Inside this issue

APPLE IS #1 in worldwide PC unit sales, according to several analysts. See this month's *Editor's Note.*
(see)

WAYNE'S GWORLD, the Nov/Dec Developer CD, features QuickTime 1.5 and versions of System 7 designed for use with the Macintosh Performa.
(see

NEW FROM APDA: the Hot Products of the Month are MacApp 3.0 and E.T.O., and *Spotlight On...* describes two new development tools.
(see)

develop, **THE APPLE** Technical Journal, is the next best thing to having an Apple engineer at your service. Its editor, Caroline Rose, describes Issue 12.
(see)

HYPERCARD 2.1 Player, a playback-only version of HyperCard, now ships with the Macintosh.
(see)

APPLE HAS FORMED a developer group for Europe and appointed a manager for its Japanese developer group.
(see)

FACT SHEETS For the Macintosh Duo and Macintosh IIvi/IIvx in last month's *Apple Direct* (October 1992) contained minor errors.
(see)

IT SHIPPED! **LISTS** third-party products that shipped in October 1992.
(see)

THE NEW APPLE Font Pack contains 43 fonts.
(see)

TRANSPARENCY is Human Interface writer Peter Bickford's fifth new HI design technique.
(see)

MAC X.400 SD is Apple's new OSI product.
(see)

BUSINESS & MARKETING

CONSULTANT GEOFF Moore writes about how to choose the best market for your products and stay with it.
(see)

BUILDING A software company by licensing products to others is the subject of this month's Developer Outlook.
(see)

Apple Modifies System Software Licensing Policy

Apple has changed its policy of permitting unlimited copying of its system software with the release of System 7.1, as reported in last month's *Apple Direct.* Under the new policy, system software will no longer be distributed free of charge through user groups or electronic bulletin boards, but must be purchased from Apple or an Apple reseller. The policy does not cover QuickTime or earlier releases of system software.

Apple developers will still have access to System 7.1 through the monthly Developer CD Series, but they may use it only for testing and development purposes. (See the license agreement on the CD for complete details about using the System 7.1 software that appears on the CD.)

To help you better understand Apple's departure from its previous policy of allowing free distribution of the Macintosh Operating System by authorized organizations, here is Apple's official explanation of the new policy and a selection of pertinent Q&As:

"Effective with the introduction of System 7.1, Apple has modified its system software licensing policies to end the distribution of Apple system software on a no-charge basis. This decision came about as part of our continual efforts to further Macintosh as the premier personal computer with the most innovative system and application software.

"Fundamental changes in the hardware business have also contributed to this decision. Changing the way Apple distributes system software will allow the Macintosh system software team to bring new system software developments to market and to reach more customers with these new technologies.

"Historically, Apple maintained an ad hoc approach toward providing new system software to our customer base. Customers purchasing a new Macintosh received the latest version of system software. However, our existing customer base had a limited set of options for obtaining new versions of system software. Those options consisted primarily of downloading the software from electronic bulletin boards or copying the disks from a Macintosh user group.

"Since only a small percentage of Apple customers have access to these services, these methods are not effective for the majority of system software

users. Furthermore, allowing system software to be distributed free of charge via these organizations has limited our ability to convince Apple resellers to distribute and promote Apple system software products to their customers.

"The rapid decline of personal computer hardware prices has made it impossible for Apple to continue the practice of allowing new system software to be distributed through user groups and electronic bulletin boards. In order to fund the development of system software enhancements and new capabilities across the entire Macintosh product family, not just on the newest models, Apple must institute programs that make our system software products pay for themselves. By establishing a new software licensing policy, Apple can offer customers new system software and extensions that will enhance their existing Macintosh systems at a modest price.

"While Apple recognizes that individuals who are accustomed to receiving system software upgrades from user groups and electronic bulletin boards may find this decision difficult to accept, we are confident that this is a positive move for Apple and all of our customers. By instituting this new software distribution policy, we can reach more customers through a wider array of software distribution channels than ever before. These new distribution channels include mail order catalogs, office supply stores, mass merchants, and software retailers, among others. And, by asking our customers to pay a modest price for their new system software, Apple will be able to accelerate future technological advances and offer technical support to more of our users via our toll-free customer support center.

"Additionally, Apple recognizes that some customers will be concerned about the impact of this new strategy on the price of moving to System 7.1. Therefore, we have introduced a low-cost system software update kit, which can be ordered directly from Apple. Customers have also asked Apple for an alternative to the System 7 Group Upgrade Kit. This new software licensing policy will enable Apple to offer a very flexible and cost-effective volume license program to our customers. Complete details on the new software volume license programs will be distributed by Apple in November."

For more information about the new policy, see AppleLink, path—Developer Support:Developer Services:Headlines for Developers.

QUESTIONS & ANSWERS

Q:Which system software products are covered by this new system software license policy?

All Apple system software is covered under this new policy, including the following system software retail products:

- M7202LL/A System 7.1 Update Kit
- M7200LL/A System 7 Personal Upgrade Kit
- M7201LL/A System 7 MultiPack Upgrade Kit
- M1252LL/A Macintosh PC Exchange
- M1269LL/A QuickTime Starter Kit
- M1436LL/A At Ease
- M1438LL/A At Ease - 10 Pack
- M4040LL/A Apple Font Pack for Macintosh

Q:What about earlier versions of Apple system software? Can user groups and electronic bulletin boards distribute the System 6 releases and version 7.0 of System 7?

Yes, Apple will continue to allow System 6 releases and version 7.0 of System 7 to be distributed by authorized organizations (such as user groups and electronic bulletin boards) that have existing agreements with Apple permitting them to distribute the software.

Q:What about the new QuickTime 1.5 system extension, tune-ups, and system enablers? Will user groups and electronic bulletin boards be able to distribute these system software components?

Apple has committed to making the QuickTime 1.5 system extension available through user groups and electronic bulletin boards, and we will stand by that commitment. Apple will continue to allow the QuickTime system extension, system software tune-up disks, and system enablers to be distributed by authorized organizations (such as user groups and electronic bulletin boards) that have existing agreements with Apple permitting them to distribute the software.

Q:Can users make copies of the new Apple software products, like Apple Font Pack for Macintosh, At Ease, or Macintosh PC Exchange?

No! These software products are covered under the standard Apple end-user license, which allows only an end-user to use Apple software on a single Apple computer and make one machine-readable copy for backup purposes.

Q:Will customers who recently purchased a System 7 Personal Upgrade Kit or System 7 Group Upgrade Kit be able to update to System 7.1 free of charge?

Customers who purchased a System 7 Personal Upgrade Kit or a System 7 Group Upgrade Kit after September 1, 1992, can obtain a free update to System 7.1 by calling Apple at (800)769-2775 (available in the United States only). Proof of purchase is required.

Q:How does this new policy affect the System 7 Group Upgrade Kit license?

Customers who have System 7 Group Upgrade Kits can update their users to System 7.1 through our new system software volume license program. Pricing and ordering information for the volume license program will be distributed by Apple in November. Effective October 19, 1992, the System 7 Group Upgrade Kit will be discontinued and no longer available for sale.

Q:Won't fewer people adopt your technology because of this new policy?

Apple is confident that we will reach more customers by modifying our system software licensing policy. Expanding our software distribution channels has been an important step in this direction. Previously, it was often difficult for a customer to find a reseller who carried new versions of Apple system software. Now, with our expanded distribution channels, system software is available in more of the places where our customers shop.

Q:Won't user groups lose an important revenue source because of this new policy?

Under the terms of the current software distribution licenses issued to user groups by Apple, user groups can charge their members only for the cost of disks and duplication. They cannot make a "profit" on the distribution of system software to subsidize the operations of the user group.

Q:If your goal is to reach more people, wouldn't making system software available from electronic bulletin boards help Apple attain that goal?

Not necessarily. Electronic bulletin boards and user groups reach fewer than 10 percent of the potential Macintosh user audience. In addition, the cost of downloading multiple system software disks from an electronic bulletin board has become rather costly—up to $250.00 in connect charges if all the System 7 disk images are downloaded. By ordering a System 7.1 Update Kit from our toll-free order line, customers can obtain a full copy of System 7.1, including user guides, within 48 hours of their order and at a very modest cost. They'll also receive one year of toll-free customer support on their new system software.

Q:Didn't Apple promise when they introduced the Macintosh that system software would always be free?

Apple never stated that Macintosh system software would always be free to Macintosh users. Our commitment has always been and will continue to be that Apple will include the latest version of system software with new Macintosh systems that we sell, as soon as possible after the introduction of a new system software release.

Q:What alternatives do Macintosh corporate customers have for updating all their users to System 7 other than purchasing several System 7 MultiPack Upgrade Kits?

A system software volume license program will be introduced in the United States in November that offers customers a discount commensurate with the number of licenses desired. System 7 Group Upgrade Kit customers can take advantage of volume license pricing, as well. Complete details on the system software volume license program will be distributed by Apple in November. ◆

Apple Anti-Piracy Initiative: An Update

The Apple Developer Group launched the Anti-Piracy Initiative over a year ago as part of its commitment to becoming a better business partner with developers. By working with you to fight the industry-wide problem of software piracy, Apple hopes to help you stay healthy and innovative, so that our mutual customers always have access to the best software products in the industry. We want to update you with our progress on this initiative over the last six months.

The Anti-Piracy Initiative includes activities in three areas:

- collaboration with existing industry associations that are working to fight software piracy
- leveraging of Apple's prominent industry position to raise awareness and to educate the Macintosh community about software piracy
- investigation of new anti-piracy technologies and other security strategies for Macintosh developers

As of our last update in the June 1992 issue of *Apple Direct,* we had distributed our anti-piracy education kit to over 20,000 developers and customers. We'd created a Macintosh version of SPAudit, software that quickly inventories the applications stored on a system and enables users to identify software that hasn't been obtained legally. And we'd translated the Software Publishers Association (SPA) anti-piracy video into French and Spanish.

This summer, Apple continued to communicate a strong anti-piracy message to constituents worldwide. Working with industry associations and developers, we educated users on the advantages of using legal software with educational brochures, sessions at industry events, and advertisements.

Key user-focused activities included the following:

- inclusion of anti-piracy messages in the user CDs (*Macintosh Demo Applications* CD and *Macintosh Demo Games* CD) shipped with every Macintosh with an internal CD-ROM drive (Performa 600 and IIvx), as well as the external CD 300 CD-ROM drives.
- presentation of anti-piracy education sessions at the Apple Education Forum, MacWorld User Group meeting, and National Education Computing Conference
- publication of anti-piracy articles and ads in user publications in Spain, Sweden, the United Kingdom, and France

- creation of an Apple/VSI anti-piracy brochure and poster targeted at the German market (VSI is the German national anti-piracy organization)
- translation of the Business Software Alliance (BSA) anti-piracy brochure into French, Italian, Spanish, Chinese, and Japanese
- distribution of a piracy user-awareness poster to all Apple retail outlets in France

Apple-initiated activities directed at developers and the computer industry included

- participation in the committee to create an industry-standard software licensing form, the License Service Application Programming Interface (LSAPI)
- piracy problem-solving sessions held at the European Software Publishers Conference, Apple's annual Worldwide Developers Conference, and the Apple Germany Developer Conference
- incorporation of anti-piracy information in numerous publications including the *Apple Multivendor Network Solutions Guide, Macintosh Services Directory, Macintosh Game Developers' Handbook,* and *Apple Direct*
- Apple Germany membership in VSI, the German national anti-piracy organization

Apple's Anti-Piracy Initiative is directed not only at our customers and developers, but within Apple as well. Anti-piracy policies and procedures have been built into Apple's corporate infrastructure. They've become an integral part of training for new employees, managers, and technical support personnel. And software audit tools have been made available to all employees.

By helping to reduce piracy, Apple hopes to restore profits to the pockets of developers like you worldwide. We encourage and applaud your efforts to raise awareness in your company and to educate your customers on the impact of software piracy. In addition, if you've had particular success or creative ideas on fighting piracy, we'd like to hear about them. Please send an AppleLink message to ANTI.PIRACY. We also encourage developers worldwide to participate in the anti-piracy discussion on the AppleLink network (path—Developer Support:Developer Talk:Anti-piracy Discussion). ◆


don't
copy
that
floppy

Developer Mailing Postdated

This month's *Apple Direct* and Developer CD, *Wayne's GWorld,* are dated "November/December." This is because we're instituting a new dating scheme for the monthly mailing that more accurately reflects when you actually receive it instead of when we mail it.

You'll still receive twelve mailings per year, each one including an issue of *Apple Direct* and a Developer CD. We're not changing what we send or how often we send it, just the way we date the materials in the mailing.

