate your product in a special product viewing area, or as part of the Apple booth if the introduction is held at a trade show. After the Apple product is launched, we may ask you to join us in field and international rollouts as well as in promotional activities.

****

		Number of Participating	
Event/time	**Product/Tech'gy**	**Developers**	**Products**
MacWorld S.F. 1991	Macintosh LC	47	60
WWDC 1991	System 7	89	117
Comdex 1991	Macintosh Quadra/ PowerBook/Classic II	85	137
MacWorld Boston 1991	System 7	35	50
MacWorld S.F. 1992	QuickTime	70	96

Table 1: 1991, 1992 comarketing activities

Broadening the Macintosh Base : Hardware and Software Sales & Distribution

In its continuing drive to broaden the Macintosh installed base, Apple announced several new sales and distribution arrangements over the past several weeks. These include an agreement with Minolta to sell Macintosh computers in Japan, a special retail pilot program, and an agreement with Ingram Micro to make Apple software available wherever software is sold.

Most recently, Apple Japan and Minolta Camera signed a sales agreement under which Minolta will sell Macintosh computers in Japan, beginning this month. Minolta's sales unit, Minolta Business Equipment Trading Co., anticipates selling 2,000 systems a year, worth 2 billion yen. This will remain a small part of Macintosh sales in Japan, which Apple projects will reach 180,000 computers this fiscal year, up from approximately 120,000 computers sold in the fiscal year that ended September 1991.

Apple USA launched a special retail pilot program for Macintosh computers this summer to expand its reach into consumer market channels. The pilot program is designed to pave the way for sales of new Apple computers through expanded consumer channels this fall. Apple will announce consumer distribution plans and specific new channel partners later in the summer.

During the pilot program, Apple will sell Macintosh LCII and Macintosh Classic®II computers equipped with system software and an application software program loaded on each computer's hard disk. (The application software may vary among the retailers participating in the program.) For the first time, Apple will provide customers with one year of toll-free telephone support and one year of on-site service. Retailers participating in the program include the electronics chain Silo and office product "superstores" Office Depot and OfficeMax.

"As part of our strategy to participate more fully in the consumer market, Apple is designing new service and support offerings to better serve the varied needs of consumers who are purchasing in the mass merchandise environment," said Keith Fox, Apple USA's vice president for consumer markets. "We anticipate being in approximately 1800 retail locations when we introduce our line of new consumer-oriented computer products later this year."

Earlier, Apple signed a software distribution agreement with Ingram Micro that enables computer buyers to purchase Apple software products wherever

software is sold, not just through Apple authorized resellers as was previously the case. Last month, Ingram made available to all U.S.-based software resellers several Apple products: System 7 upgrade kits, the QuickTime Starter Kit, Macintosh PC Exchange, AppleShare, and AppleTalk Remote Access.

The new software distribution agreement is designed to make it easier to purchase Apple software, and thus expand the base of Macintosh users. Apple expects to aggressively pursue software distribution through a variety of promotional efforts with major vendors.

APPLE DIRECT

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

EDITOR:
Paul Dreyfus (AppleLink: DREYFUS.P)

TECHNICAL WRITER/EDITOR:
Gregg Williams (GREGGW)

BUSINESS & MARKETING EDITOR:
Dee Kiamy (KIAMY)

PRODUCTION EDITOR:
Lisa Ferdinandsen (LISAFERD)

CONTRIBUTORS:
Juan Bettaglio, Cindi Cain, Pat Calderhead, Beth Dawson, Suzanne Dills, Marian Djurovich, Sharon Flowers, Lynda Lucero, Monica Meffert, Stacy Moore, Silvio Orsino, Katherine Parsons, Jessa Vartanian, Ana Wilczynski

MANAGER, DSSC:
David A. Krathwohl

CONTENT GROUP MANAGER:
Greg Joswiak

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

FILM:
Aptos Post, Aptos, CA

PREPRESS:
Prepress Assembly,
San Francisco, CA

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