

Tri-Mode Publishing

The value of traditional ink-on-paper publishing is now dropping precipitously. For many users, online and CD ROM presentation of content has become overwhelmingly more attractive.

Why? Well, immediacy for openers. Unlimited full text searches. Interactivity. Hypermedia hot links to worldwide related material. Low production costs. Ridiculously faster turnaround. The lack of distributors and middlemen. Easy revisions and upgrades. No shipping charges. Full color at negligible added cost. Absolute editorial control.

Lack of any waste, shopworns, returns, or remainders. Historic archival access. Instant availability. Lower user costs. Tighter targeting. Immediate author contact. More compact storage. Always in stock and always "on the shelf". Fewer fingers in the pie. Lack of rejections. No competing ads. Optional sound, movies, and animation. Fewer federal and local hassles. Making your own mistakes instead of paying others to make them for you. Excellent customer feedback and demographics. Total reader attention.

The list goes on and on.

Let us assume you have long been doing traditional ink-on-paper publishing. And want to upgrade to what I'll call *tri-mode* publishing. Where your products are offered as paper, as electrons, and as plastic. Here are several guidelines that seem to be working for me...

Use Acrobat

Adobe *Acrobat* version 3.01 has been newly optimized for tri-mode publishing. The keys to Acrobat are more or less ordinary compacted text files called PDF files. A single object-oriented and device-independent PDF file contains everything that you will need to accurately and quickly build an arbitrarily high quality text, photos, and graphics page. A file that shows *only* what you want in the exact way you wish it to appear.

PDF files can be read with a free reader available from www.adobe.com Or from thousands of other sources. PDF files get created and modified with commercial software costing as little as \$55 or by using free *GhostScript* utilities.

Acrobat PDF files will print on any modern printer, but perform fastest and best with PostScript speaking devices. Color separations can be invisible and automatic. Special fonts are dealt with in a number of ways. Your two most popular are towing along the exact fonts needed internal to the PDF file itself; or to substitute reasonable lookalike but standardized *multiple master* font families.

There is never any need to worry about whether an end

user has the needed fonts. Nor whether any different font versions will cause appearance differences.

Web features of Acrobat include total hot linking, as well as the ability to do *full screen* displays from within popular browsers. Also provided are powerful searching abilities, magnification and text smoothing options, thumbnails, indexes, interactive forms, and even special text extraction aides for the disabled. New *byte range retrieval* features can deliver online documents on a speedy page-at-a-time basis.

Many popular applications can now directly generate Acrobat files. Others produce generic PostScript code which can get sent to a fast and friendly *Acrobat Distiller* module to do the conversion for you. Automatically or manually. You can also write your own PostScript code and send it to Distiller. The latter is my overwhelmingly favorite method of content generation.

Distilling usually takes you a few seconds per page.

There is also an *Acrobat Exchange* module that gives you limited editing abilities as well as methods of modifying security, adding notes, doing indexes, doing compression, or providing hot links. Exchange also can get you from Acrobat back to ordinary PostScript as well.

Acrobat is *by far* the best choice for tri-mode publishing.

Optimize for PostScript

Acrobat is really just an efficient "semicompiled" subset of the general purpose PostScript language. PostScript does its thing by describing what is to appear on your page as *procedures*. Properly written PostScript code is amazingly compact and fast. But poor programming or klutzy format converters can and will go out of their way to totally trash PostScript. Usually by making the code excessively bloated or exceptionally slow running.

For instance, a PostScript fill justified line should get done all at once by use of a single *awidthshow* command, rather than individually positioning each word or (worse yet) each character.

One sure clue that something is amiss in your PostScript generation is if your typical Acrobat page sizes exceed 11K. Or roughly one megabyte per hundred pages. In general, "PostScript friendly" apps such as *Pagemaker* or *Illustrator* write infinitely better PDF code than some ancient dino mainframe format converter.

An effective way to convert older documents is to load them into any modern word processor that is capable of directly generating Acrobat output.

The real biggie here is to go out of your way to...

Avoid Scanned Bitmaps

Just about the *worst* possible thing you can do is take some old document, scan it as full page bitmaps, and shove those into Acrobat. Each bitmap is *certain* to end up huge and ugly. Again, a decent mixed figures and text Acrobat page should typically be around 11K long. When done as a bitmap, it may take five or more megs.