This month we begin the transition to the new dating scheme. According to the old scheme, this month's *Apple Direct* and Developer CD would have been dated "November." Under the new scheme, we would have labeled it "December," which would have made it appear that we skipped a month. Instead, we decided to date it with both months, to make it clear that we're changing.

The next *Apple Direct* and CD you receive (approximately at the end of December) will be dated "January." The mailing you receive at the end of January will be dated "February," and so on.

Here's why we're making the change: The monthly mailing goes out approximately the 15th of each month. This means that many developers don't receive it until close to the end of the month.

This is especially true of developers outside the United States (nearly half of the recipients of the mailing), who often don't see *Apple Direct* and the CD until early the month after we send them.

The new scheme, then, reflects the date you actually receive the mailing so it doesn't appear to contain "last month's" information. We've done this largely in response to those many developers who were receiving their materials late in the month and weren't sure that *Apple Direct* and the CD were the most current ones available.

The main thing to remember is that the frequency of the mailing and the information it contains remain the same; only the dates printed on the materials are changing. ◆

QuickTime 1.5: Delivering the Promise

Lets You Define Your Own Time-Based Data

By Gregg Williams,
Apple Direct *Staff*

Let's take a step back—QuickTime isn't just a way to run cute little videos on your Macintosh computer. QuickTime is Apple's comprehensive architecture for managing time-based data. With it, your programs can manipulate time-based data as another standard element in the Macintosh environment.

The metaphor for QuickTime is that of a *movie,* and QuickTime 1.0 delivered that, with movies that contain "tracks" of audio and video—movies that you play just like a videotape in your VCR. QuickTime 1.5 improves on the original QuickTime by delivering movies that play faster and larger. It's considerably more powerful and versatile than QuickTime 1.0, and that means you'll find more uses for it.

But that's not what *really* makes QuickTime 1.5 exciting. QuickTime 1.5 delivers the promise of a comprehensive framework for manipulating time-based data by allowing you to define new types of data. Once you put your custom time-based data in a movie, QuickTime 1.5 takes care of it automatically and makes it much easier for you to write sophisticated applications that can capture, edit, and play back complicated sequences of data.

IMPROVEMENTS SINCE QUICKTIME 1.0

In its early days at Apple, we called it QuickTime 1.1—we thought we were delivering some modest improvements to the original QuickTime 1.0. But when we brought all the pieces together and tallied up the changes, we found that the improvements were not just modest—they were substantial:

• *Movies play larger, faster.* With QuickTime 1.0, movies defaulted to 160 x 120 pixels. With the proprietary, software-only Apple Compact Video Compressor, QuickTime 1.5 defaults to playing movies at a 240 x 180 pixel size (over two times larger than the 160 x 120 size), but still doing so at the same speed as QuickTime 1.0.

• *Movies play better from CD-ROM discs and EtherTalk networks.* Apple engineers have optimized the low-level data handlers to perform better from

CD-ROM, which is the medium of choice for large movies, and from across an EtherTalk-based network.

• *Kodak Photo CD support.* With Photo CD, you can get Kodak to develop your film and give you back a CD with your pictures on it. With QuickTime 1.5, Photo CD discs appear on the desktop as normal CD icons, and files containing compressed Photo CD images appear as normal Macintosh PICT files (with a thumbnail representation of the image as the file's icon).

• *Text and derived media types.* The new text media type allows you to add tracks to your movies that contain text that changes as the movie plays. More important, QuickTime 1.5 gives you the ability to define your own types of data (called *media* in QuickTime terminology) to put into movie tracks. By overriding the QuickTime 1.5 Base Media Handler, you can let QuickTime do much of the work of manipulating this data type.

• *Hardware support makes full-screen, glitch-free 30-fps movies possible.* If you've got a hardware compression board in your Macintosh, you can play 640-x-480-pixel movies at a full 30 frames per second—in other words, in comparable quality to that of videotape. One improvement that makes this possible is QuickTime 1.5's ability to handle the compressed video images that these compression boards generate. (QuickTime 1.0 could handle only uncompressed RGB video data, which limited its performance.) Of course, hardware image compression will only get better and cheaper as time passes, so high-quality movies are definitely in the future of QuickTime.

• *Improved user interface.* Apple engineers have upgraded the Standard Movie Controller so that it has the shaded, 3-D appearance of System 7 windows (see Figure 1). Also, they have standardized the dialog boxes for configuring video-digitizer and audio-input hardware (see Figure 2). This helps vendors of digitizer hardware create the human interface with less effort, and it helps users by giving them standardized dialog boxes that work the same for both Apple and third-party hardware.

In QuickTime 1.0, when users were trying to open a movie, the StandardGetFilePreview dialog box showed a static image. Now, QuickTime 1.5 can also run a short video sequence in the Preview area of the dialog box (see Figure 3).

Apple engineers have also improved the Compression Settings dialog box, which you use whenever you are about to compress a movie or picture. As in QuickTime 1.0, this dialog box shows part of the image chosen for compression,

and when you change the compression method, the partial image changes to show what the compression method will do to the image. In QuickTime 1.5, the Compression Settings dialog box provides two new features. First, you can use the mouse pointer (which changes to a “grabber” hand pointer) to show other parts of the image about to be compressed. Second, you can use Option-click or Shift-Option-click inside the partial image to magnify or reduce the image being shown.

• *More versatile importing and exporting.* Under QuickTime 1.5, the StandardGetFilePreview dialog box can now open many nonmovie files and convert them automatically to QuickTime movies. This means that the users of your program have to worry less about file types and manually converting them into QuickTime movies. QuickTime 1.5 can automatically import AIFC, AIFF, and 'sfil' sound files, PICT and TEXT files or resources, and PICS image-sequence files. In fact, QuickTime 1.5 will automatically import any data type for which you supply an *import component* (a new kind of QuickTime component).

QuickTime 1.5 can also convert QuickTime movies to other formats by using new *export components.* QuickTime 1.5 includes export components for AIFF sound files, 'snd ' resources, and PICT image files.

• *Cross-platform support.* If you’ve decided against using QuickTime because you didn’t want to get locked into a Macintosh-only technology, it’s time for you to take another look at it; for details, see “QuickTime Does Windows—and More”. QuickTime for Windows 1.0 gives your applications the ability to play QuickTime movies, and further QuickTime support for Microsoft Windows is on the way. In other words, QuickTime is now a cross-platform technology.

• *Better audio and real-time movie capture.* With QuickTime 1.0, if you wanted full-motion movies, you had to videotape your footage, then wait for minutes or hours while a program digitized your footage frame by frame. (QuickTime 1.0 *could* capture video in real time, but only at a rate of 5 to 10 frames per second.) With QuickTime 1.5 and the right hardware, you can capture 30-fps, full-screen video and CD-quality audio (up to 44-kHz, 16-bit, stereo audio) in real time. That’s good, because when you don’t have to sweat the details, you end up with more energy to be more spontaneous and creative.

• *Better movie playback on black-and-white screens*. With QuickTime 1.5, color or gray-scale movies look better when played on a Macintosh with a 1-bit display (like a Macintosh Classic II or the PowerBook 140, 145, and 170.

COMPONENTS

Though the Component Manager was defined as part of QuickTime 1.0, it gained an independent status with System 7.1 and is now another manager in the Macintosh Toolbox. QuickTime 1.0 encapsulated much of its power in various types of components, including clock components, image compressor components, movie controller components, sequence-grabber components, sequence-grabber channel components, and video digitizer components. So what are they?

A Brief History of Components. *Components,* as implemented by the Component Manager, are software objects that allow the Macintosh to provide a standard set of services to a client application without the application having to know the component's underlying implementation details. For example, image-compression components can compress an image without needing to know what company made the component or even whether the compression is done through software or hardware.

You can create your own custom component, defining each service that you want this component to offer and giving the service a name. A client application can then find a component that meets its needs, open a *connection* to that component, and call the component's routines (which implement the component's named services) as if they were Macintosh Toolbox routines. Apple engineers have designed components so that they can be used simultaneously by multiple applications or more than once by the same application. A connection represents one instance of the component's use, and you must use the connection's value whenever you call one of the component's routines.

(For several technical articles on QuickTime, see the December 1992 issue of *develop* magazine. You can find these articles on the November/December 1992 Developer CD, titled *Wayne's GWorld.* Two articles of interest here are "Techniques for Writing and Debugging Components," by Gary Woodcock and Casey King, and the "Be Our Guest" column "Components and C++ Classes Compared," by David Van Brink.)

New Component Types. QuickTime 1.5 contains over two dozen new components, many of which are contained in one of three new component types.

As mentioned earlier, import and export components make it easier for users to get data files into and out of their QuickTime-based applications. When your application lets the user open a QuickTime movie (using the StandardGetFilePreview routine for System 7 or SFGetFilePreview routine for System 6), QuickTime 1.5 takes the files that all the import components can convert and adds them to the list of movies available for opening.

When QuickTime 1.5 opens a file that has an import component, it uses that component to convert the data inside the file into a QuickTime movie. Import components also allow the user to import something on the Clipboard and add it to a movie—either to an existing track, to multiple tracks, or as a new track. QuickTime 1.5 comes with import components for the following kinds of data: AIFC and AIFF sound files, System 7 'sfil' sound files, 'snd ' sound resources, PICT and TEXT files and resources, and PICS image-sequence files.

(The PICS import component is of considerable interest because many existing animation, video-manipulation, and multimedia applications can save their work in the PICS format. With the PICS import component, any QuickTime application running under QuickTime 1.5 can automatically open a PICS file as a movie, thus making life simpler for users and eliminating the need for them to worry about incompatible file formats.)

Similarly, your program can use export components to save QuickTime data directly to a specified resource or file format. QuickTime 1.5 comes with export components for the following kinds of data: AIFF sound files, 'snd ' sound resources, and PICT resources and files.

The last new type of component is the *sequence-grabber panel component* (see Figure 2). These implement the improved sequence-grabber dialog boxes mentioned earlier. Each sequence-grabber component includes a dialog box that lets the user change the sequence grabber's configuration and the code that implements the user's choice.

QuickTime 1.5 includes six such components, three for audio and an analogous three for video. The three for video are Compression (which sets the type and quality of compression), Image (which sets qualities such as hue, contrast, and sharpness), and Source (which specifies the type and source of the video signal and the video capture board being used). The three components for audio are similar—Compression, Sample (audio quality), and Source.

More New Components. Under QuickTime 1.0, the StandardGetFilePreview dialog box shows a single frame of the movie in the Preview area. QuickTime 1.5 includes four new preview viewers, which allow the StandardGetFilePreview dialog box to broaden the definition of *preview* to other media types.

QuickTime 1.5 uses a new movie preview viewer to show a short 3-to-5-second sequence of video in the Preview area of the StandardGetFilePreview dialog box (see Figure 3 below). Similarly, new AIFF, AIFC, and 'sfil' preview filters allow the user to preview (in this case, play) three different types of sound files.

The next three sections of this article discuss other new QuickTime 1.5 components: two new media handlers (the Base Media Handler and the Text Media Handler), one new compressor (the Compact Video Compressor), and two new decompressors (the Compact Video and the Photo CD Decompressors).

MEDIA HANDLERS

To recap the architecture of QuickTime: A QuickTime movie can be thought of as a collection of tracks, each of which contains content of a given type of media. (This is somewhat of a simplification. For more details, see my QuickTime article in the July, 1991 *Apple Direct,* the *QuickTime Developer's Guide*—available from APDA as part of the QuickTime Developer's Kit—or the new *Inside Macintosh: QuickTime* book. Guillermo Ortiz also did the cover story on QuickTime 1.0 that appeared in Issue 7 [Summer 1991] of *develop* magazine.)

In QuickTime 1.0, a track could contain one of two types of media: audio or video. QuickTime 1.0 also contained two types of components to handle these media: audio and video *media handlers.* When QuickTime 1.0 played a movie, it looked to these media handlers as "black boxes" that know how to play audio or video data on a Macintosh computer.

Base and Derived Media Handlers. QuickTime 1.5 improves on the original vision of media handlers by adding the *Base Media Handler,* which is a generalized media handler that you can customize to manage a new kind of media; the result is called a *derived media handler.* Once you have done that,

QuickTime 1.5 can create and play movie tracks that contain not audio, not video, but something entirely different—a *derived media type.*

Derived media types are very exciting because they allow you to leverage off the QuickTime metaphor of playing tracks of data. The key to making derived media types is creating a routine that knows how to "play"—that is, how to interpret the data and cause something meaningful to happen—a track's data at any give time *t*. Once you have created such a routine, QuickTime 1.5 gives you a framework, already in place, for

- mapping arbitrary actions to time
- encoding these actions into a format that QuickTime 1.5 can (without any extra programming on your part) store, recall, and edit
- automatically scheduling the actions so that they are performed on time, often in coordination with other actions

You can use derived media types, for example, to capture and play back data from a laboratory instrument or to have the Macintosh draw graphics "on top of" a movie as it plays.

