The New Paper Trail:

Alternative Information Distribution

With PC AI Magazine's decision to go digital, a brief survey of the technology of paper and publishing seems appropriate.

The Technology of Publishing through the Ages

The history of publishing spans approximately six thousand years. The Sumerians first etched cuneiform pictographs into clay tablets. Shortly thereafter, the Egyptians invented papyrus, and the "paper trail" began.

The next major stop in the paper trail occurred about two thousand years ago in China, with the actual invention of paper. Until that time, all of papers' predecessors such as papyrus required manual weaving or gluing into pages. But then in China, an inventor named Ts'ai Lun discovered a method for automatically generating paper out of plant fiber. Over the centuries, paper making spread to Japan, the Arab world, and on into Europe. Paper mills sprung up both in Europe and the New World.

Publishing content onto paper took a great stride forward in the mid 15th Century with Johann Gutenberg's invention of a printing press with moveable type. With the ease of publishing, the demand for paper became evermore "mass market." And in the late 19th Century, papermaking turned to wood pulp for mass production. Previously, paper production had relied on everything from reeds to rags.

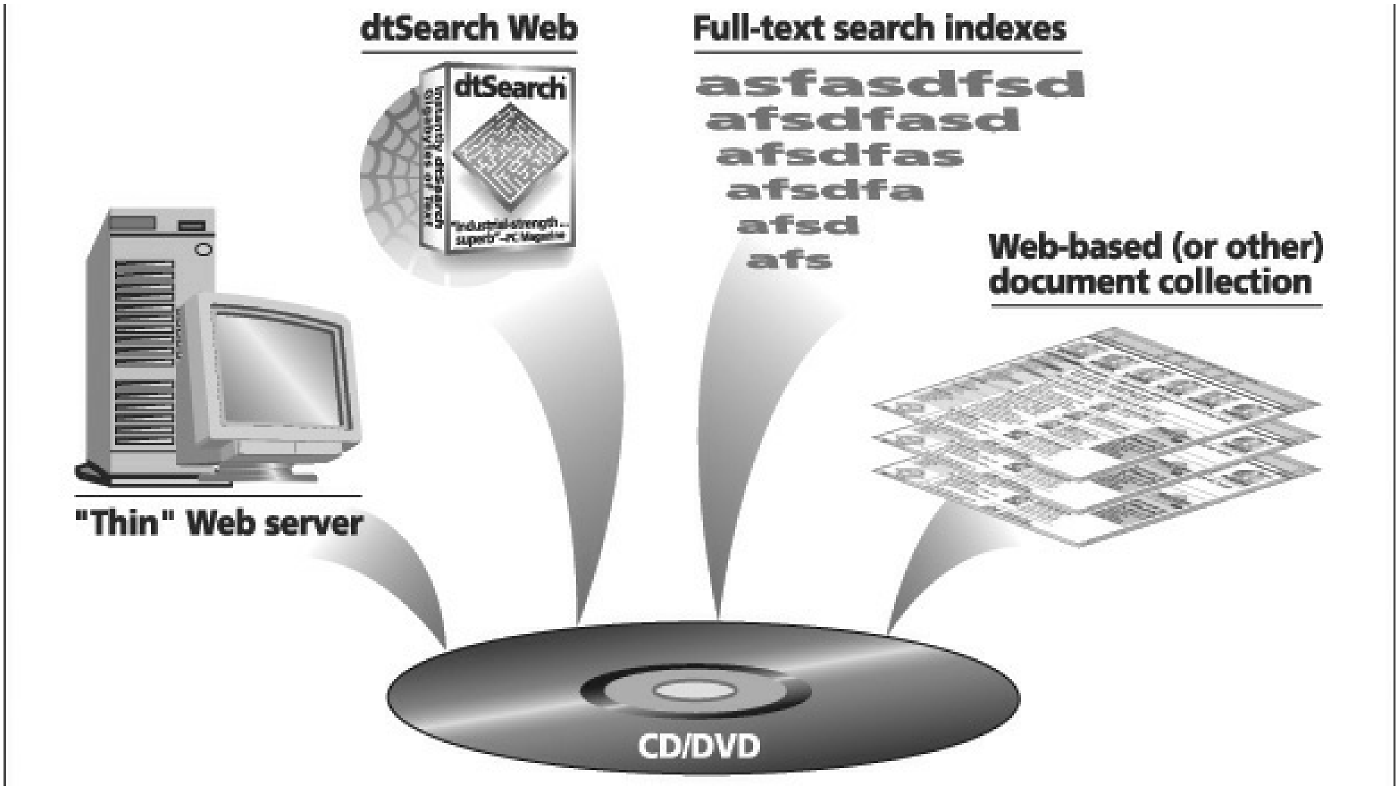
In the last few decades of the 20th century, publishing took another major step. With computers, came the option for paperless, electronic publishing. CDs, the Internet, America Online and the like have become the new publishing media.