Instead, use Adobe *Capture* to intelligently convert the incoming text into PostScript procs. Other alternatives are to trace over the bitmap with *Illustrator* or some other app that generates tightly efficient PostScript code.

Reserve bitmaps *only* for small images of exceptionally high informational value. Even then, consider reducing the resolution to the bare minimums needed for an attractive screen display and hard copy printability.

Thus, a little prep time and thought about being as "Acrobat friendly" as possible are usually well worth the extra front end effort.

Colorize

Pretty much everything viewed on screen these days is in full color. As a result, black text on a white background will usually be perceived as annoyingly harsh.

Perhaps even painfully so.

So, it is often a real good idea to *colorize* your previous files. One possible method is to provide *carefully chosen* colored text over a lightly colored background. Legibility and subtlety, of course, are primary concerns here.

It doesn't take much to make black on white text more attractive. Even a black bar or a rule or two converted to color tints might *dramatically* improve a page's onscreen viewability. As will lightly tinting any sidebars or coloring any text heads and subheads.

In many cases, *less is more*.

There's a sneaky colorizing trick I use that is reasonably fast and easy to add to older documents. What happens is this: Grays in original documents are converted to color on screen *but still print as original grays on a B/W printer*. A simple block of code prepended to the original PostScript sourcecode docs does your autoconversion.

The utilities and full details are in [COLORIZ2.PS](#).

The secret? The PostScript *setgray* operator is intercepted and replaced with an equivalent *setrgbcolor* operator. Any saturated hue could get used, and convenience operators, such as *lime*, *beige*, or *red* are easily added. As an example, "beige" might give you black, dark chocolate brown, milk chocolate brown, a rich orange, a beige, or white with increasing *setgray* values.

Your secret to printing the original gray is to use the standard video formula that $gray = 0.59 green + 0.30 red + 0.19 blue$ in setting all of your "magic" tints. Yeah, this math is rather hairy, but I've got all of that invisibly buried in an easy-to-use utility proc. Just say "beige" instead.

A demo of all the available "improved" colors appears as [COLORIZD2.PDF](#). An earlier tutorial is in [SPOTCOLR.PDF](#).

Link Up

One overwhelming advantage that a web document has over a printed one is its ability to *hot link*. Such clickable screen areas can send you to a different part of the same document, to a different doc, or go to any other *url* web

remote location anywhere in the world.

Links can serve many different purposes. They might simply get you from an index to a point on a page where your subject is covered. Or from a figure reference to the figure and back again. Or activate the table of contents. Or lead you into background tutorial material or advanced content. They might even play a sound clip, show a movie, or run an animated demo.

Note that links may be visible and fully obvious; they also can be made invisible and intuitive.

The secret to understanding Acrobat linking is Adobe Technical Note #5150, available through www.adobe.com. What you basically do is create an Acrobat *pdfmark*. A *pdfmark* is a block of "hidden" text in your PDF document that does stuff on screen but is usually ignored by printers.

A linking demo between PDF and HTML can be found as [LINKPDF1.HTML](#).

The simplest way to add a few links to an Acrobat file is to use *Exchange*, manually setting the hot link boundaries and the destination. When automated link placement and many files are needed, special procs and scripts can interact with your sourcecode to simplify making links.

Links that are made from inside your sourcecode can automatically reposition themselves on any later edit or update. They also will not have to be reentered manually anytime a new PDF file is generated.

One example of a semi-automated, sourcecode based link creator appears in [URLINDOC.PS](#). The enhanced procs I personally use are found in [MUSE115.PSL](#).

It is very important to both *test* and then *maintain* your hot links. Especially those external ones. Each link must behave in the expected manner and must in fact go where it says it is going to. Every time.

Internal broken links are easily spotted by using your website error log files. External links require retesting. More on interpreting error logs in general in [MUSE116.PDF](#), in [BANNYEAR.PDF](#) and in [WEBSITAN.PS](#).

Provide Search Options

A *search engine* is a new method of letting you find any word or phrase anywhere in a document or in an entire library. Intelligent engines can also do neat tricks such as *relevance ranking*, *word stemming*, or finding words that are near each other in context.

A search engine typically works by making a "backwards" alphabetical list of *all* words in every library document. Tabulating where they occur. You have a choice of listing *all* the words, excluding only the useless ones, or of more tightly targeting to selected formal *keywords*.

Search engines often take up a lot of space, but they certainly do make up for it in user convenience. A search engine might add *thirty percent* to a site's text storage needs. Perhaps an extra 3K for each 11K text page.