Derived media handlers are best suited to media that store moderate amounts of data. Because of the overhead of the Base Media Handler, a derived media handler works best if the data that it reads and writes flows no faster than 25 kilobytes (KB) per second.

For more technical details, see "How Derived Media Handlers Work", which also outlines what you have to do to create your own derived media handler. Apple will provide more detailed documentation to help you write your first derived media handler.

Adding Text to QuickTime. Not content to let the concepts of base and derived media handlers remain an abstraction, Apple engineers used them to create a *text media type*—a fundamental new media type built into QuickTime 1.5, implemented using the Base Media Handler.

QuickTime 1.5 movies can have text tracks, which display text in a movie while it is running. Text tracks are very versatile. When you add a text sample to a movie, you can specify over 20 parameters for it, including the text, its style(s), size, font, justification, and color; the duration of the text's display in the movie; the optional presence and color of highlighting across part of the text; and optional scrolling of text, either horizontally or vertically. Figure 4 shows an

example of a text track that highlights selected words—in this example, words being spoken on an audio track—as the movie plays.

The QuickTime 1.5 Movie Toolbox supplies two high-level routines for adding text to a movie: AddTextSample, which adds a single block of styled text to an existing text track, and AddTESample, which does the same for text that may contain multiple style runs. Within a program, you can add text to a text track by calling either of these routines in place of AddMediaSample. (For more details, see “How Derived Media Handlers Work.")

Actually, the presence of a text import component makes adding text to a movie as simple as things get on the Macintosh. To add silent-movie–style captions to a movie, all you have to do is copy some text to the Clipboard, make the movie window active, scroll to the desired insertion point, and do a paste operation into the movie.

APPLE COMPACT VIDEO

The Apple Compact Video Compressor significantly increases the quality, playback size and rate, and compactness of QuickTime movies. It is best suited for 16-bit and 24-bit video sequences that have been captured but not compressed. It is a proprietary, software-only compressor (and decompressor, to be precise) that Apple supplies as part of QuickTime 1.5.

The Compact Video format is interesting because it is *asymmetrical*—that is, the times needed to compress and decompress an image are not close to each other. The Compact Video format takes a long time to compress an image, but it decompresses an image much more quickly, which is what we want—we can have the Macintosh compress a movie overnight, but we must decompress it in real time to play it. Depending on the movie and the Macintosh used, the Compact Video Compressor can take anywhere from 30 seconds to 4 minutes to compress a frame of video.

Another feature of the Compact Video Compressor is that you can constrain it so that the compressed movie it creates will not require more than the specified amount of data per second for feedback. This allows you to optimize a movie for the storage medium from which it will be played back. In particular, by constraining the Compact Video Compressor to a playback rate of 100 KB per

second, you can create a compressed movie that will play back extremely well from a CD-ROM disc.

PHOTO CD

Kodak's Photo CD advances the state of desktop publishing by making it easy to add pictures of your dog Rusty to your documents, newsletters, and presentations. (OK, it can be used for more than that—but, hey, it's a free country, so add Rusty's picture if that's what you want to do.)

The idea behind Photo CD is that Macintosh users with cameras can take some photographs and end up with electronic versions of their images for use with any Macintosh application that supports PICT images. It works like this: You shoot your roll of 35mm film and give the roll to a Kodak Photo CD photofinisher; that person then develops your film and hands you back your prints (in paper form) and a CD-ROM disc. The disc contains each of the photographs you took in five resolutions, ranging from 192 x 128 pixels to 3072 x 2048 pixels. You can bring your Photo CD disc back to the photofinisher to have additional images added to it, up to a limit of 100 images per disc.

QuickTime 1.5 and the Photo CD Decompressor make it possible for Macintosh users to view a Photo CD disc just as they would any other CD-ROM. In other words, a Photo CD disc appears on the desktop as a volume icon. Inside the volume are normal Macintosh folders, which contain document icons for each digitized PICT image. Photo CD helps you find the image you want; each document's icon is a thumbnail representation of the image contained inside.

To find out more about incorporating Photo CD into your application, or to have general questions answered, call the Kodak Information Center at (800) 242-2424, extension 53. At this number, they can also give you the name of the nearest Photo CD photofinisher.

DEVELOPER ACTION ITEMS

So what does all of this mean to you? If you have existing movies or are planning to make some, you should know the following things:

• Worried about memory usage? When idle, QuickTime 1.5 takes about 80 KB more memory than QuickTime 1.0. After that, the amount of memory consumed

depends on what QuickTime functions are active, but the additional memory used is roughly the same for both QuickTime 1.0 and 1.5.

• Because of a design decision in QuickTime 1.0, something called *video skewing* could occur; the result was that the movie's video could play as much as 1/8 second ahead of its audio. You may have fine-tuned your QuickTime 1.0 movie to compensate for the video skewing. QuickTime 1.5 was designed in a way that doesn't result in video skewing. This means that if your original movie compensated for QuickTime 1.0 video skewing, you might want to remove the compensation that you originally put in.

• With QuickTime 1.0, Apple recommended that you support black-and-white QuickTime-capable Macintosh computers by adding a separate 1-bit-video movie track for QuickTime to play. QuickTime 1.5 has a better dithering algorithm for converting color video to a 1-bit display, so you no longer need a separate black-and-white movie track in your movies.

• In QuickTime 1.0, if your program had to use the Standard Compression dialog box, you had to explicitly link the Standard Compression component into your program. (You did this by including the StdCompressionGlue.o and StdCompression.rsrc files in your program.) QuickTime 1.5 already contains the Standard Compression component.

If your program *requires* QuickTime 1.5, you don't have to include the two files just mentioned. If you want your application to run under both QuickTime 1.0 and 1.5, continue to add the two files to your program, but make sure you use the new QuickTime 1.5 versions of those files. Doing so gives your program the use of the QuickTime 1.5 Standard Compression dialog box even if your program is running under QuickTime 1.0; it also fixes a few minor bugs.

• If you want to fine-tune your existing movies, get the uncompressed video sequence you originally started with and compress it with the Apple Compact Video Compressor. If you are going to store the compressed movie on a CD-ROM, limit the Compact Video Compressor to a playback data rate of 100 KB per second.

In a related enhancement, remember that the new AppleCD 300i (now shipping in the Macintosh Performa 600 and Macintosh IIvi and IIvx) can run at double the speed of most of today's CD-ROM drives. To get the maximum performance out of your program, you may want to store two different video tracks (one each optimized for the single-speed and double-speed CD-ROM

drives) and let your program choose the one that matches the CD-ROM drive that is present.

• • •

QuickTime 1.0 was a revolutionary technology that made it possible for Macintosh computers to do things that no other personal computer could do. QuickTime 1.5 is even better, and more people are going to be using it, on more Macintosh computers.

Every Macintosh computer that Apple sells today can run QuickTime—and Apple is selling them in record numbers. With extra memory, most Macintosh computers now in use can run it, too. (Only the 68000-based models—like the Macintosh Plus, SE, and Portable and the PowerBook 100—can't.) If that isn't a big enough installed base, add every DOS computer running Microsoft Windows 3.1, and you've got a good chunk of just about every personal computer in use today as a potential market for QuickTime-savvy products. Now that's a market—and wouldn't you like a piece of it? ◆

Figure 1: *The QuickTime 1.5 Standard Movie Controller. Notice the shaded, three-dimensional controls.*

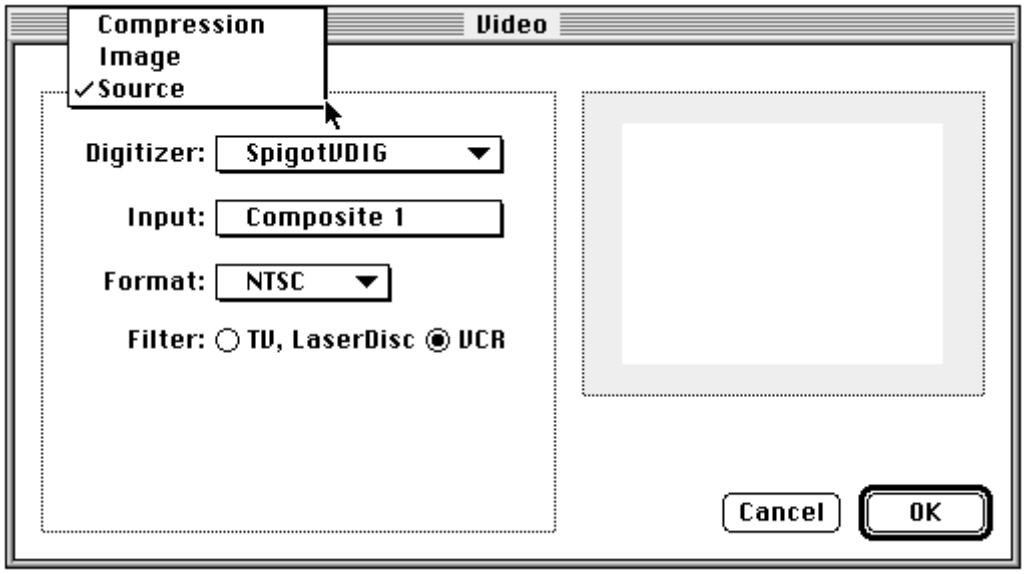

Figure 2: *The new, extensible dialog boxes for configuring video digitizer hardware. This screen shot shows a dialog box with the pointer about to choose one of three items in the pop-up menu. The dialog box then changes its contents based upon the menu item chosen. If necessary, you can add additional items to the pop-up menus; when chosen, these items would then show the user other configuration options.*

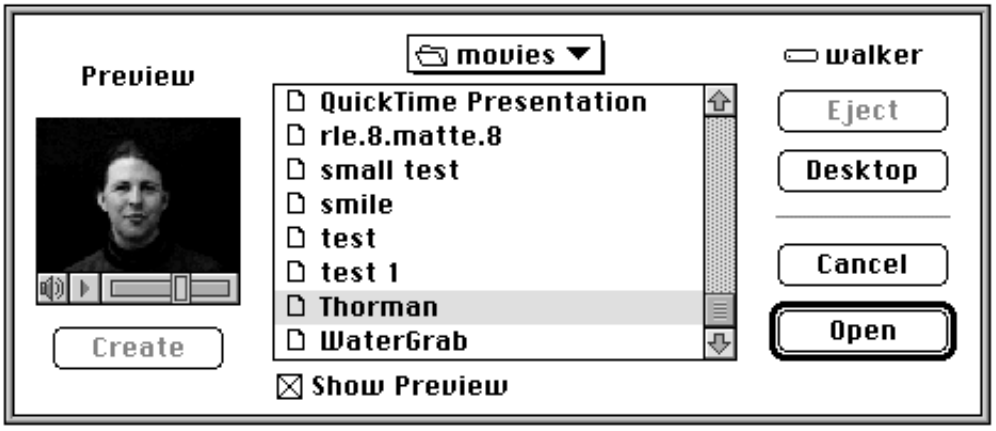

Figure 3: *Movie previews in QuickTime 1.5; note the movie controller bar underneath the preview image. Under QuickTime 1.0, this same dialog box showed a single frame from the movie and no controller bar.*

Figure 4: *Adding text to QuickTime 1.5 movies. These frames from a movie being played show how text can be displayed in parallel with the movie's audio and video. The text media type, which is a built-in media type in QuickTime 1.5, is also an example of how you can use the Base Media Handler to add new types of media to QuickTime movies.*

**********QuickTime Does Windows—and More

Cross-platform development is a reality of life—and Apple has pledged to give you the tools you need to write powerful, state-of-the-art applications for the two markets that are the most important to you: Apple Macintosh computers and DOS computers running Microsoft Windows 3.1.

At Macworld Expo in San Francisco last January, Apple CEO John Sculley showed a prototype of QuickTime running on Microsoft Windows; he also pledged to deliver a QuickTime for Windows developers' toolkit by the end of calendar year 1992. At the recent Fall 1992 COMDEX show, Apple announced the availability of the QuickTime for Windows 1.0 Software Development Kit from APDA. (To order, see the APDA information in Now Available From Apple.) Apple will also license this software to software and hardware vendors for distribution with their products.

QuickTime for Windows 1.0 allows Microsoft Windows 3.1 programs to play QuickTime 1.0 movies just as they would on a Macintosh computer. You can create and edit QuickTime movies on a Macintosh and play the same self-contained movie files on the Windows platform. Users do not need to perform any special conversion to switch between platforms, and you can publish CD-ROM QuickTime products that will work on both the Macintosh and Windows 3.1 platforms. Also, the application programming interface (API) for both the Macintosh and Windows versions are virtually identical, which means that you can write a QuickTime-aware Windows product without having to learn a new API on the Windows side.