The New "Paper" Choices

The World Wide Web, the world's most universal form of electronic

dtSearch Publish, for example, uses a setup wizard that automatically generates a master CD/DVD running the Apache Web server. The wizard also installs the document collection for CD access in HTML, XML or PDF. The end product converts other file formats—word processor, database, spreadsheet, etc.—to HTML for browser display.... The CD/DVD displays retrieved documents with multiple browser-based hit and file navigation options, including display of highlighted hits and embedded HTML and PDF images. HTML files on the CD/DVD can even contain links to other Web sites.

Reprinted with permission of PC AI Magazine, January/February 2002 V. 16 #1
For more information about PC AI Magazine, visit www.pcai.com/pcai



Sample components of a CD/DVD-based Web server.

publishing, is accessible through any terminal with an Internet connection. Standard Web browsers read text in the popular Web currency used by the vast majority of Web sites: HTML and XML. For a third Web-based standard, the PDF format discussed below, a free Adobe Reader plug-in is available at *www.adobe.com*.

While the Internet remains the universal format for electronic publishing, it still has its limitations relative to distributing information on CD/DVD. First, the user has to actively connect to the Internet to access the information. Second, restricting or controlling distribution of selected information over the Internet can be cumbersome.

An alternative approach provides immediate accessibility and the data-limiting possibilities of CD/DVD, while taking advantage of the universal browser-based Web medium. Under this approach, a CD/DVD operates as an actual mini-Web site, executing only off the CD/DVD, instead of the generally accessible Internet. To turn a CD/DVD into a Web site requires a Web server “thin” enough to run off the disk.

A text search and retrieval program like dtSearch that supports these PDF files can index and search them. After a search, the program displays the pure scanned image. Because the OCR’ed text is hidden inside the PDF, the search program highlights hits directly on the image.

The Apache Web server is just such a server. It runs directly from a CD, offering a virtual Web site directly from the end-user’s CD drive. Like any Web site, reading the contents requires a Web browser to read HTML and XML, along with the Adobe Reader for PDF. Accessing the CD requires only inserting the disk. And the setup has the additional benefit of installing absolutely nothing on the user’s hard drive.

Current tools make such a CD/DVD Web server easy to install. dtSearch Publish, for example, uses a setup wizard that automatically generates a master CD/DVD running the Apache Web server. The wizard also installs the document collection for CD access in HTML, XML or PDF. The end product converts other file formats—word processor, database, spreadsheet, etc.—to HTML for browser display.

Finally, dtSearch Web provides full-text searching of the CD/DVD contents. When the user does a search, the user’s browser sends a request to the CD/DVD similar to any other web server search request. The CD/DVD displays retrieved documents with multiple browser-based hit and file navigation options, including display of highlighted hits and embedded