If you are *only* using Acrobat files, Adobe has a *Catalog* module that makes use of a *Verity* search engine. This will automatically scan through your entire Acrobat file library and generate a useful set of searchable indexes for you.

Results are ranked by how often the words appear how near to each other in each file.

Alternately, if you are on the web, you'll probably want to provide a master search engine that works equally well with PDF files as well as HTML. Adobe newly offers a *PDF IF*

MENTIONED RESOURCES

Adobe Acrobat
1585 Charleston Rd
Mountain View CA 94039
(800) 833-6687

Aladdin/Ghostscript
Box 60264
Palo Alto CA 94306
(415) 322-0103

MicroSoft
One Microsoft Way
Redmond WA 98052
(800) 877-1900

Netscape Gold
501 E Middlefield Rd
Mountain View CA 94043
(415) 528-3777

Synergetics
Box 809
Thatcher AZ 85552
(520) 428-4073

Verity Systems
1550 Plymouth St
Mountain View CA 94043
(415) 960-7600

filter plug in for the web based *Microsoft Index Server*. Other server "Acrobat recognition" options can be expected.

Regardless of the route you choose, providing a decent search engine is a web or CD ROM must.

More details on the inner workings of Acrobat Catalog are in [ACATDATA.PS](#), [CATWORDS.PS](#), and [CATOOLS1.PS](#).

Optimize Web Viewing

There are several steps you should take to give your web or CD ROM users their fastest possible access.

A special web option called *Byte Range Retrieval* will let your longer PDF files load much faster. On a hundred page document, only your first page text has to load to get a viewable user display. That page graphics and the rest of the pages then invisibly load in the background. Similarly, if your user decides they want to see page 53, they can get to that page directly without having to wait for pages 2 through 52 to load.

A single byteserved PDF file loads ridiculously faster and far cleaner than a graphics loaded HTML multi-file group. I am utterly amazed that HTML still sees any use at all. Thus, it is usually best to use as *much* PDF and as *little* HTML as possible on your website.

Two steps are needed to provide byte range retrieval: First, you'll have to load your PDF files into the Acrobat Exchange module, select *save as* with your *optimize* box clicked, and resave all pages under new filenames. Secondly, your ISP has to turn on the byte range retrieval feature of their server. If it is not yet already activated.

Byte range retrieval is a standard feature on all newer servers. This can be added to most older ones by using a simple and short CGI script.

You should also help your user get their full screen displays. Provide a clickable box so they can get an Acrobat reader if they do not already have one. *Strongly* request that they install their reader so that appropriate plugins automatically are installed into their Netscape browsing software. The rule here is to install *Netscape first*, *Acrobat second*. Then verify installation.

Also recommend that your user go into their reader *full screen preferences* options to set "escape exits", "advance on any click" "loop after last page" "mouse cursor hidden after delay" and "wipe down". But have them turn their "auto advance" *off*. Be sure to tell them that a *control-shift-L* followed by a *control-K* will give them a full screen display. And that their arrow, home, pageup, and pagedown keys

all will work in the way your user expects.

It is probably *not* a good idea to force a full screen on everybody; full screen could create problems on older browsers or if display preferences are not chosen properly. Finally, recommend a "roller style" enhanced mouse to further improve their onscreen nav.

in general, though, you can use other *pdfmark* operators to decide how and where a file is to open. This lets you pick the opening page and its magnification. Details are once again in the *pdfmark* tech note.

More byte range retrieval details in [BYTERNGE.HTML](#) and especially at www.adobe.com.

Recognize Differences

Ink on paper, the web, and CD ROM all do have their individual strengths and weaknesses. The "touchy feely" aspects of "curling up with a good book" might remain for quite some time. The perceived value of a book is also higher. Far fewer people steal from bookstores than rip off your website. And CD's are best at giving you permanent archival stashes and reference storage.

Books are usually intense text and portrait oriented. Screens are often larger concise text and are best landscape oriented. Books are arranged by *chapters*. Screens are best arranged by *thoughts* or *concepts*. Books often use formal sentences; phrases and fragments sometimes are more appropriate on screens.

I guess you'll have to decide what your *primary* media is going to be and then optimize for it as best you can. It takes a tricky and delicate balance to get content that looks equally acceptable in all three media.