As time goes on, Apple plans to make more of QuickTime available on the Windows platform. The QuickTime for Windows 1.1 Software Development Kit is scheduled for release in early 1993. It will include selected features from QuickTime 1.5, including support for the new Compact Video compression scheme.

QuickTime for Windows 1.0 is implemented as a set of Dynamic Linked Libraries (DLLs) that provide all the functions that a Windows program needs to play back QuickTime 1.0 movies. QuickTime for Windows 1.0 requires the following software and hardware:

- an IBM-compatible DOS computer with at least an 80386 processor, 4 MB of memory, an 80 MB hard disk, and a VGA graphics adapter
- to play movies with audio, an IBM-PC sound card
- the MS-DOS 5.0 operating system and Microsoft Windows 3.1 (or later)

Beyond Windows. Apple wants QuickTime to become an industry standard so that you can port your QuickTime-aware applications to multiple software platforms. Last May, Apple announced an agreement with Silicon Graphics that will provide limited QuickTime support on SGI's Iris workstations. By directly supporting the QuickTime file format, users of SGI and Apple Macintosh computers will be able to create and play QuickTime movies on each other's computers.

To help Macintosh users benefit from graphics generated on other computer platforms, Apple has created the QuickTime Movie Exchange Toolkit. This product, available from APDA, contains utilities that convert graphics from MS-DOS/Windows, IBM RS/6000, DEC VAX, Sun-4, Silicon Graphics, A/UX, Cray YMP, and Sequent Symmetry computers into Macintosh QuickTime movies.

Beyond computers per se, Apple has licensed selected QuickTime technologies to Kaleida, an independent company formed as part of the 1991 joint agreements between Apple and IBM. Kaleida is focusing on developing media-rich environments for consumer-electronics devices and other noncomputer areas. ◆

****

QuickTime 1.5 Availability

QuickTime 1.5 has been available since the date of its announcement on October 19 of this year. It is available through various sources, including the Apple System 7.1 Upgrade Kit (except the Japanese version). QuickTime 1.5 takes the form of a System 7 extension file and, unlike System 7 itself, can be copied freely. (For more details, see the system software licensing article in this issue.)

The *QuickTime Version 1.5 for Developers* CD-ROM contains QuickTime 1.5 and the largest set of QuickTime tools, on-line documentation, and sample code available to the public. It is available from APDA, along with over 1400 pages of printed documentation, as the QuickTime Developer's Kit v. 1.5; check the APDA information in Now Available From Apple for ordering information.

Apple Direct readers who get the technical monthly mailing from APDA or the Apple Developer Programs group also get the monthly Developer CD. This month's CD, *Wayne's GWorld,* contains the QuickTime 1.5 extension and several basic tools. Developers who are Apple Associates or Partners also receive the *QuickTime Version 1.5 for Developers* CD-ROM in the same monthly mailing. ◆

****

How Derived Media Handlers Work

It takes a moderate amount of work to create a derived media type, but it's worth it—once you've done so, QuickTime 1.5 does a *lot* of work for you. The

descriptions below leave out numerous details, but they should give you a general idea of how derived media handlers work and what is involved in creating one.

Creating a Derived Media Handler. Since a derived media handler is a type of component, you must follow the rules for creating a new component.

- You start out by deciding what messages your derived media handler component will handle. Then you write one routine for each message (or, equivalently, service) that the derived media handler supports.
- Among the routines you must write is one that is called when QuickTime 1.5 opens the media. This routine (which may have a name like MyMediaOpen) must do two things to ensure that this derived media handler (which is also a component) works with the Base Media Handler. First, this routine must open a connection to the Base Media Handler. Second, this routine must also specify that it "owns" the Base Media Handler; you specify this with a routine called ComponentSetTarget.
- When some code requests a service of a given component (such as your derived media handler component), the Component Manager executes a single dispatch function that then handles all requests for a component's services. You must write this function. The function should look at the request. If the request is for one of the routines mentioned earlier, this dispatch function should call that routine. Otherwise, it passes the request on to the Base Media Handler by calling the DelegateComponentCall routine.
- The key routine that you must write is called MediaIdle. QuickTime 1.5 periodically executes this routine, and your implementation of this routine "shadows" (that is, overrides) one of the same name for the Base Media Handler.

Given a time *t*, this routine must get the media sample for time *t* (using the GetMediaSample routine) and do whatever it is that this derived media track should be doing at that particular instance—for example, updating an animation that is drawn on top of the movie or redrawing a graph that changes with time.

Once you've implemented your derived media handler, QuickTime 1.5 automatically takes responsibility for playing any track that contains samples of this derived media—and for doing so simultaneously with the content of the movie's other tracks.

• Finally, you must write some routine with a name like MyMediaClose; this routine does any cleanup operations associated with your derived media type.

Adding Derived Media to a Movie. The code outlined above enables QuickTime 1.5 to play any movie that contains your derived media type in one or more of its tracks. However, you also have to write code that will create those tracks and fill them with the right data. This process is simple and easy to implement; you will probably use the following calls:

• NewMovieTrack (adds a new track to an open movie)

• NewTrackMedia (defines the type of media the track will contain and associates the proper media handler—in this case, your derived media handler—with the track)

• BeginMediaEdits (must be called before adding samples to the track)

• AddMediaSample (adds the data for time *t* into the track; you must do this once for each time/data pair that needs to be recorded into the track)

• EndMediaEdits (performs necessary functions that must occur after your code has finished recording data into the track)

What data must be recorded into the track? That's up to you. But whatever it is, it must contain enough information to tell the MediaIdle routine what to do when it's asked to "play" the contents of the track at time *t*.

You can easily incorporate these routines into a program that, either in real time or not, records data into a track containing your derived media type.

However, if you incorporate them into an import component for your derived media type, you get two added bonuses. First, you can import data of your media type to any movie by pasting it into the movie from the Clipboard. Second, when any application displays a StandardGetFilePreview dialog box to open a movie, it shows files that contain data of your extended media type. If the user selects one of these files, QuickTime 1.5 automatically converts the data file, using your import component, and opens it as a QuickTime movie.

Usually, one of the services that you'll want your derived media handler to provide is a high-level routine that adds a sample to an open track at a given time *t*. Such a routine translates the data that has been placed into a buffer of unstructured bytes and then calls the routines listed earlier. It is usually more convenient to create such a routine than to use AddMediaSample directly; AddMediaSample can only write an arbitrary sequence of bytes into the track,

and the high-level routine you write simplifies your code by making the low-level details invisible. ◆

Editor’s Note: Of Pride and Humility

This month at *Apple Direct,* we have a great deal to be proud of, and a reason to feel humble, as well.

We’re very proud to share some great news with you: Apple has overtaken IBM to become the leader in worldwide unit shipments of personal computers.

According to International Data Corporation (IDC), Apple had sold 2,035,000 units worldwide through September 1992 while IBM had unit sales of 1,854,000.

InfoCorp also put Apple in the lead: According to its numbers, Apple sold 1,318,000 units worldwide in the first half of 1992 compared to IBM unit sales of 1,230,000.

Further, InfoWorld Editor-in-Chief Stewart Alsop wrote in July that “Apple’s recent sales strength makes it the leading manufacturer of PCs....Apple is the biggest PC manufacturer.”

Not bad, considering that only four years ago, IBM unit sales were nearly twice Apple’s and that the idea of catching them seemed, at best, preposterous.

IDC predicts that Apple sales will continue to lead the personal computing industry for at least the near future. Since we’ve only begun to announce our new Fiscal Year 1993 products (such as the new PowerBooks, the Duo System, and the Macintosh Performa and IIv-series computers), we can’t help but agree.

In a recent letter to Apple employees lauding them for making the company #1, CEO John Sculley pointed to a key factor in Apple’s success. “What’s telling,” he wrote, “is Apple’s recent first-place finish in the recent J.D. Powers customer satisfaction survey—the second year in a row. I’m certain it’s not mere coincidence that this has preceded our top-ranked unit sales.”

So where does the humility fit in? To echo the theme this column struck last month, we need to humbly thank all of you—Apple’s great developers—for your loyalty, your energy, and your perseverance in sticking by us and making us #1. It’s your great applications and other products that make the Macintosh so wonderful to use, so satisfying to customers.

So keep it up. To paraphrase the poet, if we can keep working together to deliver the best user experience, the best is yet to be.

We also hope the best of *Apple Direct* is yet to be. As I said a couple of months ago, we’re working on a new design. Thanks to all of you who’ve given me feedback on the current design. I know I promised a report on that feedback

this month, but the news about Apple becoming #1 couldn't wait. So next month I'll tell you about the feedback and where *Apple Direct* is headed.

In the meantime, I hope you enjoy this month's issue, and keep those cards and letters coming (AppleLink:APPLE. DIRECT).

Paul Dreyfus
Editor

CD Highlights

Greetings and welcome to *Wayne's GWorld,* the November/December issue of the Developer CD Series. (The CDs have remained the same, the dates have been changed to confuse the innocent. See the story on the developer mailing that explains the date change.)

Included on this disc are several exciting new packages. Among them are QuickTime 1.5 and System 7.0.1(P) and 7.1(P). Last month we included System 7.1 for the first time. Please note that in order for us to get the requisite permission to publish these materials on our disc we have been required to include a new, more stringent license. This license allows you to use this software for testing and development purposes only. Be sure to read through the new license before using the disc, and see the article on Apple's new system software Licensing policy.

Developer Notes: Developer notes provide descriptions of new hardware and software features, comparisons with existing CPUs, and expansion card design information. This month's disc brings you developer notes for two new products, the Macintosh IIvx and PowerBook 160/180. Several notes have been updated, including those for the PowerBook 145, Quadra 950, and Macintosh LC II.

QuickTime 1.5: Take a look at the new QuickTime 1.5, which includes extensions and programming interfaces, CD-ROM Set-up 4.0 for Photo CD support, MovieShop b24, MoviePlayer, and the QuickTime scrapbook. Featured in this version are a new video compressor, Compact Video for larger software-only video, support for hardware compression boards, alternate track support, and network tuning.

LISP Goodies: Thanks to the efforts of Steve Strassmann, this disc contains a number of contributions collected by the Macintosh Common LISP group in Cambridge, MA. Here you will find tools, sample code, and Q&As, and several updates to the 2.0 release of Macintosh Common Lisp (MCL), including a patch file from Apple and several user-contributed applications and utilities. For the serious LISP programmer, a much more extensive collection of LISP materials can be found on the MCL 2.0 CD-ROM available from APDA.

Developer University Training: In its new course catalog for Oct 1992 through March 1993, Developer University offers one new minicourse, *Programming With MPW QuickTime,* as well as three new classes, *Open Collaboration Environment (OCE), QuickDraw GX,* and *Writing and Using Device Drivers.* Check the catalog included on the CD for more information.

ColorSync: This disc brings you an updated version of ColorSync, an extension to the QuickDraw graphics model that integrates a color matching architecture with the Macintosh Operating System. This version features great results across all color devices, system-level color matching, and open architecture for third-party enhancements.

PSWriter: This is a beta release of the replacement for the PostScript LaserWriter driver. It supports PostScript Level 2 as well as PostScript Printer Definition files for easy addition of printer-specific features. It generates EPS files and efficient PostScript and is significantly faster than previous LaserWriter drivers.

New Tools & Apps: Among the new tools available on this disc are Wake 100, which resets the wake-up time on PowerBook computers, and an updated version of ADB Parser that allows access to most of the system's ADB record and is useful for developing or debugging new ADB devices.

Animation *ƒ* is a Color Sprite Manager library and example program that uses off-screen color pixmaps. APPL loads APPL *ƒ* is a hack that allows an application to create a heap and then loads another application inside itself. Once the application inside is launched, the parent application has no control until the new application quits. Also, don't miss the updated Network Software Installer 1.2.3 for upgrading AppleTalk, Ethernet, and Token Ring software to support AppleTalk Remote Access Protocol.

Apple DocViewer 1.0: This on-line documentation viewing application provides a standard way for Apple to distribute documentation to developers in an electronic form. Release 1.0 provides a quick on-screen reference for browsing multiple documents. It supports full Macintosh mixed-text and graphics, printing, and an interactive catalog/table of contents, with book

indexes, text search and retrieval, and the ability to copy text and pictures for use in other applications. A so-called “bookmark” feature provide the ability to flag commonly used pages in documentation.

System Software: Included on this disc are four new versions of international system software. Swedish 7.1, Russian 7.1, Polish 7.1, and Portuguese 7.1. Also find the latest versions of U.S. System 7.0.1(P) and 7.1(P), which are designed specifically for use with the Macintosh Performa CPUs.

Happy programming and, as always, be sure to let us know how you like the disc by sending us an AppleLink message to DEV.CD.

Sharon Flowers
Developer CD Project Manager

Editor’s Note: Apologies to Sharon for leaving her name off of last month’s CD Highlights column.