OCR Products

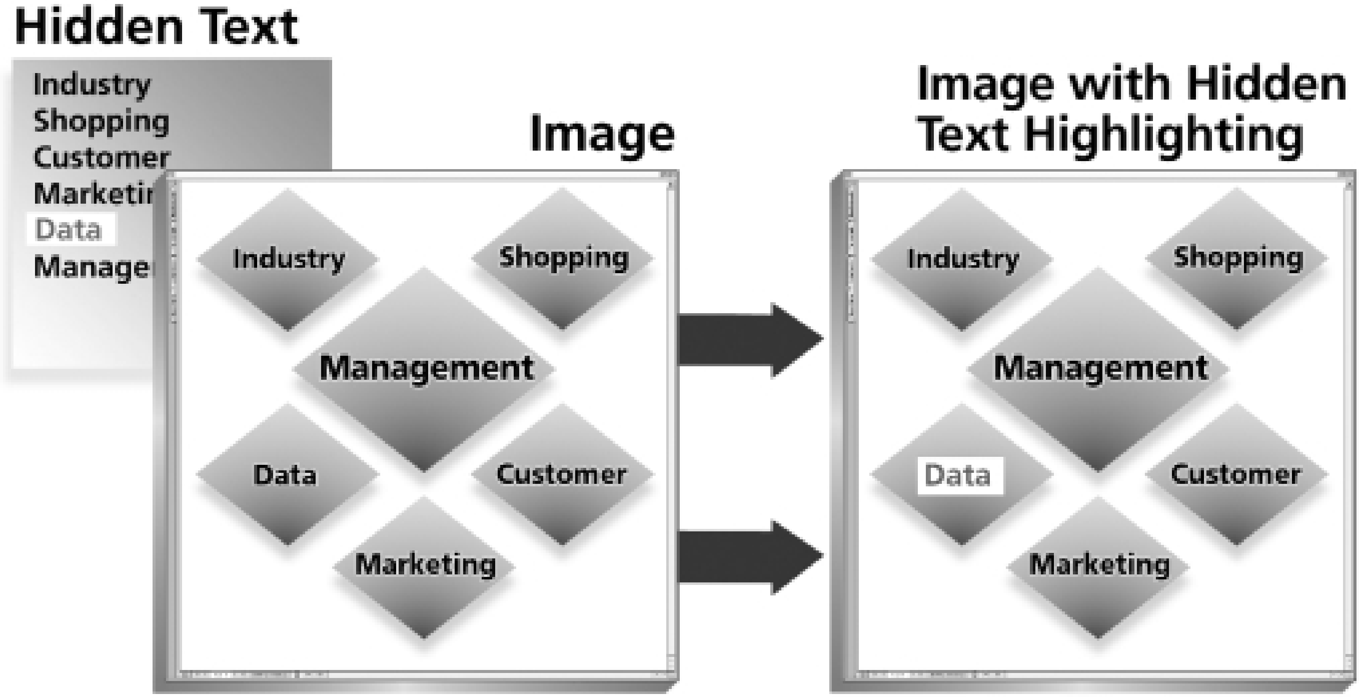
Products that can generate PDF files from scanned images include:

Abbyy Fine Reader — www.abbyy.com

Adobe Capture — www.adobe.com

Caere OmniPage Pro 11 — www.scansoft.com/products/omnipage/pro

ScanSoft’s TextBridge Pro Business Edition — www.scansoft.com/products/tbpmillbe



Because the actual OCR'ed text is "hidden," a text search and retrieval program like dtSearch that supports this format will appear to highlight hits directly on the image.

HTML and PDF images. HTML files on the CD/DVD can even contain links to other Web sites.

Today's paper trail now leads to Web sites on CD, just as easily as on the Internet. With numerous electronic publishing options available, making virtual paper stand up to the electronic publishing task is the challenge. This article discusses a few methods for accomplishing this, including: "hidden text on images" PDF for WYSWYG (defined below) paper display; Unicode to handle English and other international language text; and fuzzy and other search techniques to optimize full-text searching.

PDF With Hidden Text

The traditional method for turning paper into searchable text is to scan the document into a TIF image file. Optical Character Recognition (OCR) software converts the TIF contents into machine readable (and searchable) text. The resulting output is usually a text or word processor document.

The deficiency in the typical TIF / OCR approach is that it separates the images from the machine-readable text. For example, using a search program to index and search the text retrieves the resulting text file, including highlighting matching search terms. To visually review the original document, such as to read a handwritten note or to see a picture on the original, requires an additional step.

> *The dtSearch-type of fuzzy searching [is] fully flexible and adjustable at the time of the search. Take the word* traveler. *With a fuzziness level of 0, the program picks up an exact spelling of* traveler. *With a fuzziness level of 1, the program looks for exact spelling, as well as a single deviation in letters. For example, it picks up* traveler *as well as* traveller. *With a fuzziness level of 2, the program also looks for additional deviations of letters, picking up not only* traveler, *but also* travellller *as well as* trameller.

Taking that step is not quite as difficult as it sounds, since software can correlate the text documents with the relevant images. So if a text file were TRAVELS.TXT and its relevant TIF images were TRAVELS001.TIF, TRAVELS002.TIF and TRAVELS003.TIF, selecting an image button while viewing the text document could retrieve the sequentially relevant TIF files.

But while the text files contain highlighted hits, the TIF files do not. So while immediately jumping to *Grand Canyon w/25 river rafting* in the text files is easy, finding such a reference in the TIF files requires manual location. The solution is to find a document format that combines both text and images together.

HTML is one option. However, HTML stores the text and the images rather loosely. With HTML, the viewer does not see a pure WYSWYG (What-You-See-Is-What-You-Get) format. In contrast, PDF combines text and images into a single, compressed WYSWYG format.

The Adobe PDF file format offers two methods for combining images and text in a single file. First, the "image with hidden text" format stores the complete original TIF images of a scanned document, along with the text obtained through OCR. The text is hidden in the sense that opening the PDF file displays only the scanned image, not the underlying OCR'ed text.