For More Help

Adobe's web site at www.adobe.com is particularly useful to pick up technical details on useful tools for tri mode publishing. They also offer several developer's programs and software development kits.

Fundamentals of Acrobat appear in the *Pewter* book, otherwise known as the *Acrobat Reference Manual*. Good starting points for PostScript in general are the *Red* book, the *PostScript Reference Manual* and the *Blue* book, the *PostScript Cookbook and Tutorial*. I have these in stock per my nearby *Synergetics* ad. Along with my own PostScript products, and the *Whole Works* package for anyone who really wants to jump into PostScript in a big way.

My *Guru's Lair* website at www.tinaja.com does have lots of tri-mode info on it. Especially check out the *Acrobat*, *PostScript*, *Webmastering* and *Book-on-demand* library shelves. Lots of detailed tri-mode examples also appear here. I also have full tri-mode publishing consulting services available. Let's hear from you. ♦

Microcomputer pioneer and guru Don Lancaster is the author of 35 books and countless articles. Don maintains a US technical helpline you'll find at (520) 428-4073, besides offering all his own books, reprints and consulting services.

Don has a free new catalog crammed full of his latest insider secrets waiting for you. Your best calling times are 8-5 weekdays, Mountain Standard Time.

Don is also the webmaster of www.tinaja.com You can also reach Don at Synergetics, Box 809, Thatcher, AZ 85552. Or you can use email via don@tinaja.com

new from
DON LANCASTER

**INCREDIBLE
SECRET MONEY
MACHINE II**

Start your own technical venture! Autographed copies of the Guru's great underground classic. Includes helpline and BBS support. **\$18.50**

**BOOK-ON-DEMAND
PUBLISHING KIT**

Ongoing details on Book-on-demand publishing, Don's brand new method for producing books only when and as they are ordered. **\$39.50**

**THE CASE
AGAINST
PATENTS**

For most Midnight Engineers, patents end up a total waste of time and money. Here are tested and proven real-world alternatives. **\$28.50**

**CMOS AND TTL
COOKBOOKS**

Millions of copies in print worldwide. **THE** two books for digital integrated circuit fundamentals. About as hands-on as you can get. **\$28.50** each.

**ACTIVE FILTER
COOKBOOK**

Sixteenth (!) printing of Don's bible on all analog op-amp lowpass, bandpass, and highpass active filter circuits. Instant circuit designs. **\$28.50**

**THE BLATANT
OPPORTUNIST**

The reprints from all Don's Midnight Engineering columns. Includes full index and the Synergetics resource directory. Unique material. **\$24.50**

**FREE
SAMPLES**

Well, almost. Please join us on www.tinaja.com For all the Guru's goodies. Includes tech library, consultants net, product info, forums, bunches of site links. email: synergetics@tinaja.com

SYNERGETICS

**Box 809-ME - Thatcher, AZ 85552
(520) 428-4073**

FREE US VOICE HELP

CHECK/VISA/MC

your one-stop source for **POSTSCRIPT STUFF**

POSTSCRIPT REFERENCE MANUAL

The **RED** book. Your main encyclopedia for all PostScript commands. Level II info, EPS files, plus document structuring conventions. **\$34.50**

POSTSCRIPT TUTORIAL AND COOKBOOK

The **BLUE** book. Adobe's classic introductory text on PostScript basics. Thorough examples and complete step-by-step instructions. **\$22.50**

TYPE I FONT FORMAT

The **BLACK** book. Full details on structure and coding of Adobe Type I fonts. Create your own fonts. Shows eexec and font coding. **\$16.50**

POSTSCRIPT BY EXAMPLE

The **MAROON** book. By Henry McGilton and Mary Campoine. Ideal self study. Beginner to intermediate level. 620 pages. **\$29.50**

ACROBAT REFERENCE

The new **PEWTER** book. The secret insider's guide to Adobe Acrobat, their new "paperless" and "fontless" publishing scheme. **\$24.50**

POSTSCRIPT SECRETS

Don Lancaster's book/disk combo crammed full of free fonts, utilities, resources, more. For most any PS printer. Mac or PC format. **\$29.50**

THE WHOLE WORKS

One each of everything important by all major PostScript authors. Includes all of the above and bunches more at a substantial savings. Books, software, video, and helpline. **\$379.50**

SYNERGETICS

**Box 809-ME • Thatcher, AZ 85552
(520) 428-4073**

FREE US VOICE HELP

CHECK/VISA/MC