Hot Product of the Month

MacApp 3.0—Special Offer!

Order any of these MacApp 3.0 bundles before December 20, 1992, and receive a FREE *MacApp 3.0 Reference*

MacApp is the most advanced object-oriented framework in the industry for developing user-friendly, professional Macintosh applications. MacApp streamlines development by supplying the application's main event loop and code for all basic Macintosh features, saving valuable programming time. New features in MacApp 3.0 include built-in support for System 7, the benefits of C++, and new tool features.

Order any of the MacApp 3.0 bundles listed below before December 20, 1992, and receive a FREE copy of the new *MacApp 3.0 Reference*. This 420-page quick reference encyclopedia describes the "nuts and bolts" of the MacApp 3.0.1 Class Library and Application Framework.

- ***MacApp 3.0 Disk Version***—Includes sixteen Macintosh disks, the *MacApp Programmer's Reference, Guide to MacApp Tools, MacApp 3.0 Tutorial (C++)*, and release notes, and FREE copy of the new *MacApp 3.0 Reference*. (*B0618LL/A $395.00)*

- ***MacApp 3.0 CD-ROM Version***—Includes one CD-ROM disc, the *MacApp Programmer's Reference, Guide to MacApp Tools, MacApp 3.0 Tutorial (C++)*, and release notes. (*B0619LL/A $350.00)*

- ***MacApp 3.0 Update, Disk Version***—Includes 16 Macintosh disks with the MacApp class library, interfaces, support tools, sample programs, documentation, the on-line *MacApp Class and Method Reference* and release notes. (*B0861LL/A $250.00)*

- ***MacApp 3.0 Update, CD-ROM Version***—Updates your MacApp 2.0 and includes one CD-ROM disc with the MacApp class library, interfaces, support tools, sample programs, documentation, the on-line *MacApp Class and Method Reference* and release notes. (*B0862LL/A $200.00)*

• E.T.O., Essentials•Tools•Objects Complete New Subscriber Package

E.T.O. is a comprehensive collection of Apple development tools on a CD-ROM (with printed, final documentation) and is sold as a four-issue subscription. E.T.O. includes the Macintosh Programmer's Worksop (MPW) Development Environment; MacApp; MPW C++, MPW C, MPW Object Pascal, MPW Assembler, SADE debugger, SourceBug debugger, ResEdit, MacsBug, Virtual User, and many more utilities; essential system software, sample code, technical documents, and selected prerelease versions of tools. (*M0895LL/B $1295.00)*

MacApp, E.T.O., and other Macintosh development tools are available for site licensing. For more information, contact Apple Software Licensing at (408) 974-4667.

Source Code: HPMNOV92

Spotlight On...

New Tools for Commercial Developers November/December

NEW! XTND DEVELOPER'S KIT VERSION 1.3.6

Apple Computer, Inc.

With XTND Developer's Kit, you can include file translation capabilities in your applications and create file translators that permit users to read and write files from different applications and different computers. Macintosh, OS/2, MS-DOS, UNIX, and mini or mainframe computer text and graphics file formats can be translated. This version includes System 7 support, decreased overhead in application launch, bundled core translators, and more.

System Requirements: A Macintosh Plus or later and system software v. 6.0.7 or later.

Product Contents: One Macintosh disk, one manual, one binder, and a software distribution agreement.

APDA Product Number: R0096LL/B $30.00

APPMAKER

Bowers Development

AppMaker speeds Macintosh software development by letting you create and change the user interface for an application the same way you use the Macintosh: by pulling down menus, pointing, and clicking. You can also easily create specialized elements such as palettes, picture menus, and custom controls (such as picture buttons and sliders). AppMaker supports the THINK Class Library, so you have your choice of procedural or object-oriented code.

Licensing Note: Site licensing is available directly from Bowers Development: (508) 369-8175.

System Requirements: A Macintosh Plus or later.

Product Contents: One Macintosh disk and one manual.

APDA Product Number: T0322LL/E $299.00

Source Code: PRNOV1

develop: Bigger and Better Than Ever

If you can't have an Apple engineer by your side, showing off code and amusing you with anecdotes, *develop,* the Apple Technical Journal, is the next best thing.

Whether you subscribe to the printed version of *develop* or only read it on the Developer CD, you'll be happy to know that the latest issue, Issue 12, is bigger and better than ever. Its 144 pages cover topics such as components (now part of System 7.1), QuickTime time bases, the Apple event object model, and globals in stand-alone code. These articles will inform and entertain you for hours, and the accompanying code on the CD will help you quickly apply that knowledge in useful ways.

Here's a closer look at what's in Issue 12:

- "Techniques for Writing and Debugging Components." Components aren't just for QuickTime programmers anymore.
- "Time Bases: The Heartbeat of QuickTime." Understanding and manipulating time bases directly is sometimes helpful. Here are some tips.
- "Better Apple Event Coding Through Objects." Adding Apple event object model support to your existing OOP code may be easier than you think.
- "Another Take on Globals in Stand-Alone Code." For MPW users, here's an alternative way to implement globals in stand-alone code.
- "Components and C++ Classes Compared." Components and C++ classes have some surface similarities but underneath are very different beasts.
- "Animation at a Glance." Three basic animation techniques everyone should know about.
- "Top Ten Printing Misdemeanors." You know the felonies, now learn the lesser printing crimes.

You can also read Dave Johnson's thoughts on genetic takeovers and Lamarckian evolution, try to figure out KON & BAL's puzzle about that teeny built-in debugger, and look over the Q&A section to see if it answers any of your own queries.

I hope you'll check out the latest *develop* and let me know if you agree it's better than ever. Your praise and criticism are equally welcome at AppleLink DEVELOP. ◆

Caroline Rose, Editor

develop

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

Books

MacApp Reference
R0195LL/A
$30.00

Apple Publications Style Guide, October 1992 edition
A7G0030/E
$30.00

Macintosh Development Tools & Languages Guidebook,
1993 edition
A7Z2000/D
$6.95

Tools

QuickTime Developer's Kit, version 1.5
R0147LL/B
$250.00

VISCA Driver 1.2
R0111LL/B
$50.00

Developer University Mini Course: QuickTime Programming Tutorial

R0413LL/A
Introductory Price $80.00

Developer University Mini Course: MPW Programming Tutorial
R0247LL/A
Introductory Price $120.00

MacTCP 1.1.1 Developer's Kit
B0943LL/B
$100.00

MacTCP 1.1.1 Developer Upgrade Kit
R0446LL/A
$25.00

MacX.400 Programmer's Kit
R0131Z/A
$200.00

MacX.400 Developer's Kit
B0847Z/A
$3,200.00

Third-Party Products

VideoToolkit
Abbate Video
T0552LL/A
$279.00

Ordering Information

To place an APDA order from within the United States, contact APDA at (800) 282-2732; in Canada, call (800) 637-0029. For those who need to call the U.S.

APDA office from abroad, the number is (716) 871-6555. You can also reach us via AppleLink; the address is APDA. If you're outside the United States, you may prefer to work with you local APDA contact. For a list of non-U.S. APDA contacts, see the "International APDA Programs" page in the *APDA Tools Catalog.*

HyperCard Player to Replace HyperCard in New Macintosh Shipments

Beginning in September, Apple began bundling a playback-only version of HyperCard with Macintosh computers instead of the fully capable HyperCard it had previously shipped with each Macintosh.

With the new product, called HyperCard 2.1 Player, users can view HyperCard stacks but won't be able to write their own.

The player was first bundled with the Performa Macintosh computers for home users, which began shipping last month. By October it was anticipated that all Macintosh computers shipped in the United States would include only the player and not the complete version of HyperCard.

Neither the new player nor HyperCard will be bundled with Macintosh computers sold in Europe, except with Macintosh IIvi and IIvx units shipped with CD-ROM players and in France as part of French System 7.1.

The HyperCard 2.1 Development Kit remains available separately from Claris for stack developers. Developers can also purchase the HyperCard Developers Licensing Kit from Claris so they can distribute the HyperCard Player with their products.

A manual does not exist for the player, so developers need to consider the instructional needs of new Macintosh users who wish to view their HyperCard stacks. HyperCard documentation will no longer be shipped with the Macintosh.

HyperCard 2.1 Player is compatible with System 7.1, but HyperCard 2.1 does not support WorldScript and neither does the player. Claris is currently developing a new version of HyperCard Player that will support WorldScript. Claris also announced that future versions of HyperCard will offer color, more robust developer support, and other new features. ◆

Apple Announces ADG-Europe, Appoints Manager for ADG-Japan

Apple recently announced the formation of the Apple Developer Group–Europe (ADG-Europe) and appointed Wayne Surdam to manage the newly created ADG-Japan.

In announcing ADG-Europe, ADG Vice President Kirk Loevner said that the new group will “help raise the level of services to European developers and provide an increased focus on developer activities in Europe. The benefit will be more local European software, faster localization of U.S. software, and consistent delivery of support services worldwide.”

Loevner added that ADG intends to enhance its Cupertino-based resources that are directed at fostering the European market. This includes providing additional resources for U.S. developers trying to penetrate European markets, technical support for European developers, and additional programs for European developers.

The new group will operate out of Munich, Germany, and will report directly to Kirk Loevner in Cupertino, California, and to the General Manager of the Apple Business Services Group in Europe, a position currently held by Sören Olsson on an acting basis.

In a separate but related move, Wayne Surdam has been appointed to head the new ADG-Japan, which is based in Tokyo. Surdam started off at Apple four years ago in the Apple Developer Group and subsequently moved to Apple Pacific marketing. He will report directly to Kirk Loevner and to Apple Japan executive management. ◆

Macintosh Duo and Macintosh IIvi/IIvx Clarifications

The Macintosh Duo Dock Fact Sheet on page 7 of the October 1992 *Apple Direct* is in error in saying that the Duo Dock supports SCSI disk mode. SCSI disk mode makes sense for the portable Duo MiniDock but not for the Duo Dock, which is essentially a desktop unit. Third-party docks can be designed to include SCSI disk mode, but it is not available automatically just because the dock has a SCSI interface.

A formatting error in the Macintosh IIvi and IIvx Fact Sheet on page 10 of the same issue may have left some readers uncertain about whether the Macintosh IIvx has a cache. The correct information is as follows: 1) the Macintosh IIvi cannot have a cache (not even as an option), while the Macintosh IIvx has a 32 KB cache; and 2) both the Macintosh IIvi and IIvx come with 4 MB of main memory, and both can be expanded to a maximum of 68 MB.

The same table is also incorrect in saying that the Macintosh IIvi/IIvx on-board video supports the Macintosh Portrait Display. However, you can connect either computer to this or any other large-screen display by adding the appropriate NuBus™ display card to one of the computer's three internal NuBus slots.

We regret any confusion these errors might have caused. ◆

Apple Font Pack Ships

Apple introduced the Apple Font Pack in October. The retail product includes 43 fonts, including 25 new Apple fonts and 18 of the fonts previously shipped with the LaserWriter and StyleWriter printers. With the release of the Apple Font Pack, Apple now offers a total of 64 fonts for the Macintosh computer.

The Apple Font Pack, designed for mainstream customers, requires minimal system overhead, and makes fonts easy to use with one-click installation. It also includes a booklet that educates customers on how to effectively use type to create compelling documents. The product comes with one year of toll-free telephone support for customers in the United States. The Apple Font Pack also includes the Apple LaserWriter Driver and LaserWriter Font Utility software for using the fonts with Apple and third-party PostScript laser printers.

“Our objective is to make it easy and affordable for any Macintosh customer to take full advantage of the powerful typographic capabilities of System 7,” said Roger Heinen, senior vice president and general manager of Apple’s Macintosh software architecture division. “With the Apple Font Pack, customers have convenient access to a carefully selected library of exceptional fonts that will deliver impressive results.”

The 25 new fonts will also be added to the library of fonts included with Apple’s network printers—the LaserWriter IIf, LaserWriter IIg and Personal LaserWriter NTR, boosting the font offering to 64 for those using any Macintosh computer with these printers. ◆

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 October 1992.