Another option for combining scanned images and text in a single PDF file uses small images for the parts of each scanned page that do not appear to be text. For example, a picture or a signature would become a small image embedded in the page, while the rest of the page becomes text. This format produces much smaller files than the first alternative, because each page only stores a few small images, instead of a complete image of the whole page.

A text search and retrieval program like dtSearch that supports these PDF files can index and search them. After a search, the program displays the pure scanned image. Because the OCR'ed text is hidden inside the PDF, the search program highlights hits directly on the image.

Unicode

The Unicode standard allows everything from the operating system to the file type to the text search and retrieval program to seamlessly switch from English to Russian to Hebrew in the same document. Unicode enables the encoding of text in any language in a consistent manner. Instead of the 255 characters allowed by the ANSI character set, the Unicode character set handles over 65,000 characters. Detailed information on the Unicode specification is available at www.unicode.org.

A Unicode file typically contains a tag at the beginning of a document indicating which Unicode encoding standard to expect. For example, an HTML file might contain the META tag: <meta http-equiv="Content-Type" content="text/html; charset=utf-8">. The META tag would allow a Unicode-aware text search and retrieval program to properly treat the text in the document.

Some OCR programs can scan as well as OCR into a non-English language. In that case, the OCR program will usually output the text into a file that includes the relevant Unicode information. PDF files, like HTML files, can support Unicode text, particularly if the PDF is OCR'ed with a Unicode-supporting "OCR into PDF" capability.

A PDF file that was not created using Unicode-supporting text generation techniques, however, is a different matter. Unlike other document formats, which usually contain text in some form, PDF files are essentially drawing instructions that supply information necessary to print a document on a printer or to draw it on the screen. Many PDF files contain character encoding information in addition to the drawing instructions, enabling easy text extraction.

But in some PDF files, only the drawing instructions are clearly present. As a result, it is not possible for a program such as a text retrieval program to extract the text. In fact, the only way to obtain the text is often to OCR the files, as if they were pure images, into a new text-based PDF file, or other file type.

Fuzzy Searching

The sidebar lists some OCR programs that output OCR'ed text and accompanying images to PDF files. However, it is not uncommon for an OCR program to make the occasional typographical mistake. This is especially true with documents that are faxed or are otherwise in less-than-perfect condition.

In the real world, a text search and retrieval program must sift through OCR mistakes. The goal is to find the word *traveler*, for example, not simply if it is correctly OCR'ed as *traveller*, but also if it is incorrectly OCR'ed as *traviller* or even *tramaller*. To sift through this level of mistakes, a text search and retrieval program needs a complete fuzzy search option.

Some text search and retrieval programs use the word fuzzy to express other types of searching, such as stemming, which finds different endings on the same route word — *travel*, *travels*, *travelers*, *traveling*, etc., but not *tramelling*. Another set of programs decides in advance what the likely OCR errors might be, and hardwires that result into the index. In contrast, the fuzziness described here depicts the dtSearch-type of fuzzy searching: fully flexible and adjustable at the time of the search.

Take the word *traveler*. With a fuzziness level of 0, the program picks up an exact spelling of *traveler*. With a fuzziness level of 1, the program looks for exact spelling, as well as a single deviation in letters. For example, it picks up *traveler* as well as *traveller*. With a fuzziness level of 2, the program also looks for additional deviations of letters, finding *traveler*, but also *travelller* as well as *trameller*.

A little fuzzy searching compensates for many OCR errors. For maximum precision searching, fuzzy searching also works with other types of searching. For example, fuzzy searching works with stemming searching, phonic searching, natural language searching, proximity and boolean searching, etc. With proximity and fuzzy searching, a search for *traveler w/7 sudan* would still find the OCR'ed text: *we tramelled across the Sudan*.

However, even if the PDF or other file contains the proper Unicode information, in some cases even a fuzzy search will not find the desired information. For example, in Arabic, the surrounding context for a word (the Arabic equivalent of my, your, the, a, masculine/feminine, etc.) can be expressed as characters added in front of or behind the word. The Arabic equivalent of *the apple* or *my apple* is not two words but different prefixes or suffixes added to *apple*.

To search for text in Arabic, adding a wildcard character in front and back of the word picks up most of the variants. So, for example, instead of searching for *apple* with optional fuzziness on, a better search might be **appl**. Needless to say, this wildcard approach itself has limitations, such as overbroad word retrieval. But the world still awaits a Rosetta Stone for reliably searching electronically published document collections in any language.

For more information about dtSearch, please call (800) IT-FINDS or visit *www.dtsearch.com*

Reprinted with permission of PC AI Magazine, January/February 2002 V. 16 #1
For more information about PC AI Magazine, visit www.pcai.com/pcai