Publisher	**Product (version)**
Continental Press, Inc.	SchoolPRO: Integrated Tools for School Data Management (2.1)
Computer Literacy Press	Hands-On ClarisWorks (book + template disk) (1.0)
Dynamic Designs	ScriptGen (2.0)
Emergent Behavior	MicroGA (1.0)
Frostbyte Software Inc.	Event Magic (1.0)
LinksWare Corporation	LinksWare (2.0)
No Hands Software	Magnet (1.0)
ON Technology, Inc.	STATUS•Mac (3.0)
OBjectic Systems, Inc.	Fast Pitch PRO Videodisc Plug-In (original release)
Optima Technology	DeskTape (1.4)
Palomar Software	PLOTTERgeist (2.1)
Pericles Software	TabHouse (1.0)
Quadmation, Inc.	ZPS-230fx (A)
Radiant Enterprises, Inc.	CommonSense CNX (7.2)
T/Maker Company	ClickArt Artistry & Borders (1.0)
Utilitron, Inc.	Guaranteed Undelete (2.0)
	PowerSwap (1.0)
Vineyard Software, Inc.	FreePan Pantone Color Matching System (1.0.0)

	FreePan Focoltone Color Matching System (1.0.0)
	FreePan Toyo Color Matching System (1.0.0)
	FreePan Truematch Color Matching System (1.0.0)
Weingarten Gallery	Font Air Force (1.0)

Transparency, or
“Death Comes to Bob the Waiter”

In a world of cross-platform connectivity and 8 MB word processors, the interface solutions of yesterday just aren’t enough. If we want to deliver the power that our customers demand, while still making our applications usable, we’re going to need some new ideas.

Back in June, I proposed five new interface design techniques for dealing with our complex world. These were:

•Constraints: to lead users through a system by restricting their choices (I wrote about this in the June 1992 *Apple Direct*)

•Intelligence: to take “busy work” out of the user’s way (*Apple Direct,* August 1992)

•Elegance: to give grace and simplicity to our designs (*Apple Direct,* September 1992)

•Attention to detail: to make our interface—our “user illusions”—believable (*Apple Direct,* October 1992)

This month, I’d like to turn to the last of the five techniques, namely transparency: to keep the interface itself out of the user’s way.

Transparency is a bit of a sticky concept in human interface. To give you an idea of this, there’s been a raging debate about whether interfaces should be transparent or translucent. Academic debates like this are amusing in psychology classes, but they don’t really give you a feel for how your software should be designed. As such, I’m just going to talk about transparency by using the trusty “restaurant analogy”…

“Hi, I’m Bob! I’ll be Your Waiter for This Evening!”

A transparent interface is like a great waiter: constantly attentive without really being noticeable. Great waiters serve your food, refill your glass, and clear your dishes without ever interrupting the flow of conversation. They let you enjoy every moment of the dinner, from their graceful presentation of the menus, to the moment they elegantly present you with the bill.

Unfortunately, great waiters are none too common. People have to go to school to learn to be great waiters, and even then it’s usually a sort of initiation for first-year culinary school students.

The people who normally serve us are terrible waiters. And the very worst waiters are found in North Hollywood restaurants when you're trying to have a romantic tête-á-tête. They're the waiters that shatter any sense of romance by stopping by every five minutes to ask, "How's everything going?". The name of every one of these waiters is Bob. And they're really actors.

Bob is not, to put it nicely, transparent.

The ideal interface, like the ideal waiter, is one that you don't have to think about. We would say such an interface is transparent—not because we can't see it, but because we don't really think about it. With such an interface, our full attention is geared toard getting our work done, instead of just working the interface.

One of the biggest barriers to transparency today is the mere fact that so much of our work with computers requires a keyboard. In the future, technologies like voice recognition and pen input will help people to worry more about the work they're doing and less about how to convince the computer to do it. Needless to say, Apple is very interested in this.

In the meantime, your application can become more transparent by following a few simple rules.

Rule 1: Hide Features in Plain Sight

Strangely enough, the first step toward creating a transparent interface is to make the interface visible. Don't make your users remember secret symbols, gestures, or commands to do their work. If to add page numbers to a document users have to look on page 103 of the manual, then type the secret key they find there (Option-Shift-P), they're not going to remember what they were writing in the first place. If you make the interface visible, users don't have to waste time on these "command safaris."

Rule 2: Avoid Computerese

Don't have your program babble on in computerese unless the users are programmers. Don't say "Query" when you could say "Find." Don't say "Access" when you could say "Open," and don't say "SysErr Code:–34" when you could say "The disk is full."

RULE 3:
KEEP STATUS MESSAGES SIMPLE

Don't burden the user with needless programmatic detail when giving status messages. Let the user know (in a general way) what's happening, and how long it will take. It doesn't help most users to know that your telecommunications program has just "Initiated CCL Script Parsing" and will soon be on its way toward achieving "Name Server Access." Give frequent status information, but keep it simple enough to understand. If you can't think of anything more meaningful, just say "Connecting: Step 1," "Connecting, Step 2," and so on.

RULE 4:
DON'T INTERRUPT (OR IF YOU MUST, DO IT QUIETLY)

Our friend Bob the actor-waiter loves to barge in on the middle of romantic conversation to ask, "And how's everyone's warm duck salad tonight?!" This is why we hate Bob.

Think about this before you use the Notification Manager to display a modal dialog box from the background. Is what you're saying that important? Or would blinking the application menu icon do just as well?

Clinical studies show that whenever a background process (like a waiter) interrupts a foreground task (like hand-holding), the result is an increase in the stress level of the person performing the foreground task. This response is in direct proportion to the level of intrusiveness. So if Bob is going to loudly interrupt a romantic moment, he'd better be telling us our car is being stolen. In that case, we can channel that stress against the thief.

Otherwise, we'll direct it at Bob.

THE EXTRA HI DESIGN MILE

It's sobering to remember that both the original Macintosh System Software and a word processor fit on a single 400K disk, worked in 128K of RAM, and could be learned in minutes. What a long way we've come from that in just eight years.

If we want to keep the same ease-of-use in today's complex world, we're going to need some new ideas. I've pontificated on some of my own, but now it's time to listen to you. Don't forget, this is your column, too. Sure, we interface folks may pound the soapbox for an issue or five, but I'm also here to help with

your questions and concerns. So send me an AppleLink message at THE.DOKTOR, and I'll do my best to answer you.

Till next time,

—Doc ◆

Pete Bickford runs the Human Interface Lab at Apple's IS&T (Information Systems and Technology) organization.

Apple Adds Single Domain X.400 Product to OSI Line

Apple has recently made available MacX.400 SD (Single Domain), an affordable X.400 server for communication among proprietary electronic mail (e-mail) systems. The new offering turns the Macintosh computer into an X.400 server that connects to a single X.400 server, bringing a standards-based, easy-to-use and versatile e-mail system to small and medium-sized businesses.

MacX.400 SD, like MacX.400, is open to developers. The MacX.400 application programming interface (API) allows programmers to implement X.400 gateways for existing e-mail systems and create X.400 user agents for the Macintosh. The MacX.400 API is the same for MacX.400 and MacX.400 SD servers.

The MacX.400 API is available to client applications in one of two ways:

First, it may be on the *same* Macintosh as the MacX.400 or MacX.400 SD server. This type of configuration is used by developers who create e-mail gateways. The API may also be on any Macintosh *connected* to the MacX.400 or MacX.400 SD server. This scheme is appropriate for developers who design native X.400 user agents. In such a connection scheme, a software component provided with the MacX.400 API, the MacX.400 Mailbox Server, is in charge of distributing X.400 calls from the server to the client desktops over an AppleTalk network.

The new server is the latest in Apple's Open Systems Interconnect (OSI) product line. Apple's OSI line includes the first ever X.400 server for the Macintosh (MacX.400), a software translator that implements the Open Document Architecture (MacODA), a transport platform for OSI environments (OSI Connection for Macintosh—formerly called MacOSI Transport), and Apple's implementation of the X.25 standard for connecting Macintosh computers to packet-switched networks (MacX25).

MacX.400 SD is optimized for smaller businesses, bureau offices, or workgroups that need to communicate with only one X.400 server. Otherwise, MacX. 400 SD offers all communication capabilities of the MacX.400 product.

The MacX.400 products enable Macintosh users to send e-mail to recipients on other platforms connected to X.400 networks. Used together with an e-mail gateway, either server can distribute X.400 services to mail users over an AppleTalk network or other networks. This type of configuration allows the

exchange of e-mail among recipients within a country or worldwide. MacX.400 SD can also be used inside large companies to connect Macintosh workgroups to other platforms through the corporate X.400 backbone.

Both the single-domain and multiple-domain products support the 1984 X.400 standard, including the P1 and P2 messaging protocols. Each product comes with a full OSI stack, which allows users to connect to local and wide area networks. MacX.400 and MacX.400 SD support the ISO Transport Class 4 protocol stack for communication over IEEE 802.3 Ethernet, and both enable the exchange of e-mail with remote sites over X.25 networks (when used in conjunction with Apple's MacX25).

APDA offers three MacX.400 kits for developers:

- the MacX.400 Programmer's Kit, which includes the MacX.400 API, the MacX.400 Mailbox Server and the *MacX.400 Programmer's Guide*
- the MacX.400 Developer's Kit, which includes the MacX.400 SD server and the MacX.400 Programmer's Kit
- the MacX.400/Ethernet Developer's Kit, which includes the MacX.400 SD Server, the MacX.400 Programmer's Kit, an Apple Ethernet NB card, and an Apple Ethernet Thin Coax Transceiver

For APDA ordering information, Now Available From Apple. ◆

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).

November 16 through 20
C++ World
Secaucus, NJ
Contact: G.G. Schafran
(212) 274-0640

November 16 through 20
Comdex
Las Vegas, NV
Contact: Interface Group
(617) 449-6600

November 18 through 23
NCTE - Nat’l Council of Teachers of English
Louisville, KY
Contact: NCTE

November 20 through 23
NCSS - Nat’l Council of Social Studies
Detroit, MI
Contact: (202) 966-8740

December 1 through 4
Cause
Dallas, TX
Contact: Cause
(303) 449-4430

December 8 through 10

FCC - Federal Computer Conference

Washington, D.C.

Contact: Information Development Corporation

(301) 961-6575

January 5 through 9

IDC

International Disability Conference

Cupertino, CA

Contact: Karen Normandin

AppleLink: NORMANDIN

(408) 974-1248

January 6 through 9

Macworld

San Francisco, CA

Contact: Mitchel Hall Associates

(617) 361-8000

January 7 through 10

CES

Consumer Electronics Show

Las Vegas, NV

Contact: Eliza Lape

AppleLink: ELIZA

(408) 974-1248

February 24 through 27

NABE

National Association of Bilingual Educators

Houston, TX

Contact: Javier Villalobos

AppleLink: VILLALOBOS1

(408) 862-6426

February 24 through 27

NAIS

National Association of Independent Schools

New York, NY

Contact: Sheila Milligan

AppleLink: MILLIGAN.S

(408) 974-5891

March 27 through 29

ASCD

Washington DC

Contact: Sheila Milligan

AppleLink: MILLIGAN.S

(408) 974-5891

March 30 through April 1

Intermedia

San Jose, CA

Contact: Dave Billmaeier

AppleLink: BILLMAEIER1

(408) 974-6553

March 31 through April 3

NCTM

National Council of Teachers of Mathematics

Seattle, WA

Contact: Jeryl Gerhardt

AppleLink: JERYL

(408) 974-2368

NCTM is (703) 620-9840

April 14 through 16

Seybold

Boston, MA

Contact: Tara Vincent
AppleLink: TARA
(408) 974-4464

April 26 through 30
IRA
International Reading Association
San Antonio, TX
Contact: Tara Vincent
AppleLink: TARA
(408) 974-4464

Target Markets: Easy to Pick, Hard to Stick

How to Choose and Stick With the Best Target Market

By Geoff Moore, Geoffrey Moore Consulting

Brrring."Good afternoon," I said to my caller. "May I help you?"

"Geoff. It's me, Jim, your old buddy."

Drat. Good old Jim, always good for a lot of questions, but never a nickel in payment. "Hey, Jim, what can I do for you?" I asked, mentally cursing myself for giving him such an easy opening.

"Well, now that you mention it, I do have a problem. And actually, since it grew out of reading your book, you sort of owe me on this one.

Here it is: Remember when we last talked and we agreed that my product was in the chasm?" (The *chasm* refers to the gap in a product's life cycle between early success, spurred by sales to technology enthusiasts and visionaries, and the later mainstream market success that comes from making sales to mainstream pragmatists and conservatives. For more information, see "Crossing the Chasm: Moving From Early Success to Mainstream Market Leadership" in the May 1992 issue, and "So What's Your New Product About? Position It Using the Elevator Test" in the February 1992 issue of *Apple Direct.*)

"Yes, I think so," I responded, racking my brain to remember which product he was talking about. "The CD-ROM information base, wasn't it?" It was coming back to me. Jim had acquired the rights to a lot of 1990 census information and had packaged it for use in market analysis applications.

"Yeah, well, remember how you said the key to crossing the chasm was to pick a single target market and focus on getting a leading share within it?"

"Sure," I replied, reminding him of the D-Day analogy in which the Allies focused all their armies on taking Normandy Beach, even though their long-term goal was to liberate all of Europe.

"Yeah, well, it doesn't work."

"What doesn't work, Jim?"

"This segmentation stuff. We picked a segment, but it didn't work."

"What did you pick?" I asked, seeking to determine the precise number of paper clips it takes to build a two-foot chain.

“We picked information workers who want customer data for market analysis.”

WHAT CONSTITUTES A SEGMENT

“Hold on,” I said. (Now he had my attention.) “Remember, there are two key things that define any high-tech market segment. First, everyone in the segment must have a common use for the information. This ensures that as additional functionality is added (during further development or through partnerships or “bundles” with companion products and services), these additions all remain focused on—and offer more value to—a given set of customers. On this score, it sounds like you do have a segment.

“But the second criterion is that the group you’ve defined must constitute a single word-of-mouth community. The people in that segment must reference each other (that is, consult with each other) when making a buying decision about a high-tech product. This ensures that early customers spread the word to mainstream users more quickly, helping to create the perception that you are the market leader.

“If all the early customers are in one segment, you will very quickly develop the reputation of being the up-and-coming solution that fills this segment’s needs. But if each early customer is in a different word-of-mouth community, then each is a lone voice in the wilderness, and no perception of market leadership can be created.

“The problem, Jim, is that ‘information workers’ isn’t a community in the sense of being a group of people who would consult with each other about high-tech purchase decisions. So let’s take a step back and see what word-of-mouth communities you have touched. Who was actually using your product to analyze customer data?”

“Well, let me see,” Jim replied. “There was the product marketing department at a major consumer products company and the research department at an ad agency; a political party is a big user, and also a hospital looking at the demographics of its customer base.”

“That’s at least three, maybe four different word-of-mouth communities,” I replied. (Consumer products people talk with ad agencies a lot, so they might be considered parts of a single segment.) “You have to pick a *single* segment to focus on—for example, marketing managers at hospitals or in consumer products companies.”

“That’s ridiculous!” he exploded. “If I did that, I’d have no sales at all. I have to make at least $1 million from this product during this fiscal year or I’m in big trouble.”

“How much have you made so far?”

“Well, our sales are just getting ramped up. . . . Uh, we’ve sold several hundred copies.”

“At what price?”

“It lists for $2,450.”

“But perhaps you’ve offered discounts to early customers?”

“Well, we’re trying to seed the market.”

“So . . . ?”

“We’re almost to $100,000 in sales.”

And this was the fifth month of the quarter. “OK, Jim. Sit back and try not to interrupt for a minute, although you aren’t going to like what you hear.”

WHY ONE IS ENOUGH

“Let’s revisit some basic principles,” I said. “The whole point of marketing is to achieve the benefits of market leadership—premium pricing, higher profit margins, lower cost of sales, strong word-of-mouth recommendations, and the like. The only way to get those is to be perceived as the market leader; and the only way to become that, when your company is small, is to focus on a single market segment. It’s a big fish/small pond game. If you don’t manage the size of the pond, you’ll end up being just another minnow in the ocean with a very short life expectancy.”

“But I need the revenues!” he howled.

“Don’t we all. But you have to *earn* them. I’ve seen your product and it’s very good—that’s the first step toward earning your right to the customer’s money. But you also must establish a market leadership perception for the product. Without that, only the early adopters will be willing to risk buying it. Everyone else will see it as a fad and will steer clear.

“But if you can get at least one group of customers to adopt it more broadly, then you have a beachhead from which to attack the mainstream marketplace. Focusing is a shortcut to establishing the beachhead; from there you can expand onto the continent—into the mainstream—carrying the leadership mantle with you.”

"OK, OK," he said. "Just suppose I go along with you on this. How do I do it? Say I just pick consumer product marketing managers, for example. Now what?"

"Careful," I warned. "There's a problem with 'just picking' a target market segment: For the focus to work, it must be sustained for at least a year."

"Why a year?" Jim asked, typically *sotto voce*.

"It typically takes 90 days to develop a marketing communications roll-out that solicits initial sales for any target segment. Another 90 days passes before you can measure any impact the program has. But to get a critical mass that begins to convey that you're a market leader, it usually takes another six months," I explained. (What I didn't say, because I suspected Jim wouldn't be thrilled to hear it, is that this applies only if you target a reasonably modest-sized segment.)

Why Stick? "So to get any results," I continued, "everyone in your company has to buy into the choice of target segment and stick with it. Most targeting efforts fail because the developer keeps switching from target to target (I call it the 'target du jour approach') and achieves no focus at all. And part of the reason for not sticking to any given target is that the developer's selection process was insufficiently rigorous to choose a segment that it could make a long-term commitment to."

"Well, in *my* company, whatever I say goes, of course. So that shouldn't be a problem," Jim said.

Right, I thought. "Yes, it should be a problem," I argued, "because a) you may end up second-guessing yourself, and b) to truly be successful, a company requires internal consensus when it comes to the critical factors."

How to Select a Target

I continued: "The point is that you need a reliable mechanism or process that allows you and your key managers—your 'strategy development group'—to survey a broad range of opportunities and come to a rational decision about which beachhead target segment to focus on. The process must generate a focal point that your entire team will commit to for at least a year. The method I recommend is to generate *target application scenarios*."

I explained the concept to him: The reason we use such scenarios is to help overcome the basic problem in all market segmentation work. The boundaries

of any segment are inherently fuzzy, and if you focus on defining a segment by its boundary line—who's in or out—you'll end up in a perpetual debate. So instead you should define segments by their center point.

The center point of any segment consists of an ideal customer using the product in an *application* (not in the computer program sense, but in the "how a product is used" sense) that produces the maximum benefit. Describing that customer and application is the function of an application scenario.

The scenario is a before/after story. That is, first you describe the target user and "a day in the life *before*" he or she gets your new product, focusing on some excruciatingly frustrating moment when, because that person doesn't have your product, things turns out very badly. Then you describe "a day in the life *after,*" replaying that same moment as if the user has your product—so that things turn out very well. The resulting document, which should be no more than one page long, is the target application scenario.

Each scenario should represent a possible target market segment. The idea is for your strategy development group to generate as many scenarios as you can (40 is typical, 70 is more than enough, 20 is probably too few) until you collectively feel you've exhausted all the imaginable possibilities. Then you evaluate each one and finally select the single best target.

Each scenario stands for a possible target market segment, and you rate its attractiveness in terms of the following criteria using a 1 to 5 rating, with 5 being best:

•*Compelling reason to buy.* The question here is how strong this potential customer's motive is for adopting your product. If it is nice to use your product but not critical, then rate it low (maybe 2). But if the product solves a crucial, persistent problem that directly prevents users from accomplishing their primary functions or goals, then the scenario gets a high rating, typically 4 or 5.

•*Whole product feasibility.* The whole product is your product plus all other products and services that target customers need for fulfilling the compelling reason to buy. The question is, how realistic will it be for the user to get the complete solution set required to achieve his or her compelling reason to buy?

If, with the addition of your product, it can be done "off the shelf" using existing products in a plug-and-play way, then it is highly feasible and gets a 5. But if it requires special programming or an unusual set of companion products, then it gets a low rating.

•*Partners and allies.* Most whole products result from the cooperation of other developers or vendors to provide necessary services and support, as well as the application-specific hardware and software interfaces needed to connect your new product into the customer's installed systems.

How well known are you to these vendors, and vice versa? If you are using proven relationships to make the scenario work, then give yourself a high rating. If your scenario requires developing relationships with a new group of vendors, give yourself a low one.

•*Whole product pricing.* Pricing is critical to market development; however, for the purposes of the scenario the key is not the price of your product alone, but the price of the whole product solution. If the whole product can be delivered at a market-making price point, then give the scenario a high rating; otherwise, give it a low one.

•*Whole product distribution.* Distribution (or where customers buy the product) is also crucial to making a market take off; but again, it is not only how *your* product is distributed but also where the customer would buy the whole product that is important. If there is a natural, one-stop place for the customer to purchase the whole product and you already have established a relationship with that channel, then the scenario gets a high rating; otherwise, rate it low.

•*Competition.* The question here isn't "Is there an existing product that has similar features and benefits to yours?" Rather, it is "Is there a developer who has already won your customer's heart with respect to the target application?" That is, you shouldn't worry about whether there are other suitors with your qualifications wooing your customers. Instead, worry about whether the target of your affections is already married. If the user is already well served, give the scenario a low rating. If not, give it a high mark.

•*Positioning.* Is your com-pany's current image consistent with the role you must assume in serving the target customer as specified in the scenario? For example, if you're like me, Newton from Apple feels good. But how would you feel if Apple introduced a mainframe? If what you are proposing is the sort of thing you have succeeded with in the past, give it a high rating. If you are entering a new domain, give it a low one.

•*Leverage.* Is this market segment a dead end or can you leverage it to win additional segments? Suppose you became the market leader in a market segment: Can you leverage this lead into other segments, say, the way Apple leveraged success with corporate graphics departments (desktop publishing)

into success with designers in advertising and prepress companies—as well as into success with other sales and marketing professionals? If so, the scenario gets a high rating for leverage. But if the market segment appears to be a dead end, then give it a low rating.

"That's the rating system," I concluded. (For reference, a free copy of a hypothetical case study and three sample scenarios are available from Geoffrey Moore Consulting.)

"Great," Jim said. "So, let me get this straight. We write up the scenarios and rate them. Then what?"

GET OUTSIDE VALIDATION

"Well, after you have rated all the scenarios, throw away all but the top few. Your final target should be taken only from the top two to five scenarios. But before you make the final choice, you must do some market research."

"Market research—as in pay money for?"

That's my Jim. " 'Fraid so. See, up to this point all you've used is your own intuition and experience. That's fine for cutting your search down to the final few. But before you make the final call, you really ought to get some outside input as well."

"Such as?"

"Typically, you should interview five to ten users for each scenario that's still in the running. To help ensure that you get the right input, write a two- to three-page market development strategy statement for each scenario; it should consist of a profile of the target market, including paragraphs that address each one of the eight criteria in the target application scenario. Give this to your interviewer and ask that person to test its validity through phone conversations or focus groups with people who fit your target user profile.

"You also should talk to three to five potential partners and allies who are already serving the target customer, just to see if your strategy makes sense to them. This is particularly important if you would be highly dependent on any particular partner to make the whole product successful.

"Once you get the input from each of the two to five 'final candidates,' you're essentially done with the preparation phase; then you make your choice. The good news is that having reviewed your options this thoroughly, you are well

positioned to make the choice stick. The bad news is that the choice to focus is never comfortable."

"Why's that?" he inquired.

Focus Works

I explained that no matter how much you buy into the concept of market focusing, you'll always worry that by doing so you are somehow giving up a sale that you otherwise might have gotten. It's true: You are.

But console yourself with the thought that a sale within your target segment not only yields revenue, but also contributes to market leadership perception—and that's where the real return on investment lies. Only market leaders can charge the premium margins that make them profitable in the long run.

"I don't know, Geoff. The whole process seems too complicated to me."

"Yeah, I sympathize. But the alternatives are worse. If you don't do this kind of thorough review, you won't have a strong basis for sticking to your selection. The next time the going gets rough or a hot new opportunity pokes its head up, you'll more easily give up your focus and begin to pursue a new target. In the end, you'll never focus at all.

"The key thing to remember is that people fail at focusing not because they pick the wrong target—the truth is, any reasonable target will work and even a suboptimal target is better than none at all—but rather because they keep jumping from target to target, never sticking around long enough to develop a real market-leading position."

"All you are saying is, 'Focus works.'"

"Yeah."

"Focus works."

"Yeah, you got it, Jim. . .

Bye." ◆

Geoff Moore is the owner of Geoffrey Moore Consulting, a high-tech marketing consulting company based in Palo Alto, California. He also offers a variety of marketing seminars and is the author of Crossing the Chasm: Marketing and Selling Technology Products to Mainstream Customers.

Licensing Software Technology

Building a Company With No Venture Capital, No Loans

By Shawn Abbott,
the AND group inc.

If your company's primary asset is unique programming skills, you don't necessarily have to build a traditional business infrastructure to transform development effort into working capital. In some cases, the best way to get your technology into users' hands may be licensing—a tactic that can infuse your company with the capital needed to grow and continue development.

This can be important to a smaller company just setting up shop. The reason: In our experience, selling a technology to a small number of licensees is less resource-consuming than trying to hit a larger target market.

The AND group, like many software companies, started business by securing small development contracts with local companies. One of these projects (circa late 1988) was to develop a system to electronically distribute another company's Macintosh typeface products. During the course of the project we saw an opportunity to create a unique technology—a system for distributing software in encrypted form on CD-ROM.

This is how it works: End-users buy a CD-ROM disc (or floppy disk) containing a variety of software. To "unlock" the desired pieces of software, they purchase an access code via a telephone call.

It took no more than a second to realize we were not the right firm to market the technology to users. Instead, we decided to license it to other companies.

WHY LICENSE?

There are diverse challenges in running a successful business. Human resource management, marketing, sales, finances, and development are all important. But for our team of "crack programmers" to succeed, we felt we should concentrate on our strength (programming) and reduce the dependence on our weaknesses (those other diverse challenges). Licensing seemed to be the best alternative for several reasons:

• *It allows us to remain self-directed.* Licensing dovetailed nicely with our main business goal: to be self-directed. To be so, we needed to build the company without venture capital or loans of any kind.

Although working with limited capital sometimes requires a painful and longer-than-usual startup period, a completely self-owned company without debt has fewer burdens to interrupt development—no quarterly shareholder presentations or artificial milestones to satisfy management, and no threats of manipulation from internal groups not actively involved in technology direction.

•*It allows us to focus limited resources.* A major benefit of licensing is that it limits a firm's responsibilities by eliminating such traditional tasks as sales and marketing. Although we must find and serve licensees, overall we have been able to devote more resources to what we do best: development. This means we have a much simpler business plan than we'd need if we also concentrated on tasks not related to development.

•*It gives us more flexibility.* Having a simpler business plan means that if economic or market conditions necessitate it, we can alter the company's direction more quickly and with minimum fuss. Since our resources are highly focused and there is less infrastructure (almost everyone is in our "development department"), there are far fewer constraints on changing our overall plans.

•*Our cash flow is more predictable.* Licensing agreements give us a more controllable cash flow, rather than forcing us to wait for and depend solely on sales of a boxed product. We can negotiate such things as advances on royalties to help pay up-front development expenses.

•*We can more effectively focus on long-term objectives.* Having a more predictable cash flow also allows us to continuously and effectively work on long-term development plans, in addition to meeting immediate revenue goals. Sticking to a strategic, long-range plan is critical but difficult for a small company that is heavily influenced by immediate needs. We've found that licensing makes long-term growth easier to manage. Ultimately, this gives us a competitive edge.

•*It extends our product reach with minimum marketing effort*. Because our product is incorporated into another company's software that is subsequently sold to much larger market segments, in effect we have an installed base of several hundred thousand users. But the AND group itself has only about a dozen direct customers to support. Because licensees—who usually are more

expert in serving larger market segments—incorporate our technology into the products that they sell, we don't need to be an expert in serving a mass market.

•*We get preferential tax treatment.* In Canada, we receive preferential tax treatment for being research and development based. The eligibility requirements are complex, but meeting them results in tax credits and refunds. While this treatment may not be available everywhere else, we strongly recommend investigating the possibilities in your country.

DEFINE YOUR OFFERING

Is your work suited to licensing? There are several things to consider. (For more information, see "Is Your Work Suited To Licensing?" below.) If so, a critical success factor is understanding what you have to offer a customer (even though at first it seems a little intangible) and matching it to customer needs.

You may think you know exactly what you are selling, but putting it into customer-benefit terms may be difficult. Therefore, an important goal is to try to understand a prospective customer's business as well as possible.

While each customer's technical needs are unique, there are some common expectations. Clients often want products that can be customized or localized. To get the most out of your investment, design your product from the start to make the inevitable localization easier.

Also, customers want a *complete* package. Even though you probably aren't shipping a shrink-wrapped box, your offering must be complete. Don't offer clients solely a piece of object code on a floppy disk. Regardless of how useful the code is, on its own it will have no value to anyone. Instead, offer a package that is as complete as any other software product on the market, including bound documentation.

This doesn't necessarily mean that you must "do more." The key is to *do less,* but do it *better*—a fundamental concept that should be ingrained in every facet of client service. Your value and performance will not be measured by *how much* you provide, but *how well* you provide what is needed.

To protect everyone involved, be crystal clear about what you're offering (product, extent of services, deadlines, and so forth). Be articulate about what part of the "big picture" you will be responsible for—and be willing to make some guarantees about what, how, and when you'll deliver.

Communicating the Offer. Once you nail down the offering, appoint a spokesperson who can present the product to clients in a way that is meaningful to *them.* The presentation should be tailored to the audience, depending on whether you're speaking with fellow technical people, marketing gurus, or others.

While there are many subjects a presentation might cover, there are two points that, if not covered adequately, will thwart your pitch: exactly why clients need your product (that is, how your product enhances or benefits theirs), and why they need *you* (as opposed to doing it themselves). If you clearly communicate these concepts, you'll be halfway to the bank.

GETTING THE BEST CLIENTS

The Philosophical Outlook. While many business models call for the *most* customers possible, we want the fewest but *best* clients. For us, the best clients are those who are visibly willing to build high-quality products around our work and who devote the needed time and resources to creating a win/win working relationship.

The right companies also respect our goals: Our primary asset (development team members) should be challenged and excited, the technology equity should be of the highest quality, and the corporate entity must remain profitable.

You'll know you have approached the right people when they start the negotiations by talking about a desire to create a win/win licensing agreement, rather than by asking your price. When the conversation is two-sided and feels more like a discussion about partnership than buy-sell, you'll have a better chance of building a mutually satisfying agreement.

Our ideal is to have only one client per each person we employ. By focusing on a few clients, our small company can give them the best service possible. The revenue may not be as high as it could be with a larger clientele, but the company will be a success in the long run.

The Practical Outlook. We have learned a few very practical lessons from our experience. For example, a very small company and a large multinational one working together potentially creates the most difficult situation—but it can also be the most rewarding. Because of the sharp contrast in views about how business should be done and the potential mismatch between expectations and

capabilities, unique individuals are required on both sides to make this kind of arrangement successful.

These kinds of clients—large, multinational companies—are well worth pursuing, but make sure that the match is truly good for you.

We don't recommend working with small companies that are similar to yours. Although we have seen others succeed with that arrangement, we feel you have much less to learn or gain from a similar firm: Neither of you has particularly distinctive capabilities or approaches to offer each other.

On the other hand, larger companies that operate very differently from yours probably have the most to gain by working with you, since you offer a special, focused expertise that they don't have. Naturally, you'll also want to work with a company with deep pockets and a successful track record.

How to Find Them. The optimum approach is to make your company visible so potential clients seek you out, not vice versa. The best relationships usually result when a prospect contacts you.

However, newcomers will probably have to search for prospects. There are several good starting places. For example, we attend the annual Apple Worldwide Developers Conference, a must as far as we're concerned because we can accomplish two things there: Apple gives attendees technical and market insights about what its plans are and what's hot (which can help in our development and planning efforts), and we can meet with fellow technical people to begin talking about our products.

Approaching technical decision makers is important. Let's face it: As programmers we have a much better idea of how to excite other programmers than we do marketing people. If you can pique the interest of technical people, they can help translate your offering into benefits that are meaningful to the other decision-makers in their companies. You also will have created an in-house advocate who can teach you about what his or her company's needs are—in a language you can understand.

After you begin building up steam, expand your search. Conferences such as the Macworld Expo are excellent places for this. In our experience, you don't need to rent booth space; the cost is high for a small company, and instead of manning a booth you need to spend your valuable time visiting prospects' booths. Visit all the booths you can to help determine who might be likely

licensees. This is a golden opportunity, because your targets are all in one place at one time.

To help determine who the best prospects are for your company, first decide exactly what you can do better than prospective clients, and use that information in every step you take. Make a profile of your best prospects; learn everything you can about how they would use your technology.

Focus Your Efforts. Because in every sense it is important to leverage your efforts, you may be wise to first saturate one vertical market in which you know you can do very well. Limiting the applicability of your technology to a single market segment significantly simplifies your customer service burden. Don't try to create broad appeal until you have an established market presence and a table cash flow.

At the same time, don't limit yourself to the local market; play internationally. Overcoming distance barriers to reach prospects in a familiar, vertical market segment is much easier than trying to understand the needs of a local market segment that is new to you.

CREATING THE RELATIONSHIP AND THE LICENSE

The Relationship. Many things affect customer relationships. Here are a few key guidelines that have helped us foster better working relationships—and maintain our sanity through the rough times:

• *Demonstrate your professionalism as an organization.* You should be willing to present your programming standards for inspection by the client. A procedures document that outlines how you approach software development is a valuable mark of excellence. If you can demonstrate that your group has a solid ability to manage the *process* of software development, you'll gain the much-needed respect necessary to bargain from a strong position.

• *Earn dollars through your clients, not from them.* This means you must be willing to put your neck on the line by tying your success to that of your client's product. This helps create a positive working relationship between your companies and makes you more profitable in the long term. An advance on royalties can help sustain your company (and your sanity) until the cash from your client's sales starts flowing in.

• *Establish clear directions and goals.* Let your clients know what your expectations are. Put everything you can on the table, and make sure you both agree about who does what, how, and when.

• *Meet with prospective clients face-to-face regardless of the travel cost.* This is a clear signal that your client is important to you. It also indicates the type of relationship you expect to build. You can't do business effectively unless you know who the players are and can operate based on personal interaction.

The License. The specific issues surrounding how to create a good contract are too many and complex to cover here. This is an extremely important topic, however, and the best advice I can give in this limited space is: Get top-notch legal representation, and try to create a win/win situation for everyone involved.

To give you an idea of what the licensing process entails, here's a laundry list of some things you'll need to consider:

- dimensions of exclusivity
- time frames
- software escrows
- performance parameters
- pricing
- royalty structures
- long-term royalty versus advance, versus development fees

SET YOURSELF UP FOR SUCCESS

If your company is small and focused, it's especially crucial that it be structured and operated to get the most benefit from the least effort and investment. Of course, you need to hire the right people and create a strong team environment.

But beyond that, it is important to *work smarter, not harder.* Because your resources are lean and so highly focused, it is crucial that you use them optimally. You must be more efficient than your clients at doing this kind of development; it must be more profitable for them to work with you than to do it themselves. And to keep your efforts optimized, rather than hiring more people you'll need to work with *fewer people who are more experienced and qualified*—and equip them with the best tools possible and a productive working environment.

You also must make R&D a priority. Differentiate between R&D and product development; make it clear what this means to your team, and allocate time toward work that is not expected to immediately result in a commercial product. Otherwise, your technology will be leapfrogged by the competition. Thinking like an R&D company means you are willing to spend time and money on projects that may not succeed. Turning a profit in the face of this type of risk is not easy, but the potential payoff is high.

Furthermore, even though you are an R&D company, customer service still must come first. This is a difficult principle to practice and is always a topic of discussion (or debate) in our company. Our tendency, as programmers, is to think that customer service is not a responsibility in our day-to-day work. But nothing could be farther from the truth. You must establish a clear corporate mandate on the issue, define what customer service means to your company, and set employee expectations for performance relative to it. Establish an ombudsman for your clients and take ownership for client projects that use your work.

Quality is also paramount. In international licensing there is no room for anyone but the best. You must be able to demonstrate that your product is of excellent quality and that it will drive quality into your client's product. Within your company, focus on how your technology helps make the client's product better than a competitor's, and communicate this to your client.

At some point you will be forced to choose between getting a job done on time and doing exhaustive quality assurance. While timeliness is important, it doesn't compare with quality. The client will remember a job done quickly for a month; he or she will remember a job done correctly forever. While some people may disagree with this perspective, it has served us well time and time again.

LICENSE TO SUCCEED

Although licensing isn't right for many developers, it can be a viable alternative for companies that wish to boost their cash flow. Licensing has infused our small firm with the money needed to sustain development, allowing us to grow with no venture capital and no loans. Take that to the bank! ◆

Shawn Abbott is president of and a developer for the AND group inc., a development company that specializes in software distribution systems located in Calgary, Alberta, Canada. AppleLink: CDA0425.

**********Is Your Work Suited to Licensing?

Not all software is suitable for licensing. If you make a conscious decision to develop software for licensing right from the start, you'll be one step ahead. However, it may also be possible to craft licensable components from existing work. (Of course, licensing doesn't preclude producing boxed software, or vice versa.)

The possibilities are endless, but we've developed a few rules of thumb to help us determine whether a product is a likely licensing candidate. For example, reusable tools and engines are the best products for this. With a tool, licensees can perform a needed task (often, a repetitive one). And an engine can be used directly in a licensee's product. Our technology is used in both ways.

When creating a product for licensing, I suggest that you avoid writing platform-specific code (customers want code that can easily be ported) or producing gadget-like software, since its use may be limited and short-lived. Ideal licensing material is based upon solid academic research, which means you should be able to demonstrate an inherent value in the design of the software, as opposed to only in its implementation. ◆

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 the 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, Joel Cannon, Suzanne Dills, Chloe Freeland, Sharon Flowers, Andie Galt, Mark Johnson, Monica Meffert, Lisa Rose, Anne Szabla, Jessa Vartanian, Liz Westover, Diane Wilcox

MANAGER, DSSC:
David A. Krathwohl

CONTENT GROUP MANAGER:
Greg Joswiak

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

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